

ELECTRICAL DEMAND ANALYSIS SOFTWARE TOOL SUITE AND
AUTOMATIC REPORT GENERATION FOR ENERGY AUDITS

A Thesis

by

FRANCO JAVIER MORELLI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Bryan P. Rasmussen
Committee Members,	David E. Claridge
	Charles H. Culp
Head of Department,	Andreas A. Polycarpou

May 2015

Major Subject: Mechanical Engineering

Copyright 2015 Franco Javier Morelli

ABSTRACT

The American Society of Heating, Refrigeration and Air Conditioning Engineers (ASHRAE) defines an energy audit through a multi-tiered stratagem characterized by the level of in-depth analysis. The Level 1, or walkthrough survey is highlighted by low to no cost energy efficiency evaluations and a list of improvement measures that warrant further inquiry. Through the Industrial Assessment Center (IAC) at Texas A&M University, the Department of Energy's, Advanced Manufacturing Office maintains collaboration with academic entities to further the goal of reducing industrial and manufacturing energy consumption. As a result, the IAC at Texas A&M University performs ASHRAE Level 1 Energy Audits for manufacturing plants across gulf coast states. The IAC at Texas A&M University seeks to develop a series of electrical demand analysis and report generation software tools to optimize and enhance the electrical investigation inherent with establishing efficient industrial resource (electricity, water, natural gas) usage. Typically, such analysis are done through utility bill information, quantifying usage and capital charge characteristics, as well as usage trends over the course of the billing period. By establishing electrical analysis through the use of 15-minute or 30-minute demand data sets, available to industrial and manufacturing clients augmented with Interval Data Recorder (IDR) meters, the Industrial Assessment Center at Texas A&M has developed a suite of electrical analysis tools designed to increase analysis fidelity, identify pre-visit Energy Conservation Measures (ECM), establish unknown variables helpful in diagnosing ECMs, size systems design to optimize

electrical usage, create a simple, user friendly interface and increase ECM implementation. While the conclusions and results for the following work and tools will not be known for some time, preliminary efforts have shown that the tools are effective in interpreting and diagnosing aberrant electrical usage. In particular, one instance in usage of the demand visualization tool diagnosed an issue where a facility was being charged double the amount of their typical demand. Supporting data, along with key IAC visits will be required to determine if the following tools are effective in increasing IAC implementation rates.

TABLE OF CONTENTS

	Page
1. INTRODUCTION AND MOTIVATION	1
Industrial Energy Use Today.....	1
The Future of Industrial Energy Use.....	8
Energy Efficiency and the IAC Program	13
TAMUIAC ECM Implementation	17
Motivations for Change.....	19
2. PREVIOUS WORK	24
Behavior and Psychology.....	24
Energy Dashboards	28
TXU My Energy Dashboard	28
Energy AI	31
Seldera.....	32
Energy PlanIT	34
3. SOFTWARE TOOLS	36
Demand Visualization.....	36
Daily.....	42
Weekday.....	44
Annual	45
Demand Aberration.....	50
Demand Scheduling	55
Weather Disaggregation.....	58
Photovoltaic (PV) Analysis.....	62
Solar Dynamics	63
Solar Radiation on Earth	70
From Solar Power to Electricity.....	70
Power Factor Correction	82
Ratcheting.....	85
4. AUTOMATIC REPORT GENERATOR	89
Introduction	89
Mechanics of Automatic Report Generation.....	89
Report Templates	91
5. CASE STUDY 1	93

Introduction	93
Plant Background	93
Plant Energy Consumption.....	95
Energy Cost Analysis	98
Load Factor Analysis	99
Summary of Plant Statistics	103
Demand Visualization.....	104
Yearly	105
Predicting Operating Hours.....	109
Monthly	112
Daily	117
Weekday.....	120
Weather Disaggregation.....	133
Demand Scheduling	138
Photovoltaic Analysis.....	140
 6. CASE STUDY 2	 149
Introduction.....	149
Demand Visualization.....	149
Demand Aberration.....	155
Photovoltaic Analysis.....	157
 7. CONCLUSION	 166
 REFERENCES.....	 169
 APPENDIX A: ASSESSMENT RECOMMENDATION WORKSHEETS	 171
Retrofit Exit Signs ARC 2.7142.3	171
Install Skylights ARC 2.7145.3.....	172
Lighting Retrofit ARC 2.7142.2.....	174
Turn Off Lights ARC 2.7124.2	176
Repair Compressed Air Leaks ARC 2.4236.2	178
Reduce Compressed Air Pressure ARC 2.4231.2	180
Engineered Nozzles ARC 2.4322.2	182
High Efficiency Motors ARC 2.4322.2.....	184
Energy Efficient Belts ARC 2.4133.1	186
Synthetic Lubricants ARC 2.4314.1	188
Power Factor Correction ARC 2.3212.3	190
Tune Up Boiler (If Boiler NG Usage Is Known) ARC 2.1233.2	192
Tune Up Boiler (If Boiler NG Usage Is Unknown) ARC 2.1233.2.....	194
Late Fees ARC 2.7124.4.....	196
 APPENDIX B: LIGHTING ASSESSMENT RECOMMENDATIONS	 197

Replace Incandescent Lights in Exit Signs with L.E.D. Retrofit (1 Bulb Type)	197
Replace Incandescent Lights in Exit Signs with L.E.D. Retrofit (Multiple Bulb Type).....	207
Install Skylights and Control Lighting with Photosensors	218
(Single Zone, Single Lighting Type).....	218
Install Skylights (Multiple Zones, Multiple Lighting Types)	230
Replace <value1> Illumination with <value2> Lights (Single Lighting Type).....	239
Replace <value1> Illumination with <value2> Lights (Multiple Lighting Types)....	248
Turn Off Lights and Install Occupancy Sensors	258
 APPENDIX C: COMPRESSED AIR ASSESSMENT RECOMMENDATIONS	 268
Repair Compressed Air Leaks.....	268
Reduce Compressed Air Pressure	275
Replace Standard Blow Off Nozzles with Engineered Nozzles.....	284
 APPENDIX D: MOTORS ASSESSMENT RECOMMENDATIONS	 292
Replace Standard Efficiency Motors with High Efficiency Motors	292
Replace V-belts with Notched or Synchronous Belt Drives	302
Utilize Synthetic Motor Lubricants.....	311
 APPENDIX E: POWER FACTOR CORRECTION ASSESSMENT RECOMMENDATIONS	 320
Install Capacitor Bank for Power Factor Correction.....	320
 APPENDIX F: BOILER ASSESSMENT RECOMMENDATIONS	 327
Perform a Boiler Tune Up to Improve Boiler Efficiency (Known NG Usage)	327
Perform a Boiler Tune Up to Improve Boiler Efficiency (Unknown NG Usage)	332
 APPENDIX G: LATE FEES ASSESSMENT RECOMMENDATIONS	 338
Avoid Late Fee Penalties (Single Utility)	338
Avoid Late Fee Penalties (Multiple Utility).....	341
 APPENDIX H: MATLAB TOOLS CODE	 344
Yearly Demand Visualization Tool.....	344
Monthly Demand Visualization Tool.....	361
Day of Week Demand Visualization Tool	384
Daily of Week Demand Visualization Tool	407
Demand Aberration Tool	425
Demand Scheduling Tool.....	456

Weather Disaggregation Tool	467
Photovoltaic Analysis Tool	479
Power Factor Correction Tool.....	504
Ratcheting Tool.....	516

LIST OF FIGURES

	Page
Figure 1.1: End use consumption for the four major consumers of electricity in the United States [1].....	2
Figure 1.2: Energy consumption for the four major consumers of electricity in the United States, from 1950 to 2013 [2].....	3
Figure 1.3: Major sources of energy consumption (in quadrillion BTU’s) for the industrial sector in the United States, from 1950 to 2013 [2].....	4
Figure 1.4: Uses for petroleum in industrial applications in the United States, from 1949 to 2012 [4].....	4
Figure 1.5: Usage of renewable sources of energy (in quadrillion BTU’s) in both industrial and commercial applications, from 1949 to 2012 [8].....	7
Figure 1.6: Worldwide use of energy for both industrial and other end use consumers, from 2005 to 2040 [11].....	8
Figure 1.7: Worldwide energy use for both OECD (Organization for Economic Cooperation and Development) and Non-OECD countries, from 2010 to 2040 [11].....	9
Figure 1.8: Energy consumption in the United States by energy type for 2010 and 2040 [11].....	10
Figure 1.9: Projection of fuel prices for the industrial sector, from 2010 to 2040 [12].....	11
Figure 1.10: Projection of fuel consumption for the industrial sector, from 2010 to 2040 [12].....	12
Figure 1.11: Timeliness trends for TAMUIAC ECM to repair compressed air leaks.....	18
Figure 1.12: Implementation of TAMUIAC ECM to repair compressed air leaks.....	18
Figure 2.1: “MyEnergy Dashboard” user interface available to select TXU customers. [2].....	29
Figure 3.1: Total demand curve. February 2012.....	37
Figure 3.2: Reduced total demand curve. February 2012.....	38

Figure 3.3: Daily demand curve. February 29, 2012.....	42
Figure 3.4: Daily demand curve. February 18, 2012.....	43
Figure 3.5: Weekly demand usage. Sundays, 2012.....	45
Figure 3.6: Yearly electrical demand. 2012.....	46
Figure 3.7: Yearly electrical consumption. 2012.....	47
Figure 3.8: Yearly electrical demand curve. 2012.....	49
Figure 3.9: 2-Norm demand space. 2012.....	52
Figure 3.10: Comparison of the demand profile for July 22, 2012 and the average demand profile seen on Sundays.....	53
Figure 3.11: 1-Norm demand space. 2012.....	54
Figure 3.12: Comparison of the demand profile for September 04, 2012 and the average demand profile seen on Sundays.	55
Figure 3.13: Demand savings associated with implementing an electrical demand scheduling scheme. 2012.	57
Figure 3.14: Disaggregated demand curves P_i and P_j . 2012.....	61
Figure 3.15: Difference, ΔD , of the demand curves P_i and P_j . 2012.....	61
Figure 3.16: Scheme of information flow to predict PV system size.....	63
Figure 3.17: Change of declination angle through solar procession. [4].....	64
Figure 3.18: Relation of earths equatorial plane to Sun. [6].....	64
Figure 3.19: Declination angle as a function of the months of the year. [6]	65
Figure 3.20: Relation of objects on Earths surface to Sun. [6].....	67
Figure 3.21: Relationship between coordinate systems. [6].....	68
Figure 3.22: Capital savings curve associated with installing a PV system of different sizes.	71
Figure 3.23: Cost associated with installing a PV system of different sizes.	73

Figure 3.24: Payback period associated with installing a PV system of different sizes.....	74
Figure 3.25: Pie chart of percent time facility electrical demand needs are met and unmet by PV system.....	75
Figure 3.26: Facility demand met by solar power on a monthly basis.	76
Figure 3.27: Percent of demand met by solar power on a day per week basis.	77
Figure 3.28: Comparison of demand produced through PV and facility demand for every day of the year. 2012.	78
Figure 3.29: Percent time facility electrical consumption needs are met and unmet by PV system.....	79
Figure 3.30: Facility consumption met by solar power on a monthly basis.	80
Figure 3.31: Percent of consumption met by solar power on a day per week basis.	81
Figure 3.32: Consumption produced through PV and facility consumption for every day of the year. 2012.	82
Figure 3.33: Power triangle. [20].....	83
Figure 3.34: Depiction of power factor.	85
Figure 3.35: Ratcheting penalty for 1999.....	87
Figure 4. 1: Input/Output scheme of Automatic Report Generator.	90
Figure 5. 1: Consumption and consumption cost trend for facility.	97
Figure 5.2: Demand and demand cost trend for facility.	98
Figure 5.3: Trending of PLF and ELF.....	102
Figure 5.4: Facility monthly demand peaks for 2007.....	106
Figure 5.5: Facility monthly demand peaks for 2008.....	106
Figure 5.6: Facility monthly demand peaks for 2009.....	106
Figure 5.7: Facility monthly demand peaks for 2010.....	106
Figure 5.8: Facility monthly demand peaks for 2011.....	107

Figure 5.9: Facility monthly demand peaks for 2012.....	107
Figure 5.10: Facility monthly electrical consumption for 2007.	107
Figure 5.11: Facility monthly electrical consumption for 2008.	107
Figure 5.12: Facility monthly electrical consumption for 2009.	108
Figure 5.13: Facility monthly electrical consumption for 2010.	108
Figure 5.14: Facility monthly electrical consumption for 2011.	108
Figure 5.15: Facility monthly electrical consumption for 2012.	108
Figure 5.16: Yearly facility demand trend with model limits.	110
Figure 5.17: Zoom from figure 5.16 to show model inconsistency 1.....	111
Figure 5.18: Zoom from figure 5.16 accounting for model inconsistency 2.....	112
Figure 5.19: Aggregated demand set for March 2008.....	113
Figure 5.20: Reduced demand set for March 2008.....	114
Figure 5.21: Aggregated demand profile for August 2012.....	116
Figure 5.22: Reduced demand profile for August 2012.	116
Figure 5.23: Daily demand curve for May 12, 2008.	118
Figure 5.24: Daily demand curve for August 06, 2008.	119
Figure 5.25: Reduced demand information for Thursdays throughout 2007.	121
Figure 5.26: Reduced demand information for Wednesday throughout 2008.	122
Figure 5.27: Reduced demand information for Wednesday throughout 2010.	123
Figure 5.28: Reduced demand information for Thursday throughout 2011.....	123
Figure 5.29: 1-norm demand space for Sundays 2007.	125
Figure 5.30: 2-norm demand space for Sundays 2007.	126
Figure 5.31: Comparison of demand curve for July 15, 2007 and average demand curve for all Sundays.....	128

Figure 5.32: Comparison of demand curve for January 21, 2007 and average demand curve for all Sundays.	129
Figure 5.33: Visualization of 1-norm demand space for Mondays 2008.	130
Figure 5.34: Comparison of demand profile for March 31, 2008 and average demand profile for all Mondays throughout 2008.	131
Figure 5.35: Daily demand profile for March 31, 2008.	132
Figure 5.36: Weather separated, average demand profiles for 2010.	134
Figure 5.37: Difference in demand profiles for 2010.	135
Figure 5.38: Weather separated, average demand profiles for 2009.	136
Figure 5.39: Difference in demand profiles for 2009.=.....	137
Figure 5.40: Demand scheduling variables for 2010.....	139
Figure 5.41: Demand scheduling cost variables for 2010.	139
Figure 5.42: Savings associated with PV array of variable size.....	141
Figure 5.43: Array cost associated with PV array of variable size.....	141
Figure 5.44: Payback period associated with PV array of variable size.....	142
Figure 5.45: Percent demand to be met with PV system.....	143
Figure 5.46: Facility monthly demand peak and demand to be generated at the same time.	144
Figure 5.47: Percent of demand to be met on a day per week basis.....	145
Figure 5.48: Daily facility demand max and solar demand max at same time.....	145
Figure 5.49: Percent consumption being met for every day of the year.....	146
Figure 5.50: Total monthly consumption of facility juxtaposed against the monthly energy generated through PV.	147
Figure 5.51: Demand scheduling variables for 2010.....	147
Figure 5.52: Comparison of solar power output and facility demand for 2010.	148
Figure 6. 1: Maximum demand per month for 2012.	150

Figure 6.2: Maximum demand per month for 2013.	150
Figure 6.3: Monthly demand curves for November 2012.	151
Figure 6.4: Monthly demand curves for November 2013.	152
Figure 6.5: Daily demand curves for November 04, 2012.	153
Figure 6.6: Daily demand curves for November 03, 2013.	153
Figure 6.7: Day per week demand curves for Tuesdays in 2012.....	155
Figure 6.8: Indicated date classified as having aberrant demand peaking behavior.....	156
Figure 6.9: Indicated date classified as having an aberrant demand profile.....	156
Figure 6.10: PV system savings curve.....	158
Figure 6.11: PV system cost curve	159
Figure 6.12: PV system cost payback.....	159
Figure 6.13: PV electrical demand percent met.....	160
Figure 6.14: Percent demand met on a daily basis	160
Figure 6.15: Percent demand met on an hourly basis.....	161
Figure 6.16: PV electrical consumption percent met.....	162
Figure 6.17: Consumption met on a monthly basis.	163
Figure 6.18: Percent consumption met on daily basis.	164

LIST OF TABLES

	Page
Table 4.1: Automated report generator ECM list.....	92
Table 5.1: Facility motor list.....	94
Table 5.2: Summary of electrical utility charges.	96
Table 5.3: Electrical load factor variables.....	101
Table 5.4: Production load factor variables.....	102
Table 5.5: Plant specific variables.....	104
Table 5.6: Summary of electrical consumption, demand and cost variables.	104
Table 5.7: Summary of consumption and demand characteristics.....	109
Table 5.8: Key monthly variables in electrical consumption for March 2008.....	115
Table 5.9: Key monthly variables in electrical consumption for August 2012.....	117
Table 5.10: Key daily variables for May 12, 2008.....	118
Table 5.11: Key daily variables for August 6, 2008	120
Table 5.12: Key weekday variables for indicated years.....	124
Table 5.13: Key demand aberration variables.....	127
Table 5.14: Daily consumption variables for July 15, 2007.....	128
Table 5.15: Daily demand variables for January 21, 2007.....	129
Table 5.16: Key 1-norm variables for all Mondays throughout 2008.....	130
Table 5.17: Key demand variables for March 31, 2008.	132
Table 5.18: Key electrical variables for March 31, 2008.	133
Table 5.19: Weather based consumption variables for 2010.	135
Table 5.20: Weather based consumption variables for 2009.	138
Table 5.21: Total demand scheduling variables for 2010.	140

Table 6.1: Table of final PV metrics165

1. INTRODUCTION AND MOTIVATION

The Industrial Assessment Center (IAC) program is a partnership between the U.S. Department of Energy's Advanced Manufacturing Office (AMO) and academia to further the cause of optimizing and reducing energy consumption in small to medium sized manufacturing facilities. The need for the program entails an amalgam of political, social, economic and environmental factors, but it suffices to say that reducing utility consumption and streamlining manufacturing processes is in everybody's interest. Future projections on the consumption and cost of energy enable several objectives, which, in part are conducted through the IAC program, objectives such as: bringing innovative manufacturing technologies, encouraging improvement in energy management, as well as helping to meet environmental emissions goals. To understand the motivation for the IAC program and the subsequent tools that have been developed for the Industrial Assessment Center at Texas A&M University (TAMUIAC), this thesis will begin by establishing the current state of energy use in industry, creating a reference for a discussion on the future of energy use and energy cost in industry. After, the particulars of the TAMUIAC program, with special attention given to the ECM "implementation" metric will be explored. Finally, a discussion on the need for the ensuing software tools as they pertain to TAMUIAC operations will be examined.

Industrial Energy Use Today

In the United States, there are four major end-use consumers of energy; commercial, residential, transportation and industrial sectors.

U.S. Energy Flow, 2013 (Quadrillion Btu)

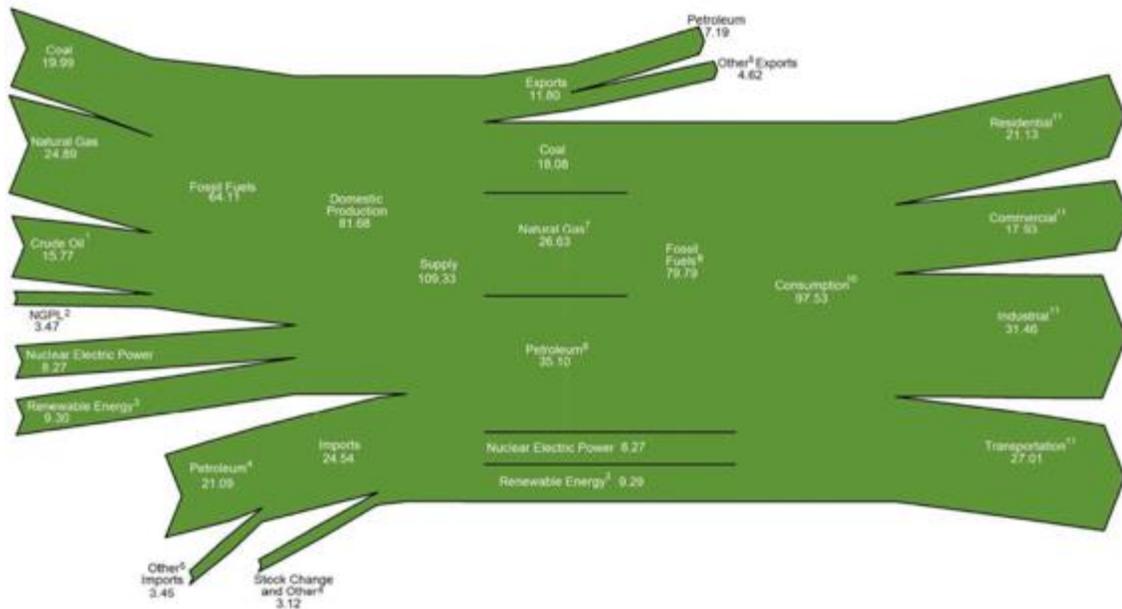


Figure 1.1: End use consumption for the four major consumers of electricity in the United States [1].

As of 2012, the largest single consumer of energy in the United States is the industrial sector, consuming 32%, as seen in 1.1. Energy consumption in the industrial sector is, by far, more varied than any other. Industry consumes energy not only to power various industrial applications, but also to produce feedstock's that are used by other facilities. Facilities geared towards petroleum distillation consume energy found in petroleum to produce fuels and other commodities that are used in transportation and power generation, to name a few. The bulk of energy use in transportation is found in fuels of various degrees; gasoline to fuel motor vehicles and diesel fuel to power transcontinental freight trucks. Both residential and commercial facilities consume energy in much the

same manner; natural gas is typically used for heating and electricity is used to power plug loads and HVAC (Heating, Ventilation and Air Conditioning).

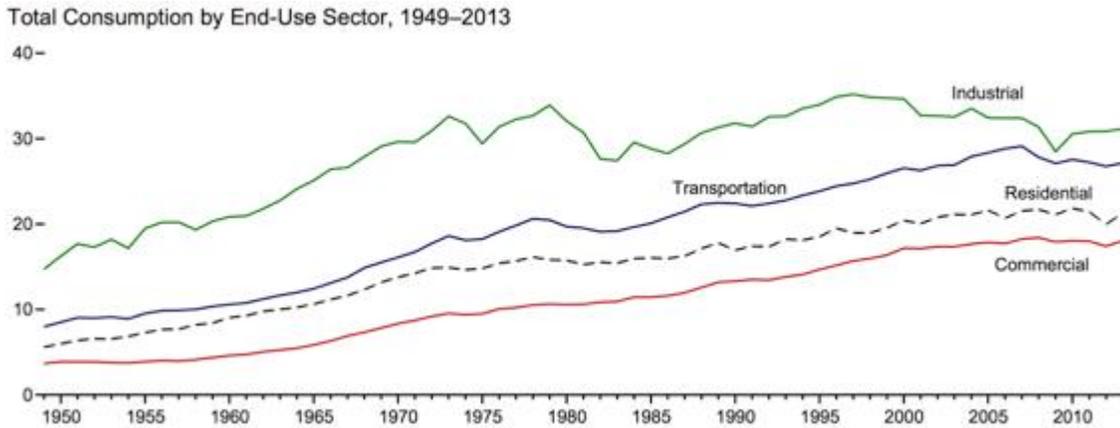


Figure 1.2: Energy consumption for the four major consumers of electricity in the United States, from 1950 to 2013 [2]

Figure 1.2 depicts how energy consumption has increased from 1950 through 2012. A steady rise from 15 quadrillion BTU's in 1949 to 35 quadrillion BTU's in the early part of the 1980's. The economic recession of the 1980s halted the steady growth of energy consumption as industry throttled back production. Through economic recovery and growth, industry started to consume electricity once again, lasting through to 2008, when again economic disaster struck, thus hindering growth [3]. Today, industry is recovering from the economic collapse of 2008, growing from just below 30 quadrillion BTU's in 2009 to just above 30 quadrillion BTU's in 2012. Three things that should be gathered from these graphs; 1) The industrial sector consumes a majority of the energy in the United States, 2) Industrial electrical consumption is increasing and, 3) That increase in more or less steady, increasing by about 200 quadrillion BTU's per year.

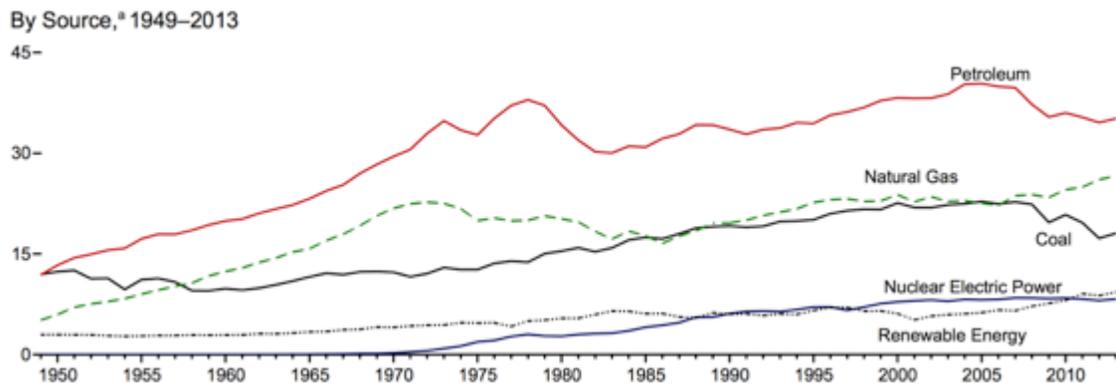


Figure 1.3: Major sources of energy consumption (in quadrillion BTU's) for the industrial sector in the United States, from 1950 to 2013 [2]

The industrial sector consumes energy in a variety of manners. The bulk of the consumption lies in five categories: petroleum, natural gas, electricity, coal and renewable resources, as seen in Figure 1.3. As of 2012, the largest consumption of energy in the industrial sector was in the form of petroleum.

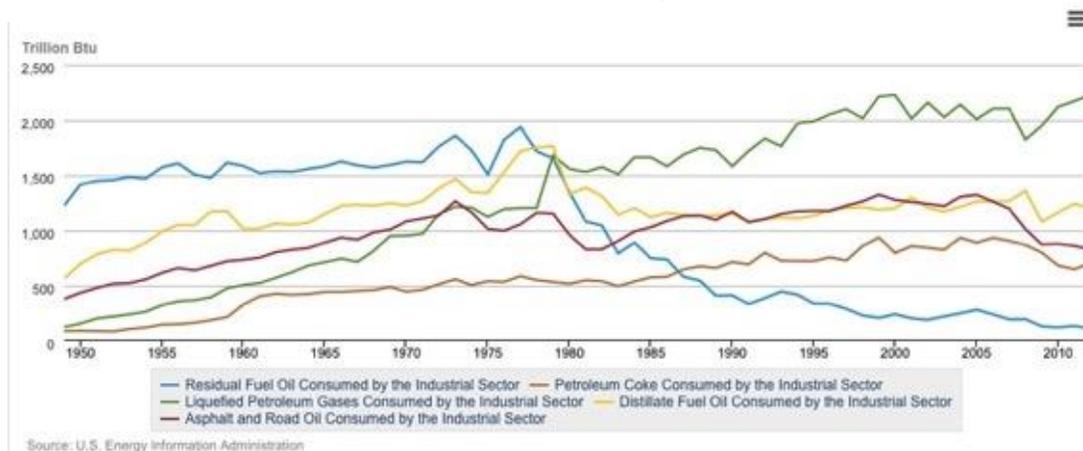


Figure 1.4: Uses for petroleum in industrial applications in the United States, from 1949 to 2012 [4]

Petroleum is predominantly consumed by the oil refining and petrochemical industries; refining crude oil into products for everyday use. As shown in Figure 1.4, these industries use petroleum in a variety of manners. The majority goes towards producing and consuming liquefied petroleum gas (LPG), topping 2,230 Trillion BTU's in 2012. LPG is derived from crude oil or natural gas. It is primarily composed of butane, propane or a mixture of the two and is predominantly used by the petrochemical industry as a feedstock to produce other commodities such as fertilizers or plastics [5]. Petroleum also goes towards distillate fuel oils; diesel fuel and heating oil. Asphalt and road oil, as the name implies are used to pave roads. In the past, coke was produced from bituminous coal, but more recently has been produced from petroleum. Coke is used a fuel source when processes require smokeless combustion. In addition, it is used as a reducing agent in the iron ore smelting process. Residual fuel oils are comprised of heavier hydrocarbons that are left over from the distillation process. They are used for steam-powered vessels in government service as well as electricity generation and space heating [4]. Since the late 1970's the use of residual fuels has been driven down due to the increasing supply of natural gas. The increased amount has driven natural gas prices down, which has caused many to use natural gas as a heating fuel rather than residual fuel oils. Some states such as Alaska and Florida still use residual fuel oils as a fuel source in residual fired power plants, but the availability of natural gas combined cycle units and stricter emission standards make using residual oils as a fuel source less common [6].

Traditionally, industry uses natural gas in a variety of manners; using it as a fuel source to fire boilers for hot water or steam needs, direct heating for melting, baking or drying such as for drying paper, melting steel, or baking ceramics. Natural gas is also used by many facilities to power operations in combined heat and power endeavors, providing electricity and heat [7].

Most industrial facilities use electricity in much the same manner; to drive electrically driven equipment such as motors, operate lighting or towards cooling. Electricity is either generated on site, or imported. When produced onsite, a variety of fuel sources may be used; coal, petroleum, wood, waste gas such as blast furnace gas or other waste gases derived from fossil fuel sources, wood or other waste (such as plastic) [8].

Industry uses coal in two ways. The first is in use with combined heat and power generation, using coal as a fuel source. Coal is a relatively inexpensive and abundant fuel source. Emission regulations and environmental standards coupled with the lowering cost of other fuels sources such as natural gas make coal as a fuel source less likely as time progresses. Industrial facilities also use coal and convert it to coke, which is used as a reducing agent in smelting iron ore as well as a fuel source. Coke is sometimes preferred to bituminous coal as fuel source since Coke does not produce smoke [9].

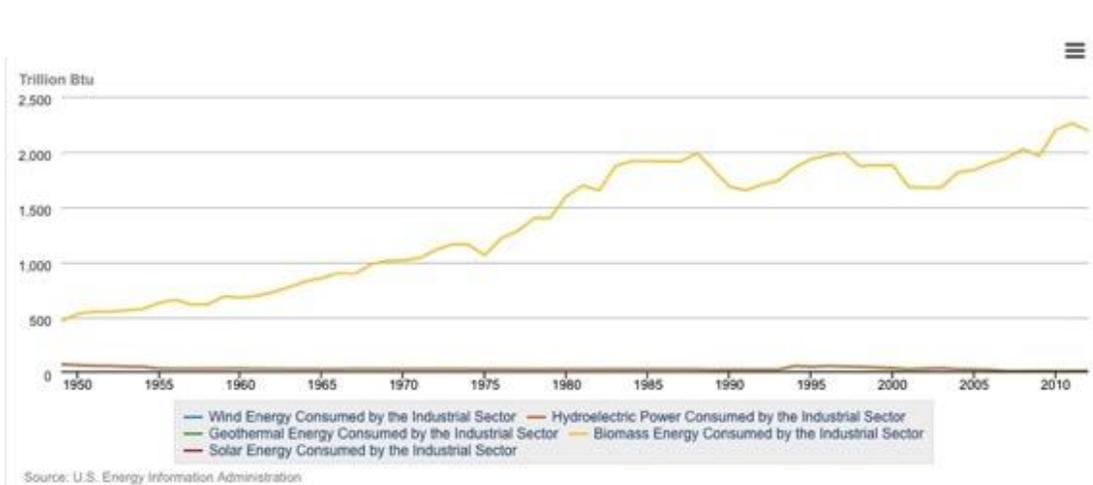


Figure 1.5: Usage of renewable sources of energy (in quadrillion BTU's) in both industrial and commercial applications, from 1949 to 2012 [8].

Traditionally, the least used energy source for industrial applications is renewable sources. Figure 1.5 depicts the different types of renewable resources in use in commercial and industrial applications as well as their usage trends for the past 61 years. The use of biomass in industrial practices is used, to a large degree for process heating in the food, tobacco, paper, pulp and printing industries. Carbon monoxide is needed to react with ferrous oxide to produce iron and carbon dioxide. The primary fuel source for this mechanism has traditionally been coal and, more recently coke. Carbon monoxide production via biomass is a leading competitor in ore processing. The petrochemical sector uses carbon in approximately 75% of its total feedstock. An alternative to carbon extraction from fossil fuels is biomass [10]. To a lesser degree, industry uses alternative sources for power generation, including wind, geothermal, solar and hydroelectric.

The Future of Industrial Energy Use

Worldwide energy consumption is expected to increase by an average of 1.4% per year.

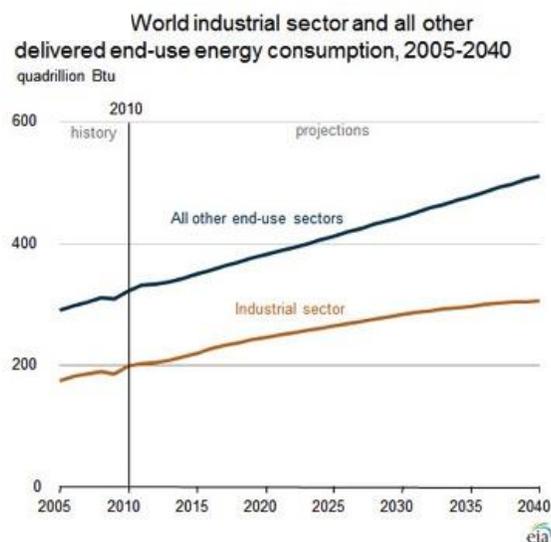


Figure 1.6: Worldwide use of energy for both industrial and other end use consumers, from 2005 to 2040 [11]

Figure 1.6 depicts the rising consumption of worldwide energy; rising from 200 quadrillion BTU's in 2010 to 307 quadrillion BTU's in 2040. The economic recession of 2008 saw a decline in industrial energy usage due to cutbacks in manufacturing.

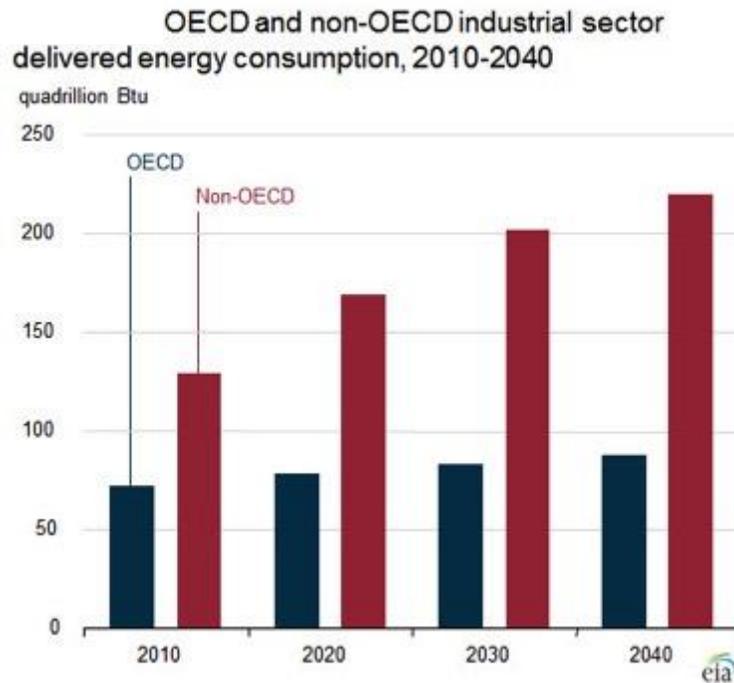


Figure 1.7: Worldwide energy use for both OECD (Organization for Economic Cooperation and Development) and Non-OECD countries, from 2010 to 2040 [11]

Figure 1.7 shows that the majority of energy usage is not expected to occur in OECD (Organization for Economic Cooperation and Development) countries, but in non-OECD ones. Energy consumption for countries such as China, India, Russia and Brazil is expected to rise by an average of 2.3% per year, in comparison to OECD countries whose contribution to energy consumption is expected to rise by only 0.4% per year. Energy consumption for the United States remains the dominant factor in the rising energy consumption for OECD countries throughout the next half-century.

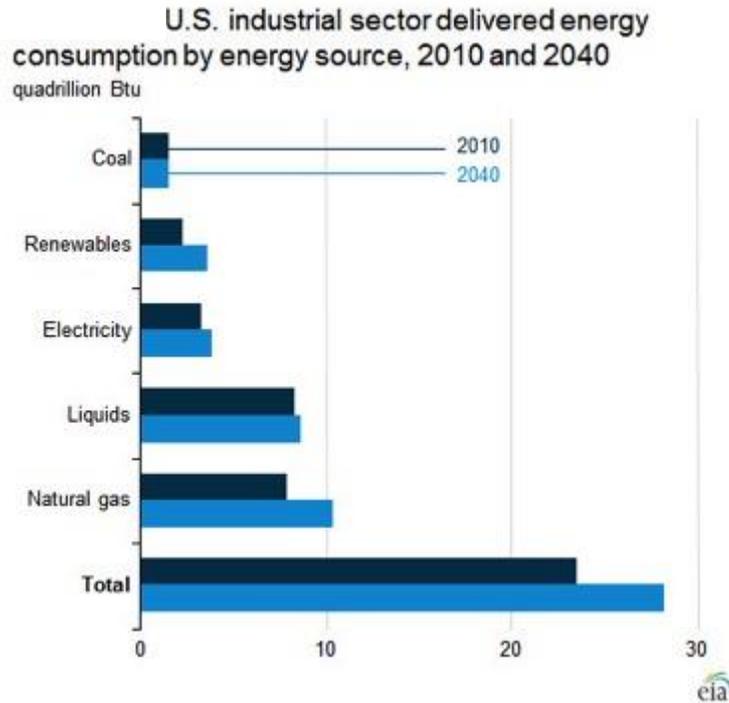


Figure 1.8: Energy consumption in the United States by energy type for 2010 and 2040 [11]

As seen in Figure 1.8, the sharp decline in coal usage for alternative sources such as natural gas shows no growth in the use of coal, for power generation, coke production or otherwise from 2010 to 2040. Renewable sources such as biomass are projected to be the fastest source of energy, rising from 10% in 2010 to 13% in 2040. Growth in the U.S. industrial sector is expected to rise to 28 quadrillion BTU's in 2040 and rise at an average annual rate of 0.6% per year. Natural Gas usage geared towards the production of chemical feedstock's is expected to increase as oil prices are expected to rise. While energy consumption in the industrial sector is expected to increase, legislation aimed at reducing energy usage will counteract some of the growth. The Department of Energy's, Office of Advanced Manufacturing, spearheaded by the Energy Policy Act of 2005 is

working to reduce industrial energy consumption, nationwide, by 25% by 2017. Government programs geared towards energy efficiency research also helps to reduce energy consumption by mitigating system inefficiencies. Although there is no formal program by the U.S. federal government to limit greenhouse emissions, some states, such as California have introduced state bills designed to cap greenhouse gas emissions. Also, the U.S. Environmental Protection Agency has the Boiler MACT program designed to reduce pollution from industrial boilers and process heaters [11].

Over the next 30 years, industrial energy prices are forecasted to rise. Figure 1.9 depicts the forecasted prices for various industrial fuels, from 2010 to 2040. The values forecasted are in nominal dollars; the price due to inflation has been adjusted.

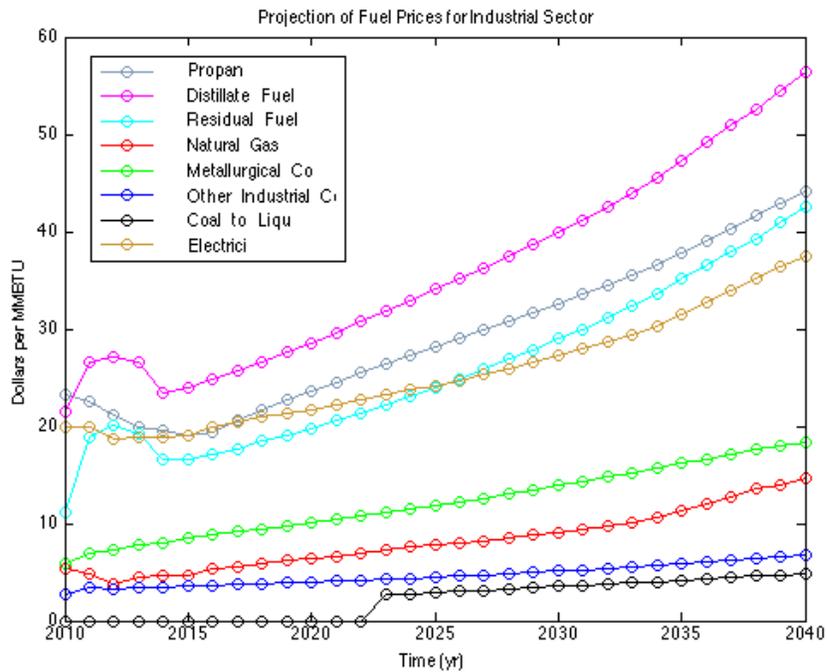


Figure 1.9: Projection of fuel prices for the industrial sector, from 2010 to 2040 [12]

There is a clear definition in the price between fuels that are harvested from petroleum stocks and those that are not. Since petroleum is a close contender with natural gas in industrial usage, it seems more likely that industry will turn to natural gas as a fuel source since the price for natural gas is about half that of petroleum based fuels. By 2040, the price for Natural gas is projected to be 14.65 dollars per MMBTU whereas the price for electricity is 37.54 dollars per MMBTU. In comparison, electricity costs 2.5 times that of natural gas. Exasperating the cost of electricity for industrial facilities is not only the consumption, but also demand, making electricity more costly than depicted [12].

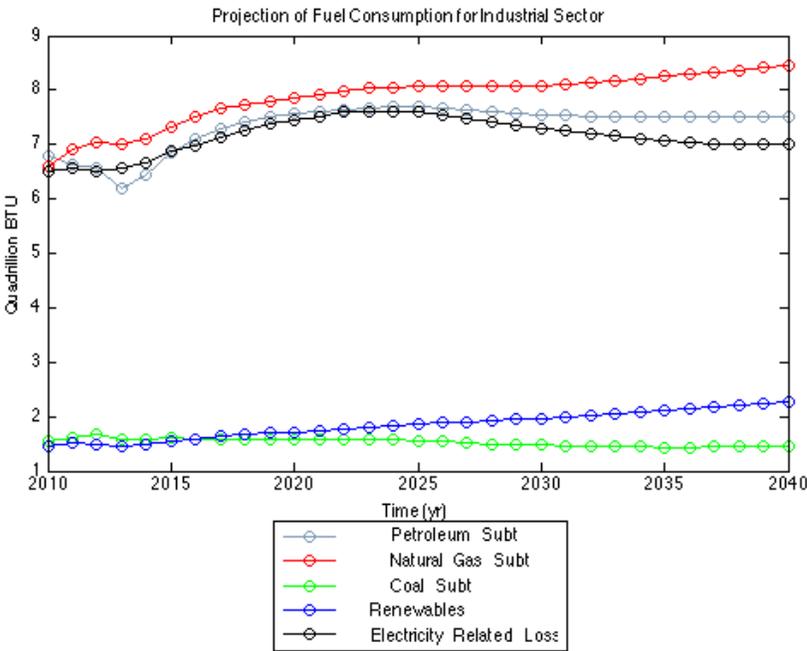


Figure 1.10: Projection of fuel consumption for the industrial sector, from 2010 to 2040 [12]

Figure 1.10 shows the projection of industrial fuel consumption for years ranging from 2010 to 2040. Figure 1.9 shows that the cost of natural gas is projected to be less than that of petroleum-based fuels. Since petroleum and natural gas have a similar consumption history, the price difference between the two ultimately will reveal natural gas as the industrial sectors largest fuel source, as seen in Figure 1.10. At 0.7%, the growth rate for the consumption of natural gas is nearly double to the 0.4% of petroleum. The next 30 years will also see an increased usage of renewable sources of fuel; renewables are projected to increase by 1.4%. The consumption of coal as a fuel source drops as other fuel sources become economically available. Electrical related losses rise by 0.4% per year, forecasting a need to optimize industrial energy consumption in a future where petroleum based electrical generation is predicted to rise [12]

Energy Efficiency and the IAC Program

Nearly 40 years after the oil embargo of 1973, the push towards conservation is still being realized. Federal organizations such as the Department of Energy's, Advanced Manufacturing Office support research initiatives to produce tools and innovate on efficient technologies geared towards industrial energy usage [13]. University based endeavors such as the Texas Engineering Experimentation Station (TEES) resolve to optimize commercial energy consumption through the Energy Systems Lab (ESL) and industrial energy conservation through the Industrial Assessment Center (IAC) [14]. Energy Service Companies (ESCO) perform energy audits across a wide range of platforms, from commercial buildings to government institutions, educational and medical [15].

The TAMUIAC is one of 24 centers spread across the continental United States. Supported by a grant from the Department of Energy's, Advanced Manufacturing Office, the motivation of these centers is to perform energy, productivity and waste audits for small to medium size industrial facilities. The following criteria must be met by a facility to be considered an audit candidate.

- No dedicated staff to perform assessments
- Less than 500 employees
- Gross annual sales less than 100M
- Annual energy bills between \$100K and 2.5M
- Within 150 mile radius of university
- Within standard industrial codes (SIC) 20-39

Once a facility meets these criteria, first contact is made. There are several traditional methods a facility is chosen; the first is through referral. A referral can be made by the Office of Advanced Manufacturing through the "Better Buildings, Better Plants" program. The program entails a pledge to reduce energy consumption by 25% within 10 years. A referral can also be made by a facility that has already been audited. In many cases, industrial sites are supported by "sister" sites in other areas that could benefit from an IAC audit. The more traditional method is for the TAMUIAC to actively seek a facility. The Texas Manufacturers Register is a database of industrial facilities throughout Texas. Through the Register, potential audit candidates are identified. Once a facility agrees to an audit by the TAMUIAC, several steps are taken; from the point of initial contact to report presentation.

After a facility agrees to proceed with the audit, the plant will supply utility bills of three types; electrical, natural gas and water. Electrical utility bills are typically focused on more than any other type, since, most ECMs tend to fall under an electrical savings category. The bulk of TAMUIAC analysis are designed to decrease electrical demand (measured in kW or kVA), consumption (measured in kWh) and improve power factor (the amount of supplied power that goes towards producing work), thus reducing capital investments made towards electrical usage. This usually takes the form of retrofits, upgrades or replacements.

After receiving utility bills, a student technician conducts a full pre analysis of the current state of utility usage for the plant. This analysis entails base lining electrical usage statistics; monthly electrical demand, monthly electrical consumption, power factor, monthly costs, unnecessary charges and trend analysis. This information is used to gauge potential aberrations in facility utility usage. Larger than normal electrical usage during the summer months may be evidence of cooling, alerting TAMUIAC personnel of the potential to investigate technologies dependent on these power supplies.

On the day of the plant visit, the first order of business is to meet with plant personnel where the results of the utility pre analysis are presented. Plant personnel are interviewed for justification of the pre analysis results. TAMUIAC personnel gauge the current state of facility affairs by establishing plant statistics. Compensation figures, process methodologies, annual product figures, hours of operation, current plant operations, etc. are all confirmed. After the initial meeting, visiting TAMUIAC personnel are taken on a tour of the plant. While on the walkthrough, the focus is to

recognize potential savings. Lighting type, the potential for savings due to over lighting, thermal system insulation, boiler maintenance, motor and lighting ballast usage, compressed air system inefficiencies, etc. are all evaluated during the walkthrough. After, TAMUIAC personnel meet to discuss possible ECMs. Such measures entail retrofit and optimizations designed to reduce plant utility consumption.. After a list of potential ECMs has been identified, the director or associate director is tasked with determining which ECMs are viable for data collections, which are constrained by time. The director or associate director then assign the chosen ECMs to students, who are then sent out to the plant to gather the relevant data necessary to calculate savings pertaining to their ECM. Repairs to the compressed air system require information pertaining to the supply pressure and system volume. Motor retrofits require information about the number of present motors and their power requirements. After collecting data, a close out meeting is conducted to alert plant personnel of the preliminary findings and the ECMs to be expected in the final report; to be handed in 60 days after the initial visit.

After returning from the plant visit, TAMUIAC personnel are given up to 2 weeks to compose their respective ECM reports. Utilizing the collected data, they calculate consumption and cost savings. All the individual ECMs are collated and gathered by the lead student, which is typically the same person who conducted the utility pre analysis. The lead student spends several weeks correcting errors and aberrations found in the collated report. The final report is turned into the industrial facility no more than 60 days from the initial visit. Six months after the final report is turned in, either the director or assistant director contact the facility to assess which

ECMs were considered viable and worthy of implementation. In addition, plant personnel are surveyed to measure the accuracy of cost and consumption savings analysis.

While there are several metrics used by the Department of Energy (DOE) to measure the efficacy of a particular IAC, ultimately it is a facility's response to the gambit of ECMs proposed that underline a centers potency. ECM implementation is perhaps the most valued of all productivity indicators.

TAMUIAC ECM Implementation

Since 1981, the IAC field management office at Rutgers University has collected public use data on past and present IAC's. Several facility visit factors, including assessment recommendation implementation have been gathered. This data, in conjunction with internal TAMUIAC metrics serve as the basis for the following analysis. Since 1981, the TAMUIAC has made over 310 different recommendations ranging from replacing metal halide illumination with efficient fluorescence to changing a pool heater temperature set point. To date, out of more than 640 visit, repairs to the compressed air system have been recommended 503 times, or approximately 78% of the time.

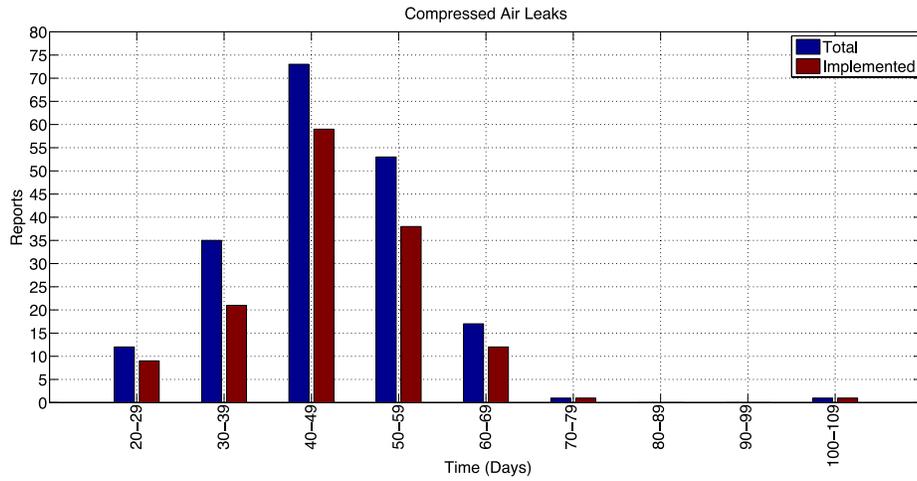


Figure 1.11: Timeliness trends for TAMUIAC ECM to repair compressed air leaks.

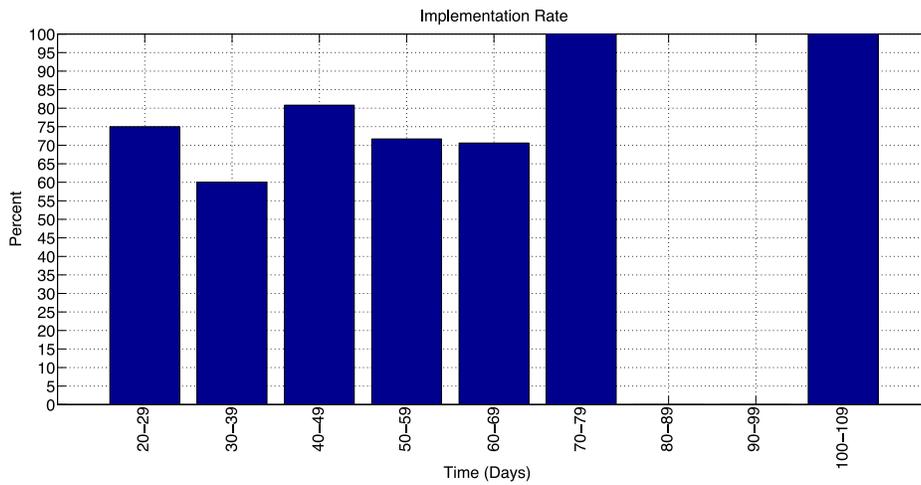


Figure 1.12: Implementation of TAMUIAC ECM to repair compressed air leaks.

Figure 1.11 shows 193 separate instances of recommendations to repair air leaks to compressed air systems; the number of days it took to provide a facility with a comprehensive report vs. the total number of reports that fell within that time frame

(blue) and the number of reports that were implemented (red). Figure 1.12 provides the implementation rate for the corresponding reporting times.

A majority of reports took between 40 and 49 days to turn in. Implementation for these reports hovered around 80%. Reports that took longer had a 100% implementation rate, partly because the sample size was extremely small. In general, there doesn't seem to be a correlation between implementation rate and report timeliness. The same type of trend can be seen with other ECMs recommended by the TAMUIAC.

Motivations for Change

The procedures adopted by the TAMUIAC have proven effective with over 640 facility visits, but while the adopted methods have proven effective, there are some limitations; the first entails the fidelity of the available data.

For the most part, electrical utility bills entail two separate electrical charges. Electrical demand refers to the rate at which electricity is consumed. A facility is charged electrical demand because their electrical needs are sizable. In response, the electrical provider is tasked with maintaining and installing the electrical infrastructure required to meet the facilities electrical needs. Since the nature of manufacturing operations usually entails periods of high productivity as well as non-production hours, the largest electrical demand in a month is used as the facility demand. In addition to electrical demand, a facility is also charged for electrical consumption. Electrical consumption refers to the total amount of electricity consumed. This method of charging for electricity is standard across the array of end users, from commercial, residential and industrial consumers. When ECMs are recommended, many take into account a

reduction in both electrical demand and consumption. To properly determine if an ECM will result in a reduction of electrical demand, the time of the demand peak must be established. If the ECM entails retrofitting motors with higher efficiency ones, then the savings resulting from a reduction in electrical demand will only be valid if motors are running during the demand peak time. The current standard for reporting electrical usage to a customer in electrical utility bills is to forego informing them of the time and day in which electrical demand peaks occur. Electrical usage data with higher resolution is needed to establish electrical demand peak time. Greater data fidelity has many other advantages; including diagnosing possible demand or consumption aberrations. Demand aberrations include unusual electrical demand peaks while consumption aberrations are usually characterized as electrical usage during unusual or non-operating hours. Using electrical utility bills as method for benchmarking facility electrical usage is an excellent “first step” in gauging overall electrical usage, but exemplifies significant shortcomings under many conditions.

Another limitation to the current auditing method involves an all too common problem, time. Since the TAMUIAC program is supported through direct funding from the United States Department of Energy, there are certain guidelines that must be adhered to and the amount of time that a center is allowed to spend at a site is constrained to a single day; the time constraint is a limiting factor. Where 20 ECMs are identified, half may be chosen as viable since the time constrains one on the amount of data that can be gathered. To increase the effective potential of the IAC program, a method for identifying ECMs before the onsite visit helps in both establishing the

validity and expertise of IAC personnel, as well strengthen the impact of the program by increasing the number of recommended ECMs

Data fidelity and the temporal impediment are only some of the challenges faced by the TAMUIAC. Another limitation to the current manner of conducting energy audits concerns the type of available information. Several key variables, specific to individual facilities are required to formulate the energy and cost saving parameters inherent in most ECMs. Many times, inquiries with plant management yield rounded figures that skew calculations. Plant management may provide an average figure on the hours of operation per year because, although a plant may run on a 9-5 schedule, 5 days per week throughout the year, there may be some weekends in which the plant may operate during the weekends. Without a record of the days of operation, plant management opt to give average or rounded figures and ECM calculations suffer.

In an effort to adjust and resolve some of the above-mentioned limitations, other avenues of exploration open. To increase the fidelity of the data available in an effort to resolve some of the questions concerning facility demand peaking allows the IAC to take advantage and size systems that reduce and optimize electrical utility usage; to size capacitor banks for power factor correction or forecast how much electrical consumption and demand can be expected from a photovoltaic (PV) power system.

Another benefit to the resolution of the above-mentioned drawbacks is the creation of a simple user interface that can be used by anyone that has some rudimentary knowledge of the mechanics involving electrical usage. The prevalence of IDR (Interval Data Recorder) meters improve the data collection necessary by electrical utility

providers to accurately gauge electrical usage by industrial, commercial and residential usage. Today, anyone with an Internet connection can download this data. In many cases, the interface provided by the utility to analyze electrical usage is minimally helpful and limited to visualization on a limited set of timescales. Also, usage comparisons are found to be poor and resistant to the in depth analysis required to make informed decisions. A result of the development of tools to be discussed in the following chapter is the creation of a set of software tools that expands on the data analysis missed by most, if not all, utility providers, thus providing an interface that is multifaceted in use and applicable to a variety of end user applications. The tools can be wielded by plant or building managers, residential or commercial end users to make informed decisions on how to optimize their electrical usage.

The final challenge that the resultant software tools endeavor to adjust and innovate on concerns ECM implementation. It has been mentioned that the TAMUIAC has 60 days from the initial plant visit to compose and return a report that outlines the various ECMs that were discussed during the closing meeting with plant personnel. Out of 631 different facility visits spanning from 1987 to 2013, there have been a total of 4,922 ECMs recommended. As it currently stands, a total of 2,944 ECMs have been implemented throughout various manufacturing facilities. One of the metrics that are used to measure the effectiveness of an IAC is the centers implementation rate; the TAMUIAC has an implementation rate of 59.8%. In an effort to increase the rate of implementation, the TAMUIAC has developed an automatic report generator to decrease the time in which ECMs are reported to the client. Since there is precedence for energy

auditors to affect the likelihood of adopting an ECM by adjusting various psychological factors, the TAMUIAC hopes to increase ECM implementation by decreasing the time between the site visit and reporting time.

2. PREVIOUS WORK

The work to be discussed is centered on two areas; an investigation of the psychological factors that effect ECM implementation and a review of electrical energy usage analysis software (energy dashboards).

Behavior and Psychology

In the area of psychology, the work of Francesca Gina and Gary Pisano in, “Toward a Theory of Behavioral Operations” [3] outlines two different approaches to effect operations management (OM) behavior. The first is prescriptive in which behavioral factors need to be included in OM models. The second is descriptive in which one tries to understand the effects of the decision making and problem solving process. Out of several descriptive biases that effect OM decision making are conservatisms, describing peoples inability to update their beliefs based on new and upcoming information, as well as overconfidence in which the tendency is to be presumptuous about their behaviors and opinions.

An article entitled, “Leaping the Efficiency Gap”, found in Science Magazine and written by Dan Charles [1], Charles expounds on several interesting case studies where external factors triggered behavioral changes to implementing ECMs over a large population. In particular, one example describes how an avalanche cut access a hydroelectric power reservoir for a large potion of the Juneau, Alaska supply area. As a result electricity consumption fell by 40% within a 6 week timeframe. Once the infrastructure was repaired (3 months later), electricity usage rose to normal levels. In

another example, the Sacramento Municipal Utility District (SMUD), in an effort to change usage patterns began an experiment to inform a select group of customers of their usage in comparison the average usage of their peers. One customer responded via mail, “I resent being told that I am below average”. The article goes on to highlight several other events that have changed end user behavior patterns.

The work of Muthulingam, et al, in “Energy Efficiency in Small and Medium Sized Manufacturing Firms: Order Effects and the Adoption of Process Improvement Recommendations” [5] goes on to describe how the order of an ECM report can effect the likelihood of adopting a particular ECM. Using the IAC database, Muthulingam, et al show that there is a statistical difference between the adoption of ECMs that are placed at the beginning of an ECM report collection, than those ECM reports that are placed towards the end.

In, “Overcoming Social and Institutional Barriers to Energy Conservation” by Carl Blumstein, et al [16]. The Lawrence Berkeley National Laboratories (LBNL) colleagues investigate the psychological and social hurdles that impinge on the adoption of ECMs. The group classifies several causes, including; misplaced incentives, lack of information or misinformation, market structures, financing and custom programs. After several involved investigations, case studies and interviews with personages in positions to adopt and implement ECMs, they found that among those who are in a position to reduce electrical consumption in residential building, the financial and informative variables that can effect a decision were not present. One owner of an insulation installation company mentioned that they were not proponents of standardizing ECM

measures because such actions because they were “concerned with the impact of regulation on the interest of the associations members.” The LBNL group concludes by suggesting several changes to overcome some of the above-mentioned barriers. The first consists of a series of strategies. A) Disseminating information where a lack of information exists, B) Government Leadership to encourage ECM behavior. C) Creating markets for ECM products and services. D) Making rules for commercial transactions. E) Effecting incentives by government to conserve and consume by changing the price of energy and conserving commodities. Finally F) Rationing to conserve scarce resources by limiting consumption. The second concerns the impact of adopting ECM measures. The suggestion is that ECMs that conflict with other economic and social goals are more likely to be unaccepted.

In “A review of intervention studies aimed at household energy conservation” by Wokje Abrahamse, et al. [17] the group from the Department of Psychology at the University of Groningen in the Netherlands, the group reviews and evaluates different psychological factors that effect conservation measures aimed at household and residential ECMs. The group identifies two types of factors; antecedent and consequence intervening factors. Antecedent factors include goal setting, workshops, mass media campaigns, modeling and tailoring home audits. These methods are designed to effect ECM implementation by empowering a decision maker with the required information to effect decision-making behavior. The second factor, consequence intervening factors is based on the underlying assumption that positive and/or negative consequence affect adoption behavior. These factors entail a variety of feedback mechanisms; continuous,

daily, comparative feedback, as well as rewards in the form of monetary motivators. Their findings concluded that both types of psychological stimulators are effective in the adoption process and concluded that evaluations in ECM adoption measures should include behavioral as well as energy related determinants.

In “Social Norms and Energy Conservation” by Hunt Alcot [18], Mr. Alcot reviews a non-price energy conservation program implemented by OPower, a Virginia based company offering cloud based software for utilities aimed to reduce electrical usage. The paper evaluates a program that mailed home energy report letters to households that compared energy usage to that of neighbors. The program was influenced by the idea that social norms caused people to adopt ECMs. Quantifying the results of the study by incorporating factors of attrition and the potential for a particular outcome, the study found that the program had the tendency to effect energy conservation over the long term. Households that participated in the program showed a 2% decrease in electrical consumption, which translated to 0.62 kWh of electricity, on average, per household.

In “Energy Conservation Through Product-Integrated Feedback: The Roles of Goal Setting and Social Orientation” by L.T. McCalley and Cees J.H. Midden [19], the group from the Technical University Eindhoven in The Netherlands attempted to deduce if there is a significant difference in energy consumption behavior when an experimental group is augmented with energy consumption feedback. Participants were scrutinized for social and personal behaviors that effected their energy conservation habits. After baselining their personal habits in consuming electricity and washing clothing,

participants were told that they could reduce their energy consumption by decreasing the wash water temperature. The study showed that when participants were informed of methods to decrease energy and outfitted with a means to see how much energy they are consuming, participants chose to comply with the recommendation. At most, the study found that participants who were allowed to set their own energy saving goals saved upwards of 20% of the energy they would have normally used to wash clothes. The researches conclude by suggesting that people be allowed to make their own decisions on how much energy they should save, thus reinforcing social and cultural norms.

The psychological impact of implementing ECMs is both selfish; the requirement to reap a benefit from decision to be energy conscious. Many of the authors mentioned above show that it is a consumer need to be rewarded for an act of pseudo altruism that is a driver in the decision making process. Additionally, a lack of information for decision makers needs to be considered. It is all too common that decision makers be uninformed, in both scope and impact, when posed with the potential to endeavor and ECM project.

Energy Dashboards

TXU My Energy Dashboard

In the dashboard genre, electrical providers use energy dashboards to inform residential customers on the amount of electrical energy they are using at any given time. The information empowers the customer to effect their consumption, enabling them to make informed decisions. TXU Energy, servicing major population areas in Texas, provides customers with the “MyEnergy Dashboard” user interface.



Figure 2.1: “MyEnergy Dashboard” user interface available to select TXU customers. [2]

As seen in Figure 2.1, the dashboard informs the customer how much energy they have consumed in any given month. If so desired, the customer can view current usage information updated every 24 to 48 hours. The temperature correlation helps the customer deduce how much consumption is due to weather dependent residential energy consumption (primarily through HVAC). In addition, the customer is able to view and compare bills. Information displayed on the TXU energy “MyEnergy Dashboard” can be broken into 5 separate categories, the first being cost and usage summary. As stated earlier, the cost and usage summary displays the current electrical usage for the utility customer as well as the estimated cost. Again, this is to inform the customer of how they are using electricity throughout the year. The information can be used by the customer to attenuate unneeded electrical utility usage throughout the months, thus promoting savings. The second category is a bill comparison; the energy dashboard allows the customer to compare bills from one month to another, allowing them to see electrical

usage increases and decreases as the various factors that influence electrical usage throughout the month change. The comparison is done with two historical electricity bills. The third category is the usage graph. The graph allows the utility customer to view, graphically, what the electrical consumption is within a 24 to 48 hour period, showing the utility customer in “real-time” what their electrical usages are. “The Last Two Bill” category shows cumulative electrical consumption and total current charges as have been shown on the last electrical bill, allowing the customer to make a side-by-side comparison with usage and cost data from one month to the next. The final category is the usage analysis tool and helps the residential electrical consumer compare how they use electricity with their residential counterparts as depicted in Figure 2.2.



Figure 2.2: “The Last Two Bill” category snapshot available to TXU utility customers through the TXU My Energy Dashboard. [2]

Using a series of inputs such as home type, square footage, heating type, home age, number of occupants, pool owner, etc., the tool allows the user to gauge how well they consume electricity compared to others of similar home types. The tool informs the user of the usage breakdown, separating electrical usage into several variables such as cooling, heating, lighting, refrigeration, pool, others, etc.

Energy AI

Another similar tool geared towards commercial and industrial electrical consumers, “EnergyAI” is an energy analysis tool whereby customers are able to upload electrical demand data, spanning anywhere from 6 to 13 months and receive an energy analysis report depicting savings related metrics. After uploading demand data into “EnergyAI” the data set is run through a data validation algorithm. Aberrations such as misreading’s, zero, negative values, or data sets not meeting the 6 to 13 month criteria are preprocessed. After processing, long-term trends are identified. Seasonal patterns such as heating and cooling cycles or load creep from added plug loads are graphed and trended. The software attempts to identify load profile patterns and analyzes hourly load changes over the data set period. The patterns identify factors such as weekday usage versus weekend usage. Included in the analysis is when load increases and decreases tend to happen as well as when facilities “Wake up” or “Go to Sleep”. Also identified are 24-hour loads from equipment that is left on throughout the day and possibly throughout the year; how much of the 24 hour load makes up for the total load and how much the 24 hour load is costing the facility. More importantly, the software helps the user to identify deviations from expected use trends. Such deviations are helpful in

identifying potential anomalies where unexpected electrical demand and consumption are impacting facility electrical costs. Such aberrations might take the form of increased demand on a Saturday afternoon when facility equipment was left on, equipment such as lights, HVAC or industrial equipment such as compressors. EnergyAI attempts to identify periods of decreased usage, termed “Shutdown”. These are periods when facility demand is at a minimum throughout a series of consecutive days. Finally the software is able to extrapolate potential energy savings from the above-mentioned characteristics as well as savings in greenhouse gas emissions. When the analysis is completed, an extensive report outlining various demand and consumption factors; such as savings opportunities, avoided electrical consumption, the cost associated with such consumption and they’re avoided greenhouse gas emissions are output. In addition to potential opportunity savings, the report also outputs savings due to minimizing 24-hour loads as well as the opportunities savings associated with decreasing and eliminating load aberrations. Also present are the daily load profiles for the facility.

Seldera

Measuring facility performance is not limited to gathering metered data. In an ideal world, the use of sensors and monitors in key positions throughout a facility may be considered as “the best” method for gauging how a facility responds to electrical consumption and demand usage. Rather than trying to disaggregate individual loads from the complete load profile of a facility (such as weather based load disaggregation) sub metering posits the potential to get accurate and unbiased data from individualized facility systems, thus removing the uncertainty inherent with modeling to disaggregate

individual loads. In much the same manner as the previously mentioned companies, Seldera uses a network of small meters and sensors to monitor facility conditions. The data from these meters are collated and analyzed for the potential to exploit energy-saving methods. Seldera claims to have a three tier of service model; the first indicative of benchmarking and remote auditing. This tier is nothing more than an energy-metering dashboard and is implemented in two phases. The first phase consists of a 5 to 12 week investigation aimed at conducting initial load disaggregation and hourly load profiling. The second phase provides the client with continuous monitoring of facility energy performance, helping energy managers to consistently tune their facilities such that optimal energy usage is achieved. The second tier consists of consumption profiling. This requires the use of simple, wireless sub meters and sensors to continuously measure and profile individual facility systems (HVAC, motors, compressors, etc.). The third and final tier consists of automating building savings. This is done by analyzing the data from sub metered components as well as facility energy consumption. From this data, trends are extracted and control points are adjusted such as to optimize building energy consumption. These controllers automatically adjust individual building equipment as well as various other electric loads. While Seldera's methods may seem thorough in way of quantifying and managing facility energy consumption there are some drawbacks to be had as well. The invasive procedure of sub metering key facility systems may not always be possible for facilities. For those adept at performing ASHRAE level one energy audits, the time and expertise required to implement Seldera's method may not be possible. Additionally, as has been mentioned from some of the previous examples,

the lack of information pertaining to the method of individual load disaggregation leaves something to be desired; optimizing facility energy consumption by controlling key control points may not be viable for all facilities. Under the best circumstances, decision making should be up to the building energy manager or energy auditor to perform adjustments to the manner in which a facility consumes electricity, after analyzing all the variables at his or her disposal.

Energy PlanIT

While the suite of tools described above are designed to impact facility electrical consumption by monitoring and trending plant electrical usage, there are dashboards that trend electrical savings after retrofit and commissioning endeavors; the “Energy planIT” is such a dashboard. The dashboard is an online cloud-based tool that helps commercial facilities to gauge the power and energy savings after assessment recommendations have been implemented. The tool gives the user a graphical representation of how a facility has used electricity within a given month. It shows how much money has been spent in electrical utility usage and compares one month to the next to show savings. Given two consecutive months and the comparison between a current month and a previous month, if electrical usage has decreased, then efficiency is trended between those two months. Over the course of several months, the efficiencies are trended such as to give an overall picture of how the implementation of assessment recommendations have affected the facilities energy consumption. Additionally, Energy planIT shows the customer the impact on utility rates as well as the impact of assessment recommendations on the total budget. If there was a 2% decrease in electrical

consumption from January to February, then that decrease translates to electrical utility savings which therefore impact the budget pertaining to energy related retrofit and commissioning endeavors. Other than giving the user a color-coded and graphical based experience with the electrical data set, the cloud-based software does little more than trend. Unlike many of the counterparts mentioned above, the software seems to lack the ability to analyze and deuce measures that can be taken to further reduce the facilities electrical energy consumption.

Although each one of the software tools mentioned above have strengths and weaknesses, the goal of the software tools to be mentioned below to aid in making reparations for what the above-mentioned software tools lack. Interaction with the data set involving a graphical user interface is preferable to rote automation. While, for the future, it may be possible, and with complete confidence to leave energy management to digital means, for now an experienced user is preferable. Energy savings recommendations require analysis and should be individualized to how unique facilities operate.

While the tools do represent a valiant “first step” in the right direction, the hope that the tools to be described in the next chapter make up for some of their failings.

3. SOFTWARE TOOLS

The following suite of tools is designed to optimize and simplify the analysis of energy consumption. They are designed to help quantify and categorize key factors that promote the identification and implementation of various ECMs. As was mentioned previously, each of the tools falls under one or multiple of the following categories; designed to increase analysis fidelity, identify pre-visit energy conservation measures (ECM), establish unknown variables helpful in diagnosing ECMs, size systems design to optimize electrical usage, create a simple, user friendly interface and increase ECM implementation.

Demand Visualization

Visualizing demand is more than the name implies. While a visual representation of electrical demand is a first step in identifying how a facility consumes electrical energy, electrical demand data is wrought with a host of informational “nuggets” that can help an experience energy auditor identify ECMs. Since the information pertaining to these measures can be best identified across multiple timescales, the demand visualization tool is broken up; visualization occurs on daily, monthly, weekly and yearly timescales. Each timescale is necessary to determine ECMs.

Monthly

What a facility pays for demand is based on the highest electrical peak demand value throughout the month. If a facility consumes electricity at a rate of 600 kW throughout the course of the month, then the electrical provider will charge them for

consuming electricity at the rate of 600 kW. If the facility consumes electricity at a rate of 600 kW for most of the month and during the course of one 15 minute period, the same facility required a usage of 1000 kW, even though they consumed electricity at the rate of 600 kW for most of the month, the facility would be charged for consuming electricity at the rate of 1000 kW regardless of the frequency of usage. For this reason, visualizing electrical demand on a monthly timescale becomes important to answer when monthly demand peaks are occurring throughout the month, why are they occurring and if whether their occurrence can be mitigated.

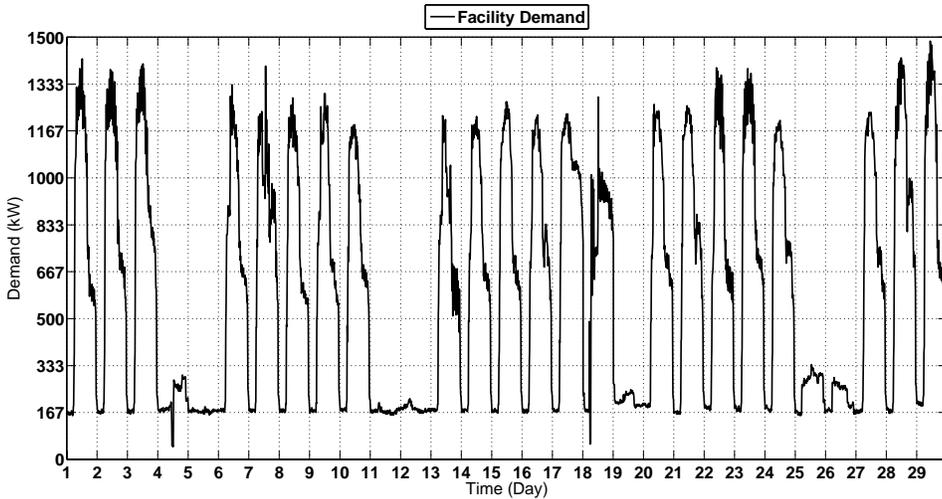


Figure 3.1: Total demand curve. February 2012.

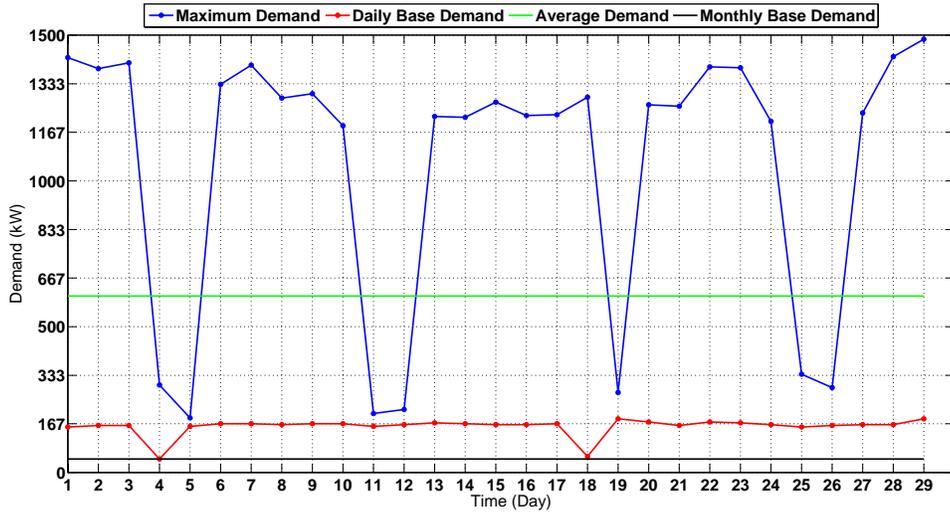


Figure 3.2: Reduced total demand curve. February 2012

Figure 3.1 exemplifies the total aggregated demand usage for an educational facility, located near Houston, TX. The demand points are taken in 15-minute increments. The weekdays are clearly visible and are represented higher demand usage days; the weekends are depicted as the interim periods between weekdays, when demand usage is at a minimum. Figure 3.2 is a reduction of Figure 3.1 and has extracted information from the aggregated demand profile; information such as the daily maximum demand (blue), daily base demand (red), average monthly demand (green) and monthly base demand (black). The reduced visualization highlights key data points designed to diagnose electrical demand and consumption abnormalities. The first diagnosis is if demand usage is typical or deviates from the traditional usage profile. In this case, there does not seem to be an atypical usage in demand throughout the month. The facility will be charged for 1,486 kW, the maximum electrical demand for this

facility was reached on the 29th. At a cost of \$8.00 per kW, the monthly demand charge will be \$11,888.

Since educational facilities shut down during weekends, it is expected that the electrical demand be larger on weekdays. A demand abnormality could be identified as increased electrical demand on a weekend, as seen on the 18th, but since this does not exemplify the largest demand usage throughout the month, it will not contribute to the facilities demand charge. This demand usage, however, will impact the consumption charge for the facility. Quantifying the consumption charge for facility management could help to justify savings and the need to impact facility usage during nonessential operating hours.

In addition to matters concerning electrical demand, industrial and commercial facilities also charged are electrical consumption (sometimes termed “electrical usage”). Unlike demand (the rate at which electrical energy is consumed), electrical consumption is the total amount of energy consumed. The relationship between demand and consumption are characterized as:

$$P = \frac{dE}{dt} \quad (i)$$

Where P is demand, E is consumption and t is time. Thus,

$$E = \int_{t_1}^{t_2} P dt \quad (ii)$$

Electrical consumption is the facility demand over a period of time. It is the area underneath the demand curve. Since the demand data set is discrete in nature, the analytical process of integrating electrical demand curves should also reflect

discretization, thus, electrical usage is derived using the process of summing across discrete trapezoidal areas; known as Riemann summing.

$$E = \sum_{i=1}^d (P_i)(t_{i+1} - t_i) \quad (\text{iii})$$

where d is the total number of interval demand data points and i is the interval value. While such a process will introduce errors, it is assumed that the time step from one data point to another is small enough to minimize said errors. After calculating the Riemann sum on the demand curve depicted in Figure 3.1, the consumption for the month of February was 421,474 kWh. At a cost of \$0.11 per kWh, the facility was charged \$46,362.

The daily base demand curve depicted in Figure 3.2 exemplifies the lowest electrical demand point for the indicated day. In many cases, even though a facility becomes dormant throughout different periods of the year, there is always an electrical power draw. Leaving equipment on, emergency lighting and unexpected nighttime operations can all be reasons for using electricity during non-operating hours. For facilities that do not run a 24/7 operation, the daily base demand can represent usage during non-operational hours and is the aggregated power for essential and non-essential systems. While it is unrealistic to expect a null power draw, it is possible to reduce the electrical power draw during non-operating hours by turning off non-essential equipment. Minimal demand usage represents an opportunity to save on electrical usage and these savings are represented in two ways. The first is by quantifying savings to completely eliminate the daily base demand. For the case depicted in Figure 3.2, this amounts to 105,768 kW, at a cost of \$11,634. Again, while it is unrealistic to assume a

facility completely eliminate their electrical usage during non-operational hours, the amount and charge for electricity during these hours does empower one to endeavor to reduce it. In this case, 25% of this facilities electrical usage happens during evening hours. The second way savings on minimum demand usage is represented, is by quantifying the savings associated with reducing it (unlike completely eliminating it).

A closer view of the daily base demand curve indicates a certain amount of variation. While it most certainly may be the case that such power usage is required, it may also be the case that the minimum demand represented is the summed demand from essential, off-hour equipment as well as nonessential off hour equipment. The curve may represent the power draw required for emergency lighting as well as the fact that some lights have been left on in the facility; lights that should have been turned off. The fact that the minimum demand curve shows variation indicates that, at least some of the electrical usage displayed in the minimum demand curve can be removed. If the minimum demand curve represented only parking lot lights, then the curve would be constant throughout the month. Variation in the curve indicates that other equipment is also being left on. While a recommendation to turn off parking lot lights may not be adequate, a recommendation to turn off nonessential equipment may be proper in this case. The amount of power consumed by this nonessential equipment thus becomes the difference between the daily base demand curve (red) and the monthly base demand curve (black).

$$E = \sum_{i=1}^d [(P_{i,DBD})(t_{i+1} - t_i)] - [(P_{MBD})(t_d)] \quad (iv)$$

Where $P_{i,DBD}$ is the power represented at the i^{th} interval for the daily base demand curve, P_{MBD} is the power for the monthly base demand.

Daily

Visualizing electrical demand on a monthly timescale can quantify consumption savings and diagnose unexpected demand usage. Visualizing demand on a more refined timescale, such as that on a daily basis has other advantages. While demand charge is the sole variable dependent on the demand peak, what is unclear by visualizing demand data on a monthly timescale is the peak time. The information could be useful in identifying if demand peak reduction is a possibility. Additionally, the daily demand curve could be used to identify non-essential equipment being operated during the peak time.

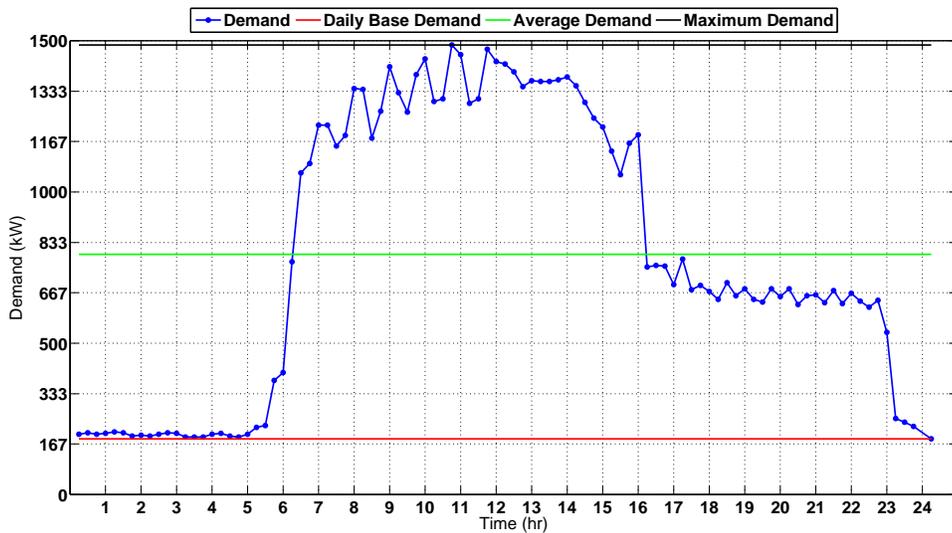


Figure 3.3: Daily demand curve. February 29, 2012.

Figure 3.3 depicts the daily electrical demand curve (blue), the average demand (green) and the daily base demand (red) for February 29, 2012; the day the electrical

demand peaked for the month of February. The visualization indicates that demand peaked at 10:45 PM and represents 19,054 kWh of consumption, at a charge of \$2,095. The profile typifies a facility whose electrical usage is dominated by environmental conditioning (HVAC) since the demand peaks around the same time when the sun reaches its zenith. Since this facility is educational, the profile is as expected. There is no indication that electrical usage was irregular. A comparison of the daily demand profile with other days supports this conclusion. Unlike the typical day exemplified in Figure 3.3, Figure 3.4 represents an atypical daily demand profile.

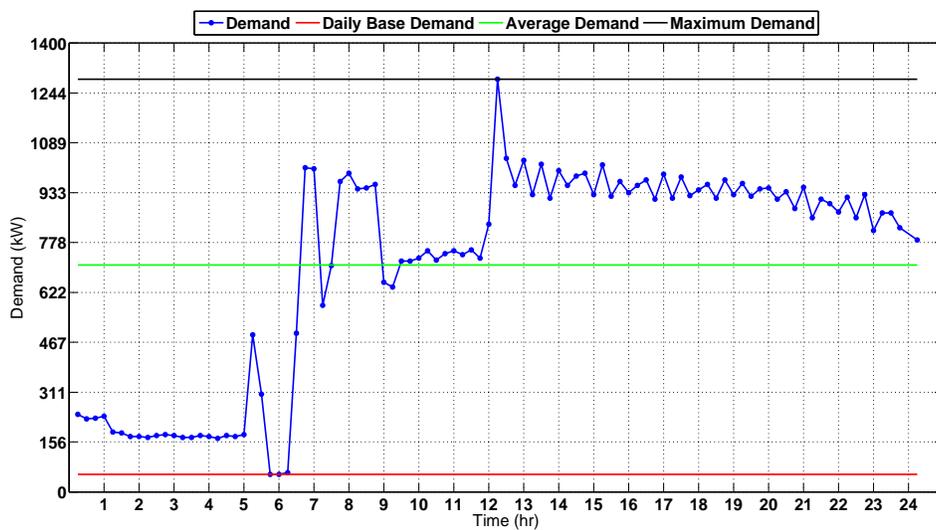


Figure 3.4: Daily demand curve. February 18, 2012.

Since February 18, 2012 represented a Saturday in which the facility should have been closed, this demand profile is unusual in comparison to other Saturdays. Activity between 6 AM and 12 PM is clearly visible. After, activity drops off, but is still elevated more than usual. Activity during this day incurred a consumption of 17,057 kWh at a

charge of \$1,876, a significant change from the previous Saturday whose consumption valued at 4,109 kWh at a charge of \$452.

Weekday

For the purpose of ascertaining if whether operations are consistent on a weekly basis, visualizing demand on a weekly timescale can help to gauge if there are aberrations in facility demand usage. For the most part, one expects for facility electrical usage to be approximately the same given a particular day of the week. Just as aberrations on a daily basis throughout the course of a month can be indicative of unneeded electrical usage, some operations are clearer by viewing demand on a weekly timescale. Electrical creep, in which electrical usage slowly rises can be diagnosed. Additionally, there is some benefit to the facility to be able to quantify how much is being spent on a daily basis throughout the course of the year. Such information could be used to reduce electrical waste and mitigate off-hour operations. Given a five-day per week work schedule, many facilities may find that electrical usage on weekends is not menial, but rather has a significant impact on electrical usage throughout the course of the year. There is also beneficial to know how many of the days per week indicate peak demand days throughout the year. Knowing that the majority of demand peaks are happening on one day, say Wednesday, could help a facility diagnose and remedy peak producing operations, thus reducing demand induced costs.

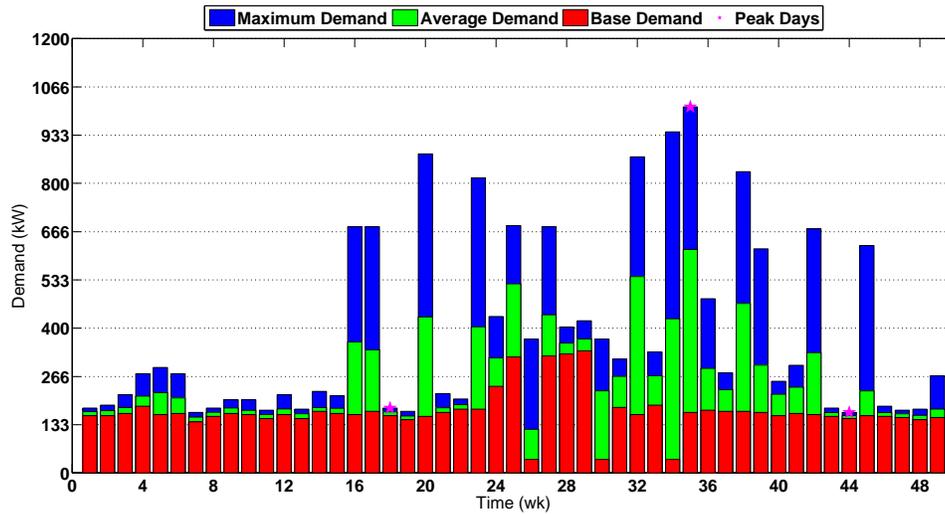


Figure 3.5: Weekly demand usage. Sundays, 2012.

Figure 3.5 shows the maximum (blue), average (green) and daily base demand (red) for all Sundays in 2012. The figure indicated that there are at least 12 out of 49 Sundays (The data set began the last week of January) in which activity was above normal. Additionally, 3 out of 12 demand peaks (indicated by the pink stars) throughout the year could be found on a Sunday. If operations were consistent throughout the year, then the demand peak for a particular month would be random, thus no day should have more than 2 demand peaks. This level of activity on Sundays may indicate an area of concern for facility management.

Annual

When a more refined timescale is inappropriate to determine long-term trends in electrical demand and consumption, a yearly timescale may be appropriate. Using electricity and paying bills on a monthly basis may obscure runaway usage trends were

electrical usage increases over a longer period. Annual visualization of electrical demand and consumption can illuminate usage dependencies such as environmental factors, production increasing and decreasing as well as perhaps, unwarranted systems requiring electrical power.

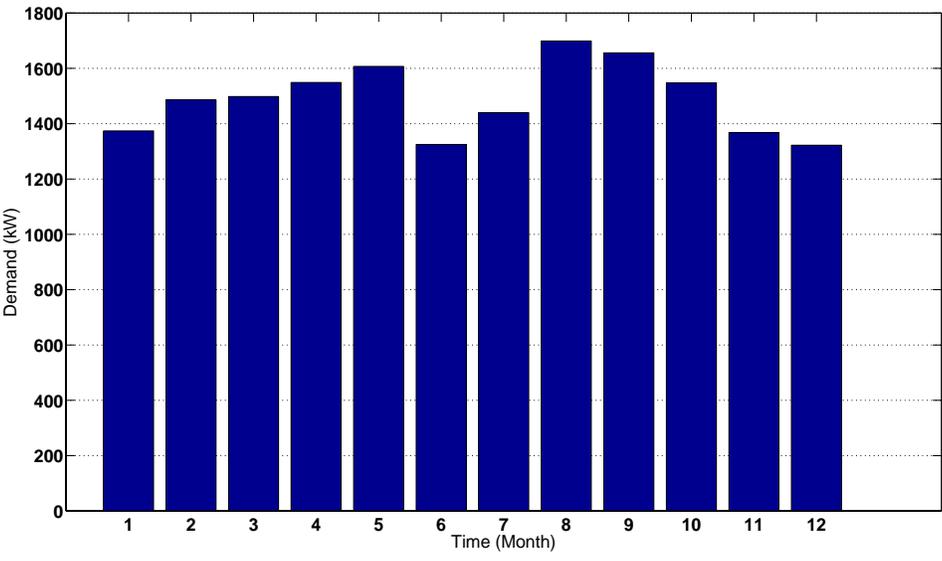


Figure 3.6: Yearly electrical demand. 2012.

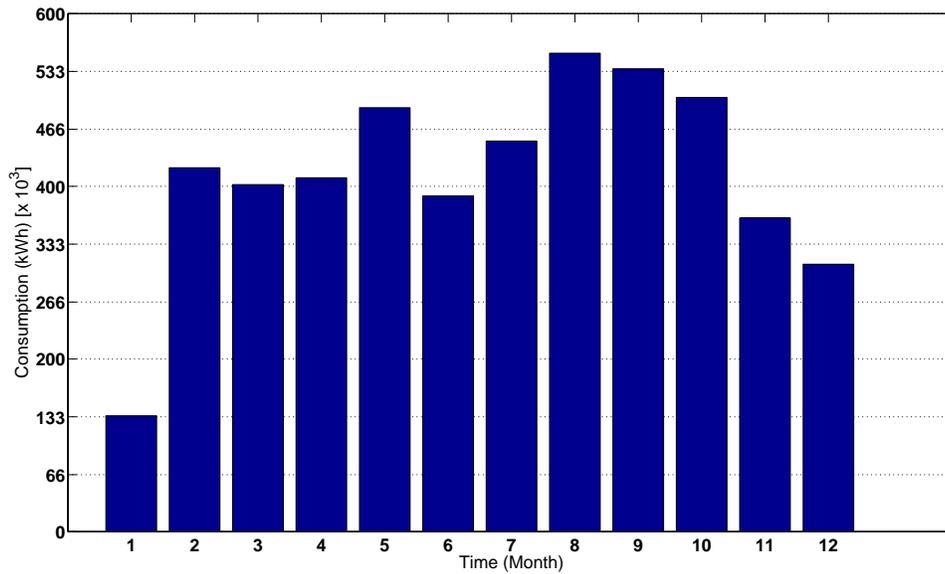


Figure 3.7: Yearly electrical consumption. 2012.

Figure 3.6 and Figure 3.7 represent the electrical demand and consumption for 2012, respectively. The yearly demand profile indicates a facility whose electrical usage is weather dependent; slowly rising from January to August, and dropping from August to December. June and July represent a reduction in electrical demand; this is to be expected since this is an educational facility whose operation decreases for the summer months. The yearly consumption profile tracks a bit differently. If the operating hours and demand remain constant on a monthly basis, then the increases and decreases reflected in the yearly demand profile should also be reflected in the yearly consumption profile. This is the case for most months, except for January. In this case, the electrical consumption is oddly low in comparison to other months with a similar demand peak.

This is normally not a problem since a lower than normal consumption means a reduce consumption charge.

Other than trending how electricity is used throughout the year, realizing demand on a yearly timescale can be used to do deuce plant operational time; the objective is to identify which demand data points represent non-operational times. To do this, two separate models will be used; the first is the fixed time model. The underlying assumption in the fixed time model is that base demand occurs during non-operational hours; Figure 3.2 can be used to demonstrate. During February 2012, the daily base demand fluctuated from 167 kW to 46 kW (also the monthly base demand). Assuming that all non-operational demand point fall within these demand points, the fixed time model counts the number of data point that fall within those demand points. Since every data point represents 15 minutes, the non-operational time predicted by the model would be the product of the number of demand points that fall within those demand points and 15. In the case of February 2012, 101 out of 2,784 data points fell between 46 kW and 167 kW, thus the constant time model predicts that there were 1,515 minutes (25.25 hours) of non-operational time.

The second model used to predict the non-operational hours is the moving boundary model. Similar to the fixed time model, the moving boundary model counts the number of demand points that fall within a boundary and also uses the monthly demand base as one of its boundaries. The difference lies in how the model choses the upper bound. A user attenuation factor, α , is used to shift the upper bound from the average demand; an α value of 1 indicates that all demand values that fall within the monthly

base demand and the monthly average demand are to be considered as non-operational data points. An α value of 0.5 means that all demand values that fall within the monthly base demand and the demand value half way between the monthly average demand and monthly base demand are to be considered as non-operational data points. Thus, the demand points that fall under

$$D_{MB} < D < [D_{MB} + (\alpha)(D_{MA} - D_{MB})]$$

where D_{MB} is the monthly base demand and D_{MA} is the monthly average demand. The benefit of the moving boundary model is that the user is able to change the number of data points that are counted as non-operational through the attenuation factor. Using the demand visualization, the user can adjust the model boundary to encompass non-operational activity as can be seen on the visualized demand curve.

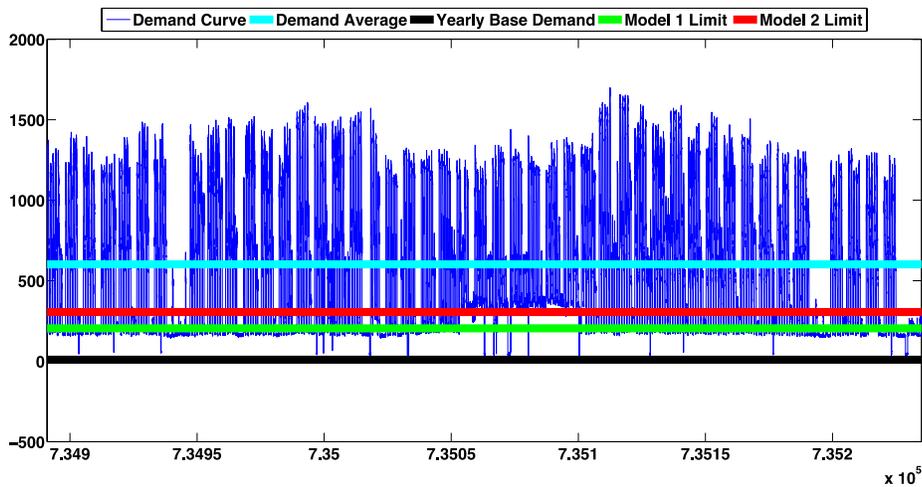


Figure 3.8: Yearly electrical demand curve. 2012.

Figure 3.8 represents the complete, yearly demand curve for 2012. Extrapolating the models used to determine non-operational times from a monthly to yearly timeframe,

the figure shows several boundaries. The lower line (black) indicates the yearly base demand. This is the same for both the fixed time and moving boundary models. The upper bound for the fixed time model is represented as green. 27,096 data points fell between the black and red lines, indicating that there were 2,258 hours of non-operation. The upper bound for the moving boundary model is represented as red. Note that since the attenuation factor, α , was set to 0.5, the boundary is half way between the yearly base demand and the yearly average demand. The model indicated that there were 41,652 data points that fell between the black and green lines, demonstrating that there were 3,471 hours of non-operation. Since the demand curve represents an educational facility, teachers and faculty are contracted to work 187 days (4,488 hours) throughout the year. Thus, it is expected to have 178 days of non-operation (4,272 hours). The fixed time model fell within 52.8% of the expected value and the moving boundary model fell within 81.2% of the expected value.

Demand Aberration

Various avenues are available to lower a facilities electrical demand. One, of which have been spoken of previously, are through equipment retrofits. Industrial processes require a certain amount of power (kW or kVA) and energy (kWh) to meet the demand of customers. Factors such as temperature, time of year, customer demand, etc. all impact electrical consumption and demand. Running the HVAC system during summer entails increased energy usage since the temperature gradient is larger. Manufacturing paper plates show a rise during summer months since the customer demand might increase due to the amenable weather. Electrical demand trends can be

rationalized if one is familiar with the process and facility equipment. However predictable, monthly demand increases reflect avoidable industrial practices. Testing a facility fire suppression system during the afternoon, when daily demand peaks are highest might be tested at the end of the day to decrease demand peaking. If these avoidable practices take place during monthly demand peak times, then preventing such practices can decrease demand and save on capital investment. Avoiding preventable demand peaking as well as unnecessary electrical consumption and demand is worthwhile, but daunting to identify. The goal of the demand profile aberration tool is to identify days in which the demand profile deviates from the “normal” profile. By identifying days that are out of the ordinary, the user can hone in on processes that contribute to demand peaking and avoidable industrial practices.

The tool identifies two demand characteristics that may exemplify abnormal electrical usage; deviation from the average demand profile and deviation from the average peak demand. To identify days which statistically deviate from the average demand profile, a comparison is made between the average demand profile for all days of the week throughout the year and a particular day of the week; Monday demand profile to average Monday demand profile for all Mondays throughout the year, Tuesday demand profile to average Tuesday demand profile for all Tuesdays throughout the year, etc. Demand profiles for particular days that fall out of a statistical range are flagged as days with potential demand or consumption aberrations. To compare one days average demand profile to the average profile, the difference between the 2-norm of the demand profile in question to the average demand profile is taken as

$$P_{i,mag} = ||P_i - P_{AVE}||_2 \quad (v)$$

where $P_{i,mag}$ is the 2-norm of the difference between the demand profile of the day in question to the average demand profile, P_i is the demand profile of the day in question and P_{AVE} is the average demand profile. The 2-norm is taken as the root mean square of the demand profile vector and is a measure of the magnitude, or length of the vector. The 2-norm is defined as

$$||x||_2 = \sqrt{\sum_{i=1}^p n_i^2} \quad (vi)$$

where n is the demand at time interval i .

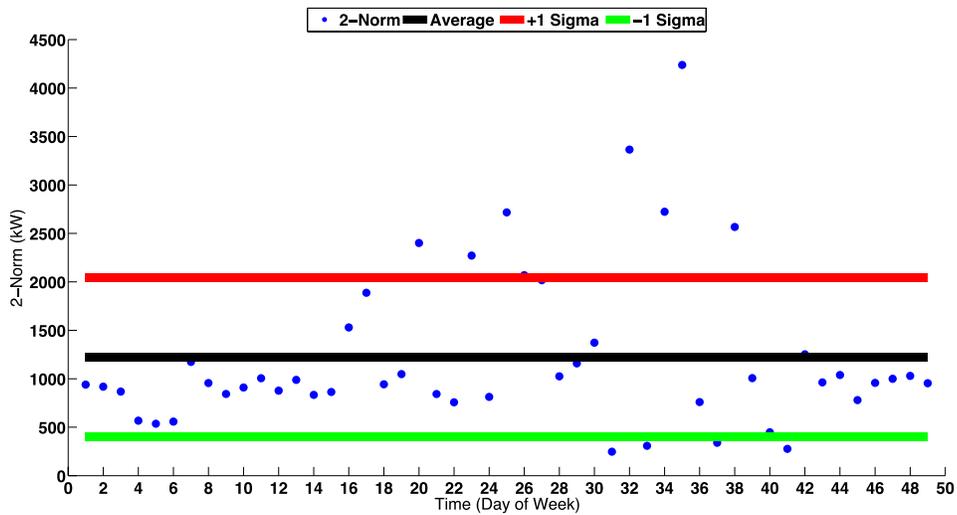


Figure 3.9: 2-Norm demand space. 2012.

Figure 3.9 depicts the 2-norm demand difference space for all Sundays throughout the 2012 electrical demand data set. Assuming that the data set represents a Gaussian distribution, then 68.2 % of all 2-norm values should fall within 1 standard

deviation of the average 2-norm value. Days that fall out of that statistical range are flagged as days with a possible demand profile aberration. In this case, 12 days were identified as being aberrational.

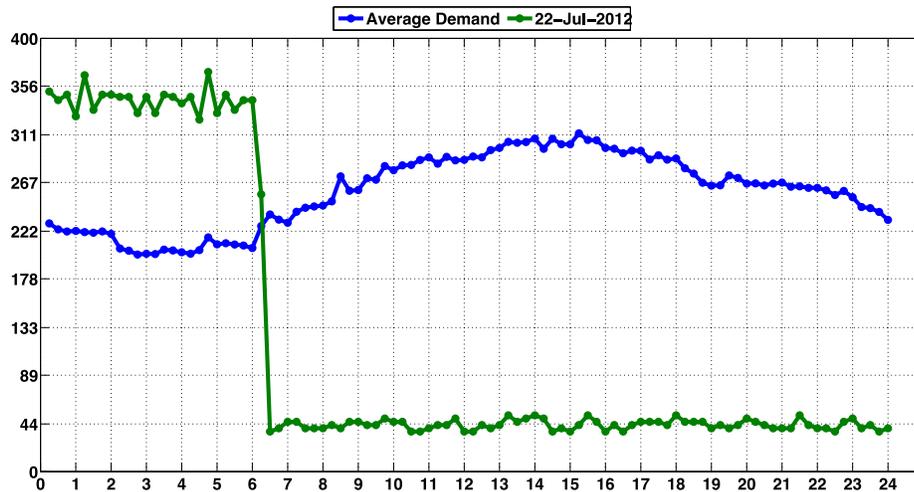


Figure 3.10: Comparison of the demand profile for July 22, 2012 and the average demand profile seen on Sundays.

Figure 3.10 depicts one of the days (green) that fell out of statistical range with the average demand (blue) profile for Sundays. The profile does seem to significantly deviate from the average profile in that there is a drastic demand shift after 6 AM. Since the usage is for July 22nd is only 2,841 kWh in comparison the 6,211 kWh consumed on average, there does not seem to be precedence for inquiry. In addition to screening demand profiles for aberrational usage, the tool also has the capability to screen for aberrational demand peaks.

In much the same manner as was used to distinguish variation of a particular demand profile to the average profile, the same process is used to determine the variation

in demand peaks. To compare one demand peak to the average demand peak for a day in question, the difference between the 1-norm for the day in question, to the 1-norm for the average demand profile is taken as

$$P_{i,\text{mag}} = ||P_i - P_{\text{AVE}}||_1 \quad (\text{vii})$$

The 1-norm is defined as

$$||x||_1 = \sum_{i=1}^p n_i^2 \quad (\text{viii})$$

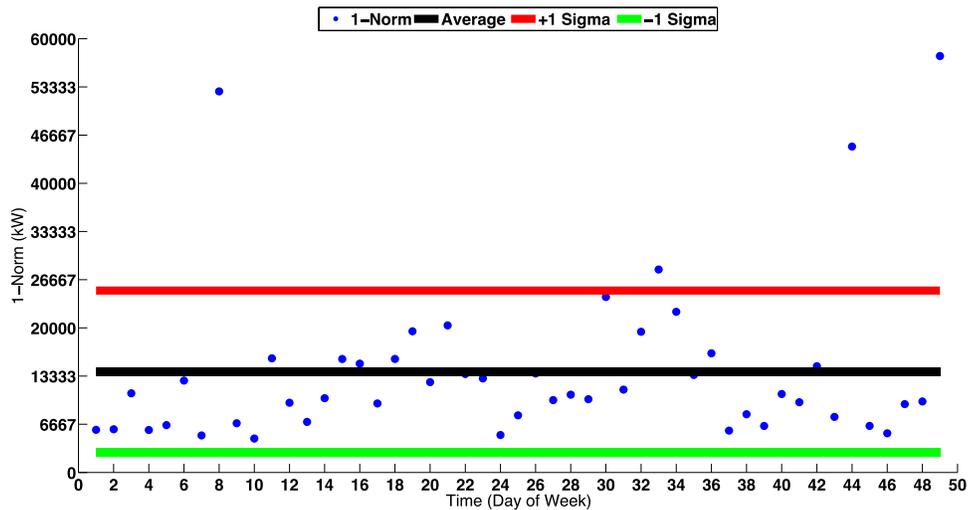


Figure 3.11: 1-Norm demand space. 2012.

Figure 3.11 depicts the 1-norm demand difference space for all Tuesdays throughout the 2012 electrical demand data set. Days that fall out of the standard deviation are flagged as days with possible aberrations in demand peaks. Unlike the concern with the overall demand profile and how it represents unwarranted electrical

consumption, the concern with the demand peak results from its contribution to the monthly peak.

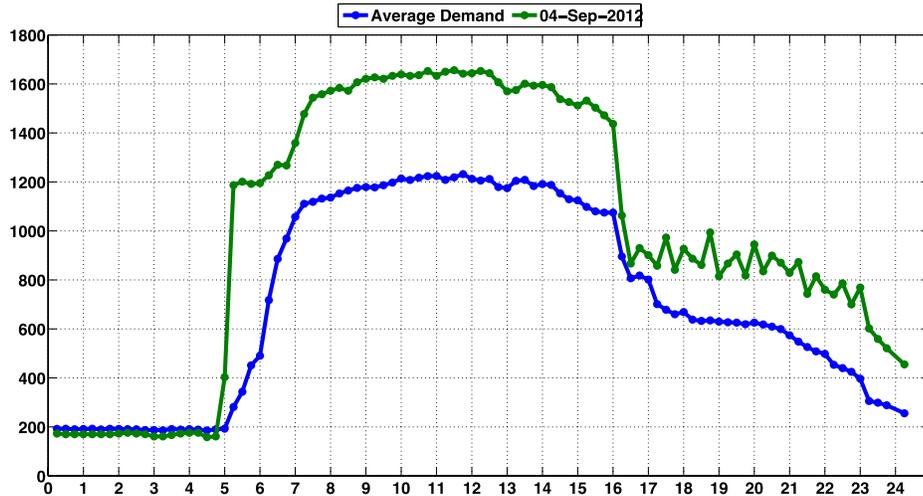


Figure 3.12: Comparison of the demand profile for September 04, 2012 and the average demand profile seen on Sundays.

Figure 3.12 depicts one of the days that fell out of statistical range with the average demand peak for all Tuesdays. In this case, we notice that the profile is very similar to the average, yet there is a larger power draw. This day represents the largest demand peak for September, prompting an investigation to help mitigate the abnormal power draw. The average demand peak for all Tuesdays throughout 2012 is 724 kW. This particular Tuesday represents a 424 kW jump.

Demand Scheduling

ECMs are made across an array of platforms; many are designed to reduce electrical consumption and demand. An equipment retrofit such as changing out lights

and motors for ones with increased efficiency can reduce demand by mitigating the power requirement used in their operation. Some recommendations come in the form of recommending changes to production to increase fabrication efficiency; adding an automated conveyor system to handle product instead of using manpower. Other than retrofitting or turning off industrial equipment, demand can be affected by shifting production to off demand peak hours of operation; this is known as demand scheduling.; the more equipment that is running during the demand peak time, the larger the demand since electricity is required to power these devices. While it may not be feasible to reduce the usage of such equipment, it may be possible to shift when the equipment is operated, thus operating the equipment at other times throughout the facility hours of operation can reduce peak demand and save on the charge associated with demand. It should be mentioned that demand scheduling saves on the charge associated with demand and not consumption, since the total amount of energy used is not affected.

Scheduling how equipment is used throughout the day is largely based on the savings associated with shifting equipment operational hours. The savings associated with demand scheduling may be significant, but when compared with the additional cost associated with keeping essential facility equipment operational (lights, compressors, etc.) as well as the added cost of employing personnel for extended operational hours, demand scheduling may or may not be amenable to energy savings.

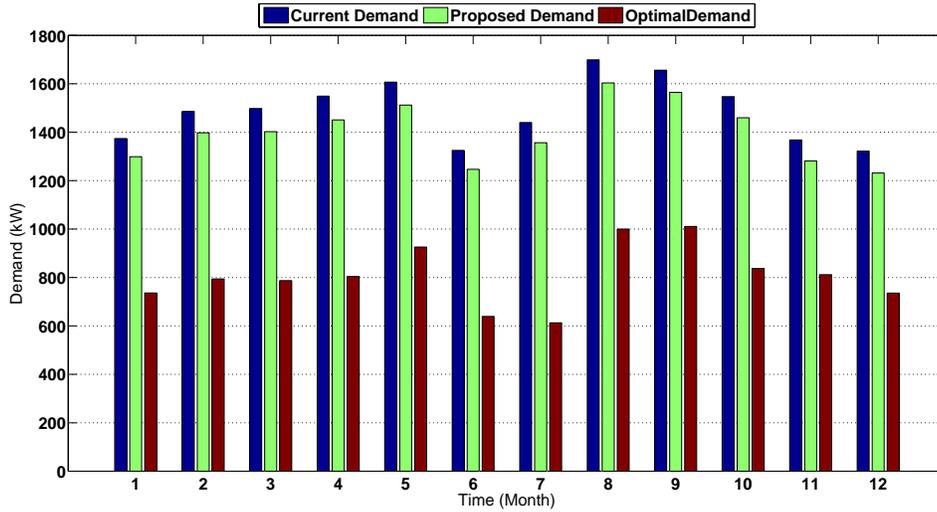


Figure 3.13: Demand savings associated with implementing an electrical demand scheduling scheme. 2012.

Since there are a variety of factors that impact how a facility consumes power, there are synonymous prescriptions for what a facility is feasibly able to do to reduce demand peaking. Figure 3.13 represents two different demand reduction scenarios. The first scenario (green) depicts what a facility can expect to use in electrical demand if they are able to reduce their demand usage by some percentage. The following was used to determine the proposed demand:

$$P_p = P - \beta(P - P_A) \quad (ix)$$

Where P_p is the proposed demand, P is the actual demand, P_A is the average demand and β is the proposed percent reduction; in this case, a 10% (0.10) reduction was used. If this facility were able to reduce their monthly demand peak by 10%, the yearly savings would total \$8,513. The optimal (red), or best case scenario for reducing monthly demand peaks is when a facility is able to run a 24 hour operation with the same

consumption. In this case, the demand is spread over a larger operating time, thus reducing the demand requirement of the facility. The following was used to determine the optimal demand

$$P_{OPT} = \frac{\int_{t_2}^{t_1} P dt}{\Delta t} \quad (x)$$

where P_{OPT} is the optimal demand and Δt is the time unit in one day. In this scenario, the facility is able to save \$65,410 in electrical demand costs by extending their operating period to a 24-hour operation. Since this is an educational facility, demand scheduling over a 24-hour period is an unlikely solution to reducing electrical demand.

Weather Disaggregation

In practice, there are two ways in which equipment power is gaged; the first is through sub metering. Sensors of various types are interfaced with industrial equipment. The data is gathered and logged for analysis at a later date. If plant personnel desire to know how much electrical power motors use, then an amperage or voltage transducers are placed on each individual motor and left over a period of time; data is gathered and analyzed at a further date. This process can be costly since the number of sensors required is equivalent to the number of motors analyzed. Many industrial facilities operate hundreds of motors; at a price of \$100 per sensor, the study of equipment electrical usage via the process of sub metering can cost thousands, if not hundreds of thousands of dollars. The second manner to gauge how much electricity various industrial equipment uses is to guess. In many cases the power requirements for industrial equipment can be found on nameplates stuck to the equipment. Using the

required voltage and upper limit current draw, the engineer can easily calculate the maximum power draw for the equipment. Using load and duty attenuation factors, the engineer is able to hone in on the possible electrical usage. The process has its drawbacks, since it usually requires knowledge of how the equipment is being used. The process also takes time since, in many cases, equipment has to be tracked down. The weather disaggregation tool is an attempt to disaggregate the load from equipment whose electrical usage is based on variations in outdoor temperature; for the most part, this means the use of HVAC equipment. Without sub metering or guessing, the process uses demand data in conjunction with archival weather data to determine the demand and consumption of temperature variant equipment.

Assuming that any heating done is not electrical, it goes without saying that the use of HVAC during summer months is usually larger than that of winter months; since it takes more energy to maintain an indoor temperature given that the outdoor temperature is high, thus the electrical usage during warmer days will be larger than cooler ones. The model used for weather disaggregation is based on the idea that electrical usage for days that are warmer will be higher than for days that are cooler. For any time interval throughout the year, there is a corresponding demand and temperature. The higher the temperature, the larger the demand; the lower the temperature, the lower the demand. Thus, there is a relationship between demand and temperature.

$$P = f(T, t) \tag{xi}$$

where T is the temperature. Some of those electrical demand points lie above a temperature threshold and some lie below, thus

$$P_i = f(T_i, t) \quad (\text{xii})$$

where

$$T_i > \bar{T} \quad (\text{xiii})$$

and

$$P_j = f(T_j, t) \quad (\text{xiv})$$

where

$$T_j < \bar{T} \quad (\text{xv})$$

where \bar{T} is the temperature threshold, P_i is the demand when the temperature is above the temperature threshold for any given time, and P_j is the demand when the temperature below the temperature threshold for any given time. Disaggregation is accomplished by taking the difference of the averages of the two demand profiles, P_i and P_j .

$$\Delta D = \frac{\sum_{i=1}^{n_1} P_i}{n_1} - \frac{\sum_{j=1}^{n_2} P_j}{n_2} \quad (\text{xvi})$$

where n_1 is the number of demand data points that fall above the temperature threshold \bar{T} and n_2 is the number of demand data points that fall below the temperature threshold \bar{T} . ΔD is the average difference that falls above the temperature threshold and below the threshold.

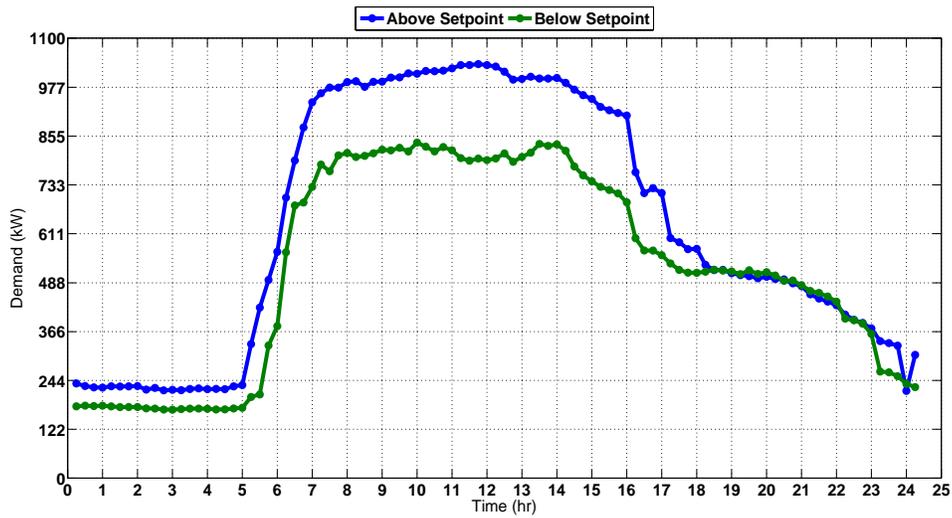


Figure 3.14: Disaggregated demand curves P_1 and P_j , 2012.

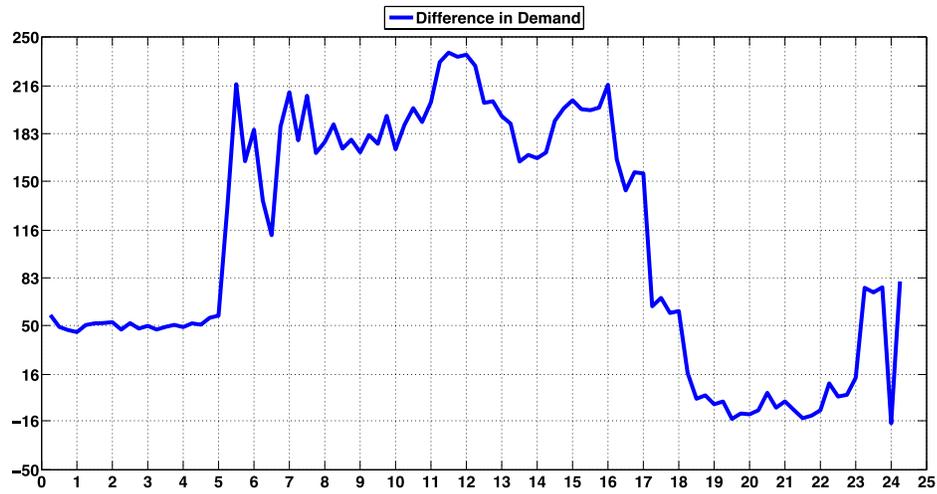


Figure 3.15: Difference, ΔD , of the demand curves P_1 and P_j , 2012.

Figure 3.14 represents the weather disaggregated demand curves. The upper demand curve (blue) represents the average demand for the aggregated demand points

when the temperature was above a 70°F threshold while the lower curve (green) represents the average demand for the aggregated demand points when the temperature was below a 70°F threshold. Figure 3.15 shows the difference between the two curves. The morning hours show that there is not much a difference (only around 50 kW) in demand when the outside air temperature is warmer. Most of the difference can be seen during the hours of operation, 6 AM to 5 PM, when its not only hotter outside, but the heat load associated with people is greatest. In this case, one can see that, on average, the demand is around 183 kW greater for days that are warmer, than cooler ones. The model predicts that, on average, this facility spends 2,599 kWh (area underneath Figure 3.15) per day on space conditioning at an average daily cost of \$285.

Photovoltaic (PV) Analysis

Using IRD meter data to diagnose possible faults in electrical demand and usage has been the focus of many of the tools described. Demand visualization, demand aberration and demand scheduling can help to diagnose pre visit ECMs. Weather disaggregation and the modeling to predict hours of operation aid in establishing unknown variables pertinent to the ECM process. The photovoltaic analysis (PVA) tool takes IDR meter data beyond fault diagnosis and into system design. Coupling the solar position model with TMY3 (Typical Meteorological Year, Version 3) data, the PVA tool attempts to predict the electrical yield from solar power. Comparing this yield with the power requirement of a facility, the model is able to size a PV system such that the electrical needs of the facility are met as optimally as possible.

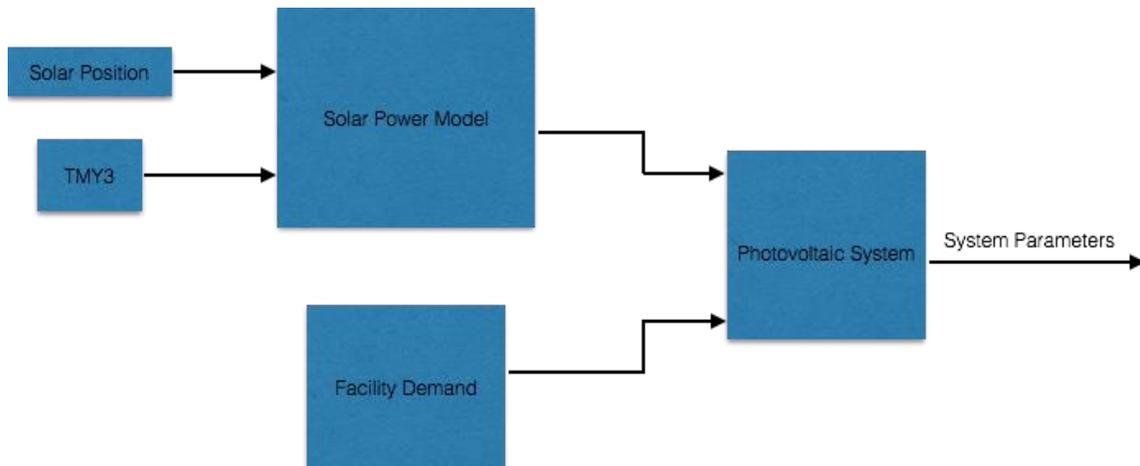


Figure 3.16: Scheme of information flow to predict PV system size.

Figure 3.16 depicts the scheme implemented by the PVA tool to predict the PV system size necessary to meet the electrical needs of a facility. The analysis begins by predicting the position of the Sun.

Solar Dynamics

The procession of the sun throughout the year can be complicated and intricate. Since the Earth is tilted about its axis, the Sun's rays come in at an angle. This angle changes throughout the year as the Earth runs its yearly course around the Sun. The same mechanism that induces the seasons of the year causes this angle to change.

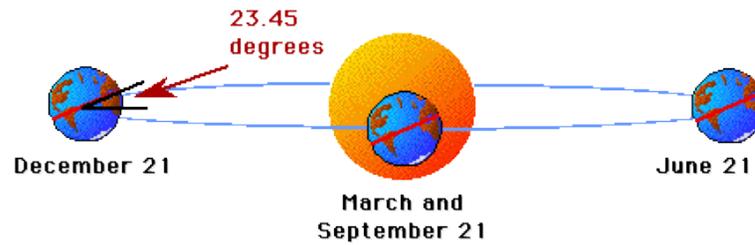


Figure 3.17: Change of declination angle through solar procession. [4]

From Figure 3.17, one can see how the angle is changing throughout the year. During December and June, the angle between the ray's of the Sun and the Earth's equatorial plane reach a maximum absolute value during December and June. This angle, the line drawn between the ray's of the Sun and the Earth's equatorial plane is known as the declination angle (δ), show in detail in Figure 3.18. It is in light of this angle that the sun rises at a particular angle throughout the day.

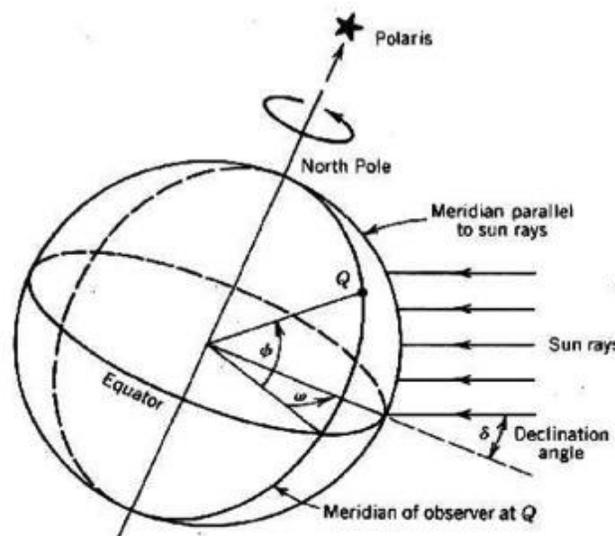


Figure 3.18: Relation of earths equatorial plane to Sun. [6]

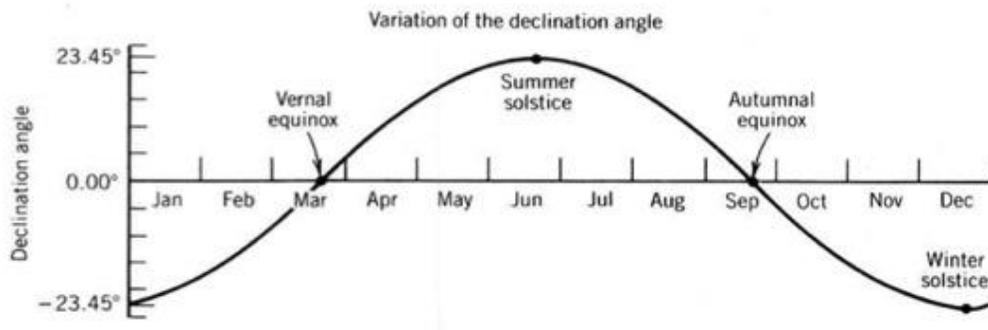


Figure 3.19: Declination angle as a function of the months of the year. [6]

Figure 3.19 shows the declination angle's periodicity. At any position on the Earth's equator, the angle the Sun makes, relative to the Earth's equator reaches a minimum on December 21st and a maximum on June 21st. The positions of the Earth during these times are also known as the summer and winter solstice. When the declination angle is 0° , the Earth's equator is parallel to the ray's of the Sun; these are known as the vernal and autumnal equinox's.

The way one typically characterizes the procession of the Sun is through a conventional time frame. The Sun is typically highest in the sky at 12 noon. It rises in the east at approximately 6 o'clock in the morning and sets in the west at approximately 6 o'clock in the afternoon. The Earth has a 24-hour cycle; 12 hours of daylight and 12 hours of night. If the Sun rises in the east and sets in the west, then throughout a 12-hour period, it moves 180° . If it does this in 12 hours, then for every hour that passes, the Sun has moves 15 degrees. When the Sun is directly overhead, it has moved 90° and we say that it is 12 o'clock in the afternoon (solar time). From figure 3.18, the angle between the two lines on the equatorial plane (the plane running through the Earth's equator) that

intersect the latitude (lines running from magnetic north to magnetic south) of the Sun's current position and the latitude of the observers position is the hour angle (ω). It is the angle used to exemplify the time of day. From Figure 3.18, one can see that when the sun is rising in the west, at approximately 6 o'clock in the morning, the hour angle is 0. At 10 o'clock, the hour angle is 60° since for every hour that passes, the sun is 15° further than it was previously.

The latitude angle (φ) is the angle of the line drawn between a point on the Earth's surface and a line drawn on the equatorial plane, as shown in Figure 3.18. This angle becomes necessary as all points on the Earth's surface have to be related to a reference point where solar dynamics are known, namely the Earth's equator.

As is with any object, position is characterized in a 3 dimensional space. One may use Cartesian or polar coordinates. Since the Sun moves about the sky in a harmonic fashion, it becomes convenient to describe the Sun's motion in polar terms.

The solar altitude (α) is defined as the angle between the Sun's ray and the horizontal plane containing the observer. For an observer on Earth, this would be the angle between the parallel ground and the sun. The solar zenith (σ_z) is the angle between the zenith and the Sun's ray. From Figure 3.20 one can see that this is merely

$$\theta_z = 90^\circ - \alpha \quad (\text{xvii})$$

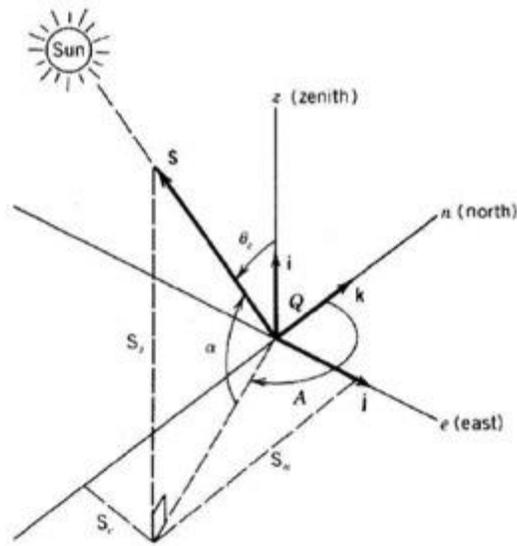


Figure 3.20: Relation of objects on Earth's surface to Sun. [6]

If one were to take their right hand, point it upward and then their left hand and point it towards the Sun, the angle between their right and left hand would be the solar zenith. The solar azimuth (A) is the angle between the north axis and the line projecting the Sun's ray onto the horizontal plane. Visualizing the Sun's motion throughout the day, one can see that in the morning, the solar azimuth is approximately 90 degrees; the altitude is almost zero. As the day progresses, the azimuth will increase by 15 degrees every hour. At 12 o'clock, the azimuth will be approximately 180 degrees and the altitude will reach its maximum throughout the day. This will change daily and throughout the year. At Sunset, again the altitude will reach a minimum and the azimuth is approximately 27°. We now characterize a coordinate system relative to the Earth's equator as

$$S = S_z \hat{i} + S_e \hat{j} + S_n \hat{k} \quad (\text{xviii})$$

Where

$$S_z = \sin \alpha \quad (\text{xix})$$

$$S_e = \cos \alpha \sin A \quad (\text{xx})$$

$$S_n = \cos \alpha \cos A \quad (\text{xxi})$$

Up until now, we have defined two different coordinate systems. The first is in relation to the Earth's equator and is characterized by ϕ , ω and δ . The second is in relation to a point on the Earth surface and is characterized by α , θ_z and A . Since we use the Earth's equatorial region as a reference point, to understand how the Sun moves throughout the day is a matter of relating the coordinate system for a point on the Earth's surface and the coordinate system for the Earth equator.

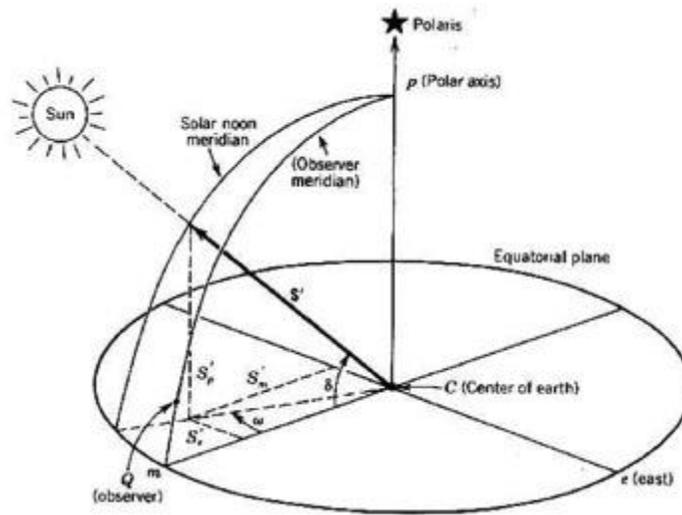


Figure 3.21: Relationship between coordinate systems. [6]

Figure 3.21 shows that a point on the Earth's surface where an observer resides can be related to the Earth's equatorial plane. From this we correlate the two coordinate systems as

$$\begin{bmatrix} S_z \\ S_e \\ S_n \end{bmatrix} = \begin{bmatrix} \cos \varphi & 0 & \sin \varphi \\ 0 & 1 & 0 \\ -\sin \varphi & 0 & \cos \varphi \end{bmatrix} \begin{bmatrix} \acute{S}_m \\ \acute{S}_e \\ \acute{S}_p \end{bmatrix} \quad (\text{xxii})$$

where the vectors associated with an equatorial based coordinate system are

$$\acute{S} = \acute{S}_m \hat{i} + \acute{S}_e \hat{j} + \acute{S}_p \hat{k} \quad (\text{xxiii})$$

and

$$\acute{S}_m = \cos \delta \cos \omega \quad (\text{xxiv})$$

$$\acute{S}_e = \cos \delta \sin \omega \quad (\text{xxv})$$

$$\acute{S}_p = \sin \delta \quad (\text{xxvi})$$

Austin, Texas resides at 30.26° north latitude and 97.74° west longitude, meaning that if one were to draw a line from Austin to the center of the Earth, the angle between that line and the Earth's equatorial plane would be 30.26° ; this is the latitude angle (φ). The longitude angle is the angle of the lines on the equatorial plane intersecting two latitudes; the first is the latitude intersecting Austin and the second is the prime meridian, defined to be angle 0° and passing through the Royal Observatory in Greenwich, England. As mentioned previously, solar time can be represented as an angle. If for every hour that passes, the Sun progresses 15° , then the following represents the correlation between solar time (t_s) and the hour angle (ω).

$$\omega = 15(t_s - 12) \quad (\text{xxvii})$$

Solar Radiation on Earth

The maximum amount of power coming from the sun, hitting the Earth is approximately $1,367 \frac{W}{m^2}$ and this is the maximum amount of power one could hope to harness outside the Earth's atmosphere. Unfortunately (or perhaps fortunately) for those living on the surface, not all of that energy reaches the surface of the Earth. As that energy passes through the atmosphere, atmospheric gases absorb much of it. In turn, the gases re-emit that energy; some of it goes back to outer space and some is emitted to the surface. This is the main reason we experience subtle shades of red during sunset. Atmospheric gasses absorb light radiation from the sun and re-emit at different wavelength. In addition to atmospheric gases, cloud cover imposes a major factor of the amount of radiation incident of the surface. From both cloud cover and atmospheric disturbances, 55 to 83 percent of the energy outside the atmosphere reaches the surface.

From Solar Power to Electricity

The Liu Jordan, isotropic sky model is used to derive solar insolation (solar power reaching the earths surface.) on a flat plate, tilted, photovoltaic panel.

$$I_T = [(I_B)(R_B)] + \left[(I_D) \left(\frac{1+\cos\beta}{2} \right) \right] + \left[(I)(\rho_G) \left(\frac{1-\cos\beta}{2} \right) \right] \quad (\text{xxviii})$$

where I_B is the solar beam radiation on a horizontal, flat plate, R_B is the flat plate tilt factor, I_D is the diffuse solar radiation on a horizontal, flat plate, β is the tilt angle of the flat plate, solar collector, I is the total solar radiation on a horizontal, flat plate and ρ_G is the ground reflectivity. The optimal angle for a flat plate PV panel is equivalent to its azimuth position, derived above. I_B , I_D and I are determined from TMY3 solar data. The

tilt factor, R_B , is a function of the position of the sun relative to the angular position of the flat plate collector and is characterized as

$$R_B = \sin[(\alpha)(\cos \beta) + (\cos \alpha)(\sin \beta)(\cos[180 - A])] \quad (\text{xxix})$$

Determining the rate of derivable power becomes a matter of applying a PV collector efficiency (usually between 10% and 18%) to the solar insolation determined by the Liu Jordan model.

After comparing the amount of power generated through PV to the facility electrical demand and consumption needs, determining the cost saving associated with installing a PV system becomes a matter of sizing the system; every kW generated during facility demand peak time and every kWh produced over the course of the year translates to energy that is not required from the electrical provider.

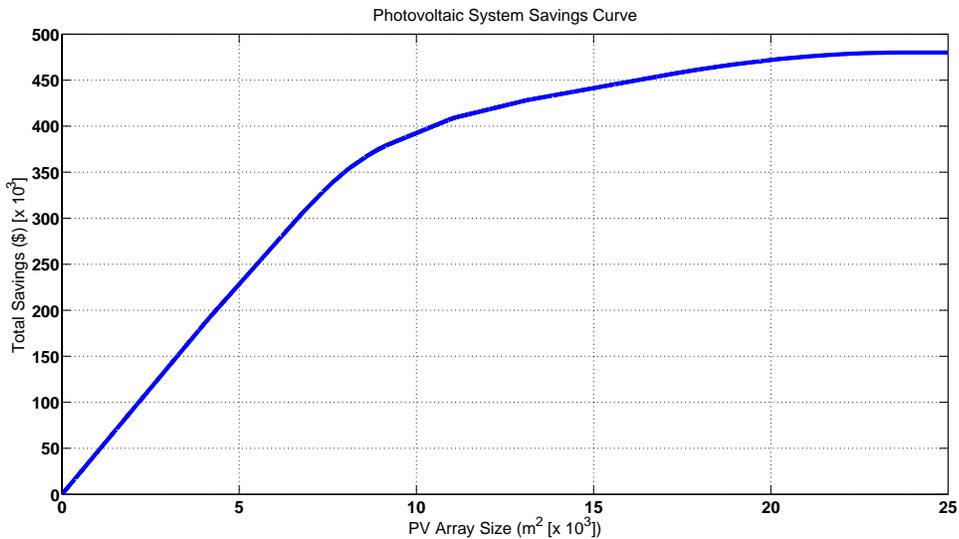


Figure 3.22: Capital savings curve associated with installing a PV system of different sizes.

Figure 3.22 depicts the savings associated with installing a PV system. Since the TMY3 data is normalized over an area, the amount power and energy to be derived through such a system is also normalized in the same manner. The figure entails two distinct curves. The first can be seen from 0 to 10,000 m² sized system. These savings entail both demand and consumption savings. The second curve can be seen from 10,000 m² to 25,000 m². Demand savings become exhausted (a facility cannot save more than is needed in electrical demand) and electrical savings become dominated by consumption savings. The figure predicts that a 25,000 m² sized system would be required generate enough energy cost savings to outweigh the cost associated with demanding electricity from a provider.

PV system costs are characterized as an installed cost. As of 2014, a typical installed cost for an industrial system is approximately \$4,600 per kW (\$4.60 per Watt). Determining the total cost of the system becomes a matter of finding the product of the installed cost $\left(\frac{\$}{\text{kW}}\right)$ and the power output of the system (kW). Since the solar data has been normalized per unit area and to be conservative with the payback period, the system cost was determined by taking in product of the installed cost $\left(\frac{\$}{\text{kW}}\right)$, the size of the system (m²) and the maximum amount of derivable power over the course of a month $\left(\frac{\text{kW}}{\text{m}^2}\right)$.

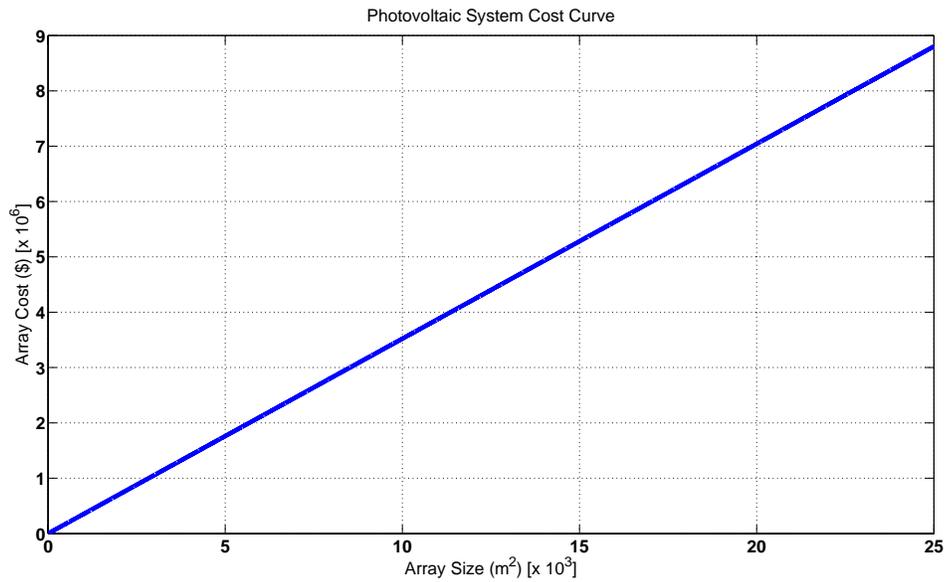


Figure 3.23: Cost associated with installing a PV system of different sizes.

Figure 3.23 depicts the total system cost as a function of the array size. To nullify the cost associated with consuming electricity from a provider, a 25,000 m^2 system would cost close to \$9,000,000 installed. The simple payback is the quotient of the system cost to the system savings.

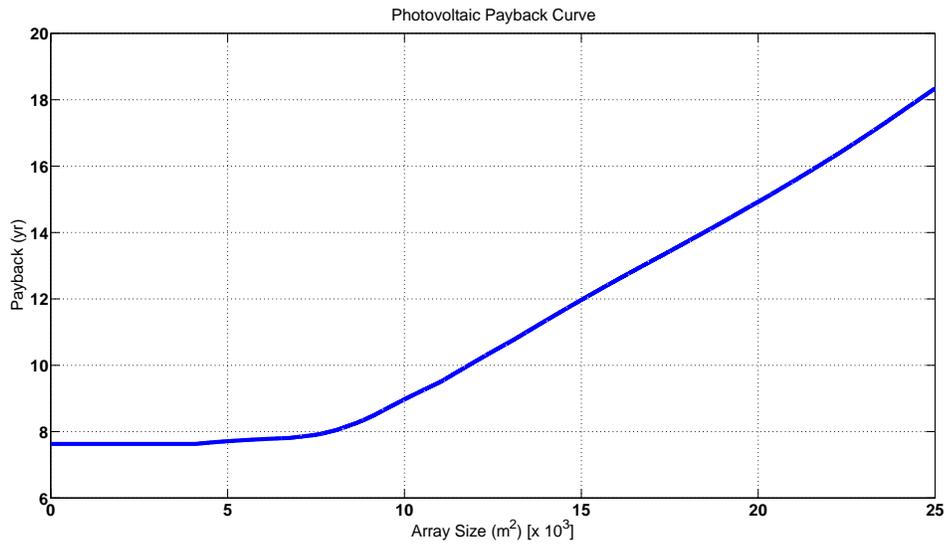


Figure 3.24: Payback period associated with installing a PV system of different sizes.

Figure 3.24 depicts the payback period for a PV system with varying array sizes. A 25,000 m² system would entail a simple payback period of a little more than 18 years, consistent with what one would expect from a PV system. After determining the type of system desired, the tool attempts to predict electrical yield over the course of a year.

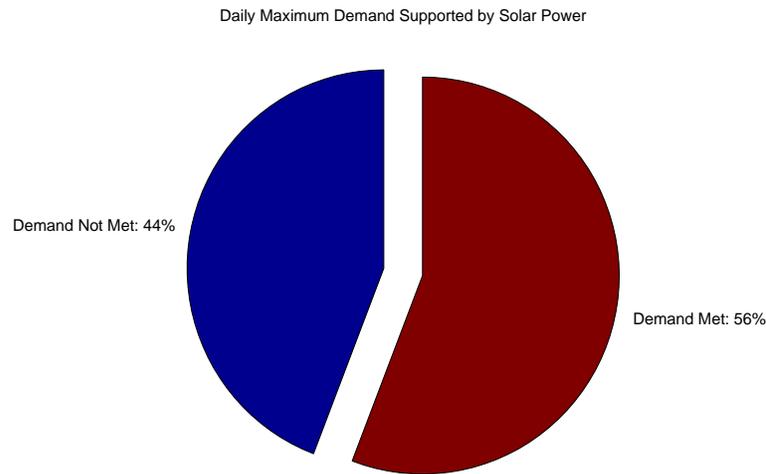


Figure 3.25: Pie chart of percent time facility electrical demand needs are met and unmet by PV system.

Figure 3.25 depicts a pie chart of when the facility can expect their electrical demand requirements to be met through solar power and the percent time the facility can expect for solar power to be inadequate. In this case, a 25,000 m² system would meet facility demand requirements 56% of the time. If the facility requires electricity throughout 24 hours of the day, and the sun is only shining approximately 12 hours per day, then meeting demand over 50% of the time is adequate. In this case, a 25,000 m² system is more than adequate.

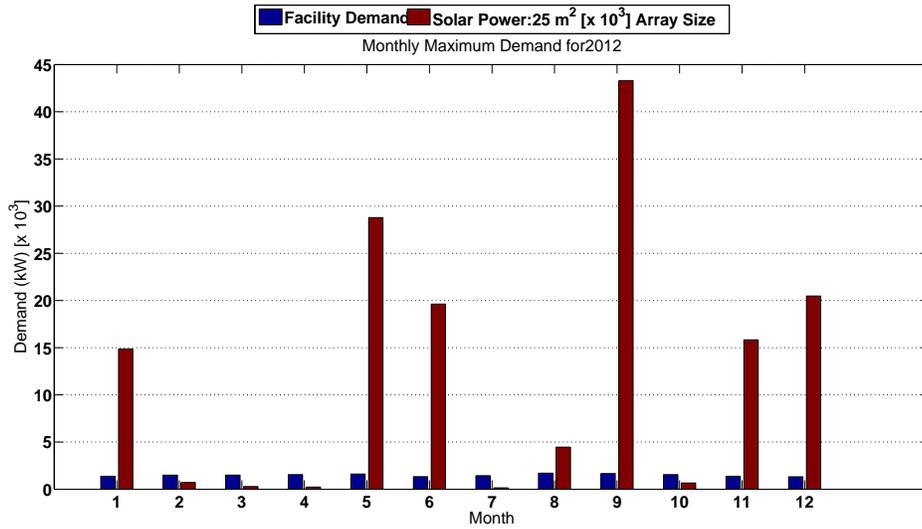


Figure 3.26: Facility demand met by solar power on a monthly basis.

Figure 3.26 juxtaposes the monthly maximum demand required by the facility against the expected power from a PV system at the same time as the facility demand peak. The selected PV system (25,000 m²) at a panel efficiency of 15% will meet facility needs 7 out of 12 months of the year. In this case, the facility will either require a larger system or be grid tied to meet the demand requirements throughout the year.

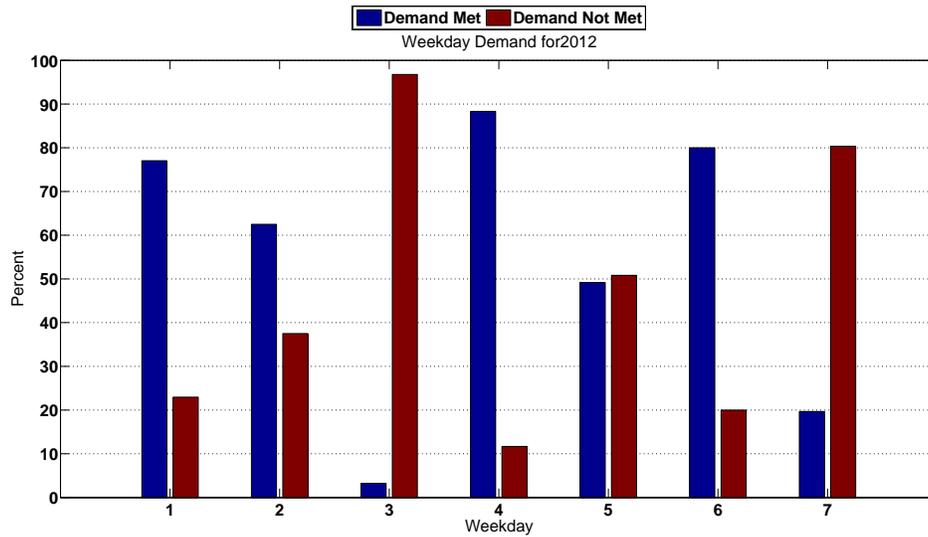


Figure 3.27: Percent of demand met by solar power on a day per week basis.

Figure 3.27 shows the percent of electrical demand to be met and unmet on a day per week basis. Since a PV system produces electricity only 50% of the day, we are interested in days in which the unmet demand exceeds 50%. In this case, there will be an issue in Saturdays and Tuesdays. The chosen PV system should be adequate for most every day of the week throughout the year.

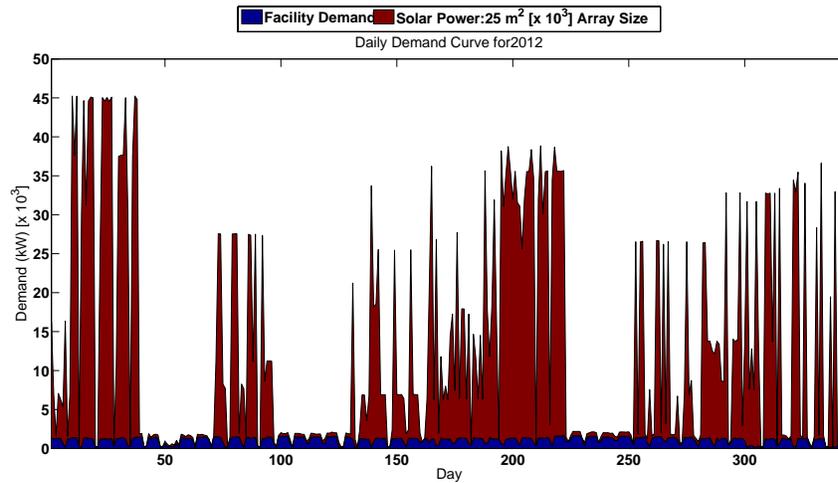


Figure 3.28: Comparison of demand produced through PV and facility demand for every day of the year. 2012.

Figure 3.28 shows what the expected demand yield should be for a 25,000 m² PV system as compared to facility demand needs. For most of the year, the figure depicts a system whose output is more than adequate, producing peak electrical power at a value much larger than required by the facility. The resolution of the figure is insufficient to depict the inadequacy of the system to produce the required electrical demand during nighttime, when the sun is not shining. The analysis does not take into account storing electricity, only using it as it is produced. Also of relevance to a facility is the electrical usage produced by the system.

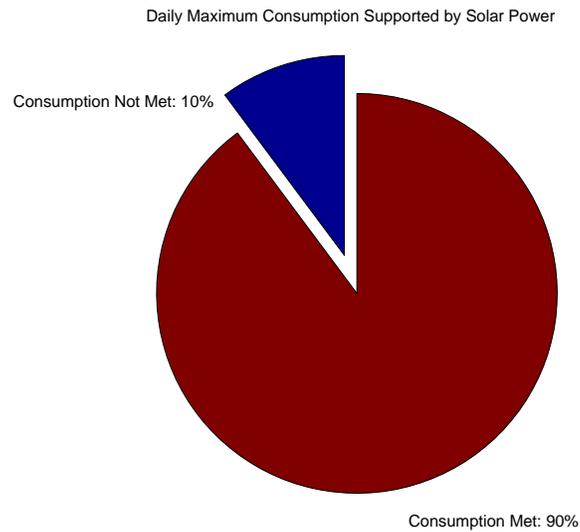


Figure 3.29: Percent time facility electrical consumption needs are met and unmet by PV system.

Figure 3.29 represents the adequacy of a 25,000 m² PV system to meet the consumption needs of the facility throughout the year. Facility consumption and consumption produced by the PV system was compared on a daily basis, so even though the facility required electrical consumption at night, when the system was not providing electricity, the consumption was still considered as met since the total electrical consumption for that day, produced by the PV system was larger than the consumption needs of the facility. This manner of considering electrical consumption as being met, on a daily basis is akin to considering that the electricity be stored, for later use. This may, or may not be the case, since this facility will, more than likely need to choose to be tied into the grid to meet its electrical demand requirements.

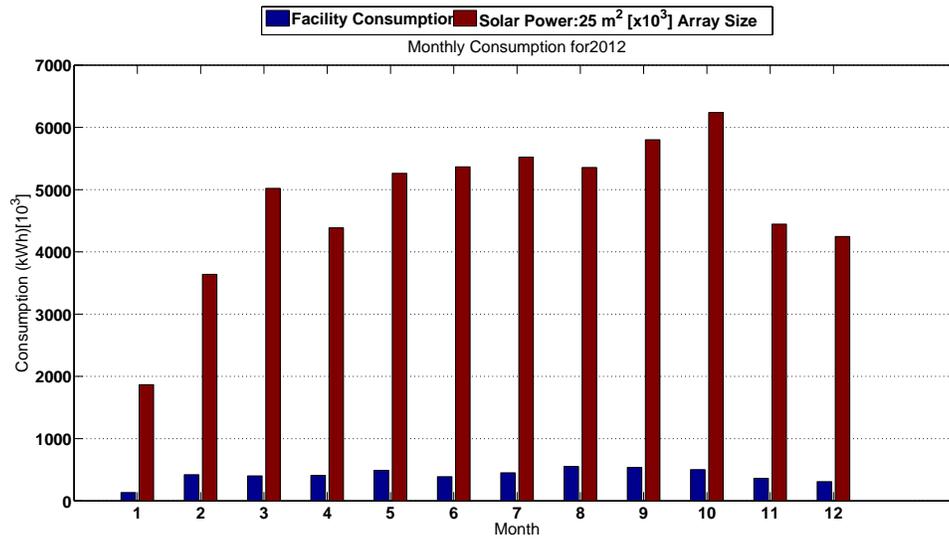


Figure 3.30: Facility consumption met by solar power on a monthly basis.

Figure 3.30 juxtaposes the facility monthly consumption and the consumption produced by the 25,000 m² PV system. Assuming that the electricity is stored, the system is more than adequate to meet the consumption needs of the facility on a monthly basis.

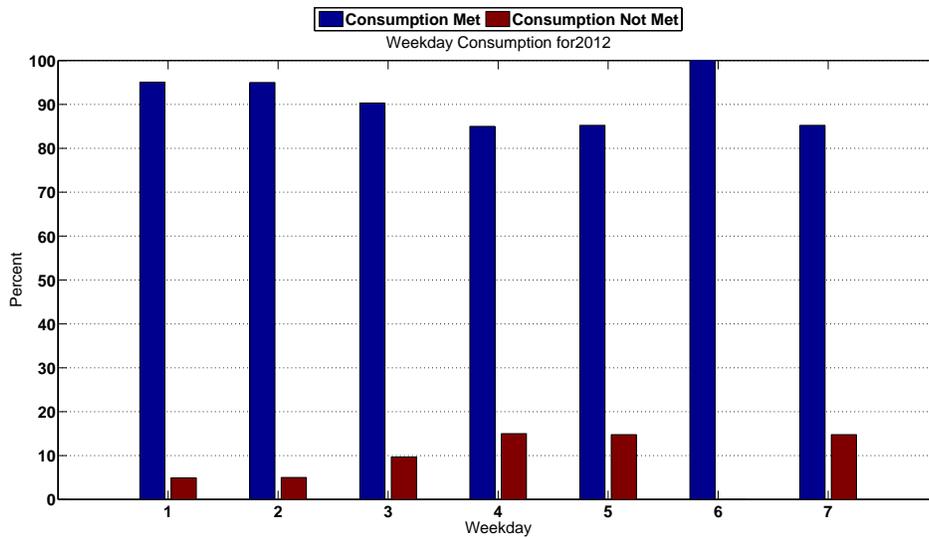


Figure 3.31: Percent of consumption met by solar power on a day per week basis.

Figure 3.31 shows what can be expected, in terms of the percent consumption met and unmet by the selected system. Most days of the week hover around 90% of their electrical consumption needs being met by the system. To make up the difference, this system would have to be grid tied.

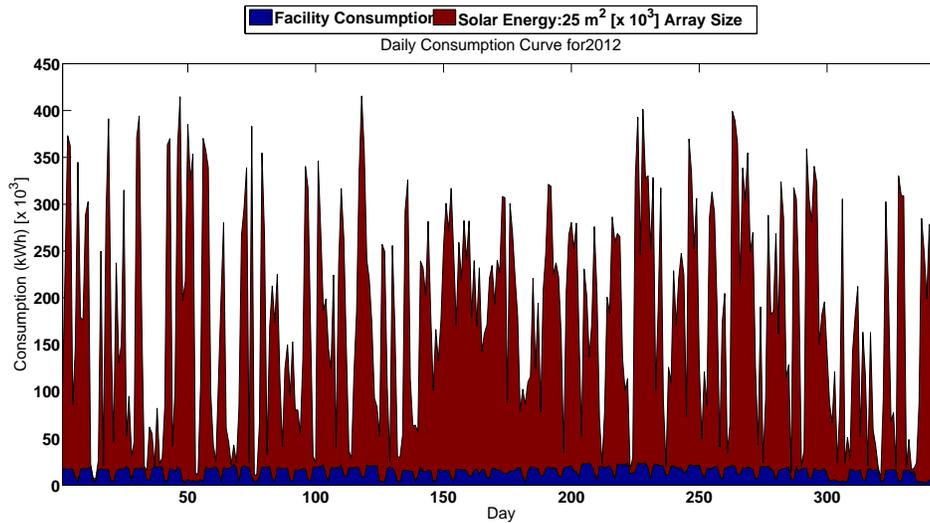


Figure 3.32: Consumption produced through PV and facility consumption for every day of the year. 2012.

Figure 3.32 juxtaposes the daily solar consumption produced by the system to the consumption needs of the facility. The figure depicts a system that produces a lot more electrical energy, than is required by the facility on a daily basis. Storing that amount of electricity is infeasible. More than likely, the system will be grid tied and the electricity will be introduced back into the electrical system. If the electrical provider paid \$0.06 per kWh put back into the system, this facility could expect a less than 2 year (1.58 yr) payback on the system.

Power Factor Correction

When electrical power is supplied to a customer, they use the power in two different ways; the first is to produce work. Termed “Real Power” and measured in kW, it is the electrical power that goes to turn motors, operate lights, run chillers, etc. The second way a customer uses the supplied electricity does not produce work, but is used

to energize the magnetic field in motor coils and lighting ballasts. This type of power is termed “Reactive Power” and is measured in KVAR. The electrical power supplied by the electrical provider is termed “Apparent Power” and is measured in kVA. The relationship between the different types of power can be seen in Figure 3.33

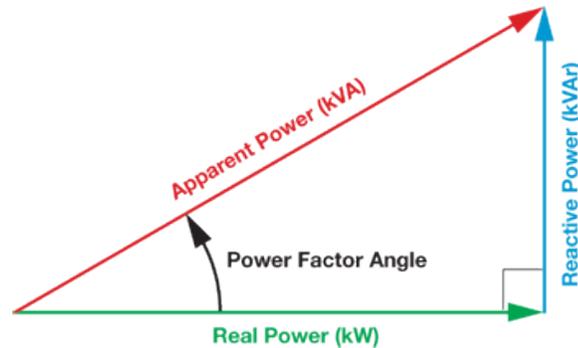


Figure 3.33: Power triangle. [20]

As can be seen in Figure 3.33, the power factor is the ratio of the real power to the apparent power, and is equivalent to the cosine of the angle between the real power and apparent power. The power factor indicates how much of the power the facility is using in proportion to the amount that the utility is actually supplying, thus

$$\text{Power Factor (PF)} = \frac{\text{Real Power (kW)}}{\text{Apparent Power (kVA)}} = \frac{\text{kW}}{\sqrt{(\text{kW})^2 + (\text{KVAR})^2}}$$

When the power factor is low, the real power usage by the facility is low in comparison to the supplied power, or in other words, the plant has a large reactive power load. This means that more power than necessary is supplied to the facility. Since the added demand results in added infrastructure to supply the power, the electrical supply company passes on the cost to the client. For the most part, facilities require a facility maintain a power factor of 0.90 to 0.95. Facilities that exhibit a power factor lower than

the required value are charged a power factor penalty, which is exhibited by an increase in the charged demand, thus a facility with an inadequate power factor is characterized by two demand values; actual demand and billed demand. Actual demand is the electrical demand required by the facility and is the same as the electrical demand described above. Billed demand is the penalized demand for a facility whose power factor is lower than required. The billed demand is determined by the following

$$P_B = \frac{(P)(PF_R)}{PF}$$

where PF_R is the required power factor and PF is the facility power factor.

To mitigate the negative effect of a low power factor, facilities are augmented with a capacitor bank. Capacitor banks increase the power factor by nullifying a facilities reactive power. Capacitor banks are sized using two criteria; fixed and variable capacitance. The fixed capacitance is a facilities lowest reactive power experienced for all 12 demand peaks throughout the year. The variable capacitance is the difference between the fixed capacitance and the capacitance required to raise a facilities power factor to the required value.

The following visualization was generated from a data set for a hotel situated in Texas.

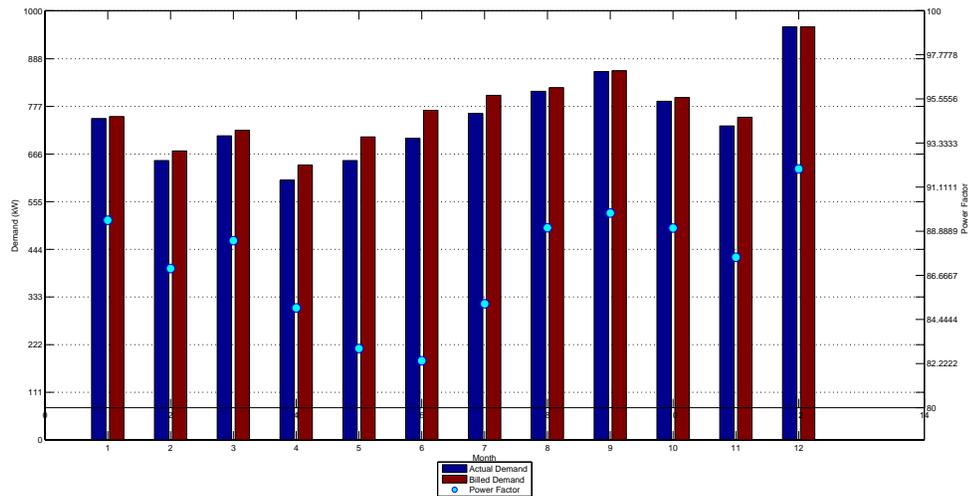


Figure 3.34: Depiction of power factor.

Figure 3.34 depicts the facility demand data, including power factor for the hotels usage during 1998. Note that billed demand is larger than actual demand for months where the power factor was below 0.90. In this case, the facility incurred an additional demand cost of \$2,216 for power factor issues. To correct for this and install a capacitor bank that promotes a power factor of 0.90, a system with a fixed capacitance 115 KVAR and a variable capacitance of 368 KVAR will be required. At a cost of \$50 per KVAR for fixed capacitance and \$70 per KVAR for variable capacitance, the system will cost \$26,495, at a simple payback period of 11 years.

Ratcheting

Electrical demand ratcheting is a cost scheme implemented by electrical service providers to stabilize the cost associated with transmission to large facilities whose systems are connected to the grid on a transmission level (110 kV and above). In some

circumstances, a manufacturing facilities substantial electrical load makes it necessary to dedicate large and expensive pieces of electrical equipment to manufacturing operations, usually in the form of dedicated transformers. Even though the facility may have a large load, their manufacturing operations may characteristically exemplify large swings in electrical demand, being much larger during some months and relatively low during others. Under these circumstances (large electrical load and large variation in load throughout the year), the electrical provider will impose a demand ratchet to recover the cost associated with installing equipment (transmission lines and transformers) without passing the cost to other customers during periods of the year when the transmission equipment is not being used to full capacity. Additionally, demand ratchets promote facilities with large demand swings to increase and stabilize their loads, promoting favorable load characteristics for the electrical provider.

Demand ratchets are imposed through two characteristics; a ratcheting percentage and a time window. The time window is the period of time a ratcheting clause takes effect; usually either 9 or 12 months. The ratcheting percentage is the electrical demand required by the provider. The demand for one month is gauged against the largest demand in the ratcheting time window. If the demand for the month in question is equivalent or larger than the ratcheting percentage of the largest demand within the window, the facility is charged for the actual demand of said month. If the demand is less than the ratcheting percentage of the largest demand within the window, then the facility is charged for the largest demand peak found within the time window. Suppose the actual demand measured in November 2012 was 1,000 kW. Also suppose

that the largest demand peak measured from November 2011 to November 2012 was measured as 500 kW. Since the demand peak found in November 2012 was larger than 80% of the largest demand peak found in a 12 month period prior, the actual demand was used to charge the facility. Now say that the demand found in November 2012 was 100 kW. Since the demand was less than 80% of the largest demand peak found within a 12-month window, the facility will be charged for 500 kW, not the original 100 kW consumed. In this example, the time window is 12 months and the ratcheting percentage is 80%; these will vary by provider. The following figure outlines the effects for a possible ratcheting scheme for a hotel found in Texas. The ratcheting time window was 12 months and the ratcheting percentage was 80%.

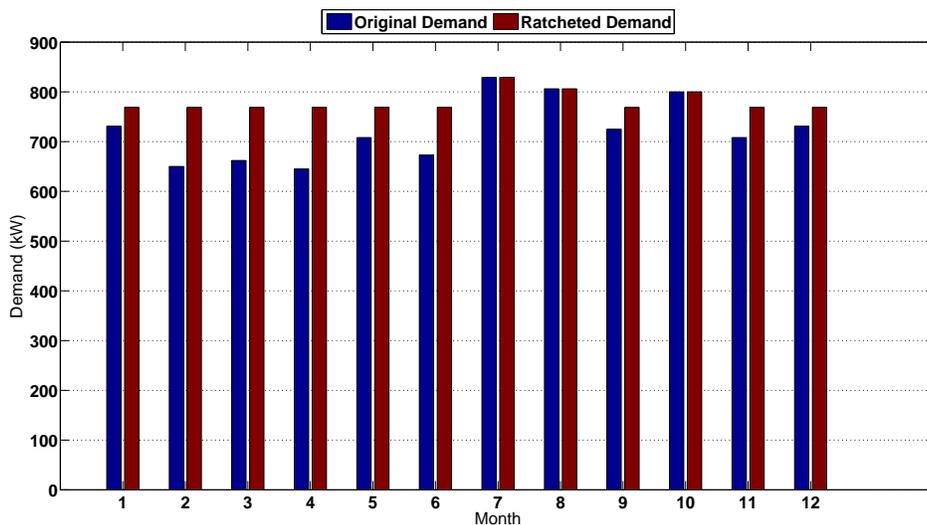


Figure 3.35: Ratcheting penalty for 1999.

Figure 3.35 depicts the results of a theoretical ratcheting scheme. From January 1999 to June 1999, this facility was charged more for electrical demand than consumed,

since the demand ratchet for these months was set on December 25, 1998; the same is true for September 1999, November 1999 and December 1999. Since the demand for July 1999, August 1999 and October 1999 was more than 80% of the largest demand 12 months prior, the demand charge for these months were the actual demand.

In this example the facility incurred an additional cost of \$5,504 in electrical demand. Facilities with demand ratchets are bit out of the ordinary. Demand savings associated with various ECMs take effect after the ratcheting window, this payback calculations will be shifted by the ratcheting window period.

4. AUTOMATIC REPORT GENERATOR

Introduction

The tool suite that has been described in the previous two chapters has focused on demand analysis. By exploiting electrical demand information, this thesis has shown: (a) energy conservation measures can be identified before an onsite visit. Demand visualization can help an energy auditor establish when electrical demand peaks are occurring, thus providing an opportunity to reduce electrical usage during peak operating hours. (b) Establish unknown variables that are required to determine usage and savings opportunities. Weather disaggregation separates weather-based loads, such as HVAC, from the total demand profile to determine how much electricity goes towards weather effected systems. (c) Size and design systems that enable a facility to optimize their electrical usage. Power correction uses demand information to determine the proper variable and fixed capacitance necessary to increase a facilities power factor. (d) Provide an interface for those who are so inclined to request access to their demand data (such as through “green button” portals) and aid them in identifying electrical usage aberrations and savings measures. Additionally, all of the tools help to reduce the time required to perform an in depth and rigorous analysis. The final tool in the suite to be discussed is the automatic report generator (ARG).

Mechanics of Automatic Report Generation

A traditional energy audit is streamlined through a series of objectives that aid in reducing and optimizing electrical energy consumption. (a) An analysis is done on the

current state of utility usage. Whether by utility bills or metered demand data, yearly and monthly electrical consumption, demand and associated costs are baselined. (b) An onsite energy audit helps to identify processes and equipment that would benefit from commissioning, retro-commissioning or retrofitting endeavors. This usually involves the use of measuring equipment (combustion analyzers, data loggers, motor power, etc.) to aid in calculating savings recommendations. (c) A period of in depth analysis to quantify energy saving measures. Sometimes this required the use of software (Energy Plus, Trace, MotorMaster International Plus, 3E Plus, etc.), while other times, calculations are fairly straightforward. Finally, (d) monitoring and verifying savings; sometimes done through sub metering. The automatic report generator aims to reduce the analysis period required by the energy auditing process. The hope is that by reducing the analysis time, recommendations are more likely to be implemented by plant management.

Automatic report generation is the amalgam of two separate programs; Mathworks' Matlab and Microsoft's Word. Figure 4.1 exemplifies the ARG input/output characteristics.

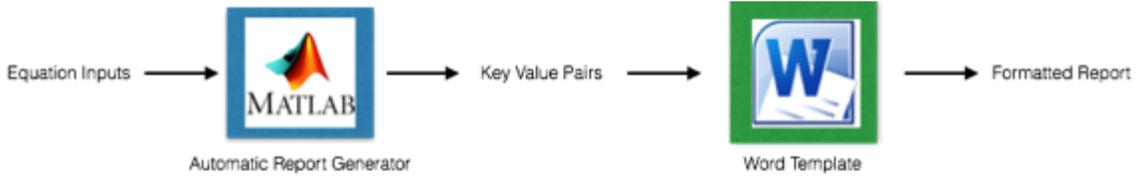


Figure 4.1: Input/Output scheme of Automatic Report Generator.

Using the report generator module in Matlab, the user constructs a report file. The report file has two separate components, evaluate and paragraph. The evaluate component is much like a traditional m-file. A user defines input variables which are

used in a function; variables such as the number of metal halide fixtures, the power consumption of the bulb, annual operating hours, etc. to calculate the yearly energy consumption. The outputs of these functions are passed to the paragraph component, where they are matched with keywords. The keyword “value1” is matched with the output of the “consumption” variable from the evaluate component. The keyword “value2” is matched with the “demand” variable in the evaluate component, etc. The final output from the paragraph component is a key value pair map, matching all keywords with their corresponding values calculated in the evaluate component. The key value pair map is passed on to a word template which has been pre formatted with the keywords; “value1”, “value2”, etc. A word macro uses the key value pair map to replace the keywords, found in the pre formatted template, with the values found on the key value pair map. The final result is a formatted report; keywords replaced with numbers.

Report Templates

The ARG requires two files; they key value pair text file output from Matlab’s ARG, and a word template with value specifiers. The following templates represent the most common ECMs recommended by the TAMUIAC. While each facility is unique, the development of ARG word templates has attempted to encompass those permutations regarded as most prevalent throughout various facilities. Additionally, a data input sheet, useful in identifying equation inputs, accompanies each word template. Templates are separated into six categories; lighting, compressed air, motors, power factor correction, boilers and late fees. These reports may be found in the appendix. Table 4.1 comprises the various ECMs that have been automated.

Table 4.1: Automated report generator ECM list

ECM (Assessment Recommendation)	Type
Retrofit Exit Signs (Single Lamp Type)	Lighting
Retrofit Exit Signs (Multiple Lamp Type)	Lighting
Install Skylights (Single Area)	Lighting
Install Skylights (Multiple Area)	Lighting
Lighting Retrofit (Single Lamp Type)	Lighting
Lighting Retrofit (Multiple Lamp Type)	Lighting
Turn off Lights	Lighting
Repair Compressed Air Leaks	Compressed Air
Reduce Compressed Air Pressure	Compressed Air
Engineered Nozzles	Compressed Air
Retrofit to High Efficiency Motors	Motors
Use Energy Efficiency Belts	Motors
Use Synthetic Lubricants	Motors
Install Capacitor Bank to Increase Power Factor	Power Factor
Boiler Tune Up (NG Usage Known)	Boiler
Boiler Tune Up (NG Usage Unknown)	Boiler
Avoid Late Fee Penalties (Single Utility Type)	Late Fees
Avoid Late Fees (Multiple Utility Type)	Late Fees

5. CASE STUDY 1

Introduction

The following chapter will showcase the electrical demand analysis tools described in the previous chapter, as applied to the electrical demand data set for a manufacturing facility found in the north western portion of the United States; in an ASHRAE climate zone 6B – cold and dry. The data set was obtained during a TAMUIAC site visit in 2013 and exemplifies 15-minute demand data from January 2007 to December 2012.

Plant Background

This section entails information gathered during the initial IAC site visit in 2013. This section details the energy findings for the facility. Annual production was approximately 60,000 tons with gross annual sales of approximately \$60 million. The plant operates 4 days a week, with a single shift of workers. The work schedule is staggered for different employees depending on the production area. Most major equipment is running from approximately 4:00 am to 6:30 pm on production days. The plant allows 10 holidays per year and takes 3 weeks off, both in the winter and summer. Current employment at the plant is around 100-150 people. Workers take staggered breaks during their shift to keep production going.

The plant consists of one main building with 153,500 ft² area. Only the office space is air-conditioned; the production area of the plant has no space conditioning. There is no insulation in the production area. Large equipment used in the manufacturing

process includes two 400 hp air compressors, one 300 hp air compressor, a large oven used for annealing, seven casting stations, and several large motors. The total horsepower from all larger motors totals 13,380 hp. Table 5.1 shows a breakdown of the motors is detailed below.

Table 5.1: Facility motor list.

No. Motors	Power (hp)	Power (kW)	Efficiency
64	5	3.73	0.89
17	7.5	5.595	0.89
37	10	7.46	0.89
20	15	11.19	0.89
9	20	14.92	0.9
11	25	18.65	0.91
10	30	22.38	0.91
11	40	29.84	0.91
33	50	37.3	0.91
7	60	44.76	0.91
2	75	55.95	0.91
6	100	74.6	0.92
2	112	83.55	0.92
2	125	93.25	0.92
5	150	111.9	0.93
4	250	186.5	0.93
2	300	223.8	0.94
4	350	261.1	0.94
2	400	298.4	0.94
2	450	335.7	0.94
1	725	540.8	0.95
2	800	596.8	0.96
TOTAL	13,380	9,980	-

Plant Energy Consumption

This section details the initial energy analysis done by IAC personnel. Bills pertaining to electrical usage and natural gas usage were supplied by the facility. One electrical meter measures the electrical energy consumption for the plant. Bills for the months of October 2012 through September 2013 were provided for analysis. The bills helped to establish the state of energy consumption for the facility and a diagnosis of avoidable cost incursions; power factor degradation, sales tax and late fee penalties. In this instance, the facility was not charged for sales tax, there were no late fees and the electrical utility did not monitor the facilities power factor. Table 5.2 outlines a breakdown of the electrical costs associated with plant operations from October 2012 to September 2013.

Table 5.2: Summary of electrical utility charges.

Month	Oct. '12	Nov.	Dec.	Jan. '13	Feb.	Mar.	Apr.	May	Jun.	Jul.	Aug.	Sep.	Total	Average
From	9/14/12	10/15/12	11/14/12	12/17/12	1/17/13	2/15/13	3/18/13	4/16/13	5/15/13	6/14/13	7/16/13	8/15/13		
To	10/15/12	11/14/12	12/17/12	1/17/13	2/15/13	3/18/13	4/16/13	5/15/13	6/14/13	7/16/13	8/15/13	9/16/13		
Consumption, Offpeak (kWh)	912,000	472,000	584,000	376,000	672,000	768,000	624,000	744,000	1,048,000	1,080,000	712,000	584,000	8,576,000	714,667
Consumption, Onpeak (kWh)	640,001	616,000	712,000	432,000	912,000	728,000	864,000	504,000	344,000	336,000	232,000	188,000	6,508,001	542,333
Consumption, Total (kWh)	1,552,001	1,088,000	1,296,000	808,000	1,584,000	1,496,000	1,488,000	1,248,000	1,392,000	1,416,000	944,000	772,000	15,084,001	1,257,000
Consumption Charge, Offpeak (\$)	23,447	12,842	15,889	10,230	18,284	20,896	16,978	20,243	28,514	29,385	19,372	32,270	248,350	20,696
Consumption Charge, Onpeak (\$)	23,026	20,050	23,174	14,061	29,684	23,695	28,121	19,188	14,870	14,524	10,029	16,503	236,925	19,744
Consumption Charge, Total (\$)	46,473	32,892	39,064	24,291	47,968	44,591	45,099	39,431	43,384	43,909	29,401	48,772	485,274	40,440
Demand, Offpeak (kW)	4,744	4,208	4,320	3,896	4,296	4,512	4,544	4,456	4,744	5,104	4,992	4,968	54,784	4,565
Demand, Onpeak (kW)	4,656	4,824	4,664	4,224	4,680	4,600	4,856	4,720	4,528	4,840	4,664	4,760	56,016	4,668
Demand Charge (\$)	48,179	42,258	40,857	37,002	40,997	40,296	42,539	51,503	58,502	62,533	60,259	62,451	587,376	48,948
Facilities Charge (\$)	9276.82	9937.44	9607.84	8701.44	9640.8	9476	10003.36	9723.2	9772.64	10,514	10,284	10,383	117,320	9,777
Renewable Energy Credit (\$)	-293.42	-232.96	-247.75	-190.01	-275.79	-263.15	-271.68	-281.9	-268.3	-223.53	-188.29	-233.57	-2,970	-248
Energy Balancing (\$)	1,391	1,105	1174.83	901.01	1,308	1,386	1,534	1,591	1,783	1862.73	1569.04	1946.41	17,552	1,463
Customer Efficiency Services (\$)	3371.38	2759.29	2903.61	2269.65	3198.35	3065.08	3174.79	3273.14	3632.86	3806.9	3252.5	3958.56	38,666	3,222

The information entailed in Table 5.2 makes possible yearly trending for both consumption and demand. Figure 5.1 shows the facility consumption curves while Figure 5.2 shows the facility demand curves. The curves show no deducible trend, weather, production or otherwise.

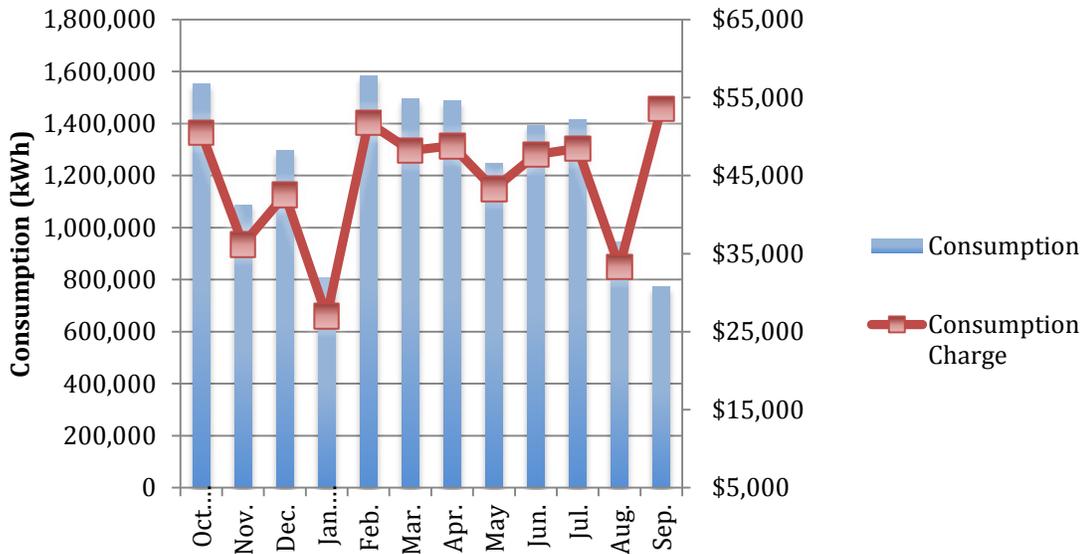


Figure 5.1: Consumption and consumption cost trend for facility.

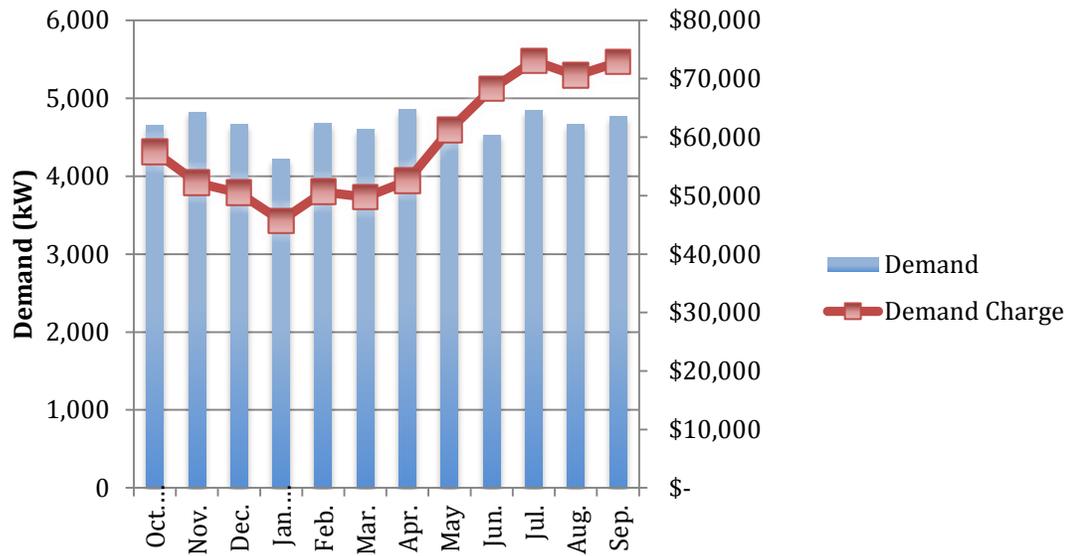


Figure 5.2: Demand and demand cost trend for facility.

Energy Cost Analysis

The cost per unit of electrical energy (consumption) is derived from the bills and was calculated thusly.

$$\begin{aligned}
 & \text{Off Peak Base kWh Charge } \left(\frac{\$}{\text{kWh}} \right) + \text{On Peak Base kWh Charge Summer } \left(\frac{\$}{\text{kWh}} \right) + \text{On Peak Base} \\
 & \text{kWh Charge Winter } \left(\frac{\$}{\text{kWh}} \right) + \text{Renewable Energy Credit } \left(\frac{\$}{\text{kWh}} \right) + \text{Energy Balancing Account } \left(\frac{\$}{\text{kWh}} \right) \\
 & + \text{Customer Efficiency Services } \left(\frac{\$}{\text{kWh}} \right) + \text{Customer Efficiency Services Offset } \left(\frac{\$}{\text{kWh}} \right) \\
 \\
 & = \left(\frac{\$ 0.028048}{\text{kWh}} \right) \left(\frac{1}{2} \right) + \left(\frac{\$ 0.044551}{\text{kWh}} \right) \left(\frac{1}{2} \right) \left(\frac{5 \text{ mo}}{12 \text{ mo}} \right) + \left(\frac{\$ 0.032548}{\text{kWh}} \right) \left(\frac{1}{2} \right) \left(\frac{7 \text{ mo}}{12 \text{ mo}} \right) - \left(\frac{\$ 0.00210}{\text{kWh}} \right) + \left(\frac{\$ 0.01750}{\text{kWh}} \right) \\
 & + \\
 & \left(\frac{\$ 0.03210}{\text{kWh}} \right) - \left(\frac{\$ 0.00730}{\text{kWh}} \right) \\
 \\
 & = \frac{\$ 0.05410}{\text{kWh}}
 \end{aligned}$$

Every kWh of energy consumed costs the facility 5.410 cents; this value is dubbed the avoided cost of electrical energy. Similarly, the cost per unit of power relayed is also determined from the electrical utility bills. It was calculated as.

$$\begin{aligned}
 &= \text{Base Charge Summer} \left(\frac{\$}{\text{kW}\cdot\text{mo}} \right) + \text{Base Charge Winter} \left(\frac{\$}{\text{kW}\cdot\text{mo}} \right) + \text{Facilities Charge} \left(\frac{\$}{\text{kW}\cdot\text{mo}} \right) \\
 &= \left(\frac{\$13.32}{\text{kW}\cdot\text{mo}} \right) \left(\frac{5 \text{ mo}}{12\text{mo}} \right) + \left(\frac{\$8.76}{\text{kW}\cdot\text{mo}} \right) \left(\frac{7 \text{ mo}}{12\text{mo}} \right) + \left(\frac{\$2.12}{\text{kW}\cdot\text{mo}} \right) \\
 &= \frac{\$ 12.78}{\text{kW}\cdot\text{mo}}
 \end{aligned}$$

Every kW of electricity relayed to the facility, for every month costs \$12.78 and is known as the avoided cost of electrical demand.

Load Factor Analysis

Load factor analysis consists of two variables, the production load factor and the electrical load factor. The production load factor (PLF) helps to gauge if equipment is being left on during non-operational hours. A PLF of 0.85 is considered good. A PLF of 1.0 occurs when the facility uses the monthly peak demand at all production hours, and a PLF over 1.0 indicates that some of the equipment is on when the plant is not in operation. The PLF is defined as

$$\text{PLF} = \frac{\text{Monthly Energy Consumption (kWh)}}{\text{Monthly Billing Demand (kW)} \times \text{Oper. Hours per Billing Period (hrs)}}$$

In the case of plant operating around the clock, the PLF and the ELF will be exactly the same. The electrical load factor (ELF) is a measure of how well the facility's electrical

capacity is used on a monthly basis. The ELF is the ratio of energy consumption to the product of actual demand and the number of hours in the billing period. This yields an average fraction of plant capacity used during the billing period. The ELF is defined as

$$\text{ELF} = \frac{\text{Monthly Energy Consumption (kWh)}}{\text{Monthly Billing Demand (kW)} \times \text{Hours per Billing Period (hrs)}}$$

The nominal single-shift electrical load factor ranges 0.20 to 0.25 for a plant operating five days a week at full capacity. The same plant with a two shift operation should be about 0.45 to 0.60, while the load factor for a nominal three shift operation should ideally be between 0.75 and 0.85. An ELF of one represents the best possible use of equipment, and would indicate 24 hours per day, full load operation of all electrical equipment, every day of the month, at a constant capacity all year long. Table 5.3 represents the facility load factor, ELF. Table 5.4 depicts the facility load factor, PLF. Figure 5.3 depicts a graphical representation of the facility load factors, PLF and ELF for every month of the year.

Table 5.3: Electrical load factor variables.

Month	Days	Hours	Consumption (kWh)	Demand (kW)	ELF
Oct '12	31	744	1,552,001	4,656	0.45
Nov	30	720	1,088,000	4,824	0.31
Dec	33	792	1,296,000	4,664	0.35
Jan '13	31	744	808,000	4,224	0.26
Feb	29	696	1,584,000	4,680	0.49
Mar	31	744	1,496,000	4,600	0.44
Apr	29	696	1,488,000	4,856	0.44
May	29	696	1,248,000	4,720	0.38
Jun	30	720	1,392,000	4,528	0.43
Jul	32	768	1,416,000	4,840	0.38
Aug	30	720	944,000	4,664	0.28
Sep	32	768	772,000	4,760	0.21
				Average	0.37

Table 5.4: Production load factor variables.

Month	Days	Hours	Consumption (kWh)	Demand (kW)	PLF
Oct '12	31	177	1,552,001	4,656	1.88
Nov	30	171	1,088,000	4,824	1.32
Dec	33	189	1,296,000	4,664	1.47
Jan '13	31	177	808,000	4,224	1.08
Feb	29	166	1,584,000	4,680	2.04
Mar	31	177	1,496,000	4,600	1.84
Apr	29	166	1,488,000	4,856	1.85
May	29	166	1,248,000	4,720	1.59
Jun	30	171	1,392,000	4,528	1.80
Jul	32	183	1,416,000	4,840	1.60
Aug	30	171	944,000	4,664	1.18
Sep	32	183	772,000	4,760	0.89
				Average	1.54

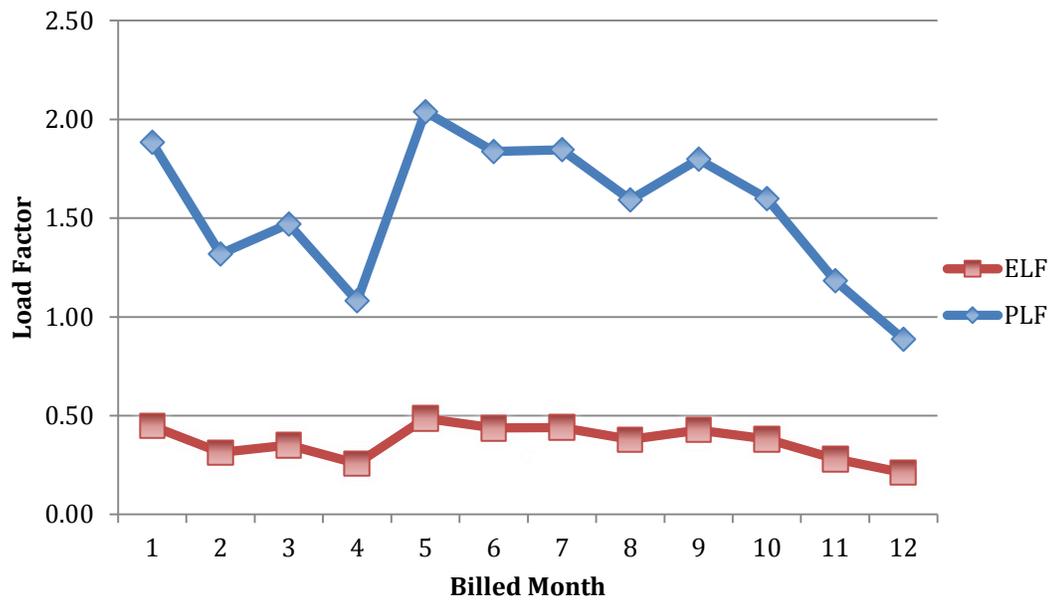


Figure 5.3: Trending of PLF and ELF.

Summary of Plant Statistics

The Summary of Plant Statistics is an exhibition of the characteristics inherent to any one manufacturing facility. It outlines many of the values that are necessary when calculating assessment recommendation variables such as yearly savings and implementation cost. Table 5.5 highlights many of the attributes for the facility that will be analyzed. The source for every category comes from one of three places; the Texas A&M University Industrial Assessment Center, Plant Management or an analysis of the monthly bills. Standard classifications codes, such as the Standard Industrial Classification (SIC) Code or the North American Industry Classification System (NAICS) Code are used to identify the type of facility being audited. This is for the purpose of gathering and analyzing statistical data related to the facility, without identifying the facility by name, the type of data that is collected, in part is identified by plant management during the initial site visit meeting. The type of product, number of employees, hours of operation and other facility related statistics are gathered. Finally, the bills are used to calculate the type of savings that are to be had if electrical consumption and demand are reduced. Table 5.5 depicts the summary of plant statistics while Table 5.6 shows the yearly electrical demand and consumption characteristics for the plant.

Table 5.5: Plant specific variables.

Category	Value	Source
Standard Industrial Classification (SIC) Code	3321	TAMUIAC Personnel
North American Industry Classification System (NAICS) Code	331511	TAMUIAC Personnel
Site Visit Date	November 21, 2013	TAMUIAC Personnel
Production Amount	40,000 $\frac{\text{tons}}{\text{year}}$	Plant Management
Operating Hours	2,353 $\frac{\text{hrs}}{\text{yr}}$	Plant Management
No. of Employees	100-150	Plant Management
Maintenance Labor Rate	$\frac{\$ 35}{\text{hr}}$	Plant Management
Hourly Labor Rate	$\frac{\$ 40}{\text{hr}}$	Plant Management
Avoided Cost of Electrical Consumption	$\frac{\$ 0.05410}{\text{kWh}}$	Bills
Avoided Cost of Electrical Demand	$\frac{\$ 12.78}{\text{kW} \cdot \text{mo}}$	Bills

Table 5.6: Summary of electrical consumption, demand and cost variables.

Average Demand (kW)	Total Demand (kW)	Total Consumption (kWh)	Average Consumption (kWh)	Total (\$)
4,668	56,016	15,084,000	1,257,000	1,239,959

Demand Visualization

The following sections will exemplify the power of the demand visualization tool to data mine quickly and accurately electrical statistics of the facility throughout

different time frames. As mentioned previously, demand visualization helps an energy auditor establish trends and denote periods of unusually electrical usage. This information can be used to prompt facility management and inquire as to why such erroneous usage characteristics are present. Demand peaks, electrical consumption, and the cost associated with both the electrical characteristics are only some useful point of inquiry. The following analysis will show how this particular facility consumes and uses electricity from 2007 to 2012.

Yearly

Facility electrical demand usage is dynamic, changing with a variety of factors; weather, plant activities, shutdown and startup schedules as well as other reasons that are particular to facility environments. Visualizing facility demand on a yearly basis helps to establish trends and aids in the diagnosis of consumption and demand aberrations. Figure 5.4 through Figure 5.9 depict the maximum monthly demand for each month spanning from January 2007 to December 2012. Demand throughout the time frame fluctuates between 4000 and 5000 kW, with no discernable trend. Since there is no obvious peaking during the summer or winter months, there does not seem to be equipment whose power usage is dependent on the weather. For 2008, there is a drastic decrease in demand for March, but this could be associated with the 3-week period off schedule explained in the plant background section. Drastic increases in demand usage are of more concern than extraneous below average usage characteristics, thus the below average demand in March 2008 is of no concern. Plant operations seem to be steady throughout the analyzed time frame, exemplifying a lack of demand issues.

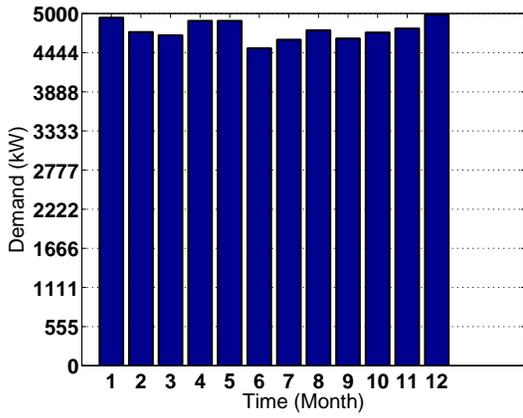


Figure 5.4: Facility monthly demand peaks for 2007.

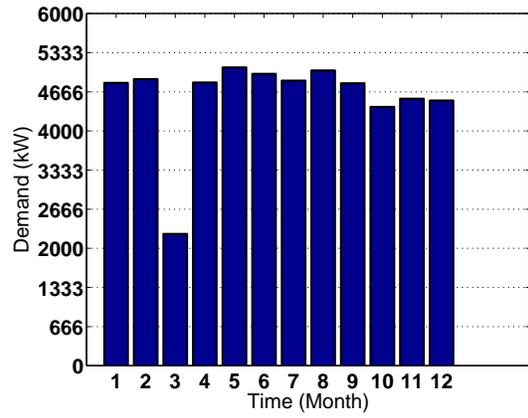


Figure 5.5: Facility monthly demand peaks for 2008.

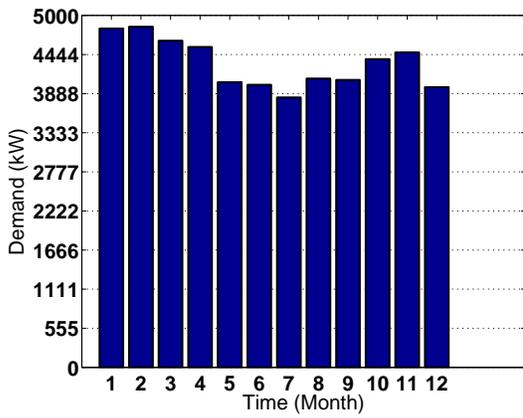


Figure 5.6: Facility monthly demand peaks for 2009.

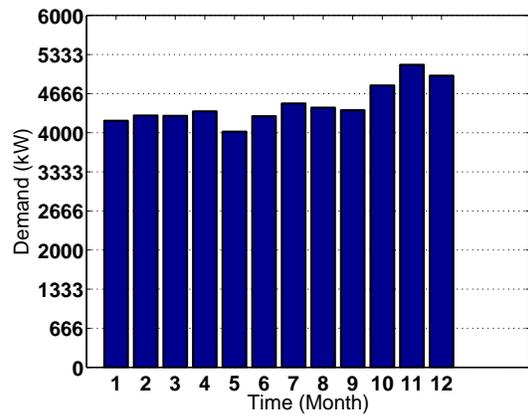


Figure 5.7: Facility monthly demand peaks for 2010.

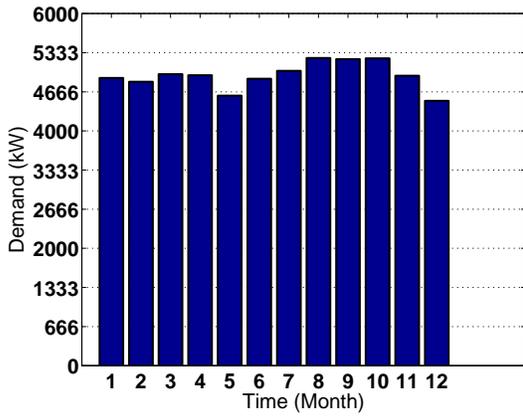


Figure 5.8: Facility monthly demand peaks for 2011.

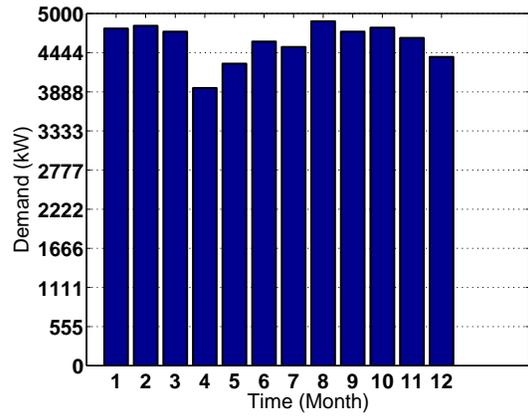


Figure 5.9: Facility monthly demand peaks for 2012.

Figure 5.10 through Figure 5.15 depict the maximum monthly consumption for each month spanning from January 2007 to December 2012. Electrical consumption is staggered in comparison to the yearly demand profiles. This may be due to fluctuating operational hours or a change in production, requiring more product during some months and less product in others.

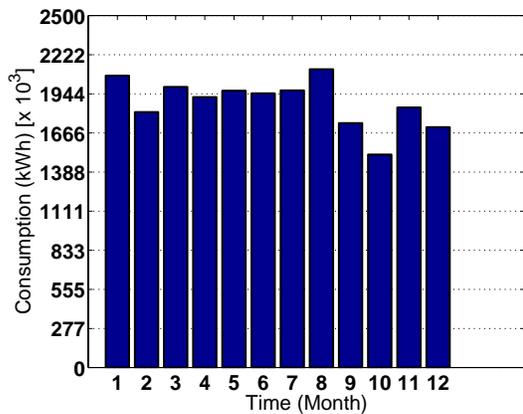


Figure 5.10: Facility monthly electrical consumption for 2007.

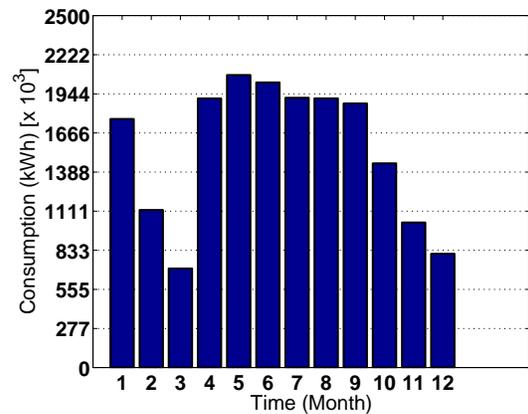


Figure 5.11: Facility monthly electrical consumption for 2008.

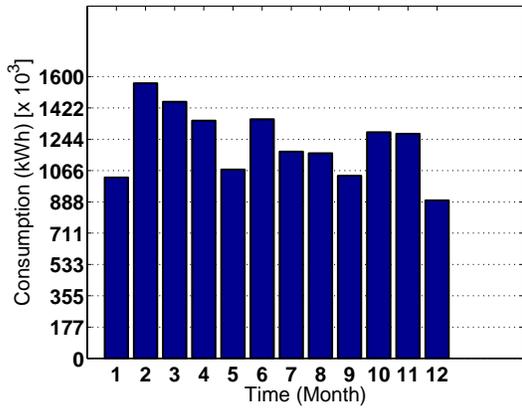


Figure 5.12: Facility monthly electrical consumption for 2009.

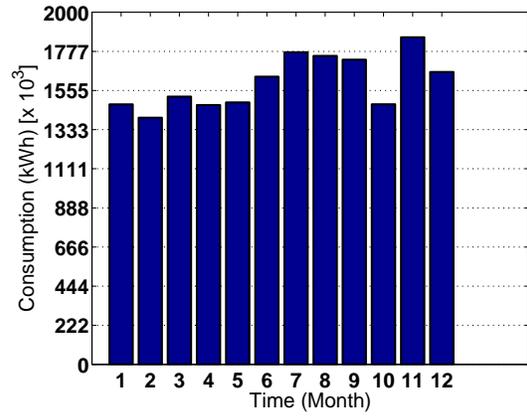


Figure 5.13: Facility monthly electrical consumption for 2010.

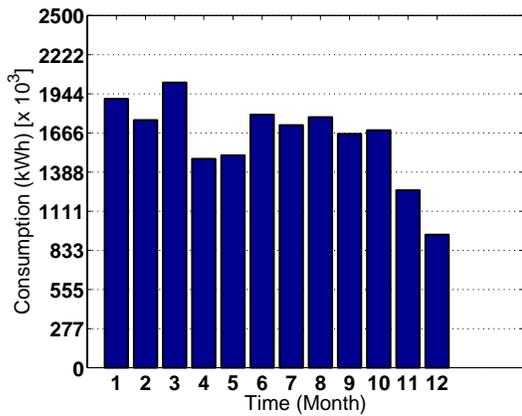


Figure 5.14: Facility monthly electrical consumption for 2011.

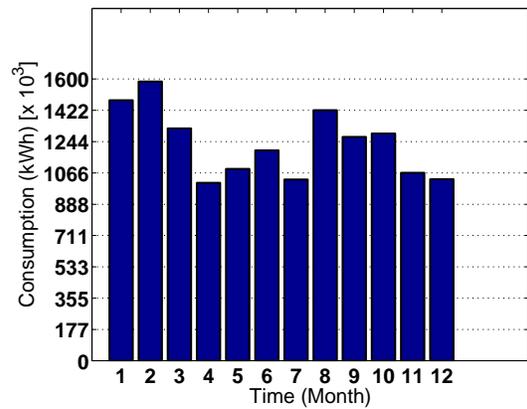


Figure 5.15: Facility monthly electrical consumption for 2012.

Table 5.7 outlines the total consumption and total demand associated with their respective years, as well as the costs associated with demand and consumption. The largest electrical consumption is found for 2007. After 2007, there is a staggered increase and decrease trend through to 2012. Although it is unclear as to why this seems to be the case, one possible explanation may have its roots in the economic depression of 2008.

Table 5.7: Summary of consumption and demand characteristics.

Year	2007	2008	2009	2010	2011	2012
Total Demand (kW)	57,203	55,008	51,756	53,702	59,321	55,180
Total Demand Cost (\$)	731,056	703,002	661,451	686,316	758,130	705,210
Total Consumption (kWh)	22,604,234	18,591,390	14,656,265	19,240,451	19,523,369	14,786,938
Total Consumption Cost (\$)	\$1,222,889	\$1,005,794	\$792,903	\$1,040,908	\$1,056,214	\$799,973

Predicting Operating Hours

Using the yearly demand curve, the yearly visualization tool allows the user to predict what the yearly hours of operation are. Model 1 looks at the base demand points for every day throughout the year and creates a limit between the minimum daily base demand and maximum daily base demand point. All values that lie between the limits are considered non-operational hour data points. Model 2 allows for the user to move the upper limit to better reflect what is seen on the yearly demand curve. Figure 5.16 shows

the lower limit in black, the average demand in cyan, the upper limit for Model 1 in green and the upper limit for model 2 in red.

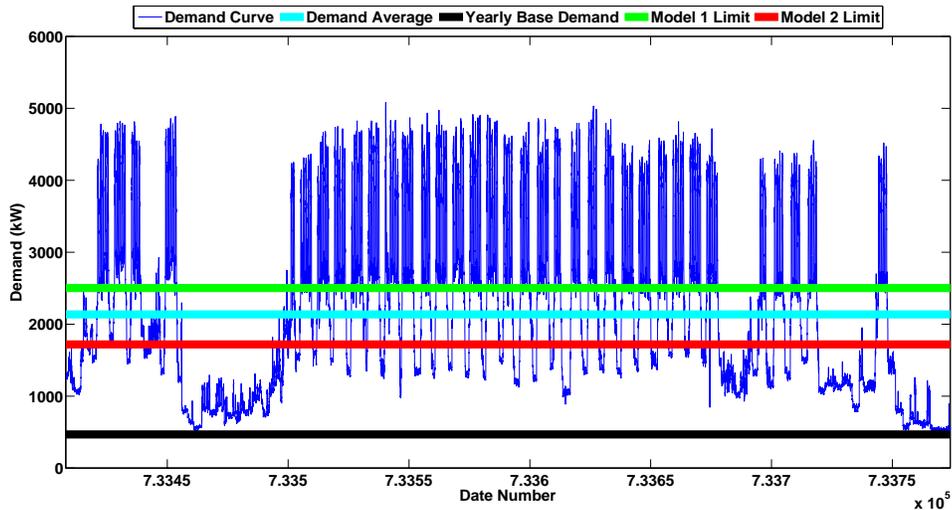


Figure 5.16: Yearly facility demand trend with model limits.

When we zoom in, it becomes clear that the model 1 upper limit does not account for much of the variation found between weeks. In this case, model 1 predicts that there are 5,734 hours of inactivity for this plant throughout the year. When we zoom in closer, as seen in Figure 5.17 it becomes evident that many of the nonoperational data points that fall between days throughout the work week are not being accounted for. In this case, model 2 can be used to shift the upper limit of the non operational hour data threshold to encompass these points.

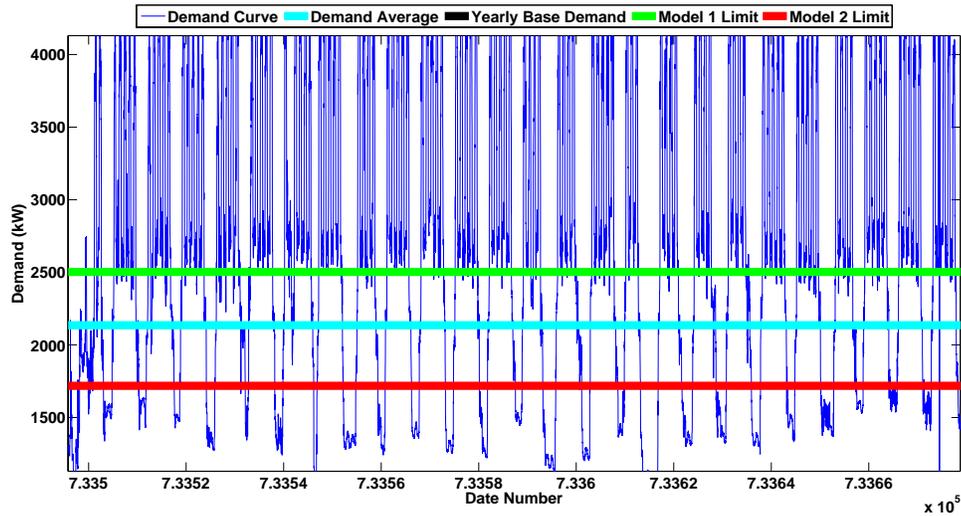


Figure 5.17: Zoom from figure 5.16 to show model inconsistency 1.

Figure 5.18 shows an increase in the attenuation factor from 1 to 1.45, many of those data points that fall though out the work week are encompasses, changing the non operational hours from 5,353 hours to 6,595 hours of non operation, thus there are 2,165 hours of operation through the year. An interview with plant management revealed that there were 2,353 hours of operation throughout the year, a 7% difference from the predicted value

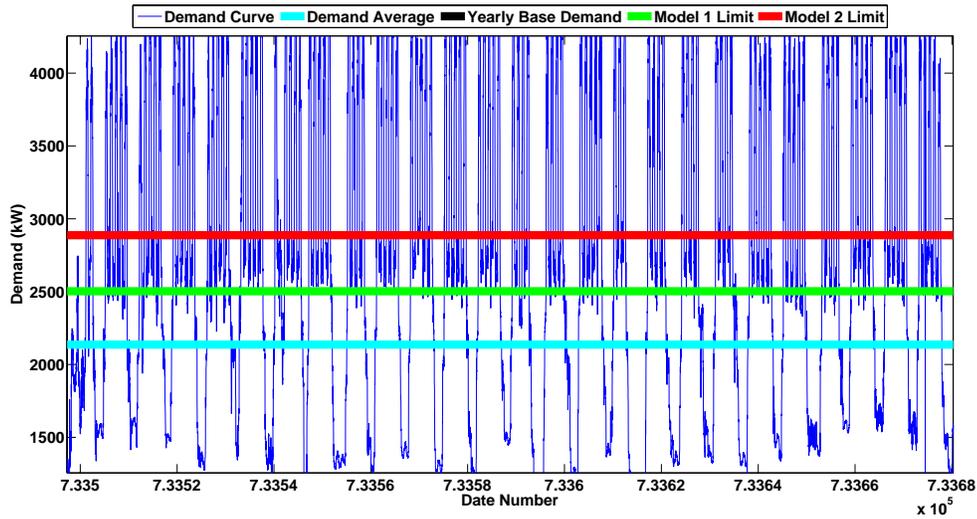


Figure 5.18: Zoom from figure 5.16 accounting for model inconsistency 2.

Monthly

Since facility demand is charged as the largest demand peak throughout the month, it is natural to visualize demand on a monthly timescale. For the most part, there was nothing to be seen on any month throughout the year that would alarm one to any sort of inefficient operation of equipment, operation that would cause a drastic increase in electrical demand; Figure 5.19 represents an atypical month. From an interview with plant personnel, it is known that the plant becomes non-operational for 3 weeks throughout the winter and summer. This month seems to exemplify this characteristic, showing minimal demand usage for the first 3 weeks throughout the month, and then ramping up during the last week of the month. Figure 5.5 shows that for every month throughout the data set, March of 2008 represents the lowest usage of electricity.

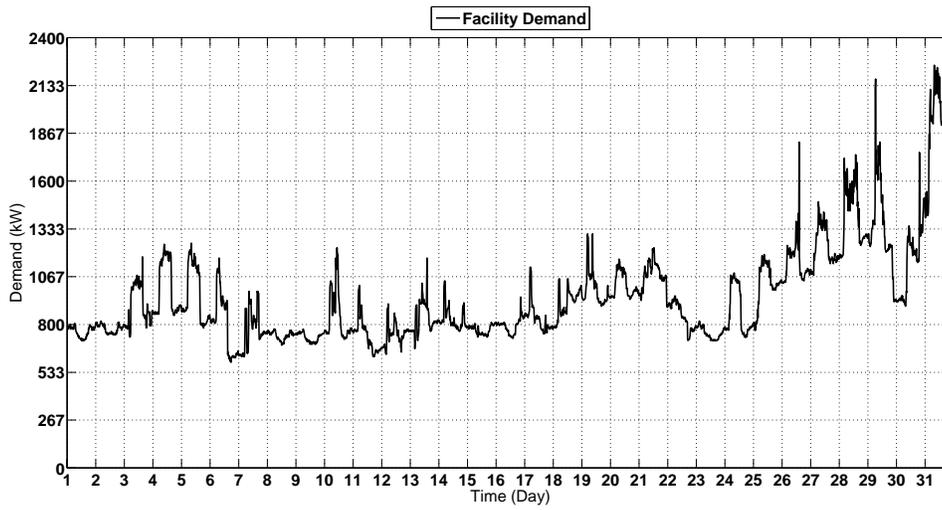


Figure 5.19: Aggregated demand set for March 2008.

By reducing the data to show maximum daily demand and the daily base demand, several aspects of this particular demand profile becomes highlighted. First is that there is some characteristic of daily demand usage, as one would expect an increase in electrical usage throughout the weekday and minimal usage throughout the weekend. This facility is on a 4 day per week, 4 am to 6 pm workweek. One can also deduce that this month shows production ramping since the maximum demand was less than 2,400 kW; a typical month's demand fluctuates between 4,000 and 5,000 kW. Also, there seems to be a large base demand. If all of this electrical usage were accounted for in necessary equipment usage, then more than likely there would not be such a large variation from day to day. From Figure 5.20, this facility is capable of running a base load of approximately 600 kW, meaning that and base load above that is avoidable. If it were

possible to reduce daily electrical minimal usage to the monthly base load, 600 kW, then this facility could reasonably save close to 725,000 kWh and nearly \$40,000 per year.

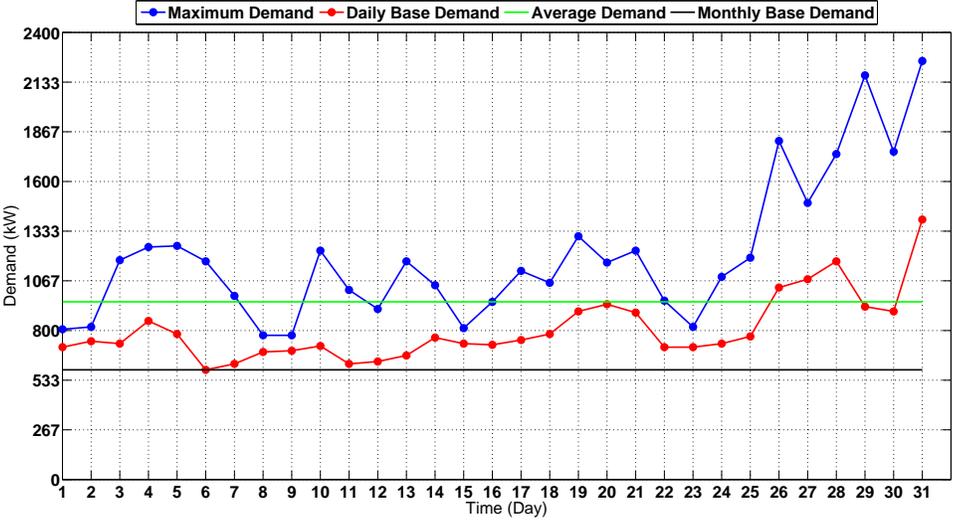


Figure 5.20: Reduced demand set for March 2008.

Several beneficial key variables are available from the monthly visualization tool, and are detailed in Table 5.8

Table 5.8: Key monthly variables in electrical consumption for March 2008.

Monthly Base Demand (kW)	409
Average Demand (kW)	2,005
Maximum Demand (kW)	4,787
Demand Charge	\$61,180
Peak Demand Day	12
Monthly Consumption (kWh)	1,491,852
Consumption Charge	\$80,709
Daily Base Consumption Elimination (kWh)	1,019,596
Daily Base Consumption Elimination Cost Savings	\$55,160
Daily Base Consumption Reduction Savings (kWh)	724,684
Daily Base Consumption Reduction Cost Savings	\$39,205

While March 2008 exemplified a non-traditional month, Figure 5.21 and Figure 5.22 show what a more traditional month looks like. The demand peaks represent facility electrical demand while operations are fully engaged; the troughs are non-operational hours between days. This month shows that there were approximately 13 days when the facility was operational; minimal operations during weekends and during the last week of the month. There does not seem to be any extraneous demand usage as the maximum demand for everyday is approximately the same, around 5,000 kW. Similar to Figure 5.19 the daily base load during operational days fluctuates drastically from the monthly base load, more than likely due to equipment being left on. Again, reducing the daily

base to more closely match the monthly base could save close to 327,000 kWh and \$18,000.

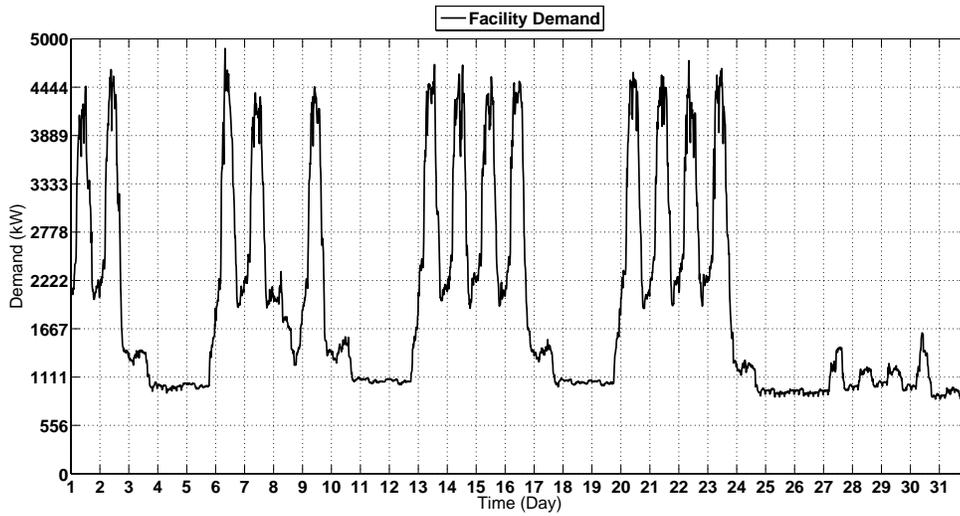


Figure 5.21; Aggregated demand profile for August 2012.

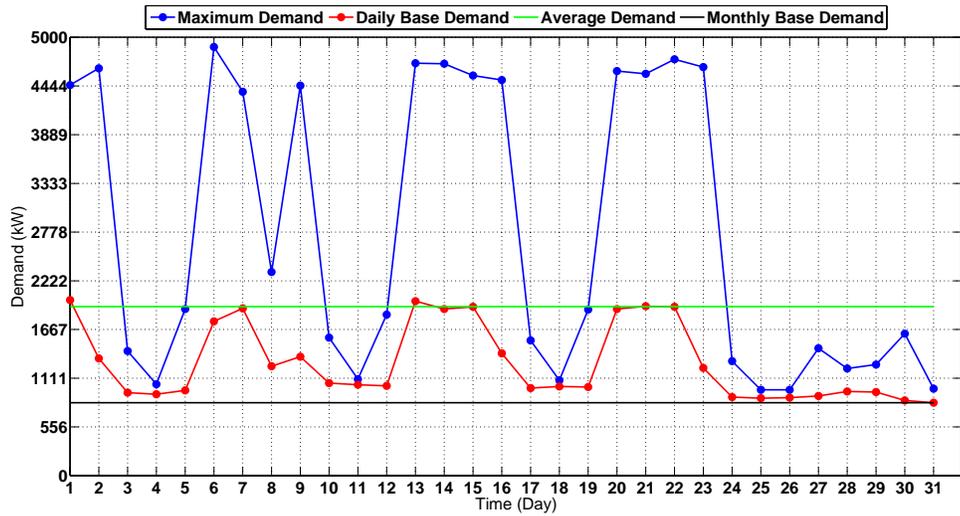


Figure 5.22: Reduced demand profile for August 2012.

Table 5.9 shows some of the savings and usage variables associated with electrical usage for August 2012.

Table 5.9: Key monthly variables in electrical consumption for August 2012.

Monthly Base Demand (kW)	832
Average Demand (kW)	1,927
Maximum Demand (kW)	4,889
Demand Charge	\$62,489
Peak Demand Day	6
Monthly Consumption (kWh)	1,433,940
Consumption Charge	\$77,576
Daily Base Consumption Elimination (kWh)	929,670
Daily Base Consumption Elimination Cost Savings	\$50,078
Daily Base Consumption Reduction Savings (kWh)	326,630
Daily Base Consumption Reduction Cost Savings	\$17,670

Daily

Visualization of daily demand curves helps to hone in on aberrations found during visualization on other timescales. If a drastic demand peak was found during a month, the daily visualization helps to see at what time of day the demand peak occurred and if weather the daily demand curve represents a typical daily demand curve. Figure 5.23 shows what a non-traditional daily demand curve looks like and represents May 12, 2008. Since the demand curve does not seem to have any noticeable trend, it seems more than likely that electrical usage during this day was not due to daily operations of the

facility, and more likely due to other type of equipment either being left on or necessary for base facility processes such as security lighting. The average demand hovered around 750 kW throughout the day, but drastically peaked around 5 AM and 12 PM.

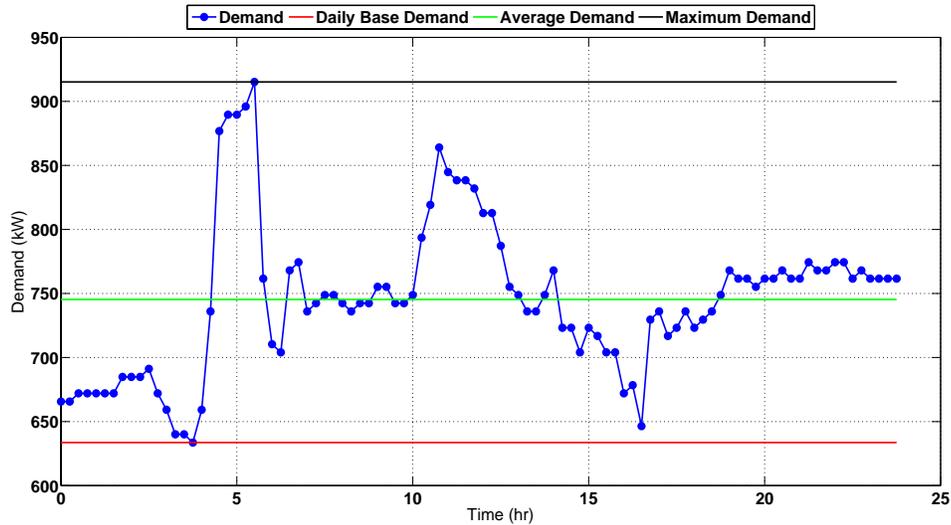


Figure 5.23: Daily demand curve for May 12, 2008.

Table 5.10 characterizes many of the variables that are of interest, including the daily consumption, consumption charge and the time in which the demand peaked.

Table 5.10: Key daily variables for May 12, 2008

Maximum Demand (kW)	915
Average Demand (kW)	745
Daily Base Demand (kW)	633
Consumption (kWh)	17,709
Consumption Charge	\$958
Peak Time	5:45

Figure 5.24 shows what a typical daily consumption profile looks like for this facility; it is a daily demand snapshot for August 06, 2008. Operation begins to ramp up at around 4 AM and ramp down at around 4 PM. Demand peaks at 8 AM, which is fine for a facility with no climate control. Demand peaks at a value of 4,889 kW, average for this facility. No aberration seems to be present.

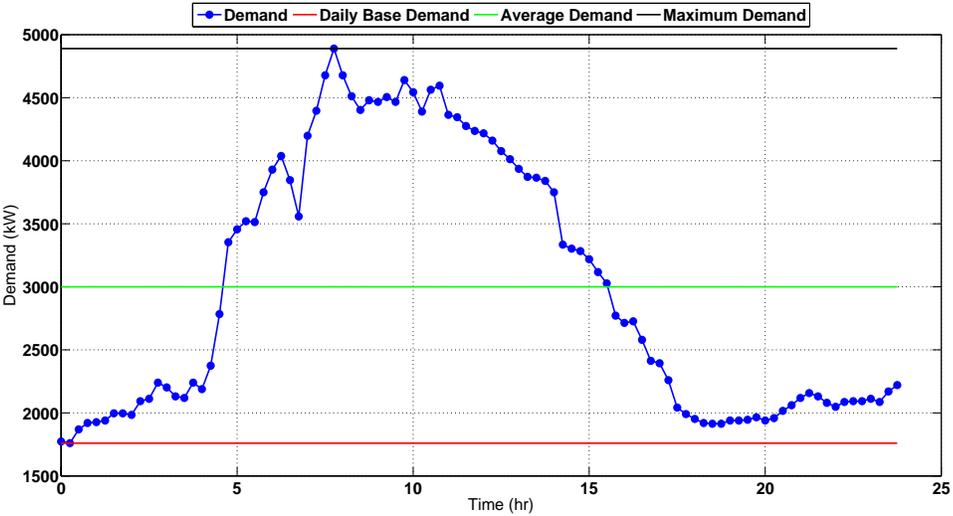


Figure 5.24: Daily demand curve for August 06, 2008.

Table 5.11 shows some of the variable of interest for this day.

Table 5.11: Key daily variables for August 6, 2008

Maximum Demand (kW)	4,889
Average Demand (kW)	3,000
Daily Base Demand (kW)	1,760
Consumption (kWh)	71,502
Consumption Charge	3,868
Peak Time	8:00

Weekday

Visualizing demand on a per day of the week basis helps to visualize any aberrations that are present in a facilities electrical consumption that is happening repeatedly, on the same day of the week, throughout the year. Figure 5.25 show the peak, average and base demand usage for every Thursday throughout 2007. Demand seems to be steady; there are no drastic uses in electricity. Some days are much lower than others, but this could be due to holiday or the facility work schedule. Immediately apparent is that 5 of these days did account for monthly demand peaks. If this were random, one would expect no more than two of these days to represent demand peaks; 5 days is suspicious and may indicate some industrial process that is taking place on these days.

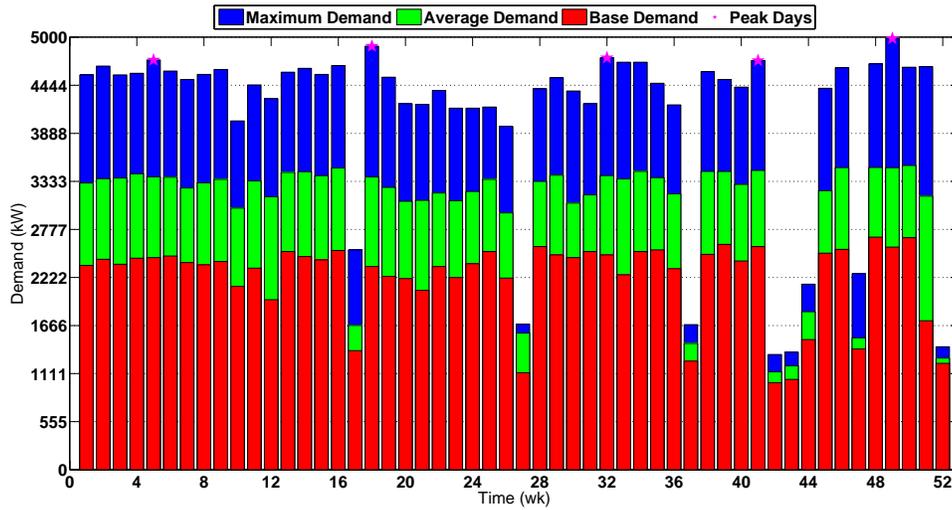


Figure 5.25: Reduced demand information for Thursdays throughout 2007.

A similar profile can be seen in Figure 5.26 which represents every Wednesday throughout 2008. The reduced electrical usage can be seen in the month of March as discussed previously. Again, 5 out of 12 demand peaks can be seen on this day, indicating that there are some processes particular to this day that is pushing the electrical demand to peak beyond most days.

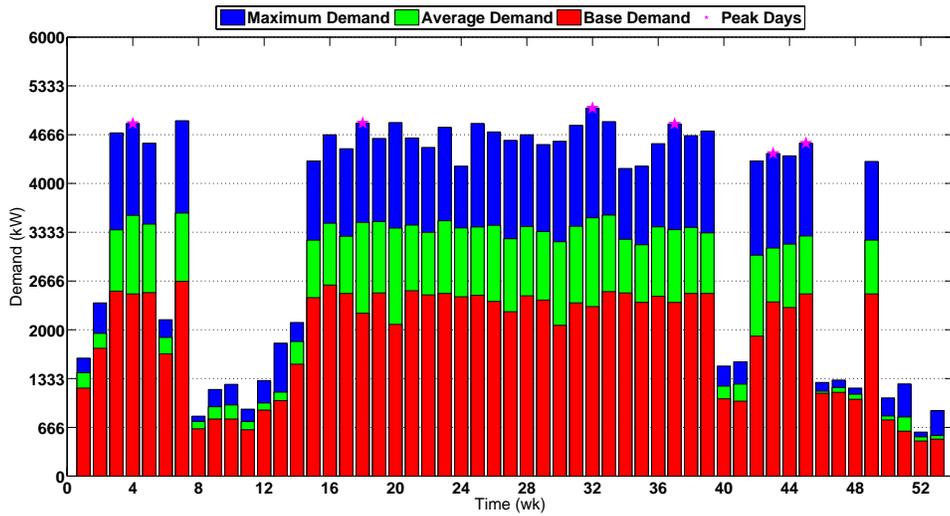


Figure 5.26: Reduced demand information for Wednesday throughout 2008.

Figure 5.27 and Figure 5.28 show a similar trend as those spoken of in the previous 2 figures. Wednesday in 2010 and Thursdays in 2011 show an abnormal number of demand peaks. Other than that there no odd usage as the maximum, average and minimum usage seems to be on par with the other days throughout the year.

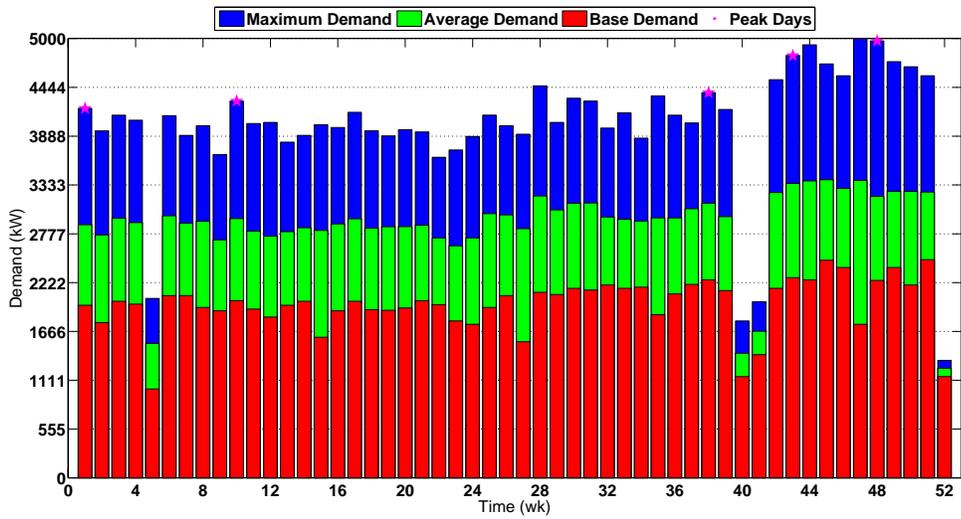


Figure 5.27: Reduced demand information for Wednesday throughout 2010.

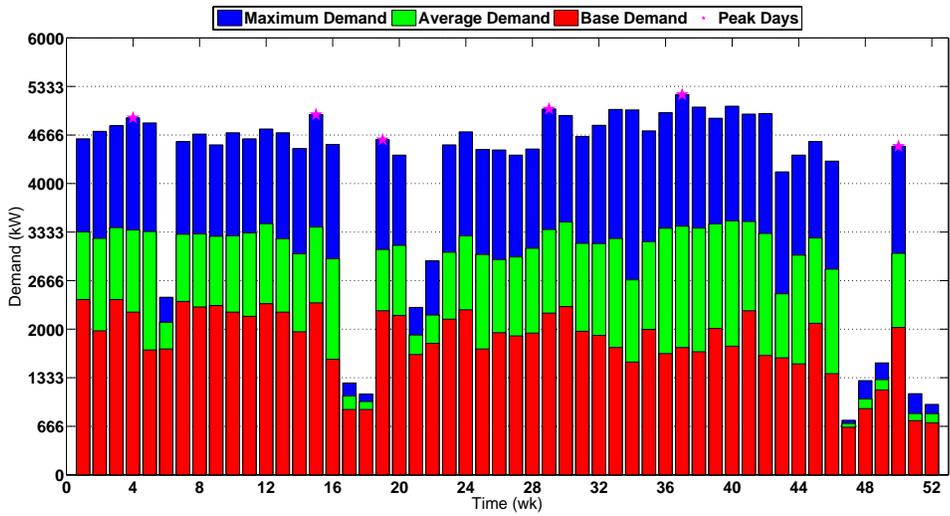


Figure 5.28: Reduced demand information for Thursday throughout 2011.

Table 5.12 shows some of the relevant figures, demand and consumption based variables that may be of benefit to the auditor.

Table 5.12: Key weekday variables for indicated years.

Year	2007	2008	2010	2011
Days	Thursday	Wednesday	Wednesday	Thursday
Maximum Demand (kW)	4,985	5,030	4,998	5,222
Base Demand (kW)	1,004	480	1,011	652
Total Demand (kW)	24,108	28,448	22,655	29,209
Total Demand Cost (\$)	308,110	363,565	289,543	373,298
Average Consumption (kW)	72,913	63,199	68,576	68,630
Average Consumption Cost (\$)	3,944	3,419	3,709	3,712
Total Consumption (kWh)	4,228,989	3,791,958	3,977,423	3,980,580
Total Consumption Cost (\$)	228,788	205,144	215,178	215,349
Peak Occurrences (%)	41	50	41	50

Demand Aberration

The demand aberration tool looks for daily electrical usage that lies outside of a statistical norm. It condenses the daily demand profile into the 1-norm to characterize statistical differences in demand peaks, and the 2-norm to characterize statistical differences in the average demand profile. It compares the 1-norm and 2-norm values with the same day of the week (i.e. every Monday, Tuesday, etc.) throughout the year to gauge which days are statistically different from each other. Figure 5.29 represents the 1-norm distribution for every Sunday throughout 2007.

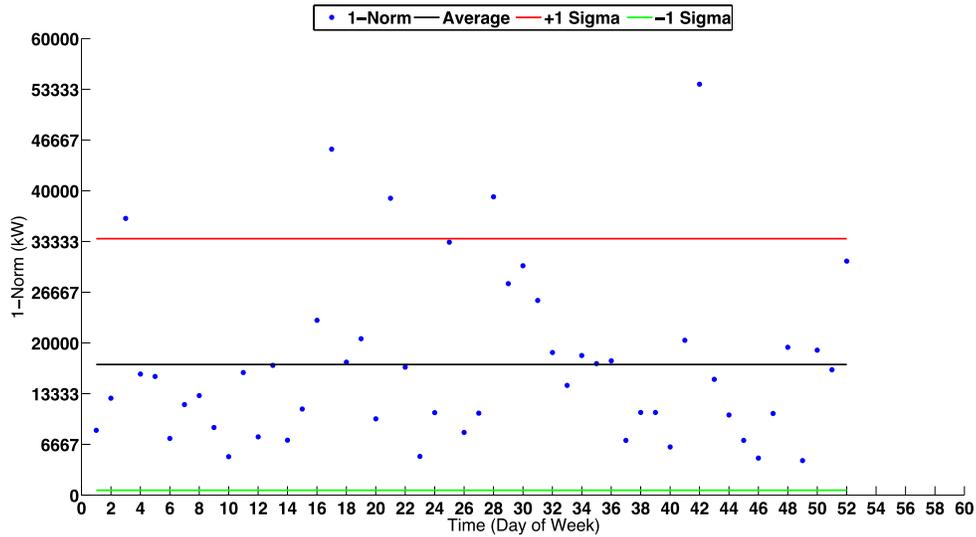


Figure 5.29: 1-norm demand space for Sundays 2007.

Similarly, Figure 5.30 shows the 2-norm distribution for every Sunday throughout 2007. The graph of the 1-norm distribution indicates that there are 5 days that fall out of 1 standard deviation from the average while the graph of the 2-norm indicates that there are potentially 6 days that have a non-typical demand profile. The user is able to change the number of days may be of concern by changing a standard deviation multiplier, thus expanding or contracting days that are of possible concern. In this case, both 5 to 6 days is a manageable number to view, thus the multiplier will be left as 1. If there were too many days that fell out of the standard distribution, then the multiplier would be increased to encompass more days. If there were only 1 or 2 days, the multiplier would be decreased to contract the standard deviation. Looking at Figures 5.29 and 5.30, it is evident that the distribution is large and spread out, thus the standard deviation is expected to be large as well. Also evident is that there do not seem to be

aberrational points that are below -1 sigma for either distribution. This is more likely due to the fact that the distributions are not completely Gaussian.

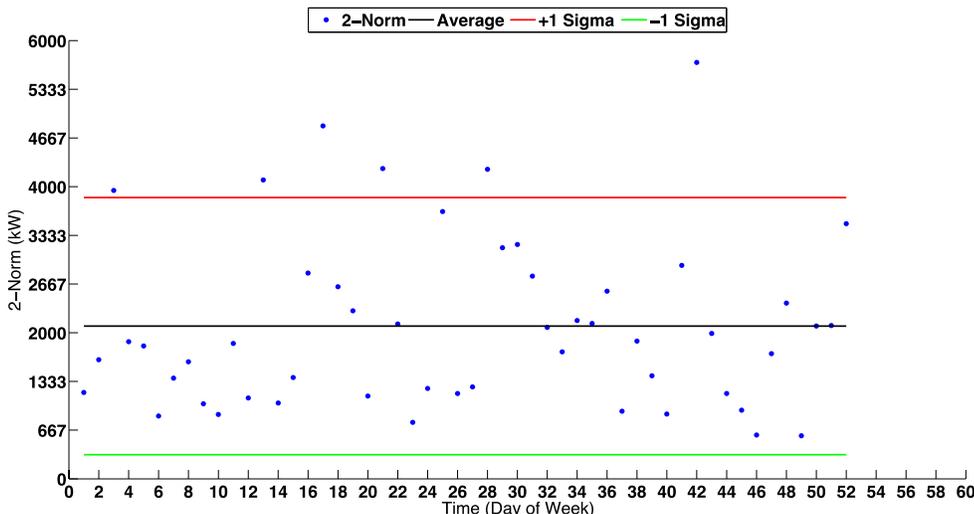


Figure 5.30: 2-norm demand space for Sundays 2007.

Table 5.13 shows some of the variables that characterize the 1-norm and 2-norm distributions, including averages and standard deviations.

Table 5.13: Key demand aberration variables.

Median Demand (Average Demand Curve) [kW]	1,573
Average Consumption (kWh)	37,311
Average Consumption Cost (\$)	2,019
Average 1-Norm Demand (kW)	17,168
Standard Deviation of 1-Norm Demand (kW)	11,016
Average 2-Norm Demand (kW)	2,091
Standard Deviation of 2-Norm Demand (kW)	1,173

Looking over the course of the dataset, from 2007 to 2012 for every day of the week through out every year, it became clear that most of the days that fell out of statistical norm for average demand profile (looking at 2-norm) did so because there was some point throughout the day that showed a significant reduction in electrical usage Figure 5.31 represents July 15, 2007, a Sunday that fell out of norm from the rest of the Sundays throughout 2007. A visual representation of the demand profile in comparison to the average profile shows that between 8 and 11, demand fell drastically and was not continuous in comparison to the average demand profile for all Sundays throughout the year, which caused to fall out of norm.

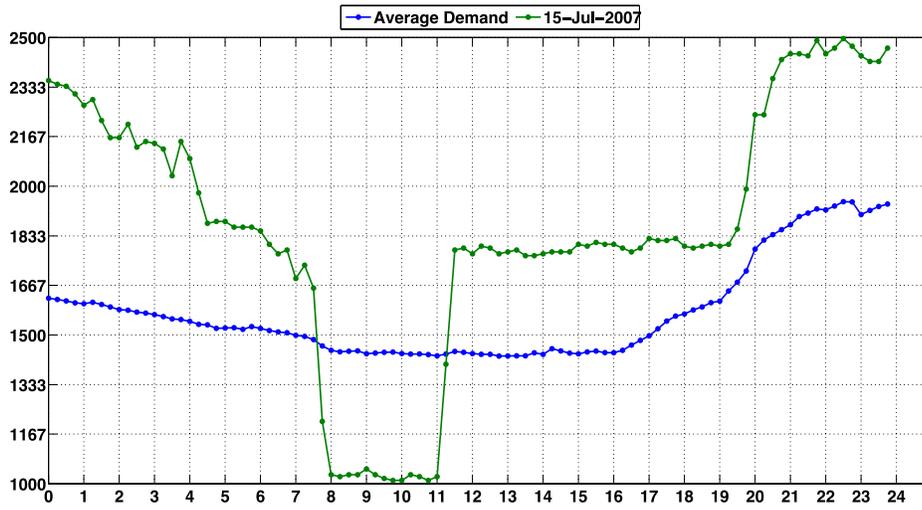


Figure 5.31: Comparison of demand curve for July 15, 2007 and average demand curve for all Sundays.

Table 5.14 shows some of the electrical consumption variables that took place on July 15, 2007.

Table 5.14:: Daily consumption variables for July 15, 2007.

Consumption (kWh)	44,113
Consumption Cost (\$)	2387

Figure 5.32 represents the demand profile for January 21, 2007, one of the days that fell out of demand peak (looking at 1-norm) norm. In comparison to the average profile for all Sundays throughout 2007, the peak was larger, but did not account for the monthly demand peak, meaning that the facility did not get charged for this particular days electrical usage. From the yearly demand visualizations, most days that accounted for

demand peaks showed a maximum power draw of approximately 5,000 kW; this Sunday shows a power draw of less than that.

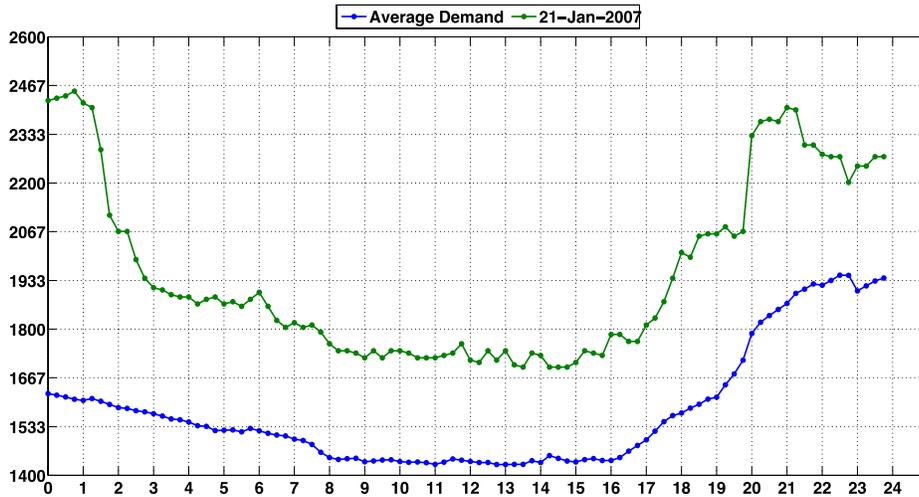


Figure 5.32 Comparison of demand curve for January 21, 2007 and average demand curve for all Sundays.

Table 5.15 shows relevant demand information pertaining to January 21, 2007.

Table 5.15: Daily demand variables for January 21, 2007.

Peak Demand	2,451
Peak Demand Difference	503
Monthly Peak	No

Figure 5.33 shows the 1-norm distribution for every Monday throughout 2008. A visual inspection shows that 17 out of 52 days fell out of norm, with most days hovering around an average value of 96,417 kW. In March of 2008, approximately 3 weeks of downtime were observed, as seen in Figure 5.19. March of 2008, out of any other month through out the data set represents the strongest non-traditional use of electricity. The

demand aberration tool has accurately identified the day in which the demand peaked in March 2008 for further scrutiny.

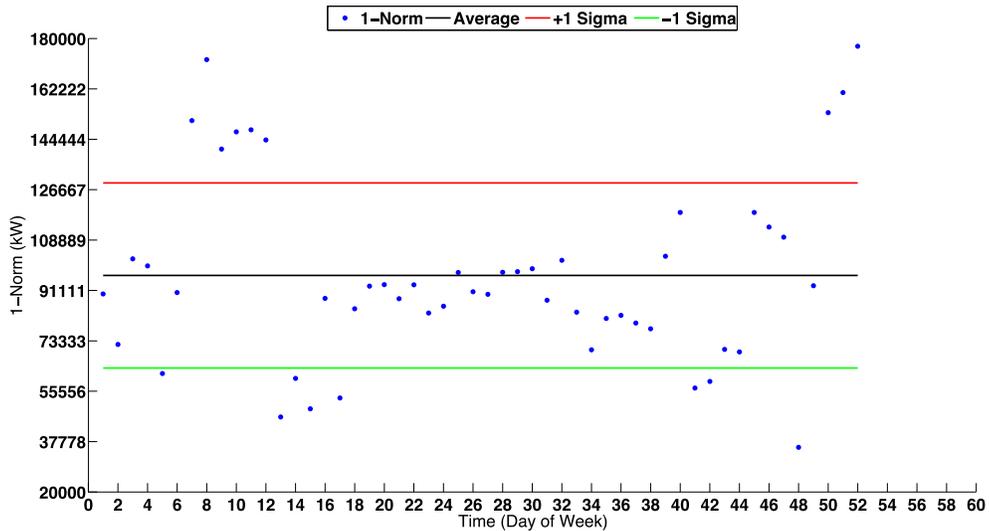


Figure 5.33: Visualization of 1-norm demand space for Mondays 2008.

Table 5.16 shows some of the statistical information relevant to the 1-norm distribution for all Mondays throughout 2008.

Table 5.16: Key 1-norm variables for all Mondays throughout 2008.

Median Demand (Average Demand Curve) [kW]	2,376
Average Consumption (kWh)	56,545
Average Consumption Cost (\$)	3,059
Average 1-Norm Demand (kW)	96,417
Standard Deviation of 1-Norm Demand (kW)	32,650

After identifying aberrant days, the tool allows one to compare the average profile demand profile for all Mondays through out 2008 and the aberrant profile, as

seen in Figure 5.34. One can see that the demand peak for March 31, 2008 is drastically different than the average demand profile. The demand profile for the day in question seems random, although there does seem to be some element of usage as demand increases at around the same time the facility is supposed to be operating. Even though this day does represent a demand peak, in comparison to other months through out the same year, the demand peak is less than half of average, meaning that it is not something that should be of concern.

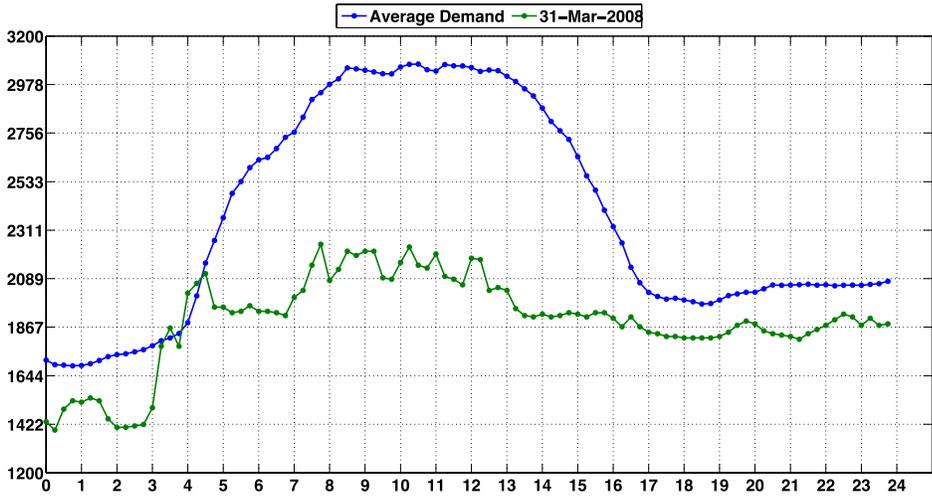


Figure 5.34: Comparison of demand profile for March 31, 2008 and average demand profile for all Mondays throughout 2008.

Table 5.17 shows some of the relevant statistics for March 31, 2008.

Table 5.17: Key demand variables for March 31, 2008.

Peak Demand	2,246
Peak Demand Difference	825
Monthly Peak	Yes

A closer examination of the day using the daily demand visualization tool as seen in Figure 5.35 reveals that demand peaked at 8:00 AM, which is within reason as demand for most days peaks between 8 AM and 10 AM. Other than the unusual profile, there are no other factors that would indicate March 31, 2008 as a day to scrutinize about with plant management.

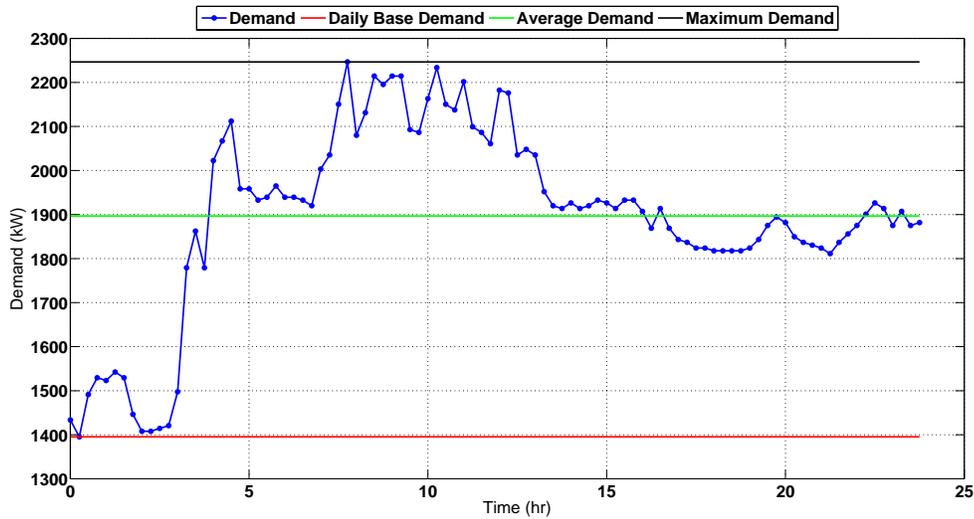


Figure 5.35: Daily demand profile for March 31, 2008.

Table 5.18 shows some of the relevant electrical usage statistics for March 31, 2008.

Table 5.18: Key electrical variables for March 31, 2008.

Maximum Demand (kW)	2,246
Average Demand (kW)	1,896
Daily Base Demand (kW)	1,395
Consumption (kWh)	45,105
Consumption Charge	2,440
Peak Time	8:00

Weather Disaggregation

The weather disaggregation tool is an attempt to separate weather based electrical demand from the aggregate demand dataset. Figure 5.36 represents an attempt to disaggregate weather-affected loads for the facility. The dataset is specific for 2010 and the temperature threshold was set at 65°F. The graph shows that, on average, there was more electrical demand for hours of the day that were above 65°F than below. The morning and the afternoon exhibit a close relation because the plant is not operating during these hours.

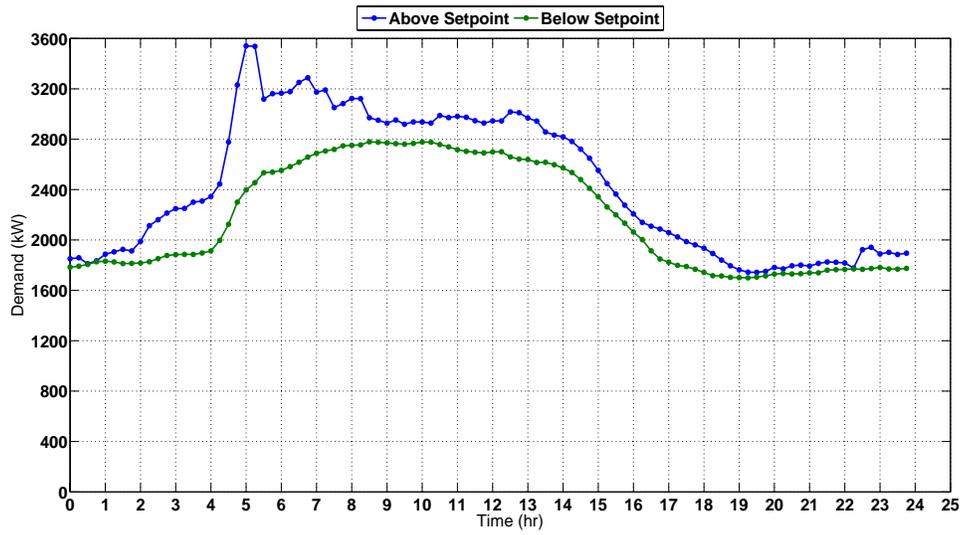


Figure 5.36: Weather separated, average demand profiles for 2010.

Figure 5.37 shows the demand difference for the average demand curves above the set point (green curve) and below the set point (blue curve). The model presupposes that the difference in demand is due to weather effects, thus the largest demand gap due to weather was found at around 5 AM.

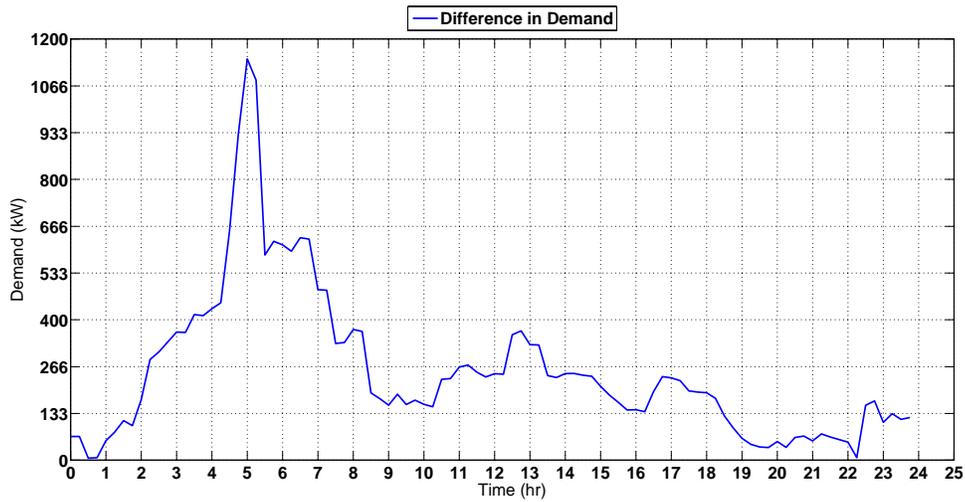


Figure 5.37: Difference in demand profiles for 2010.

Table 5.19 gives data pertaining to electrical consumption and cost for the increase in demand.

Table 5.19: Weather based consumption variables for 2010.

Average Daily Consumption (kWh)	6,084
Average Daily Consumption Cost (\$)	19,794

An understanding of the type of equipment present in the facility can help to diagnose the accuracy of the model. It is natural to assume that a facility whose electrical usage is dominated by HVAC during the summers will show a significant increase in electrical demand, and even though the tool does focus on HVAC usage, it can also be used to diagnose other equipment whose operations are sensitive to the weather. In this

case, a majority of the electrical usage was strewn between several large compressors, a host of motors and a series of annealing ovens. Figure 5.38 represents the weather disaggregated demand curves for 2009 at a temperature threshold set point temperature of 65°F. We see that this facility, during this particular year, on average consumed more electricity when it was colder. This makes sense in areas that experience severe cold temperatures, such as this plant. Electrically driven furnaces may have to increase their power draw to account for the added heat loss, even in light of insulation. In this case, the furnaces were heated with natural gas, so it is unclear why they tend to use more power when it is colder. It may be that, for some reason, their business increases in the winter. Whatever the case may be, the separation between both curves is not drastic. Figure 5.39 exemplifies the difference between both demand curves.

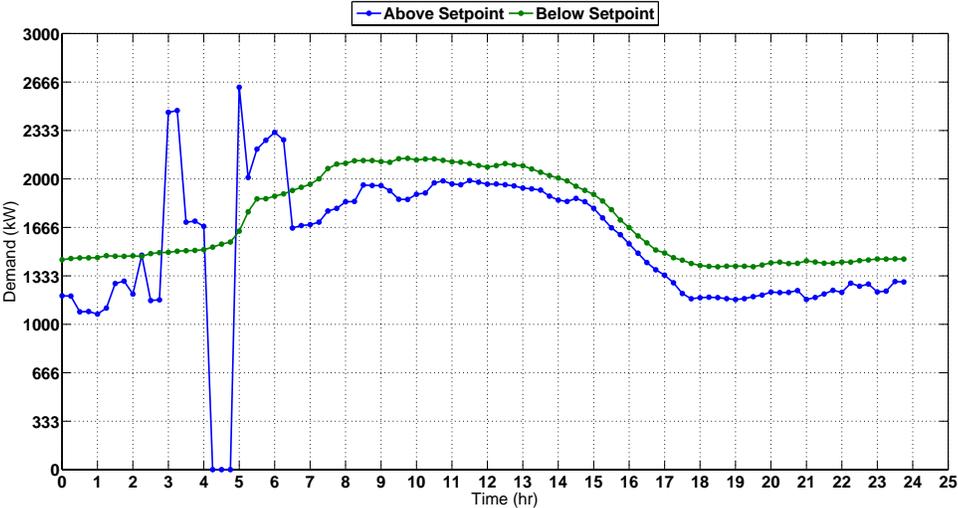


Figure 5.38: Weather separated, average demand profiles for 2009.

On average, the difference between both demand curves is right around 333 kW, a fraction (approximately 6%) of the average 5,000 kW they typically use. Even though the disaggregation is too close to definitively conclude weather based demand usage, it does give the auditor an idea of how electricity is used as a function of temperature. Such information could be use to interview personnel and perhaps conclude that weather effected equipment be outfitted with more efficient counterparts.

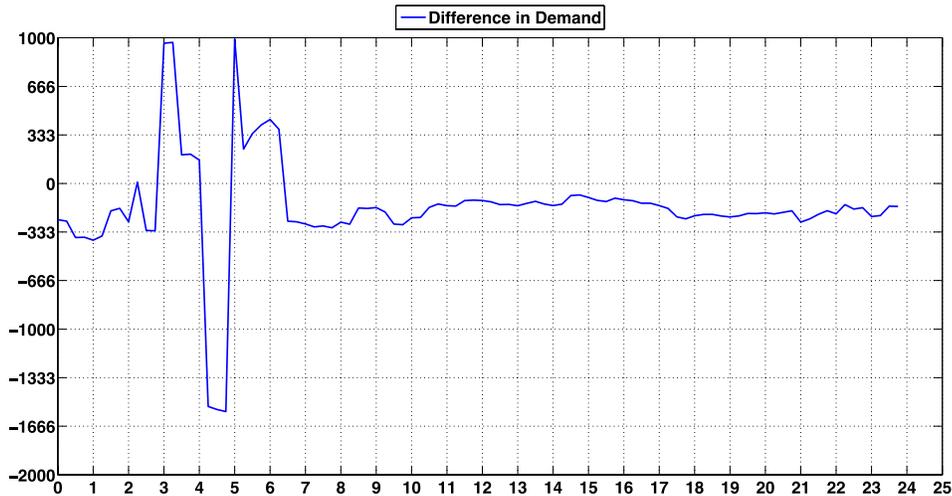


Figure 5.39: Difference in demand profiles for 2009.=

Table 5.20 gives data pertaining to electrical consumption and cost for the increase in demand.

Table 5.20: Weather based consumption variables for 2009.

Average Daily Consumption (kWh)	3,855
Average Daily Consumption Cost (\$)	12,514

Demand Scheduling

The demand-scheduling tool is designed to analyze savings associated with reducing demand. This is accomplished by shifting energy intensive production operations from peak hours to non-peak ones. Figure 5.40 show three different types of demand. The blue bars represent the facilities current demand peaks for 2010. The green colored bars show facility demand after a 10% reduction from the average demand. The red bars show the lowest reachable demand, assuming no change in electrical consumption and a 24-hour operation.

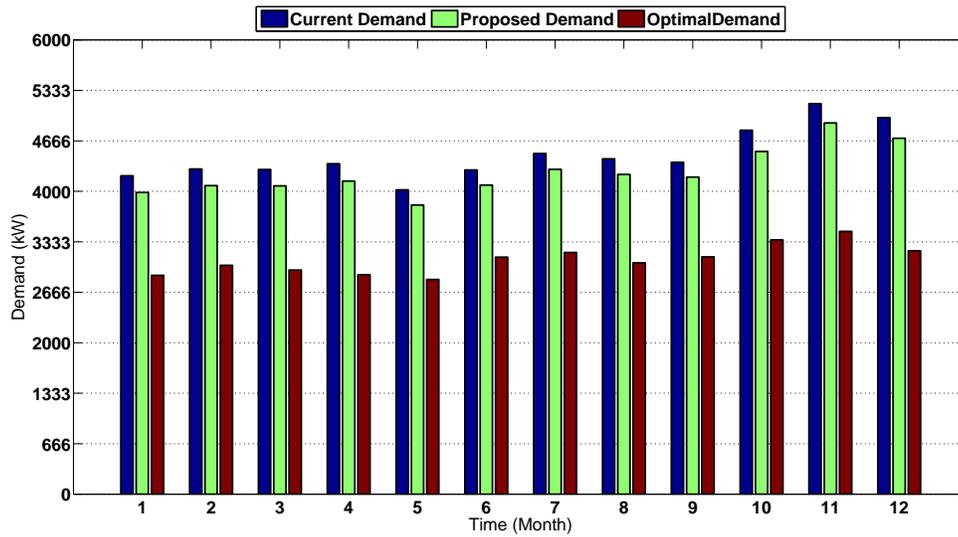


Figure 5.40: Demand scheduling variables for 2010

Figure 5.41 exemplifies the cost associated with the current, proposed and optimal demand peaks.

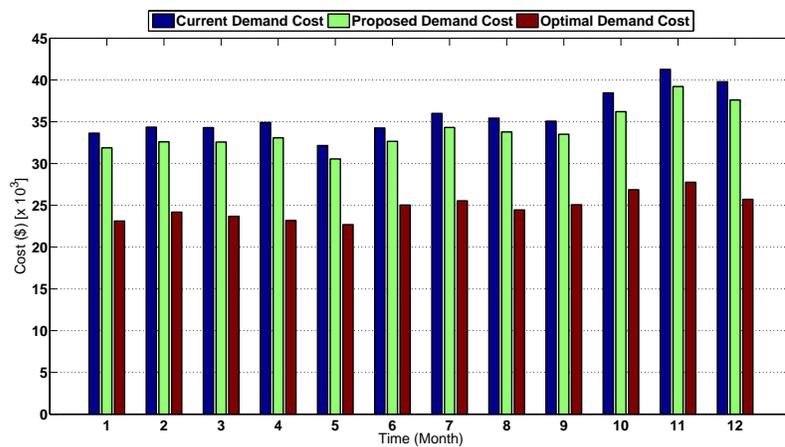


Figure 5.41: Demand scheduling cost variables for 2010.

Table 5.21 shows the total annual demand costs for 2010, as well as the total savings for the proposed 10% demand reduction and the optimal demand usage.

Table 5.21: Total demand scheduling variables for 2010.

Current Annual Demand Cost (\$)	429,619
Proposed Annual Demand Cost Savings (\$)	21,642
Optimal Annual Demand Cost Savings (\$)	132,329

Since this particular plant is not running a 24/7 operation, it is more likely that a demand reduction of 10% will reduce demand associated costs, rather than opting to run on a 24 hours schedule to reap the benefit of an optimal demand time frame.

Photovoltaic Analysis

The photovoltaic (PV) analysis tool is designed to give a reasonable prediction on the metrics associated with installing and using solar power. This is done by coupling a facilities IDR demand data with TMY3 solar data. The following analysis was done for 2007. Figure 5.42 shows the savings that are to be generated by using solar power only. Savings are reflected in 2 categories; electrical consumption savings and electrical demand savings. One can see that there are 2 curves; one from 0 to 10 km² and another from 10 to 40 km². The curve seen from 0 to 10 km² is the aggregate savings from demand and consumption. The figure shows that a 50,000 m² will be necessary to generate enough electrical savings as is currently being spend on electricity.

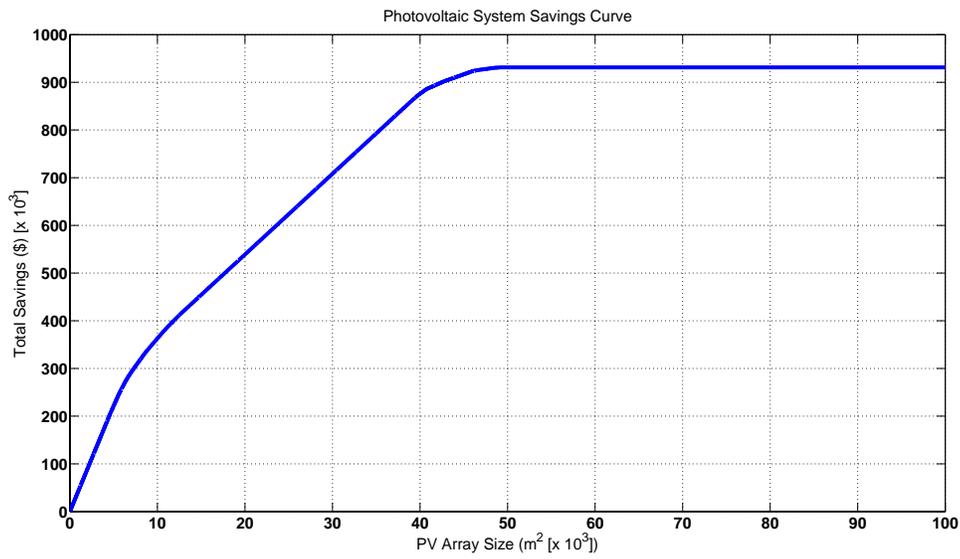


Figure 5.42: Savings associated with PV array of variable size.

Figure 5.43 shows the installed cost of the system for varying system sizes. A 50,000 m² will be necessary to generate enough electrical cost savings as is currently being spent on electricity at a cost of approximately \$40,250,000.

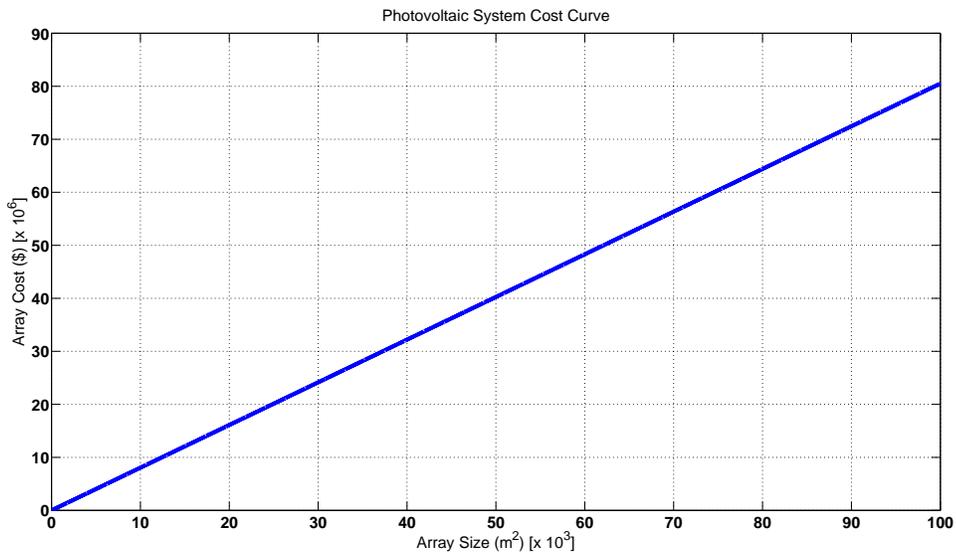


Figure 5.43: Array cost associated with PV array of variable size.

Figure 5.44 shows the payback period associated with the PV system of varying sizes, thus a 50,000 m² system will payback in 43 years.

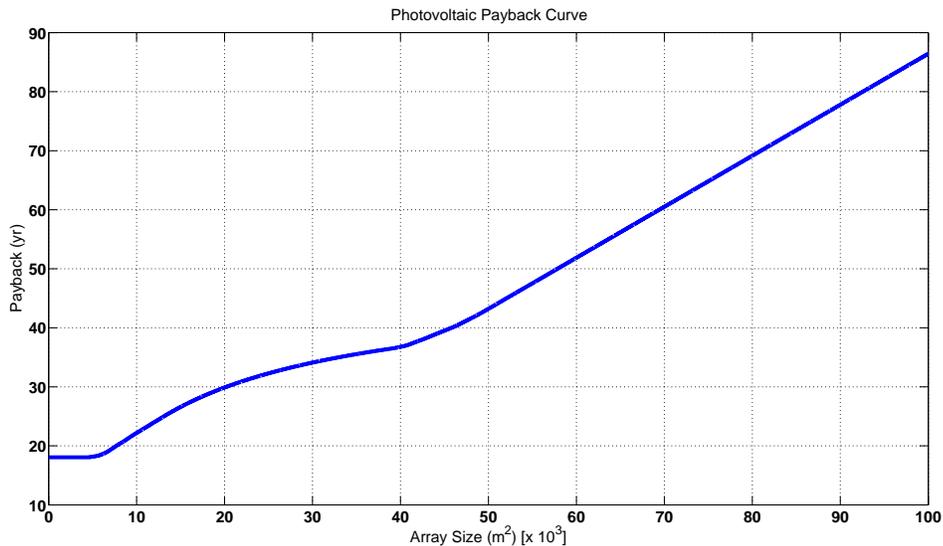


Figure 5.44: Payback period associated with PV array of variable size.

Figure 5.45 depicts a pie chart of when solar power can be used to meet the facility demand. Given every 15-minute demand data point, 56% of those points show that solar power generated from PV is enough to meet facility demand and 44% of the time, solar power cannot meet a facilities electrical demand. While this may seem to be unacceptable, the percentage of unmet demand is due to the daily base demand found during the evening, when the sun is below the horizon. Figure 5.46 show how solar power meets monthly demand peaks.

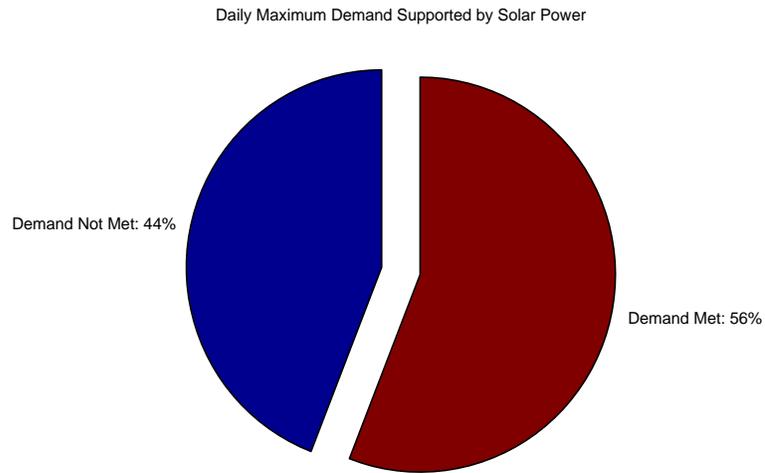


Figure 5.45: Percent demand to be met with PV system.

For every month, except March, a 50,000 m² system is more than enough to meet the monthly demand peaks. The facility may also want to know what days of the week demand is met. This may be important to show for a facility that only runs 5 days a week and may not care about demand being met on weekends. In this case, the facility runs on a 5 day per week basis. Figure 5.47 shows the percentage of demand being met for every day of the week throughout the year.

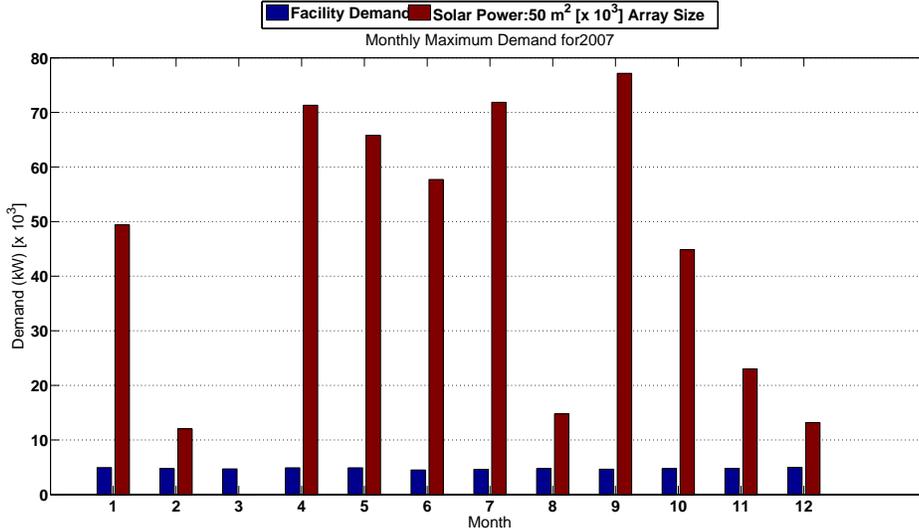


Figure 5.46: Facility monthly demand peak and demand to be generated at the same time.

If you assume, on average 12 hours of daylight and 12 hours of nighttime, demand should not be met at least 50% of the time, since the sun is below the horizon during half of the day and no power is being generated from the PV panels. Demand being met over 50% is associated with daylight hours, when the facility is operational, thus anything over 50% exemplifies that the panel are enough to meet the facility electrical demand. The largest gap can be seen on Sunday (day 1) because the facility is non operational and the facility electrical demand is so low. There may be a problem on Mondays and Thursdays since the percent of the facility demand being met is under 50%. All other days seem to be fine. Figure 5.48 shows the daily maximum demand for the facility, juxtaposed against the solar power being generated at the same time.

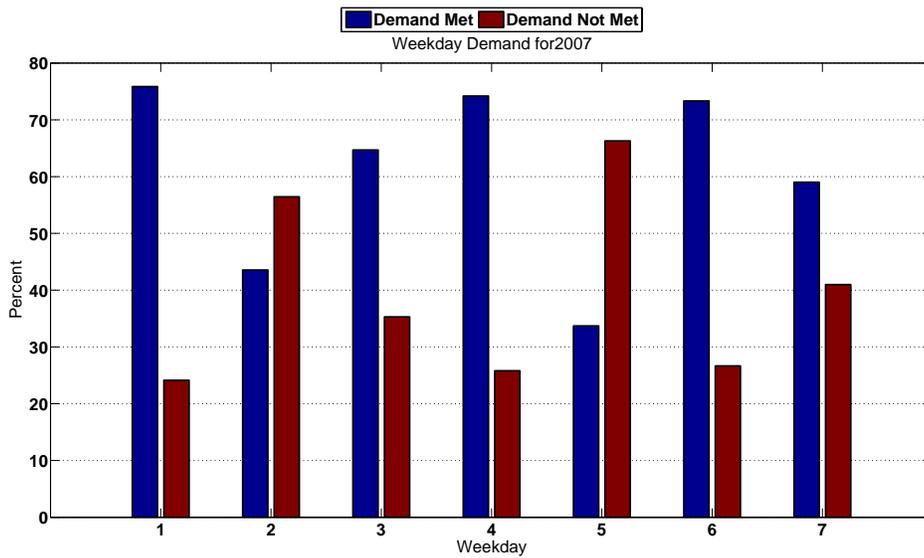


Figure 5.47: Percent of demand to be met on a day per week basis.

With the exception of a couple of weeks in February and a couple of weeks at the end of October, the figure shows that more than enough electrical demand is being generated for most days throughout the year.

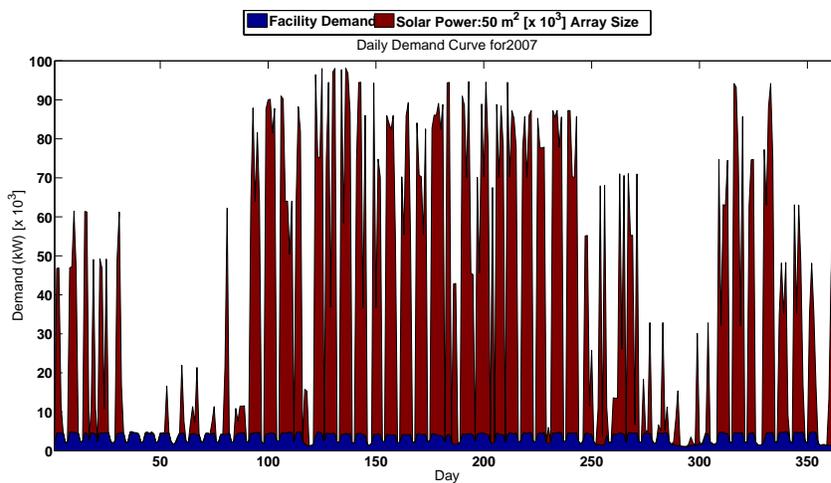


Figure 5.48: Daily facility demand max and solar demand max at same time.

In addition to a demand analysis, the photovoltaic analysis tool can also be used to analyze PV consumption parameters. Figure 5.49 shows that for every day throughout the year, a 50,000 m² system can meet facility consumption 92% of those days.

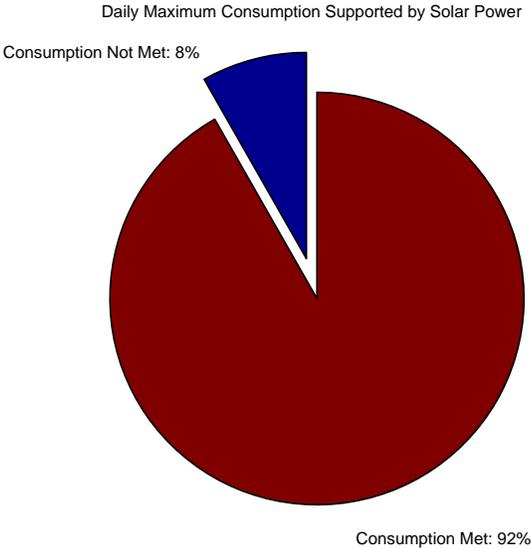


Figure 5.49: Percent consumption being met for every day of the year.

Figure 5.50 show the total monthly electrical consumption for the facility juxtaposed against the electrical energy produced by the PV system. The size of the system is more than adequate to meet the total consumption for the month. In addition, the facility could opt to sell electrical energy not being used, thus lowering their payback period since they are profiting from the excess amount of energy generated.

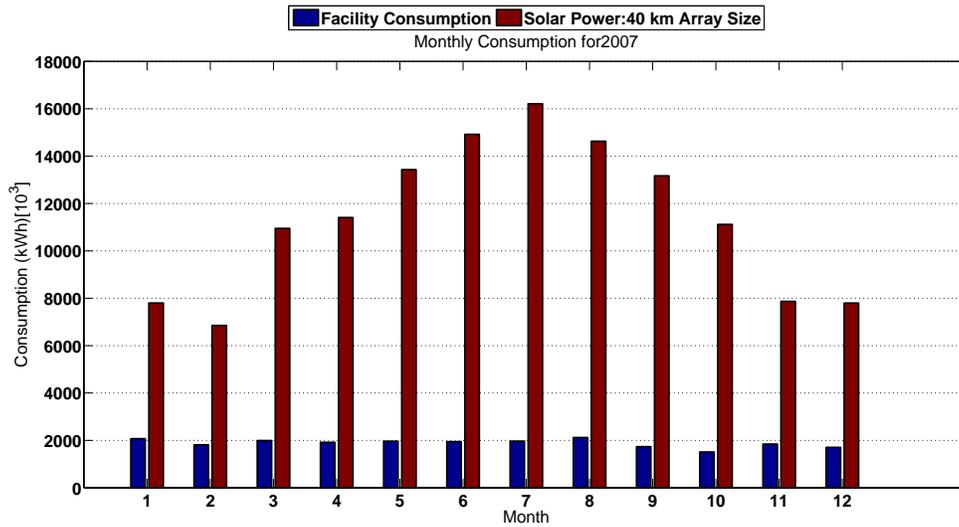


Figure 5.50: Total monthly consumption of facility juxtaposed against the monthly energy generated through PV.

Figure 5.51 shows the percent of consumption met for every weekday throughout 2007. Everyday shows that consumption is met over 80% of the time. Figure 5.52 is an overlay of the facility electrical consumption against the PV system energy generation.

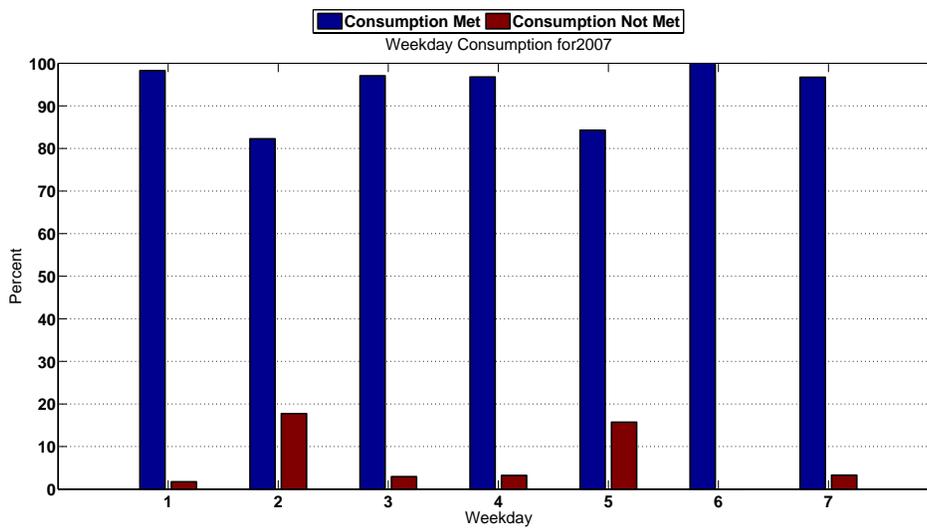


Figure 5.51: Demand scheduling variables for 2010

Most days, especially during spring and summer months show that the system is adequate to meet the facilities needs. Some winter days exemplify electrical energy generation below 100,000 kWh which probably accounts for the consumption not met seen in Figure 5.49.

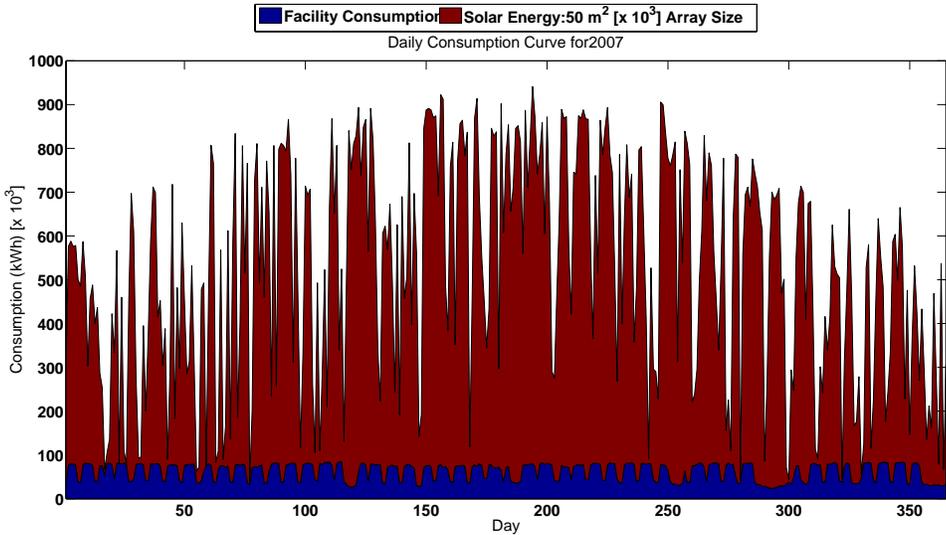


Figure 5.52: Comparison of solar power output and facility demand for 2010.

If the facility sold the amount of unneeded electrical energy at a rate of $\frac{\$0.06}{\text{kWh}}$, the 43 year payback period could be reduced to a 3 year payback.

6. CASE STUDY 2

Introduction

This chapter will attempt to re-examine some of the demand analysis tools in light of the previous demand data sets functional shortcoming. The following analysis is made possible through a variety of demand data sets obtained by the TAMUIAC across a variety of industrial and commercial clients.

Demand Visualization

The previous chapter attempted to highlight the power of the demand visualization tool for a facility that inherently had no drastic or overt demand or consumption usage defects. This thesis will again use the demand visualization tool to show just how demand visualization can be powerful in ascertaining inconsistent and erroneous usage in electrical demand. The following data set was obtained from a TAMUIAC client situated near Houston, TX. This particular facility ran a 24 hour operation, shutting down for 36 hours per month for maintenance and repair.

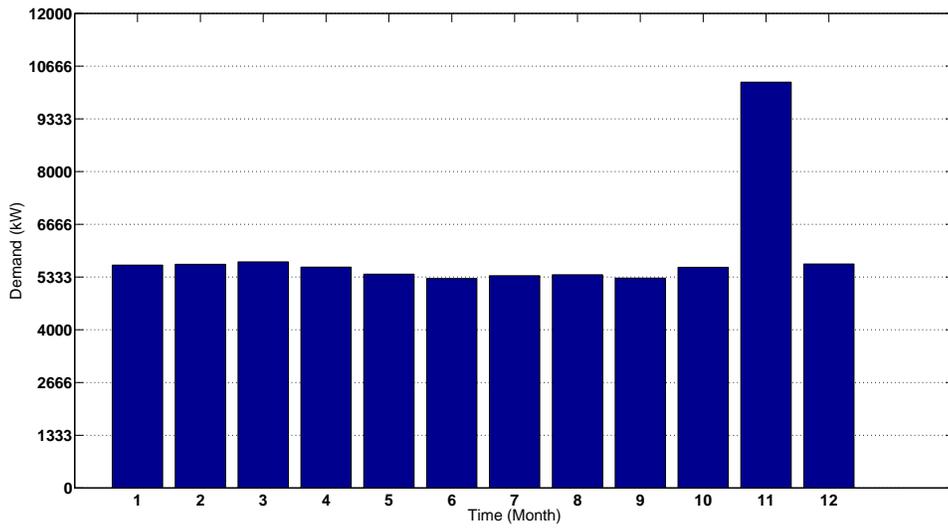


Figure 6.1: Maximum demand per month for 2012.

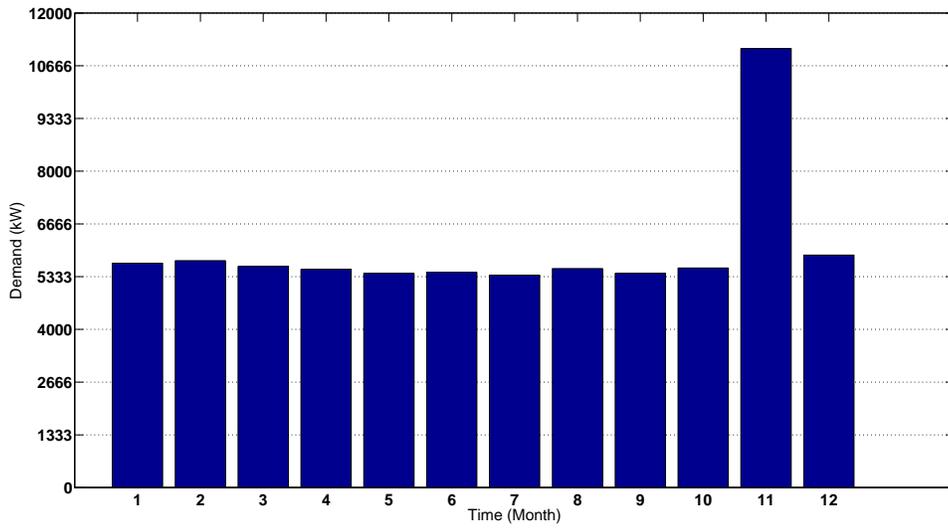


Figure 6.2: Maximum demand per month for 2013.

Figures 6.1 and 6.2 detail a facility whose electrical usage is very consistent; consistently drawing power of about 5,333 kW, except for November. During this time of the month, the demand suddenly jumps to 10,666 kW, twice as much as the typical monthly power draw. The drastic increase in demand is inconsistent with how this facility should be using electricity. Facilities that run on a 24 hour basis are usually very stable and consistent in their electrical usage; there are no sudden changes in electrical demand because the operating hours don't change. These facilities typically run at full load all of the time. This is also exemplified in Figures 0.3 and 0.4 where electrical demand is fairly constant throughout the months.

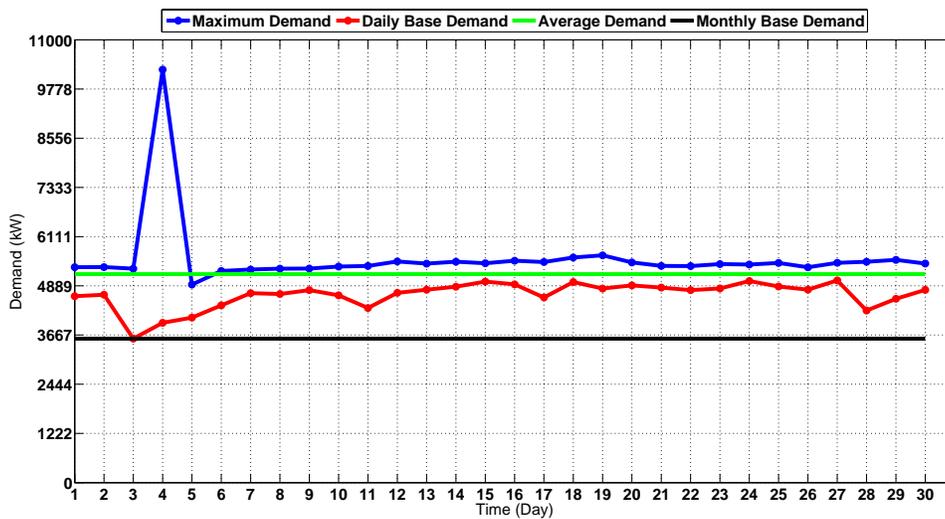


Figure 6.3: Monthly demand curves for November 2012.

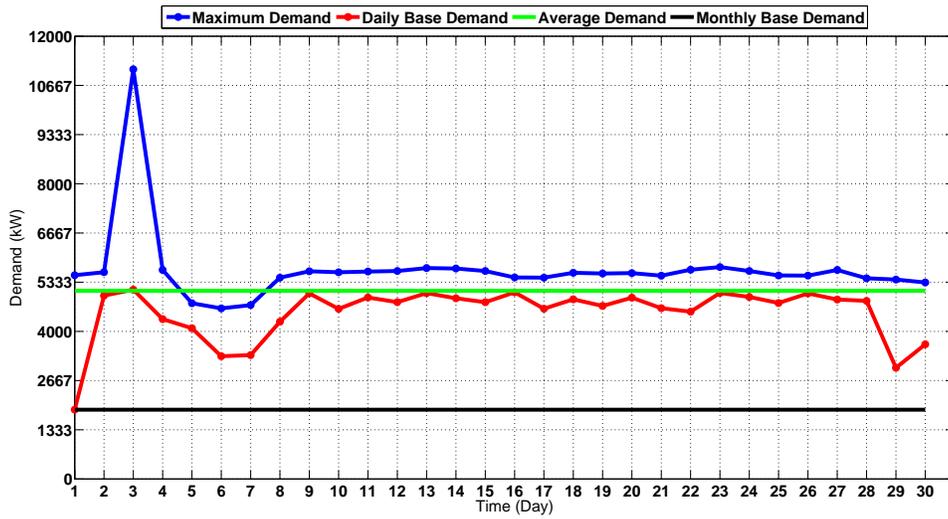


Figure 6.4: Monthly demand curves for November 2013.

The figures do, however show this drastic use in electricity on the 4th of November in 2012 and the 3rd of November in 2013, both Sundays. Electrical usage for both years throughout the rest of the month is smooth.

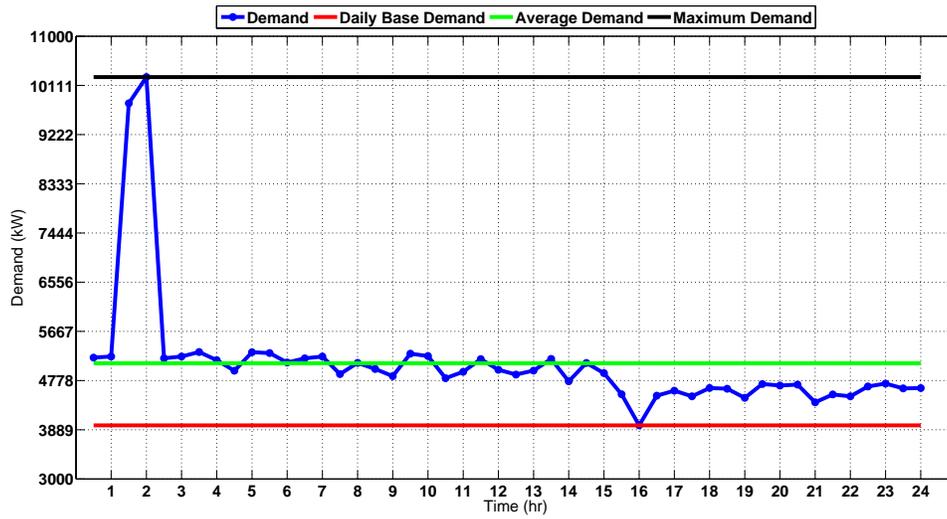


Figure 6.5: Daily demand curves for November 04, 2012.

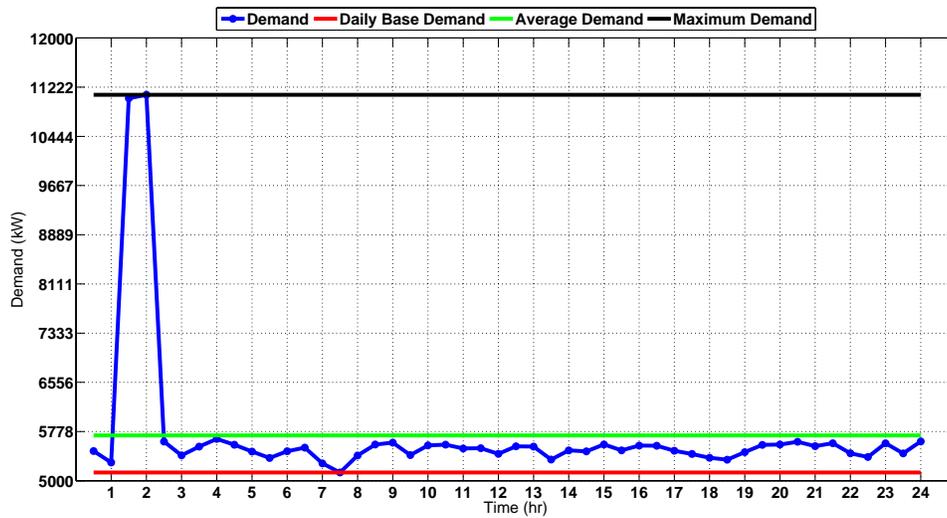


Figure 6.6: Daily demand curves for November 03, 2013.

Figures 6.5 and 6.6 exemplify the daily demand curves for both Novembers in 2012 and 2013. Demand for the facility jumps to double its usual value at 2 AM in the morning. These jumps represent a cost of \$40,000 and could possibly be mitigated by understanding and probing plant management of the plant operations that are occurring during this time of year.

After an investigation of the yearly occurrence on both of these Sundays, it was determined that the end of daylight savings occurred on both of these days. To compensate for the change in time, the electrical provider was aggregating the demand data point for the same time stamp, effectively doubling the demand. In this case, the provider was aware of the practice and compensated by removing the doubled charge on the utility bill. The demand visualization tool was able to detect a possible aberration in demand usage, and determine the exact time of occurrence. The tool gave TAMUIAC personnel the information necessary to inquire with plant management and inform them of the aberrant demand usage. If this facility did end up paying for the erroneous usage in demand, the tool would have been responsible in saving the plant hundreds of thousands of dollars in the coming years.

In addition to the demand peaking issue in November, the visualization tool also discovered a possible issue with electrical demand usage on a day per week basis.

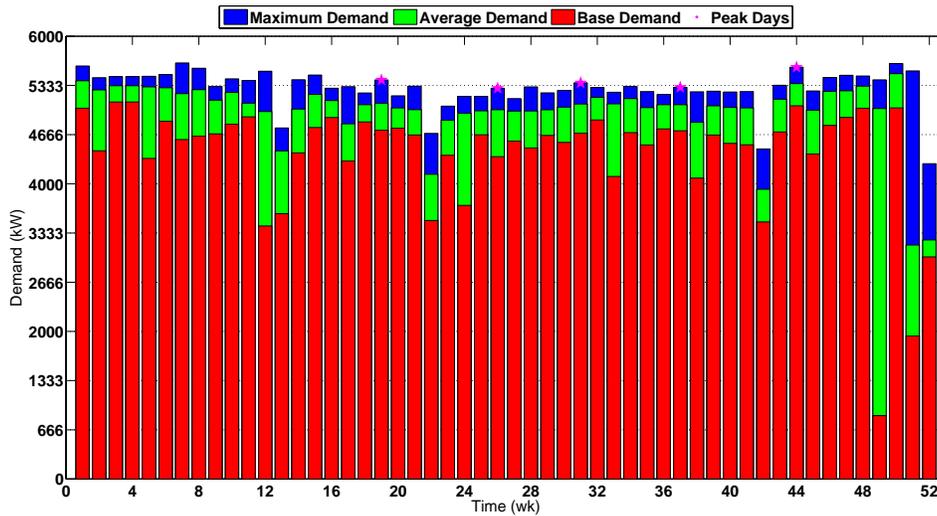


Figure 6.7: Day per week demand curves for Tuesdays in 2012

Figure 6.7 shows that for every Tuesday throughout 2012, almost half of days exemplifying demand peaks took place on a Tuesday. In this instance, there may be some manufacturing practice that is occurring on this day that is causing demand to peak. The tool is helpful and gives inquiring parties some necessary information to track down industrial processes that are causing demand to peak. For a facility that is running a 24 hour operation, it may be possible to shift the demand to other hours of operation to reduce demand peaking.

Demand Aberration

After determining obvious demand irregularities using the demand visualization tool, the demand aberration tool can be used to determine days in which demand aberrations are not so apparent

Other than the aberrant demand peaks that occurred in November, the tool shows that most aberrant days were classified as such because of a reduced usage in electricity in comparison to the average daily usage characteristic.

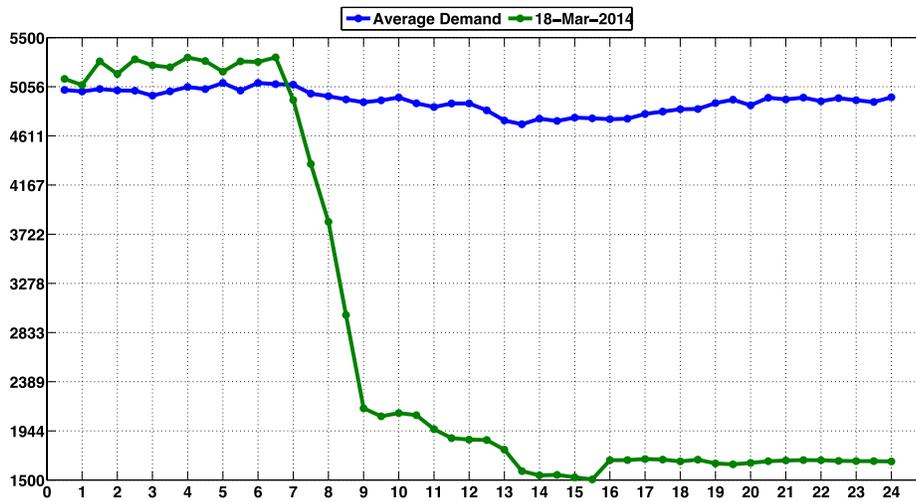


Figure 6.8: Indicated date classified as having aberrant demand peaking behavior.

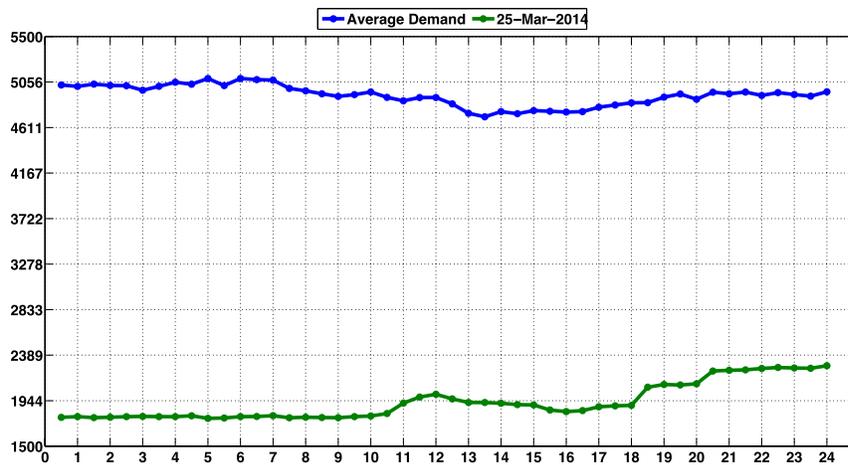


Figure 6.9: Indicated date classified as having an aberrant demand profile.

Figures 6.8 and 6.9 are two examples of days which were classified as aberrant. Most days which fell out of statistical norm with the average demand peak and average demand profile did so because their electrical usage was much lower than the average demand usage characteristic, similar to those exemplified in the Figures 0.8 and 0.9.

Photovoltaic Analysis

The following PV analysis was done for a TAMUIAC client seeking to augment their office with PV based power generation. TAMUIAC personnel were furnished with whole facility 15-minute demand data. In an attempt to disaggregate the office-based load from the whole facility load, it was assumed that the office accounted for 10% of the whole facility load. Additionally, since the office hours were significantly reduced in comparison to the 24 hour operation of the manufacturing side, only daylight hours were considered. The chosen system size was determined by sizing the system area required to produce enough electrical demand and consumption savings as is currently being spent.

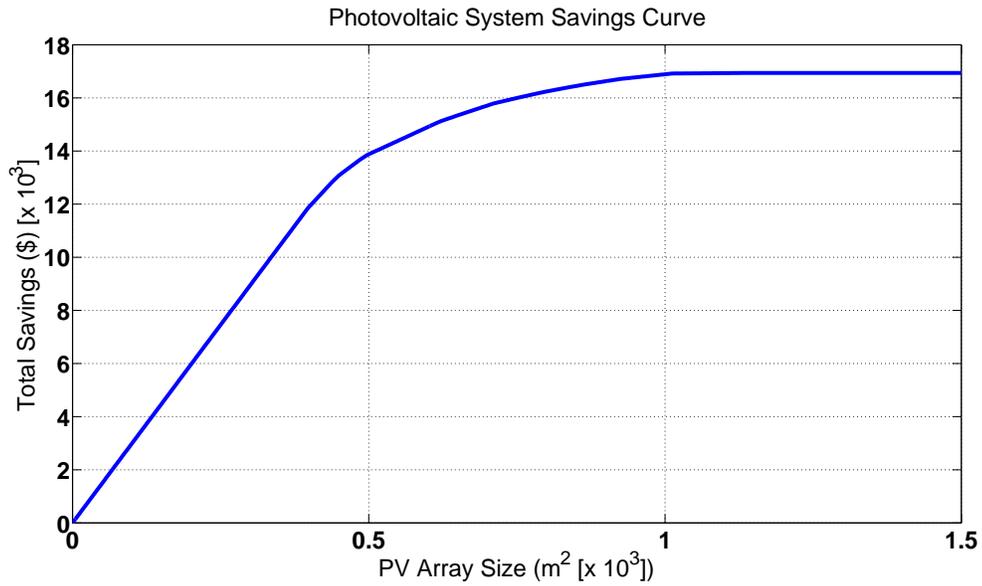


Figure 6.10: PV system savings curve

Figure 6.10 relates the amount of savings to be had from a PV system of a particular size and 15% efficiency. PV panels come with different power ratings (usually 200 W to 300 W per panel), but even though a panel is rated for a specified maximum power supply, it will only supply a fraction of the solar power it is receiving. For this reason, we scale a system over the area required to meet the needs of the facility. In this case, to receive savings comparable to what would have been spent on electricity, in both electrical demand and consumption, a PV system encompassing an area of 1,000 m² would be required.

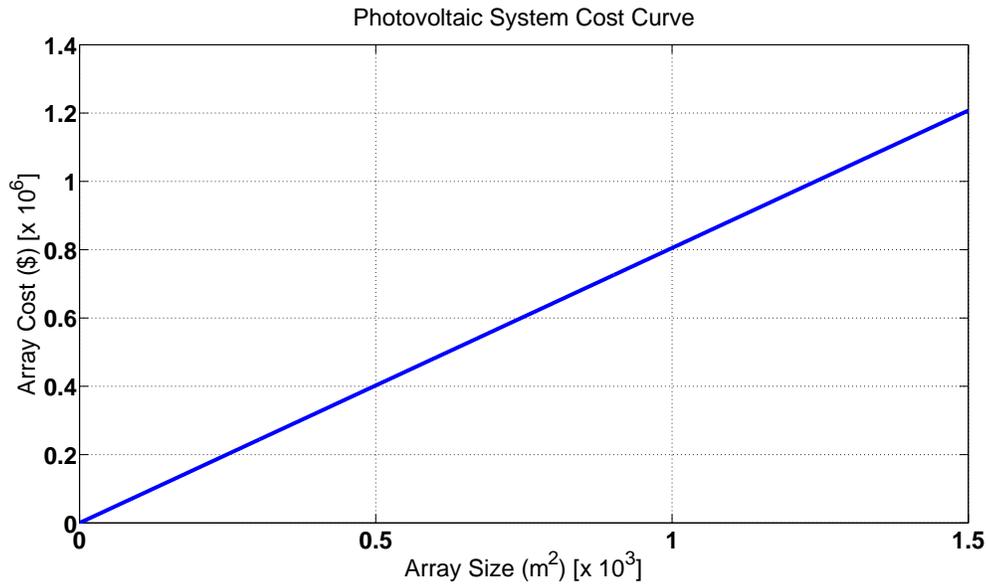


Figure 6.11: PV system cost curve

Figure 6.11 displays what the anticipated cost of a PV system encompassing various areas. Systems are typically sold on a dollar per kW basis. A typical value for larger industrial systems is \$4,600 per kW installed. A higher efficiency panel, at maximum, will produce 0.174 kW per m², thus a system encompassing 1,000 m² will cost \$800,400

P

Figure 6.12: PV system cost payback.

Figure 6.12 displays the projected simple payback for systems of various sizes. Since the avoided cost of consumption and the avoided cost of demand are so low in comparison to the cost of the PV system, the payback period for an 1,000 m² system is nearly 50 years.

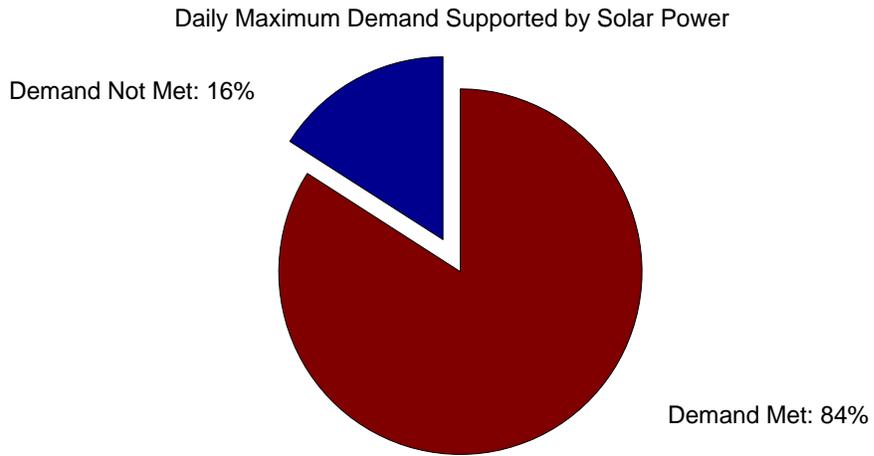


Figure 6.13: PV electrical demand percent met

Figure 6.13 shows what the expectation of solar power meeting demand on a yearly basis for every 15 minutes. Since we are only considering daylight hours to reflect the operating hours of the office, electrical demand will be met 84% of the time for a PV system spanning 1,000 m²

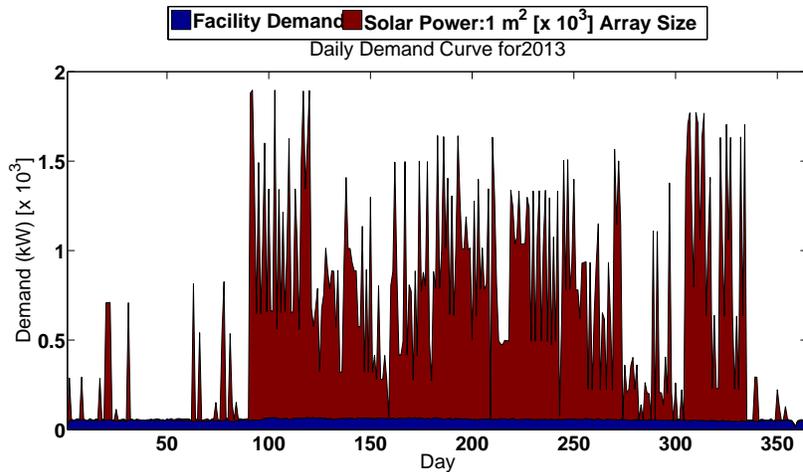


Figure 6.14: Percent demand met on a daily basis

Figure 6.14 juxtaposes the facility demand against the electrical demand supplied by the PV system. Notice that there are several days during the first and last quarter of the year in which the system will not meet the facility needs. This is due to several factors; mainly the number of sunlight hours and weather. From the figure, the system will be more than adequate to meet the demand needs of the office for a majority of the year.

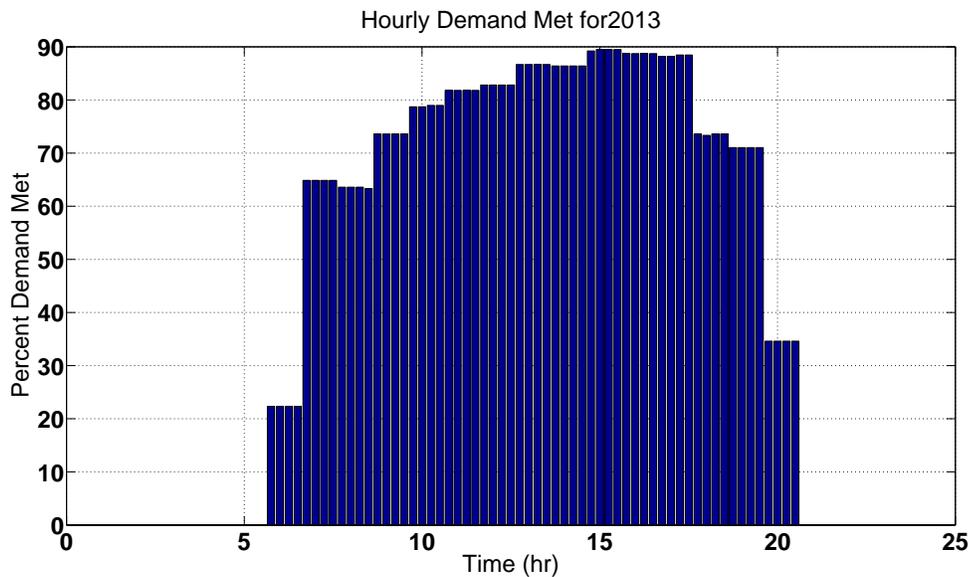


Figure 6.15: Percent demand met on an hourly basis

Figure 6.15 depicts the percent that PV meets facility demand needs on an hourly basis. A system to meet facility needs 100% of the time would be very large and very expensive to meet demand 10% of the time more than a smaller system, thus a system that can handle the facility load over 80% of the time is considered adequate for most grid tied applications. In this case, one can expect demand to be met approximately 90% of the time from 2:00 PM to 3:00 PM.

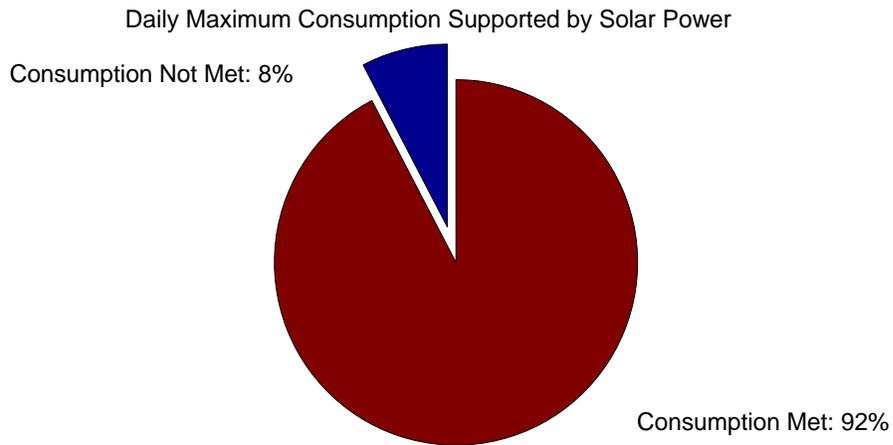


Figure 6.16: PV electrical consumption percent met

Figure 6.16 shows what can be expected from a PV system that encompasses an area of 1,000 m² on a daily basis. For a majority of the days throughout the year, the system will produce enough electrical energy to meet the needs of the office. Since the system had to be sized to meet the demand needs of the facility for a majority of the year, many days will produce more than enough electrical energy. For grid tied systems, this electricity can be put back into the grid and sold. In turn, the savings generated by putting electricity back into the grid can reduce the 50 year payback period of the system.

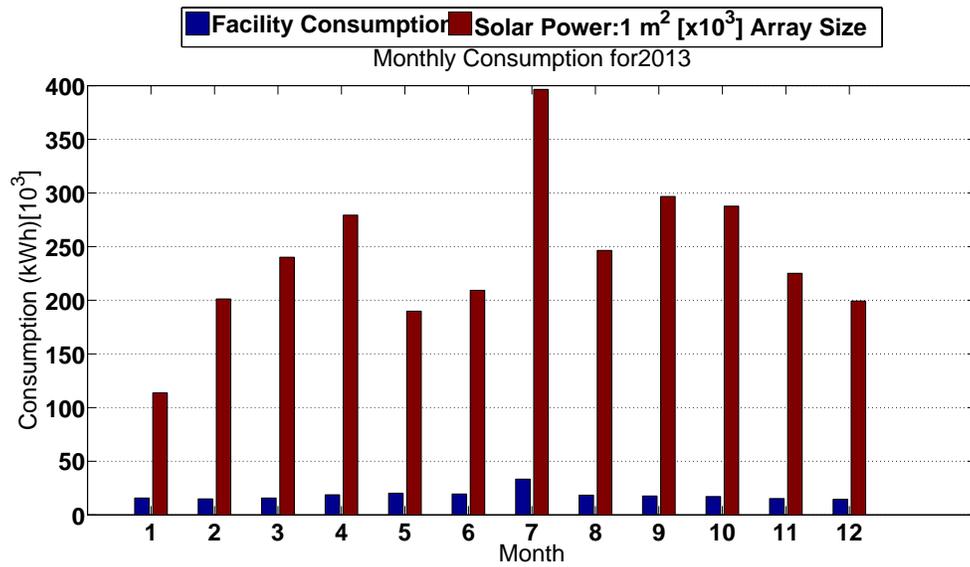


Figure 6.17: Consumption met on a monthly basis.

Figure 6.17 juxtaposes what the facility electrical consumption needs against the energy produced from a PV system with a size of 1,000 m². The difference between the facility consumption and the solar energy is considered as the energy to be sold and put back into the grid.

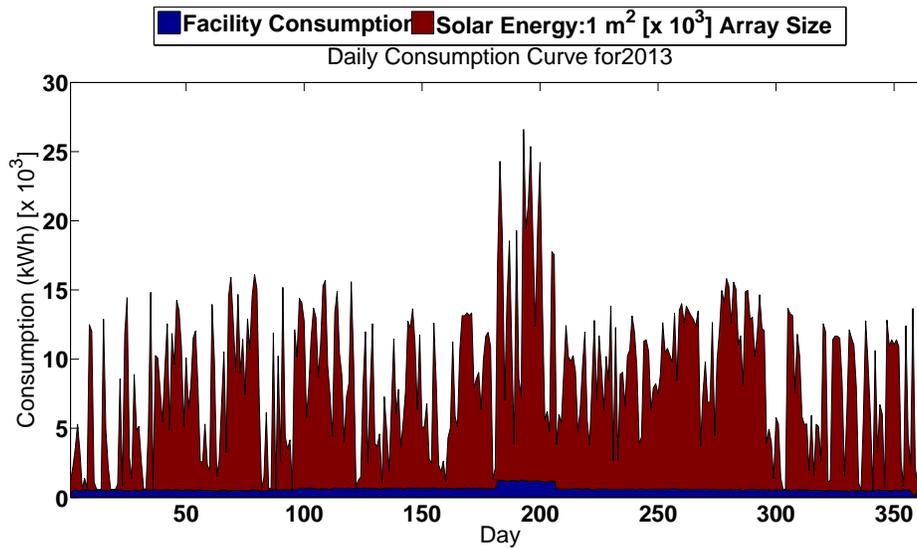


Figure 6.18: Percent consumption met on daily basis.

Figure 6.18 juxtaposes the daily electrical energy needs of the facility against the energy produced by the PV system. For most of the year, the system is more than adequate, providing the opportunity to sell electricity back to the grid and reducing the simple payback period. Selling the unused electrical energy back to the grid at a price of \$0.04 per kWh could reduce the payback period to 3 years. The unused energy was taken as the difference between the electrical energy produced by the system and the electrical consumption required by the facility on a daily basis. Table 6.1 displays the final electrical metrics to be expected by installing the sized system.

Table 6.1: Table of final PV metrics

Size of PV System (m ²)	1,000
Peak Output of Array (kW)	1,847
Annual energy Production (kWh)	2,885,554
System Cost (Installed) [\$]	805,000
Annual Consumption Savings (\$)	-175,829
Annual Demand Savings (\$)	682
Simple Payback (yr)	48 yr

Note: Negative values denote PV system is augmenting beyond facility requirement.

7. CONCLUSION

In an effort to enhance the auditing and ECM identification process, the TAMUIAC has set out to develop a set of demand analysis software tools to aid in the ECM identification process. The tools were designed with one or several of the following criteria: 1) increase analysis fidelity, 2) identify pre-visit energy conservation measures, 3) establish unknown variables helpful in diagnosing ECMs, 4) size systems design to optimize electrical usage, 5) create a simple, user friendly interface and 6) increase ECM implementation.

The demand visualization tool, realizing facility demand usage across a variety of timescales can be used to identify facility demand usage aberrations. Since these abnormalities have the potential to be costly, identifying these demand usage irregularities can aid in identifying processes that may negatively affect utility usage. With the increase in data and analysis fidelity, two separate models have been developed for identifying plant operating hours, information that might otherwise be unknown. The tool is simple to use and displays relevant usage, demand and cost data that may be of interest to decision makers.

The demand aberration uses statistical analysis to determine days in which electrical usage was abnormal. Peculiar demand usage can be costly, thus the tool helps to identify yearly processes that effect electrical usage; mitigating these process, either by negating them or shifting them can help to save electrical demand, thus helping to diagnose and remedy a possible energy conservation measure and helping TAMUIAC personnel to pre diagnose possible ECMs

The demand-scheduling tool reveals electrical savings to be had by shifting manufacturing processes to other times in the day. The practice can help to save the cost associated with electrical demand by reducing the maximum demand peak for the month. The user can interface with the tool by inputting a percent reduction in electrical demand. The tool will display the savings associated with reduction, as well as the savings associated with optimizing the demand by shifting manufacturing processes throughout a 24-hour period. If feasible, the tool can help to diagnose and remedy issues with demand usage.

The weather disaggregation tool uses an averaging model to separate the weather-based load from the aggregated demand profile. By separating what a facility might use in electricity because of the weather, quantifying savings due to replacements or retrofits becomes feasible, thus identifying possible ECMs to retrofit or replace HVAC equipment.

The photovoltaic (PV) analysis tool is an attempt to merge a facility's electrical usage with the potential power generation from photovoltaics. By understanding what a facility requires in electrical demand and consumption, a photovoltaic system can be sized to meet the partial or whole needs of a facility. The PV analysis tool can size a custom PV system, tailor made for a facility with a unique electrical requirement.

The power factor correction tool helps to quantify the savings associated with optimizing a facility's power factor by installing a capacitor bank. The tool will predict the required capacitance and the cost associated with installing a capacitor bank.

The ratcheting tool, with help the user can visualize and quantify the electrical demand and cost associated with demand ratcheting.

The automatic report generator uses Mathworks Matlab and Microsoft Word to output a report outlining the various savings associated with the implementation of various ECMs. Fifteen different reports encompassing ECMs associated with lighting, compressed air, motor, power factor correction and avoiding late fees have been developed.

Together, all of the tools are an attempt to maximize the possibility of what can be done with IDR meter data. Identifying ECMs, sizing system, identifying variables, and creating a user interface for someone with rudimentary knowledge of the dynamics associated with facility electrical usage is only half of the story. In a push to increase the implementation rate for many of the TAMUIAC audits, the tools are also an attempt to affect the psychology of ECM implementation. Being multifaceted in the type of information available to a client helps, but it is what the TAMUIAC can do with the right data, for a client that makes these tools unique. In some ways they are similar to many of the tools on the market today, but together they become integrated and aid TAMUIAC personnel to develop a characteristic map of the unique electrical usage profile inherent in any application. As of the writing of this thesis, it is unclear if the tools will live up to their expectations, but even if they do not, it is certain that they are a step in the right direction.

REFERENCES

- [1] U. E. I. Administration, "US Energy Flows," ed. Available: <http://www.eia.gov/totalenergy/data/monthly/#summary>
- [2] U. E. I. Administration, "Total Energy Consumption by Sector," ed, 2014. Available: http://www.eia.gov/totalenergy/data/monthly/pdf/sec2_3.pdf
- [3] U. S. E. I. Administration. (2013, November 18, 2013). Industrial Sector Energy Consumption. Available: <http://www.eia.gov/forecasts/ieo/industrial.cfm>
- [4] U. S. E. I. Administration. Total Energy [Online]. Available: <http://www.eia.gov/totalenergy/data/monthly/index.cfm - consumption>
- [5] W. o. S. Government. November 19, 2013). Liquefied Petroleum Gas (LPG). Available: <http://www.window.state.tx.us/specialrpt/energy/nonrenewable/lpg.php>
- [6] U. S. E. I. Administration. (2011, November 19, 2013). Residual Fuel Oil Consumption in the U.S. Continues to Decline. Available: <http://www.eia.gov/todayinenergy/detail.cfm?id=4250>
- [7] U. S. E. I. Administration. (2013, November 19, 2013). Industrial Sector Natural Gas Usage Rising. Available: <http://www.eia.gov/todayinenergy/detail.cfm?id=11771>
- [8] U. S. E. I. Administration. (2013, Monthly Energy Review. Available: <http://www.eia.gov/totalenergy/data/monthly/index.cfm - electricity>
- [9] U. S. E. I. Administration. U.S. Coal Consumption by End Use Sector [Online]. Available: <http://www.eia.gov/coal/production/quarterly/pdf/t32p01p1.pdf>
- [10] U. N. I. D. Organization, "Renewable Energy in Industrial Applications." Available: http://www.unido.org/fileadmin/user_media/Services/Energy_and_Climate_Change
- [11] U. S. E. I. Administration. November 19, 2013). International Energy Outlook 2013. Available: <http://www.eia.gov/forecasts/ieo/industrial.cfm>
- [12] U. S. E. I. Administration. Energy Prices by Sector and Source, United States, Reference Case [Online]. Available: <http://www.eia.gov/oiaf/aeo/tablebrowser/ - release=AEO2013&subject=3-AEO2013&table=3-AEO2013®ion=1-0&cases=ref2013-d102312a>
- [13] A. M. Office. (October 09). *About the Office*. Available: <https://www1.eere.energy.gov/manufacturing/about/index.html>
- [14] T. A. M. E. E. Station. (2013, October 09). *The Energy Systems Laboratory*. Available: <http://esl.tamu.edu>
- [15] NAESCO. (2011, October 09). *What is an ESCO*. Available: <http://www.naesco.org/resources/esco.htm>
- [16] B. K. Carl Blumstein, Lee Schipper, Carl York, "Overcoming Social and Institutional Barriers to energy Conservation," *Energy*, vol. 5, pp. 335-371, 1979.
- [17] L. S. Wokje Abrahamse, Charles Vlek, Talib Rothengatter, "A Review of Intervention Studies Aimed at Household Energy Conservation," *Journal of Environmental Psychology*, vol. 25, pp. 273-291, 2005.

- [18] H. Allcott, "Social Norms and Energy Conservation," *Journal of Public Economics*, pp. 1082-10925, 2011.
- [19] C. J. H. M. L.T McCalley, "Energy Conservation Through product-Integrated Feedback: The Roles of Goal Setting and Social Orientation," *Journal of Economic Psychology*, vol. 23, pp. 589-603, 2002.
- [20] (2014). *Electrical Power Factor / Calculation and Power Factor Improvement*. Available: <http://electrical4u.com/electrical-power-factor/>

APPENDIX A: ASSESSMENT RECOMMENDATION WORKSHEETS

Assessment Recommendation:

Retrofit Exit Signs

ARC 2.7142.3

IAC Assessment #: _____

Date:

IAC Technician: _____

Exit signs come in two types, 2-bulb and 1-bulb signs. It is recommended to retrofit these to LED illumination because LED bulbs consume less energy. They also require less maintenance since their life expectancy is upwards of 25 year, savings on replacing incandescent and maintenance costs. They need to be manually inspected to determine their type.

Required Measurements	
Description	Value
Number of 1 Bulb Signs	
Number of 2 Bulb Signs	

Assessment Recommendation:

Install Skylights

ARC 2.7145.3

IAC Assessment #: _____

Date:

IAC Technician: _____

Skylights are useful for operating hours between 6 AM and 3PM. Their usefulness on a yearly basis is determined by the number of sunshine days particular to a site; this information can be gathered from www.homefacts.com. The type of light is determined by sight while the lighting ballasts of the more commonly seen industrial type lights are 1.18 for Metal Halide, 1.25 for High Pressure Sodium, 1.15 for Mercury Vapor, 1.10 for T12 Fluorescent and 0.90 for T8 Fluorescent. The “per bulb” power of the more commonly seen lights are 32 W for T8 Fluorescent, 34 W for T12 Fluorescent, 400 W or 1,000 W for Metal Halide, 400 W for High Pressure Sodium and 175 W for Mercury vapor bulbs. To determine the number of photo sensors required to meet the needs of the facility, you will need the area of the space. Photo sensors are rated to service a maximum area, thus the number of sensors will be the ratio of the space area and the serviced area of the sensor. Determine the cost by gathering the price of the sensors from several retailers and taking the average.

Required Measurements	
Description	Value
Current Lighting Type	
Current Number of Fixtures	
Current Number of Bulbs per Fixture	
Current Lighting Type Ballast Factor	
Percent of Sunshine Days per Year (http://www.homefacts.com)	
Number of Photo Sensors	
Cost per Photo Sensor (\$)	
Room Length (ft)	
Room Width (ft)	
Room Height (ft)	

Assessment Recommendation:

Lighting Retrofit

ARC 2.7142.2

IAC Assessment #: _____

Date:

IAC Technician: _____

A lighting retrofit entails replacing power intensive lighting with more efficient lights. Efficient lights produce the same output of light with less power. The type of light is determined by sight while the lighting ballasts of the more commonly seen industrial type lights are 1.18 for Metal Halide, 1.25 for High Pressure Sodium, 1.15 for Mercury Vapor, 1.10 for T12 Fluorescent and 0.90 for T8 Fluorescent. The “per bulb” power of the more commonly seen lights are 32 W for T8 Fluorescent, 34 W for T12 Fluorescent, 400 W or 1,000 W for Metal Halide, 400 W for High Pressure Sodium and 175 W for Mercury vapor bulbs. To determine the proposed lighting fixture and bulb costs, use several sources (usually internet) and take the average cost. Be aware that the proposed bulb lighting output should be the current lighting output (usually characterized as Lumen).

Required Measurements	
Description	Value
Current Lighting Type	
Proposed Lighting Type	
Current Lighting Power (per bulb) [Watt]	
Current Lighting Ballast Factor	
Current Lighting Bulbs per Fixture	
Proposed Lighting Power (per bulb) [Watt]	
Proposed Lighting Ballast Factor	
Proposed Lighting Bulbs per Fixture	
Proposed Lighting Fixture Cost	
Proposed Lighting Bulb Cost	

Assessment Recommendation:

Turn Off Lights

ARC 2.7124.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Lighting can add up to a significant portion of a facilities electrical usage. Many times, a facility leaves lights on in areas that are unoccupied during most of the operating hours. Under these circumstances, the recommendation is to turn off lights by using either an occupancy sensor or a photo sensor. The type of light is determined by sight while the lighting ballasts of the more commonly seen industrial type lights are 1.18 for Metal Halide, 1.25 for High Pressure Sodium, 1.15 for Mercury Vapor, 1.10 for T12 Fluorescent and 0.90 for T8 Fluorescent. The “per bulb” power of the more commonly seen lights are 32 W for T8 Fluorescent, 34 W for T12 Fluorescent, 400 W or 1,000 W for Metal Halide, 400 W for High Pressure Sodium and 175 W for Mercury vapor bulbs. Photo sensors and occupancy are rated to service a maximum area, thus the number of sensors will be the ratio of the space area and the serviced area of the sensor. Determine the cost by gathering the price of the sensors from several retailers and taking the average. The proposed time that the lights should be turned on can be gathered by using either a light and occupancy sensor, or by inquiring with plant personnel.

Required Measurements	
Description	Value
Current Lighting Type	
Current Lighting Power (per bulb) [Watt]	
Current Lighting Ballast Factor	
Current Lighting Bulbs per Fixture	
Current Percent Time On	
Proposed Percent Time On	
Number of Sensors	
Cost per Sensor	

Assessment Recommendation:

Repair Compressed Air Leaks

ARC 2.4236.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Repairs to the compressed air system are the most common (as of Sept 2014) recommendation of the TAMU IAC. Since energy goes into producing compressed air, wasting it unnecessarily is a waste of energy. Tank measurements are typically taken with a tape measure. Compressed air pipe length is taken by stepping off the distance (it can also be done using a laser range finder). IAC personnel usually estimate load factor and duty factors after a discussion with plant personnel. The pressure drop is usually either 5 or 10 psi and the time is determined using a stopwatch (or a phone will do). Compressor specific efficiency is determined from supplemental IAC material. Motor efficiencies are gathered using the DOE program, Motor Master International +.

Required Measurements	
Description	Value
Tank Radius (ft)	
Tank Length (ft)	
Pipe Radius (ft)	
Pipe Length (ft)	
Total Compressed Air Motor Power (hp)	
Drop Test Pressure Drop (psi)	
Drop Test Time (min)	
Compressor Specific Efficiency (see supplemental material)	
Motor Efficiency (Motor Master)	
Motor Load Factor	
Motor Duty Factor	

Assessment Recommendation:

Reduce Compressed Air Pressure

ARC 2.4231.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Under many circumstances, facilities set their compressed air pressure to a higher than needed value to fight system losses (namely compressed air leaks). Since it takes energy to create compressed air at a set pressure, repairing compressed air leaks and decreasing the compressed air pressure can realize energy savings. The compressor power can be read off of the compressor motor nameplate. Since a facilities compressed air system is usually fundamental to manufacturing operations, plant management will more than likely know the compressor power. The current pressure can be read off of the compressor controls (if digital) or by the pressure gauge fixed to the compressed air tank, compressor or compressed air line. Since operations vary throughout manufacturing facilities, the required pressure may vary. A discussion with plant management is needed to deduce how much pressure is required by equipment; the pressure setting of the system should not be larger than the amount required by the equipment. Under many circumstances, this should be no more than 100 psi. The compressor duty factor can be determined from either using a data logger or with a discussion of how the equipment is used during normal operations with plant management. The load factor is also determined by inquiring with plant management.

Required Measurements	
Description	Value
Number of Compressors	
Total Compressor Power (hp)	
Current Pressure (psig)	
Proposed Pressure (psig)	
Compressor Load Factor	
Compressor Duty Factor	
Compressor Motor Efficiency	

Assessment Recommendation:

Engineered Nozzles

ARC 2.4322.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Engineered Nozzles are recommended because they reduce the amount of compressed air used by entraining outside air into the compressed air stream, thus reducing the amount of compressed air needed for a process while stabilizing the volumetric air flow rate. The nozzle diameter is gathered either by using a tape measure or, usually when small enough, by sight. The compressor specific efficiency and the nozzle airflow are gathered from IAC supplemental literature. IAC personnel determine load and duty factors after a discussion with plant personnel.

Required Measurements	
Description	Value
Number of Nozzles	
Nozzle Diameter (in)	
Nozzle Airflow (CFM)	
Nozzle Operating Time (hrs / yr)	
Compressor Specific Efficiency	
Compressor Load Factor	

Assessment Recommendation:

High Efficiency Motors

ARC 2.4322.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Retrofitting current motors with premium efficiency ones decreases the power consumption of the motor while maintaining the same power output. The lower power input to the motor, over the course of a year can save a considerable amount of electrical consumption. To determine the current efficiency of the motor, the proposed efficiency of the premium efficiency motor and the price difference between the two, use the DOE program, Motor Master International +. Load and duty factors are determined after a discussion of how they are used with plant personnel. Under many circumstances, this recommendation will yield a larger than 2 year payback, thus the recommendation is to retrofit current motors with premium efficiency ones as the current plethora of motors become inoperable.

Required Measurements	
Description	Value
Motor Power (hp)	
Number of Motors	
Current Motor Efficiency	
Premium Motor Efficiency	
Motor Cost Difference (\$)	
Load Factor	
Duty Factor	
Room Width (ft)	
Room Height (ft)	

Assessment Recommendation:

Energy Efficient Belts

ARC 2.4133.1

IAC Assessment #: _____

Date:

IAC Technician: _____

Energy efficient belts are cogged or notched to reduce the bending friction of the belt against the motor. Motor Power is gathered from the motor nameplate, although under some circumstances the facility may have a motor list. Motor Efficiency is determined from the DOE program, Motor Master International +. IAC personnel estimate load factors and duty factors after a discussion with plant personnel. Belt cost may vary depending on the type of motor they are applied to. Look at various references for cost (usually internet references) and take an average.

Required Measurements	
Description	Value
Motor Power (hp)	
Number of Motors	
Motor Efficiency (Motor Master)	
Load Factor	
Duty Factor	
Belt Cost (\$ / Belt)	

Assessment Recommendation:

Synthetic Lubricants

ARC 2.4314.1

IAC Assessment #: _____

Date:

IAC Technician: _____

Adding synthetic lubricants to motor bearings decreases the fictional inefficiency by 10%, resulting in significant energy savings as the motor is operated throughout the year. Only motors outfitted with Zerk fittings are candidates for synthetic lubricants. Rated motor horsepower can be determined from the motor nameplate affixed to the motor. Under some circumstances, a facility may catalog the facility motors in a “motor list”. The motor efficiency is determined by using the DOE program, Motor Master International +. The duty factor can be determined by either using a data logger to determine its operation, but under most circumstances is determined by speaking with plant personnel on the motors operation. The load factor is also determined by an interview with plant personnel.

Required Measurements	
Description	Value
Motor Power (hp)	
Number of Motors	
Current Motor Efficiency	
Load Factor	
Duty Factor	

Assessment Recommendation:

Power Factor Correction

ARC 2.3212.3

IAC Assessment #: _____

Date:

IAC Technician: _____

When an electrical provider supplies power to a facility, termed “apparent power” (measured in kVA [kilo Volt-Ampere]), the facility consumes the power in two ways. The first is to produce work (illuminating lights, turning motors, compressing air, etc.) and is termed “real power” (measured in kW [kilo Watts]). The second way a facility consumes power is in non-work operation (energizing the magnetic field in motors and lighting ballasts, etc.) and is termed “reactive power” (measured in kVAr [kilo Volt-Ampere Reactive]). The power factor is the ratio of the real power to the supplied power, thus the more power that goes to produce work, the larger the power factor. The electrical provider measures the quality of the power a facility consumes by measuring the power factor and imposes a charge on the facility when the power factor drops below a set threshold (usually 0.95 or 0.90 [meaning that 95% or 90% of the supplied power should be used to produce work]). A low power factor is indicative of a plant that has a significant portion of its electrical usage in non-work operations. To increase the power factor, a capacitor bank is used to mitigate the reactive power, thus increasing the power factor to an acceptable value. The billed demand and current power factor can be gathered from the electrical utility bills. Under some circumstances, the power factor has to be calculated from other bill information (search “Power Triangle” on the internet). The fixed and variable capacitance costs are determined by contacting capacitor bank installers and vendors. As of Sept. 2014, the costs are \$50 per kVAr for fixed capacitance and \$70 per kVAr for variable capacitance.

Required Measurements	
Description	Value
Billed Demand (kW) [monthly]	
Current Power Factor (monthly)	
Desired Power Factor	
Fixed Capacitance Cost	
Variable Capacitance Cost	

Assessment Recommendation:

Tune Up Boiler (If Boiler NG Usage Is Known)

ARC 2.1233.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Boilers need to be serviced to maintain their combustion efficiency at least once a year. Maintaining the efficiency means producing the same amount of heat with less energy, thus translating into fuel savings. This recommendation is specific to those facilities that know exactly how natural gas their boilers are using. The oxygen percentage and the combustion efficiency are gathered using a combustion analyzer.

Required Measurements	
Description	Value
Number of Boilers	
Boiler NG Usage	
Oxygen Percentage	
Current Boiler Efficiency	
Proposed Boiler Efficiency (Usually 83%)	

Assessment Recommendation:

Tune Up Boiler (If Boiler NG Usage Is Unknown)

ARC 2.1233.2

IAC Assessment #: _____

Date:

IAC Technician: _____

Boilers need to be serviced to maintain their combustion efficiency at least once a year. Maintaining the efficiency means producing the same amount of heat with less energy, thus translating into fuel savings. This recommendation is specific to those facilities that do not know exactly how much natural gas their boilers are using. The boiler load factor and duty factor are determined after a discussion with plant personnel. The current boiler efficiency is determined using a combustion analyzer. The boiler operating time is usually the plant hours of operation, although this may be different for various facilities.

Required Measurements	
Description	Value
Number of Boilers	
Boiler Name	
Boiler fuel consumption (Unit / hr)	
Boiler Load Factor	
Boiler Duty Factor	
Current Boiler Efficiency	
Proposed Boiler Efficiency (Usually 83%)	
Boiler Operating Time	

Assessment Recommendation:

Late Fees

ARC 2.7124.4

IAC Assessment #: _____

Date:

IAC Technician: _____

A late fee is incurred by a facility when they do not make an on time payment. Since the owed amount is, essentially loaned by the utility provider, there is an annual percentage rate associated with the loan. The month, charge and utility type are gathered from the utility bills. The grace period and percent penalty are determined from the utility rate schedule, available by the utility.

Required Measurements	
Description	Value
Utility Type	
Month	
Charge	
Grace Period	
Percent Penalty	

APPENDIX B: LIGHTING ASSESSMENT RECOMMENDATIONS

Assessment Recommendation No. 1

Replace Incandescent Lights in Exit Signs with L.E.D. Retrofit (1 Bulb Type)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value21>	\$<value25>			
<value20> kWh	<value24> kW	\$<value28>	CO2: <value31> tons NOX: <value32> lbs	<value29> > yr (<value30>mo)
<value33>%	<value34>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace incandescent lamps found in exit signs with energy efficient L.E.D.'s.

Current Operations and Observations

During the plant walkthrough, I.A.C. personnel viewed <value6> exit signs illuminated using incandescent bulbs. These exit signs are currently using <value4> Watt incandescent bulbs, which can be replaced with <value8> Watt L.E.D. lamps that are more energy efficient. Use of L.E.D. lamps will also result in lower maintenance costs, as L.E.D. lamps have a rated life of 20+ years, whereas the incandescent lamps currently in use are only rated for a life of 2,000 hours. Implementation of this recommendation will reduce both energy and maintenance costs for the plant.

Calculations

The following example calculations are done for the <value5> bulb exit signs. Subsequent calculations for all sign types can be seen in Table 1.1.

Table 1.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P _{Inc}	Incandescent Bulb Power	<value4> W
X _{Bulb}	No. of Bulbs per Sign	Table 1.1
X _{Sign}	No. of Signs	Table 1.1
D _{Inc}	Incandescent Bulb Cost	$\frac{\$ <value37>}{\text{Bulb}}$
T _{Inc}	Time to Install Incandescent Bulb	<value39> $\frac{\text{hrs}}{\text{Bulb}}$
K _{W2KW}	Watt to Kilowatt Conversion Factor	<value7> $\frac{\text{kW}}{\text{W}}$
K _{M2YR}	Month to Year Conversion Factor	<value10> $\frac{\text{mo}}{\text{yr}}$
K _{hrs2yr}	Hours to Year Conversion	<value36> $\frac{\text{hrs}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P _{L.E.D.}	L.E.D. Light Power	<value8> W
X _{Kit}	Bulbs per Kit	<value12> $\frac{\text{Bulbs}}{\text{Kit}}$
D _{Kit}	Cost per Kit ¹	$\frac{\$ <value13 >}{\text{Kit}}$
T _{Kit}	Kit Installation Time ²	<value14> $\frac{\text{hrs}}{\text{Kit}}$

¹ Cost has been averaged from various sources. Contact IAC for further details.

² Estimated by IAC Personnel.

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ³	$\frac{\$<value9>}{kWh}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$<value11>}{kW\cdot mo}$
D_L	Maintenance Labor Cost ⁴	$\frac{\$<value15>}{hr}$
T_O	Operating Hours ⁴	$<value8> \frac{hrs}{yr}$
E_{CO2}	CO ₂ Emission Factor ⁵	$<value16> \frac{tons}{kWh}$
E_{NOX}	NO _x Emission Factor ⁵	$<value17> \frac{lbs}{kWh}$

³ See energy cost analysis section.

⁴ See summary of plant statistics section.

⁵ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	Table 1.1
E _P	Proposed Energy Usage	Table 1.1
E _S	Energy Savings	Table 1.1
E _{CS}	Total Energy Cost Savings	Table 1.1
D _C	Current Demand Usage	Table 1.1
D _P	Proposed Demand Usage	Table 1.1
D _S	Demand Savings	Table 1.1
D _{CS}	Demand Cost Savings	Table 1.1
D _{MCS}	Current Maintenance Cost Savings	Table 1.1
C _T	Total Cost Savings	Table 1.1
C _C	Capital Cost	Table 1.1
C _I	Implementation Cost	Table 1.1
T _P	Simple Payback	<value29> yr (<value30> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value31> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value32> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting exit signs is the product of the lights power consumption and the total number of hours per year, multiplied by the number of bulbs found in each sign.

$$E_C = (P_{Inc})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{hrs2yr})$$

$$\left(\langle \text{value4} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{Signs} \right) \left(\langle \text{value7} \rangle \right)$$

$$> \frac{kW}{W} \left(\langle \text{value36} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value18} \rangle \frac{kWh}{yr}$$

$$\text{Total Current Energy Usage} = \langle \text{value40} \rangle \frac{kWh}{yr}$$

Proposed Energy Usage

Retrofitting the exit signs with L.E.D. illumination can save on electrical consumption while providing the same amount of light. The following calculation depicts L.E.D. lighting electrical consumption.

$$E_P = (P_{L.E.D.})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{hrs2yr})$$

$$\left(\langle \text{value8} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{Signs} \right) \left(\langle \text{value7} \rangle \right)$$

$$> \frac{kW}{W} \left(\langle \text{value36} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value19} \rangle \frac{kWh}{yr}$$

$$\text{Total Proposed Energy Usage} = \langle \text{value41} \rangle \frac{kWh}{yr}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$

$$\langle \text{value18} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value19} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Energy Savings} = \langle \text{value42} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$

$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value9} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value21} \rangle}{\text{yr}}$$

$$\text{Total Energy Cost Savings} = \frac{\$ \langle \text{value43} \rangle}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, multiplied by the number of bulbs found in each sign, operating on a monthly basis over the course of a year.

$$D_C = (P_{\text{Inc.}})(X_{\text{Bulb}})(X_{\text{Sign}})(K_{W2KW})(K_{M2YR})$$

$$\left(\langle \text{value4} \rangle \frac{\text{W}}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{Signs} \right) \left(\langle \text{value7} \right.$$

$$\left. \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value10} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Current Demand Usage} = \langle \text{value44} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Retrofitting the signs with efficient L.E.D. illumination can save on electrical demand. The following calculation depicts L.E.D. lighting demand.

$$D_P = (P_{L.E.D.})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{M2YR})$$

$$\left(< \text{value8} > \frac{W}{\text{Bulb}} \right) \left(< \text{value5} > \frac{\text{Bulbs}}{\text{Sign}} \right) \left(< \text{value6} > \text{Signs} \right) \left(< \text{value7} > \frac{K_{M2YR}}{K_{W2KW}} \right) > \frac{kW}{W} \left(< \text{value10} > \frac{\text{mo}}{\text{yr}} \right) = < \text{value23} > \frac{kW \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = < \text{value45} > \frac{kW \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$

$$< \text{value22} > \frac{kW \cdot \text{mo}}{\text{yr}} - < \text{value23} > \frac{kW \cdot \text{mo}}{\text{yr}} = < \text{value24} > \frac{kW \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Demand Savings} = < \text{value46} > \frac{kW \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$

$$\left(< \text{value24} > \frac{kW \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ < \text{value11} >}{kW \cdot \text{mo}} \right) = \frac{\$ < \text{value25} >}{\text{yr}}$$

$$\text{Total Demand Cost Savings} = \frac{\$ < \text{value47} >}{\text{yr}}$$

Current Maintenance Cost

Since incandescent bulbs have an operational time of 2,000 hours, there are cost savings to be had from replacing the incandescent bulbs as well as maintenance.

$$D_{MCS} = [(X_{Sign})(X_{Bulb})] \left[\left(\frac{1}{T_{Bulb}} \right) (K_{hrs2yr})(D_{Inc}) + (T_{Inc})(D_L) \right]$$

$$\left[(< value6 > \text{ Signs}) \left(< value5 > \frac{\text{Bulbs}}{\text{Sign}} \right) \right] \left[\left(\frac{1 \text{ Bulb}}{< value39 > \text{ hrs}} \right) \left(< value36 > \frac{\text{hrs}}{\text{yr}} \right) \left(\frac{\$ < value37 >}{\text{Bulb}} \right) + \left(< value39 > \frac{\text{hrs}}{\text{Bulb}} \right) \left(\frac{\$ < value15 >}{\text{hr}} \right) \right]$$

$$= \frac{\$ < value38 >}{\text{yr}}$$

$$\text{Total Current Maintenance Cost} = \frac{\$ < value48 >}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS} + D_{MCS}$$

$$\frac{\$ < value20 >}{\text{yr}} + \frac{\$ < value25 >}{\text{yr}} + \frac{\$ < value38 >}{\text{yr}} = \frac{\$ < value26 >}{\text{yr}}$$

$$\text{Total Current Maintenance Cost} = \frac{\$ < value49 >}{\text{yr}}$$

Capital Cost

The capital cost is the amount of money needed to buy exit sign lighting retrofit kits.

$$C_c = (X_{Sign})(X_{Bulb}) \left(\frac{1}{X_{Kit}} \right) (D_{Kit})$$

$$(< value6 > \text{ Signs}) \left(< value5 > \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\frac{1 \text{ Kit}}{< value12 > \text{ Bulbs}} \right) \left(\frac{\$ < value13 >}{\text{Kit}} \right) = \$ < value27 >$$

Implementation Cost

The implementation cost is the sum of the capital cost and the manpower cost associated with installation.

$$C_I = C_C + (X_{\text{Sign}})(X_{\text{Bulb}})(X_{\text{Kit}})\left(\frac{1}{T_{\text{Kit}}}\right)(D_L)$$

$$\begin{aligned} & \$ \langle \text{value27} \rangle + (\langle \text{value6} \rangle \text{ Signs}) \left(\langle \text{value5} \right. \\ & \left. > \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\frac{1}{\langle \text{value12} \rangle \text{ Bulbs}} \right) \left(\langle \text{value14} \rangle \frac{\text{hr}}{\text{Kit}} \right) \left(\frac{\$ \langle \text{value15} \rangle}{\text{hr}} \right) = \$ \\ & \langle \text{value28} \rangle \end{aligned}$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ \langle \text{value28} \rangle}{\frac{\$ \langle \text{value26} \rangle}{\text{yr}}} = \langle \text{value29} \rangle \text{ yr } (\langle \text{value30} \rangle \text{ mo})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$

$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value16} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value31} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_S)(E_{\text{NOX}})$$

$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value17} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value32} \rangle \frac{\text{lbs}}{\text{kWh}}$$

Assessment Recommendation No. 2

Replace Incandescent Lights in Exit Signs with L.E.D. Retrofit (Multiple Bulb Type)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value21>	\$<value25>			
<value20> kWh	<value24> kVA*mo	\$<value28>	CO2: <value31> tons NOX: <value32> lbs	<value29> > yr (<value30>mo)
<value33>%	<value34>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace incandescent lamps found in exit signs with energy efficient L.E.D.'s.

Current Operations and Observations

During the plant walkthrough, I.A.C. personnel recorded <value6> exit signs illuminated using incandescent bulbs. These exit signs are currently using <value4> Watt incandescent bulbs, which can be replaced with <value8> Watt L.E.D. lamps that are more energy efficient. Use of L.E.D. lamps will also result in lower maintenance costs, as L.E.D. lamps have a rated life of 20+ years, whereas the incandescent lamps currently in use are only rated for a life of 2,000 hours. Implementation of this recommendation will reduce both energy and maintenance costs for the plant.

Calculations

The following example calculations are done for the <value5> bulb exit signs. Subsequent calculations for all sign types can be seen in Table 2.1.

Table 2.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P_{Inc}	Incandescent Bulb Power	<value4> W
X_{Bulb}	No. of Bulbs per Sign	Table 2.1
X_{Sign}	No. of Signs	Table 2.1
D_{Inc}	Incandescent Bulb Cost	$\frac{\$ <value37>}{\text{Bulb}}$
T_{Inc}	Time to Install Incandescent Bulb	<value39> $\frac{\text{hrs}}{\text{Bulb}}$
K_{W2KW}	Watt to Kilowatt Conversion Factor	<value7> $\frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	<value10> $\frac{\text{mo}}{\text{yr}}$
K_{hrs2yr}	Hours to Year Conversion	<value36> $\frac{\text{hrs}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$P_{L.E.D.}$	L.E.D. Light Power	<value8> W
X_{Kit}	Bulbs per Kit	<value12> $\frac{\text{Bulbs}}{\text{Kit}}$
D_{Kit}	Cost per Kit ⁶	$\frac{\$ <value13 >}{\text{Kit}}$
T_{Kit}	Kit Installation Time ⁷	<value14> $\frac{\text{hrs}}{\text{Kit}}$

⁶ Cost has been averaged from various sources. Contact IAC for further details.

⁷ Estimated by IAC Personnel.

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ⁸	$\frac{\$<value9>}{\text{kWh}}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$<value11>}{\text{kVA}\cdot\text{mo}}$
D_L	Maintenance Labor Cost ⁹	$\frac{\$<value15>}{\text{hr}}$
T_O	Operating Hours ⁴	$<value8> \frac{\text{hrs}}{\text{yr}}$
E_{CO2}	CO ₂ Emission Factor ¹⁰	$<value16> \frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ⁵	$<value17> \frac{\text{lbs}}{\text{kWh}}$

⁸ See energy cost analysis section.

⁹ See summary of plant statistics section.

¹⁰ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	Table 2.1
E _P	Proposed Energy Usage	Table 2.1
E _S	Energy Savings	Table 2.1
E _{CS}	Total Energy Cost Savings	Table 2.1
D _C	Current Demand Usage	Table 2.1
D _P	Proposed Demand Usage	Table 2.1
D _S	Demand Savings	Table 2.1
D _{CS}	Demand Cost Savings	Table 2.1
D _{MCS}	Current Maintenance Cost Savings	Table 2.1
C _T	Total Cost Savings	Table 2.1
C _C	Capital Cost	Table 2.1
C _I	Implementation Cost	Table 2.1
T _P	Simple Payback	<value29> yr (<value30> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value31> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value32> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting exit signs is the product of the lights power consumption and the total number of hours per year, multiplied by the number of bulbs found in each sign.

$$E_C = (P_{Inc})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{hrs2yr})$$

$$\left(\langle \text{value4} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{Signs} \right) \left(\langle \text{value7} \rangle \frac{K_{hrs2yr}}{W} \right) = \langle \text{value18} \rangle \frac{kWh}{yr}$$

$$\text{Total Current Energy Usage} = \langle \text{value40} \rangle \frac{kWh}{yr}$$

Proposed Energy Usage

Retrofitting the exit signs with L.E.D. illumination can save on electrical consumption while providing the same amount of light. The following calculation depicts L.E.D. lighting electrical consumption.

$$E_P = (P_{L.E.D.})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{hrs2yr})$$

$$\left(\langle \text{value8} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{Signs} \right) \left(\langle \text{value7} \rangle \frac{K_{hrs2yr}}{W} \right) = \langle \text{value19} \rangle \frac{kWh}{yr}$$

$$\text{Total Proposed Energy Usage} = \langle \text{value41} \rangle \frac{kWh}{yr}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$
$$\langle \text{value18} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value19} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}}$$
$$\text{Total Energy Savings} = \langle \text{value42} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$
$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value9} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value21} \rangle}{\text{yr}}$$
$$\text{Total Energy Cost Savings} = \frac{\$ \langle \text{value43} \rangle}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, multiplied by the number of bulbs found in each sign, operating on a monthly basis over the course of a year.

$$D_C = (P_{Inc.})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{M2YR})$$

$$\left(\langle \text{value4} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{ Signs} \right) \left(\langle \text{value7} \right.$$

$$\left. > \frac{kW}{W} \right) \left(\langle \text{value10} \rangle \frac{\text{mo}}{\text{yr}} \right) \div \left(\langle \text{value52} \rangle \frac{kW}{kVA} \right) = \langle \text{value22} \rangle$$

$$> \frac{kVA \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Current Demand Usage} = \langle \text{value44} \rangle \frac{kVA \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Retrofitting the signs with efficient L.E.D. illumination can save on electrical demand. The following calculation depicts L.E.D. lighting demand.

$$D_P = (P_{L.E.D.})(X_{Bulb})(X_{Sign})(K_{W2KW})(K_{M2YR})$$

$$\left(\langle \text{value8} \rangle \frac{W}{\text{Bulb}} \right) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Sign}} \right) \left(\langle \text{value6} \rangle \text{ Signs} \right) \left(\langle \text{value7} \right.$$

$$\left. > \frac{kW}{W} \right) \left(\langle \text{value10} \rangle \frac{\text{mo}}{\text{yr}} \right) \div \left(\langle \text{value52} \rangle \frac{kW}{kVA} \right) = \langle \text{value23} \rangle$$

$$> \frac{kVA \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = \langle \text{value45} \rangle \frac{kVA \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$
$$\langle \text{value22} \rangle \frac{\text{kVA} \cdot \text{mo}}{\text{yr}} - \langle \text{value23} \rangle \frac{\text{kVA} \cdot \text{mo}}{\text{yr}} = \langle \text{value24} \rangle \frac{\text{kVA} \cdot \text{mo}}{\text{yr}}$$
$$\text{Total Demand Savings} = \langle \text{value46} \rangle \frac{\text{kVA} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value24} \rangle \frac{\text{kVA} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value11} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value25} \rangle}{\text{yr}}$$
$$\text{Total Demand Cost Savings} = \frac{\$ \langle \text{value47} \rangle}{\text{yr}}$$

Current Maintenance Cost

Since incandescent bulbs have an operational time of 2,000 hours, there are cost savings to be had from replacing the incandescent bulbs as well as maintenance.

$$D_{MCS} = [(X_{Sign})(X_{Bulb})] \left[\left(\frac{1}{T_{Bulb}} \right) (K_{hrs2yr})(D_{Inc}) + (T_{Inc})(D_L) \right]$$

$$\left[(<value6> \text{ Signs}) \left(<value5> \frac{\text{Bulbs}}{\text{Sign}} \right) \right] \left[\left(\frac{1 \text{ Bulb}}{<value39> \text{ hrs}} \right) \left(<value36> \right. \right.$$

$$\left. \left. > \frac{\text{hrs}}{\text{yr}} \right) \left(\frac{\$ <value37>}{\text{Bulb}} \right) + \left(<value39> \frac{\text{hrs}}{\text{Bulb}} \right) \left(\frac{\$ <value15>}{\text{hr}} \right) \right]$$

$$= \frac{\$ <value38>}{\text{yr}}$$

$$\text{Total Current Maintenance Cost} = \frac{\$ <value48>}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS} + D_{MCS}$$

$$\frac{\$ <value20>}{\text{yr}} + \frac{\$ <value25>}{\text{yr}} + \frac{\$ <value38>}{\text{yr}} = \frac{\$ <value26>}{\text{yr}}$$

$$\text{Total Cost Savings} = \frac{\$ <value49>}{\text{yr}}$$

Capital Cost

The capital cost is the amount of money needed to buy exit sign lighting retrofit kits.

$$C_c = (X_{\text{Sign}})(X_{\text{Bulb}})\left(\frac{1}{X_{\text{Kit}}}\right)(D_{\text{Kit}})$$

$$(< \text{value6} > \text{ Signs})\left(< \text{value5} > \frac{\text{Bulbs}}{\text{Sign}}\right)\left(\frac{1}{< \text{value12} > \frac{\text{Kit}}{\text{Bulbs}}}\right)\left(\frac{\$ < \text{value13} >}{\text{Kit}}\right) = \$$$

$$< \text{value27} >$$

$$\text{Total Capital Cost} = \frac{\$ < \text{value50} >}{\text{yr}}$$

Implementation Cost

The implementation cost is the sum of the capital cost and the manpower cost associated with installation.

$$C_I = C_c + (X_{\text{Sign}})(X_{\text{Bulb}})(X_{\text{Kit}})\left(\frac{1}{T_{\text{Kit}}}\right)(D_L)$$

$$\$ < \text{value27} > + (< \text{value6} > \text{ Signs})\left(< \text{value5} >$$

$$> \frac{\text{Bulbs}}{\text{Sign}}\right)\left(\frac{1}{< \text{value12} > \frac{\text{Kit}}{\text{Bulbs}}}\right)\left(< \text{value14} > \frac{\text{hr}}{\text{Kit}}\right)\left(\frac{\$ < \text{value15} >}{\text{hr}}\right) = \$$$

$$< \text{value28} >$$

$$\text{Total Implementation Cost} = \frac{\$ < \text{value51} >}{\text{yr}}$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ \langle \text{value28} \rangle}{\frac{\$ \langle \text{value26} \rangle}{\text{yr}}} = \langle \text{value29} \rangle \text{ yr } (\langle \text{value30} \rangle \text{ mo})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$

$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value16} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value31} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{NO_x} = (E_S)(E_{NO_x})$$

$$\left(\langle \text{value20} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value17} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value32} \rangle \frac{\text{lbs}}{\text{kWh}}$$

Assessment Recommendation No. 3
Install Skylights and Control Lighting with Photosensors
(Single Zone, Single Lighting Type)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value35>	\$<value41>			<value45>
<value32> kWh	<value39> kW	\$<value44>	CO ₂ : <value49> tons NO _x : <value50> lbs	> yr (<value46> mo)
<value51>%	<value52>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Install Skylights through out the various production areas of the facility.

Current Operations and Observations

On the day of the visit, IAC personnel observed various areas of the plant that could benefit from augmenting the existing roofing structure with skylights. Adding skylights will supplement artificial lighting while cutting illumination costs.

Calculations

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\langle\text{value18}\rangle,\text{Fixture}}$	No. of $\langle\text{value18}\rangle$ Fixtures	$\langle\text{value19}\rangle$ Fixtures
$X_{\langle\text{value18}\rangle,\text{Bulb}}$	No of $\langle\text{value18}\rangle$ Bulbs per Fixture	$\langle\text{value20}\rangle \frac{\text{Bulbs}}{\text{Fixture}}$
$P_{\langle\text{value18}\rangle}$	$\langle\text{value18}\rangle$ Power	$\langle\text{value21}\rangle$ W
$B_{\langle\text{value18}\rangle}$	$\langle\text{value18}\rangle$ Ballast Factor	$\langle\text{value22}\rangle$
K_{W2KW}	Watt to Kilowatt Conversion Factor	$\langle\text{value23}\rangle \frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	$\langle\text{value36}\rangle \frac{\text{mo}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
F_{Sky}	Sky Lighting Factor	<value26 >
F_{Sun}	Sunshine Factor	<value30>
F_{RSDD}	Room Surface Depreciation Coefficient	<value4>
F_{RSD}	Skylight Dirt Depreciation Coefficient	<value5>
L	Room Length	<value8>
W	Room Width	<value9>
H	Room Height	<value7>
τ	Skylight Transmittance	<value13>
E_{T}	Proposed Illuminance Horizontal	<value14> ft•cd
E_{H}	Illuminance Produced by Sky	<value15> ft•cd
A_{S}	Gross Skylight Area	<value16> $\frac{\text{ft}^2}{\text{Skylight}}$
D_{S}	Installation Cost	$\$ < \text{value43} >$ $\frac{\text{ft}^2}{\text{ft}^2}$
X_{Photo}	No. of Photosensors	<value55> Photosensors
D_{Photo}	Photosensor Cost	$\$ < \text{value56} >$ $\frac{\text{Photosensor}}{\text{Photosensor}}$
T_{Photo}	Photosensor Installation Time	<value58> $\frac{\text{hr}}{\text{Photosensor}}$

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ¹¹	$\frac{\$ < \text{value34} >}{\text{kWh}}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$ < \text{value40} >}{\text{kW} \cdot \text{mo}}$
D_L	Maintenance Labor Cost ¹²	$\frac{\$ < \text{value59} >}{\text{hr}}$
T_O	Operating Hours ⁴	$< \text{value24} > \frac{\text{hrs}}{\text{yr}}$
E_{CO2}	CO ₂ Emission Factor ¹³	$< \text{value47} > \frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ⁶	$< \text{value48} > \frac{\text{lbs}}{\text{kWh}}$

¹¹ See energy cost analysis section.

¹² See summary of plant statistics section.

¹³ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	<value25> $\frac{\text{kWh}}{\text{yr}}$
E _P	Proposed Energy Usage	<value31> $\frac{\text{kWh}}{\text{yr}}$
E _S	Energy Savings	<value32> $\frac{\text{kWh}}{\text{yr}}$
E _{CS}	Energy Cost Savings	$\frac{\$ < \text{value35} >}{\text{yr}}$
D _C	Current Demand Usage	<value37> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D _P	Proposed Demand Usage	<value38> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D _S	Demand Savings	<value39> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D _{CS}	Demand Cost Savings	$\frac{\$ < \text{value41} >}{\text{yr}}$
C _T	Total Cost Savings	$\frac{\$ < \text{value42} >}{\text{yr}}$
R _S	Skylight Area Ratio	<value3>
F _L	Lighting Loss Factor	<value6>
R _C	Room Cavity Ratio	<value10>
K _R	Room Coefficient of Utilization	<value11>
K _U	Utilization Coefficient	<value13>
X _S	No. of Skylights	<value17> Skylights
C _C	Capital Cost	\$<value57>
C _I	Implementation Cost	\$<value44>

T_P	Simple Payback	<value45> yr (<value46> mo)
G_{CO2}	Equivalent CO ₂ Emissions Reductions	<value49> tons
G_{NOX}	Equivalent NO _x Emissions Reductions	<value50> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting is the product of the lights power consumption, ballast factor and the operational time, multiplied by the number of lights found.

$$E_C = (X_{<value18>, Fixture})(X_{<value18>, Bulb})(P_{<value18>})(B_{<value18>})(K_{W2KW})(T_O)$$

$$(< value19 > \text{ Fixtures}) \left(< value20 > \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(< value21 > \frac{W}{\text{Bulb}} \right) (< value22 >)$$

$$>) \left(< value23 > \frac{kW}{W} \right) \left(< value24 > \frac{\text{hrs}}{\text{yr}} \right) = < \mathbf{value25} > \frac{\mathbf{kWh}}{\mathbf{yr}}$$

Proposed Energy Usage

Sky lighting is typically useful from 9 a.m. to 3 p.m. on any given day, or about 6 hours per day, thus electrically based lighting should be used throughout <value53>% of the workday. <value27> has an average of <value28> days per year with adequate sunshine, or <value29>% of the time, thus electrical lighting will be required <value54>% of days throughout the year

$$E_P = (E_C)(F_{SKY})(F_{SUN})$$

$$\left(< value25 > \frac{\mathbf{kWh}}{\mathbf{yr}} \right) (< value26 >) (< value30 >) = < \mathbf{value31} > \frac{\mathbf{kWh}}{\mathbf{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$

$$\langle \text{value25} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value31} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value32} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$

$$\left(\langle \text{value32} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value34} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value35} \rangle}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, ballast factor and the number of lights found, operating on a monthly basis over the course of a year. Assuming that the lights are off during peak demand hours, skylights will reduce the amount of power required by the plant.

$$D_C = (X_{\langle \text{value18} \rangle, \text{Fixture}})(X_{\langle \text{value18} \rangle, \text{Bulb}})(P_{\langle \text{value18} \rangle})(B_{\langle \text{value18} \rangle})(K_{W2KW})(K_{M2YR})$$

$$\left(\langle \text{value19} \rangle \text{ Fixtures} \right) \left(\langle \text{value20} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value21} \rangle \frac{\text{W}}{\text{Bulb}} \right) \left(\langle \text{value22} \right.$$

$$\left. \rangle \right) \left(\langle \text{value23} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value36} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value37} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Since the lights are expected to be off during demand peak hours, there will be no electrical demand due to lighting

$$D_P = 0 \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$
$$\langle \text{value37} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value38} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value39} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value39} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value40} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value41} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value35} \rangle}{\text{yr}} + \frac{\$ \langle \text{value41} \rangle}{\text{yr}} = \frac{\$ \langle \text{value42} \rangle}{\text{yr}}$$

Skylight Area Ratio

The skylight area ratio is the quotient of the skylight area that allows for illumination to the total skylight area. It is the percent area that can transmit light. This value assumes that there is a 1.5" overlap on each side of the skylight.

$$R_S = \langle \text{value3} \rangle$$

Lighting Loss Factor

There are two factors that will decrease the maximum amount of allowable sunlight illumination. The room surface dirt depreciation coefficient, RSDD, takes into account the light absorbing properties of a room due to environmental factors such as cleanliness. The skylight dirt depreciation coefficient, RSD, takes into account the surface dirt buildup on the skylight over a period of time. The maximum amount of allowable sunlight to pass through the skylights will be diminished by these two factors.

$$F_L = (F_{RSDD})(F_{RSD})$$
$$(\langle \text{value4} \rangle)(\langle \text{value5} \rangle) = \langle \text{value6} \rangle$$

Room Cavity Ratio

The room cavity ratio quantifies how effective an area is at utilizing skylight illumination. It is a factor of the room's physical dimensions.

$$R_C = (5)(H) \left[\frac{L + W}{(L)(W)} \right]$$
$$(5)(\langle \text{value7} \rangle \text{ ft}) \left[\frac{\langle \text{value8} \rangle \text{ ft} + \langle \text{value9} \rangle \text{ ft}}{(\langle \text{value8} \rangle \text{ ft})(\langle \text{value9} \rangle \text{ ft})} \right] = \langle \text{value10} \rangle$$

Room Coefficient of Utilization

The coefficient of utilization takes into account the amount of skylight that finds its way to a horizontal surface. It is a function of the room cavity ratio.

$$K_R = \frac{1}{1 + (0.0995)(R_C)^{1.087}}$$

$$\frac{1}{1 + (0.0995)(\langle \text{value10} \rangle)^{1.087}} = \langle \text{value11} \rangle$$

Utilization Coefficient

The utilization coefficient is much like the room coefficient of utilization, but also takes into account the lighting transmitted through the skylight. It is a function of the room coefficient of utilization.

$$K_U = (\tau)(K_R)$$

$$(\langle \text{value12} \rangle)(\langle \text{value11} \rangle) = \langle \text{value13} \rangle$$

Number of Skylights

This calculation depicts the number of skylights required to produce an adequate illumination of $\langle \text{value14} \rangle \text{ ft} \cdot \text{cd}$.

$$X_S = \frac{(E_T)(L)(W)}{(E_H)(A_S)(R_S)(K_U)(F_L)}$$

$$\frac{(\langle \text{value14} \rangle \text{ ft} \cdot \text{cd})(\langle \text{value8} \rangle \text{ ft})(\langle \text{value9} \rangle \text{ ft})}{(\langle \text{value15} \rangle \text{ ft} \cdot \text{cd}) \left(\langle \text{value16} \rangle \frac{\text{ft}^2}{\text{Skylight}} \right) (\langle \text{value3} \rangle)(\langle \text{value13} \rangle)(\langle \text{value6} \rangle)} =$$

$$\langle \text{value17} \rangle \text{ Skylights}$$

Capital Cost

The capital cost is the cost associated with purchasing the number of photosensors used to control lighting when skylighting becomes inadequate.

$$C_C = (X_{\text{Photo}})(D_{\text{Photo}})$$

$$(< \text{value55} > \text{ Photosensors}) \left(\frac{\$ < \text{value56} >}{\text{Photosensor}} \right) = \$ < \text{value57} >$$

Implementation Cost

The implementation cost is the cost associated with purchasing and installing the <value17> skylights needed for proper illumination as well as the installation cost for the photosensors.

$$C_I = C_C + (X_{\text{Photo}})(T_{\text{Photo}})(D_L) + (X_S)(A_S)(D_S)$$

$$(\$ < \text{value57} >)$$

$$+ (< \text{value55} > \text{ Photosensors}) (< \text{value58}$$

$$> \frac{\text{hr}}{\text{Photosensor}}) \left(\frac{\$ < \text{value59} >}{\text{hr}} \right)$$

$$+ (< \text{value17} > \text{ Skylights}) \left(< \text{value16} > \frac{\text{ft}^2}{\text{Skylight}} \right) \left(\frac{\$ < \text{value43} >}{\text{ft}^2} \right) = \$$$

$$< \text{value44} >$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ < \text{value44} >}{\frac{\$ < \text{value42} >}{\text{yr}}} = < \text{value45} > \text{ yr } (< \text{value46} > \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$

$$\left(\langle \text{value32} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value47} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value49} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NO}_x} = (E_S)(E_{\text{NO}_x})$$

$$\left(\langle \text{value32} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value48} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value50} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 4

Install Skylights (Multiple Zones, Multiple Lighting Types)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value58>	\$<value65>	CO ₂ : <value49> tons NO _x : <value50> lbs	<value45> yr (<value46> mo)
<value57> kWh <value51>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Install skylights through out the various production areas of the facility.

Current Operations and Observations

On the day of the visit, IAC personnel observed various areas of the plant that could benefit from augmenting the existing roofing structure with skylights. Adding skylights will supplement artificial lighting while cutting illumination costs.

Calculations

The following example calculations have been done for the <value54> lights found in the <value55>. Usage and savings calculation variables can be found in the following table.

Table 4.1

<table1>

Table 4.2 exemplifies the calculation variables used to deduce the number of skylights per facility area.

Table 4.2

<table2>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\langle\text{value18}\rangle,\text{Fixture}}$	No. of $\langle\text{value18}\rangle$ Fixtures	Table 4.1
$X_{\langle\text{value18}\rangle,\text{Bulb}}$	No of $\langle\text{value18}\rangle$ Bulbs per Fixture	Table 4.1
$P_{\langle\text{value18}\rangle}$	$\langle\text{value18}\rangle$ Power	Table 4.1
$B_{\langle\text{value18}\rangle}$	$\langle\text{value18}\rangle$ Ballast Factor	Table 4.1
K_{W2KW}	Watt to Kilowatt Conversion Factor	$\langle\text{value23}\rangle \frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	$\langle\text{value36}\rangle \frac{\text{mo}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
F_{Sky}	Sky Lighting Factor	<value26 >
F_{Sun}	Sunshine Factor	<value30>
F_{RSDD}	Room Surface	<value4>
	Depreciation Coefficient	
F_{RSD}	Skylight Dirt	<value5>
	Depreciation Coefficient	
L	Room Length	Table 4.2
W	Room Width	Table 4.2
H	Room Height	Table 4.2
τ	Room Reflectance	<value13>
E_{T}	Proposed Illuminance Horizontal	<value14> ft•cd
E_{H}	Illuminance Produced by Sky	<value15> ft•cd
A_{S}	Gross Skylight Area	<value16> $\frac{\text{ft}^2}{\text{Skylight}}$
D_{S}	Installation Cost	$\frac{\$ < \text{value43} >}{\text{ft}^2}$

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ¹⁴	$\frac{\$ < \text{value34} >}{\text{kWh}}$
D_{L}	Maintenance Labor Cost ¹⁵	$\frac{\$ < \text{value29} >}{\text{hr}}$
T_{O}	Operating Hours ⁴	<value24> $\frac{\text{hrs}}{\text{yr}}$
E_{CO2}	CO ₂ Emission Factor ¹⁶	<value47> $\frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ⁶	<value48> $\frac{\text{lbs}}{\text{kWh}}$

¹⁴ See energy cost analysis section.

¹⁵ See summary of plant statistics section.

¹⁶ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	Table 4.1
E _P	Proposed Energy Usage	Table 4.1
E _S	Energy Savings	Table 4.1
E _{CS}	Energy Cost Savings	Table 4.1
C _T	Total Cost Savings	Table 4.1
R _S	Skylight Area Ratio	Table 4.2
F _L	Lighting Loss Factor	Table 4.2
R _C	Room Cavity Ratio	Table 4.2
K _R	Room Coefficient of Utilization	Table 4.2
K _U	Utilization Coefficient	Table 4.2
X _S	No. of Skylights	Table 4.2
C _I	Implementation Cost	Table 4.2
T _P	Simple Payback	<value45> yr (<value46> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value49> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value50> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting is the product of the lights power consumption, ballast factor and the operational time, multiplied by the number of lights found.

$$E_C = (X_{\text{Fixture}})(X_{\text{Bulb}})(P_{\text{W}})(B_{\text{KW}})(T_0)$$

$$(< \text{value19} > \text{ Fixtures}) \left(< \text{value20} > \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(< \text{value21} > \frac{\text{W}}{\text{Bulb}} \right) (< \text{value22} >)$$

$$>) \left(< \text{value23} > \frac{\text{kW}}{\text{W}} \right) \left(< \text{value24} > \frac{\text{hrs}}{\text{yr}} \right) = < \text{value25} > \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Current Energy Usage} = < \text{value55} > \frac{\text{kWh}}{\text{yr}}$$

Proposed Energy Usage

Sky lighting is typically useful from 9 a.m. to 3 p.m. on any given day, or about 6 hours per day, thus electrically based lighting should be used throughout <value53>% of the workday. <value27> has an average of <value28> days per year with adequate sunshine, or <value29>% of the time, thus electrical lighting will be required <value54>% of days throughout the year

$$E_P = (E_C)(F_{SKY})(F_{SUN})$$

$$\left(< \text{value25} > \frac{\text{kWh}}{\text{yr}} \right) (< \text{value26} >) (< \text{value30} >) = < \text{value31} > \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Proposed Energy Usage} = < \text{value56} > \frac{\text{kWh}}{\text{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_{CS} = E_C - E_P$$

$$< \text{value25} > \frac{\text{kWh}}{\text{yr}} - < \text{value31} > \frac{\text{kWh}}{\text{yr}} = < \text{value32} > \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Energy Savings} = < \text{value57} > \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{Elec})$$

$$\left(< \text{value32} > \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ < \text{value34} >}{\text{kWh}} \right) = \frac{\$ < \text{value35} >}{\text{yr}}$$

$$\text{Total Energy Cost Savings} = \frac{\$ < \text{value58} >}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS}$$

$$C_T = \frac{\$ \langle \text{value42} \rangle}{\text{yr}}$$

$$\text{Total Aggregate Cost Savings} = \frac{\$ \langle \text{value63} \rangle}{\text{yr}}$$

Skylight Area Ratio

The skylight area ratio is the quotient of the skylight area that allows for illumination to the total skylight area. It is the percent area that can transmit light. This value assumes that there is a 1.5" overlap on each side of the skylight.

$$R_S = \langle \text{value3} \rangle$$

Lighting Loss Factor

There are two factors that will decrease the maximum amount of allowable sunlight illumination. The room surface dirt depreciation coefficient, RSDD, takes into account the light absorbing properties of a room due to environmental factors such as cleanliness. The skylight dirt depreciation coefficient, RSD, takes into account the surface dirt buildup on the skylight over a period of time. The maximum amount of allowable sunlight to pass through the skylights will be diminished by these two factors.

$$F_L = (F_{RSDD})(F_{RSD})$$

$$(\langle \text{value4} \rangle)(\langle \text{value5} \rangle) = \langle \text{value6} \rangle$$

Room Cavity Ratio

The room cavity ratio quantifies how effective an area is at utilizing skylight illumination. It is a factor of the room's physical dimensions.

$$R_C = (5)(H) \left[\frac{L + W}{(L)(W)} \right]$$
$$(5)(\text{< value7 > ft}) \left[\frac{\text{< value8 > ft} + \text{< value9 > ft}}{(\text{< value8 > ft})(\text{< value9 > ft})} \right] = \text{< value10 >}$$

Room Coefficient of Utilization

The coefficient of utilization takes into account the amount of skylight that finds its way to a horizontal surface. It is a function of the room cavity ratio.

$$K_R = \frac{1}{1 + (0.0995)(R_C)^{1.087}}$$
$$\frac{1}{1 + (0.0995)(\text{< value10 >})^{1.087}} = \text{< value11 >}$$

Utilization Coefficient

The utilization coefficient is much like the room coefficient of utilization, but also takes into account the lighting being reflected off of the various room surfaces. It is a function of the room coefficient of utilization.

$$K_U = (\tau)(K_R)$$
$$(\text{< value12 >})(\text{< value11 >}) = \text{< value13 >}$$

Number of Skylights

This calculation depicts the number of skylights required to produce an adequate illumination of <value14> ft•cd.

$$X_S = \frac{(E_T)(L)(W)}{(E_H)(A_S)(R_S)(K_U)(F_L)}$$

$$\frac{(<value14> \text{ ft} \cdot \text{cd})(<value8> \text{ ft})(<value9> \text{ ft})}{(<value15> \text{ ft} \cdot \text{cd}) \left(<value16> \frac{\text{ft}^2}{\text{Skylight}} \right) (<value3>)(<value13>)(<value6>)} =$$

<value17> Skylights

Total Number of Skylights = **<value64> Skylights**

Implementation Cost

The implementation cost is the cost associated with purchasing and installing the <value17> skylights needed for proper illumination.

$$C_I = (X_S)(A_S)(D_S)$$

$$(<value17> \text{ Skylights}) \left(<value16> \frac{\text{ft}^2}{\text{Skylight}} \right) \left(\frac{\$ <value43>}{\text{ft}^2} \right) = \$ <value44>$$

Total Implementation Cost = \$ **<value65>**

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ <value65>}{\frac{\$ <value63>}{\text{yr}}} = <value45> \text{ yr } (<value46> \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$

$$\left(< \text{value57} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value47} > \frac{\text{tons}}{\text{kWh}} \right) = < \text{value49} > \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NO}_x} = (E_S)(E_{\text{NO}_x})$$

$$\left(< \text{value57} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value48} > \frac{\text{lbs}}{\text{kWh}} \right) = < \text{value50} > \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 5

Replace <value1> Illumination with <value2> Lights (Single Lighting Type)

(Prepared by <value39>)

A.R.C.: <value3>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value17>	\$<value23>			
<value15> kWh	<value21> kW•mo	\$<value30>	CO2: <value35> tons NOX: <value36> lbs	<value31> yr (<value32> mo)
<value37>%	<value38>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace the current <value6> W <value1> with more energy efficient <value2> lights.

Current Operations and Observations

It was estimated that <value4> <value1> fixtures and 30 were in use to light various locations around the plant. These lamps should be replaced with the higher efficiency <value2> lights, thus reducing the power requirement and energy use throughout the year.

Calculations

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{<value1>, \text{Fixture}}$	No. of <value1> Fixtures	<value4> Fixtures
$X_{<value1>, \text{Bulb}}$	No of <value1> Bulbs per Fixture	$<value5> \frac{\text{Bulbs}}{\text{Fixture}}$
$P_{<value1>}$	<value1> Power	<value6> W
$B_{<value1>}$	<value1> Ballast Factor	<value7>
K_{W2KW}	Watt to Kilowatt Conversion Factor	$<value8> \frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	$<value18> \frac{\text{mo}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\langle\text{value2}\rangle,\text{Fixture}}$	No. of $\langle\text{value2}\rangle$ Fixtures	$\langle\text{value11}\rangle$ Fixtures
$X_{\langle\text{value2}\rangle,\text{Bulb}}$	No of $\langle\text{value2}\rangle$ Bulbs per Fixture	$\langle\text{value12}\rangle \frac{\text{Bulbs}}{\text{Fixture}}$
$P_{\langle\text{value2}\rangle}$	$\langle\text{value2}\rangle$ Power	$\langle\text{value13}\rangle$ W
$B_{\langle\text{value2}\rangle}$	$\langle\text{value2}\rangle$ Ballast Factor	$\langle\text{value37}\rangle$
$D_{\langle\text{value2}\rangle,\text{Bulb}}$	Cost of $\langle\text{value2}\rangle$ Bulbs	$\frac{\$ \langle\text{value25}\rangle}{\text{Bulb}}$
$D_{\langle\text{value2}\rangle,\text{Fixture}}$	Cost of $\langle\text{value2}\rangle$ Fixtures	$\frac{\$ \langle\text{value26}\rangle}{\text{Fixture}}$

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ¹⁷	$\frac{\$<value16>}{kWh}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$<value22>}{kW\cdot mo}$
D_L	Maintenance Labor Cost ¹⁸	$\frac{\$<value29>}{hr}$
T_O	Operating Hours ⁴	$<value9> \frac{hrs}{yr}$
T_I	Installation Time ¹⁹	$<value28> \frac{hrs}{Fixture}$
E_{CO2}	CO ₂ Emission Factor ²⁰	$<value33> \frac{tons}{kWh}$
E_{NOX}	NO _x Emission Factor ⁶	$<value34> \frac{lbs}{kWh}$

¹⁷ See energy cost analysis section.

¹⁸ See summary of plant statistics section.

¹⁹ Estimate by IAC.

²⁰ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E_C	Current Energy Usage	$\langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}}$
E_P	Proposed Energy Usage	$\langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}}$
E_S	Energy Savings	$\langle \text{value15} \rangle \frac{\text{kWh}}{\text{yr}}$
E_{CS}	Total Energy Cost Savings	$\frac{\$ \langle \text{value17} \rangle}{\text{yr}}$
D_C	Current Demand Usage	$\langle \text{value19} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$
D_P	Proposed Demand Usage	$\langle \text{value20} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$
D_S	Demand Savings	$\langle \text{value21} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$
D_{CS}	Demand Cost Savings	$\frac{\$ \langle \text{value23} \rangle}{\text{yr}}$
C_T	Total Cost Savings	$\frac{\$ \langle \text{value24} \rangle}{\text{yr}}$
C_C	Capital Cost	$\$ \langle \text{value27} \rangle$
C_I	Implementation Cost	$\$ \langle \text{value30} \rangle$
T_P	Simple Payback	$\langle \text{value31} \rangle \text{ yr}$ $(\langle \text{value32} \rangle \text{ mo})$
G_{CO_2}	Equivalent CO ₂ Emissions Reductions	$\langle \text{value35} \rangle \text{ tons}$
G_{NO_x}	Equivalent NO _x Emissions Reductions	$\langle \text{value36} \rangle \text{ lbs}$

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting is the product of the lights power consumption, ballast factor and the operational time, multiplied by the number of lights found. The total current energy usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$E_C = (X_{\langle \text{value1} \rangle, \text{Fixture}})(X_{\langle \text{value1} \rangle, \text{Bulb}})(P_{\langle \text{value1} \rangle})(B_{\langle \text{value1} \rangle})(K_{W2KW})(T_0)$$

$$(\langle \text{value4} \rangle \text{ Fixtures}) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value6} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value7} \rangle) \left(\langle \text{value8} \right.$$

$$\left. \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value9} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}}$$

Proposed Energy Usage

Retrofitting the current lighting system with efficient <value2> illumination can save on electrical consumption while providing the same amount of light. The following calculation depicts <value2> lighting electrical consumption.

$$E_P = (X_{\langle \text{value2} \rangle, \text{Fixture}})(X_{\langle \text{value2} \rangle, \text{Bulb}})(P_{\langle \text{value2} \rangle})(B_{\langle \text{value2} \rangle})(K_{W2KW})(T_0)$$

$$(\langle \text{value11} \rangle \text{ Fixtures}) \left(\langle \text{value12} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value13} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value37} \right.$$

$$\left. \rangle) \left(\langle \text{value8} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value9} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$

$$\langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value15} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{Elec})$$

$$\left(< \text{value15} > \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ < \text{value16} >}{\text{kWh}} \right) = \frac{\$ < \text{value17} >}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, ballast factor and the number of lights found, operating on a monthly basis over the course of a year. The total current demand usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$D_C = (X_{< \text{value1} >, \text{Fixture}})(X_{< \text{value1} >, \text{Bulb}})(P_{< \text{value1} >})(B_{< \text{value1} >})(K_{W2KW})(K_{M2YR})$$

$$\left(< \text{value4} > \text{ Fixtures} \right) \left(< \text{value5} > \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(< \text{value6} > \frac{\text{W}}{\text{Bulb}} \right) \left(< \text{value7} > \right) \left(< \text{value8} > \frac{\text{kW}}{\text{W}} \right) \left(< \text{value18} > \frac{\text{mo}}{\text{yr}} \right) = < \text{value19} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Retrofitting the current lighting system with efficient <value2> illumination can save on demand consumption while providing the same amount of light. The following calculation depicts <value2> lighting demand consumption.

$$D_P = (X_{< \text{value2} >, \text{Fixture}})(X_{< \text{value2} >, \text{Bulb}})(P_{< \text{value2} >})(B_{< \text{value2} >})(K_{W2KW})(K_{M2YR})$$

$$\left(< \text{value11} > \text{ Fixtures} \right) \left(< \text{value12} > \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(< \text{value13} > \frac{\text{W}}{\text{Bulb}} \right) \left(< \text{value7} > \right) \left(< \text{value8} > \frac{\text{kW}}{\text{W}} \right) \left(< \text{value18} > \frac{\text{mo}}{\text{yr}} \right) = < \text{value20} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$

$$\langle \text{value19} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value20} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value21} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$

$$\left(\langle \text{value21} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value22} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value23} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$

$$\frac{\$ \langle \text{value17} \rangle}{\text{yr}} + \frac{\$ \langle \text{value23} \rangle}{\text{yr}} = \frac{\$ \langle \text{value24} \rangle}{\text{yr}}$$

Capital Cost

The capital cost is the amount of money needed to buy fixtures and bulbs.

$$C_c = (X_{\langle \text{value2} \rangle, \text{Fixture}}) [(X_{\langle \text{value2} \rangle, \text{Bulb}})(D_{\langle \text{value2} \rangle, \text{Bulb}}) + (D_{\langle \text{value2} \rangle, \text{Fixture}})]$$

$$(\langle \text{value11} \rangle \text{ Fixtures}) \left[\left(\langle \text{value12} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\frac{\$ \langle \text{value25} \rangle}{\text{Bulb}} \right) + \left(\frac{\$ \langle \text{value26} \rangle}{\text{Fixture}} \right) \right]$$

$$= \$ \langle \text{value27} \rangle$$

Implementation Cost

The implementation cost is the sum of the capital cost and the manpower cost associated with installation.

$$C_I = C_C + (X_{MH} + X_{LPS})(T_I)(D_L)$$
$$\$ \langle \text{value27} \rangle + (\langle \text{value11} \rangle \text{ Fixtures}) \left(\langle \text{value28} \rangle \frac{\text{hr}}{\text{Fixture}} \right) \left(\frac{\$ \langle \text{value29} \rangle}{\text{hr}} \right) = \$$$
$$\langle \text{value30} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$
$$\frac{\$ \langle \text{value30} \rangle}{\frac{\$ \langle \text{value24} \rangle}{\text{yr}}} = \langle \text{value31} \rangle \text{ yr } (\langle \text{value32} \rangle \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$
$$\left(\langle \text{value15} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value33} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value35} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_S)(E_{\text{NOX}})$$

$$\left(< \text{value15} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value34} > \frac{\text{lbs}}{\text{kWh}} \right) = < \text{value36} > \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 6

Replace <value1> Illumination with <value2> Lights (Multiple Lighting Types)

(Prepared by <value39>)

A.R.C.: <value3>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value45>	\$<value49>			
<value44> kWh	<value48> kW•mo	\$<value52>	CO2: <value35> tons NOX: <value36> lbs	<value31> > yr (<value32> mo)
<value37>%	<value38>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace the current <value1> lights with more energy efficient <value2> lights.

Current Operations and Observations

During the facility walkthrough, it was noticed that several <value1> lights were in use throughout the facility. Considering that there are energy efficient lighting alternatives that will produce the same amount of light while consuming less energy, we recommend that these lights be replaced with higher efficiency <value2> Lights.

Calculations

The following example calculations have been done for the retrofit from <value40> illumination to <value41> lights. Table 6.1 depicts the usage and savings calculations for all retrofitted lighting types found in the facility

Table 6.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\text{Current,Fixture}}$	No. of Current Fixtures	Table 6.1
$X_{\text{Current,Bulb}}$	No. of <value1 Bulbs per Fixture>	Table 6.1
P_{Current}	Current Power	Table 6.1
B_{Current}	Current Ballast Factor	Table 6.1
T_o	Lighting Operating Hours	Table 6.1
K_{W2KW}	Watt to Kilowatt Conversion Factor	<value8> $\frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	<value18> $\frac{\text{mo}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\text{Proposed,Fixture}}$	No. of Proposed Fixtures	Table 6.1
$X_{\text{Proposed,Bulb}}$	No. of Proposed Bulbs per Fixture	Table 6.1
P_{Proposed}	Proposed Power	Table 6.1
B_{Proposed}	Proposed Ballast Factor	Table 6.1
$D_{\text{Proposed,Bulb}}$	Cost of Proposed Bulbs	Table 6.1
$D_{\text{Proposed,Fixture}}$	Cost of Proposed Fixtures	Table 6.1

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ²¹	$\frac{\$ \langle \text{value16} \rangle}{\text{kWh}}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$ \langle \text{value22} \rangle}{\text{kW} \cdot \text{mo}}$
D_{L}	Maintenance Labor Cost ²²	$\frac{\$ \langle \text{value29} \rangle}{\text{hr}}$
T_{O}	Operating Hours ⁴	$\langle \text{value9} \rangle \frac{\text{hrs}}{\text{yr}}$
T_{I}	Installation Time ²³	$\langle \text{value28} \rangle \frac{\text{hrs}}{\text{Fixture}}$
E_{CO2}	CO ₂ Emission Factor ²⁴	$\langle \text{value33} \rangle \frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ⁶	$\langle \text{value34} \rangle \frac{\text{lbs}}{\text{kWh}}$

²¹ See energy cost analysis section.

²² See summary of plant statistics section.

²³ Estimate by IAC.

²⁴ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	Table 6.1
E _P	Proposed Energy Usage	Table 6.1
E _S	Energy Savings	Table 6.1
E _{CS}	Total Energy Cost Savings	Table 6.1
D _C	Current Demand Usage	Table 6.1
D _P	Proposed Demand Usage	Table 6.1
D _S	Demand Savings	Table 6.1
D _{CS}	Demand Cost Savings	Table 6.1
C _T	Total Cost Savings	Table 6.1
C _C	Capital Cost	Table 6.1
C _I	Implementation Cost	Table 6.1
T _P	Simple Payback	<value31> yr (<value32> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value35> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value36> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with lighting is the product of the lights power consumption, ballast factor and the operational time, multiplied by the number of lights found. The total current energy usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$E_C = (X_{\langle \text{value40} \rangle, \text{Fixture}})(X_{\langle \text{value40} \rangle, \text{Bulb}})(P_{\langle \text{value40} \rangle})(B_{\langle \text{value40} \rangle})(K_{W2KW})(T_{\langle \text{value40} \rangle})$$

$$(\langle \text{value4} \rangle \text{ Fixtures}) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value6} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value7} \rangle) \left(\langle \text{value8} \right.$$

$$\left. \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value9} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Current Energy Usage} = \langle \text{value42} \rangle \frac{\text{kWh}}{\text{yr}}$$

Proposed Energy Usage

Retrofitting the current lighting system with efficient $\langle \text{value41} \rangle$ illumination can save on electrical consumption while providing the same amount of light. The following calculation depicts Proposed lighting electrical consumption.

$$E_P = (X_{\langle \text{value41} \rangle, \text{Fixture}})(X_{\langle \text{value41} \rangle, \text{Bulb}})(P_{\langle \text{value41} \rangle})(B_{\langle \text{value41} \rangle})(K_{W2KW})(T_{\langle \text{value40} \rangle})$$

$$(\langle \text{value11} \rangle \text{ Fixtures}) \left(\langle \text{value12} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value13} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value53} \right.$$

$$\left. \rangle) \left(\langle \text{value8} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value9} \rangle \frac{\text{hrs}}{\text{yr}} \right) = \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Proposed Energy Usage} = \langle \text{value43} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$
$$\langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value15} \rangle \frac{\text{kWh}}{\text{yr}}$$
$$\text{Total Energy Savings} = \langle \text{value44} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$
$$\left(\langle \text{value15} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value16} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value17} \rangle}{\text{yr}}$$
$$\text{Total Energy Cost Savings} = \frac{\$ \langle \text{value45} \rangle}{\text{yr}}$$

Current Demand Usage

Demand usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, ballast factor and the number of lights found, operating on a monthly basis over the course of a year. The total current demand usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$D_C = (X_{\langle \text{value40} \rangle, \text{Fixture}})(X_{\langle \text{value40} \rangle, \text{Bulb}})(P_{\langle \text{value40} \rangle})(B_{\langle \text{value40} \rangle})(K_{W2KW})(K_{M2YR})$$

$$(\langle \text{value4} \rangle \text{ Fixtures}) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value6} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value7} \rangle) \left(\langle \text{value8} \right.$$

$$\left. \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value18} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value19} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Current Demand} = \langle \text{value46} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Retrofitting the $\langle \text{value40} \rangle$ lighting system with efficient $\langle \text{value41} \rangle$ illumination can save on demand while providing the same amount of light. The following calculation depicts $\langle \text{value41} \rangle$ lighting demand.

$$D_P = (X_{\langle \text{value41} \rangle, \text{Fixture}})(X_{\langle \text{value41} \rangle, \text{Bulb}})(P_{\langle \text{value41} \rangle})(B_{\langle \text{value41} \rangle})(K_{W2KW})(K_{M2YR})$$

$$(\langle \text{value11} \rangle \text{ Fixtures}) \left(\langle \text{value12} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value13} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value53} \right.$$

$$\left. \rangle) \left(\langle \text{value8} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value18} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value20} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = \langle \text{value47} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$
$$\langle \text{value19} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value20} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value21} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$
$$\text{Total Demand Difference} = \langle \text{value48} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value21} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value22} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value23} \rangle}{\text{yr}}$$
$$\text{Total Demand Cost Savings} = \frac{\$ \langle \text{value49} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value17} \rangle}{\text{yr}} + \frac{\$ \langle \text{value23} \rangle}{\text{yr}} = \frac{\$ \langle \text{value24} \rangle}{\text{yr}}$$
$$\text{Total Aggregate Cost Savings} = \frac{\$ \langle \text{value50} \rangle}{\text{yr}}$$

Capital Cost

The capital cost is the amount of money needed to buy fixtures and bulbs.

$$C_C = (X_{\langle \text{value41} \rangle, \text{Fixture}}) [(X_{\langle \text{value41} \rangle, \text{Bulb}})(D_{\langle \text{value41} \rangle, \text{Bulb}}) + (D_{\langle \text{value41} \rangle, \text{Fixture}})]$$

$$(\langle \text{value11} \rangle \text{ Fixtures}) \left[\left(\langle \text{value12} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\frac{\$ \langle \text{value25} \rangle}{\text{Bulb}} \right) + \left(\frac{\$ \langle \text{value26} \rangle}{\text{Fixture}} \right) \right]$$

$$= \$ \langle \text{value27} \rangle$$

$$\text{Total Capital Cost} = \$ \langle \text{value51} \rangle$$

Implementation Cost

The implementation cost is the sum of the capital cost and the manpower cost associated with installation.

$$C_I = C_C + (X_{\langle \text{value41} \rangle, \text{Fixture}})(T_I)(D_L)$$

$$\$ \langle \text{value27} \rangle + (\langle \text{value11} \rangle \text{ Fixtures}) \left(\langle \text{value28} \rangle \frac{\text{hr}}{\text{Fixture}} \right) \left(\frac{\$ \langle \text{value29} \rangle}{\text{hr}} \right) = \zeta$$

$$\langle \text{value30} \rangle$$

$$\text{Total Implementation Cost} = \$ \langle \text{value52} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ \langle \text{value52} \rangle}{\frac{\$ \langle \text{value50} \rangle}{\text{yr}}} = \langle \text{value31} \rangle \text{ yr } (\langle \text{value32} \rangle \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$
$$\left(\langle \text{value44} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value33} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value35} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NO}_x} = (E_S)(E_{\text{NO}_x})$$
$$\left(\langle \text{value44} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value34} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value36} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 7

Turn Off Lights and Install Occupancy Sensors

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value20>	\$<value30>			<value45>
<value17> kWh	<value27> kW	\$<value44>	CO ₂ : <value35> tons NO _x : <value36> lbs	yr (<value46> mo)
<value37>%	<value38>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Turn off the various types of lights, depicted in Table 7.1 and use occupancy sensors to control when lighting operational time.

Current Operations and Observations

During the plant visit, it was observed that there were various types of lights that were left on. Turning off lights when they are unnecessary provide electrical savings. Over the course of a year, these savings add up and can be substantial. Additionally, installing occupancy sensors to control when lighting is necessary can guarantee that lights are turned off when unnecessary.

Calculations

The following calculations outline the savings measures associated with <value3> lighting. Table .1 depicts the relevant calculations for the different lights that are to be turned off throughout the facility.

Table 7.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$X_{\langle\text{value3}\rangle,\text{Fixture}}$	No. of <value3> Fixtures	Table 7.1
$X_{\langle\text{value3}\rangle,\text{Bulb}}$	No. of <value3> Bulbs per Fixture	Table 7.1
$P_{\langle\text{value3}\rangle}$	<value3> Power	Table 7.1
$B_{\langle\text{value3}\rangle}$	<value3> Ballast	Table 7.1
$K_{\text{On,Current}}$	Percent <value3> Lighting is Currently Left On	<value47>
K_{W2KW}	Watt to Kilowatt Conversion Factor	<value9> $\frac{\text{kW}}{\text{W}}$
K_{M2YR}	Month to Year Conversion Factor	$12 \frac{\text{mo}}{\text{yr}}$

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$K_{On,Proposed}$	Percent <value3> Lighting is Proposed to be Left On	<value13>

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ²⁵	$\frac{\$<value18>}{\text{kWh}}$
D_{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$<value28>}{\text{kW}\cdot\text{mo}}$
D_L	Maintenance Cost	$\frac{\$<value48>}{\text{hr}}$
E_{CO2}	CO_2 Emission Factor ²⁶	$<value33> \frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO_x Emission Factor ⁶	$<value34> \frac{\text{lbs}}{\text{kWh}}$

²⁵ See energy cost analysis section.

²⁶ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	Table 7.1
E _P	Proposed Energy Usage	Table 7.1
E _S	Energy Savings	Table 7.1
E _{CS}	Total Energy Cost Savings	Table 7.1
D _C	Current Demand Usage	Table 7.1
D _P	Proposed Demand Usage	Table 7.1
D _S	Demand Savings	Table 7.1
D _{CS}	Demand Cost Savings	Table 7.1
C _T	Total Cost Savings	Table 7.1
C _C	Capital Cost	\$<value41>
C _I	Implementation Cost	\$0
T _P	Simple Payback	Immediate
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value35> $\frac{\text{tons}}{\text{yr}}$
G _{NOX}	Equivalent NO _x Emissions Reductions	<value36> $\frac{\text{lbs}}{\text{yr}}$

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy use associated with <value3> lighting is the product of the lights power consumption, ballast factor and the operational time, multiplied by the number of lights found. The total current energy usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$E_C = (X_{<value3>,Fixture})(X_{<value3>,Bulb})(P_{<value3>})(B_{<value3>})(K_{On,Current})(K_{W2KW})(T_O)$$

$$(<value4> \text{ Fixtures}) \left(<value5> \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(<value6> \frac{\text{W}}{\text{Bulb}} \right) (<value7>) (<value47>)$$

$$>) \left(<value9> \frac{\text{kW}}{\text{W}} \right) \left(<value10> \frac{\text{hrs}}{\text{yr}} \right) = <value11> \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Current Energy Usage} = <value12> \frac{\text{kWh}}{\text{yr}}$$

Proposed Energy Usage

By turning off lights, utility savings are realized through a reduction in electrical usage. The following calculation assumes that the lights will need to be operated at a significantly reduced time in comparison to their current usage.

$$E_P = (X_{<value3>,Fixture})(X_{<value3>,Bulb})(P_{<value3>})(B_{<value3>})(K_{On,Proposed})(K_{W2KW})(T_O)$$

$$(<value4> \text{ Fixtures}) \left(<value5> \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(<value6> \frac{\text{W}}{\text{Bulb}} \right) (<value7>) (<value13>)$$

$$>) \left(<value9> \frac{\text{kW}}{\text{W}} \right) \left(<value10> \frac{\text{hrs}}{\text{yr}} \right) = <value14> \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Proposed Energy Usage} = <value15> \frac{\text{kWh}}{\text{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$
$$\langle \text{value11} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}}$$
$$\text{Total Energy Savings} = \langle \text{value17} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$E_{CS} = (E_S)(D_{Elec})$$
$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value18} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value19} \rangle}{\text{yr}}$$
$$\text{Total Energy Cost Savings} = \frac{\$ \langle \text{value20} \rangle}{\text{yr}}$$

Current Demand Usage

Demand usage refers to the rate at which a facility uses energy. The demand use associated with lighting is the product of the lights power consumption, ballast factor and the number of lights found, operating on a monthly basis over the course of a year. The total current demand usage is found to be the sum of the energy usage for the different lighting types found throughout the facility.

$$D_C = (X_{\langle \text{value3} \rangle, \text{Fixture}})(X_{\langle \text{value3} \rangle, \text{Bulb}})(P_{\langle \text{value3} \rangle})(B_{\langle \text{value3} \rangle})(K_{\text{On, Current}})(K_{\text{M2YR}})$$

$$(\langle \text{value4} \rangle \text{ Fixtures}) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value6} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value7} \rangle) (\langle \text{value47} \rangle)$$

$$\left(\langle \text{value9} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value21} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Current Demand Usage} = \langle \text{value23} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

In addition to electrical usage savings, turning off lights will generate electrical demand savings as well. The following calculation depicts the proposed <value3> lighting demand usage.

$$D_P = (X_{\langle \text{value3} \rangle, \text{Fixture}})(X_{\langle \text{value3} \rangle, \text{Bulb}})(P_{\langle \text{value3} \rangle})(B_{\langle \text{value3} \rangle})(K_{\text{On, Proposed}})(K_{\text{M2YR}})$$

$$(\langle \text{value4} \rangle \text{ Fixtures}) \left(\langle \text{value5} \rangle \frac{\text{Bulbs}}{\text{Fixture}} \right) \left(\langle \text{value6} \rangle \frac{\text{W}}{\text{Bulb}} \right) (\langle \text{value7} \rangle) (\langle \text{value13} \rangle)$$

$$\left(\langle \text{value9} \rangle \frac{\text{kW}}{\text{W}} \right) \left(\langle \text{value21} \rangle \frac{\text{mo}}{\text{yr}} \right) = \langle \text{value24} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = \langle \text{value25} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$
$$\langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value24} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value26} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$
$$\text{Total Demand Savings} = \langle \text{value27} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value26} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value28} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value29} \rangle}{\text{yr}}$$
$$\text{Total Demand Cost Savings} = \frac{\$ \langle \text{value30} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value19} \rangle}{\text{yr}} + \frac{\$ \langle \text{value29} \rangle}{\text{yr}} = \frac{\$ \langle \text{value31} \rangle}{\text{yr}}$$
$$\text{Aggregated Total Cost Savings} = \frac{\$ \langle \text{value32} \rangle}{\text{yr}}$$

Capital Cost

The capital cost is the cost associated with purchasing occupancy sensors to control the various lighting zones.

$$C_C = (X_{\text{Sens}})(C_{\text{Sens}})$$
$$(< \text{value39} > \text{ Sensors}) \left(\frac{\$ < \text{value40} >}{\text{Sensor}} \right) = \$ < \text{value41} >$$

Implementation Cost

The implementation cost is the capital cost and the cost associated with installing the sensors.

$$C_I = C_C + (X_{\text{Sens}})(T_N)(D_L)$$
$$\$ < \text{value41} > + (< \text{value39} > \text{ Sensors})(< \text{value42} > \text{ hrs}) \left(\frac{\$ < \text{value48} >}{\text{hr}} \right) = \$$$
$$< \text{value44} >$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$
$$\frac{\$ < \text{value44} >}{\frac{\$ < \text{value31} >}{\text{yr}}} = < \text{value45} > \text{ yrs } (< \text{value46} > \text{ mo})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$
$$\left(< \text{value17} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value33} > \frac{\text{tons}}{\text{kWh}} \right) = < \text{value35} > \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_S)(E_{\text{NOX}})$$

$$\left(\langle \text{value17} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value34} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value36} \rangle \frac{\text{lbs}}{\text{yr}}$$

APPENDIX C: COMPRESSED AIR ASSESSMENT RECOMMENDATIONS

Assessment Recommendation No. 8

Repair Compressed Air Leaks

(Prepared by <value1>)

A.R.C: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value16>	\$<value20>			<value23>
< value14 > kWh	<value18> kW•mo	\$<value22>	CO ₂ :<value27> tons NO _x :<value28> lbs	> yr (<value24> mo)
<value29>%	<value30>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Repair existing air leaks. Regularly inspect and perform necessary maintenance on the compressed air system to prevent any further leaks.

Current Operations and Observations

During the visit, the IAC personnel noticed a compressed air system using a total of <value31> hp in addition to compressed air leaks in the facility. Repairing air leaks will reduce the load on the compressor, generating energy and cost savings. A maintenance program consisting of a tagging protocol or incentive based reporting of the leaks is recommended to reduce future leaks.

Calculations

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
V_A	Total Tank Volume ²⁷	<value3> ft ³
V_L	Total Line Volume ²⁷	<value4> ft ³
P_C	Pressure Drop	<value6> psi
P_{atm}	Atmospheric Pressure	<value7> psi
T_D	Pressure Drop Time ²⁸	<value8> min
η_C	Compressor Specific Efficiency	<value11> $\frac{\text{kW}}{\text{SCFM}}$
F_L	Load Factor ²⁹	<value12>
F_D	Duty Factor	<value32>
η_M	Motor Efficiency ³⁰	<value13>
K_{M2YR}	Month to Year Conversion Factor	<value17> $\frac{\text{mo}}{\text{yr}}$

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_E	Avoided cost of electrical energy ³¹	$\$ < \text{value15} >$ $\frac{\text{kWh}}{\text{kWh}}$
D_D	Avoided cost of electrical demand ³¹	$\$ < \text{value19} >$ $\frac{\text{kW} \cdot \text{mo}}{\text{kW} \cdot \text{mo}}$
T_O	Operating Hours ³²	<value10> $\frac{\text{hrs}}{\text{yr}}$
E_{CO2}	CO ₂ Emission Factor ³³	<value25> $\frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ³³	<value26> $\frac{\text{lbs}}{\text{kWh}}$

²⁷ Measured by IAC Personnel.

²⁸ Measured by Plant Personnel.

²⁹ Estimated by IAC Personnel.

³⁰ Motor Efficiency was deduced using MotorMaster + International.

³¹ See energy cost analysis section.

³² See summary of plant statistics.

³³ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
V _T	Total System Volume	<value5> ft ³
L _T	Total Leakage Rate	<value9> SCFM
E _S	Energy Savings	<value14> $\frac{\text{kWh}}{\text{yr}}$
C _S	Energy Cost Savings	$\frac{\$ < \text{value16} >}{\text{yr}}$
D _S	Demand Savings	<value18> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D _{CS}	Demand Cost Savings	$\frac{\$ < \text{value20} >}{\text{yr}}$
C _T	Total Cost Savings	$\frac{\$ < \text{value21} >}{\text{yr}}$
C _I	Implementation Cost	\$<value22>
T _P	Simple Payback	<value23> yr (<>value24 mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value27> $\frac{\text{tons}}{\text{yr}}$
G _{NOX}	Equivalent NO _x Emissions Reductions	<value28> $\frac{\text{lbs}}{\text{yr}}$

Total Volume

The total volume is the sum of the volume of the receiver tank and the pipeline volume. Note that only major compressed air lines were counted. Including the smaller lines will increase the total volume, which will increase the measured leakage rate. Therefore, not including the smaller lines provides a conservative estimate.

$$V_T = V_A + V_L$$

$$(< \text{value3} > \text{ ft}^3) + (< \text{value4} > \text{ ft}^3) = < \text{value5} > \text{ ft}^3$$

Leakage Rate

The leakage rate is calculated as follows: During a time where there is no production use of compressed air, the compressor is turned off. Due to the presence of leaks, the pressure in the system will drop. The time taken by the compressor pressure to drop by <value6> psi is recorded. The leakage rate is related to the volume of air within the system, pressure drop, atmospheric pressure and time taken for the pressure drop.

$$L_T = \frac{(P_C)(V_T)}{(P_{atm})(T_D)}$$

$$\frac{(< \text{value6} > \text{ psi})(< \text{value5} > \text{ ft}^3)}{(< \text{value7} > \text{ psi})(< \text{value8} > \text{ min})} = < \text{value9} > \text{ SCFM}$$

Energy Savings

The energy saved is the energy required to compress the leaked air and is the product of the leakage rate, operating hours, factor relating leakage rate to power consumption, and load factor, all divided by the efficiency. Note that this number assumes that the air compressors only remain on during normal operating hours. If the compressors are left on during other hours of the day, the results will be proportionally larger.

$$E_S = \frac{(L_T)(T_O)(\eta_C)(F_L)(F_D)}{\eta_M}$$

$$\frac{(< \text{value9} > \text{ SCFM}) \left(< \text{value10} > \frac{\text{hrs}}{\text{yr}} \right) \left(< \text{value11} > \frac{\text{kW}}{\text{SCFM}} \right) (< \text{value12} >) (< \text{value32} >)}{< \text{value13} >}$$

$$= < \text{value14} > \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The total energy cost savings is found by multiplying the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_E)$$

$$\left(\langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value15} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value16} \rangle}{\text{yr}}$$

Demand Savings

The demand saved is the demand required to compress the leaked air. It is the product of the leakage rate, factor relating leakage rate to power consumption, load factor, and a month-to-year conversion, all divided by the efficiency.

$$D_S = \frac{(L_T)(K_{M2YR})(\eta_C)(F_L)(F_L)}{\eta_M}$$

$$\frac{(\langle \text{value9} \rangle \text{ SCFM}) \left(\langle \text{value17} \rangle \frac{\text{mo}}{\text{yr}} \right) \left(\langle \text{value11} \rangle \frac{\text{kW}}{\text{SCFM}} \right) (\langle \text{value12} \rangle) (\langle \text{value32} \rangle)}{\langle \text{value13} \rangle}$$

$$= \langle \text{value18} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of electrical demand.

$$D_{CS} = (D_S)(D_D)$$

$$\left(\langle \text{value18} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value19} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value20} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the electrical cost savings and the demand cost savings.

$$C_T = C_S + D_{CS}$$
$$\frac{\$ \langle \text{value16} \rangle}{\text{yr}} + \frac{\$ \langle \text{value20} \rangle}{\text{yr}} = \frac{\$ \langle \text{value21} \rangle}{\text{yr}}$$

Implementation Cost

The implementation cost accounts for the costs of repairing pipes, hoses, clamps, etc. This amount includes an estimate to cover the cost of materials and for the labor to find and repair the leaks in the compressed air system; most of the repairs can be handled by the maintenance staff.

$$C_I = \$ \langle \text{value22} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the total cost savings.

$$T_P = \frac{C_I}{C_T}$$
$$\frac{\$ \langle \text{value22} \rangle}{\frac{\$ \langle \text{value21} \rangle}{\text{yr}}} = \langle \text{value23} \rangle \text{ yr } (\langle \text{value24} \rangle \text{ mo})$$

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (E_S)(E_{\text{CO}_2})$$
$$\left(\langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value25} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value27} \rangle \frac{\text{tons}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NO}_x} = (E_S)(E_{\text{NO}_x})$$
$$\left(\langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value26} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value28} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 9

Reduce Compressed Air Pressure

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value18>	\$<value24>			
<value16> kWh	<value22> kW•mo	\$<value28>	CO ₂ : <value31> tons NO _x : <value32> lbs	<value29> > yr (<value30>mo)
<value38>%	<value39>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Reduce the plant's compressed air system pressure from <value3> psig to <value4> psig.

Current Operations and Observations

During the plant visit, IAC personnel observed a total motor power of <value6> hp dedicated to compressing air. Under many circumstances, plants opt to run their systems at higher pressures to make up for the pressure loss due to compressed air leaks. Repairing leaks and decreasing the system pressure can realize cost savings and pollution reduction. Many industrial systems require no more than <value4> psig, thus this recommendation will exemplify a pressure reduction from <value3> psig to <value4> psig.

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P_C	Compressor Power	<value6> hp
X_C	No. of Compressors	<value36> Compressors
F_L	Compressor Load Factor ³⁴	<value8>
F_D	Compressor Duty Factor ³⁴	<value9>
η_m	Motor Efficiency ³⁵	<value37>
$R_{<value3>}$	Current System Pressure	<value3> psi
R_{atm}	Atmospheric Pressure	<value11> psi
k	Specific Heat	<value12>
K_{hp2kW}	Horsepower to Kilowatt Conversion Factor	<value7> $\frac{kW}{hp}$
K_{mo2yr}	Month to Year Conversion Factor	<value19> $\frac{mo}{yr}$

³⁴ Estimated by IAC Personnel.

³⁵ Value taken from Motormaster International +.

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
R _{<value4>}	Proposed System Pressure	<value4> psi

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D _{ELEC}	Avoided Cost of Electrical Energy ³⁶	$\frac{\$<value17>}{\text{kWh}}$
D _{DEM}	Avoided Cost of Electrical Demand ³	$\frac{\$<value23>}{\text{kW}\cdot\text{mo}}$
D _L	Maintenance Labor Cost ³⁷	$\frac{\$<value27>}{\text{hr}}$
T _O	Operating Hours ⁴	<value10> $\frac{\text{hrs}}{\text{yr}}$
T _I	Adjustment Time ³⁴	<value26> $\frac{\text{hr}}{\text{Compressor}}$
E _{CO2}	CO ₂ Emission Factor ³⁸	<value33> $\frac{\text{tons}}{\text{kWh}}$
E _{NOX}	NO _x Emission Factor ⁶	<value34> $\frac{\text{lbs}}{\text{kWh}}$

³⁶ See energy cost analysis section.

³⁷ See summary of plant statistics section.

³⁸ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E_C	Current Energy Usage	<value13> $\frac{\text{kWh}}{\text{yr}}$
E_P	Proposed Energy Usage	<value15> $\frac{\text{kWh}}{\text{yr}}$
η_C	Percent Energy Savings Due to Pressure Reduction	<value14>
E_S	Energy Savings	<value16> $\frac{\text{kWh}}{\text{yr}}$
E_{CS}	Total Energy Cost Savings	$\frac{\$<value18>}{\text{yr}}$
D_C	Current Demand Usage	<value20> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D_P	Proposed Demand Usage	<value21> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D_S	Demand Savings	<value22> $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$
D_{CS}	Demand Cost Savings	$\frac{\$<value24>}{\text{yr}}$
C_T	Total Cost Savings	$\frac{\$<value25>}{\text{yr}}$
C_C	Capital Cost	$\$<value35>$
C_I	Implementation Cost	$\$<value28>$
T_P	Simple Payback	<value29> yr (<value30> mo)
G_{CO_2}	Equivalent CO ₂ Emissions Reductions	<value31> tons
G_{NOX}	Equivalent NO _x Emissions Reductions	<value32> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). For a compressor, this is equivalent to the total compressor power needed to produce pressurized air over the operational period throughout the year.

$$E_C = \frac{(P_C)(K_{hp2W})(F_L)(F_D)(T_O)}{\eta_m}$$

$$\frac{(< \text{value6} > \text{ hp}) \left(< \text{value7} > \frac{\text{kW}}{\text{hp}} \right) (< \text{value8} >) (< \text{value9} >) \left(< \text{value10} > \frac{\text{hrs}}{\text{yr}} \right)}{< \text{value37} >} =$$

$$< \text{value13} > \frac{\text{kWh}}{\text{yr}}$$

Percent Energy Savings due to Pressure Reduction

Reducing the compressed air pressure will reduce the load on the compressed air motor. Assuming adiabatic and reversible compression, the brake horsepower reduction is the ratio of the work needed to produce compressed air at <value3> psi and the reduced work required to produce compressed air at <value4> psi.

$$\eta_C = \frac{1 - \left(\frac{R_{< \text{value4} >}}{R_{\text{atm}}} \right)^{\frac{k-1}{k}}}{1 - \left(\frac{R_{< \text{value3} >}}{R_{\text{atm}}} \right)^{\frac{k-1}{k}}}$$

$$\frac{1 - \left(\frac{< \text{value4} > \text{ psi}}{< \text{value11} > \text{ psi}} \right)^{\frac{< \text{value12} > - 1}{< \text{value12} >}}}{1 - \left(\frac{< \text{value3} > \text{ psi}}{< \text{value11} > \text{ psi}} \right)^{\frac{< \text{value12} > - 1}{< \text{value12} >}}} = < \text{value14} > \frac{\text{kWh}}{\text{yr}}$$

Proposed Energy Usage

The proposed energy usage is the current energy usage reduced by the percent energy savings due to pressure reduction.

$$E_P = (E_C)(1 - \eta_C)$$

$$\left(< \text{value13} > \frac{\text{kWh}}{\text{yr}} \right) (1 - < \text{value14} >) = < \text{value15} > \frac{\text{kWh}}{\text{yr}}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_S = E_C - E_P$$

$$< \text{value13} > \frac{\text{kWh}}{\text{yr}} - < \text{value15} > \frac{\text{kWh}}{\text{yr}} = < \text{value16} > \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$

$$\left(< \text{value16} > \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ < \text{value17} >}{\text{kWh}} \right) = \frac{\$ < \text{value18} >}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. Assuming that the compressor is operational and contributes to the monthly maximum demand, the total demand for the year is the compressor power over every month through out the year.

$$D_C = \frac{(P_C)(K_{hp2W})(F_L)(F_D)(K_{mo2yr})}{\eta_m}$$

$$\frac{(< \text{value6} > \text{ hp}) \left(< \text{value7} > \frac{\text{kW}}{\text{hp}} \right) (< \text{value8} >) (< \text{value9} >) \left(< \text{value19} > \frac{\text{mo}}{\text{yr}} \right)}{< \text{value37} >} =$$

$$< \text{value20} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Since there is a reduction in the compressor load to be had by reducing the compressed air pressure, the motor will require less power. Thus, the demand will also decrease by the percent savings due to reducing the compressed air pressure.

$$D_P = (D_C)(1 - \eta_C)$$

$$\left(< \text{value20} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) (1 - < \text{value14} >) = < \text{value21} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$

$$< \text{value20} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - < \text{value21} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = < \text{value22} > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value23} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value24} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value18} \rangle}{\text{yr}} + \frac{\$ \langle \text{value24} \rangle}{\text{yr}} = \frac{\$ \langle \text{value25} \rangle}{\text{yr}}$$

Capital Cost

The capital cost is the amount of money needed to components to implement this recommendation. Since implementation is based on changing the control scheme on the compressor, there are no components to be purchased, thus no capital cost to be incurred.

$$C_c = \langle \text{value35} \rangle \$$$

Implementation Cost

The implementation cost is the sum of the capital cost and the manpower cost associated with reducing the air pressure.

$$C_I = C_c + (X_c)(T_I)(D_L)$$
$$\$0 + (\langle \text{value36} \rangle \text{ Compressors}) \left(\langle \text{value26} \rangle \frac{\text{hr}}{\text{Compressor}} \right) \left(\frac{\$ \langle \text{value27} \rangle}{\text{hr}} \right) = \$$$
$$\langle \text{value28} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ \langle \text{value28} \rangle}{\frac{\$ \langle \text{value25} \rangle}{\text{yr}}} = \langle \text{value29} \rangle \text{ yr } (\langle \text{value30} \rangle \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value33} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value31} \rangle \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{NO_x} = (E_S)(E_{NO_x})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value34} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value32} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 10

Replace Standard Blow Off Nozzles with Engineered Nozzles

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$ < value16 >	\$<value22>			<value29
<value13> kWh	<value19> kW · mo	\$<value28>	CO ₂ : <value33>tons NO _x : <value34> lbs	> yr (<value3 0> mo)
<value35>%	<value36>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Install engineered nozzles on line machines to reduce the amount of compressed air lost.

Current Operations and Observations

IAC personnel observed the use of blow off nozzles in throughout various areas of the facility. Exchanging these for more efficient engineered nozzles will significantly impact the amount of compressed air needed. Engineered nozzles entrain outside air to the flow stream, allowing for the same blow off force, while decreasing the amount of compressed air. Decreasing the amount of compressed air needed minimizes the load on the air compressor, thus saving resources and electrical costs.

Calculations

The following depict sample calculations for <value34>” nozzle(s). The following table outlines the electrical usage and savings variables for all nozzle types found in the facility.

Table 10.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
V_{AF}	Nozzle Volumetric Flow Rate ³⁹	Table 10.1
X_N	No. nozzles in use ³	Table 10.1
R_A	Amplification Ratio	<value5> CFM
O_N	Nozzle Operational Use ³	<value8> $\frac{\text{hrs}}{\text{yr}}$
E_S	Compressor Specific Efficiency ¹	<value9> $\frac{\text{kW}}{\text{CFM}}$
E_M	Motor Efficiency ⁴⁰	<value11>
L	Compressor Load Factor ²	<value10>
C_N	Nozzle Cost ⁴¹	$\frac{\$<value24>}{\text{Nozzle}}$
T_N	Nozzle Installation Time ²	<value26> $\frac{\text{hrs}}{\text{Nozzle}}$
F_N	Nozzle Duty Factor	<value40>

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_E	Avoided Cost of Electrical Energy ⁴²	$\frac{\$<value14>}{\text{kWh}}$
D_D	Avoided Cost of Electrical Demand ⁵	$\frac{\$<value20>}{\text{kW}\cdot\text{mo}}$
D_L	Maintenance Labor Cost ⁴³	$\frac{\$<value27>}{\text{hr}}$
G_{CO_2}	CO ₂ Emission Factor ⁴⁴	<value31> $\frac{\text{tons}}{\text{kWh}}$
G_{NO_x}	NO _x Emission Factor ⁷	<value32> $\frac{\text{lbs}}{\text{kWh}}$

³⁹ Value gathered from discharge of air through orifice specification sheet from Sullair

⁴⁰ Estimate by IAC personnel

⁴¹ Average cost for various online sources

⁴² See energy cost analysis section

⁴³ See summary of plant statistics section

⁴⁴ See energy management and emission reduction section

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
V_{SAV}	Airflow Savings	Table 10.1
S_E	Electrical Energy Savings	Table 10.1
A_E	Electrical Cost Savings	Table 10.1
S_D	Demand Savings	Table 10.1
A_D	Demand Cost Savings	Table 10.1
C_T	Total Cost Savings	Table 10.1
C_C	Capital Cost	Table 10.1
C_I	Implementation Cost	Table 10.1
T_P	Simple Payback	<value29> yr (<value30> mo)
G_{CO2}	Equivalent CO ₂ Emissions Reductions	<value33> $\frac{\text{tons}}{\text{yr}}$
G_{NOX}	Equivalent NO _x Emissions Reductions	<value34> $\frac{\text{lbs}}{\text{yr}}$

Airflow Savings

Engineered nozzles entrain atmospheric air into the flow stream, reducing the need for compressed air. The reduction in compressed air needed is based on an amplification ratio.

$$V_{SAV} = (X_N)(V_{AF}) \left[1 - \left(\frac{1}{R_A} \right) \right]$$

$$(< \text{value3} > \text{ Nozzle}) \left(< \text{value4} > \frac{\text{CFM}}{\text{Nozzle}} \right) \left[1 - \left(\frac{1}{< \text{value5} > \frac{\text{CFM}}{\text{CFM}}} \right) \right] = < \text{value6} > \text{ CFM}$$

$$\text{Total Airflow Savings} = < \text{value7} > \text{ CFM}$$

Electrical Energy Savings

Electrical energy savings from using engineered nozzles are derived from the reduction in compressed air needed. Using less compressed air means the compressor has to work less, thus less energy is consumed.

$$S_E = \frac{(V_{SAV})(O_N)(E_S)}{(E_M)}$$

$$\frac{(< \text{value6} > \text{ CFM}) \left(< \text{value8} > \frac{\text{hrs}}{\text{yr}} \right) \left(< \text{value9} > \frac{\text{kW}}{\text{CFM}} \right) (< \text{value10} >)}{(< \text{value11} >)} = < \text{value12} >$$

$$> \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Electrical Energy Savings} = < \text{value13} > \frac{\text{kWh}}{\text{yr}}$$

Electrical Energy Cost Savings

The electrical energy cost savings are found to be the product of the electrical energy savings and the avoided cost of electrical energy.

$$A_E = (S_E)(D_E)$$

$$\left(\langle \text{value12} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value14} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value15} \rangle}{\text{yr}}$$

$$\text{Total Electrical Energy Cost Savings} = \frac{\$ \langle \text{value16} \rangle}{\text{yr}}$$

Demand Savings

Demand is the rate at which energy is consumed. Reducing the work necessary for the compressor to produce reduces the demand. Since industrial facilities are charged for demand, there are demand savings associated with using less compressed air.

$$S_D = \frac{[(V_{SAV})(E_S)(K_{M2YR})(L)(F_N)]}{(E_M)}$$

$$\frac{(\langle \text{value6} \rangle \text{ CFM}) \left(\langle \text{value9} \rangle \frac{\text{kW}}{\text{CFM}} \right) \left(\langle \text{value17} \rangle \frac{\text{mo}}{\text{yr}} \right) (\langle \text{value10} \rangle) (\langle \text{value40} \rangle)}{(\langle \text{value11} \rangle)} =$$

$$\langle \text{value18} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Demand Savings} = \langle \text{value19} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of electrical demand.

$$A_D = (S_D)(D_D)$$
$$\left(\langle \text{value18} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value20} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value21} \rangle}{\text{yr}}$$
$$\text{Total Demand Cost Savings} = \frac{\$ \langle \text{value22} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the electrical cost savings and the demand cost savings.

$$C_T = D_S + S_D$$
$$\frac{\$ \langle \text{value15} \rangle}{\text{yr}} + \frac{\$ \langle \text{value21} \rangle}{\text{yr}} = \frac{\$ \langle \text{value37} \rangle}{\text{yr}}$$
$$\text{Aggregate Total Cost Savings} = \frac{\$ \langle \text{value23} \rangle}{\text{yr}}$$

Capital Cost

The capital cost includes the expense of the new nozzles. The plant will need to purchase them for each compressed air line used.

$$C_C = (X_N)(C_N)$$
$$(\langle \text{value3} \rangle \text{ Nozzles}) \left(\frac{\$ \langle \text{value24} \rangle}{\text{Nozzle}} \right) = \$ \langle \text{value25} \rangle$$
$$\text{Total Capital Cost} = \$ \langle \text{value38} \rangle$$

Implementation Cost

The implementation cost accounts for maintenance expenses for installation.

$$C_I = (C_C) + (X_N)(T_N)(D_L)$$

$$C_I = \$ < \text{value25} > + (< \text{value3} > \text{ Nozzles}) \left(< \text{value26} > \frac{\text{hrs}}{\text{Nozzle}} \right) \left(\frac{\$ < \text{value27} >}{\text{hr}} \right) = \$$$

< value28 >

Total Implementation Cost = **\\$<value39>**

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ < \text{value28} >}{\frac{\$ < \text{value23} >}{\text{yr}}} = < \text{value29} > \text{ yr } (< \text{value30} > \text{ mo})$$

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_E)(E_{CO_2})$$

$$\left(< \text{value13} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value31} > \frac{\text{tons}}{\text{kWh}} \right) = < \text{value33} > \frac{\text{tons}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_E)(E_{\text{NOX}})$$

$$\left(< \text{value13} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value32} > \frac{\text{lbs}}{\text{kWh}} \right) = < \text{value34} > \frac{\text{lbs}}{\text{yr}}$$

APPENDIX D: MOTORS ASSESSMENT RECOMMENDATIONS

Assessment Recommendation No. 11

Replace Standard Efficiency Motors with High Efficiency Motors

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value19>	\$<value29>			<value39>
<value16> kWh	<value26> kW•mo	\$<value38>	CO2: <value43> tons NOX: <value44> lbs	> yr (<value40> mo)
<value45>%	<value46>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace existing motors with premium efficiency motors as they wear out.

Current Operations and Observations

The IAC obtained power ratings and original efficiencies for selected AC drive motors that are 10-hp or greater on the day of the assessment. The motors used throughout the facility can be replaced with premium efficiency motors that cost more, but pay for themselves through energy savings. The efficiencies for all new motors were obtained from MotorMaster International +⁴⁵, which can be downloaded, along with various other programs, free of charge from the Department of Energy’s website.

Calculations

The following example calculations will be performed for the <value3> hp motors strewn throughout the facility. The following table outlines the usage parameters, cost savings and totals for all cataloged motors.

⁴⁵ Average values of costs and efficiencies were taken from MotorMaster+ v4.0 Software, Washington Energy Office, Olympia, WA, 2003. Downloadable from: <http://www1.eere.energy.gov/industry/bestpractices/software.html#mm>.

Table 11.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
X_{hp}	No. of Motors	See Table 11.1
$P_{X_{hp}}$	Motor Horsepower	See Table 11.1
F_L	Motor Load Factor ⁴⁶	See Table 11.1
F_D	Motor Duty Factor ⁴⁶	See Table 11.1
$\eta_{X, Current}$	Current Motor Efficiency ⁴⁷	See Table 11.1
K_{hp2kW}	Horsepower to Kilowatt Conversion Factor	<value7> $\frac{kW}{hp}$
K_{M2YR}	Month to Year Conversion Factor	<value20> $\frac{mo}{yr}$

⁴⁶ Estimated by IAC Personnel.

⁴⁷ Value derived from MotorMaster International + Software.

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
$\eta_{X, \text{Proposed}}$	Proposed Motor Efficiency ⁴⁷	See Table 11.1
$D_{X \text{ hp}}$	Cost of High Efficiency Motor ⁴⁷	See Table 11.1

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{ELEC}	Avoided Cost of Electrical Energy ⁴⁸	$\frac{\$ \langle \text{value17} \rangle}{\text{kWh}}$
D_{DEM}	Avoided Cost of Electrical Demand ⁴⁸	$\frac{\$ \langle \text{value27} \rangle}{\text{kW} \cdot \text{mo}}$
D_{L}	Maintenance Labor Cost ⁴⁹	$\frac{\$ \langle \text{value36} \rangle}{\text{hr}}$
T_{O}	Operating Hours ⁴⁹	$\langle \text{value8} \rangle \text{ hrs}$
T_{I}	Installation Time ⁴⁶	$\langle \text{value35} \rangle \text{ hr}$
E_{CO2}	CO ₂ Emission Factor ⁵⁰	$\langle \text{value41} \rangle \frac{\text{tons}}{\text{kWh}}$
E_{NOX}	NO _x Emission Factor ⁶	$\langle \text{value42} \rangle \frac{\text{lbs}}{\text{kWh}}$

⁴⁸ See energy cost analysis section.

⁴⁹ See summary of plant statistics section.

⁵⁰ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Usage	See Table 11.1
E _P	Proposed Energy Usage	See Table 11.1
E _S	Energy Savings	See Table 11.1
E _{CS}	Total Energy Cost Savings	See Table 11.1
D _C	Current Demand Usage	See Table 11.1
D _P	Proposed Demand Usage	See Table 11.1
D _S	Demand Savings	See Table 11.1
D _{CS}	Demand Cost Savings	See Table 11.1
C _T	Total Cost Savings	See Table 11.1
C _C	Capital Cost	See Table 11.1
C _I	Implementation Cost	See Table 11.1
T _P	Simple Payback	<value39> yr (<value40> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value43> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value44> lbs

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy required to operate all of the <value3> hp motors is the product of the motors power requirement over the facilities operating hours through out the year.

$$E_C = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(T_O)}{\eta_{<value3>hp,current}}$$

$$\frac{(<value4>Motors)(<value3>\frac{hp}{Motor})(<value5>)(<value6>)(<value7>\frac{kW}{hp})(<value8>\frac{hrs}{yr})}{<value9>} = <$$

value10 > $\frac{kWh}{yr}$

Total Current Energy Usage = **<value11>** $\frac{kWh}{yr}$

Proposed Energy Usage

Replacing the <value3> hp motors with energy efficient counterparts will reduce the required power to day to day operations. The proposed energy usage is the reduced power over the facility operational hours throughout the year.

$$E_p = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(T_O)}{\eta_{<value3>hp,high\ efficiency}}$$

$$\frac{(<value4>Motors)(<value3>\frac{hp}{Motor})(<value5>)(<value6>)(<value7>\frac{kW}{hp})(<value8>\frac{hrs}{yr})}{<value12>} = <$$

value13 > $\frac{kWh}{yr}$

$$\text{Total Proposed Energy Usage} = <value14> \frac{kWh}{yr}$$

Energy Savings

The energy savings found are as the difference between the current energy usage and the proposed energy usage.

$$E_s = E_c - E_p$$

$$<value10> \frac{kWh}{yr} - <value13> \frac{kWh}{yr} = \text{value15} \frac{kWh}{yr}$$

$$\text{Total Energy Savings} = <value16> \frac{kWh}{yr}$$

Energy Cost Savings

The electrical cost savings can be found to be the product of the total energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{Elec})$$

$$\left(< \text{value15} > \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ < \text{value17} >}{\text{kWh}} \right) = \frac{\$ < \text{value18} >}{\text{yr}}$$

$$\text{Total Energy Cost Savings} = \frac{\$ < \text{value19} >}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy.. The demand associated with all of the <value3> hp motors is the motor power for every month of the year.

$$D_C = \frac{(X_{< \text{value3} > \text{hp}})(P_{< \text{value3} > \text{hp}})(F_L)(F_D)(K_{\text{hp}2\text{KW}})(K_{\text{M}2\text{YR}})}{\eta_{< \text{value3} > \text{hp, current}}}$$

$$\frac{(< \text{value4} > \text{Motors}) \left(< \text{value3} > \frac{\text{hp}}{\text{Motor}} \right) (< \text{value5} >) (< \text{value6} >) \left(< \text{value7} > \frac{\text{kW}}{\text{hp}} \right) \left(< \text{value20} > \frac{\text{mo}}{\text{yr}} \right)}{< \text{value12} >} = <$$

value21 > $\frac{\text{kW}\cdot\text{mo}}{\text{yr}}$

$$\text{Total Current Demand Usage} = < \text{value22} > \frac{\text{kW}\cdot\text{mo}}{\text{yr}}$$

Proposed Demand Usage

Since high efficiency motors use less power, assuming that the motors are operational during peak demand hours, the demand associated with motor operation should also decrease.

$$D_P = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(K_{M2YR})}{\eta_{<value3>hp,high\ efficiency}}$$

$$\frac{(<value4>Motors)(<value3>\frac{hp}{Motor})(<value5>)(<value6>)(<value7>\frac{kW}{hp})(<value20>\frac{mo}{yr})}{<value12>} = <$$

value23 > $\frac{kW \cdot mo}{yr}$

$$\text{Total Proposed Demand Usage} = <value24> \frac{kW \cdot mo}{yr}$$

Demand Savings

The energy savings found as the difference between the current energy usage and the proposed energy usage.

$$D_S = D_C - D_P$$

$$<value21> \frac{kW \cdot mo}{yr} - <value23> \frac{kW \cdot mo}{yr} = <value25> \frac{kW \cdot mo}{yr}$$

$$\text{Total Demand Savings} = <value26> \frac{kW \cdot mo}{yr}$$

Demand Cost Savings

The demand cost savings can be found to be the product of the total demand savings and the avoided cost of demand.

$$D_{CS} = (D_S)(D_{Dem})$$
$$\left(\langle \text{value25} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value27} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value28} \rangle}{\text{yr}}$$
$$\text{Total Demand Cost Savings} = \langle \text{value29} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the demand cost savings and the demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value18} \rangle}{\text{yr}} + \frac{\$ \langle \text{value28} \rangle}{\text{yr}} = \frac{\$ \langle \text{value30} \rangle}{\text{yr}}$$
$$\text{Aggregated Total Cost Savings} = \frac{\$ \langle \text{value31} \rangle}{\text{yr}}$$

Capital Cost

The capital cost is the increased amount of money needed to buy the new $\langle \text{value3} \rangle$ hp, high efficiency motors.

$$C_c = (X_{\langle \text{value3} \rangle \text{hp}})(D_{\langle \text{value3} \rangle \text{hp}})$$
$$(\langle \text{value4} \rangle \text{ Motors}) \left(\frac{\$ \langle \text{value32} \rangle}{\text{Motor}} \right) = \$ \langle \text{value33} \rangle$$
$$\text{Total Capital Cost} = \$ \langle \text{value34} \rangle$$

Implementation Cost

The implementation cost is the capital cost since the recommendation is to replace motors as they become defective.

$$C_I = C_C$$

$$C_I = \$ < \text{value37} >$$

$$\text{Total Implementation Cost} = \$ < \text{value38} >$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ < \text{value38} >}{\frac{\$ < \text{value31} >}{\text{yr}}} = < \text{value39} > \text{ yr } (< \text{value40} > \text{ mo.})$$

CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$

$$\left(< \text{value16} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value41} > \frac{\text{tons}}{\text{kWh}} \right) = < \text{value43} > \frac{\text{tons}}{\text{yr}}$$

NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_S)(E_{\text{NOX}})$$

$$\left(< \text{value16} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value42} > \frac{\text{lbs}}{\text{kWh}} \right) = < \text{value44} > \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 12

Replace V-belts with Notched or Synchronous Belt Drives

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Annual Demand Savings	Implementation Cost	Annual Pollution Reduction	Payback
<value17> kWh	<value27> kW	<value36>	CO ₂ : <value41> tons	<value37> yr
\$<value20>	\$<value30> kW•mo	\$<value36>	NO _x : <value42> lbs	(<value38> mo)
<value43>%	<value44>%			

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Replace V-belts with synchronous belts for all new installations, and install notched belts where the retrofit of a synchronous belt is not cost effective.

Current Operations and Observations

Instead of using friction to pull, like V-belts, notched belts have grooves perpendicular to the belt's length which reduces the bending resistance of the belt. Notched belts can use the same pulleys as V-belts and are about 2% more efficient. Synchronous belts are toothed and require mating grooved sprockets. These belts have a consistent 98% efficiency over a wide load range.

Calculations

IAC Personnel were supplied with a facility motor list. This list, amongst other motor characteristics describes which motors are outfitted with belts. From this list, the following table was composed and the appropriate savings were calculated. The following equations will exemplify the usage and savings calculations for the <value3> hp motors.

Table 12.1: Motor data with respective savings.

<Table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P	Motor Power	See Table 12.1
η_M	Motor Efficiency	See Table 12.1
X	Number of Motors	See Table 12.1
F_D	Duty Factor ⁵¹	See Table 12.1
F_L	Load Factor ⁵¹	See Table 12.1
K_{HP2kW}	Horsepower to Kilowatt Conversion Factor	<value7> $\frac{kW}{HP}$
K_{M2YR}	Month to Year Conversion Factor	<value21> $\frac{mo}{yr}$

⁵¹ Estimated by IAC Personnel

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
η_v	Efficiency Increase due to Synchronous Belt Usage	<value13>
D_v	Cost of Synchronous Belts ⁵²	$\frac{\$ < \text{value33} >}{\text{Belt}}$

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
AC_E	Avoided Cost of Electrical Energy ⁵³	$\frac{\$ < \text{value18} >}{\text{kWh}}$
AC_D	Avoided Cost of Electrical Demand ⁵³	$\frac{\$ < \text{value28} >}{\text{kW}\cdot\text{mo}}$
T_o	Operating Hours ⁴	<value8> $\frac{\text{hrs}}{\text{yr}}$
E_{CO_2}	CO ₂ Emission Factor ⁵⁴	<value39> $\frac{\text{tons}}{\text{kWh}}$
E_{NO_x}	NO _x Emission Factor ⁵⁴	<value40> $\frac{\text{lbs}}{\text{kWh}}$

⁵² Since the cost of the belt will be dependent on the size, the cost is an average based on multiple online sources.

⁵³ See Summary of Plant Statistics.

⁵⁴ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Motor Energy Consumption	See Table 12.1
E _P	Proposed Motor Energy Consumption	See Table 12.1
E _S	Energy Savings	See Table 12.1
E _{CS}	Energy Cost Savings	See Table 12.1
D _C	Current Motor Demand	See Table 12.1
D _P	Proposed Motor Demand	See Table 12.1
D _S	Demand Savings	See Table 12.1
D _{CS}	Demand Cost Savings	See Table 12.1
C _T	Total Cost Savings	See Table 12.1
C _C	Capital Cost	See Table 12.1
C _I	Implementation Cost	See Table 12.1
T _P	Simple Payback	<value37> yr (<value38> mo)
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value41> tons
G _{NOX}	Equivalent NO _x Emissions Reductions	<value42> lbs

Current Motor Energy Consumption

The following is an example calculation for the <value3> hp motors strewn throughout the facility. The current energy consumption is the power consumption of the motor, over the course of the yearly operating hours.

$$E_C = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(T_O)}{\eta_{<value3>hp,current}}$$

$$\frac{(<value4>Motors)(<value3>\frac{hp}{Motor})(<value5>)(<value6>)(<value7>\frac{kW}{hp})(<value8>\frac{hrs}{yr})}{<value9>} = <value10>$$

value10 > $\frac{kWh}{yr}$

Total Current Energy Usage = **<value11>** $\frac{kWh}{yr}$

Proposed Motor Energy Consumption

Synchronous Belts are known to increase motor efficiency by <value12>%, since their design decreases the friction inherent in rotating the belts, thus the proposed energy consumption is the current energy consumption decreased by the synchronous belt efficiency.

$$E_P = E_C + [(E_C)(\eta_V)]$$

$$<value10> \frac{kWh}{yr} - \left[\left(<value10> \frac{kWh}{yr} \right) (<value13>) \right] = <value14> \frac{kWh}{yr}$$

Total Proposed Energy Usage = **<value15>** $\frac{kWh}{yr}$

Energy Savings

The energy savings due to implementing synchronous belts retrofits on belted motors is the difference between the current motor usage and the proposed motor usage.

$$E_S = E_C - E_P$$

$$\langle \text{value10} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Energy Savings} = \langle \text{value17} \rangle \frac{\text{kWh}}{\text{yr}}$$

Energy Cost Savings

The capital energy savings due to implementing a retrofit on all motor driven belts is the product of the energy savings and the avoided cost of electrical energy.

$$C_S = (E_S)(D_{\text{Elec}})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value18} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value19} \rangle}{\text{yr}}$$

$$\text{Total Energy Cost Savings} = \frac{\$ \langle \text{value20} \rangle}{\text{yr}}$$

Current Motor Demand

The current motor demand is the amount of power consumed by motors belted with standard belts.

$$D_C = \frac{(X_{\langle \text{value3} \rangle \text{hp}})(P_{\langle \text{value3} \rangle \text{hp}})(F_L)(F_D)(K_{\text{hp2KW}})(K_{\text{M2YR}})}{\eta_{\langle \text{value3} \rangle \text{hp,current}}}$$

$$\frac{(\langle \text{value4} \rangle \text{Motors}) \left(\langle \text{value3} \rangle \frac{\text{hp}}{\text{Motor}} \right) (\langle \text{value5} \rangle) (\langle \text{value6} \rangle) \left(\langle \text{value7} \rangle \frac{\text{kW}}{\text{hp}} \right) \left(\langle \text{value21} \rangle \frac{\text{mo}}{\text{yr}} \right)}{\langle \text{value9} \rangle} = \langle \text{value22} \rangle \frac{\text{kW}\cdot\text{mo}}{\text{yr}}$$

$$\text{Total Current Demand Usage} = \langle \text{value23} \rangle \frac{\text{kW}\cdot\text{mo}}{\text{yr}}$$

Proposed Motor Demand

Since synchronous belts increase motor efficiency by 2%, the proposed demand usage is the current motor demand decreased by the synchronous belt efficiency.

$$D_P = D_C + [(D_C)(\eta_V)]$$

$$\langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \left[\left(\langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) (\langle \text{value13} \rangle) \right] = \langle \text{value24} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = \langle \text{value25} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The demand savings associated with retrofitting standard motor belts with synchronous belts is the difference between the current motor demand and the proposed motor demand.

$$D_S = D_C - D_P$$

$$\langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value24} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value26} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Demand Savings} = \langle \text{value27} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The capital demand savings due to implementing a synchronous belt retrofit is the product of the demand savings and the avoided cost of electrical demand.

$$D_{CS} = (D_S)(D_{DEM})$$

$$\left(\langle \text{value26} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value28} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value29} \rangle}{\text{yr}}$$

$$\text{Total Demand Cost Savings} = \langle \text{value30} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the energy cost savings and demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ < \text{value19} >}{\text{yr}} + \frac{\$ < \text{value29} >}{\text{yr}} = \frac{\$ < \text{value31} >}{\text{yr}}$$
$$\text{Aggregated Total Cost Savings} = \frac{\$ < \text{value32} >}{\text{yr}}$$

Capital Cost

The capital cost is the cost associated with purchasing the synchronous belts. This depends on a variety of factors, including belts size and motor rating. Since there are a variety of motors, each with their own rated synchronous belt, an average price was chosen and based on several online sources.

$$C_C = (D_V)(X_{< \text{value3} > \text{hp}})$$
$$\left(\frac{\$ < \text{value33} >}{\text{Belt}} \right) (< \text{value4} > \text{ Belts}) = \$ < \text{value35} >$$
$$\text{Total Capital Cost} = \$ < \text{value36} >$$

Implementation Cost

This recommendation is to retrofit the current standing belts with synchronous belts as they wear out. Since the belts have to be replaced anyways, there is no personnel cost associated with this retrofit. The implementation cost is thus the capital cost for purchasing the synchronous belts.

$$IC = C_C$$
$$IC = \$ < \text{value35} >$$
$$\text{Total Implementation Cost} = \$ < \text{value36} >$$

Simple Payback

The simple payback is the quotient between the capital of what needs to be spent to implement the recommendation, and the savings associated with the recommendation.

$$T_P = \frac{C_I}{C_T}$$

$$\frac{\$ \langle \text{value36} \rangle}{\frac{\$ \langle \text{value32} \rangle}{\text{yr}}} = \langle \text{value37} \rangle \text{ yr } (\langle \text{value38} \rangle \text{ mo})$$

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value39} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value41} \rangle \frac{\text{tons}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{NO_x} = (E_S)(E_{NO_x})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value40} \rangle \frac{\text{lbs}}{\text{kWh}} \right) = \langle \text{value42} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 13

Utilize Synthetic Motor Lubricants

(Prepared by <value1>)

A.R.C.: <value2>

Annual Resource Savings	Annual Demand Savings	Annual Cost Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$<value20>	\$<value30>	\$<value32>	\$0	CO ₂ : <value35> tons NO _x : <value36> lbs	Immediate
<value17> kWh <value38>%	< value27 > kW · mo <value39>%				

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Use synthetic lubricants on electric motors rather than conventional lubricants to reduce energy consumption and electrical costs.

Current Operations and Observations

On the day of the plant visit, IAC personnel observed that the current motors are using conventional lubricants, as opposed to synthetic lubricants. Utilizing synthetic lubricants reduces the motors frictional losses by 10%. The only candidates for synthetic lubricants are those motors with zerk fittings because the presence of a zerk fitting indicates non-sealed bearings.

Calculations

The following example calculations have been done for the <value3> hp motors. Usage and savings calculations for all motors found throughout the facility are exemplified in the following table

Table 13.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
P	Motor Horsepower ⁵⁵	See Table 13.1
X	No. of Motors	See Table 13.1
η	Efficiency of Motors ⁵⁶	See Table 13.1
F _D	Duty Factor	See Table 13.1
F _L	Load Factor	See Table 13.1
K _{hp2kW}	Horsepower to Kilowatt Conversion Factor	<value8> $\frac{\text{kW}}{\text{hp}}$
K _{M2YR}	Month to Year Conversion Factor	<value21> $\frac{\text{mo}}{\text{yr}}$

⁵⁵Observed by IAC personnel.

⁵⁶Values were derived using Motormaster + International.

Proposed Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
R	Percentage of Energy Savings Using Lubricants ⁵⁷	<value37>% (<value10>)

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D _{ELEC}	Avoided Cost of Electrical Energy ⁵⁸	$\frac{\$ < \text{value18} >}{\text{kWh}}$
D _{DEM}	Avoided Cost of Electrical Demand ⁵³	$\frac{\$ < \text{value28} >}{\text{kW} \cdot \text{mo}}$
T _O	Operating Hours ⁵³	<value9> $\frac{\text{hrs}}{\text{yr}}$
E _{CO2}	CO ₂ Emission Factor ⁵³	<value33> $\frac{\text{tons}}{\text{kWh}}$
E _{NOX}	NO _x Emission Factor ⁵³	<value34> $\frac{\text{lbs}}{\text{kWh}}$

⁵⁷Estimated by IAC.

⁵⁸ See Summary of Plant Statistics.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _C	Current Energy Consumption	See Table 13.1
E _P	Proposed Energy Consumption	See Table 13.1
E _S	Electrical Savings	See Table 13.1
E _{CS}	Electrical Cost Savings	See Table 13.1
D _C	Current Demand	See Table 13.1
D _P	Proposed Demand	See Table 13.1
D _S	Demand Savings	See Table 13.1
D _{CS}	Demand Cost Savings	See Table 13.1
C _T	Total Cost Savings	See Table 13.1
I	Implementation	$\frac{\$0}{\text{yr}}$
T _P	Simple Payback	Immediate
G _{CO2}	Equivalent CO ₂ Emissions Reductions	<value35> $\frac{\text{tons}}{\text{yr}}$
G _{NOX}	Equivalent NO _x Emissions Reductions	<value36> $\frac{\text{lbs}}{\text{yr}}$

Current Energy Usage

Energy usage refers to the total amount of energy used; this is measured in kWh (kilowatt hours). The energy required to operate all of the <value3> hp motors is the product of the motors power requirement over the facilities operating hours through out the year.

$$E_C = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(T_O)}{\eta_{<value3>hp,current}}$$

$$\frac{(<value4>Motors)(<value5>\frac{hp}{Motor})(<value6>)(<value7>)(<value8>\frac{kW}{hp})(<value9>\frac{hrs}{yr})}{<value10>} = <$$

value11 > $\frac{kWh}{yr}$

Total Current Energy Usage = < **value12** > $\frac{kWh}{yr}$

Proposed Energy Usage

Since synthetic lubricants reduce friction to a greater degree in comparison to traditional lubricants, using then has the potential to reduce motor friction energy consumption by 10%. The reduction in friction is exemplified as an increase in motor efficiency

$$E_P = \frac{(X_{<value3>hp})(P_{<value3>hp})(F_L)(F_D)(K_{hp2KW})(T_O)}{\eta_{<value3>hp,current} + [(1 - \eta_{<value3>hp,current})(R)]}$$

$$\frac{(<value4>Motors)(<value5>\frac{hp}{Motor})(<value6>)(<value7>)(<value8>\frac{kW}{hp})(<value9>\frac{hrs}{yr})}{<value10> + [(1 - <value10>)(<value13>)]} = <$$

value14 > $\frac{kWh}{yr}$

Total Proposed Energy Usage = < **value15** > $\frac{kWh}{yr}$

Electrical Energy Savings

The energy savings are calculated as the difference between the current and proposed energy usage.

$$E_S = E_C - E_P$$

$$\langle \text{value11} \rangle \frac{\text{kWh}}{\text{yr}} - \langle \text{value14} \rangle \frac{\text{kWh}}{\text{yr}} = \langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}}$$

$$\text{Total Electrical Energy Savings} = \langle \text{value17} \rangle \frac{\text{kWh}}{\text{yr}}$$

Electrical Cost Savings

The electrical cost savings is the product of the energy savings and the avoided cost of electrical energy.

$$E_{CS} = (E_S)(D_{ELEC})$$

$$\left(\langle \text{value16} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value18} \rangle}{\text{kWh}} \right) = \frac{\$ \langle \text{value19} \rangle}{\text{yr}}$$

$$\text{Total Electrical Energy Savings} = \frac{\$ \langle \text{value20} \rangle}{\text{yr}}$$

Current Demand Usage

Demand Usage refers to the rate at which a facility uses energy. The demand associated with all of the <value3> hp motors is the motor power for every month of the year.

$$D_C = \frac{(X_{\langle \text{value3} \rangle \text{hp}})(P_{\langle \text{value3} \rangle \text{hp}})(F_L)(F_D)(K_{\text{hp}2\text{KW}})(K_{\text{M}2\text{YR}})}{\eta_{\langle \text{value3} \rangle \text{hp, current}}}$$

$$\frac{(\langle \text{value4} \rangle \text{Motors}) \left(\langle \text{value3} \rangle \frac{\text{hp}}{\text{Motor}} \right) (\langle \text{value5} \rangle) (\langle \text{value6} \rangle) \left(\langle \text{value7} \rangle \frac{\text{kW}}{\text{hp}} \right) \left(\langle \text{value21} \rangle \frac{\text{mo}}{\text{yr}} \right)}{\langle \text{value12} \rangle} = \langle \text{value22} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Current Demand Usage} = \langle \text{value23} \rangle \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Proposed Demand Usage

Since high efficiency motors use less power, assuming that the motors are operational during peak demand hours, the demand associated with motor operation should also decrease.

$$D_P = \frac{(X_{\langle \text{value3} \rangle \text{hp}})(P_{\langle \text{value3} \rangle \text{hp}})(F_L)(F_D)(K_{\text{hp2KW}})(K_{M2YR})}{\eta_{\langle \text{value3} \rangle \text{hp,current}} + [(1 - \eta_{\langle \text{value3} \rangle \text{hp,current}})(R)]}$$

$$\frac{(\langle \text{value4} \rangle \text{Motors})(\langle \text{value5} \rangle \frac{\text{hp}}{\text{Motor}})(\langle \text{value6} \rangle)(\langle \text{value7} \rangle)(\langle \text{value8} \rangle \frac{\text{kW}}{\text{hp}})(\langle \text{value21} \rangle \frac{\text{mo}}{\text{yr}})}{\langle \text{value10} \rangle + [(1 - \langle \text{value10} \rangle)(\langle \text{value13} \rangle)]} = \langle \text{value24} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Proposed Demand Usage} = \langle \text{value25} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Savings

The demand savings are calculated as the difference between the current and proposed demand usage.

$$D_S = D_C - D_P$$

$$\langle \text{value22} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} - \langle \text{value24} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} = \langle \text{value26} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

$$\text{Total Demand Savings} = \langle \text{value27} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}}$$

Demand Cost Savings

The demand cost savings is the product of the energy savings and the avoided cost of electrical demand.

$$D_{CS} = (D_S)(D_{DEM})$$

$$\left(\langle \text{value26} \rangle > \frac{\text{kW} \cdot \text{mo}}{\text{yr}} \right) \left(\frac{\$ \langle \text{value28} \rangle}{\text{kW} \cdot \text{mo}} \right) = \frac{\$ \langle \text{value29} \rangle}{\text{yr}}$$

$$\text{Total Demand Cost Savings} = \frac{\$ \langle \text{value30} \rangle}{\text{yr}}$$

Total Cost Savings

The total cost savings is the sum of the savings realized through electrical cost savings and demand cost savings.

$$C_T = E_{CS} + D_{CS}$$
$$\frac{\$ \langle \text{value19} \rangle}{\text{yr}} + \frac{\$ \langle \text{value29} \rangle}{\text{yr}} = \frac{\$ \langle \text{value31} \rangle}{\text{yr}}$$
$$\text{Total Aggregated Cost Savings} = \frac{\$ \langle \text{value32} \rangle}{\text{yr}}$$

Implementation Cost

The higher cost of synthetic lubricants is offset by a longer life for the lubricant, so there are no additional implementation costs.

Simple Payback

Since there is no implementation cost, the simple payback is immediate.

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (E_S)(E_{CO_2})$$
$$\left(\langle \text{value17} \rangle \frac{\text{kWh}}{\text{yr}} \right) \left(\langle \text{value33} \rangle \frac{\text{tons}}{\text{kWh}} \right) = \langle \text{value35} \rangle \frac{\text{ton}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NOX}} = (E_S)(E_{\text{NOX}})$$
$$\left(< \text{value17} > \frac{\text{kWh}}{\text{yr}} \right) \left(< \text{value34} > \frac{\text{lbs}}{\text{kWh}} \right) = < \text{value36} > \frac{\text{lbs}}{\text{yr}}$$

Caution: Prior to using synthetic lubricants, you should contact a lubrication specialist or the original equipment vendor to determine the suitability and compatibility of the proposed lubricant for the equipment. Some equipment may have seals that are incompatible with some types of synthetic lubricants.

APPENDIX E: POWER FACTOR CORRECTION ASSESSMENT
RECOMMENDATIONS

Assessment Recommendation No. 14

Install Capacitor Bank for Power Factor Correction

(Prepared by <value1>)

A.R.C.: <value2>

Annual Demand Savings	Implementation Cost	Payback
\$<value12>		<value25>
<value9> kW	\$<value24>	> yr (<value26> mo)
<value27>%		6> mo)

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Install a capacitor bank to raise the average power factor for <value3> to <value4>.

Current Operations and Observations

An analysis of electrical utility bills indicates an average power factor of <value3>. Power factor is recorded by the utility to measure the amount of usable power being consumed. When the power factor is low, this indicates that a plant uses a diminished portion of the supplied power towards work. The usable or real power in the facility is measured in kilowatts (kW). This power goes towards performing work such as turning motors, compressing air and turning on lights. The reactive power, measured in kilovolt-ampere reactive (kVAR) is static and does not perform work. It is used to energize the magnetic field in motor coils or lighting ballasts. The power supplied by the utility, called apparent power is measured in kilovolt-amperes (kVA). The power factor is the ratio of real power to apparent power and is equal to the cosine of the angle in the power

triangle The power factor indicates how much of the power the facility is using in proportion to the amount that the utility is actually supplying.

The power factor is the ratio of the real power (the power going towards work) and the apparent power (the power supplied by the utility).

$$\text{Power Factor} = \frac{\text{Real Power (kW)}}{\text{Apparent Power (kVA)}} = \frac{\text{kW}}{\sqrt{(\text{kW})^2 + (\text{kVAR})^2}}$$
$$= \cos(\text{Power Factor Angle})$$

The electrical utility requires that a facility retain a power factor of <value4> or above. A facility incurs a penalty when the power factor is below this requirement. The penalty represents a billed demand that is larger than the facility actual demand usage and is calculated as:

$$D_B = \frac{(D_A)(F_P)}{(F_C)}$$

where D_B is the billed demand incurred by power factor, D_A is the actual demand, F_P is the proposed power factor and F_C is the current power factor.

Calculations

The following example calculations characterize power factor correction variables for <value5>. The following tables outline demand usage and power factor variables for the facility outlined on the utility bills.

Table 14.1: Power Factor relevant variables.

<table1>

Capacitor banks are installed to mitigate and reduce reactive power. By reducing the reactive power, the apparent power comes closer to aligning with the real power, thus raising the power factor. The following table outlines the reactive power required to raise the power factor to <value4> and the capacitance necessary.

Table 14.2: Reactive Power Variables.

<table2>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D _B	Billed Demand	Table 14.1
F _C	Current Power Factor	Table 14.1

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
F_P	Proposed Power Factor	<value4>
C_{Fix}	Fixed Capacitance Cost ⁵⁹	$\frac{\$<value22>}{kVAR}$
C_{Var}	Variable Capacitance Cost ⁵⁹	$\frac{\$<value23>}{kVAR}$
V_{Max}	Maximum Proposed Capacitance	<value19> kVAR
V_{Min}	Minimum Proposed Capacitance	<value20> kVAR

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_{Dem}	Avoided Cost of Electrical Demand ⁶⁰	$\frac{\$<value10>}{kW}$

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D_A	Actual Demand	Table 14.1
D_S	Demand Savings	Table 14.1
D_{CS}	Demand Cost Savings	Table 14.1
$P_{R,Current}$	Current Reactive Power	Table 14.2
$P_{R,Prop}$	Proposed Reactive Power	Table 14.2
V_C	Proposed Capacitance	Table 14.2
V_{Fix}	Fixed Capacitance	<value17> kVAR
V_{Var}	Variable Capacitance	<value21> kVAR
C_C	Implementation Cost	$\$<value24>$
T_P	Simple Payback	<value25> yr (<value26> mo)

⁵⁹ Cost is an average from several sources. Contact IAC for details.

⁶⁰ See Energy Cost Analysis section.

Actual Demand

The electric utility increases the billed demand when the power factor is below <value4>. This billed demand is provided on the electric bills. The actual demand is calculated by rearranging the previous formula for computing the billed demand.

$$D_A = \left[\frac{(D_B)(F_C)}{(F_P)} \right]$$
$$\left[\frac{(< \text{value6} > \text{ kW})(< \text{value7} >)}{(< \text{value4} >)} \right] = < \text{value28} > \text{ kW}$$

Demand Savings

The savings associated with correcting power factor is the difference of the actual facility demand and the power factor incurred billed demand.

$$D_S = D_B - D_A$$
$$(< \text{value6} > \text{ kW}) - (< \text{value28} > \text{ kW}) = < \text{value8} > \text{ kW}$$
$$\text{Total Demand Savings} = < \text{value9} > \text{ kW}$$

Demand Cost Savings

The cost associated with the larger billed demand is the incurred charge for electric demand. By installing a capacitor bank, these charges are mitigated.

$$D_{CS} = (D_S)(D_{Dem})$$
$$(< \text{value8} > \text{ kW}) \left(\frac{\$ < \text{value10} >}{\text{kW}} \right) = \$ < \text{value11} >$$
$$\text{Total Demand Cost Savings} = \frac{\$ < \text{value12} >}{\text{yr}}$$

Current Reactive Power

Using standard trigonometric relationships for the triangle depicted in **Error! Reference source not found.**1, it is possible to calculate the current reactive power using the actual facility demand and the current monthly power factor.

$$P_{R,Current} = \sqrt{\frac{(D_A)^2}{(F_C)^2} - (D_A)^2}$$
$$\sqrt{\frac{(< value28 > \text{ kW})^2}{(< value7 >)^2} - (< value28 > \text{ kW})^2} = < \text{value13} > \text{ kVAr}$$

Proposed Reactive Power

Two factors affect power factor, real power and the reactive power. By reducing reactive power one is able to increase power factor. A capacitor bank is used to mitigate reactive power. The following calculates the necessary reactive power to reflect a power factor of <value4>.

$$P_{R,Proposed} = \sqrt{\frac{(D_A)^2}{(F_P)^2} - (D_A)^2}$$
$$\sqrt{\frac{(< value28 > \text{ kW})^2}{(< value4 >)^2} - (< value28 > \text{ kW})^2} = < \text{value14} > \text{ kVAr}$$

Proposed Capacitance

The proposed capacitance is the difference between the current and proposed reactive power. The proposed capacitance for subsequent months can be seen in Table 14.2.

$$V_C = P_{R,Current} - P_{R,Proposed}$$
$$< \text{value13} > \text{ kVAr} - < \text{value14} > \text{ kVAr} = < \text{value15} > \text{ kVAr}$$

Fixed Capacitance

A capacitor bank is outfitted with two types of capacitance; fixed and variable. Fixed capacitance is the minimum capacitance required to raise the power factor to <value4>. From Table 14.2 this was observed on <value16>.

$$V_{\text{Fix}} = \text{< value17 > kVAR}$$

Variable Capacitance

Variable Capacitance accounts for how much the capacitance has to change from month to month to guarantee a power factor of <value4>. It is the difference between the minimum and maximum capacitance. From Table 14.2 the maximum capacitance was observed on <value18>.

$$V_{\text{Var}} = V_{\text{Max}} - V_{\text{Min}}$$

$$\text{< value19 > kVAR} - \text{< value20 > kVAR} = \text{< value21 > kVAR}$$

Implementation Cost

The implementation cost is the total cost for fixed and variable capacitance

$$C_C = (V_{\text{Fix}})(C_{\text{Fix}}) + (V_{\text{Var}})(C_{\text{Var}})$$

$$\begin{aligned} (\text{< value17 > kVAR}) \left(\frac{\$ \text{< value22 >}}{\text{kVAR}} \right) + (\text{< value21 > kVAR}) \left(\frac{\$ \text{< value23 >}}{\text{kVAR}} \right) = \$ \\ \text{< value24 >} \end{aligned}$$

Simple Payback

The simple payback is the quotient of the capital cost and the annual cost savings.

$$T_P = \frac{C_C}{D_{CS}}$$

$$\frac{\$ \text{< value24 >}}{\frac{\$ \text{< value12 >}}{\text{yr}}} = \text{< value25 > yr (< value26 > mo)}$$

APPENDIX F: BOILER ASSESSMENT RECOMMENDATIONS

Assessment Recommendation No. 15

Perform a Boiler Tune Up to Improve Boiler Efficiency (Known NG Usage)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$ < value12 >			<value14 > yr
< value10 > < value21 > <value20>%	\$<value13>	CO ₂ : <value18> tons NO _x : <value19> lbs	(<value15> mo)

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Perform a tune-up on the boiler to improve efficiency.

Current Operations and Observations

The plant uses one boiler. Using a combustion analyzer, IAC personnel determined the oxygen level in the exhaust gas is <value3>% when the ideal level is 2 to 4%.⁶¹ The analyzer also indicated an efficiency of <value4>%. By reducing the oxygen content to between 2-4%, a boiler tune up could increase efficiency to <value5>%.⁶¹ The higher boiler efficiency would result in natural gas savings and decreased emission of CO₂ and NO_x.

⁶¹ Turner, Wayne, C., Energy Management Handbook, p. 91.

Calculations

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
G _C	Current Natural Gas Consumption	<value6> $\frac{\text{<value21>}}{\text{yr}}$
E _C	Current Boiler Efficiency ⁶²	<value4>

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
E _P	Proposed Boiler Efficiency ⁶³	<value5> %
I _T	Estimated Boiler Tune-up Cost ⁶⁴	\$<value13>

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
D _G	Avoided Cost of Natural Gas ⁶⁵	$\frac{\text{\$<value11>}}{\text{<value21>}}$
E _{CO2}	CO2 Emission Factor ⁶⁶	<value16> $\frac{\text{tons}}{\text{<value21>}}$
E _{NOX}	NOX Emission Factor ⁶⁶	<value17> $\frac{\text{lbs}}{\text{<value21>}}$

⁶² Measured with a combustion analyzer by IAC personnel.

⁶³ Turner, Wayne, C., *Energy Management Handbook*, p. 91.

⁶⁴ Estimated by IAC personnel.

⁶⁵ See energy cost analysis section.

⁶⁶ See energy management and emission reduction section.

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
G _P	Proposed Natural Gas Consumption	$\frac{\langle \text{value9} \rangle}{\langle \text{value21} \rangle \text{ yr}}$
G _S	Natural Gas Savings	$\frac{\langle \text{value10} \rangle}{\langle \text{value21} \rangle \text{ yr}}$
S _G	Cost Savings	$\frac{\$ \langle \text{value12} \rangle}{\text{yr}}$
C _I	Implementation Cost	$\$ \langle \text{value13} \rangle$
T _P	Simple Payback	$\langle \text{value14} \rangle \text{ yr}$ $(\langle \text{value15} \rangle \text{ mo})$
G _{CO2}	Equivalent CO ₂ Emissions Reductions	$\langle \text{value18} \rangle \frac{\text{tons}}{\text{yr}}$
G _{NOX}	Equivalent NO _x Emissions Reductions	$\langle \text{value19} \rangle \frac{\text{lbs}}{\text{yr}}$

Proposed Natural Gas Use

The proposed natural gas use is a function of the current natural gas use of the boiler, the current boiler efficiency, and the proposed boiler efficiency. It is found by multiplying the current usage by the current efficiency, all divided by the proposed efficiency.

$$G_P = \frac{(G_C)(E_C)}{E_P}$$

$$\frac{(\langle \text{value6} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}}) (\langle \text{value7} \rangle)}{(\langle \text{value8} \rangle)} = \langle \text{value9} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}}$$

Natural Gas Savings

The savings from natural gas are found by the difference of the current and proposed natural gas use.

$$G_S = G_C - G_P$$
$$\langle \text{value6} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} - \langle \text{value9} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} = \langle \text{value10} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}}$$

Cost Savings

The cost savings are found by multiplying the natural gas savings by the avoided cost of natural gas.

$$S_G = (G_S)(D_G)$$
$$\left(\langle \text{value10} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\frac{\$ \langle \text{value11} \rangle}{\langle \text{value21} \rangle} \right) = \frac{\$ \langle \text{value12} \rangle}{\text{yr}}$$

Implementation Cost

The implementation cost is the cost required for a boiler tune-up. A conservative cost estimate including parts and labor for the whole process is \$500.

$$C_I = \$ \langle \text{value13} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{S_G}$$
$$\frac{\$ \langle \text{value13} \rangle}{\frac{\$ \langle \text{value12} \rangle}{\text{yr}}} = \langle \text{value14} \rangle \text{ yr } (\langle \text{value15} \rangle \text{ mo})$$

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{\text{CO}_2} = (G_S)(E_{\text{CO}_2})$$
$$\left(\langle \text{value10} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\langle \text{value16} \rangle \frac{\text{tons}}{\langle \text{value21} \rangle} \right) = \langle \text{value18} \rangle \frac{\text{tons}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{\text{NO}_x} = (G_S)(E_{\text{NO}_x})$$
$$\left(\langle \text{value10} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\langle \text{value17} \rangle \frac{\text{lbs}}{\langle \text{value21} \rangle} \right) = \langle \text{value19} \rangle \frac{\text{lbs}}{\text{yr}}$$

Assessment Recommendation No. 16

Perform a Boiler Tune Up to Improve Boiler Efficiency (Unknown NG Usage)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Consumption Savings	Implementation Cost	Annual Pollution Reduction	Payback
\$ < value20 >			<value23
< value17 > < value21 > <value29>%	\$<value22>	CO ₂ : <value27> tons NO _x : <value28> lbs	> yr (<value24> mo)

Note: Percentages reflect resource savings from current total usage.

Recommended Action

Perform a tune-up on the boiler to improve efficiency.

Current Operations and Observations

The plant uses <value3> boiler. Using a combustion analyzer, IAC personnel determined an average boiler efficiency of of <value4>%. By reducing the oxygen content to between 2-4%, a boiler tune up could increase efficiency to <value5>%.⁶¹ The higher boiler efficiency would result in natural gas savings and decreased emission of CO₂ and NO_x

Calculations

The following example calculations have been done the <value30> Boiler. Consumption and usage calculations for other boilers have been outlined in the following table

Table 16.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
G_{Boiler}	Current Natural Gas Consumption	Table 16.1
F_L	Load Factor	Table 16.1
F_D	Duty Factor	Table 16.1
T_o	Boiler Operating time	Table 16.1
η_c	Current Boiler Efficiency ⁶⁷	Table 16.1

⁶⁷ Measured with a combustion analyzer by IAC personnel.

Proposed Operation

<u>Variable</u>	<u>Description</u>	<u>Value</u>
η_P	Proposed Boiler Efficiency ⁶⁸	<value5> %

Plant Information

<u>Variable</u>	<u>Description</u>	<u>Value</u>
C_G	Avoided Cost of Natural Gas ⁶⁹	$\frac{\$<value11>}{<value21>}$
E_{CO_2}	CO ₂ Emission Factor ⁷⁰	<value16> $\frac{\text{tons}}{<value21>}$
E_{NOX}	NO _x Emission Factor ⁶⁶	<value17> $\frac{\text{lbs}}{<value21>}$

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
G_P	Proposed Natural Gas Consumption	Table 16.1
G_S	Natural Gas Savings	Table 16.1
S_G	Cost Savings	Table 16.1
C_I	Implementation Cost	Table 16.1
T_P	Simple Payback	<value23> yr (<value24> mo)
G_{CO_2}	Equivalent CO ₂ Emissions Reductions	< value27 > $\frac{\text{tons}}{\text{yr}}$
G_{NOX}	Equivalent NO _x Emissions Reductions	< value28 > $\frac{\text{lbs}}{\text{yr}}$

⁶⁸ Turner, Wayne, C., *Energy Management Handbook*, p. 91.

⁶⁹ See energy cost analysis section.

⁷⁰ See energy management and emission reduction section.

Current Natural Gas Usage

The current natural gas use is a function of the rated natural gas usage of the boiler and the current boiler efficiency.

$$G_C = \frac{(G_{Boiler})(F_L)(F_D)(T_0)}{\eta_C}$$

$$\frac{(< \text{value6} > \frac{< \text{value21} >}{\text{yr}}) (< \text{value7} >) (< \text{value8} >) (< \text{value9} > \frac{\text{hrs}}{\text{yr}})}{(< \text{value10} >)} = < \text{value11} >$$

$$> \frac{< \text{value21} >}{\text{yr}}$$

$$\text{Total Current Natural Gas Usage} = < \text{value12} > \frac{< \text{value21} >}{\text{yr}}$$

Proposed Natural Gas Usage

The proposed natural gas use is a function of the current natural gas use of the boiler, the current boiler efficiency, and the proposed boiler efficiency. It is found by multiplying the current usage by the current efficiency, all divided by the proposed efficiency.

$$G_P = \frac{(G_{Boiler})(F_L)(F_D)(T_0)}{\eta_P}$$

$$\frac{(< \text{value6} > \frac{< \text{value21} >}{\text{yr}}) (< \text{value7} >) (< \text{value8} >) (< \text{value9} > \frac{\text{hrs}}{\text{yr}})}{(< \text{value13} >)} = < \text{value14} >$$

$$> \frac{< \text{value21} >}{\text{yr}}$$

$$\text{Total Proposed Natural Gas Usage} = < \text{value15} > \frac{< \text{value21} >}{\text{yr}}$$

Natural Gas Savings

The savings from natural gas are found by the difference of the current and proposed natural gas use.

$$G_S = G_C - G_P$$
$$\langle \text{value11} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} - \langle \text{value14} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} = \langle \text{value16} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}}$$
$$\text{Total Natural Gas Savings} = \langle \text{value17} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}}$$

Cost Savings

The cost savings are found by multiplying the natural gas savings by the avoided cost of natural gas.

$$C_S = (G_S)(C_G)$$
$$\left(\langle \text{value16} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\frac{\$ \langle \text{value18} \rangle}{\langle \text{value21} \rangle} \right) = \frac{\$ \langle \text{value19} \rangle}{\text{yr}}$$
$$\text{Total Cost Savings} = \frac{\$ \langle \text{value20} \rangle}{\text{yr}}$$

Implementation Cost

The implementation cost is the cost required for a boiler tune-up.

$$C_I = \$ \langle \text{value31} \rangle$$
$$\text{Total Implementation Cost} = \$ \langle \text{value22} \rangle$$

Simple Payback

The amount of time it takes one to recover an initial investment is the quotient of the implementation cost and the cost savings.

$$T_P = \frac{C_I}{C_S}$$

$$\frac{\$ \langle \text{value22} \rangle}{\frac{\$ \langle \text{value17} \rangle}{\text{yr}}} = \langle \text{value23} \rangle \text{ yr } (\langle \text{value24} \rangle \text{ mo})$$

Equivalent CO₂ Emissions Reductions

The amount of CO₂ reduced is the product of the energy saved and the CO₂ reduction factor.

$$G_{CO_2} = (G_S)(E_{CO_2})$$

$$\left(\langle \text{value17} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\langle \text{value25} \rangle \frac{\text{tons}}{\langle \text{value21} \rangle} \right) = \langle \text{value27} \rangle \frac{\text{tons}}{\text{yr}}$$

Equivalent NO_x Emissions Reductions

The amount of NO_x reduced is the product of the energy saved and the NO_x reduction factor.

$$G_{NO_x} = (G_S)(E_{NO_x})$$

$$\left(\langle \text{value17} \rangle \frac{\langle \text{value21} \rangle}{\text{yr}} \right) \left(\langle \text{value26} \rangle \frac{\text{lbs}}{\langle \text{value21} \rangle} \right) = \langle \text{value28} \rangle \frac{\text{lbs}}{\text{yr}}$$

APPENDIX G: LATE FEES ASSESSMENT RECOMMENDATIONS

Assessment Recommendation No. 17
Avoid Late Fee Penalties (Single Utility)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Savings	Implementation Cost	Payback
\$<value7>	\$0	Immediate

Recommended Action

Pay utility bills on time to avoid late fee penalty.

Current Operations and Observations

An examination of the <value8> utility bills by IAC personnel revealed an assessment of late fee penalties. <value3> assesses a penalty of <value4> if a bill is more than <value5> days late. This effectively result in borrowing money from the utility provider at a high interest rate.

Calculations

The following table outlines the late fee penalties assessed for the following billing periods

Table 17.1

<table1>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
R	Late Fee Percentage	<value4>
D	Bill Payment Grace Period	<value5> days

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
C _T	Total Cost Savings	Table 17.1
I	Effective Interest Rate	<value6>%
C _I	Implementation Cost	\$0
T _P	Simple Payback	Immediate

Total Cost Savings

The total cost savings associated with this assessment recommendation is the late fee penalty assessed for the months described in Table 17.1.

$$C_T = \$ < \text{value7} >$$

Evaluation of Effective Interest Rate

When a late fee penalty is implemented, a fee of <value4> of the bill charge is assessed <value4> after the bills due date. This penalty results in a high effective annual percentage rate for the borrowed amount.

$$I = \left(\left[\left[1 + \left(\frac{R}{100} \right) \right]^{30-D} \right] - 1 \right) (100)$$

$$\left(\left[\left[1 + \left(\frac{< \text{value4} >}{100} \right) \right]^{30-< \text{value5} >} \right] - 1 \right) (100) = < \text{value6} > \%$$

Implementation

Since there is no cost associated with this assessment recommendation, the implementation is null.

$$C_I = \$0$$

Payback

Since there is no implementation cost, the payback is immediate.

$$T_p = \text{Immediate}$$

Assessment Recommendation No. 18 Avoid Late Fee Penalties (Multiple Utility)

(Prepared by <value1>)

A.R.C.: <value2>

Annual Savings	Implementation Cost	Payback
\$<value8>	\$0	Immediate

Recommended Action

Pay utility bills on time to avoid late fee penalty.

Current Operations and Observations

An examination of the utility bills by IAC personnel revealed an assessment of late fee penalties. Utilities usually assess a rate percentage penalty after a certain grace period. This effectively result in borrowing money from the utility provider at a high interest rate.

Calculations

The following sample calculations have been done for late fees assessed from the <value3> utility bills. The following table outlines the late fee penalties assessed for the following billing periods for all utility types.

Table 18.1

<table1>

Table 18.2 outlines the penalty specifications for each utility.

Table 18.2

<table2>

Current Operations

<u>Variable</u>	<u>Description</u>	<u>Value</u>
R	Late Fee Percentage	Table 18.2
D	Bill Payment Grace Period	Table 18.2

Calculated Variables

<u>Variable</u>	<u>Description</u>	<u>Value</u>
C _T	Total Cost Savings	Table 18.1
I	Effective Interest Rate	Table 18.2
C _i	Implementation Cost	\$0
T _p	Simple Payback	Immediate

Total Cost Savings

The total cost savings associated with this assessment recommendation is the late fee penalty assessed for the months described in Table 18.1 for the <value3> utility bills.

$$C_T = \$ < \text{value4} >$$

Evaluation of Effective Interest Rate

When a late fee penalty is implemented, a fee of <value5> of the bill charge is assessed <value6> after the bills due date. This penalty results in a high effective annual percentage rate for the borrowed amount.

$$I = \left(\left[\left[1 + \left(\frac{R}{100} \right) \right]^{\frac{360}{30-D}} \right] - 1 \right) (100)$$

$$\left(\left[\left[1 + \left(\frac{< \text{value5} >}{100} \right) \right]^{\frac{360}{30-< \text{value6} >}} \right] - 1 \right) (100) = < \text{value7} > \%$$

Implementation

Since there is no cost associated with this assessment recommendation, the implementation is null.

$$C_I = \$0$$

Payback

Since there is no implementation cost, the payback is immediate.

$$T_P = \text{Immediate}$$

APPENDIX H: MATLAB TOOLS CODE

Yearly Demand Visualization Tool

```
function varargout = Demand_Visualization_Yearly(varargin)
% DEMAND_VISUALIZATION_YEARLY MATLAB code for
Demand_Visualization_Yearly.fig
%     DEMAND_VISUALIZATION_YEARLY, by itself, creates a new
DEMAND_VISUALIZATION_YEARLY or raises the existing
%     singleton*.
%
%     H = DEMAND_VISUALIZATION_YEARLY returns the handle to a new
DEMAND_VISUALIZATION_YEARLY or the handle to
%     the existing singleton*.
%
%
DEMAND_VISUALIZATION_YEARLY('CALLBACK', hObject, eventData, handles,...)
calls the local
%     function named CALLBACK in DEMAND_VISUALIZATION_YEARLY.M with
the given input arguments.
%
%     DEMAND_VISUALIZATION_YEARLY('Property','Value',...) creates a
new DEMAND_VISUALIZATION_YEARLY or raises the
%     existing singleton*. Starting from the left, property value
pairs are
%     applied to the GUI before Demand_Visualization_Yearly_OpeningFcn
gets called. An
%     unrecognized property name or invalid value makes property
application
%     stop. All inputs are passed to
Demand_Visualization_Yearly_OpeningFcn via varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Demand_Visualization_Yearly

% Last Modified by GUIDE v2.5 18-Jun-2014 11:10:29

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn',  @Demand_Visualization_Yearly_OpeningFcn, ...
                  'gui_OutputFcn',  @Demand_Visualization_Yearly_OutputFcn, ...
```

```

        'gui_LayoutFcn', [] , ...
        'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Demand_Visualization_Yearly is made visible.
function Demand_Visualization_Yearly_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Demand_Visualization_Yearly (see
VARARGIN)

% Choose default command line output for Demand_Visualization_Yearly
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Demand_Visualization_Yearly wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;
guidata(hObject, handles);

%Set Intital Years
Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;

%Setting Initial Consumption and Demand Costs and Alpha
var1 = get(handles.edit1, 'String');
var2 = get(handles.edit2, 'String');
var3 = get(handles.edit4, 'String');

```

```

handles.Demand_Cost = var1;
handles.Consumption_Cost = var2;
handles.alpha = var3;

handles.PowerType = 'kW';
guidata(hObject,handles)

% --- Outputs from this function are returned to the command line.
function varargout = Demand_Visualization_Yearly_OutputFcn(hObject,
eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject     handle to pushbutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
cla(handles.axes1,'reset')
cla(handles.axes2,'reset')

try
    dlg = ProgressDialog( ...
        'StatusMessage', 'Working...', ...
        'Indeterminate', true);

data = handles.data;
Year = str2num(handles.Year);
Demand_Cost = str2num(handles.Demand_Cost);
Consumption_Cost = str2num(handles.Consumption_Cost);
alpha = str2num(handles.alpha);
Power_Type = handles.PowerType;

var1 = find(data(:,1) == Year);
var2 = data(var1,:);

var32 = find(var2(:,end) == 0);
var2(var32,:) = [];

var3 = unique(var2(:,2));

for i = 1:length(var3(:,1))
    var4 = var3(i);

```

```

var5 = find(var2(:,2) == var4);
var6 = var2(var5,end);
var28 = var2(var5,4);
Monthly_Demand(i) = max(var6);
Monthly_Demand_Cost(i) = Monthly_Demand(i) * Demand_Cost;
var29 = find(var6 == Monthly_Demand(i));
Demand_Time(i) = var28(var29(1)) * (1/60);
var9 = unique(var2(var5,3));
for j = 1:length(var9(:,1))
    var10 = var9(j);
    var11 = find(var2(:,2) == var4 & var2(:,3) == var10);
    var16{i,1}(j,1) = min(var2(var11,end));
    var12 = var2(var11,4);
    var13 = var2(var11,end);
    var14{i,1}(j,1) = trapz(var12,var13) * (1 / 60);
    var14{i,1}(j,2) = var14{i,1}(j,1) * Consumption_Cost;
end
Monthly_Consumption(i) = sum(var14{i,1}(:,1));
Monthly_Consumption_Cost(i) = sum(var14{i,1}(:,2));
end

var15 = cell2mat(var14);
Yearly_Consumption = sum(var15(:,1));
Yearly_Consumption_Cost = sum(var15(:,2));
Yearly_Demand_Cost = sum(Monthly_Demand_Cost);

bar(handles.axes1,var3,Monthly_Demand);
xlabel(handles.axes1,'Time (Month)')
ylabel(handles.axes1, strcat('Demand (' ,Power_Type, ')'))
set(handles.axes1,'YGrid','on')
var28 = get(handles.axes1,'YTick');
set(handles.axes1,'YTick',fix(linspace(var28(1),var28(end),10)))

bar(handles.axes2,var3,Monthly_Consumption ./ 10^3);
xlabel(handles.axes2,'Time (Month)')
ylabel(handles.axes2, strcat('Consumption (' ,Power_Type, 'h)', ' [x 10^3]'))
set(handles.axes2,'YGrid','on')
var28 = get(handles.axes2,'YTick');
set(handles.axes2,'YTick',fix(linspace(var28(1),var28(end),10)))

bar(handles.axes3,var3,Monthly_Demand_Cost ./ 10^3)
xlabel(handles.axes3,'Time (Month)')
ylabel(handles.axes3,'Demand Cost ($) [x 10^3]')
set(handles.axes3,'YGrid','on')
% var28 = get(handles.axes3,'YTick');
% set(handles.axes3,'YTick',fix(linspace(var28(1),var28(end),10)))

bar(handles.axes4,var3,Monthly_Consumption_Cost ./ 10^3)
xlabel(handles.axes4,'Time (Month)')
ylabel(handles.axes4,'Consumption Cost ($) [x 10^3]')

```

```

set(handles.axes4,'YGrid','on')
var28 = get(handles.axes4,'YTick');
set(handles.axes4,'YTick',fix(linspace(var28(1),var28(end),10)))

for i = 1:length(var3(:,1))
    var17 = min(var16{i,1});
    var18 = max(var16{i,1});
    var19 = var3(i);
    var20 = find(var2(:,2) == var19);
    var21 = var2(var20,end);
    var22 = find(var21 >= var17 & var21 <= var18);
    Modell1_Percent_Offtime(i) = length(var22) * (1 / 4);
end

for i = 1:length(var3(:,1))
    var24 = var3(i);
    var25 = find(var2(:,2) == var24);
    var26 = var2(var25,end);
    Average_Monthly_Demand = mean(var26);
    Minimum_Demand = min(var26);
    func1 = Minimum_Demand + alpha .* (Average_Monthly_Demand -
Minimum_Demand);
    var27 = find(var26 >= Minimum_Demand & var26 <= func1);
    Model2_Percent_Offtime(i) = length(var27) * (1 / 4);
end

func1 = min(var2(:,6)) + alpha .* (mean((var2(:,6))) - min(var2(:,6)));

var30 =
datenum(cat(2,var2(:,1:3),zeros(length(var2(:,1)),1),var2(:,4),zeros(le
ngth(var2(:,1)),1)));
var31 = mean(var2(:,6));

figure(1);
plot(var30,var2(:,6));hold on
plot(var30, repmat(var31,1,length(var30)), 'c')
plot(var30, repmat(min(var2(:,6)),1,length(var30)), 'k')
plot(var30, repmat(var18,1,length(var30)), 'g')
plot(var30, repmat(func1,1,length(var30)), 'r')

Size = 19;
h1 = legend('Demand Curve','Demand Average','Yearly Base Demand','Model
1 Limit','Model 2 Limit');
set(h1, 'Location', 'NorthOutside','Orientation','Horizontal')
xlim([min(var30),max(var30)])
xlabel('Date Number')
ylabel(strcat('Demand (',Power_Type,')'))
% ylabel(strcat('Demand
(',Power_Type,')'),'FontSize',Size,'FontWeight','Bold')
% set(gca,'fontsize',Size,'FontWeight','Bold')
% xlabel('Date Number','FontSize',Size,'FontWeight','Bold')

```

```

% plot(var30, repmat(func1,1,length(var30)), 'r', 'LineWidth',10)

set(handles.edit5, 'String', num2str(fix(Yearly_Demand_Cost)))
set(handles.edit6, 'String', num2str(fix(Yearly_Consumption)))
set(handles.edit7, 'String', num2str(fix(Yearly_Consumption_Cost)))
set(handles.edit8, 'String', num2str(fix(sum(Modell1_Percent_Offtime))))
set(handles.edit9, 'String', num2str(sum(fix(Model2_Percent_Offtime))))
set(handles.edit10, 'String', num2str(fix(sum(Monthly_Demand))))
set(handles.edit11, 'String', Power_Type)
set(handles.edit12, 'String', strcat(Power_Type, 'h'))

var110 = find(data(:,1) == Year);
var120 = data(var110,:);
var130 = unique(var120(:,2));

for i = 1:length(var130)
    var140 = var130(i);
    var150 = find(var120(:,2) == var140);
    [C10 I10] = max(var120(var150,end));
    var160(i) = var120(I10,end-2);
    var170(i) = min(var120(var150,end));
    var180(i) = mean(var120(var150,end));
end

figure(2)
bar(var130,var160 ./ 60)
xlabel('Time (Month)')
ylabel('Peak Time (Hr)')
set(gca, 'YTick', 0:2:24)
grid on

figure(3)
bar(var130, [var170;var180]', 'grouped')
xlabel('Time (Month)')
ylabel('Demand (kW)')
legend('Minimum Demand (kW)', 'Average Demand (kW)')
grid on

catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

handles.var3 = var3;
handles.Monthly_Consumption_Cost = Monthly_Consumption_Cost;
handles.Monthly_Demand_Cost = Monthly_Demand_Cost;
handles.Monthly_Consumption = Monthly_Consumption;
handles.Power_Type = Power_Type;
handles.Monthly_Demand = Monthly_Demand;

guidata(hObject, handles)

```

```

function edit5_Callback(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit7_Callback(hObject, eventdata, handles)
% hObject      handle to edit7 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%         str2double(get(hObject,'String')) returns contents of edit7 as
a double

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit7 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit8_Callback(hObject, eventdata, handles)
% hObject      handle to edit8 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%         str2double(get(hObject,'String')) returns contents of edit8 as
a double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit8 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit9_Callback(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%         str2double(get(hObject,'String')) returns contents of edit9 as
a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject      handle to popupmenu1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to popupmenu1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
% str2double(get(hObject,'String')) returns contents of edit4 as
a double
    var1 = get(hObject,'String');
    alpha = var1;
    handles.alpha = alpha;
    guidata(hObject,handles);

```

```

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
% str2double(get(hObject,'String')) returns contents of edit1 as
a double
    var1 = get(hObject,'String');
    Demand_Cost = var1;
    handles.Demand_Cost = Demand_Cost;

```

```

guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%       str2double(get(hObject,'String')) returns contents of edit2 as
a double
    var1 = get(hObject,'String');
    Consumption_Cost = var1;
    handles.Consumption_Cost = Consumption_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1

```

```

% eventdata structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles     structure with handles and user data (see GUIDATA)

var1 = get(hObject,'String');
handles.PowerType = var1;
guidata(hObject,handles)

set(handles.edit13,'String',strcat('$ / ',var1,' mo'))
set(handles.edit14,'String',strcat('$ / ',var1,' h'))
set(handles.edit11,'String',var1)
set(handles.edit12,'String',strcat(var1,' h'))

function edit10_Callback(hObject, eventdata, handles)
% hObject     handle to edit10 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%         str2double(get(hObject,'String')) returns contents of edit10
as a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit10 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject     handle to edit11 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit11 as text
%         str2double(get(hObject,'String')) returns contents of edit11
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit12 as text
%         str2double(get(hObject,'String')) returns contents of edit12
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%         str2double(get(hObject,'String')) returns contents of edit13
as a double

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit13 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit14_Callback(hObject, eventdata, handles)
% hObject      handle to edit14 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%         str2double(get(hObject,'String')) returns contents of edit14
as a double

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit14 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject      handle to togglebutton1 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

var3 = handles.var3;
Monthly_Demand = handles.Monthly_Demand;
Power_Type = handles.Power_Type;

Fig = figure;

bar(var3,Monthly_Demand);
xlabel('Time (Month)')
ylabel(strcat('Demand (' ,Power_Type,')'))
set(gca,'YGrid','on')
var28 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(var28(1),var28(end),10)))

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton2

var3 = handles.var3;
Monthly_Consumption = handles.Monthly_Consumption;
Power_Type = handles.Power_Type;

Fig = figure;

bar(var3,Monthly_Consumption ./ 10^3);
xlabel('Time (Month)')
ylabel(strcat('Consumption (' ,Power_Type,'h)', ' [x 10^3]'))
set(gca,'YGrid','on')
var28 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(var28(1),var28(end),10)))

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

```

```

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton3

Fig = figure;

var3 = handles.var3;
Monthly_Demand_Cost = handles.Monthly_Demand_Cost;

bar(var3,Monthly_Demand_Cost ./ 10^3)
xlabel('Time (Month)')
ylabel('Demand Cost ($) [x 10^3]')
set(gca,'YGrid','on')
var28 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(var28(1),var28(end),10)))

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on button press in togglebutton4.
function togglebutton4_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton4

var3 = handles.var3;
Monthly_Consumption_Cost = handles.Monthly_Consumption_Cost;

Fig = figure;

bar(var3,Monthly_Consumption_Cost ./ 10^3)
xlabel('Time (Month)')
ylabel('Consumption Cost ($) [x 10^3]')
set(gca,'YGrid','on')
var28 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(var28(1),var28(end),10)))

```

```
answer1 = inputdlg('FileName?');  
answer2 = cellstr(answer1);  
saveas(gcf,answer2{1},'epsc')  
saveas(gcf,answer2{1},'fig')  
close(Fig)
```

Monthly Demand Visualization Tool

```
function varargout = Demand_Visualization_Monthly(varargin)
% DEMAND_VISUALIZATION_MONTHLY MATLAB code for
Demand_Visualization_Monthly.fig
% DEMAND_VISUALIZATION_MONTHLY, by itself, creates a new
DEMAND_VISUALIZATION_MONTHLY or raises the existing
% singleton*.
%
% H = DEMAND_VISUALIZATION_MONTHLY returns the handle to a new
DEMAND_VISUALIZATION_MONTHLY or the handle to
% the existing singleton*.
%
%
DEMAND_VISUALIZATION_MONTHLY('CALLBACK',hObject,eventData,handles,...)
calls the local
% function named CALLBACK in DEMAND_VISUALIZATION_MONTHLY.M with
the given input arguments.
%
% DEMAND_VISUALIZATION_MONTHLY('Property','Value',...) creates a
new DEMAND_VISUALIZATION_MONTHLY or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before
Demand_Visualization_Monthly_OpeningFcn gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
Demand_Visualization_Monthly_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Demand_Visualization_Monthly

% Last Modified by GUIDE v2.5 18-Jun-2014 10:51:01

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn',
@Demand_Visualization_Monthly_OpeningFcn, ...
                  'gui_OutputFcn',
@Demand_Visualization_Monthly_OutputFcn, ...
                  'gui_LayoutFcn', [] , ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
```

```

end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Demand_Visualization_Monthly is made
% visible.
function Demand_Visualization_Monthly_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Demand_Visualization_Monthly
% (see VARARGIN)

% Choose default command line output for Demand_Visualization_Monthly
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;
data = handles.data;

Years = cellstr(num2str(unique(data(:,1))));
Months = cellstr(num2str(unique(data(:,2))));
set(handles.YearPopUp, 'String', Years)
% set(handles.MonthPopUp, 'String', Months)
guidata(hObject, handles)

var1=get(handles.YearPopUp, 'String');
var2=get(handles.YearPopUp, 'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject, handles)

var7 = find(data(:,1) == str2num(Year));
var8 = data(var7,2);
var9 = unique(var8);
set(handles.MonthPopUp, 'String', cellstr(num2str(var9)))

var3=get(handles.MonthPopUp, 'String');
var4=get(handles.MonthPopUp, 'Value');
Month=var3{var4};
handles.Month = Month;
guidata(hObject, handles)

```

```

var5 = get(handles.DemandCostText, 'String');
handles.Demand_Cost = var5;
guidata(hObject,handles)

var6 = get(handles.ConsumptionCostText, 'String');
handles.Consumption_Cost = var6;
guidata(hObject,handles)

handles.PowerType = 'kW';
guidata(hObject,handles)

% UIWAIT makes Demand_Visualization_Monthly wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Demand_Visualization_Monthly_OutputFcn(hObject,
eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject     handle to axes1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: place code in OpeningFcn to populate axes1

% --- Executes on selection change in YearPopUp.
function YearPopUp_Callback(hObject, eventdata, handles)
% hObject     handle to YearPopUp (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject, 'String')) returns YearPopUp
contents as cell array
%           contents{get(hObject, 'Value')} returns selected item from
YearPopUp
var1=get(handles.YearPopUp, 'String');

```

```

var2=get(handles.YearPopUp, 'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

data = handles.data;
var7 = find(data(:,1) == str2num(Year));
var8 = data(var7,2);
var9 = unique(var8);
set(handles.MonthPopUp, 'String', cellstr(num2str(var9)))

% handles.Year = str2num(get(hObject, 'String'));
% guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function YearPopUp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YearPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

% --- Executes on selection change in MonthPopUp.
function MonthPopUp_Callback(hObject, eventdata, handles)
% hObject    handle to MonthPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject, 'String')) returns MonthPopUp
contents as cell array
%         contents{get(hObject, 'Value')} returns selected item from
MonthPopUp
var1=get(handles.MonthPopUp, 'String');
var2=get(handles.MonthPopUp, 'Value');
Month=var1{var2};
handles.Month = Month;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function MonthPopUp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MonthPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function DemandCostText_Callback(hObject, eventdata, handles)
% hObject    handle to DemandCostText (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of DemandCostText as
text
%       str2double(get(hObject,'String')) returns contents of
DemandCostText as a double

```

```

var1 = get(hObject,'String');
Demand_Cost = var1;
handles.Demand_Cost = Demand_Cost;
guidata(hObject,handles)

```

```

% --- Executes during object creation, after setting all properties.
function DemandCostText_CreateFcn(hObject, eventdata, handles)
% hObject    handle to DemandCostText (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function ConsumptionCostText_Callback(hObject, eventdata, handles)
% hObject    handle to ConsumptionCostText (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of ConsumptionCostText
as text
%       str2double(get(hObject,'String')) returns contents of
ConsumptionCostText as a double

```

```

    var1 = get(hObject, 'String');
    Consumption_Cost = var1;
    handles.Consumption_Cost = Consumption_Cost;
    guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function ConsumptionCostText_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ConsumptionCostText (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.

if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

% --- Executes during object creation, after setting all properties.
function PowerTypeGroup_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PowerTypeGroup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% --- Executes on button press in EngageButton.
function EngageButton_Callback(hObject, eventdata, handles)
% hObject    handle to EngageButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cla

try

%     dlg = ProgressDialog( ...
%     'StatusMessage', 'Working...', ...
%     'Indeterminate', true);

Month = str2num(handles.Month);
Year = str2num(handles.Year);
Demand_Cost = str2num(handles.Demand_Cost);
Consumption_Cost = str2num(handles.Consumption_Cost);
Power_Type = handles.PowerType;

```

```

num = handles.data;

var4 = find(num(:,1) == Year & num(:,2) == Month);
var5 = num(var4,:);
var6 = unique(var5(:,3));

for i = 1:numel(var6)
    var7 = var6(i);
    var8 = find(var5(:,3) == var7);
    var9 = var5(var8,6);
    var10(1,i) = max(var9);
    var10(2,i) = min(var9);
end

%Demand Info Stuff
Minimum_Demand = min(var10(2,:));
Maximum_Demand = max(var10(1,:));
Demand_Charge = Demand_Cost * Maximum_Demand;
Maximum_Demand_Row = find(var5(:,end) == Maximum_Demand);
Max_Day = var5(Maximum_Demand_Row,3);
Max_Time = var5(Maximum_Demand_Row,4);
Average_Demand = mean(var5(:,end));
Peak_Drop = Maximum_Demand - Average_Demand;
Peak_Drop_Savings = Peak_Drop .* Demand_Cost;

set(handles.LowestDemand,'String',num2str(fix(Minimum_Demand)))
set(handles.MaximumDemandEdit,'String',num2str(fix(Maximum_Demand)))
set(handles.DemandChargeEdit,'String',num2str(fix(Demand_Charge)))
set(handles.MaximumDemandDayEdit,'String',num2str(fix(Max_Day)))
set(handles.AverageDemandEdit,'String',num2str(fix(Average_Demand)))

%Consumption Info Stuff
var11 = (var5(:,3)) .* 1440 + var5(:,4);
Consumption = trapz(var11,var5(:,6)) * (1 / 60);
Consumption_Charge = Consumption * Consumption_Cost;
Minimum_Consumption_Elimination_Savings = trapz(var6',var10(2,:)) * 24;
Minimum_Consumption_Elimination_cost_Savings =
Minimum_Consumption_Elimination_Savings * Consumption_Cost;
Possible_Minimum_Consumption_Elimination_Savings =
Minimum_Consumption_Elimination_Savings -
trapz(var6',repmat(Minimum_Demand,1,numel(var6')) * 24;
Possible_Minimum_Consumption_Elimination_Cost_Savings =
Possible_Minimum_Consumption_Elimination_Savings * Consumption_Cost;

set(handles.MonthlyConsumptionEdit,'String',num2str(fix(Consumption)))
set(handles.ConsumptionChargeEdit,'String',num2str(fix(Consumption_Charge)))
set(handles.MinimumConsumptionEliminationSavingsEdit,'String',num2str(fix(Minimum_Consumption_Elimination_Savings)))
set(handles.MinimumConsumptionEliminationCostSavingsEdit,'String',num2str(fix(Minimum_Consumption_Elimination_cost_Savings)))

```

```

set(handles.PossibleMinimumConsumptionEliminationSavingsEdit,'String',num2str(fix(Possible_Minimum_Consumption_Elimination_Savings)))
set(handles.PossibleMinimumConsumptionEliminationCostSavings,'String',num2str(fix(Possible_Minimum_Consumption_Elimination_Cost_Savings)))

%Plotting
plot(handles.axes2,var6',var10(1,:), 'b', 'Marker', '.');hold on
plot(handles.axes2,var6',var10(2,:), 'r', 'Marker', '.');hold on
plot(handles.axes2,var6', repmat(Average_Demand,1,numel(var6')), 'g');hold on
plot(handles.axes2,var6', repmat(Minimum_Demand,1,numel(var6')), 'k');hold on
plot(handles.axes1,var11 ./ 1440,var5(:,end), 'k');hold on

h = legend(handles.axes2, 'Maximum Demand', 'Base Demand', 'Average Demand', 'Monthly Base Demand');
set(h, 'Location', 'SouthOutside')
xlabel(handles.axes1, 'Time (Day)')
xlabel(handles.axes2, 'Time (Day)')
ylabel(handles.axes1, strcat('Demand (',Power_Type,')'))
ylabel(handles.axes2, strcat('Demand (',Power_Type,')'))
b = get(gca, 'YTick');
% set(handles.axes1, 'YTick', fix(linspace(0,b(end),10)))
% set(handles.axes2, 'YTick', fix(linspace(0,b(end),10)))
set(handles.axes1, 'YTick', fix(linspace(0,max(var5(:,end))+10,10)))
set(handles.axes2, 'YTick', fix(linspace(0,max(var5(:,end))+10,10)))
set(handles.axes1, 'XTick', var6)
set(handles.axes2, 'XTick', var6)
grid on
grid(handles.axes1, 'On')
xlim(handles.axes1, [var6(1), var6(end) + 1])
xlim(handles.axes2, [var6(1), var6(end)])
var100 = get(handles.axes1, 'YTick');
ylim(handles.axes1, [0, var100(end)])
var100 = get(handles.axes2, 'YTick');
ylim(handles.axes2, [0, var100(end)])

h2 = legend(handles.axes1, 'Facility Demand');
set(h2, 'Location', 'SouthOutside')

% var11 = find(num(:,1) == Year);
% var12 = num(var11,:);
% var13 = unique(var12(:,2));
%
% for i = 1:length(var13)
%     var14 = var13(i);
%     var15 = find(var12(:,2) == var14);
%     [C1 I1] = max(var12(var15,end));
%     var16(i) = var12(I1,end-2);
%     var17(i) = min(var12(var15,end));
%     var18(i) = mean(var12(var15,end));
% end
%

```

```

% figure(1)
% bar(var13,var16 ./ 60)
% xlabel('Time (Month)')
% ylabel('Peak Time (Hr)')
% set(gca,'YTick',0:2:24)
% grid on
%
% figure(2)
% bar(var13,[var17;var18]','grouped')
% xlabel('Time (Month)')
% ylabel('Demand (kW)')
% legend('Minimum Demand (kW)','Average Demand (kW)')
% grid on

catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj,'extended','hyperlinks','on'),'Error');
end

handles.var6 = var6;
handles.var10 = var10;
handles.Average_Demand = Average_Demand;
handles.Minimum_Demand = Minimum_Demand;
handles.var11 = var11;
handles.var5 = var5;

guidata(hObject,handles);

% --- Executes when selected object is changed in PowerTypeGroup.
function PowerTypeGroup_SelectionChangeFcn(hObject, eventdata, handles)
var1 = get(hObject,'String');
handles.PowerType = var1;
guidata(hObject,handles);

set(handles.edit16,'String',var1)
set(handles.edit17,'String',var1)
set(handles.edit18,'String',var1)
set(handles.edit20,'String',strcat(var1,'h'))
set(handles.edit21,'String',strcat(var1,'h'))
set(handles.edit22,'String',strcat(var1,'h'))
set(handles.edit23,'String',strcat('$ / ',' ',var1,' mo'))
set(handles.edit24,'String',strcat('$ / ',' ',var1,' h'))

% switch var1 % Get Tag of selected object.
%     case 'kWButton'
%         Power_Type = 'kW'
%         handles.PowerType = Power_Type;
%         guidata(hObject,handles)
%     case 'kVAButton'
%         Power_Type = 'kVA'

```

```

%         handles.PowerType = Power_Type;
%         guidata(hObject,handles)
% end
% hObject    handle to the selected object in PowerTypeGroup
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)

% --- Executes during object creation, after setting all properties.
function kWButton_CreateFcn(hObject, eventdata, handles)
% hObject    handle to kWButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

function LowestDemand_Callback(hObject, eventdata, handles)
% hObject    handle to LowestDemand (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of LowestDemand as text
%        str2double(get(hObject,'String')) returns contents of
LowestDemand as a double

% --- Executes during object creation, after setting all properties.
function LowestDemand_CreateFcn(hObject, eventdata, handles)
% hObject    handle to LowestDemand (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function MaximumDemandDayEdit_Callback(hObject, eventdata, handles)
% hObject    handle to MaximumDemandDayEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of MaximumDemandDayEdit
as text
%         str2double(get(hObject,'String')) returns contents of
MaximumDemandDayEdit as a double

```

```

% --- Executes during object creation, after setting all properties.
function MaximumDemandDayEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MaximumDemandDayEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function AverageDemandEdit_Callback(hObject, eventdata, handles)
% hObject    handle to AverageDemandEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of AverageDemandEdit as
text
%         str2double(get(hObject,'String')) returns contents of
AverageDemandEdit as a double

```

```

% --- Executes during object creation, after setting all properties.
function AverageDemandEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AverageDemandEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function MaximumDemandEdit_Callback(hObject, eventdata, handles)

```

```

% hObject    handle to MaximumDemandEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of MaximumDemandEdit as
text
%          str2double(get(hObject,'String')) returns contents of
MaximumDemandEdit as a double

% --- Executes during object creation, after setting all properties.
function MaximumDemandEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MaximumDemandEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function DemandChargeEdit_Callback(hObject, eventdata, handles)
% hObject    handle to DemandChargeEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of DemandChargeEdit as
text
%          str2double(get(hObject,'String')) returns contents of
DemandChargeEdit as a double

% --- Executes during object creation, after setting all properties.
function DemandChargeEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to DemandChargeEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function MonthlyConsumptionEdit_Callback(hObject, eventdata, handles)
% hObject      handle to MonthlyConsumptionEdit (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
MonthlyConsumptionEdit as text
%          str2double(get(hObject,'String')) returns contents of
MonthlyConsumptionEdit as a double

% --- Executes during object creation, after setting all properties.
function MonthlyConsumptionEdit_CreateFcn(hObject, eventdata, handles)
% hObject      handle to MonthlyConsumptionEdit (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function ConsumptionChargeEdit_Callback(hObject, eventdata, handles)
% hObject      handle to ConsumptionChargeEdit (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
ConsumptionChargeEdit as text
%          str2double(get(hObject,'String')) returns contents of
ConsumptionChargeEdit as a double

% --- Executes during object creation, after setting all properties.
function ConsumptionChargeEdit_CreateFcn(hObject, eventdata, handles)
% hObject      handle to ConsumptionChargeEdit (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

```

```

        set(hObject, 'BackgroundColor', 'white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject      handle to edit11 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit11 as text
%          str2double(get(hObject, 'String')) returns contents of edit11
as a double

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit11 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function MinimumConsumptionEliminationSavingsEdit_Callback(hObject,
eventdata, handles)
% hObject      handle to MinimumConsumptionEliminationSavingsEdit (see
GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of
MinimumConsumptionEliminationSavingsEdit as text
%          str2double(get(hObject, 'String')) returns contents of
MinimumConsumptionEliminationSavingsEdit as a double

% --- Executes during object creation, after setting all properties.
function MinimumConsumptionEliminationSavingsEdit_CreateFcn(hObject,
eventdata, handles)
% hObject      handle to MinimumConsumptionEliminationSavingsEdit (see
GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns

```

called

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function MinimumConsumptionEliminationCostSavingsEdit_Callback(hObject,
 eventdata, handles)
% hObject    handle to MinimumConsumptionEliminationCostSavingsEdit
%           (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of
MinimumConsumptionEliminationCostSavingsEdit as text
%       str2double(get(hObject,'String')) returns contents of
MinimumConsumptionEliminationCostSavingsEdit as a double
```

```
% --- Executes during object creation, after setting all properties.
```

```
function
MinimumConsumptionEliminationCostSavingsEdit_CreateFcn(hObject,
 eventdata, handles)
% hObject    handle to MinimumConsumptionEliminationCostSavingsEdit
%           (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function
PossibleMinimumConsumptionEliminationSavingsEdit_Callback(hObject,
 eventdata, handles)
% hObject    handle to PossibleMinimumConsumptionEliminationSavingsEdit
%           (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```

% Hints: get(hObject,'String') returns contents of
PossibleMinimumConsumptionEliminationSavingsEdit as text
%         str2double(get(hObject,'String')) returns contents of
PossibleMinimumConsumptionEliminationSavingsEdit as a double

% --- Executes during object creation, after setting all properties.
function
PossibleMinimumConsumptionEliminationSavingsEdit_CreateFcn(hObject,
 eventdata, handles)
% hObject    handle to PossibleMinimumConsumptionEliminationSavingsEdit
%             (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%             called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
 get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function
PossibleMinimumConsumptionEliminationCostSavings_Callback(hObject,
 eventdata, handles)
% hObject    handle to PossibleMinimumConsumptionEliminationCostSavings
%             (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
PossibleMinimumConsumptionEliminationCostSavings as text
%         str2double(get(hObject,'String')) returns contents of
PossibleMinimumConsumptionEliminationCostSavings as a double

% --- Executes during object creation, after setting all properties.
function
PossibleMinimumConsumptionEliminationCostSavings_CreateFcn(hObject,
 eventdata, handles)
% hObject    handle to PossibleMinimumConsumptionEliminationCostSavings
%             (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%             called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),

```

```

get(0,'defaultUicontrolBackgroundColor')
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
% hObject      handle to edit16 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as text
%         str2double(get(hObject,'String')) returns contents of edit16
as a double

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit16 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(hObject, eventdata, handles)
% hObject      handle to edit17 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit17 as text
%         str2double(get(hObject,'String')) returns contents of edit17
as a double

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit17 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.

```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject      handle to edit18 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%          str2double(get(hObject,'String')) returns contents of edit18
as a double

% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit18 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit19_Callback(hObject, eventdata, handles)
% hObject      handle to edit19 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%          str2double(get(hObject,'String')) returns contents of edit19
as a double

% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit19 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as text
%       str2double(get(hObject,'String')) returns contents of edit20
%       as a double

% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as text
%       str2double(get(hObject,'String')) returns contents of edit21
%       as a double

% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit22 as text
%       str2double(get(hObject,'String')) returns contents of edit22
as a double

% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit23 as text
%       str2double(get(hObject,'String')) returns contents of edit23
as a double

% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit24_Callback(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit24 as text
%       str2double(get(hObject,'String')) returns contents of edit24
%       as a double

% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%           called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

var5 = handles.var5;
var11 = handles.var11;
Power_Type = handles.PowerType;
var6 = handles.var6;

Fig = figure;
plot(var11 ./ 1440,var5(:,end),'k')

```

```

h = legend('Facility Demand');
set(h, 'Location', 'SouthOutside')
xlabel('Time (Day)')
ylabel(strcat('Demand (',Power_Type,')'))
b = get(gca,'YTick');
set(gca,'YTick',round(linspace(0,b(end),10)))
set(gca,'XTick',var6)
grid on
xlim([var6(1),var6(end) + 1]);
var100 = get(gca,'YTick');
ylim([0,var100(end)])

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject      handle to togglebutton2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton2
var6 = handles.var6;
var10 = handles.var10;
Average_Demand = handles.Average_Demand;
Minimum_Demand = handles.Minimum_Demand;
Power_Type = handles.PowerType;

Fig = figure;
plot(var6',var10(1,:), 'b','Marker','.');hold on
plot(var6',var10(2,:), 'r','Marker','.');hold on
plot(var6',repmat(Average_Demand,1,numel(var6')), 'g');hold on
plot(var6',repmat(Minimum_Demand,1,numel(var6')), 'k');hold on

h = legend('Maximum Demand','Daily Base Demand','Average
Demand','Monthly Base Demand');
set(h, 'Location', 'SouthOutside')
xlabel('Time (Day)')
ylabel(strcat('Demand (',Power_Type,')'))
b = get(gca,'YTick');
set(gca,'YTick',round(linspace(0,b(end),10)))
set(gca,'XTick',var6)
grid on
xlim([var6(1),var6(end) + 1]);
var100 = get(gca,'YTick');
ylim([0,var100(end)])

answer1 = inputdlg('FileName?');

```

```
answer2 = cellstr(answer1);  
saveas(gcf,answer2{1},'epsc')  
saveas(gcf,answer2{1},'fig')  
close(Fig)
```

Day of Week Demand Visualization Tool

```
function varargout = Demand_Visualization_DayoftheWeek(varargin)
% DEMAND_VISUALIZATION_DAYOFTHEWEEK MATLAB code for
Demand_Visualization_DayoftheWeek.fig
% DEMAND_VISUALIZATION_DAYOFTHEWEEK, by itself, creates a new
DEMAND_VISUALIZATION_DAYOFTHEWEEK or raises the existing
% singleton*.
%
% H = DEMAND_VISUALIZATION_DAYOFTHEWEEK returns the handle to a
new DEMAND_VISUALIZATION_DAYOFTHEWEEK or the handle to
% the existing singleton*.
%
%
DEMAND_VISUALIZATION_DAYOFTHEWEEK('CALLBACK',hObject,eventData,handles,
...) calls the local
% function named CALLBACK in DEMAND_VISUALIZATION_DAYOFTHEWEEK.M
with the given input arguments.
%
% DEMAND_VISUALIZATION_DAYOFTHEWEEK('Property','Value',...)
creates a new DEMAND_VISUALIZATION_DAYOFTHEWEEK or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before
Demand_Visualization_DayoftheWeek_OpeningFcn gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
Demand_Visualization_DayoftheWeek_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Demand_Visualization_DayoftheWeek

% Last Modified by GUIDE v2.5 18-Jun-2014 11:02:03

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn',
@Demand_Visualization_DayoftheWeek_OpeningFcn, ...
                  'gui_OutputFcn',
@Demand_Visualization_DayoftheWeek_OutputFcn, ...
                  'gui_LayoutFcn', [] , ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
```

```

end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Demand_Visualization_DayoftheWeek is made
% visible.
function Demand_Visualization_DayoftheWeek_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to
Demand_Visualization_DayoftheWeek (see VARARGIN)

% Choose default command line output for
Demand_Visualization_DayoftheWeek
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Demand_Visualization_DayoftheWeek wait for user response
(see UIRESUME)
% uiwait(handles.figure1);

%Getting data set
guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;

%Populating Years Popup
Years = cellstr(num2str(unique(data(:,1))));
set(handles.YearPopUp, 'String', Years)

%Setting Years handle
var1=get(handles.YearPopUp, 'String');
var2=get(handles.YearPopUp, 'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject, handles)

%Populating Days Popup
var7 = str2num(Year);
var8 = find(data(:,1) == var7);
var9 = data(var8,:);

```

```

var10 = unique(var9(:,5));
var11 = cellstr(num2str(var10));
set(handles.WeekDayPopUp, 'String', var11);
guidata(hObject, handles)

%Setting Days Handle
var12 = get(handles.WeekDayPopUp, 'String');
var13 = get(handles.WeekDayPopUp, 'Value');
Day = var12{var13};
handles.WeekDay = Day;
guidata(hObject, handles)

%Set Initial bin number
% var20 = get(handles.edit9, 'String');
% handles.bins = var20;
% guidata(hObject, handles)

%Setting Consumption Cost
var21 = get(handles.ConsumptionCostEdit, 'String');
handles.Consumption_Cost = var21;
var22 = get(handles.edit23, 'String');
handles.Demand_Cost = var22;

handles.PowerType = 'kW';
guidata(hObject, handles)

% --- Outputs from this function are returned to the command line.
function varargout =
Demand_Visualization_DayoftheWeek_OutputFcn(hObject, eventdata,
handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function ConsumptionCostEdit_Callback(hObject, eventdata, handles)
% hObject handle to ConsumptionCostEdit (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of ConsumptionCostEdit
as text
% str2double(get(hObject, 'String')) returns contents of
ConsumptionCostEdit as a double
var1 = get(hObject, 'String');
Consumption_Cost = var1;
handles.Consumption_Cost = Consumption_Cost;

```

```

guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function ConsumptionCostEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ConsumptionCostEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in EngageButton.
function EngageButton_Callback(hObject, eventdata, handles)
% hObject    handle to EngageButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cla reset

try

%     dlg = ProgressDialog( ...
%     'StatusMessage', 'Working...', ...
%     'Indeterminate', true);

num = handles.data;
Weekday = str2num(handles.WeekDay);
Year = str2num(handles.Year);
Power_Type = handles.PowerType;
Consumption_Cost = str2num(handles.Consumption_Cost);
Demand_Cost = str2num(handles.Demand_Cost);

var11 = find(num(:,1) == Year & num(:,5) == Weekday);
var12 = num(var11,:);
var13 = unique(var12(:,2));

for i = 1:numel(var13)%Looping throught the months
    var14 = var13(i);%Pick Month
    var15 = find(var12(:,2) == var14);%Find index of picked month
    var16 = var12(var15,3);%Getting the days in that month
    var17 = unique(var16);%finding the unique days in that month
    for j = 1:numel(var17)%looping through the days
        var18 = var17(j);%pick day
    end
end

```

```

        var19 = find(var12(:,2) == var14 & var12(:,3) == var18);%find
the index of the month and the day
        var20 = var12(var19,6);%get the demand for the indexed month and
day
        var21{i,1}(j,1) = max(var20);
        var22{i,1}(j,1) = min(var20);
        var35{i,1}(j,1) = mean(var20);
    end
end

var63 = cell2mat(var21);
var64 = cell2mat(var22);
var65 = cell2mat(var35);

bar(handles.axes2,1:length(var63),var63,'b');hold on
bar(handles.axes2,1:length(var65),var65,'g');hold on
bar(handles.axes2,1:length(var64),var64,'r');hold on
xlabel(handles.axes2,'Time (wk)')
ylabel(handles.axes2, strcat('Demand (',Power_Type,')'))
var61 = get(handles.axes2,'XTick');
var62 = get(handles.axes2,'YTick');
set(handles.axes2,'XTick',var61(1):4:var61(end))
set(handles.axes2,'YTick',fix(linspace(var62(1),var62(end),10)))
set(handles.axes2,'YGrid','on')
xlim([0,length(var63)+1])
legend('Maximum Demand','Average Demand','Base
Demand','Location','SouthOutside')

%Peak Times Monthly
var27 = find(num(:,1) == Year);
var28 = num(var27,:);
var29 = unique(var28(:,2));

for i = 1:length(var29)
    var30 = var29(i);
    var31 = find(var28(:,2) == var30);
    var32 = var28(var31,:);
    [C I] = max(var32(:,end));
    var33(i,:) = var32(I,1:3);
end

for i = 1:length(var13(:,1))
    var48 = find(var12(:,1) == Year & var12(:,2) == var13(i,1));
    var49 = var12(var48,3);
    var50 = unique(var49);
    for j = 1:length(var50(:,1))
        var51 = find(var12(:,1) == Year & var12(:,2) == var13(i,1) &
var12(:,3) == var50(j,1));
        var52 = var12(var51,4);
        var53 = var12(var51,6);
        var54(j,1) = trapz(var52,var53) * (1 / 60);
        var54(j,2) = var54(j,1) * Consumption_Cost;
    end
end

```

```

    var56{i,1} = var54;
end

var57 = cell2mat(var56);
Total_Consumption = sum(var57(:,1));
set(handles.edit7, 'String', num2str(fix(Total_Consumption)))
Total_Cost = sum(var57(:,2));
set(handles.edit24, 'String', num2str(fix(Total_Cost)))
Average_Consumption = mean(var57(:,1));
set(handles.edit5, 'String', num2str(fix(Average_Consumption)))
Average_Cost = mean(var57(:,2));
set(handles.edit6, 'String', num2str(fix(Average_Cost)))

Max_Peak = max(var63);
set(handles.edit2, 'String', num2str(fix(Max_Peak)))
Min_Peak = min(var64);
set(handles.edit3, 'String', num2str(fix(Min_Peak)))

var58 = weekday(datenum(var33));
var59 = find(var58 == Weekday);
var60 = var33(var59,:);

if isempty(var59) == 1
    Peak_Frequency = 0;

set(handles.PeakOccuranceEdit, 'String', num2str(fix(Peak_Frequency)))
    Total_Demand = 0;
    set(handles.edit19, 'String', num2str(fix(Total_Demand)))
    Total_Demand_Cost = 0;
    set(handles.edit20, 'String', num2str(fix(Total_Demand_Cost)))
    Total_Total_Cost = Total_Cost;
set(handles.edit8, 'String', num2str(fix(Total_Total_Cost)))
else
for i = 1:length(var60(:,1))
    c = cat(2, var60(i,1:3), 0, 0, 0);
    week_num(i) = ceil(diff(datenum([c(1), 1, 1, 0, 0, 0; c])) / 7);
end

scatter(week_num, var63(week_num,1)', 300, [1 0 1], 'fill', 'p')
legend('Maximum Demand', 'Average Demand', 'Base Demand', 'Peak
Days', 'Location', 'SouthOutside')

Peak_Frequency = (length(week_num) / length(var13)) * 100;
set(handles.PeakOccuranceEdit, 'String', num2str(fix(Peak_Frequency)))
Total_Demand = sum(var63(week_num,1));
set(handles.edit19, 'String', num2str(fix(Total_Demand)))
Total_Demand_Cost = Total_Demand * Demand_Cost;
set(handles.edit20, 'String', num2str(fix(Total_Demand_Cost)))
Total_Total_Cost = Total_Cost + Total_Demand_Cost;
set(handles.edit8, 'String', num2str(fix(Total_Total_Cost)))
handles.week_num = week_num;
guidata(hObject, handles)

```

```

end

catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

handles.var63 = var63;
handles.var64 = var64;
handles.var65 = var65;

guidata(hObject, handles)
% function edit9_Callback(hObject, eventdata, handles)
% % hObject      handle to edit9 (see GCBO)
% % eventdata    reserved - to be defined in a future version of MATLAB
% % handles      structure with handles and user data (see GUIDATA)
%
% % Hints: get(hObject, 'String') returns contents of edit9 as text
% %          str2double(get(hObject, 'String')) returns contents of edit9
as a double
%     var1 = get(hObject, 'String');
%     bins = var1;
%     handles.bins = bins;
%     guidata(hObject, handles);

% % --- Executes during object creation, after setting all properties.
% function edit9_CreateFcn(hObject, eventdata, handles)
% % hObject      handle to edit9 (see GCBO)
% % eventdata    reserved - to be defined in a future version of MATLAB
% % handles      empty - handles not created until after all CreateFcns
called
%
% % Hint: edit controls usually have a white background on Windows.
% %          See ISPC and COMPUTER.
% if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
%     set(hObject, 'BackgroundColor', 'white');
% end

function edit2_Callback(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit2 as text
%          str2double(get(hObject, 'String')) returns contents of edit2 as

```

a double

```
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as
a double
```

```
% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function PeakOccuranceEdit_Callback(hObject, eventdata, handles)
% hObject    handle to PeakOccuranceEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of PeakOccuranceEdit as
```

```

text
%         str2double(get(hObject,'String')) returns contents of
PeakOccuranceEdit as a double

% --- Executes during object creation, after setting all properties.
function PeakOccuranceEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PeakOccuranceEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%         str2double(get(hObject,'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit6 as text
%         str2double(get(hObject,'String')) returns contents of edit6 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit7 as text
%         str2double(get(hObject,'String')) returns contents of edit7 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as
a double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit8 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%           See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in YearPopUp.
function YearPopUp_Callback(hObject, eventdata, handles)
% hObject      handle to YearPopUp (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns YearPopUp
contents as cell array
%           contents{get(hObject,'Value')} returns selected item from
YearPopUp
var1=get(handles.YearPopUp,'String');
var2=get(handles.YearPopUp,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

var3 = handles.data;
var4 = find(var3(:,1) == str2num(Year));
var5 = var3(var4,:);
var6 = unique(var5(:,5));
var7 = cellstr(num2str(var6));
set(handles.WeekDayPopUp,'String',var7);

% --- Executes during object creation, after setting all properties.
function YearPopUp_CreateFcn(hObject, eventdata, handles)
% hObject      handle to YearPopUp (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in WeekDayPopUp.
function WeekDayPopUp_Callback(hObject, eventdata, handles)
% hObject     handle to WeekDayPopUp (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns WeekDayPopUp
contents as cell array
%           contents{get(hObject,'Value')} returns selected item from
WeekDayPopUp
var1=get(handles.WeekDayPopUp,'String');
var2=get(handles.WeekDayPopUp,'Value');
WeekDay=var1{var2};
handles.WeekDay = WeekDay;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function WeekDayPopUp_CreateFcn(hObject, eventdata, handles)
% hObject     handle to WeekDayPopUp (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in PowerToggleGroup.
function PowerToggleGroup_SelectionChangeFcn(hObject, eventdata,
handles)
% hObject     handle to the selected object in PowerToggleGroup
% eventdata   structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles     structure with handles and user data (see GUIDATA)
var1 = get(hObject,'String');

```

```

handles.PowerType = var1;
guidata(hObject,handles)

set(handles.edit10,'String',var1)
set(handles.edit11,'String',var1)
set(handles.edit22,'String',var1)
set(handles.edit17,'String',strcat(var1,'h'))
set(handles.edit18,'String',strcat(var1,'h'))
set(handles.edit28,'String',strcat('$ / ', ' ',var1,' mo'))
set(handles.edit27,'String',strcat(var1,'h'))

function edit10_Callback(hObject, eventdata, handles)
% hObject      handle to edit10 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%         str2double(get(hObject,'String')) returns contents of edit10
as a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit10 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject      handle to edit11 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%         str2double(get(hObject,'String')) returns contents of edit11
as a double

% --- Executes during object creation, after setting all properties.

```

```

function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%       str2double(get(hObject,'String')) returns contents of edit12
as a double

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%       str2double(get(hObject,'String')) returns contents of edit13
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%       str2double(get(hObject,'String')) returns contents of edit14
as a double

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%       str2double(get(hObject,'String')) returns contents of edit15
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as text
%         str2double(get(hObject,'String')) returns contents of edit16
as a double

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function PowerToggleGroup_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PowerToggleGroup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit17 as text
%        str2double(get(hObject,'String')) returns contents of edit17
as a double

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%        str2double(get(hObject,'String')) returns contents of edit18
as a double

% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit19_Callback(hObject, eventdata, handles)
% hObject      handle to edit19 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%         str2double(get(hObject,'String')) returns contents of edit19
as a double

% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit19 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
% hObject      handle to edit20 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as text
%         str2double(get(hObject,'String')) returns contents of edit20
as a double

% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit20 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit21_Callback(hObject, eventdata, handles)
% hObject      handle to edit21 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as text
%          str2double(get(hObject,'String')) returns contents of edit21
as a double

% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit21 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit22_Callback(hObject, eventdata, handles)
% hObject      handle to edit22 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit22 as text
%          str2double(get(hObject,'String')) returns contents of edit22
as a double

% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit22 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit23_Callback(hObject, eventdata, handles)
% hObject      handle to edit23 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit23 as text
%          str2double(get(hObject,'String')) returns contents of edit23
as a double
    var1 = get(hObject,'String');
    Demand_Cost = var1;
    handles.Demand_Cost = Demand_Cost;
    guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit23 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit24_Callback(hObject, eventdata, handles)
% hObject      handle to edit24 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit24 as text
%          str2double(get(hObject,'String')) returns contents of edit24
as a double

% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit24 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit25_Callback(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit25 as text
%       str2double(get(hObject,'String')) returns contents of edit25
%       as a double

% --- Executes during object creation, after setting all properties.
function edit25_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit26_Callback(hObject, eventdata, handles)
% hObject    handle to edit26 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit26 as text
%       str2double(get(hObject,'String')) returns contents of edit26
%       as a double

% --- Executes during object creation, after setting all properties.
function edit26_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit26 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit27_Callback(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit27 as text
%       str2double(get(hObject,'String')) returns contents of edit27
%       as a double

% --- Executes during object creation, after setting all properties.
function edit27_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit28_Callback(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit28 as text
%       str2double(get(hObject,'String')) returns contents of edit28
%       as a double

% --- Executes during object creation, after setting all properties.
function edit28_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%       called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject     handle to togglebutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

Fig = figure;

var63 = handles.var63;
var64 = handles.var64;
var65 = handles.var65;
Power_Type = handles.PowerType;
week_num = handles.week_num;

bar(1:length(var63),var63,'b');hold on
bar(1:length(var65),var65,'g');hold on
bar(1:length(var64),var64,'r');hold on
xlabel('Time (wk)')
ylabel(strcat('Demand (',Power_Type,')'))
var61 = get(gca,'XTick');
var62 = get(gca,'YTick');
set(gca,'XTick',var61(1):4:var61(end))
set(gca,'YTick',fix(linspace(var62(1),var62(end),10)))
set(gca,'YGrid','on')
xlim([0,length(var63)+1])
legend('Maximum Demand','Average Demand','Base
Demand','Location','SouthOutside')
scatter(week_num,var63(week_num,1)',300,[1 0 1],'fill','p')
legend('Maximum Demand','Average Demand','Base Demand','Peak
Days','Location','SouthOutside')

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

```

Daily of Week Demand Visualization Tool

```
function varargout = Demand_Visualization_Daily(varargin)
% DEMAND_VISUALIZATION_DAILY MATLAB code for
Demand_Visualization_Daily.fig
% DEMAND_VISUALIZATION_DAILY, by itself, creates a new
DEMAND_VISUALIZATION_DAILY or raises the existing
% singleton*.
%
% H = DEMAND_VISUALIZATION_DAILY returns the handle to a new
DEMAND_VISUALIZATION_DAILY or the handle to
% the existing singleton*.
%
%
DEMAND_VISUALIZATION_DAILY('CALLBACK', hObject, eventData, handles,...)
calls the local
% function named CALLBACK in DEMAND_VISUALIZATION_DAILY.M with the
given input arguments.
%
% DEMAND_VISUALIZATION_DAILY('Property','Value',...) creates a new
DEMAND_VISUALIZATION_DAILY or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before Demand_Visualization_Daily_OpeningFcn
gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
Demand_Visualization_Daily_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Demand_Visualization_Daily

% Last Modified by GUIDE v2.5 18-Jun-2014 10:53:27

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn',
@Demand_Visualization_Daily_OpeningFcn, ...
                  'gui_OutputFcn',
@Demand_Visualization_Daily_OutputFcn, ...
                  'gui_LayoutFcn', [] , ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
```

```

end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Demand_Visualization_Daily is made visible.
function Demand_Visualization_Daily_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Demand_Visualization_Daily (see
VARARGIN)

% Choose default command line output for Demand_Visualization_Daily
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;

%Set Intital Years
Years = cellstr(num2str(unique(data(:,1))));
set(handles.YearPopUp, 'String', Years)

guidata(hObject, handles);

%Set Initial Month
var1=get(handles.YearPopUp, 'String');
var2=get(handles.YearPopUp, 'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)
var7 = str2num(Year);
var8 = find(data(:,1) == var7);
var9 = data(var8, :);
var10 = unique(var9(:,2));
var11 = cellstr(num2str(var10));
set(handles.MonthPopUp, 'String', var11);
guidata(hObject,handles)

%Set Initial Day
var12=get(handles.MonthPopUp, 'String');

```

```

var13=get(handles.MonthPopUp,'Value');
var14=var12{var13};
var15 = str2num(var14);
var16 = find(data(:,1) == var7 & data(:,2) == var15);
var17 = data(var16,:);
var18 = unique(var17(:,3));
var19 = cellstr(num2str(var18));
set(handles.DayPopUp,'String',var19);
guidata(hObject,handles)

%Set Initial Consumption Cost
var20 = get(handles.edit3,'String');
var21 = str2num(var20);
handles.Consumption_Cost = var21;
guidata(hObject,handles)

handles.PowerType = 'kW';
guidata(hObject,handles)

% var1 = get(handles.PowerTypeGroup,'Tag');
% handles.PowerType = var1;
% guidata(hObject,handles)

% UIWAIT makes Demand_Visualization_Daily wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Demand_Visualization_Daily_OutputFcn(hObject,
eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject     handle to pushbutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
cla reset

try

```

```

% dlg = ProgressDialog( ...
%     'StatusMessage', 'Working...', ...
%     'Indeterminate', true);

Day1 = get(handles.DayPopUp, 'String');
Day2 = get(handles.DayPopUp, 'Value');
Day3 = str2num(Day1{Day2});

Month1 = get(handles.MonthPopUp, 'String');
Month2 = get(handles.MonthPopUp, 'Value');
Month3 = str2num(Month1{Month2});

Year1 = get(handles.YearPopUp, 'String');
Year2 = get(handles.YearPopUp, 'Value');
Year3 = str2num(Year1{Year2});

Power_Type = handles.PowerType;

num = handles.data;

Consumption_Cost = handles.Consumption_Cost;

var1 = find(num(:,1) == Year3 & num(:,2) == Month3 & num(:,3) == Day3);
var2 = num(var1,4);

Demand = num(var1,6);
Average = mean(Demand);
Maximum = max(Demand);
Minimum = min(Demand);

Consumption = trapz(var2,Demand) * (1 / 60);
Consumption_Charge = Consumption * Consumption_Cost;

Peak_Time1 = find(num(var1,end) == Maximum);
Peak_Time2 = num(Peak_Time1,4) / 60

set(handles.edit4, 'String', num2str(fix(Minimum)))
set(handles.edit15, 'String', num2str(fix(Average)))
set(handles.edit10, 'String', num2str(fix(Maximum)))
set(handles.edit9, 'String', num2str(fix(Consumption)))
set(handles.edit6, 'String', num2str(fix(Consumption_Charge)))
set(handles.edit7, 'String', num2str(Peak_Time2(1)))

plot(handles.axes1,var2,Demand);

plot(handles.axes1,var2 ./ 60,Demand,'b','Marker','.');hold on
plot(handles.axes1,var2 ./ 60, repmat(Minimum,1,numel(var2')), 'r');hold
on
plot(handles.axes1,var2 ./ 60, repmat(Average,1,numel(var2')), 'g');hold

```

```

on
plot(handles.axes1,var2 ./ 60,repmat(Maximum,1,numel(var2)),'k');hold
on

h = legend(handles.axes1,'Demand','Daily Base Demand','Average
Demand','Maximum Demand');
set(h, 'Location', 'SouthOutside')
xlabel(handles.axes1,'Time (hr)')
ylabel(handles.axes1, strcat('Demand (',Power_Type,')'))
var3 = get(gca,'XTick');
var4 = get(gca,'YTick');
set(handles.axes1,'YTick',round(linspace(var4(1),var4(end),10)))
set(handles.axes1,'XTick',1:1:24)
grid on

catch errorObj
% If there is a problem, we display the error message
erroridg(getReport(errorObj,'extended','hyperlinks','on'),'Error');
end

handles.var2 = var2;
handles.Minimum = Minimum;
handles.Average = Average;
handles.Maximum = Maximum;
handles.Demand = Demand;
handles.Power_Type = Power_Type;

guidata(hObject,handles)

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as
a double

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```
end
```

```
function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as
a double
```

```
% --- Executes during object creation, after setting all properties.
```

```
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as
a double
```

```
% --- Executes during object creation, after setting all properties.
```

```
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```

        set(hObject, 'BackgroundColor', 'white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit9 as text
%         str2double(get(hObject, 'String')) returns contents of edit9 as
a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit10_Callback(hObject, eventdata, handles)
% hObject      handle to edit10 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit10 as text
%         str2double(get(hObject, 'String')) returns contents of edit10
as a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit10 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),

```

```

get(0,'defaultUicontrolBackgroundColor')
    set(hObject,'BackgroundColor','white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11
as a double

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12
as a double

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.

```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit13_Callback(hObject, eventdata, handles)
% hObject      handle to edit13 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%         str2double(get(hObject,'String')) returns contents of edit13
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit13 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit14_Callback(hObject, eventdata, handles)
% hObject      handle to edit14 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%         str2double(get(hObject,'String')) returns contents of edit14
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit14 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as
a double

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject      handle to edit3 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%         str2double(get(hObject,'String')) returns contents of edit3 as
a double

    var1 = get(hObject,'String');
    Consumption_Cost = var1;
    handles.Consumption_Cost = str2num(Consumption_Cost);
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)

```

```

% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in YearPopUp.
function YearPopUp_Callback(hObject, eventdata, handles)
% hObject    handle to YearPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns YearPopUp
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
YearPopUp
var1=get(handles.YearPopUp,'String');
var2=get(handles.YearPopUp,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

var3 = handles.data;
var4 = find(var3(:,1) == str2num(Year));
var5 = var3(var4,:);
var6 = unique(var5(:,2));
var7 = cellstr(num2str(var6));
set(handles.MonthPopUp,'String',var7);

% --- Executes during object creation, after setting all properties.
function YearPopUp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YearPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in MonthPopUp.

```

```

function MonthPopUp_Callback(hObject, eventdata, handles)
% hObject      handle to MonthPopUp (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns MonthPopUp
contents as cell array
%           contents{get(hObject,'Value')} returns selected item from
MonthPopUp
var1=get(handles.MonthPopUp,'String');
var2=get(handles.MonthPopUp,'Value');
Month=var1{var2};
handles.Month = Month;
guidata(hObject,handles)

var3=get(handles.YearPopUp,'String');
var4=get(handles.YearPopUp,'Value');
Year=var3{var4};

var5 = handles.data;
var6 = find(var5(:,1) == str2num(Year) & var5(:,2) == str2num(Month));
var7 = var5(var6,:);
var8 = cellstr(num2str(unique(var7(:,3))));
set(handles.DayPopUp,'String',var8);

% --- Executes during object creation, after setting all properties.
function MonthPopUp_CreateFcn(hObject, eventdata, handles)
% hObject      handle to MonthPopUp (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%           See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in DayPopUp.
function DayPopUp_Callback(hObject, eventdata, handles)
% hObject      handle to DayPopUp (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns DayPopUp
contents as cell array
%           contents{get(hObject,'Value')} returns selected item from
DayPopUp
var1=get(handles.DayPopUp,'String');
var2=get(handles.DayPopUp,'Value');

```

```

Day=var1{var2};
handles.Day = Day;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function DayPopUp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to DayPopUp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%       str2double(get(hObject,'String')) returns contents of edit15
as a double

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in uipanel3.
function uipanel3_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel3
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none

```

```

was selected
%   NewValue: handle of the currently selected object
% handles      structure with handles and user data (see GUIDATA)
var1 = get(hObject, 'String');
handles.PowerType = var1;
guidata(hObject, handles)
set(handles.edit16, 'String', var1)
set(handles.edit17, 'String', var1)
set(handles.edit18, 'String', var1)
set(handles.edit19, 'String', strcat(var1, 'h'))
set(handles.edit20, 'String', strcat('$ / ', var1, 'h'))

function edit16_Callback(hObject, eventdata, handles)
% hObject      handle to edit16 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit16 as text
%          str2double(get(hObject, 'String')) returns contents of edit16
as a double

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit16 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject      handle to edit18 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit18 as text
%          str2double(get(hObject, 'String')) returns contents of edit18
as a double

% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)

```

```

% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit17 as text
%         str2double(get(hObject,'String')) returns contents of edit17
as a double

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%         str2double(get(hObject,'String')) returns contents of edit19
as a double

% --- Executes during object creation, after setting all properties.

```

```

function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as text
%       str2double(get(hObject,'String')) returns contents of edit20
as a double

% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

var2 = handles.var2;
Minimum = handles.Minimum;
Average = handles.Average;

```

```

Maximum = handles.Maximum;
Demand = handles.Demand;
Power_Type = handles.Power_Type;

Fig = figure;
plot(var2,Demand);

plot(var2 ./ 60,Demand,'b','Marker','.');hold on
plot(var2 ./ 60, repmat(Minimum,1,numel(var2')), 'r');hold on
plot(var2 ./ 60, repmat(Average,1,numel(var2')), 'g');hold on
plot(var2 ./ 60, repmat(Maximum,1,numel(var2')), 'k');hold on

h = legend('Demand','Daily Base Demand','Average Demand','Maximum
Demand');
set(h, 'Location', 'SouthOutside')
xlabel('Time (hr)')
ylabel(strcat('Demand (',Power_Type,')'))
var3 = get(gca,'XTick');
var4 = get(gca,'YTick');
set(gca,'YTick',round(linspace(var4(1),var4(end),10)))
set(gca,'XTick',1:1:24)
grid on

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'eps')
saveas(gcf,answer2{1},'fig')
close(Fig)

```

Demand Aberration Tool

```
function varargout = Daily_Abberations(varargin)
%DAILY_ABBERATIONS M-file for Daily_Abberations.fig
%   DAILY_ABBERATIONS, by itself, creates a new DAILY_ABBERATIONS or
raises the existing
%   singleton*.
%
%   H = DAILY_ABBERATIONS returns the handle to a new
DAILY_ABBERATIONS or the handle to
%   the existing singleton*.
%
%   DAILY_ABBERATIONS('Property','Value',...) creates a new
DAILY_ABBERATIONS using the
%   given property value pairs. Unrecognized properties are passed
via
%   varargin to Daily_Abberations_OpeningFcn. This calling syntax
produces a
%   warning when there is an existing singleton*.
%
%   DAILY_ABBERATIONS('CALLBACK') and
DAILY_ABBERATIONS('CALLBACK',hObject,...) call the
%   local function named CALLBACK in DAILY_ABBERATIONS.M with the
given input
%   arguments.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Daily_Abberations

% Last Modified by GUIDE v2.5 14-Jul-2014 16:43:55

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Daily_Abberations_OpeningFcn, ...
                  'gui_OutputFcn',  @Daily_Abberations_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```

end
% End initialization code - DO NOT EDIT

% --- Executes just before Daily_Abberations is made visible.
function Daily_Abberations_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% varargin    unrecognized PropertyName/PropertyValue pairs from the
%             command line (see VARARGIN)

% Choose default command line output for Daily_Abberations
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Daily_Abberations wait for user response (see UIRESUME)
% uiwait(handles.figure1);

guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;
guidata(hObject, handles);

%Set Intital Years
Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;

% var1 = get(handles.edit1, 'String');
var2 = get(handles.edit2, 'String');
var3 = get(handles.edit10, 'String');
var4 = get(handles.edit11, 'String');
% handles.Demand_Cost = var1;
handles.Consumption_Cost = var2;
handles.gamma1 = var3;
handles.gamma2 = var4;
guidata(hObject, handles)

%Populating Days PopUp
var7 = str2num(handles.Year);
var8 = find(data(:,1) == var7);
var9 = data(var8, :);
var10 = unique(var9(:,5));

```

```

var11 = cellstr(num2str(var10));
set(handles.popupmenu2, 'String', var11);
guidata(hObject, handles)

%Setting Days Handle
var12 = get(handles.popupmenu2, 'String');
var13 = get(handles.popupmenu2, 'Value');
Day = var12{var13};
handles.WeekDay = Day;

handles.PowerType = 'kW';
guidata(hObject, handles)

% --- Outputs from this function are returned to the command line.
function varargout = Daily_Abberations_OutputFcn(hObject, eventdata,
handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject      handle to figure
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject      handle to the selected object in uipanel1
% eventdata    structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue:  handle of the previously selected object or empty if none
was selected
%   NewValue:  handle of the currently selected object
% handles      structure with handles and user data (see GUIDATA)
var1 = get(hObject, 'String');
handles.PowerType = var1;
set(handles.edit19, 'String', var1)
set(handles.edit20, 'String', var1)
set(handles.edit21, 'String', var1)
set(handles.edit22, 'String', var1)
set(handles.edit23, 'String', var1)
set(handles.edit24, 'String', var1)
set(handles.edit25, 'String', var1)

guidata(hObject, handles)

set(handles.edit26, 'String', strcat('$ / ', ' ', var1, 'h'))
set(handles.edit27, 'String', strcat(var1, 'h'))
set(handles.edit28, 'String', strcat(var1, 'h'))

% --- Executes on selection change in popupmenu3.

```

```

function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu3

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu4.
function popupmenu4_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu4
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu4

% --- Executes during object creation, after setting all properties.
function popupmenu4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject      handle to pushbutton1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
try

dlg = ProgressDialog( ...
'StatusMessage', 'Working...', ...
'Indeterminate', true);

data = handles.data;
Year = str2num(handles.Year);
day = str2num(handles.WeekDay);
gamma1 = str2num(handles.gamma1);
gamma2 = str2num(handles.gamma2);
Consumption_Cost = str2num(handles.Consumption_Cost);
Power_Type = handles.PowerType;

var1 = find(data(:,1) == Year);
var2 = data(var1,:);
var3 = unique(var2(:,5));

for i = 1:length(var3(:,1))%Weekday
    var4 = var3(i);
    var5 = find(var2(:,5) == var4);
    var6 = var2(var5,:);
    var7 = unique(var6(:,2));
    for j = 1:length(var7(:,1))%Month
        var8 = var7(j);
        var9 = find(var6(:,2) == var8);
        var10 = var6(var9,:);
        var11 = unique(var10(:,3));
        var200{i}(1:length(var11),j) = var11;
        for k = 1:length(var11(:,1))%Day
            var12 = var11(k);
            var13 = find(var10(:,3) == var12);
            var100{1,j}(1:length(var13),k) = var10(var13,end);
            var102{1,j}(1:length(var13),k) = var10(var13,end-2);
            var104{1,j}(1,k) = datenum([Year,var8,var12]);
        end
    end

var101{i} = cell2mat(var100);
var103{i} = cell2mat(var102);
var105{i} = cell2mat(var104);
var15 = var101;%demand
var53 = var103;%minutes
var22 = var105;%datenum

```

```

clear var100 var102 var104
end

%%Finding Average Profile
for i = 1:length(var15)
    var16 = var15{i};
    var54 = var53{i};
    var17{i} = mean(var16,2);
    Average_Demand(i) = mean(var17{i});
    Average_Consumption(i) = trapz(var54(:,1),var17{i}) * (1 / 60);
    Average_Consumption_Cost(i) = Average_Consumption(i) *
Consumption_Cost;
end

%%Finding E_Max and E Average
for i = 1:length(var15)
    var18 = var17{i};%Average profile
    var19 = var15{i};%Demand Profile
    for j = 1:length(var19(1,:))
        var29{i}(:,j) = var19(:,j) - var18;
    end
    for k = 1:length(var29{i}(1,:))
        E_Max{i}(1,k) = norm(var29{i}(:,k),1);%{day}(1,difference in
peak from average peak)
        E_Avg{i}(1,k) = norm(var29{i}(:,k),2);%{day}(1,difference in
profile from overage profile)
    end
    Std_Dev_2Norm(i) = std(E_Avg{i});
    Std_Dev_1Norm(i) = std(E_Max{i});
    mean_E_max(i) = mean(E_Max{i});
    mean_E_avg(i) = mean(E_Avg{i});
end

%%Comparing 1 and 2 norm to multiplicate of standard deviation to find
out
%%which ones fall out of standard deviation
for i = 1:length(var15)
    var23 = gamma1 * Std_Dev_2Norm(i);
    var27 = E_Avg{i};
    var32 = mean_E_avg(i);
    var33 = gamma2 * Std_Dev_1Norm(i);
    var34 = E_Max{i};
    var35 = mean_E_max(i);
    for j = 1: numel(var27)
        var25 = var27(j);%2-norm
        var28 = var32 + var23;%2-norm +1 SD
        var31 = var32 - var23;%2-Norm -1SD
        var36 = var34(j);
        var37 = var35 + var33;
        var38 = var35 - var33;
        if var25 > var28 || var25 < var31
            var26{i}(1,j) = 1;%positive for 2-norm being out of bounds
        else

```

```

        var26{i}(1,j) = 0;
    end

    if var36 > var37 || var36 < var38
        var39{i}(1,j) = 1;%positive for 1-norm being out of bounds
    else
        var39{i}(1,j) = 0;
    end
end
end

for i = 1:length(var15)
    var106 = var26{i};
    var107 = var22{i};
    var108 = find(var106 == 1);
    var109{i} = datestr(var107(var108));%{day of the week}(dates with 2
norm out of SD)
    var110 = var39{i};
    var111 = find(var110 == 1);
    var112{i} = datestr(var107(var111));%{day of the week}(dates with 1
norm out of SD)
end

for i = 1:length(var109{day}(:,1))
    var113{1,i} = var109{day}(i,:);%2-Norm Dates
end

for i = 1:length(var112{day}(:,1))
    var114{1,i} = var112{day}(i,:);%1-Norm Dates
end

set(handles.edit3, 'String', num2str(Average_Demand(day)))
set(handles.edit4, 'String', num2str(Average_Consumption(day)))
set(handles.edit5, 'String', num2str(Average_Consumption_Cost(day)))
set(handles.edit6, 'String', num2str(mean_E_avg(day)))
set(handles.edit7, 'String', num2str(Std_Dev_2Norm(day)))
set(handles.edit8, 'String', num2str(mean_E_max(day)))
set(handles.edit9, 'String', num2str(Std_Dev_1Norm(day)))

set(handles.popupmenu3, 'String', var113)
set(handles.popupmenu3, 'Value', length(var113))

set(handles.popupmenu4, 'String', var114)
set(handles.popupmenu4, 'Value', length(var114))

figure(1)
var61 = mean_E_max(day);
var62 = repmat(var61,1,numel(E_Max{day}));

```

```

var63 = repmat((var61 + gamma2 *
Std_Dev_1Norm(day)),1,numel(E_Max{day}));
var64 = repmat((var61 - gamma2 *
Std_Dev_1Norm(day)),1,numel(E_Max{day}));
scatter(1:numel(E_Max{day}),E_Max{day},'fill');hold on
plot(1:numel(E_Max{day}),var62,'k');hold on
plot(1:numel(E_Max{day}),var63,'r');hold on
plot(1:numel(E_Max{day}),var64,'g')

xlabel('Time (Day of Week)')
ylabel(strcat('1-Norm (',Power_Type,')'))
h1 = legend('1-Norm','Average','+1 Sigma','-1 Sigma');
set(h1, 'Location', 'SouthOutside')
h4 = get(gca,'Xtick');
set(gca,'XTick',h4(1):2:h4(end))
h2 = get(gca,'YTick');
set(gca,'YTick',linspace(h2(1),h2(end),10))
h3 = get(gca,'YTick');
set(gca,'YTick',round(h3))
set(gca, 'YTickMode', 'manual')
set(gca, 'YTickLabel', num2str(get(gca, 'YTick')))

figure(2)
var61 = mean_E_avg(day);
var62 = repmat(var61,1,numel(E_Avg{day}));
var63 = repmat((var61 + gamma1 *
Std_Dev_2Norm(day)),1,numel(E_Avg{day}));
var64 = repmat((var61 - gamma1 *
Std_Dev_2Norm(day)),1,numel(E_Avg{day}));
scatter(1:numel(E_Avg{day}),E_Avg{day},'fill');hold on
plot(1:numel(E_Avg{day}),var62,'k');hold on
plot(1:numel(E_Avg{day}),var63,'r');hold on
plot(1:numel(E_Avg{day}),var64,'g')

xlabel('Time (Day of Week)')
ylabel(strcat('2-Norm (',Power_Type,')'))
h1 = legend('2-Norm','Average','+1 Sigma','-1 Sigma');
set(h1, 'Location', 'SouthOutside')
h4 = get(gca,'Xtick');
set(gca,'XTick',h4(1):2:h4(end))
h2 = get(gca,'YTick');
set(gca,'YTick',linspace(h2(1),h2(end),10))
h3 = get(gca,'YTick');
set(gca,'YTick',round(h3))
set(gca, 'YTickMode', 'manual')
set(gca, 'YTickLabel', num2str(get(gca, 'YTick')))

handles.averageprofile = var17{day};
guidata(hObject,handles)

% msgbox('All Done!')
catch errorObj
% If there is a problem, we display the error message

```

```

errorDlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cla(handles.axes1, 'reset')

try
data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var1=get(handles.popupmenu3, 'String');
var2=get(handles.popupmenu3, 'Value');
daystring = var1{var2};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));
var7 = data(var6,end);
var8 = data(var6,end-2);
curveconsumption = trapz(var8 ./ 60,var7);
curveconsumptioncost = curveconsumption *
str2num(handles.Consumption_Cost);
set(handles.edit12, 'String', curveconsumption);
set(handles.edit13, 'String', curveconsumptioncost);

timeconvert = var8 ./ 60;
plot(handles.axes1, var8 ./ 60, averageprofile, var8 ./
60, var7, 'Marker', '.');
xlabel(handles.axes1, 'Time (Hour)')
ylabel(handles.axes1, strcat('Demand (', PowerType, ')'))
h1 = legend(handles.axes1, 'Average Demand', daystring);
set(h1, 'Location', 'SouthOutside')
set(handles.axes1, 'XTick', 0:1:24)
h2 = get(handles.axes1, 'YTick');
set(handles.axes1, 'YTick', linspace(h2(1), h2(end), 10))
h3 = get(handles.axes1, 'YTick');
set(handles.axes1, 'YTick', round(h3))
grid(handles.axes1, 'on')

catch errorObj
% If there is a problem, we display the error message
errorDlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

```

```

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

cla(handles.axes2,'reset')
try
data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var1=get(handles.popupmenu4,'String');
var2=get(handles.popupmenu4,'Value');
daystring = var1{var2};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));
var7 = data(var6,end);
var8 = data(var6,end-2);

timeconvert = var8 ./ 60;
plot(handles.axes2,var8 ./ 60,averageprofile,var8 ./
60,var7,'Marker','.');
xlabel(handles.axes2,'Time (Hour)')
ylabel(handles.axes2,strcat('Demand (',PowerType,')'))
h1 = legend(handles.axes2,'Average Demand',daystring);
set(h1, 'Location', 'SouthOutside')
set(handles.axes2,'XTick',0:1:24)
h2 = get(handles.axes2,'YTick');
set(handles.axes2,'YTick',linspace(h2(1),h2(end),10))
h3 = get(handles.axes2,'YTick');
set(handles.axes2,'YTick',round(h3))
grid(handles.axes2,'on')

max(averageprofile)
var9 = abs(max(averageprofile) - max(var7));
set(handles.edit17,'String',var9);

var10 = max(var7);
set(handles.edit16,'String',var10);

var11 = find(data(:,1) == var5(1) & data(:,2) == var5(2));
var12 = max(data(var11,end));

if var12 == var10
    set(handles.edit18,'String','Yes');
else
    set(handles.edit18,'String','No');
end
catch errorObj
% If there is a problem, we display the error message

```

```

errorldg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

```

```

function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit16 as text
%        str2double(get(hObject, 'String')) returns contents of edit16
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

```

```

function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit17 as text
%        str2double(get(hObject, 'String')) returns contents of edit17
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))

```

```

    set(hObject, 'BackgroundColor', 'white');
end

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit12 as text
%        str2double(get(hObject, 'String')) returns contents of edit12
as a double

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit13 as text
%        str2double(get(hObject, 'String')) returns contents of edit13
as a double

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),

```

```

get(0,'defaultUicontrolBackgroundColor')
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject      handle to edit3 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%         str2double(get(hObject,'String')) returns contents of edit3 as
a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit3 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject      handle to edit4 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%         str2double(get(hObject,'String')) returns contents of edit4 as
a double

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit4 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.

```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit5_Callback(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%         str2double(get(hObject,'String')) returns contents of edit5 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit6_Callback(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%         str2double(get(hObject,'String')) returns contents of edit6 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%       str2double(get(hObject,'String')) returns contents of edit7 as
a double

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%       str2double(get(hObject,'String')) returns contents of edit8 as
a double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%       str2double(get(hObject,'String')) returns contents of edit9 as
a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%       contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

```

```

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%       str2double(get(hObject,'String')) returns contents of edit1 as
a double
    var1 = get(hObject,'String');
    Demand_Cost = var1;
    handles.Demand_Cost = Demand_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as
a double
    var1 = get(hObject,'String');
    Consumption_Cost = var1;
    handles.Consumption_Cost = Consumption_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
popupmenu2
var1=get(handles.popupmenu2,'String');
var2=get(handles.popupmenu2,'Value');
WeekDay=var1{var2};
handles.WeekDay = WeekDay;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

```

```

    set(hObject, 'BackgroundColor', 'white');
end

function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit10 as text
%        str2double(get(hObject, 'String')) returns contents of edit10
as a double
    var1 = get(hObject, 'String');
    gamma1 = var1;
    handles.gamma1 = gamma1;
    guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit11 as text
%        str2double(get(hObject, 'String')) returns contents of edit11
as a double
    var1 = get(hObject, 'String');
    gamma2 = var1;
    handles.gamma2 = gamma2;
    guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)

```

```

% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%         str2double(get(hObject,'String')) returns contents of edit14
as a double

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%         str2double(get(hObject,'String')) returns contents of edit15
as a double

% --- Executes during object creation, after setting all properties.

```

```

function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%       str2double(get(hObject,'String')) returns contents of edit18
as a double

% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit24_Callback(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit24 as text
%       str2double(get(hObject,'String')) returns contents of edit24
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit25_Callback(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit25 as text
%       str2double(get(hObject,'String')) returns contents of edit25
as a double

% --- Executes during object creation, after setting all properties.
function edit25_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%       str2double(get(hObject,'String')) returns contents of edit19
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as text
%       str2double(get(hObject,'String')) returns contents of edit20
as a double

% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as text
%       str2double(get(hObject,'String')) returns contents of edit21
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit22 as text
%         str2double(get(hObject,'String')) returns contents of edit22
as a double

% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit23 as text
%         str2double(get(hObject,'String')) returns contents of edit23

```

as a double

```
% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns listbox1
contents as cell array
%       contents{get(hObject,'Value')} returns selected item from
listbox1
```

```
% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit26_Callback(hObject, eventdata, handles)
% hObject    handle to edit26 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```

% Hints: get(hObject,'String') returns contents of edit26 as text
%         str2double(get(hObject,'String')) returns contents of edit26
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit26_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit26 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit28_Callback(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit28 as text
%         str2double(get(hObject,'String')) returns contents of edit28
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit28_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit28 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit27_Callback(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of edit27 as text
%         str2double(get(hObject,'String')) returns contents of edit27
as a double

% --- Executes during object creation, after setting all properties.
function edit27_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit27 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1
data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var1=get(handles.popupmenu3,'String');
var2=get(handles.popupmenu3,'Value');
daystring = var1{var2};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));
var7 = data(var6,end);
var8 = data(var6,end-2);
curveconsumption = trapz(var8 ./ 60,var7);
curveconsumptioncost = curveconsumption *
str2num(handles.Consumption_Cost);
set(handles.edit12,'String',curveconsumption);
set(handles.edit13,'String',curveconsumptioncost);

Fig = figure
timeconvert = var8 ./ 60;
plot(var8 ./ 60,averageprofile,var8 ./ 60,var7,'Marker','.');
xlabel('Time (Hour)')
ylabel(strcat('Demand (',PowerType,')'))
h1 = legend('Average Demand',daystring);

```

```

set(h1, 'Location', 'SouthOutside')
set(gca, 'XTick', 0:1:24)
h2 = get(gca, 'YTick');
set(gca, 'YTick', linspace(h2(1), h2(end), 10))
h3 = get(gca, 'YTick');
set(gca, 'YTick', round(h3))
grid on

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf, answer2{1}, 'epsc')
saveas(gcf, answer2{1}, 'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject, 'Value') returns toggle state of togglebutton2

data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var1=get(handles.popupmenu4, 'String');
var2=get(handles.popupmenu4, 'Value');
daystring = var1{var2};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));
var7 = data(var6, end);
var8 = data(var6, end-2);

Fig = figure;
timeconvert = var8 ./ 60;
plot(var8 ./ 60, averageprofile, var8 ./ 60, var7, 'Marker', '.');
xlabel('Time (Hour)')
ylabel(strcat('Demand (', PowerType, ')'))
h1 = legend('Average Demand', daystring);
set(h1, 'Location', 'SouthOutside')
set(gca, 'XTick', 0:1:24)
h2 = get(gca, 'YTick');
set(gca, 'YTick', linspace(h2(1), h2(end), 10))
h3 = get(gca, 'YTick');
set(gca, 'YTick', round(h3))
grid on

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);

```

```

saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on selection change in listbox2.
function listbox2_Callback(hObject, eventdata, handles)
% hObject    handle to listbox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listbox2
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
listbox2

% --- Executes during object creation, after setting all properties.
function listbox2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton3

data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var9 = get(handles.popupmenu3,'String');

for i = 1:length(var9)
daystring = var9{i,1};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));

```

```

var7 = data(var6,end);
var8 = data(var6,end-2);

temp1(1:length(var8),i) = var8;
temp2(1:length(var7),i) = var7;

timeconvert = var8 ./ 60;
figure(i)
plot(var8 ./ 60,averageprofile(1:length(var8)),var8 ./
60,var7,'Marker','.');
xlabel(gca,'Time (Hour)')
ylabel(gca, strcat('Demand (',PowerType,')'))
h1 = legend(gca,'Average Demand',daystring);
set(h1, 'Location', 'SouthOutside')
set(gca,'XTick',0:1:24)
h2 = get(gca,'YTick');
set(gca,'YTick',linspace(h2(1),h2(end),10))
h3 = get(gca,'YTick');
set(gca,'YTick',round(h3))
grid(gca,'on')
end

% --- Executes on button press in togglebutton4.
function togglebutton4_Callback(hObject, eventdata, handles)
% hObject      handle to togglebutton4 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton4

data = handles.data;
averageprofile = handles.averageprofile;
PowerType = handles.PowerType;

var9 = get(handles.popupmenu4,'String');

for i = 1:length(var9)
daystring = var9{i,1};
var5 = datevec(daystring);
var6 = find(data(:,1) == var5(1) & data(:,2) == var5(2) & data(:,3) ==
var5(3));
var7 = data(var6,end);
var8 = data(var6,end-2);

temp1(1:length(var8),i) = var8;
temp2(1:length(var7),i) = var7;

timeconvert = var8 ./ 60;
figure(i)

```

```

plot(var8 ./ 60,averageprofile(1:length(var8)),var8 ./
60,var7,'Marker','.');
xlabel(gca,'Time (Hour)')
ylabel(gca, strcat('Demand (',PowerType,')'))
h1 = legend(gca,'Average Demand',daystring);
set(h1, 'Location', 'SouthOutside')
set(gca,'XTick',0:1:24)
h2 = get(gca,'YTick');
set(gca,'YTick',linspace(h2(1),h2(end),10))
h3 = get(gca,'YTick');
set(gca,'YTick',round(h3))
grid(gca,'on')

var10 = max(var7);
var11 = find(data(:,1) == var5(1) & data(:,2) == var5(2));
var12 = max(data(var11,end));

if var12 == var10
    set(gca,'Color','r');
else
    set(gca,'Color',[1 1 1]);
end
end

% --- Executes on button press in togglebutton5.
function togglebutton5_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton5
figHandles = get(0,'Children');
close(figHandles)

```

Demand Scheduling Tool

```
function varargout = Demand_Visualization_Demand_Scheduling(varargin)
% DEMAND_VISUALIZATION_DEMAND_SCHEDULING MATLAB code for
Demand_Visualization_Demand_Scheduling.fig
% DEMAND_VISUALIZATION_DEMAND_SCHEDULING, by itself, creates a new
DEMAND_VISUALIZATION_DEMAND_SCHEDULING or raises the existing
% singleton*.
%
% H = DEMAND_VISUALIZATION_DEMAND_SCHEDULING returns the handle to
a new DEMAND_VISUALIZATION_DEMAND_SCHEDULING or the handle to
% the existing singleton*.
%
%
DEMAND_VISUALIZATION_DEMAND_SCHEDULING('CALLBACK', hObject, eventData, handles,...) calls the local
% function named CALLBACK in
DEMAND_VISUALIZATION_DEMAND_SCHEDULING.M with the given input
arguments.
%
% DEMAND_VISUALIZATION_DEMAND_SCHEDULING('Property','Value',...)
creates a new DEMAND_VISUALIZATION_DEMAND_SCHEDULING or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before
Demand_Visualization_Demand_Scheduling_OpeningFcn gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
Demand_Visualization_Demand_Scheduling_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Demand_Visualization_Demand_Scheduling

% Last Modified by GUIDE v2.5 18-Jun-2014 15:29:56

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn', ...
                  'gui_OutputFcn', ...
                  'gui_LayoutFcn', [], ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
```

```

    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Demand_Visualization_Demand_Scheduling is
% made visible.
function Demand_Visualization_Demand_Scheduling_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to
Demand_Visualization_Demand_Scheduling (see VARARGIN)

% Choose default command line output for
Demand_Visualization_Demand_Scheduling
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Demand_Visualization_Demand_Scheduling wait for user
response (see UIRESUME)
% uiwait(handles.figure1);

guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;
guidata(hObject, handles);

%Set Intital Years
Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;

%Setting Demand Costs and Reduction
var1 = get(handles.edit1, 'String');
var2 = get(handles.edit2, 'String');
handles.Demand_Cost = var1;
handles.Reduction = var2;

```

```

handles.PowerType = 'kW';
guidata(hObject,handles)

% --- Outputs from this function are returned to the command line.
function varargout =
Demand_Visualization_Demand_Scheduling_OutputFcn(hObject, eventdata,
handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)
% hObject     handle to edit1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%         str2double(get(hObject,'String')) returns contents of edit1 as
a double
    var1 = get(hObject,'String');
    alpha = var1;
    handles.Demand_Cost = alpha;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject     handle to edit2 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
% str2double(get(hObject,'String')) returns contents of edit2 as
a double
    var1 = get(hObject,'String');
    alpha = var1;
    handles.Reduction = alpha;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject handle to edit2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
% contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.

```

```

%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject     handle to pushbutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
cla(handles.axes1,'reset')
cla(handles.axes2,'reset')
try

    dlg = ProgressDialog( ...
        'StatusMessage', 'Working...', ...
        'Indeterminate', true);

num = handles.data;
Year = str2num(handles.Year);
Demand_Cost = str2num(get(handles.edit1,'String'));
Power_Type = handles.PowerType;
Reduction = str2num(handles.Reduction);

var6 = find(num(:,1) == Year);
var7 = num(var6,2);
var8 = unique(var7);

for i = 1:length(var8)
    Month = var8(i);
    var4 = find(num(:,1) == Year & num(:,2) == Month);
    var5 = num(var4,:);

    Average_Demand(i) = mean(var5(:,end));
    Maximum_Demand(i) = max(var5(:,end));
    Difference(i) = Maximum_Demand(i) - Average_Demand(i);
    New_Peak(i) = (Average_Demand(i) + ((1 - Reduction) *
Difference(i)));
    Savings(i) = Demand_Cost * (Maximum_Demand(i) - New_Peak(i));

    var13 = find(var5(:,end) == Maximum_Demand(i));
    Max_Date(i,:) = var5(var13(1),1:3);
end

%Finding Demand Profiles for Max Days
for i = 1:length(Max_Date(:,1))
    var14 = Max_Date(i,:);
    var15 = find(num(:,1) == var14(1) & num(:,2) == var14(2) & num(:,3)
== var14(3));
    var16{i} = num(var15,:);

```

```

end

%Finding Usage for Max Days
for i = 1:length(var16)
    var31 = var16{i};
    var32(i) = trapz(var31(:,4),var31(:,end)) * (1/60);
    Optimal_Demand(i) = var32(i) / ((var31(end,4) - var31(1,4)) * (1 /
60));
    Optimal_Demand_Cost(i) = Optimal_Demand(i) * Demand_Cost;
end

Total_Proposed_Savings = sum((Maximum_Demand*Demand_Cost) -
(New_Peak*Demand_Cost));
Total_Optimal_Savings = sum((Maximum_Demand*Demand_Cost) -
(Optimal_Demand_Cost));

var35 = cat(1,Maximum_Demand,New_Peak,Optimal_Demand);
bar(handles.axes1,var8',var35','grouped')
xlabel(handles.axes1,'Time (Month)')
ylabel(handles.axes1, strcat('Demand (',Power_Type,')'))
h1 = legend(handles.axes1,'Current Demand','Proposed
Demand','OptimalDemand');
set(h1, 'Location', 'SouthOutside')
var32 = get(handles.axes1,'XTick');
set(handles.axes1,'XTick',1:1:var32(end))
var33 = get(handles.axes1,'YTick');
var34 = fix(linspace(0,var33(end),10));
set(handles.axes1,'YTick',var34)
xlim(handles.axes1,[var8(1)-1,var8(end)+1])
ylim(handles.axes1,[0 var34(end)])
set(handles.axes1,'YGrid','on')

var35 = cat(1,(Maximum_Demand.* Demand_Cost) ./ 10^3,(New_Peak.*
Demand_Cost) ./ 10^3,(Optimal_Demand.* Demand_Cost) ./ 10^3);
bar(handles.axes2,var8',var35','grouped')
xlabel(handles.axes2,'Time (Month)')
ylabel(handles.axes2,'Cost ($) [x 10^3]')
h1 = legend(handles.axes2,'Current Demand Cost','Proposed Demand
Cost','Optimal Demand Cost');
set(h1, 'Location', 'SouthOutside')
var32 = get(handles.axes2,'XTick');
set(handles.axes2,'XTick',1:1:var32(end))
var33 = get(handles.axes2,'YTick');
var34 = fix(linspace(0,var33(end),10));
% set(handles.axes2,'YTick',var34)
xlim(handles.axes2,[var8(1)-1,var8(end)+1])
ylim(handles.axes2,[0 var34(end)])
set(handles.axes2,'YGrid','on')

set(handles.edit3,'String',num2str(fix(sum(Maximum_Demand.*
Demand_Cost))))

```

```

set(handles.edit4,'String',num2str(fix(Total_Proposed_Savings)))
set(handles.edit5,'String',num2str(fix(Total_Optimal_Savings)))

catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj,'extended','hyperlinks','on'),'Error');
end

handles.Maximum_Demand = Maximum_Demand;
handles.Demand_Cost = Demand_Cost;
handles.New_Peak = New_Peak;
handles.Optimal_Demand = Optimal_Demand;
handles.var8 = var8;
handles.Power_Type = Power_Type;

guidata(hObject,handles)

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
var1 = get(hObject,'String');
handles.PowerType = var1;
guidata(hObject,handles)
set(handles.edit6,'String',strcat('$ / ', ' ',var1, ' mo'))

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as
a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%       str2double(get(hObject,'String')) returns contents of edit4 as
a double

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%       str2double(get(hObject,'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%       str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

Maximum_Demand = handles.Maximum_Demand;
New_Peak = handles.New_Peak;
Optimal_Demand = handles.Optimal_Demand;
var8 = handles.var8;
Power_Type = handles.Power_Type;

Fig = figure;
var35 = cat(1,Maximum_Demand,New_Peak,Optimal_Demand);
bar(var8',var35','grouped')

```

```

xlabel('Time (Month)')
ylabel(strcat('Demand (',Power_Type,')'))
h1 = legend('Current Demand','Proposed Demand','OptimalDemand');
set(h1, 'Location', 'SouthOutside')
var32 = get(gca,'XTick');
set(gca,'XTick',1:1:var32(end))
var33 = get(gca,'YTick');
var34 = fix(linspace(0,var33(end),10));
set(gca,'YTick',var34)
xlim([var8(1)-1,var8(end)+1])
ylim([0 var34(end)])
set(gca,'YGrid','on')

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton2

Maximum_Demand = handles.Maximum_Demand;
Demand_Cost = handles.Demand_Cost;
New_Peak = handles.New_Peak;
Optimal_Demand = handles.Optimal_Demand;
var8 = handles.var8;

Fig = figure;
var35 = cat(1,(Maximum_Demand.* Demand_Cost) ./ 10^3,(New_Peak.*
Demand_Cost) ./ 10^3,(Optimal_Demand.* Demand_Cost) ./ 10^3);
bar(var8',var35','grouped')
xlabel('Time (Month)')
ylabel('Cost ($) [x 10^3]')
h1 = legend('Current Demand Cost','Proposed Demand Cost','Optimal
Demand Cost');
set(h1, 'Location', 'SouthOutside')
var32 = get(gca,'XTick');
set(gca,'XTick',1:1:var32(end))
var33 = get(gca,'YTick');
var34 = fix(linspace(0,var33(end),10));
set(gca,'YTick',var34)
xlim([var8(1)-1,var8(end)+1])
ylim([0 var34(end)])
set(gca,'YGrid','on')

```

```
answer1 = inputdlg('FileName?');  
answer2 = cellstr(answer1);  
saveas(gcf,answer2{1},'epsc')  
saveas(gcf,answer2{1},'fig')  
close(Fig)
```

Weather Disaggregation Tool

```
function varargout = Weather_Disaggregation(varargin)
% WEATHER_DISAGGREGATION MATLAB code for Weather_Disaggregation.fig
% WEATHER_DISAGGREGATION, by itself, creates a new
WEATHER_DISAGGREGATION or raises the existing
% singleton*.
%
% H = WEATHER_DISAGGREGATION returns the handle to a new
WEATHER_DISAGGREGATION or the handle to
% the existing singleton*.
%
% WEATHER_DISAGGREGATION('CALLBACK',hObject,eventData,handles,...)
calls the local
% function named CALLBACK in WEATHER_DISAGGREGATION.M with the
given input arguments.
%
% WEATHER_DISAGGREGATION('Property','Value',...) creates a new
WEATHER_DISAGGREGATION or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before Weather_Disaggregation_OpeningFcn gets
called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
Weather_Disaggregation_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Weather_Disaggregation

% Last Modified by GUIDE v2.5 18-Jun-2014 11:34:16

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Weather_Disaggregation_OpeningFcn, ...
                  'gui_OutputFcn',  @Weather_Disaggregation_OutputFcn,
                  ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Weather_Disaggregation is made visible.
function Weather_Disaggregation_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Weather_Disaggregation (see
VARARGIN)

% Choose default command line output for Weather_Disaggregation
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Weather_Disaggregation wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

guidata(hObject, handles);
data = getappdata(0, 'data');
handles.data = data;
guidata(hObject, handles);

guidata(hObject, handles);
weatherdata = getappdata(0, 'weatherdata');
handles.weatherdata = weatherdata;
guidata(hObject, handles);

%Set Intital Years
Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;

var1 = get(handles.edit1, 'String');
var2 = get(handles.edit2, 'String');

```

```

handles.Temperature_Setpoint = var1;
handles.Consumption_Cost = var2;

handles.PowerType = 'kW';
guidata(hObject,handles)

% --- Outputs from this function are returned to the command line.
function varargout = Weather_Disaggregation_OutputFcn(hObject,
eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject      handle to figure
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject      handle to pushbutton1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
cla reset

try

dlg = ProgressDialog( ...
    'StatusMessage', 'Working...', ...
    'Indeterminate', true);

num = handles.weatherdata; % GO TO
"http://cd0.ncdc.noaa.gov/qclcd/QCLCD?prior=N" FOR WEATHER DATA
num2 = handles.data;
setpoint_temp = str2num(handles.Temperature_Setpoint);
Year = str2num(handles.Year);
ConsumptionCost = str2num(handles.Consumption_Cost);
Power_Type = handles.PowerType;

temp1 = num;
temp2 = isnan(temp1);
[a1 a2] = find(temp2 == 1);
num(a1,:)=[];

weather_date = num(:,1);
weather_time = num(:,2);
weather_temperature = num(:,3);

```

```

%% This part orders the weather data
for i = 1:length(weather_date)
    weather_date_temp1 = weather_date(i,1);
    weather_date_temp2 = num2str(weather_date_temp1);
    weather_year(i,1) = str2num(weather_date_temp2(1,1:4));
    weather_month(i,1) = str2num(weather_date_temp2(1,5:6));
    weather_day(i,1) = str2num(weather_date_temp2(1,7:8));
    weather_time_temp1 = weather_time(i,1);
    weather_time_temp1_length = length(num2str(weather_time_temp1));

    switch weather_time_temp1_length
        case 2
            weather_time_hour(i,1) = 0;
            weather_time_min(i,1) = weather_time_temp1;
        case 3
            weather_time_temp2 = num2str(weather_time_temp1);
            weather_time_hour(i,1) = str2num(weather_time_temp2(1,1));
            weather_time_min(i,1) = str2num(weather_time_temp2(1,2:3));
        case 4
            weather_time_temp2 = num2str(weather_time_temp1);
            weather_time_hour(i,1) =
str2num(weather_time_temp2(1,1:2));
            weather_time_min(i,1) = str2num(weather_time_temp2(1,3:4));
    end
end

H2min = (weather_time_hour.*60)+weather_time_min;
weather_map =
cat(2,weather_year,weather_month,weather_day,H2min,weather_temperature)
;%(year,month,day,hour,minute,temperature)

%% Matching Demand with Weather

for i = 1:length(num2(:,1))
    var1 = num2(i,1:3);
    var2 = find(weather_map(:,1) == var1(1) & weather_map(:,2) ==
var1(2) & weather_map(:,3) == var1(3));

    if isempty(var2)==1
        var18 = datenum(weather_map(:,1:3));
        var19 = datenum(var1);
        var20 = abs(var19 - var18);
        var21 = min(var20);
        var22 = find(var20 == var21);
        var23 = var18(var22(1));
        var24 = datevec(var23);
        var25 = var24(1,1:3);
        var2 = find(weather_map(:,1) == var25(1) & weather_map(:,2) ==
var25(2) & weather_map(:,3) == var25(3));
        var3 = weather_map(var2,end - 1);%temp minute
        var10 = weather_map(var2,end);%temp temp
    end
end

```

```

        var4 = num2(i,4);%demand minute
        var5 = abs(var3 - var4);
        [idx1 idx2] = min(var5);
        var8 = var10(idx2);
        var9(i,:) = cat(2,num2(i,:),var8);
    else
        var3 = weather_map(var2,end - 1);%temp minute
        var10 = weather_map(var2,end);%temp temp
        var4 = num2(i,4);%demand minute
        var5 = abs(var3 - var4);
        [idx1 idx2] = min(var5);
        var8 = var10(idx2);
        var9(i,:) = cat(2,num2(i,:),var8);
    end
end

%% Disaggregating Weather

var10 = unique(var9(:,4));

for i = 1:length(var10)
    var11 = var10(i);
    var12 = find(var9(:,1) == Year & var9(:,4) == var11);
    var13 = var9(var12,:);
    var14 = find(var13(:,end) > setpoint_temp);

    if isempty(var14) == 1
        var15(i) = 0;
    else
        var15(i) = mean(var13(var14,end - 1));
    end

    var16 = find(var13(:,end) <= setpoint_temp);

    if isempty(var16) == 1
        var17(i) = 0;
    else
        var17(i) = mean(var13(var16,end - 1));
    end

end

var20 = var10 ./ 60;
h3 = plot(handles.axes1,var10 ./ 60,var15,var10 ./
60,var17,'Marker','.');
grid(handles.axes1,'on');
h4 = plot(handles.axes2,var10 ./ 60,var15 - var17);
grid on;
xlabel(handles.axes1,'Time (hr)')
xlabel(handles.axes2,'Time (hr)')
ylabel(handles.axes1, strcat('Demand (',Power_Type,')'))
ylabel(handles.axes2, strcat('Demand (',Power_Type,')'))

```

```

h1 = legend(handles.axes1, 'Above Setpoint', 'Below Setpoint');
set(h1, 'Location', 'SouthOutside')
h2 = legend(handles.axes2, 'Difference in Demand');
set(h2, 'Location', 'SouthOutside')
var26 = get(handles.axes1, 'XTick');
set(handles.axes1, 'XTick', var26(1):1:var26(end))
set(handles.axes2, 'XTick', var26(1):1:var26(end))
var27 = get(handles.axes1, 'YTick');
set(handles.axes1, 'YTick', fix(linspace(0, var27(end), 10)))
var28 = get(handles.axes2, 'YTick');
set(handles.axes2, 'YTick', fix(linspace(var28(1), var28(end), 10)))
ylim(handles.axes1, [0, var27(end)])

var18 = trapz(var10, var15 - var17) * (1/60);
var19 = trapz(var10, var15 - var17) * ConsumptionCost;

set(handles.edit3, 'String', num2str(fix(var18)))
set(handles.edit4, 'String', num2str(fix(var19)))
set(handles.edit5, 'String', strcat(Power_Type, 'h'))

handles.var10 = var10;
handles.var15 = var15;
handles.var17 = var17;
handles.Power_Type = Power_Type;

guidata(hObject, handles)

catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit3 as text
%        str2double(get(hObject, 'String')) returns contents of edit3 as
a double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),

```

```

get(0,'defaultUicontrolBackgroundColor')
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as
a double

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit1_Callback(hObject, eventdata, handles)
% hObject handle to edit1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
% str2double(get(hObject,'String')) returns contents of edit1 as
a double
    var1 = get(hObject,'String');
    alpha = var1;
    handles.Temperature_Setpoint = alpha;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject handle to edit1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject handle to edit2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
% str2double(get(hObject,'String')) returns contents of edit2 as
a double

```

```

    var1 = get(hObject, 'String');
    alpha = var1;
    handles.Consumption_Cost = alpha;
    guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
var1 = get(hObject, 'String');
handles.PowerType = var1;
guidata(hObject, handles)

set(handles.edit6, 'String', strcat('$ / ', var1, 'h'))
set(handles.edit5, 'String', strcat(var1, 'h'))

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit5 as text
%       str2double(get(hObject, 'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)

```

```

% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%         str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

var10 = handles.var10;
var15 = handles.var15;
var17 = handles.var17;
Power_Type = handles.Power_Type;

```

```

Fig = figure;

plot(var10 ./ 60,var15,var10 ./ 60,var17,'Marker','.');
grid on;
xlabel('Time (hr)')
ylabel(strcat('Demand (',Power_Type,')'))
h1 = legend('Above Setpoint','Below Setpoint');
set(h1, 'Location', 'SouthOutside')
var26 = get(gca,'XTick');
set(gca,'XTick',var26(1):1:var26(end))
var27 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(0,var27(end),10)))
ylim([0,var27(end)])

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton2

var10 = handles.var10;
var15 = handles.var15;
var17 = handles.var17;
Power_Type = handles.Power_Type;

Fig = figure;

plot(var10 ./ 60,var15 - var17);
grid on;
xlabel('Time (hr)')
ylabel(strcat('Demand (',Power_Type,')'))
h2 = legend('Difference in Demand');
set(h2, 'Location', 'SouthOutside')
var26 = get(gca,'XTick');
set(gca,'XTick',var26(1):1:var26(end))
var28 = get(gca,'YTick');
set(gca,'YTick',fix(linspace(var28(1),var28(end),10)))

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);

```

```
saveas(gcf,answer2{1},'epsc')  
saveas(gcf,answer2{1},'fig')  
close(Fig)
```

Photovoltaic Analysis Tool

```
function varargout = Photovoltaic_Analysis(varargin)
% PHOTOVOLTAIC_ANALYSIS MATLAB code for Photovoltaic_Analysis.fig
% PHOTOVOLTAIC_ANALYSIS, by itself, creates a new
% PHOTOVOLTAIC_ANALYSIS or raises the existing
% singleton*.
%
% H = PHOTOVOLTAIC_ANALYSIS returns the handle to a new
% PHOTOVOLTAIC_ANALYSIS or the handle to
% the existing singleton*.
%
% PHOTOVOLTAIC_ANALYSIS('CALLBACK',hObject,eventData,handles,...)
% calls the local
% function named CALLBACK in PHOTOVOLTAIC_ANALYSIS.M with the
% given input arguments.
%
% PHOTOVOLTAIC_ANALYSIS('Property','Value',...) creates a new
% PHOTOVOLTAIC_ANALYSIS or raises the
% existing singleton*. Starting from the left, property value
% pairs are
% applied to the GUI before Photovoltaic_Analysis_OpeningFcn gets
% called. An
% unrecognized property name or invalid value makes property
% application
% stop. All inputs are passed to Photovoltaic_Analysis_OpeningFcn
% via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
% one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
% Photovoltaic_Analysis

% Last Modified by GUIDE v2.5 03-Nov-2014 11:33:15

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn', @Photovoltaic_Analysis_OpeningFcn,
                  ...
                  'gui_OutputFcn', @Photovoltaic_Analysis_OutputFcn,
                  ...
                  'gui_LayoutFcn', [] , ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```

if narginout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Photovoltaic_Analysis is made visible.
function Photovoltaic_Analysis_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Photovoltaic_Analysis (see
VARARGIN)

% Choose default command line output for Photovoltaic_Analysis
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

data_num = getappdata(0, 'data_num');
handles.data_num = data_num;

data_raw = getappdata(0, 'data_raw');
handles.data_raw = data_raw;

solardata_num = getappdata(0, 'tmy3data_num');
handles.solardata_num = solardata_num;

solardata_raw = getappdata(0, 'tmy3data_raw');
handles.solardata_raw = solardata_raw;

var1 = get(handles.edit1, 'String');
handles.Demand_Cost = var1;
guidata(hObject, handles)

var2 = get(handles.edit2, 'String');
handles.Consumption_Cost = var2;
guidata(hObject, handles)

var3 = get(handles.edit3, 'String');
handles.reflectivity = var3;
guidata(hObject, handles)

var4 = get(handles.edit4, 'String');
handles.Panel_Efficiency = var4;

```

```

guidata(hObject,handles)

var5 = get(handles.edit5,'String');
handles.System_Size = var5;
guidata(hObject,handles)

var6 = get(handles.edit6,'String');
handles.Array_Cost = var6;
guidata(hObject,handles)

var7 = get(handles.edit7,'String');
handles.chosen_array_size = var7;
guidata(hObject,handles)

var8 = get(handles.edit8,'String');
handles.Consumption_Sold = var8;
guidata(hObject,handles)

var9 = get(handles.edit18,'String');
handles.panel_constant = var9;
guidata(hObject,handles)

Years = cellstr(num2str(unique(data_num(:,1))));
set(handles.popupmenu1,'String',Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

handles.Power_Type = 'kW';

guidata(hObject,handles)

% UIWAIT makes Photovoltaic_Analysis wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Photovoltaic_Analysis_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

```

```

function edit1_Callback(hObject, eventdata, handles)
% hObject      handle to edit1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%         str2double(get(hObject,'String')) returns contents of edit1 as
a double
var1 = get(hObject,'String');
Demand_Cost = var1;
handles.Demand_Cost = Demand_Cost;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as
a double
    var1 = get(hObject,'String');
    Consumption_Cost = var1;
    handles.Consumption_Cost = Consumption_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns

```

```

called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%       str2double(get(hObject,'String')) returns contents of edit3 as
a double
var1 = get(hObject,'String');
reflectivity = var1;
handles.reflectivity = reflectivity;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%       contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');

```

```

var2=get(handles.popupmenu1,'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as
a double
var1 = get(hObject,'String');
Panel_Efficiency = var1;
handles.Panel_Efficiency = Panel_Efficiency;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

function edit5_Callback(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%         str2double(get(hObject,'String')) returns contents of edit5 as
a double
var1 = get(hObject,'String');
System_Size = var1;
handles.System_Size = System_Size;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit5 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%         str2double(get(hObject,'String')) returns contents of edit6 as
a double
var1 = get(hObject,'String');
Array_Cost = var1;
handles.Array_Cost = Array_Cost;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
% hObject     handle to edit7 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%     str2double(get(hObject,'String')) returns contents of edit7 as
a double
var1 = get(hObject,'String');
chosen_array_size = var1;
handles.chosen_array_size = chosen_array_size;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit7 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
var1 = get(hObject,'String');
handles.Power_Type = var1;
guidata(hObject,handles);

set(handles.edit10,'String',strcat('$ / ', ' ',var1,' mo'))
set(handles.edit11,'String',strcat('$ / ', ' ',var1,' h'))
% hObject     handle to the selected object in uipanel1
% eventdata   structure with the following fields (see UIBUTTONGROUP)
%     EventName: string 'SelectionChanged' (read only)
%     OldValue: handle of the previously selected object or empty if none

```

```

was selected
%   NewValue: handle of the currently selected object
%   handles   structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject     handle to pushbutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
cla reset

try

dlg = ProgressDialog( ...
    'StatusMessage', 'Working...', ...
    'Indeterminate', true);

%% Input Variables
% [num,txt,row] = handles.solardata;
%http://rredc.nrel.gov/solar/old_data/nsrdb/1991-
2005/tmy3/by_state_and_city.html#T
num = handles.solardata_num;
row = handles.solardata_row;
% [num2,txt2,row2] = handles.data;
num2 = handles.data_num;
row2 = handles.data_row;
phi = cell2mat(row(1,5)); %latitude angle of city in degrees
Tilt = phi; %degrees
reflectivity = str2num(handles.reflectivity);
Demand_Cost = str2num(handles.Demand_Cost);% $ / kW
Consumption_Cost = str2num(handles.Consumption_Cost);% $ / kWh
Consumption_Sold = str2num(handles.Consumption_Sold); % $ / kWh
System_Size = str2num(handles.System_Size);
Array_Size = 0:1:System_Size; %m^2
Array_Cost = str2num(handles.Array_Cost); %Dollars per kW;
Panel_Efficiency = str2num(handles.Panel_Efficiency);
Year = str2num(handles.Year);
chosen_array_size = str2num(handles.chosen_array_size); %m^2
Power_Type = handles.Power_Type;
Panel_Constant = str2num(handles.panel_constant); % kW/m^2

% removing zero values
var200 = find(num2(:,end) == 0);
num2(var200,:) = [];

%% Calculating Position of the Sun in the Sky

N = 1:1:365; % Day Number (N = 1 on Jan 1st; J = 365 on Dec. 31st)
t_s = linspace(1,24,24);%(t_s = 1 being 1 AM)

dec = asind(0.39795 .* cosd(0.98563 * (N - 173))); %Declination Angle

```

```

in Degrees
omega = 15.*(t_s-12); %Hour Angle in Degrees
sunset_angle = acosd(-1 .* tand(phi) .* tand(dec));
conversion_matrix = [cosd(phi),0,sind(phi);0,1,0;-
sind(phi),0,cosd(phi)];

[a1 a2] = size(dec);
[a3 a4] = size(omega);

for i = 1:a2
    for j = 1:a4
        %Suns position vector in a 3D Cartesian Grid relative to
equator
        S_m(i,j) = cosd(dec(i)) * cosd(omega(j));
        S_e(i,j) = cosd(dec(i)) * sind(omega(j));
        S_p(i,j) = sind(dec(i));
        %Conversion from axis relative at the equator to axis relative
to
        %another point on earth
        City{i,j} = conversion_matrix * [S_m(i,j);S_e(i,j);S_p(i,j)];
    end
end

for i = 1:a2
    for j = 1:a4
        temp1 = City{i,j};
        temp2 = temp1(1,1);
        temp3 = temp1(2,1);
        temp4 = temp1(3,1);
        %Finding Azimuth and Solar Elevation Angles From Cartesian
%Translational Points
        alpha(i,j) = asind(temp2); %Solar Elevation
        A1(i,j) = acosd(temp4 / cosd(alpha(i,j)));
        A2(i,j) = asind(temp3 / cosd(alpha(i,j)));
        Angles{i,j} = [alpha(i,j);A1(i,j)]; %[solar altitude,Azimuth];
Rows are the days and Columns are the hours
        Cosine_Theta_i(i,j) = sind(alpha(i,j)) * cosd(Tilt) +
cosd(alpha(i,j)) * sind(Tilt) * cosd(180 - A1(i,j));
    end
end

var1 = cell2mat(raw(3:end,1)) + cell2mat(raw(3:end,2)) + 693960;
var2 = datevec(var1);
var3 = cell2mat(raw(3:end,5)) .* (1 / 1000);%global horizontal; 1 /
1000 to convert from W to kW
var5 = cell2mat(raw(3:end,8)) .* (1 / 1000);%beam

var6 = var3 - var5 .*
reshape(transpose(Cosine_Theta_i),numel(Cosine_Theta_i),1);
var4 = var3 .* reflectivity .* ((1 - cosd(Tilt)) / (2)) + var5 .*
reshape(transpose(Cosine_Theta_i),numel(Cosine_Theta_i),1) + var5 .*
((1 + cosd(Tilt)) / (2));
var7 = cat(2,var2(:,2:end),var4);

```

```

var8 = unique(var7(:,1));

for i = 1:length(var8)
    var9 = var8(i);
    var10 = find(var7(:,1) == var9);
    var11(1:length(var10),i) = var7(var10,end) .* Panel_Efficiency;
    Consumption(1,i) = trapz(1:length(var11(:,i)),var11(:,i));%by month
    Demand(1,i) = mean(nonzeros(var11));
end;

%% Matching Demand with Radiation

var12 = var7(:,3) * 60;
var13 = cat(2,var7(:,1:2),var12,var7(:,6));

for i = 1:length(num2(:,1))
    var14 = num2(i,:);
    var15 = find(var13(:,1) == var14(2) & var13(:,2) == var14(3));
    var16 = var13(var15,:);
    val = var14(4);
    vec = var16(:,3);
    tmp = abs(vec-val);
    [idx idx] = min(tmp);
    var17(i,1) = var16(idx,end);
end

var18 = cat(2,num2,var17); %solar data matched with demand data

var100 = find(var18(:,1) == Year);
var101 = var18(var100,:);
var19 = unique(var101(:,2));

for i = 1:length(var19)
    var20 = var19(i);
    var21 = find(var18(:,1) == Year & var18(:,2) == var20);
    [var22 var23] = max(var18(var21,end - 1));
    var24(1,i) = var22;
    var24(2,i) = var18(var23,end);
    Facility_Demand(i) = var22;
    Facility_Consumption(i) = trapz(var18(var21,4),var18(var21,6)) * (1
/ 60);
    Facility_Cost(i) = var22 * Demand_Cost +
trapz(var18(var21,4),var18(var21,6)) * (1 / 60) * Consumption_Cost; %$
end

for i = 1:length(Consumption)
    var32(:,i) = Consumption(i) .* Array_Size;
    var33(:,i) = Demand(i) .* Array_Size;
end

for i = 1:length(Facility_Consumption)

```

```

var34 = var32(:,i);
var35 = Facility_Consumption(i);
var36(:,i) = repmat(var35,length(var34),1) - var34;%Facility
Consumption with array [Array Size,Month]

var37 = var33(:,i);
var38 = Facility_Demand(i);
var39(:,i) = repmat(var38,length(var37),1) - var37; %Facility
Demand with array [Array Size,Month]
end

for i = 1:length(var36(:,1))
var40 = var36(i,:);
var41 = find(var40 > 0);
var42(i,1) = sum(var40(var41));

var43 = var39(i,:);
var44 = find(var43 > 0);
var45(i,1) = sum(var43(var44));
end

var46 = var42 .* Consumption_Cost; %Facility Consumption Cost with
Array
var47 = var45 .* Demand_Cost; %Facility Demand Cost with Array
var48 = sum(Facility_Consumption) * Consumption_Cost; %Facility
consumption Cost without Array
var49 = sum(Facility_Demand) * Demand_Cost; %Facility demand Cost
without array
var50 = var48 - var46; %Facility Consumption Savings
var51 = var49 - var47; %Facility Demand Savings
var52 = var50 + var51; %Total Facility Savings

plot(handles.axes1,Array_Size ./ (10^3),var52 ./ (10^3))
xlabel(handles.axes1,'PV Array Size (m^2) [x 10^3]')
ylabel(handles.axes1,'Total Savings ($) [x 10^3]')
title(handles.axes1,'Photovoltaic System Savings Curve')
grid(handles.axes1,'on')

Total_Array_Cost = Array_Size .* Array_Cost * Panel_Constant;

plot(handles.axes2,Array_Size ./ 10^3,Total_Array_Cost ./ 10^6)
xlabel(handles.axes2,'Array Size (m^2) [x 10^3]')
ylabel(handles.axes2,'Array Cost ($) [x 10^6]')
title(handles.axes2,'Photovoltaic System Cost Curve')
grid(handles.axes2,'on')

Payback = Total_Array_Cost' ./ var52;

plot(handles.axes3,Array_Size ./ 10^3,Payback)
xlabel(handles.axes3,'Array Size (m^2) [x 10^3]')
ylabel(handles.axes3,'Payback (yr)')
title(handles.axes3,'Photovoltaic Payback Curve')

```

```

grid(handles.axes3, 'on')

% var60 = unique(var18(:,2));%find months in year
var60 = var19;
for i = 1:length(var60);
    var61 = var60(i);%isolate month
    var62 = find(var18(:,1) == Year & var18(:,2) == var61);%find
entries with month and year
    var63 = var18(var62,:);%extract entries with month and year
    var64 = unique(var63(:,3));%find days in extracted month and year
    var79(i) = max(var63(:,end-1));
    [C1 I1] = max(var63(:,end-1));
    var80(i) = var63(I1,end);
    for j = 1:length(var64)
        var65 = var64(j);%isolate day
        var66 = find(var63(:,3) == var65);%extract month day and year
entries
        var67{i,1}(j,1) = max(var63(var66,end-1));%save the facility
demand
        [C2,I2] = max(var63(var66,end-1));
        var68{i,1}(j,1) = var63(I2,end);%save the solar radiation
        var71{i,1}(j,1) = var63(I2,5);
        var83{i,1}(j,1) = trapz(var63(var66,4),var63(var66,end - 1)) ./
60; %save facility consumption
        var84{i,1}(j,1) = trapz(var63(var66,4),var63(var66,end)) ./
60;%save solar energy
    end
end

var69 = cell2mat(var67);%facility demand max
var70 = cell2mat(var68);%solar max
var72 = cell2mat(var71);%weekday of max
var85 = cell2mat(var83);%daily facility consumption
var86 = cell2mat(var84);%daily solar consumption

var55 = var70 .* choosen_array_size;
var57 = var69 - var55;

var58 = find(var57 > 0);
percent_notmet = (length(var58) / length(var57));
percent_met = 1 - percent_notmet;

figure(4)
h = pie([percent_notmet percent_met],[1 1]);
textObjs = findobj(h, 'Type', 'text');
oldStr = get(textObjs, {'String'});
val = get(textObjs, {'Extent'});
oldExt = cat(1, val{:});
Names = {'Demand Not Met: '; 'Demand Met: '};
newStr = strcat(Names, oldStr);
set(textObjs, {'String'}, newStr)

```

```

vall = get(textObjs, {'Extent'});
newExt = cat(1, vall{:});
offset = sign(oldExt(:,1)).*(newExt(:,3)-oldExt(:,3))/2;
pos = get(textObjs, {'Position'});
textPos = cat(1, pos{:});
textPos(:,1) = textPos(:,1)+offset;
set(textObjs,{'Position'},num2cell(textPos,[3,2]))
title('Daily Maximum Demand Supported by Solar Power')

figure(5)
area(1:length(var69),[var69 ./ 10^3,var55 ./ 10^3])
legend('Facility Demand',strcat('Solar Power:
',num2str(choosen_array_size ./ 1000),' m^2 [x 10^3]',' Array Size'))
xlabel('Day')
ylabel(strcat('Demand (',Power_Type,') [x 10^3]'))
title(strcat('Daily Demand Curve for',num2str(Year)))
%

var96 = trapz(1:length(var69),var69) * (24);
var97 = trapz(1:length(var55),var55) * (24);
var98 = var97 - var96;

if var98 > 0
    var99 = var98 * Consumption_Sold;
else
    var99 = 0;
end

var131 = unique(var101(:,5));
for i = 1:length(var131)
    var132 = var131(i);
    var133 = var101(find(var101(:,5) == var132),:);
    var134 = cat(2,var133(:,1:end-1),var133(:,end) .*
choosen_array_size);
    weekday_met(i) = length(find((var134(:,end) - var134(:,end-1)) >
0)) / length(var134(:,end));
    weekday_not_met(i) = 1 - weekday_met(i);
end

var78 = [weekday_met;weekday_not_met];

figure(6)
bar(var131,var78' .* 100,'grouped')
xlabel('Weekday');
ylabel('Percent')
legend('Demand Met','Demand Not Met')
title(strcat('Weekday Demand for',num2str(Year)))
set(gca,'ygrid','on')

```

```

figure(7)
var81 = var80 .* choosen_array_size;
var82 = [var79;var81];
bar(var60,var82' ./ 1000,'grouped')
xlabel('Month')
ylabel(strcat('Demand (',Power_Type,') [x 10^3]'))
legend('Facility Demand',strcat('Solar Power:
',num2str(choosen_array_size ./ 1000),' m^2 [x 10^3]',' Array Size'))
title(strcat('Monthly Maximum Demand for',num2str(Year)))
set(gca,'ygrid','on')

var87 = var86 .* choosen_array_size;
figure(8)
area(1:length(var69),[var85 ./ 10^3,var87 ./ 10^3])
legend('Facility Consumption',strcat('Solar Energy:
',num2str(choosen_array_size ./ 1000),' m^2 [x 10^3]',' Array Size'))
xlabel('Day')
ylabel(strcat('Consumption (',Power_Type,'h)', ' [x 10^3]'))
title(strcat('Daily Consumption Curve for',num2str(Year)))

var88 = var85 - var87;
var89 = find(var88 > 0);
consumption_not_met = length(var89) / length(var88);
consumption_met = 1 - consumption_not_met;

figure(9)
h = pie([consumption_not_met consumption_met],[1 1]);
textObjs = findobj(h,'Type','text');
oldStr = get(textObjs,{'String'});
val = get(textObjs,{'Extent'});
oldExt = cat(1,val{:});
Names = {'Consumption Not Met: ','Consumption Met: '};
newStr = strcat(Names,oldStr);
set(textObjs,{'String'},newStr)
val1 = get(textObjs, {'Extent'});
newExt = cat(1, val1{:});
offset = sign(oldExt(:,1)).*(newExt(:,3)-oldExt(:,3))/2;
pos = get(textObjs, {'Position'});
textPos = cat(1, pos{:});
textPos(:,1) = textPos(:,1)+offset;
set(textObjs,{'Position'},num2cell(textPos,[3,2]))
title('Daily Maximum Consumption Supported by Solar Power')

for i = 1:length(var83)
    var90 = var83{i};
    var91(i) = sum(var90) ./ 10^3;
    var92 = var84{i};
    var93(i) = sum(var92) ./ 10^3;
end

var94 = var93 .* choosen_array_size;

```

```

figure(10)
bar(var60,[var91 ./ 1000;var94 ./ 1000]','grouped')
xlabel('Month')
ylabel(strcat('Consumption (',Power_Type,'h',') ', '[10^3]'))
legend('Facility Consumption',strcat('Solar Power:
',num2str(choosen_array_size ./ 1000),' m^2 [x10^3]',' Array Size'))
title(strcat('Monthly Consumption for',num2str(Year)))
set(gca,'ygrid','on')

var140 = unique(var101(:,5));

for i = 1:length(var140)
    var141 = var140(i);
    var142 = var101(find(var101(:,5) == var141),:);
    var143 = unique(var142(:,2));%month
    var144 = unique(var142(:,3));%day
    for j = 1:length(var143)
        for k = 1:length(var144)
            var145 = find(var142(:,2) == var143(j) & var142(:,3) ==
var144(k));
                if isempty(var145) == 1
                    continue
                else
                    var146(j,k) = trapz(var142(var145,4),var142(var145,end)
.* choosen_array_size);%solar
                    var147(j,k) = trapz(var142(var145,4),var142(var145,end-
1));%facility
                end
            end
        end
    end
    var148(1:numel(var146),i) = reshape(var146,numel(var146),1);%solar
    var149(1:numel(var147),i) =
reshape(var147,numel(var147),1);%facility
end

for i = 1:length(var148(1,:))
    var150 = var148(:,i);%solar
    var151 = var149(:,i);%facility
    var300 = find((var150 - var151) == 0);
    var150(var300) = [];
    var151(var300) = [];
    %    var150(var150 == 0) = [];%solar
    %    var151(var151 == 0) = [];%facility
    var152 = var150 - var151;
    weekday_consumption_met(i) = length(find(var152 > 0)) /
length(var152);
    weekday_consumption_not_met(i) = 1 - weekday_consumption_met(i);
end

var78 = [weekday_consumption_met;weekday_consumption_not_met];

```

```

figure(11)
bar(var140,var78' .* 100,'grouped')
xlabel('Weekday');
ylabel('Percent')
legend('Consumption Met','Consumption Not Met')
title(strcat('Weekday Consumption for',num2str(Year)))
set(gca,'ygrid','on')

var120 = unique(var101(:,4));

for i = 1:length(var120)
    var121 = var120(i);
    var122 = var101(find(var101(:,4) == var121),:);
    var123 = cat(2,var122(:,1:(end-1)),var122(:,end) .*
chosen_array_size);
    var124(1,i) = var121;
    var124(2,i) = (length(find((var123(:,end) - var123(:,end-1)) > 0))
/ length(var123(:,end))) * 100;
end

figure(12)
bar(var124(1,:) ./ 60,var124(2,:))
grid on
xlabel('Time (hr)')
ylabel('Percent Demand Met')
title(strcat('Hourly Demand Met for',num2str(Year)))

[C3 I3] = min(abs(chosen_array_size - Array_Size));
New_Payback = Total_Array_Cost(I3) / (var52(I3,1) + var99);
set(handles.edit9,'String',num2str(New_Payback))

maximum_power_output = max(var55);
annual_energy_production = sum(var87);
temp1 = find(Array_Size == chosen_array_size);
installed_system_cost = Total_Array_Cost(temp1);
annual_consumption_savings = (sum(var85) - sum(var87)) *
Consumption_Cost;
annual_demand_savings = (var79 - var81) .* Demand_Cost;

annual_demand_savings(find((var79 - var81) < 0)) = 0;
annual_demand_savings = sum(annual_demand_savings);

simple_payback = Payback(temp1);

set(handles.edit12,'String',num2str(fix(maximum_power_output)))
set(handles.edit13,'String',num2str(fix(annual_energy_production)))
set(handles.edit14,'String',num2str(fix(installed_system_cost)))
set(handles.edit15,'String',num2str(fix(annual_consumption_savings)))
set(handles.edit16,'String',num2str(fix(annual_demand_savings)))
set(handles.edit17,'String',num2str(fix(simple_payback)))

```

```

% close(dlg)
catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj, 'extended', 'hyperlinks', 'off'), 'Error');
end

handles.Array_Size = Array_Size;
handles.Payback = Payback;
handles.Total_Array_Cost = Total_Array_Cost;
handles.var52 = var52;
guidata(hObject, handles)

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit8 as text
%        str2double(get(hObject, 'String')) returns contents of edit8 as
a double
var1 = get(hObject, 'String');
Consumption_Sold = var1;
handles.Consumption_Sold = Consumption_Sold;
guidata(hObject, handles)

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit9 as text
%        str2double(get(hObject, 'String')) returns contents of edit9 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%       str2double(get(hObject,'String')) returns contents of edit10
as a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%       str2double(get(hObject,'String')) returns contents of edit11

```

as a double

```
% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
Array_Size = handles.Array_Size;
var52 = handles.var52;
```

```
Fig = figure;
plot(Array_Size ./ (10^3),var52 ./ (10^3))
xlabel('PV Array Size (m2 [x 103])')
ylabel('Total Savings ($) [x 103])')
title('Photovoltaic System Savings Curve')
grid on
```

```
answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)
```

```
% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hint: get(hObject,'Value') returns toggle state of togglebutton2
Array_Size = handles.Array_Size;
Total_Array_Cost = handles.Total_Array_Cost;
```

```

Fig = figure;
plot(Array_Size ./ 10^3, Total_Array_Cost ./ 10^6)
xlabel('Array Size (m^2) [x 10^3]')
ylabel('Array Cost ($) [x 10^6]')
title('Photovoltaic System Cost Curve')
grid on

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf, answer2{1}, 'epsc')
saveas(gcf, answer2{1}, 'fig')
close(Fig)

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject, 'Value') returns toggle state of togglebutton3
Array_Size = handles.Array_Size;
Payback = handles.Payback;

Fig = figure;
plot(Array_Size ./ 10^3, Payback)
xlabel('Array Size (m^2) [x 10^3]')
ylabel('Payback (yr)')
title('Photovoltaic Payback Curve')
grid on

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf, answer2{1}, 'epsc')
saveas(gcf, answer2{1}, 'fig')
close(Fig)

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit12 as text
%        str2double(get(hObject, 'String')) returns contents of edit12
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%       str2double(get(hObject,'String')) returns contents of edit13
as a double

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
%       str2double(get(hObject,'String')) returns contents of edit14
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%         str2double(get(hObject,'String')) returns contents of edit15
as a double

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as text
%         str2double(get(hObject,'String')) returns contents of edit16
as a double

```

```

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit17 as text
%         str2double(get(hObject,'String')) returns contents of edit17
as a double

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%         str2double(get(hObject,'String')) returns contents of edit18

```

as a double

```
% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Power Factor Correction Tool

```
function varargout = Power_Factor_Correction(varargin)
% POWER_FACTOR_CORRECTION MATLAB code for Power_Factor_Correction.fig
%   POWER_FACTOR_CORRECTION, by itself, creates a new
POWER_FACTOR_CORRECTION or raises the existing
%   singleton*.
%
%   H = POWER_FACTOR_CORRECTION returns the handle to a new
POWER_FACTOR_CORRECTION or the handle to
%   the existing singleton*.
%
%
POWER_FACTOR_CORRECTION('CALLBACK',hObject,eventData,handles,...) calls
the local
%   function named CALLBACK in POWER_FACTOR_CORRECTION.M with the
given input arguments.
%
%   POWER_FACTOR_CORRECTION('Property','Value',...) creates a new
POWER_FACTOR_CORRECTION or raises the
%   existing singleton*. Starting from the left, property value
pairs are
%   applied to the GUI before Power_Factor_Correction_OpeningFcn
gets called. An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to
Power_Factor_Correction_OpeningFcn via varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Power_Factor_Correction

% Last Modified by GUIDE v2.5 18-Jun-2014 16:32:02

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Power_Factor_Correction_OpeningFcn, ...
                  'gui_OutputFcn',  @Power_Factor_Correction_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Power_Factor_Correction is made visible.
function Power_Factor_Correction_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% varargin    command line arguments to Power_Factor_Correction (see
VARARGIN)

% Choose default command line output for Power_Factor_Correction
handles.output = hObject;

% Update handles structure

data = getappdata(0, 'data');
handles.data = data;

powerfactordata = getappdata(0, 'powerfactordata');
handles.powerfactordata = powerfactordata;

handles.Power_Type = 'kW';

Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
guidata(hObject, handles);
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;

var1 = get(handles.edit1, 'String');
handles.Allowed_PF = var1;

var2 = get(handles.edit2, 'String');
handles.Demand_Cost = var2;

var3 = get(handles.edit3, 'String');
handles.Fixed_Capacitance_Cost = var3;

var4 = get(handles.edit4, 'String');
handles.Variable_Capacitance_Cost = var4;

```

```

guidata(hObject, handles);

% UIWAIT makes Power_Factor_Correction wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Power_Factor_Correction_OutputFcn(hObject,
eventdata, handles)
% varargout    cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject     handle to pushbutton1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
cla reset
try

dlg = ProgressDialog( ...
    'StatusMessage', 'Working...', ...
    'Indeterminate', true);

num1 = handles.data;
num2 = handles.powerfactordata;

Year = str2num(handles.Year);
Allowed_PF = str2num(handles.Allowed_PF);
Demand_Cost = str2num(handles.Demand_Cost);
Fixed_Capacitance_Cost = str2num(handles.Fixed_Capacitance_Cost);
Variable_Capacitance_Cost = str2num(handles.Variable_Capacitance_Cost);
Power_Type = handles.Power_Type;

var1 =
datenum(cat(2,num1(:,1:3),zeros(length(num1(:,1)),1),num1(:,4),zeros(le
ngth(num1(:,1)),1)));
var2 =
datenum(cat(2,num2(:,1:3),zeros(length(num2(:,1)),1),num2(:,4),zeros(le
ngth(num2(:,1)),1)));

```

```

for i = 1:length(var1(:,1))
    var3 = var1(i);
    var4 = var3 - var2;
    [C1,I1] = min(abs(var4));
    var5(i,1) = num1(I1,end);
    var5(i,2) = num2(I1,end);
end

var6 = cat(2,num1(:,1:end-1),var5);
var7 = find(var6(:,1) == Year);
var8 = var6(var7,:);
var9 = unique(var8(:,2));

for i = 1:length(var9);
    var10 = var9(i);
    var11 = find(var8(:,2) == var10);
    var12 = var8(var11,:);
    [C2,I2] = max(var12(:,end-1));
    var13(i,:) = var12(I2,:);
end

var14 = find(var13(:,end) < Allowed_PF * 100);
var15 = var13(var14,end - 1) .* ((Allowed_PF * 100) ./
var13(var14,end));
var16 = var13;
var16(var14,end-1) = var15;
var17 = cat(2,var13(:,1:7),var16(:,6));%[date,demand,PF,demand penalty]

[AX H1 H2] = plotyy(handles.axes1,var17(:,2),[var17(:,end-
2),var17(:,end)],var17(:,2),var17(:,end-1),@bar,@scatter);
set(H2,'MarkerEdgeColor','b','MarkerFaceColor','c','SizeData',100)

xlabel('Month')
ylabel(AX(1),strcat('Demand (',Power_Type,')'))
ylabel(AX(2),'Power Factor')
h1 = legend(handles.axes1,'Actual Demand','Billed Demand','Power
Factor');
set(handles.axes1,'ygrid','on')
temp1 = get(AX(1),'YTick');
set(AX(1),'YTick',fix(linspace(temp1(1),temp1(end),10)))
temp2 = get(AX(2),'YTick');
set(AX(2),'YTick',(linspace(temp2(1),temp2(end),10)))
temp3 = get(handles.axes1,'XTick');
set(h1,'Location','SouthOutside')

bar(handles.axes2,var17(:,2),[(var17(:,end-2) .* Demand_Cost) ./
10^3,(var17(:,end) .* Demand_Cost) ./ 10^3],'grouped')
xlabel(handles.axes2,'Month')
ylabel(handles.axes2,'Cost ($) [x 1000]')
h2 = legend(handles.axes2,'Actual Charge','Penalty Charge');
set(h2,'Location','SouthOutside')

```

```

set(handles.axes2,'ygrid','on')

var21 = sum(var17(:,end) .* Demand_Cost - var17(:,end-2) .*
Demand_Cost); %total cost incurred by power factor penalty

if Power_Type == 'kW'
var18 = tand(acosd(var17(:,7) ./ 100)) .* var17(:,6);%kVAR from kW data
else
var18 = sind(acosd(var17(:,7) ./ 100)) .* var17(:,6);%kVAR from kVA
data
end

var19 = min(var18);%Fixed Capacitance
var20 = max(var18) - var19;%Variable Capacitance
var22 = var19 * Fixed_Capacitance_Cost + var20 *
Variable_Capacitance_Cost;%Capital Cost
var23 = var22 / var21;%Payback

set(handles.edit5,'String',num2str(fix(var19)))
set(handles.edit6,'String',num2str(fix(var20)))
set(handles.edit7,'String',num2str(fix(var22)))
set(handles.edit8,'String',num2str(fix(var23)))

% close(dlg)
catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj,'extended','hyperlinks','on'),'Error');
end

handles.var17 = var17;
handles.Demand_Cost = Demand_Cost;
handles.Power_Type = Power_Type;
guidata(hObject, handles);

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%         str2double(get(hObject,'String')) returns contents of edit5 as
a double

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%       str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%       str2double(get(hObject,'String')) returns contents of edit7 as
a double

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit8_Callback(hObject, eventdata, handles)
% hObject handle to edit8 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
% str2double(get(hObject,'String')) returns contents of edit8 as
a double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject handle to edit8 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
% contents{get(hObject,'Value')} returns selected item from
popupmenu1
var1=get(handles.popupmenu1,'String');
var2=get(handles.popupmenu1,'Value');
Year=var1{var2};

```

```

handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%       str2double(get(hObject,'String')) returns contents of edit1 as
a double
    var1 = get(hObject,'String');
    Allowed_PF = var1;
    handles.Allowed_PF = Allowed_PF;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)

```

```

% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as
a double
    var1 = get(hObject,'String');
    Demand_Cost = var1;
    handles.Demand_Cost = Demand_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%         str2double(get(hObject,'String')) returns contents of edit3 as
a double
    var1 = get(hObject,'String');
    Fixed_Capacitance_Cost = var1;
    handles.Fixed_Capacitance_Cost = Fixed_Capacitance_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.

```

```

%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%       str2double(get(hObject,'String')) returns contents of edit4 as
a double
    var1 = get(hObject,'String');
    Variable_Capacitance_Cost = var1;
    handles.Variable_Capacitance_Cost = Variable_Capacitance_Cost;
    guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)

var1 = get(hObject,'String');
handles.Power_Type = var1;
guidata(hObject,handles)

```

```

set(handles.edit9,'String',strcat('$ / ', ' ',var1,' mo'))

function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%         str2double(get(hObject,'String')) returns contents of edit9 as
a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

var17 = handles.var17;
Power_Type = handles.Power_Type;

Fig = figure;
[AX H1 H2] = plotyy(var17(:,2),[var17(:,end-
2),var17(:,end)],var17(:,2),var17(:,end-1),@bar,@scatter);
set(H2,'MarkerEdgeColor','b','MarkerFaceColor','c','SizeData',100)

xlabel('Month')
ylabel(AX(1),strcat('Demand (',Power_Type,')'))
ylabel(AX(2),'Power Factor')
h1 = legend('Actual Demand','Billed Demand','Power Factor');
set(gca,'ygrid','on')
temp1 = get(AX(1),'YTick');

```

```

set(AX(1), 'YTick', fix(linspace(temp1(1), temp1(end), 10)))
temp2 = get(AX(2), 'YTick');
set(AX(2), 'YTick', fix(linspace(temp2(1), temp2(end), 10)))
temp3 = get(gca, 'XTick');
set(h1, 'Location', 'SouthOutside')

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf, answer2{1}, 'epsc')
saveas(gcf, answer2{1}, 'fig')
close(Fig)

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject      handle to togglebutton2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hint: get(hObject, 'Value') returns toggle state of togglebutton2

var17 = handles.var17;
Demand_Cost = handles.Demand_Cost;

Fig = figure;
bar(var17(:,2), [(var17(:,end-2) .* Demand_Cost) ./ 10^3, (var17(:,end)
.* Demand_Cost) ./ 10^3], 'grouped')
xlabel('Month')
ylabel('Cost ($) [x 1000]')
h2 = legend('Actual Charge', 'Penalty Charge');
set(h2, 'Location', 'SouthOutside')
set(gca, 'ygrid', 'on')

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf, answer2{1}, 'epsc')
saveas(gcf, answer2{1}, 'fig')
close(Fig)

```

Ratcheting Tool

```
function varargout = Ratcheting(varargin)
% RATCHETING MATLAB code for Ratcheting.fig
%   RATCHETING, by itself, creates a new RATCHETING or raises the
existing
%   singleton*.
%
%   H = RATCHETING returns the handle to a new RATCHETING or the
handle to
%   the existing singleton*.
%
%   RATCHETING('CALLBACK',hObject,eventData,handles,...) calls the
local
%   function named CALLBACK in RATCHETING.M with the given input
arguments.
%
%   RATCHETING('Property','Value',...) creates a new RATCHETING or
raises the
%   existing singleton*. Starting from the left, property value
pairs are
%   applied to the GUI before Ratcheting_OpeningFcn gets called. An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to Ratcheting_OpeningFcn via
varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Ratcheting

% Last Modified by GUIDE v2.5 18-Jun-2014 16:24:25

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Ratcheting_OpeningFcn, ...
                  'gui_OutputFcn',  @Ratcheting_OutputFcn, ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```

    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Ratcheting is made visible.
function Ratcheting_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Ratcheting (see VARARGIN)

% Choose default command line output for Ratcheting
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

data = getappdata(0, 'data');
handles.data = data;

handles.Power_Type = 'kW';

Years = cellstr(num2str(unique(data(:,1))));
set(handles.popupmenu1, 'String', Years)
var6=get(handles.popupmenu1, 'String');
var7=get(handles.popupmenu1, 'Value');
Year=var6{var7};
handles.Year = Year;
guidata(hObject, handles)

var1 = get(handles.edit1, 'String');
handles.Input_Demand = var1;

var2 = get(handles.edit4, 'String');
handles.Date = var2;

var3 = get(handles.edit2, 'String');
handles.Ratchet_Percentage = var3;

var4 = get(handles.edit3, 'String');
handles.Window_Period = var4;

var5 = get(handles.edit5, 'String');
handles.Demand_Cost = var5;
guidata(hObject, handles);

% UIWAIT makes Ratcheting wait for user response (see UIRESUME)
% uiwait(handles.figure1);

```

```

% --- Outputs from this function are returned to the command line.
function varargout = Ratcheting_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% function Ratcheting()

cla reset
try

    dlg = ProgressDialog( ...
        'StatusMessage', 'Working...', ...
        'Indeterminate', true);

num = handles.data;

Year = str2num(handles.Year);
Input_Demand = str2num(handles.Input_Demand); %kW
Ratchet_Percentage = str2num(handles.Ratchet_Percentage);
Window_Period = str2num(handles.Window_Period); %Months
Date = handles.Date;
Demand_Cost = str2num(get(handles.edit5, 'String')); %$/kW mo;
Power_Type = handles.Power_Type;

date4 = datenum(Date);
date1 = datevec(date4);
date2 = repmat(date1(1,1:3), Window_Period, 1);

var1 = unique(num(:,1));

for i = 1:length(var1)
    var2 = var1(i);
    var3 = find(num(:,1) == var2);
    var4 = num(var3,:);
    var5 = unique(var4(:,2));
    for j = 1:length(var5)
        var6 = var5(j);
        var7 = find(var4(:,2) == var6);

```

```

        var8 = var4(var7,:);
        var9{i,1}(j,1) = max(var8(:,end));
        [C1 I1] = max(var8(:,end));
        var10{i,1}(j,1:3) = var8(I1,1:3);
    end
end

var11 = cell2mat(var9);
var12 = cell2mat(var10);
date3 = cat(1,date2,var12);
var13 = cat(1, repmat(Input_Demand,Window_Period,1),var11);

for i = (Window_Period + 1):length(var13)
    var14(i-Window_Period) = max(var13((i-Window_Period):i));
    [C1 I1] = max(var13((i-Window_Period):i));
    var20(i-Window_Period,1) = I1;
    date5{i-Window_Period} = date3((i-Window_Period):i,1:3);
end

for i = 1:length(date5)
    var23 = var20(i);
    var24 = date5{i};
    var25(i,:) = var24(var23,:);
end

var15 = cat(2,var11,var14');
var16 = find(var15(:,1) ~= var15(:,2) & var15(:,1) < (var15(:,2) .*
Ratchet_Percentage));
var15(var16,1) = fix(var15(var16,2) .* Ratchet_Percentage);
var17 = fix(cat(2,var12,var11,var15(:,end-1)));

var18 = find(var17(:,1) == Year);
var19 = var17(var18,:);
ratchet_date_listbox = var25(var18,:);

bar(handles.axes1,var19(:,2),var19(:,end-1:end),'grouped')
h1 = legend(handles.axes1,'Original Demand','Ratched Demand');
xlabel(handles.axes1,'Month')
ylabel(handles.axes1, strcat('Demand (',Power_Type,')'))
set(handles.axes1,'ygrid','on')
set(h1, 'Location', 'SouthOutside')

bar(handles.axes2,var19(:,2),(var19(:,end-1:end) * Demand_Cost) ./
10^3,'grouped')
h2 = legend(handles.axes2,'Original Demand Cost','Ratched Demand
Cost');
xlabel(handles.axes2,'Month')
ylabel(handles.axes2,'Cost ($) [x 1000]')
set(h2, 'Location', 'SouthOutside')
set(handles.axes2,'ygrid','on')

```

```

var20 = var19(:,end-1:end) * Demand_Cost;
var21 = sum(var20(:,end) - var20(:,end-1));
set(handles.edit8, 'String', num2str(fix(var21)))

for i = 1:length(ratchet_date_listbox(:,1))
    var22{i} = datestr([ratchet_date_listbox(i,1:3),0,0,0]);
end

set(handles.listbox1, 'String', var22);
catch errorObj
% If there is a problem, we display the error message
errordlg(getReport(errorObj, 'extended', 'hyperlinks', 'on'), 'Error');
end

handles.var19 = var19;
handles.Demand_Cost = Demand_Cost;
handles.Power_Type = Power_Type;
guidata(hObject, handles);

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject, 'String')) returns popupmenu1
% contents as cell array
% contents{get(hObject, 'Value')} returns selected item from
% popupmenu1
var1=get(handles.popupmenu1, 'String');
var2=get(handles.popupmenu1, 'Value');
Year=var1{var2};
handles.Year = Year;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
% called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

```

```

function edit1_Callback(hObject, eventdata, handles)
% hObject      handle to edit1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%         str2double(get(hObject,'String')) returns contents of edit1 as
a double
var1 = get(hObject,'String');
Input_Demand = var1;
handles.Input_Demand = Input_Demand;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as
a double
var1 = get(hObject,'String');
Ratchet_Percentage = var1;
handles.Ratchet_Percentage = Ratchet_Percentage;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject     handle to edit3 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%     str2double(get(hObject,'String')) returns contents of edit3 as
a double
var1 = get(hObject,'String');
Window_Period = var1;
handles.Window_Period = Window_Period;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit3 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject     handle to edit4 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%     str2double(get(hObject,'String')) returns contents of edit4 as
a double
var1 = get(hObject,'String');
Date = var1;
handles.Date = Date;

```

```

guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%       str2double(get(hObject,'String')) returns contents of edit5 as
a double
var1 = get(hObject,'String');
Demand_Cost = var1;
handles.Demand_Cost = Demand_Cost;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as
a double

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit7_Callback(hObject, eventdata, handles)
% hObject      handle to edit7 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as
a double

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit7 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uipanel1
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)
var1 = get(hObject, 'String');
handles.Power_Type = var1;
guidata(hObject, handles)
set(handles.edit9, 'String', strcat('$ / ', ' ', var1, ' mo'))
% function txt = myupdatefcn(empty, event_obj)
% % Customizes text of data tips
% pos = get(event_obj, 'Position');
% var1 = datestr(datenum(ratchet_date_set(fix(pos(1)), 1:3)));
% txt = var1;
% end

% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject, 'String')) returns listbox1
contents as cell array
%   contents{get(hObject, 'Value')} returns selected item from
listbox1

% --- Executes during object creation, after setting all properties.
function listbox1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%   See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8 as
%        a double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit8 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as
%        a double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit9 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject      handle to togglebutton1 (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

```

```

var19 = handles.var19;
Power_Type = handles.Power_Type;

```

```

Fig = figure;
bar(var19(:,2),var19(:,end-1:end),'grouped')
h1 = legend('Original Demand','Ratcheted Demand');
xlabel('Month')
ylabel(strcat('Demand (',Power_Type,')'))
set(gca,'ygrid','on')
set(h1, 'Location', 'SouthOutside')

```

```

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

```

```

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hint: get(hObject,'Value') returns toggle state of togglebutton2

```

```

var19 = handles.var19;
Demand_Cost = handles.Demand_Cost;

```

```

Fig = figure;
bar(var19(:,2),(var19(:,end-1:end) * Demand_Cost) ./ 10^3,'grouped')
h2 = legend('Original Demand Cost','Ratcheted Demand Cost');
xlabel('Month')
ylabel('Cost ($) [x 1000]')
set(h2, 'Location', 'SouthOutside')
set(gca,'ygrid','on')

```

```

answer1 = inputdlg('FileName?');
answer2 = cellstr(answer1);
saveas(gcf,answer2{1},'epsc')
saveas(gcf,answer2{1},'fig')
close(Fig)

```

