PARAMETRIC OPTIMIZATION: APPLICATIONS IN SYSTEMS DESIGN

A Dissertation

by

EDGAR GALVAN

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

DOCTOR OF PHILOSOPHY

Chair of Committee,	Richard J. Malak
Committee Members,	Raymundo Arróyave
	Daniel A. McAdams
	Douglas Allaire
	Darren J. Hartl
Head of Department,	Andreas A. Polycarpou

December 2016

Major Subject: Mechanical Engineering

ABSTRACT

The aim of this research is to introduce the notion of parametric optimization (PO) as a useful approach for solving systems design challenges. In this research, we define PO as the process of finding the optimal solution as a function of one or more parameters. Parameters are variables that affect the optimal solution but, unlike the decision variables, are not directly controlled by the designer. The principal contributions of this research are **(1)** a novel formulation of the PO problem relevant to systems design, **(2)** a strategy for empirically assessing the performance of parametric search algorithms, **(3)** the development and evaluation of novel algorithms for PO, and **(4)** a demonstration of the use of PO for two real-world systems design challenges. The real-world demonstrations, include the design of **(i)** a multi-ratio vehicle transmission, and **(ii)** a Liquid Metal Magnetohydrodynamic Pump.

A practical challenge of applying the notion PO to systems design is that existing methods are limited to problems where the models are accessible algebraic equations and single objectives. However, many challenges in systems design involve inaccessible models or are too complicated to be manipulated algebraically and have multiple objectives. If PO is to be used widely in systems design, there is a need for search methods that can approximate the solution to a general PO problem. As a step toward this goal, a strategy for performance assessment is developed. The use of the mean Hausdorff distance is proposed as a measure of solution quality for the PO problem. The mean Hausdorff distance has desirable properties from a mathematical and decision theoretic basis. Using the proposed performance assessment strategy, two algorithms for parametric optimization are evaluated, **(a)** p-NSGAII which is a straightforward extension of existing methods to the case with parameters, and **(b)**

P3GA an algorithm intended to exploit the parametric structure of the problem. The results of the study indicate that a considered approach, P3GA, to the PO problem results in considerable computational advantage.

DEDICATION

Para mi mamá y papá

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Richard J. Malak for his advise and support over the years. I appreciate the direction and funding to make my Ph.D. experience *surprisingly* productive. I would also like to thank Dr. Arróyave, Dr. McAdams, Dr. Allaire, and Dr. Hartl for serving on my committee. I would like to thank Dr. Arróyave and Dr. Hartl for the difficult problems that inspire much of this work.

Thanks to the members, past and present, of the Design Systems Laboratory who have contributed to this dissertation, especially Ben, Chuck, Joanna, and Sean. The group has been a source of constant support.$^{\text{distraction}}$

A special thanks to my family. I am grateful to my parents for the sacrifices they've made and the support they've given me. Finally I'd like to thank my friends outside of the lab, Lorenzo, Beto, Davis, and another special thanks to Kat.

NOMENCLATURE

σ	Electrical conductivity.
$\bar{P}$	Fluid volumetric thermal power input.
β_i	Lagrangian multiplier
$\dot{m}$	Fluid mass flow rate.
μ	Fluid dynamic viscosity.
ω	angular velocity of a single wheel on the automobile
ω_e	angular velocity of the engine shaft
Φ	Data space → feature space
ϕ	Voltage potential.
ρ	Material density.
ρ	density of air
ε_0	Electrical permittivity of free space.
ε_r	Material-relative permittivity.
$\boldsymbol{\tau}$	Fluid viscous stress tensor.
$\boldsymbol{\theta}$	Set of design parameters.
$\boldsymbol{\theta}^{lb}, \boldsymbol{\theta}^{ub}$	Lower and upper bounds on design parameters.
$\boldsymbol{a}$	centroid of hyper-sphere

$\boldsymbol{B}$	Magnetic flux density vector.
$\boldsymbol{b}$	centroid of feature space hyper-sphere
$\boldsymbol{B}_r$	Remnant magnetic flux density vector.
$\boldsymbol{D}$	Electric displacement vector.
$\boldsymbol{E}$	Electric field vector.
$\boldsymbol{F}$	Set of objective functions (e.g., to be minimized).
$\boldsymbol{f}$	Fluid volumetric force vector.
$\boldsymbol{f}_L$	Lorentz volumetric force vector.
$\boldsymbol{f}_{oth}$	Additional fluid volumetric force contributions.
$\boldsymbol{I}$	Identity tensor.
$\boldsymbol{J}$	Current density vector.
$\boldsymbol{S}$	Fluid strain rate tensor.
$\boldsymbol{u}$	Fluid velocity.
$\boldsymbol{v}$	Set of design variables.
$\boldsymbol{v}^*$	Set of design variables satisfying a constrained optimization problem.
$\boldsymbol{v}^{lb}, \boldsymbol{v}^{ub}$	Lower and upper bounds on design variables.
$\boldsymbol{x}_i$	A vector in the design variable space X
$\boldsymbol{z}$	Test point
ξ_d	speed ratio for the differential

ξ_i	Slack variable
Ξ_t	discretely adjustable speed ratio for the mechanical transmission
ζ_0	Magnetic permeability of free space.
ζ_r	Material-relative magnetic permeability.
A	effective frontal area for the automobile
A_{elec}	Area of MHD pump electrodes.
c	SVDD Parameter
C_d	dimensionless drag coefficient for the automobile
c_p	Fluid specific heat at constant pressure.
C_r	rolling resistance coefficient for the automobile
d_{chan}	Fluid channel depth.
d_{mag}	Permanent magnet depth.
g	gravitational constant
I_{app}	Total current applied to MHD pump electrodes.
K	Kernel function
k	Fluid thermal conductivity.
l_{chan}	Fluid channel length in flow direction.
l_{elec}	Length of MHD pump electrodes in flow direction.
l_{mag}	Permanent magnet length in flow direction.

l_{res}	Hot/cold reservoir length in flow direction.
m	mass of the automobile
m_{chan}	Fluid-filled channel mass (reservoirs neglected).
N	Dimensionality of thermodynamic conditions space
n	Number of data points
p	Fluid pressure.
P_{EM}	Thermal power added to fluid channel by electromagnetic effect of MHD pump.
P_{hot}	Thermal power added to fluid channel by electronic subsystem.
p_{ref}	Fluid reference pressure.
Q	Fluid volumetric flow rate.
q	Gaussian kernel parameter
R	radius of a single tire on the automobile
r	Hyper-sphere radius
r_{chan}	Fluid channel aspect ratio (w_{chan}/d_{chan}).
T	Fluid temperature.
T	torque on a single wheel of the automobile
t	time
T_e	torque on the engine shaft

T_{cold}	Cold reservoir sink temperature.
T_{EM}	MHD pump temperature.
T_{hot}	Hot reservoir temperature.
V_{app}	Voltage applied across MHD pump electrodes.
w_{chan}	Fluid channel width.
w_{mag}	Permanent magnet width.
w_{res}	Hot/cold reservoir width.
X	Training data set
y_i	Training-data label

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION: PARAMETRIC OPTIMIZATION FOR SYSTEMS DESIGNERS

1.1 Parametric Optimization for Systems Design

1.1.1 Motivation

Before optimization can be done, the decision problem must be properly modeled. Correctly modeling the problem is as important as the optimization since good optimization techniques do not remedy a poor model. A mathematical approach to a decision problem is often divided into four steps

(a) selection of mathematical process model

(b) selection of the decision variables available for control

(c) selection of preferences among decision criteria

(d) selection of appropriate search strategy

Much of engineering literature is concerned with (a) and (b), i.e., the development of models of engineered artifacts. The focus of this research is instead on the selection of (c) decision criteria and (d) search strategy. Specifically, we consider decision problems where some preference information is not known.

Parametric optimization is a type of mathematical optimization where the optimization problem is solved as a function of one or multiple parameters. We define **parameters** as variables relevant to optimization that are outside of the control of the designer; often these are environmental variables or variables controlled by other designers. By solving the optimization problem in this way, we can remain agnostic about preferences in the parameter(s).

The general notion of parametric optimization, as we have defined it, has been used in several fields, including economics, mathematical optimization, and more recently model based controls engineering. The broad aim of this research is to present parametric optimization in a way that is relevant to systems design challenges. In this research, an **engineered system** is defined as something designed by more than one person [1]. Such systems require disciplinary expertise that are to broad for a single person or the system is too extensive (containing too many parts). The design of engineered systems requires not only in-depth multidisciplinary technical knowledge, but also the careful management of its conception and realization. Systems design traditionally encompasses architecture definition, specification of interfaces, performance requirements, etc. We draw a distinction between systems design and systems engineering, the latter is typically considered a broader field that includes customer elicitation, life-cycle planning, and documentation, to name a few. Because systems engineering subsumes systems design, we will refer to systems designers as "systems engineers." This research is not focused on a particular area or problem in systems design, rather the aim is to **(i)** demonstrate how some systems design challenges can be better formulated as PO problems, and **(ii)** to develop and evaluate parametric approaches for addressing specific challenges.

1.1.2 Terminology

Prior to discussing parametric optimization for systems design further, it is useful to establish terminology used throughout this work. Terms such as parameter, design variable, attributes, etc. are not consistently defined in the literature. The following definitions are used throughout this dissertation.

Definition 1. *(Discipline Engineer) Specialists in a specific engineering field or discipline, such as mechanical engineering, civil engineering, etc. A discipline engineer* *may refer to a single person or a team and may have expertise in multiple areas. The goals of the discipline engineers are typically performance-related, such as minimize mass, maximize speed, etc.*

Definition 2. *(Systems Engineer) Engineer focused on the system architecture and interfaces between subsystems. A key task of the systems engineer is to manage the discipline engineering tasks. The goal of the systems engineer are typically project-related, such as project cost, safety, value, etc.*

Definition 3. *(Designer) Engineer involved in the design of a system. May refer to either systems and discipline engineers. The designer has some decision making ability.*

Definition 4. *(Design Variable) The variables specify the alternatives available to the designer. The designer has direct control over the design variables. Also sometimes referred to as decision variables.*

Definition 5. *(Objective) A design goal with measurable progress. For example, "to minimize structural mass."*

Definition 6. *(Objective Attribute) A measure of progress relevant to decision making. Also referred to as performance metric, or cost metric. We chose the term "objective attribute" rather than the more common " performance attribute" to avoid confusion when discussing "algorithmic performance."*

Definition 7. *(Parameter Attribute) The variables relevant to decision making that are outside of the control of the designer. These are often environmental variables or variables controlled by other designers. Here in, the terms* parameter attribute *and* parameter *are used synonymously.*

Definition 8. *(Attribute) Characteristics of an alternative relevant to decision making. These include parameters and objective attributes.*

1.2 Major Research Tasks

The aim of this research is *to introduce the concept of parametric optimization as a useful approach for systems design challenges.* As a first step towards this goal, we consider the following major research tasks.

1.2.1 Application of Parametric Optimization to Systems Design Challenges

Many designers use parametric optimization informally for systems design. Mathematical descriptions allow for a better understanding of complex design challenges. Therefore, designers would benefit from a mathematical treatment of parametric optimization problems. Specifically, the parametric optimization problems will be motivated by **(1)** the problem of computing the expected value of information, and **(2)** capability modeling.

1.2.1.1 Approximating the Value of Gathering Information

It is important for engineering firms to be able to develop forecasts of recommended courses of action based on available information. In this context, good decisions are characterized by careful management of uncertainty. Information gathering activities, such as prototype development, or consulting experts, can reduce uncertainty but at a cost. The first step in deciding whether or not to perform an information gathering action is to compute its value. We term this the **information decision**. For example, an automobile manufacturer may use a computer simulation of a hydraulic motor and pump in the design of a new vehicle. The model may contain random variables that can be more accurately determined through expensive experiments (information gathering actions). To decide whether or not to perform the experiments, the automobile manufacturer must be able to first quantify the gain of conducting them.

Under several decision making frameworks, the information decision is addressed by considering the *expected reward* of performing the information gathering activity. One considers the distribution of information that might be revealed and how that information is expected to improve the design. Mathematically, one takes an expectation of the design decision over the distribution of the random variable: the revealed information. Problematically, in engineering the design decision often involves complex optimization, resulting in a "nested" optimization loop. As has been shown in the literature, solving the "nested" loop can become prohibitive, even for relatively straightforward information decision problems. The computational expense may even exceed the cost of actually performing the information gathering activity. Consequently, if information decisions are to be addressed algorithmically, there exists a need for novel approaches to reduce the computational expense associated with computing the expected value of gathering information.

In this research, we will show that by recasting the nested optimization as a PO problem, one can potentially eliminate the "nested" loop that arises in many information decision frameworks. In this research, we will show that the information decision problem can be more efficiently solved by exploiting its parametric structure.

1.2.1.2 Subsystem Capability Modeling

Engineering projects typically involve many individuals, each contributes knowledge to the project and none has complete knowledge about the system under development. Although this specialization of knowledge is necessary and even desirable from certain perspectives, it can create challenges for system-level decision making.

This challenge is evident in the relationship between systems engineers and discipline engineers during the early stages of a systems engineering project. Systems engineers must make decisions such as to define the system architecture (relationships between subsystems, especially at the interfaces) and to allocate requirements or objectives to subsystems (setting performance and compatibility requirements) [2, 3]. They do this with system-level objectives in mind (e.g., to maximize system value) but must consider carefully the capabilities and limitations of the subsystem technologies available for use in the system. Failure to recognize, for example, that particular combinations of properties for a subsystem are impossible can lead to an inferior system definitions or requirements that are altogether infeasible.

Discipline engineers have relevant expertise about particular engineering technologies, but individual discipline engineers generally are knowledgeable about only a subset of the technologies under consideration and may know nothing about system-level considerations. Thus, communication of capabilities from discipline engineers to systems engineers can be an important factor in the success of a systems engineering project.

Consider for example, space mission design where a key step is the evaluation of alternative mission architectures. The mission architecture includes launch element, spacecraft bus, payload, etc. The spacecraft trajectory that optimizes performance is dependent on the mission architecture. In order to evaluate the various mission architectures that may be under consideration, it is necessary to compute the optimal trajectory for each architecture. A common approach to this problem is for the astrodynamics engineer to precompute the optimal trajectory as a function of various mission parameters. In other words, the astrodynamics engineer solves a challenging parametric optimization problem: maximize performance as a function of mission parameters. The aim is to facilitate the evaluation of the large number of available mission architectures. In this scenario, parametric optimization is used to allow for concurrent or off-line optimization independent of the rest of the system. Once the parametric problem is solved by the discipline engineer, that solution can be used to evaluate mission architectures at a system level. This process of "capability modeling," where the optimal capabilities of a subsystem are determined as a function of external parameters is a powerful tool that is commonly used by practicing designers; typically handled ad hoc. Designers would benefit from a formal mathematical technique for capability modeling. Such a mathematical representation would allow for the development of tailored optimization techniques.

The parameterized Pareto frontier (PPF) [4] has been introduced as a mathematical basis for modeling the capabilities of a subsystem. The PPF contains only designs that may potentially be optimal and can serve a compact representation of the capabilities of a subsystem. Importantly, the PPF features multiple parameter and performance attributes. In this research we will extend the parametric optimization problem formulation to the case with multiple objectives. The solution to the mutltiobjective multi-parametric optimization problem is the PPF.

1.2.2 Performance Assessment of Approximation Algorithms for Parametric Optimization

It is important in any optimization field to be able to quantitative evaluate and compare different optimization algorithms. The performance assessment of algorithms allows us to simplify judgment between algorithms and helps us better understand the algorithm's behavior. The appropriate framework for comparison depends on the nature of the algorithms of interest and the problems which those algorithms aim to solve.

We develop an empirical framework for assessing the performance of parametric optimization algorithms. The focus is on the question of how to determine whether one solution is better than another for parametric optimization. For traditional single-objective optimization algorithms, comparing solution quality is straightforward: one only needs to consider the objective value, smaller or larger is better. This however is not the case for parametric optimization. Instead of a single solution, parametric optimization problems have a solution set. Evaluating the quality of the solution set is a practical challenge for the empirical performance assessment of parametric algorithms.

1.2.3 Propose and Evaluate Novel Algorithms for Parametric Optimization

Engineering models are often nonlinear (dynamical systems, regression models) or non-analytical (Finite Element Analysis, Process Models, etc.). Thus, this research will be focused on general nonlinear problems that may be non-convex, multi-modal, and discontinuous.

As a first step in as a first step in introducing parametric optimization to systems design, we develop two heuristic algorithms for parametric optimization (i) the parametric non-dominated sorting algorithm (p-NSGAII) which is an extension of NSGAII to the case with parameters and (ii) predictive parametric Pareto genetic algorithm (P3GA). At the cost of solution accuracy, the benefits of heuristic approaches are (1) they are easy to implement since they require little knowledge about the mathematical structure of the problem (2) and can be applied to a wide range of problems.

1.2.4 Demonstration of Parametric Optimization for Systems Design

Finally, we demonstrate the use of parametric optimization on realistic engineering problems. Specifically, we demonstrate the use of parametric optimization for computing the expected value of gathering information in the design of a multi-ratio transmission and capability modeling of a cooling system.

1.3 Contributions and Scope

A summary of the major contributions of this research is as follows

1. Demonstration of the use of parametric optimization in the context of systems design.

2. Development of a performance assessment strategy for approximation algorithms for parametric optimization

3. Assessment of novel heuristic algorithms for parametric optimization

4. Demonstration of parametric optimization for real-world systems design challenges.

In this dissertation, we demonstrate how some common systems design challenges can be modeled as parametric optimization problems. The aim in modeling the problem this way is to allow for the development of tailored solution techniques that can exploit the parametric structure of the problem. As a first step, we develop two general purpose heuristic algorithms for parametric optimization. Because of the wide applicability of heuristic algorithms, they are a reasonable first step in introducing parametric optimization to systems design. However, heuristic algorithms are limited in their performance, especially in terms of the optimally of the solution. The aim of presenting the heuristic algorithms is not to present *the way* to do parametric optimization in systems design but simply *one way*.

The purpose of solving a parametric optimization problem is to *use* the solution. However, in approximating the solution rather than finding it exactly, a number of new problems arise. Two particularly difficult issues are (1) how to best use the approximation and (2) what is the penalty we pay for using an approximation? In this dissertation, we present an approach for using the approximation but do not explore alternatives or what the consequences are of using the approximation.

2. BACKGROUND

2.1 Definition of Parametric Optimization

2.1.1 Standard Optimization

Let $X \subseteq \mathbb{R}^n$ be the set of feasible decision vectors $\boldsymbol{x}$. The feasible set X is typically defined by geometric constraints, physical laws, and practical considerations. The objective function $f : X \to \mathbb{R}$ ranks each decision vector. We will use minimization herein without loss of generality since maximizing f is equivalent to minimizing $-f$. Using this notation, the standard optimization problem as

$$\begin{aligned} y^* = \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \end{aligned} \tag{2.1}$$

The aim is to find a solution, denoted y^*, that is the minimum of the objective function $f(\boldsymbol{x})$, and $\boldsymbol{x} \in X$. Generally, mathematical optimization techniques are used to find the extrema (local or global) of the function f. In engineering, optimization is often used in the development of computational models. For example, in computational materials science, the total Gibbs energy of all phases is minimized for the calculation of phase equilibria [5]. Because focus of this work engineering *design*, we use optimization for the selection of the "best" alternative(s).

2.1.2 Parametric Optimization

In parametric optimization, one seeks to find the solution to a standard optimization problem as a function of some parameters. The parameters are variables that relevant to decision making but are outside of the control of the designer. In a systems design context, parameters may include unspecified design requirements, environmental conditions, or properties of other subsystems. Intuitively, the parametric search problem can be thought of as an application of standard optimization at every combination of some parameter value(s). Let $\Theta \subseteq \mathbb{R}^p$ be the set of feasible parameter vectors $\boldsymbol{\theta}$. We extend Eq. 2.1 to the parametric case as

$$\begin{aligned} y^*(\boldsymbol{\theta}) = \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}, \boldsymbol{\theta}) \leq 0 \; \forall \; i = 1, \ldots, q \\ & h_i(\boldsymbol{x}, \boldsymbol{\theta}) = 0 \; \forall \; i = 1, \ldots, r \\ & \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p \\ & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \end{aligned} \tag{2.2}$$

In the case of parametric optimization, the objective function is optimized *as a function* of the parameter vector. In other words, $y^*(\boldsymbol{\theta})$ is a relation between $\boldsymbol{\theta}$ and $\min_{x \in X} f(\boldsymbol{x}, \boldsymbol{\theta})$. Thus, the solution rather than being a single point is a (potentially infinite) set. The constraints g_i and h_i are those that involve both design variables and parameters.

Figure 2.1 is an illustration of a solution in the 2-dimensional case. The shaded region is the feasible region in the objective space. The bold line is the maximum of the objective function for a given θ. Note, the x dimension not shown would be into the page.

The feasible parameter space may be defined simply by upper and lower bounds as in Figure 2.1 or by more general (nonlinear) constraints. In some applications, it may be useful to define constraints in terms of both $\boldsymbol{x}$ and $\boldsymbol{\theta}$. For example, the parameter may be a response of the design variables $\boldsymbol{\theta} : \mathbb{R}^n \to \mathbb{R}^p$. The constraint $\boldsymbol{\theta} = \boldsymbol{\theta}(\boldsymbol{x})$ would be represented as an equality constraint in Eq. 2.2.

Consider the scenario where two analysis models (each belonging to different

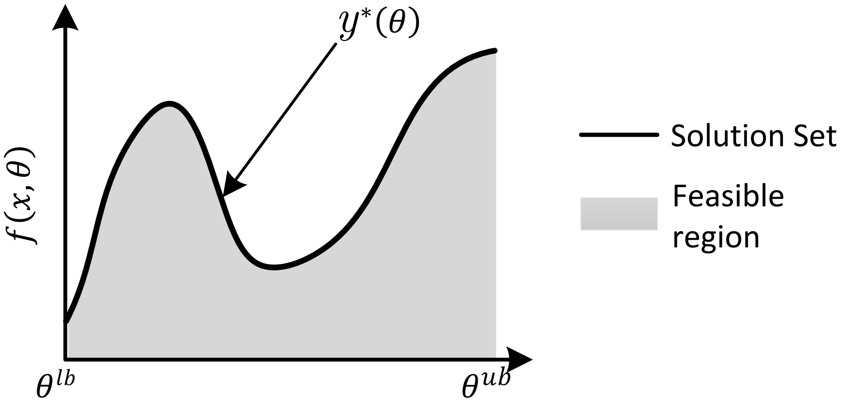


Figure 2.1: Notional illustration of a solution set to a parametric optimization problem. [a]

designers) may share an interface property. For example, a suspension model and chassis model may share a spring constant in their analysis. The spring constant is a function of design variables, e.g., wire thickness, number of coils, etc. In this case, the parameter space would be the set of feasible spring constants. The parametric optimum would be the best suspensions/chassis designs possible for every possible spring constant.

2.1.3 Parametric Optimization Example

Consider the standard optimization problem

$$y^* = \max_{-2 \leq x_1, x_2 \leq 2} f(x_1, x_2) \tag{2.3}$$

where $f(x_1, x_2) = 0.5 + x_1 e^{-x_1^2 - x_2^2}$. It is easy to see that $y^* = \frac{2+\sqrt{2e}}{2\sqrt{2e}}$ and occurs at $(x_1, x_2) = \left(\sqrt{\frac{1}{2}}, 0\right)$. See Fig. 2.2 for a graphical representation of the optimum.

[a] Reprinted with permission from “A Parallel Approach for Computing the Expected Value of Gathering Information” Galvan, E., Hsiao, C., Vermillion, S., & Malak, R., 2015. SAE Int. J. Mater. Manf., 8(2):271-282, Copyright 2015 by SAE International.

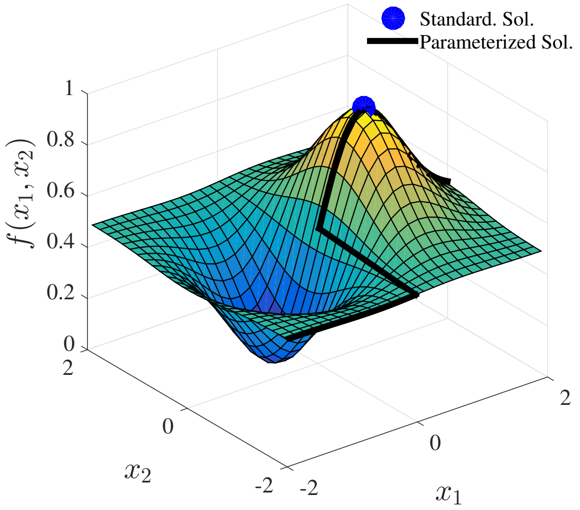


Figure 2.2: Comparison between standard and parametric optimization.

Say that we wish to "parameterize" this simple optimization problem along x_1 as

$$\begin{gathered} y^*(x_1) = \max_{-2 \leq x_2 \leq 2} f(x_1, x_2) \\ -2 \leq x_1 \leq 2 \end{gathered} \tag{2.4}$$

In other words, we want to find the x_2 that maximizes the objective function $f(x_1, x_2)$ for any $-2 \leq x_1 \leq 2$. See Fig. 2.2 for a graphical representation of the parametric solution. In this case, the parametric solution is an infinite set, a 1-dimensional “frontier” or surface.

2.2 Applications and Techniques for Parametric Optimization

In this research we have defined parametric optimization notionally as the process of determining how the solution to an optimization problem changes as a function of one or more variables, which we termed parameters. This general concept has applications in several fields, notably in economics and process engineering. The details of problem formulation and solution techniques vary from discipline to discipline. In this section, we provide a limited review of some of the applications and solution techniques for parametric optimization in the literature.

2.2.1 Economics: Comparative Statics

The concept of parametric optimization has been used previously in economics and [6, 7], typically with the aim of investigating solution sensitivity to one or a few parameter variables. This type of analysis is useful in understanding a system's response to uncertain parameters. For example, fluctuations in market requirements, and prices in economics, or boundary conditions, and system characteristics in engineered systems. When such fluctuating conditions are considered in an optimization problem, a parametric optimization technique can be used to understand their effect on the solution.

For example, in the area of economics known as comparative statics, one seeks to study how a firm's optimal strategy or the equilibrium of a market is affected by changes in an exogenous parameter such as the cost of labor [8]. Consider a firm's profit expression

$$\pi = \theta \cdot x - C(x) \tag{2.5}$$

where a firm chooses x to maximize profit, and $C(x)$ is the cost of producing x. The parameter θ represents market price of x. Comparative statics asks the question "How does the optimal choice change as a function of the parameter θ?" In Equation 2.5, the optimal argument as a function of the parameter is $x^*(\theta)$. By substituting the optimal argument into first order optimality conditions, we see that $dx^*(\theta)/d\theta > 0$, that is a profit maximizing firm increases output as market price increases. This result is perhaps obvious but it is illustrative of the analysis involved in comparative statics and the types of results one can expect. In general, comparative statics is interested in parametric problems of the form

$$\begin{aligned} \min_{x} \quad & f(x,\theta) \\ \text{subject to} \quad & x \in X \\ & \theta \in \Theta \end{aligned} \tag{2.6}$$

where one seeks to find x that minimizes $f(x,\theta)$ subject to the constraint that $x \in X$ for any value of the parameter $\theta \in \Theta$. In economics, parametric optimization focuses predominantly on theoretical results and simplified problem formulations (e.g., linear or quadratic approximations, monotonic relationships) to learn about behavioral trends (such as the output-price relationship) rather than exact information about an exact outcome.

2.2.2 Mathematics: Fuzzy Optimization

Another application of parametric optimization is so-called "fuzzy optimization." The approach was first proposed by Zimmermann [9] to solve the multiobjective optimization problem (or vectormaximum problem), where there are multiple objective that are simultaneously optimized. A fuzzy objective and constraints are characterized by their membership functions. Fuzzy optimization is intended to model situations where the designer cannot specify clearly the goal and constraints of the problem. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ be a vector of decision variables. The general fuzzy optimization problem can be stated as

$$\begin{aligned} \min_{\boldsymbol{x} \in X} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \lesssim 0 \ \forall \ i = 1, \ldots, q \end{aligned} \tag{2.7}$$

where '$\sim$' indicates imprecision on the part of the designer. The fuzzy formulation in Equation 2.7 can be formulated as a parametric optimization problem as [10]

$$\begin{aligned} \min_{\boldsymbol{x}\in X} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \leq \theta_i \ \forall \ i = 1, \dots, p \\ & \boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p. \end{aligned} \tag{2.8}$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_p)$.

2.2.3 Engineering: Process Optimization

More recently, the concept of parametric optimization has been applied to areas in process engineering. The design of engineered processes, such as the control of chemical processes, involve optimization for a fixed set of parameters, such as process coefficients, scheduling constraints, plant parameters, etc. However, these parameters are subject to change during operation such that the optimal solution at nominal parameter values ceases to be truly optimal. By recasting the problem as a parametric optimization problem, a designer can solve for the optimal as every value of the parameter. The result is an optimal "frontier" rather than a single point, see Figure 2.1. When the parameters deviate from their nominal value, one can obtain the corresponding optimal solution from the frontier rather than solving another optimization problem. This approach is particularly useful for control and operation problems, where the time delay in solving an optimization problem at each step is prohibitive for on-line corrective action. Thus, the goal is to reduce real-time optimization to a simple function evaluation. In process optimization, the parametric optimization problem is typically stated as

$$
\begin{aligned}
\boldsymbol{x}^*(\boldsymbol{\theta}) = \underset{\boldsymbol{x}}{\operatorname{argmin}} \quad & f(\boldsymbol{x}, \boldsymbol{\theta}) \\
\text{subject to} \quad & g_i(\boldsymbol{x}, \boldsymbol{\theta}) \leq 0 \;\forall\; i = 1, \ldots, q \\
& h_i(\boldsymbol{x}, \boldsymbol{\theta}) = 0 \;\forall\; i = 1, \ldots, r \\
& \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p \\
& \boldsymbol{x} \in X \subseteq \mathbb{R}^n
\end{aligned} \tag{2.9}
$$

where g_i and h_j are constraints involving both the design variables and parameters. Again, f is the objective attribute, $\boldsymbol{x}$ is a vector or design variables, and $\boldsymbol{\theta}$ is a vector of parameters. In the case of process optimization, the aim is to obtain $\boldsymbol{x}^*(\boldsymbol{\theta})$, the argument that minimizes in the objective as a function of the parameter. In this research, we are instead primarily interested in the objective function value of $\boldsymbol{x}^*$.

2.2.4 Collaborative Design

Another application of parametric optimization is in the area of collaborative design. In a collaborative design setting, the optimal performance of a system or subsystem will depend on some parameters that may be outside of the control of the designer, for example, changing constraints and requirements. In such cases, designers may be interested in finding not a single optimum but a family of optimal solutions as a function of the changing parameter value.

In [11], Wagner and Papalambros propose the use of the Parametric sensitivity analysis to investigate how the optimal performance of an engine varies with packaging objectives. The aim is to use the set of parametric optimal solutions as a point of negotiation between designers. Another relevant design method is Bilevel Integrated System Synthesis (BLISS) developed by Sobieszczanski et al [12]. Under

BLISS, the aim is to concurrently optimize an engineered system by decomposition. Each decomposed optimization problem is linked, resulting in an improvement of the objective at each iteration. However, the designs are linked using sensitivity analysis (how the solution changes in the *neighborhood* of the parameter) rather than parametric optimization (how the solution changes for *any* parameter value). Still, the aim is related in the sense that the design process is improved (in terms of converging to a more desirable solution with less effort or time) by knowing some information about how the performance of a subsystem changes with the parameter value.

2.2.5 Sensitivity Analysis Based Techniques for Parametric Optimization

In the literature, the most common technique approach to parametric optimization is to combine traditional optimization techniques for mathematical optimization of linear, quadratic, convex nonlinear, etc. with results in sensitivity analysis. Generally, sensitivity analysis aims at describing how the response of a model changes with the **perturbation** of a parameter. Parametric optimization is concerned with the optimum of a model changes for the **full range** of parametric variability.

In the literature, most sensitivity analysis based techniques for parametric optimization are limited to linear, quadratic problems, and mixed integer problems. However, within this field there exists a diverse range of algorithms and general results. The purpose of this section is not to provide the reader with a broad survey of the existing literature since several thorough surveys exist [13, 14, 15, 16, 17]. Instead the aim is to give the reader a general understanding of existing parametric optimization techniques and their limitations. To this end, we focus only on the multi-parametric Quadratic Programming (mp-QP) algorithm (see [15] for a detailed description).

First, we consider the general multi-parametric optimization problem

$$\begin{aligned}
\min_{\boldsymbol{x}} \quad & f(\boldsymbol{x},\boldsymbol{\theta}) \\
\text{subject to} \quad & g_i(\boldsymbol{x},\boldsymbol{\theta}) \leq 0 \ \forall \ i = 1,\ldots,q \\
& h_i(\boldsymbol{x},\boldsymbol{\theta}) = 0 \ \forall \ i = 1,\ldots,r \\
& \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p \\
& \boldsymbol{x} \in X \subseteq \mathbb{R}^n
\end{aligned} \tag{2.10}$$

where f, g, and h are twice continuously differentiable in $\boldsymbol{x}$ and $\boldsymbol{\theta}$. The first-order Karush-Kuhn-Tucker (KKT) conditions for optimality are

$$\begin{aligned}
& \nabla\mathcal{L} = 0, \\
& \lambda_i g_i(\boldsymbol{x},\boldsymbol{\theta}) = 0, \ \lambda_i \geq 0, \ \forall i = 1,\ldots,q, \\
& h_j(\boldsymbol{x},\boldsymbol{\theta}) = 0 \ \forall i = 1,\ldots,r, \\
& \mathcal{L} = f(\boldsymbol{x},\boldsymbol{\theta}) + \sum_{i=1}^{p} \lambda_i g_i(\boldsymbol{x},\boldsymbol{\theta}) + \sum_{i=1}^{r} \mu_i h_j(\boldsymbol{x},\boldsymbol{\theta})
\end{aligned} \tag{2.11}$$

Theorem 1. *Basic Sensitivity Theorem. Assume that (i) strict complementary slackness holds, that is, only one variable in every complementary pair $\mu_i h_j(\boldsymbol{x},\boldsymbol{\theta})$ is zero (ii) linear independence constraint qualification, and (iii) second-order sufficiency conditions (SOSC) hold. Let $(\boldsymbol{x}_0, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0)$ be a triple corresponding to the KKT conditions that satisfy Eq. 2.11 and 2.10 for some parameter value $\boldsymbol{\theta}_0$. In the neighborhood of $\boldsymbol{\theta}_0$, there exists a unique, differentiable function $\boldsymbol{z}(\boldsymbol{\theta}) = (\boldsymbol{x}(\boldsymbol{\theta}), \boldsymbol{\lambda}(\boldsymbol{\theta}), \boldsymbol{\mu}(\boldsymbol{\theta}))^T$*

that satisfies Eq. 2.11. at $\boldsymbol{\theta}_0$, with an isolated minimizer of Eq. 2.10, and

$$M(\boldsymbol{\theta}_0)\begin{pmatrix} \frac{d\boldsymbol{x}(\boldsymbol{\theta}_0)}{d\theta} \\ \frac{d\boldsymbol{\lambda}(\boldsymbol{\theta}_0)}{d\theta} \\ \frac{d\boldsymbol{\mu}(\boldsymbol{\theta}_0)}{d\theta} \end{pmatrix} = -N(\boldsymbol{\theta}_0) \tag{2.12}$$

where M and N are the Jacobian Matrix of Eq. 2.10 with respect to $(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ and $\boldsymbol{\theta}$, respectively

Proof of Theorem 1 can be found in [18]. The first order approximation of $z(\boldsymbol{\theta})$ in some neighborhood of $\boldsymbol{\theta}_0$ is

$$\boldsymbol{z}(\boldsymbol{\theta}) = \boldsymbol{z}(\boldsymbol{\theta}_0) + M(\boldsymbol{\theta}_0)^{-1}N(\boldsymbol{\theta}_0) \cdot \boldsymbol{\theta} \tag{2.13}$$

With these general results in mind, we now consider a quadratic optimization problem with parameterized linear constraints, where the parameters appear on the RHS of the constraints.

$$\begin{aligned} y^*(\theta) = \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^T Q\boldsymbol{x} \\ \text{subject to} \quad & A\boldsymbol{x} \leq \boldsymbol{b} + F\boldsymbol{\theta} \\ & \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p \\ & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \end{aligned} \tag{2.14}$$

where $\boldsymbol{c} \in \mathbb{R}^n$, Q is an $(n \times n)$ symmetric positive definite matrix, A is a $(q \times n)$ matrix, F is a $(q \times p)$ matrix, $\boldsymbol{b} \in \mathbb{R}^q$, and $X \subseteq \mathbb{R}^n$ and $\Theta \subseteq \mathbb{R}^p$ are compact polyhedral convex sets.

From Theorem 1, in the neighborhood of a feasible parameter value $\boldsymbol{\theta}_0$

$$\begin{pmatrix} \boldsymbol{x}_0(\boldsymbol{\theta}) \\ \boldsymbol{\lambda}_0(\boldsymbol{\theta}) \end{pmatrix} = -M(\boldsymbol{\theta}_0)^{-1} N(\boldsymbol{\theta}_0) \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}_0) + \begin{pmatrix} \boldsymbol{x}(\boldsymbol{\theta}_0) \\ \boldsymbol{\lambda}(\boldsymbol{\theta}_0) \end{pmatrix} \tag{2.15}$$

where 2.15, $\boldsymbol{x}_0$ is the optimum in the neighborhood of $\boldsymbol{\theta}_0$. We define the space of $\boldsymbol{\theta}$ where Eq. 2.15 remains optimal as the critical region, denoted R^0. The critical region is defined by the inequalities in Eq. 2.14 and the optimality condition that $\boldsymbol{\lambda}_0(\boldsymbol{\theta}) \geq \mathbf{0}$. Let the initial given region R^{ig} be defined by the linear inequalities in Eq. 2.14. The remaining region is $R^{rm} = R^0 - R^{ig}$. Another set of parametric solutions is found in R^{rm}. The procedure is repeated until termination criteria are met or all regions have been explored.

The mp-QP algorithm leverages sensitivity analysis techniques to solve a parametric optimization problem. The algorithm assumes linear constraints with the parameters on the RHS. As a result, the parameters must be additive linear terms. The mp-QP algorithm iteratively partitions the search space to find the set of optimal solutions as a function of the parameter. Ultimately, the solution is a set of piece-wise linear functions over the parameter.

2.3 Multiobjective Optimization

As stated in Section 1.1.1, our focus is on the use of optimization in decision problems where some preference information is not known. Specifically, we are concerned with the situation where *no preference information is available* about a parameter that is relevant for decision making, i.e., parametric optimization. Multiobjective optimization is related in the sense that it is concerned with the case where *only partial preference information is available.* In the Multiobjective case, the decision maker may know the preference ordering in each objective (e.g., to maximize or minimize) but does not know how to trade-off among the competing objectives. Thus, some preference information is missing in the decision problem.

Most real-world engineering optimization problems require that the designer simultaneously consider multiple objectives. Often, these engineering problems are cast as a multiobjective optimization (MO) problems, where the objectives typically conflict with one another. For example, for the design of an internal combustion engine, designers must balance efficiency, emissions, power, cost, etc.

The overall aim may be to select the design that increases the overall value of the project to the stakeholders. In the case of the design of an internal combustion engine, the project may be the design, production, marketing, sale, etc. of a vehicle. In this scenario, value may be increased by maximizing profit to the firm. For mathematical optimization in this context, a profit model that correlates all objectives to project value would be needed. However, correlation between objectives is often complex and difficult and expensive to elicit from the stake holder(s). How would one elicit the preferences of Toyota's investors for the internal combustion engine design? Even the simpler problem of expressing the system engineer's preferences may be impractical. As a result, so-called "a-posteriori" techniques for MO are common, where the aim is to find *all* solutions that maximize the objectives.

2.3.1 Mathematical Formulation

Without loss of generality, the multiobjective optimization problem can be stated as follows. Let $X \subseteq \mathbb{R}^n$ be the set of feasible decision vectors, denoted $\boldsymbol{x}$. Let $\boldsymbol{f} : X \to \mathbb{R}^m, \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$ be the vector valued objective function.

The multiobjective search problem is as follows [19]:

$$\begin{aligned} \boldsymbol{J}^* = \min \quad & \boldsymbol{f}(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \\ \text{where} \quad & \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \end{aligned} \tag{2.16}$$

The optimization problem has multiple objectives, $f_j(\boldsymbol{x})$, for $j = 1, 2, \ldots, m$, that must be minimized simultaneously. Like the PO problem, the MO problem has an infinite number of solutions.

2.3.2 Pareto Dominance

In the non-trivial case, a MO problem has multiple solutions that simultaneously optimize each objective. That is, no objective can be further improved without worsening at least one other. These solutions are said to be Pareto efficient. The concept of Pareto efficiency is named after Vilfredo Pareto (1848-1923), an engineer and economist who used the concept to study income distribution, political systems, resource allocation [20]. A Pareto efficient outcome is one where resources are allocated in such a way that it is impossible to make any individual better off without making at least one other individual worse off. Thus, a Pareto efficient outcome provides a minimal notion of efficiency. Those outcomes that are not Pareto efficient are said to be *Pareto dominated.* Thus, Pareto dominance is a mathematical decision rule for eliminating alternatives from consideration that could not be the most preferred. In engineering, the concept can be used to restrict selection to the set of alternatives to those that are Pareto efficient.

Consider a system with a vector function $\boldsymbol{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that maps the feasible set of alternatives $X \subseteq \mathbb{R}^n$ to the attribute space $\mathbb{R}^m$. Let $A = \{\boldsymbol{y} : \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{x} \in X\}$ be the set of feasible attribute vectors. Suppose preference direction is known for each attribute y_i for $i = 1, \ldots, r$. An alternative $\boldsymbol{y}'$ is preferred to another $\boldsymbol{y}''$ is denoted $\boldsymbol{y}' \succcurlyeq \boldsymbol{y}''$. Pareto dominance (PD) is defined as follows [21]:

Definition 9. *(Pareto Dominance) An alternative having attributes $\boldsymbol{y}'' \in A$ is Pareto dominated by one with attributes $\boldsymbol{y}' \in Y$ if and only if, $y_i' \succcurlyeq y_i''$ $\forall i = 1, \ldots, m$ and $y_i' \succ y_i''$ $\exists i = 1, \ldots, m$.*

The set of all Pareto non-dominated solutions is termed the Pareto frontier (PF).

2.4 Algorithms for Multiobjective Optimization

2.4.1 Epsilon-Constrained Method

One of the most widely used approaches a-posteriori methods for MO, is the ε-constraint method [22, 23]. In the ε-constraint method, the multiobjective problem is reformulated as several single-objective problems. One of the multiple objectives is arbitrarily selected to be the only objective and the remaining objectives are constrained. By systematically varying the constraints, the ε-constraint method achieves a relatively even distribution of points along the Pareto frontier.

Consider, the general multiobjective search problem in Section 2.3. Let $f_j(\boldsymbol{\theta})$ denote the primary objective, which is selected arbitrarily from the set of multiple objectives. The ε-constraint subproblem is

$$\begin{aligned} \min_{x} \quad & f_j(\boldsymbol{x}) \\ \text{subject to} \quad & f_i(\boldsymbol{x}) \leq \varepsilon_i \ \forall \ i = 1, \ldots, m \ i \neq j \\ & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \end{aligned} \tag{2.17}$$

The principal limitation of the ε-constraint method is a uniform distribution of constraints does not guarantee a uniform distribution of Pareto-optimal solutions.

To illustrate this limitation, consider the following simple MO problem

$$\begin{aligned} \min_{x_x, x_x} \quad & x_1,\ x_2 \\ \text{subject to} \quad & \left(\frac{x_2 - 20}{20}\right)^8 + \left(\frac{x_1 - 1}{1}\right)^8 \leq 1 \\ & 0 \leq x_1 \leq 1 \\ & 0 \leq x_2 \leq 20 \end{aligned} \tag{2.18}$$

If we let x_1 be the primary objective, the ε-constraint subproblems are

$$\begin{aligned} \min_{x_x, x_x} \quad & x_1 \\ \text{subject to} \quad & \left(\frac{x_2 - 20}{20}\right)^8 + \left(\frac{x_1 - 1}{1}\right)^8 \leq 1 \\ & 0 \leq x_1 \leq 1 \\ & x_2 \leq \varepsilon \end{aligned} \tag{2.19}$$

For this case, an equidistant sampling of ε produces the results illustrated in Figure 2.3.

2.4.2 Normal Boundary Intersection Method

The Normal Boundary Intersection Method (NBI) algorithm developed by Das and Dennis [24] is based on a scalarizing scheme to achieve "good" diversity along the solution frontier. The algorithm begins by establishing a utopia plane, which is a plane that passes through the individual minimizers of each objective function. In other words, one minimizes in each objective individually. The convex-hull of the resulting minimizers is the utopia plane. Next, one finds evenly distributed points along the utopia plane. Each of these points is minimized perpendicular to the utopia plane. The result is a nearly uniform distribution of points along the Pareto frontier.

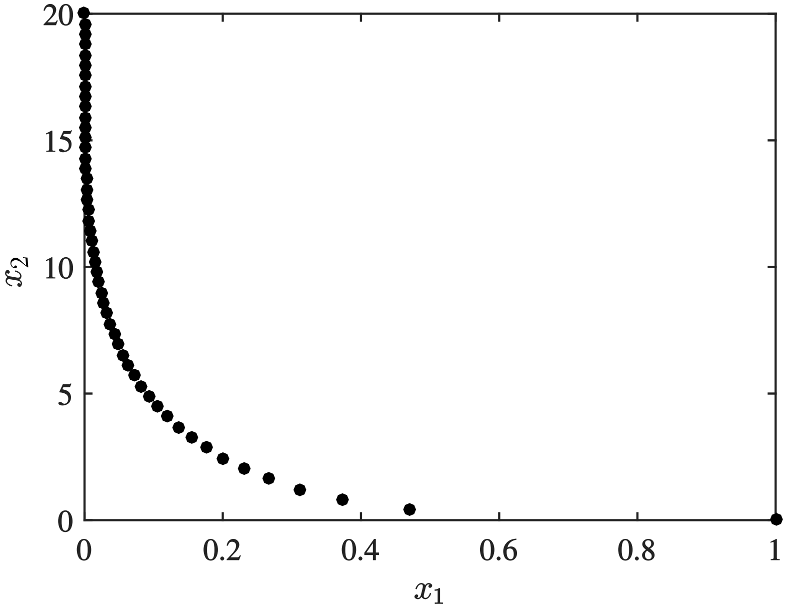


Figure 2.3: Solution to Equation 2.18 found using ε-constraint method with equidistant ε constraint values.

Figure 2.4 is an illustration of the NBI method in the 2 dimensional case.

Consider again the general multiobjective search problem in Section 2.3. Let $\boldsymbol{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$ be a vector of individual minimizers and $\boldsymbol{f}^* = (f_1^*, f_2^*, \ldots, f_m^*)^T$ be the corresponding objective attribute vector. The origin is shifted such that f^* is non-negative. Next, the convex hull of $\boldsymbol{f^*}$ is found, which can be expressed as the simplex $\boldsymbol{\phi\beta}$, where $\boldsymbol{\phi} = (\boldsymbol{f}(\boldsymbol{x}_1^*), \boldsymbol{f}(\boldsymbol{x}_2^*), \ldots, \boldsymbol{f}(\boldsymbol{x}_m^*))$ is an $m \times m$ matrix and $\boldsymbol{\beta} = \{(b_1, b_2, \ldots, b_m)^T | \sum_{i=1}^m b_i = 1\}$. Next, the $\boldsymbol{\beta}$ vectors are determined such that they are uniformly distributed along the utopia plane. Let $(\beta_1, \beta_2, \ldots, \beta_N)$ be the uniformly distributed vectors. The NBI subproblem (NBI$_{\boldsymbol{\beta}_i}$) for a given vector $\boldsymbol{\beta}_i$ is

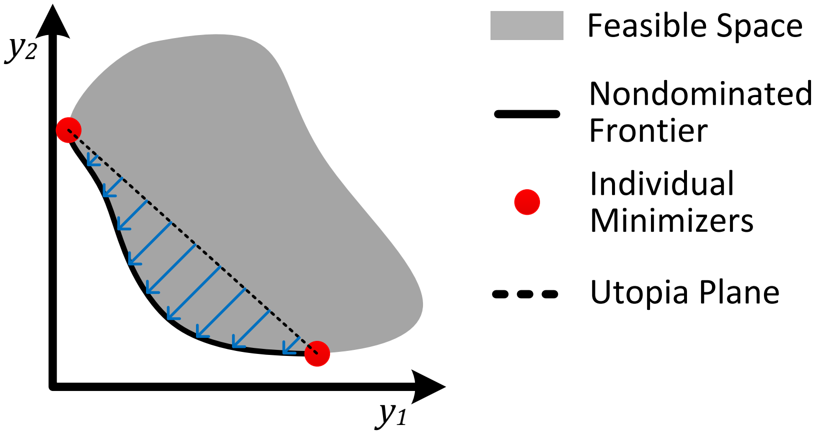


Figure 2.4: Illustration of the Normal Boundary Intersection Method. The blue arrows are perpendicular to and evenly distributed along the utopia plane

$$\begin{aligned} \max_{\boldsymbol{x},t} \quad & t \\ \text{subject to} \quad & \boldsymbol{\phi\beta} + t\hat{n} = \boldsymbol{f}(\boldsymbol{x}) \\ & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \\ & t \in \mathbb{R} \end{aligned} \tag{2.20}$$

where $\hat{n}$ is the normal direction at $\boldsymbol{\phi\beta}$ towards the origin. The principal difficulty in the NBI method is that, unlike the ε-constraint method, solutions to the NBI subproblems may not be Pareto-optimal. That is, the NBI method finds boundary points rather than Pareto-optimal ones. As a result, the NBI subproblem solutions must be filtered, typically using Pareto dominance.

2.4.3 Multiobjective Genetic Algorithms

In the MO literature, genetic algorithms are the most widely used to solve problems with complex objective functions [19]. Genetic algorithms (GAs) are population based meta-heuristic optimization algorithms. A GA uses concepts inspired by biological evolution, such as selection, reproduction, mutation, and recombination. A fitness function determines the quality of the solutions (i.e., the current population). The aim is to bias the subsequent generation towards the desirable region of the search space. In a multiobjective genetic algorithm (MOGA), the fitness value is assigned relative to two meta-objectives

1. Find a set of solutions close to the PF.
2. Maintain diversity in the solution set.

The advantage of the GA approach to multiobjective optimization is that by iterating over a population of solutions rather than a single point, an GA is able find a set of solutions in a single run [25]. Furthermore, via the recombination operator, a type of "knowledge sharing" among solutions is achieved by combining promising features to create new solutions in unexplored parts of the space.

Importantly, most MOGAs rely on the concept of *dominance* to assign higher fitness values to members closer to the PF. In these algorithms, two members are compared on the basis of whether one dominates the other or not. The nondominated member is closer to the PF and is assigned a higher fitness value. The most commonly used dominance rule is PD, but other dominance rules are possible.

The second goal is typically achieved through the use of diversity preservation operator. Then a termination condition is checked. If the termination criterion is not satisfied three main operators modify the population: selection, crossover, and mutation. Each new population is termed a generation. The following is a high-level description of a typical MOGA

1. Randomly initialize population.
2. Assign fitness based on dominance and a diversity operator.

3. If termination criteria have not been met, use selection, crossover, and mutation, operators to generate offspring and combine with parents. Else, terminate.

4. Return to step 2.

Several researchers have successfully approximated the set of solutions to MO problems using MOGAs [26, 27, 28, 29, 30, 31, 32, 33, 34]. A drawback of many MOGAs however, is that their performance deteriorates with the number of objectives. The principal cause of this loss in performance is that the fraction of the population that is Pareto non-dominated increases with the dimensionality of the objective space [35]. In order to better understand the ability of Pareto dominance to order solutions in terms of objective function values, we randomly generate 200 designs in an M-dimensional unit hyper-cube for $M = 2, 6, \ldots, 30$. The average percentage of non-dominated designs over 10 runs is presented in Figure 2.5. The percentage of non-dominated solutions increases with the number of objectives. For dimensions greater than 10, nearly all solutions are non-dominated. The high number of non-dominated solutions weakens the selection bias towards the true Pareto frontier. In the following subsection is a brief review of alternative dominance criteria.

Several *alternative* dominance criteria have been proposed to order to address the limitations of Pareto dominance. Alternative dominance criteria typically involve some modification of Pareto dominance intended to increase selection bias towards the Pareto frontier. For example, Pareto dominance may be modified as shown in Figure 2.6. Using (a) Pareto dominance all the alternatives are non-dominated. Using (b) modified dominance, alternative c is dominated. The modified dominance approach increases solution bias towards the true frontier but at the cost of losing diversity in the solution set [37].

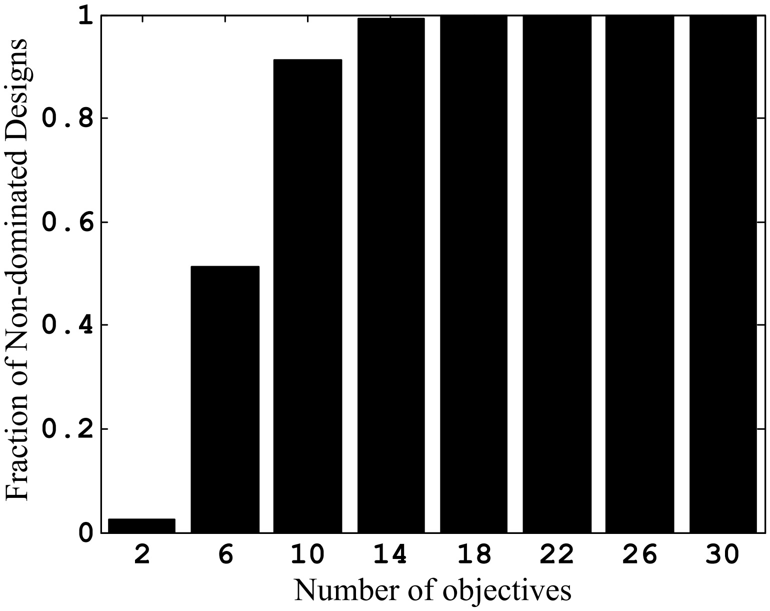
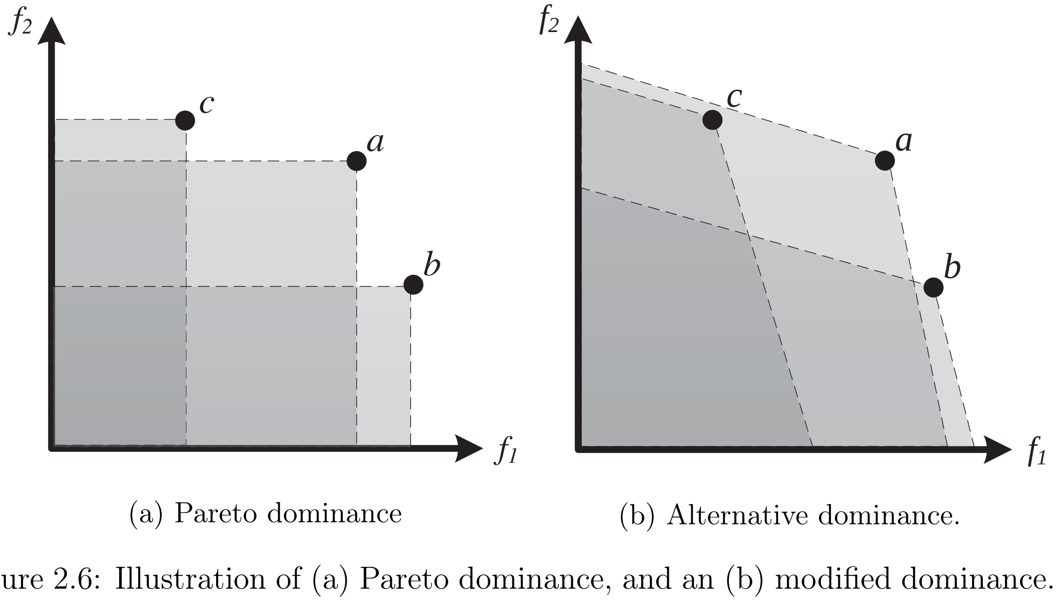


Figure 2.5: Mean fraction of predictive Pareto non-dominated designs from a set of 200 uniformly distributed designs in an M-dimensional unit hyper-cube. [36]



(a) Pareto dominance

(b) Alternative dominance.

Figure 2.6: Illustration of (a) Pareto dominance, and an (b) modified dominance. In each case, preferences are to minimize in each objective. The shaded region indicates the space dominated by the corresponding alternative. Alternative c is dominated only in (b). [38]

Again consider a system with a vector function $\boldsymbol{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that maps the feasible set of alternatives $X \subseteq \mathbb{R}^n$ to the attribute space $\mathbb{R}^m$. Let $A = \{\boldsymbol{y} : \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{x} \in X\}$ be the set of feasible attribute vectors. Without loss of generality, assume preference direction is to minimize each attribute y_i for $i = 1, ..., m$.

Under α-dominance lower and upper bounds are set for trade-off rates between two objectives such that solutions characterized by only small improvements in some objectives are rejected. This approach requires that the designer provide n^2 operating parameters that define the trade-off rate between the objectives, where n is the number of objectives. Another alternative, ε-dominance is a relaxation of strict dominance. There are several different versions of ε-dominance. Under additive *ε-dominance*, an alternative $\boldsymbol{y}' \in Y$ is ε-dominated by $\boldsymbol{y}'' \in Y$ if and only if, $y_i^{'} - \varepsilon \leq y_i''$ for all $i = 1, \ldots m$. As a result of their superior ordering ability, these dominance criteria have been shown to be more appropriate than Pareto dominance for many-objective problems [39]. However, the performance of the dominance criteria depends on values of several operating parameters specified by the designer.

The relation *favour* is based on the number of objectives for which one solution is better than another. An alternative $\boldsymbol{y}' \in Y$ is dominated by $\boldsymbol{y}'' \in Y$ if and only if the following relation holds $\left|\left\{j : y_j'' < \ y_i'\right\}, 1 \leq j \leq m\right| < \left|\{i : y_i' < \ y_i'\}, 1 \leq i \leq m\right|$. This dominance criteria introduces different ranks to the non-dominated solutions leading to increased selection bias towards the true Pareto frontier. However, it also results in a decrease in solution diversity, sometimes converging to very few solutions [40].

2.4.4 Quality Measure Directed Approaches

There exist in the literature a number of quality measures for the set of non-dominated solutions [41]. Several approaches form many-objective problems proposed in the literature directly optimize the quality measure. The hyper-volume indicator is a quality measure that is often used to assign fitness values to the members of the population. The hypervolume indicator is hypervolume of the space dominated by a solution set bounded by a reference point. Optimizing the hypervolume indicator is an attractive approach since: **(1)** it is a comprehensive search metric that captures convergence to the Pareto frontier and diversity in the solution set, and **(2)** it is the only known strictly monotonic quality measure with respect to the Pareto frontier. The hypervolume indicator of solution set S is larger than that of solution set S' if and only if S is better than S' [42]. Several hypervolume indicator based MOGAs have been proposed [43, 44, 45, 46]. The principal drawback of hypervolume indicator based approaches is the high computational expense for the hypervolume calculation; most algorithms are not feasible for problems with greater than 6 objectives. The HypE algorithm developed by Bader and Zitzler addresses this limitation by approximating the hypervolume using Monte Carlo sampling [47]. By approximating the hypervolume rather than computing it directly, the many-objective problems become feasible with hypervolume indicator based search algorithms.

2.5 Parametric Pareto Dominance

As discussed in the previous section, Pareto dominance (PD) criteria and can be applied to cases where *partial preference information is available.* Specifically, PD can be applied to decision problems with multiple competing objectives, i.e., the multiobjective case. A key benefit of the notion of PD is that one is guaranteed, from a decision-theoretic basis, not to eliminate any designs that could *potentially* be the most preferred. A limitation is that one must have a well-defined preference order over each objective attribute. As a result, PD cannot be applied in the case where one or more attributes is a parameter.

The *parameterized* Pareto dominance (PPD) rule is intended to address this limitation and is derived from a decision-theoretic basis and is mathematically sound in the sense that, like Pareto dominance, it will never eliminate an alternative that could be the most preferred solution. Thus, it captures all the potentially desirable alternatives [48, 4]. The relevant subset, termed the parameterized Pareto frontier (PPF), contains all points that potentially could be the most-preferred option in an engineered system.

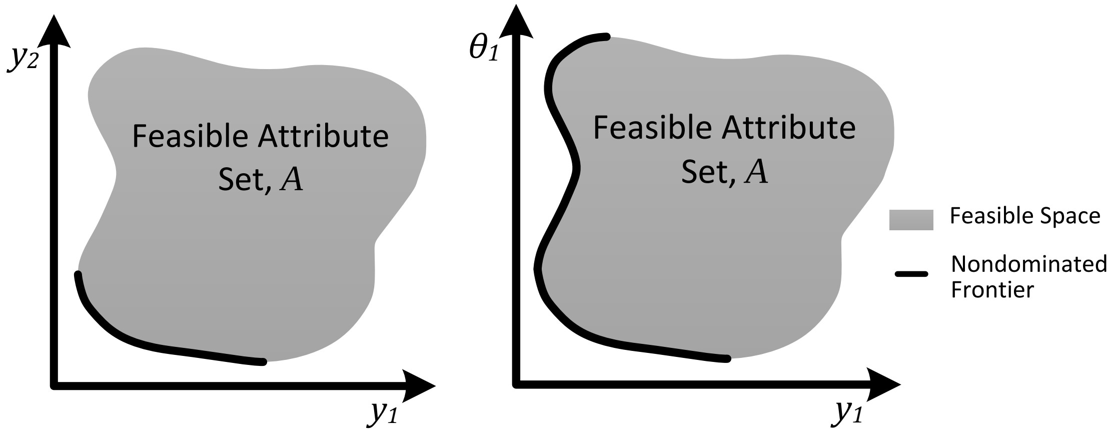
2.5.1 Mathematical Definition

Again, we consider a system with a vector function $\boldsymbol{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$. Let $\Theta \subseteq \mathbb{R}^p$ be the set of parameter vectors, denoted $\boldsymbol{\theta}$. In the parametric case, we define the attribute space more generally as the space of combined objective and parameter attributes $A = \{(\boldsymbol{y}, \boldsymbol{\theta}) : \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}), \boldsymbol{x} \in X, \boldsymbol{\theta} \in \Theta\}$. Suppose preference direction is known for each objective attribute y_i for $i = 1, \ldots, m$. The mathematical formulation of parameterized Pareto dominance (PPD) is as follows [48]:

Definition 10. *(parametric Pareto dominance) An alternative having attributes* $(\boldsymbol{y}'', \boldsymbol{\theta}'') \in A$ *is parametrically Pareto dominated by one with attributes* $(\boldsymbol{y}', \boldsymbol{\theta}') \in A$ *if and only if* $\theta'_i = \theta''_i \ \forall i = 1, \ldots, p,\ y'_j \succcurlyeq y''_j \ \forall j = 1, \ldots, m$ *and* $y'_j \succ y''_j \ \exists j = 1, \ldots, m.$

This definition can be considered an extension of PD to the case involving parameters, which are those attributes over which the designer does not have preference over. The Pareto frontier is the set of feasible alternatives that are not Pareto dominated. The Pareto frontier is the set of solution to the multiobjective optimization problem. Similarly, the solution to a parametric optimization problem is those that are not parametrically Pareto dominated.

Figure 2.7 is a comparison of PD and PPD. In Figure 2.7a preferences are to



(a) Classic Pareto Dominance. Minimize y_1 and y_2

(b) Parameterized Pareto Dominance. Minimize y_1; parameter θ_1

Figure 2.7: Comparison between (a) PD and (b) PPD. In this example, there are $m = 2$ attributes; the feasible attribute set $A \subset \mathbb{R}^2$.[b]

minimize both attributes (both are objectives) resulting in a PF. In Figure 2.7b preference is to minimize in y_1 (dominator attribute) but the preference order in θ_1 is unknown (parameter attribute). In this case, one can use PPD to identify inferior alternatives. The result is a PPF, which can be used as an efficient representation (not containing irrelevant alternatives) of the capabilities of a subsystem.

It is important to note that PPD is not an **alternative** dominance criterion such as *α-dominance, ϵ-dominance*, or *favour*, [49, 50, 25]. Alternative dominance criteria typically involve some modification of PD and are not derived from a decision-theoretic foundation. As a result, "dominated" solution under an alternative dominance criterion may actually be the most preferred. On the other hand, any alternative dominated under PPD provably cannot be the most preferred solution. Additionally, for any non-dominated alternative there exists some aggregating func-

[b] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

tion for which it is the most preferred solution.

2.5.1.1 Parametric Pareto Dominance for Capability Modeling

It is important for engineers to understand the capabilities and limitations of the technologies they consider for use in their systems. Failure to appreciate what is achievable (or not achievable) by subsystem technologies can result in the selection of poor concepts, the derivation of poor design requirements (that are either unachievable or overly conservative), and excessive design iteration. However, it can be challenging for engineers to navigate the many competing considerations that arise in the development of large systems that have many interacting subsystems.

Several researchers have investigated approaches for modeling the capabilities of a technology quantitatively with the aim of supporting technology selection, design space exploration, and trade-off analyses. A common thread among these works is that in order to model the capabilities of a technology, one abstracts away information about its physical form. Mathematically, the result is a model defined in the attribute space for a given system. With these models, designers can represent the abstracted capabilities, performance traits, or metrics of a component or system.

It can be advantageous for engineers to use abstract models of competing technologies. Abstract models help engineers focus on the part of a problem most germane to decision making (i.e., what a subsystem can or cannot achieve rather than how it achieves it). Abstract models can also provide engineers with the capacity to consider multiple physically heterogeneous technologies in the same variable space. For example, although batteries and fuel cells are very different physically, engineers can consider them both as energy storage technologies with the same attributes such as power density, energy density, and efficiency. Furthermore, abstract models can be desirable in a multi-organizational collaborative design setting. Because they abstract away low-level implementation details, they tend to hide proprietary information that collaborating organizations may not want to share.

Several approaches have been proposed in the literature for capturing and modeling the capabilities of a component or system. Under set-based design approaches to capability modeling, one uses a mathematical representation of the set of performance attributes that are technically feasible [51, 52]. Initially, designers catalog components or designs into a hierarchical structure. Then, designers abstract the performance attributes of the feasible set of designs to form descriptions for use in higher-levels in the hierarchy. The union of the basic sets represents the achievable performance characteristics of the system. Designs are eliminated from consideration through propagation of interval constraints in the performance space [53]. Taking an engine design example, the constraints may be design specifications for airflow rate, horsepower, torque, and so on. Finally, designers use some cost function to select one of the remaining designs. A drawback of this approach is that designers must make restrictive assumptions about what is desirable in the performance space. When design is done by this approach, many degrees of freedom are removed from the design problem. This can potentially result in eliminating the most preferred design from consideration [54]. Furthermore, as the constraints are made more complex, the inherent computational complexity of propagating the constraints increases rapidly [53].

Another approach involves the use of the PD. One models the capabilities of a system constructs as the subset of alternatives that are Pareto non-dominated [55, 56, 57, 58, 59, 60, 61]. The benefits of this approach are that the dominance analysis step is relatively computationally inexpensive and will not eliminate the most preferred design [62]. A limitation, for the purposes of capability modeling, is that designers must have a preference order over each attribute, e.g., prefer to maximize or minimize.

Although such a formulation is likely at the **system-level**, it is unlikely at the **discipline-level** due to the presence of parameter attributes. For example, consider the problem of modeling the capabilities of a spring damper subsystem that is to be a part of some larger system. An important attribute of the design of a spring damper subsystem is the spring constant. The most preferred spring constant will depend on the details of the larger system. In this scenario, it is insufficient to optimize for cost, mass, reliability, etc. without considering spring constant. Thus, at the discipline-level, designers often wish to represent a system in terms of parameters, e.g., system-level specifics, variables shared between designers, etc. In such cases, PPD is a more appropriate dominance rule.

In [48, 4], Malak and Paredis show that PPD is sound from a decision making perspective; that is, it does not eliminate any designs that may be the most preferred. Parameterized Pareto dominance requires fewer assumptions than PD, which results in capability models that are more widely reusable and that can be composed to model a larger system. These benefits have been demonstrated on various mechanical, power generation, and fluid power system design studies [48, 4, 63, 64].

2.6 Conclusions and Chapter Summary

In this chapter, we have defined the parametric optimization problem and compared it to the single and multi-objective cases. The principal difference between a traditional single objective optimization (SO) problem and a parametric one (PO) is the condition that a parametric problem is solved as a function of the variable (potentially a vector) θ.

We provide Table 2.1 to help the reader better understand POs in the context of more familiar search problems: MOs and SOs. In SO, all preference information is

Table 2.1: Comparison of the information required for the formulation of POs, MOs, and SOs as well as their solutions.

	Available Information	Search Problem	Solution
More Information ↓	Preference is unknown in at least one attribute (parameter) and known in others (dominators)	PO	PPF
	Preference ordering is known in all attributes	MO	PF
	All preference information is known and well defined	SO	Point Solution

available and each alternative can be ranked. In MO, some preference information is missing, specifically how to trade-off between some attributes relevant to decision making (multiple conflicting objectives). The solution to a MO problem is the PF. Finally, PO can be thought of as the most general case where no preference ordering ordering is available for at least one attribute. The solution to the PO is the parameterized Pareto frontier (PPF).

The general notion of parametric optimization has been applied in fields such as economics and process optimization. Existing techniques for parametric optimization are exact but limited to problems with specific mathematical structures: linear, quadratic, convex nonlinear. This is a significant limitation for engineering design challenges where the models are often highly nonlinear (dynamical systems) or non-analytical (finite element analysis, simulations involving conditionals).

We also introduced MO and some common MO algorithms, since the problem of finding the Pareto frontier (MO) is analogous to the problem of finding the parametric Pareto frontier (PO). Specifically we explored the ε-constraint method and the Normal Boundary Intersect Method, a genetic approach and a quality directed approach.

3. PARAMETRIC OPTIMIZATION FOR SYSTEMS DESIGN

The first step in this research is to define useful parametric problem formulations motivated by systems design. Once the problem is formally defined, an algorithmic solution approach can be developed to exploit the parametric structure and specific features of the problem.

In this chapter, we will demonstrate the use of PO to compute the Expected Value of Information (EVI). We will recast the EVI problem as a PO with the aim of reducing computational expense. Next, we extend the definition of PO to the case with multiple objectives. We term this formulation multiobjective parametric optimization (MPO). The motivation for this formulation is capability modeling, since many technologies cannot be fully characterized by a single attribute.

Th MPO form is a novel formulation and cannot be expressed using the parametric optimization formulations from economics, process optimization, or fuzzy optimization. We will show that the MPO formulation is necessary if we are to describe the capabilities of a subsystem or technology in terms of it responses.

3.1 Single Objective Parametric Optimization

3.1.1 Mathematical Formulation

The Single-objective Parameterized optimization problem can be stated as

$$\begin{aligned} y^*(\boldsymbol{\theta}) = \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} \quad & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \\ & \boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p \end{aligned} \tag{3.1}$$

where p is the number of parameters, and q and r are the number of inequality and equality constraints, respectively. Note that minimization is used in the general SPO formulation since maximizing $f(\boldsymbol{x})$ is equivalent to minimizing $-f(\boldsymbol{x})$. The p parameter(s) are denoted $\boldsymbol{\theta}$ and Θ is the parameter space. The objective function is optimized as a function of the parameters. In other words, the solution to Eq. 3.1 is the maximum of $f(\boldsymbol{x}, \boldsymbol{\theta})$ for every value of $\boldsymbol{\theta}$ in the parameter space. Thus, the solution rather than being a single point is a (potentially infinite) set, denoted $y^*(\boldsymbol{\theta})$.

3.1.2 Engineering Relevance: Expected Value of Information

The design of complex engineered systems requires decision making under uncertainty. In this context, good decisions are characterized careful management of that uncertainty. The systems engineer should account for all available and relevant information to deduce which decision alternative is "best." A class of alternatives that is often overlooked is information activities to reduce the uncertainty. Information gathering activities might include prototype development, increasing the sample size of an experiment, conducting more expensive tests, etc. Thus, the first step in deciding whether or not to perform an information gathering action is to compute its value. Engineering firms must be able to assess the benefit of performing information-gathering actions. For example, an automobile manufacturer may use a computer simulation of a hydraulic motor and pump in the design of a new vehicle. The model may contain random variables that can be more accurately determined through expensive experiments (information-gathering actions). The designer must choose to either design the transmission given the current state of information or perform an information-gathering action to reduce uncertainty at a cost. The designer must determine whether the benefit of gathering information is greater than the cost of gathering that information. Standard decision making approaches tend to neglect the information-gathering aspects of the decision making process, instead focusing exclusively on decision making with current information. Good engineering practice must take a broader view of decision making and consider the value of gathering information.

A framework for considering the resources involved in gathering information with how it improves decision making is Expected Value of Perfect Information (EVPI) [65, 66]. The EVPI is the price one is willing to pay for perfect information. One examines how much the expected reward would improve if the uncertain information were perfectly accurate. The expected increased reward is compared to the cost of gathering that information to determine whether gathering the information is worthwhile. Thus, the primary decision in an EVPI problem is choosing whether to gather information. For clarity, choosing what information to gather will be called the *information decision* while choosing which alternative based on that information will be called the *design decision*. In these types of problems, the information decision is made before the design decision, since the design decision depends on the outcome of the information decision.

Expected Value of Partial Perfect Information (EVPPI) is an extension that considers multiple random variables. Under EVPPI, the engineer chooses the set of parameters for which to gather perfect information that maximizes the engineer's expected reward. However, many sources of information are not perfect, but can have uncertainty themselves. This may be due to the information-gathering method itself or with its predictive power on the parameter(s) of interest. Additionally, the cost of information-gathering typically increases dramatically with experimental accuracy, thus in practice uncertainty is typically not reduced to zero. Instead, the amount of uncertainty reduced is traded off against the cost of reducing the uncertainty. EVPI does not account for this. Instead, as a practical matter, the value for having perfect information sets an upper bound on the value of gathering information.

In contrast to EVPI, the Expected Value of Sample Information (EVSI) framework does account for a reduction in the uncertainty based on an imperfect source of information. Each information-bearing test or sample gives information about a parameter of interest, but does not reveal the parameter directly. The designer takes this uncertainty into account in calculating the expected value of information. A deeply related problem is deciding not just whether or not to collect a sample, but how many samples to collect as a batch. This approach is used in applications such as medical clinical trials, whether the number of subjects needed to test an experimental drug is decided in advance of the actual trials.

Although the EVPI framework has been extensively studied, a practical challenge to is that the cost of computing the expected value of information (through optimization, Monte Carlo sampling, etc.) grows exponentially with the amount of information that is to be gathered and can often exceed the cost of actually gathering the information. One might suggest that rigor requires a decision method that accounts for this cost, that is, an "expected value" of computing the expected value of information gathering. This view leads to a type of an infinite regress, where to evaluate the alternatives one must factor in a "meta" analysis of the benefit of evaluating the alternatives, and so on. A practical resolution to this problem is to assume that the cost of computing the expected value of information is negligible. Thus, if information decisions are to be addressed algorithmically, there exists a need for novel algorithmic approaches to reduce the computational expense associated with computing the expected value of gathering information.

The contribution of this section is a novel algorithmic approach for approximating the expected value of perfect information (EVPI) for engineering design problems. In this research, we propose to recast the EVPI as a "parametric" problem. The value of recasting the problem is an exponential reduction in the computational complexity.

The proposed approach is validated against a Monte Carlo sampling based approach, the traditional approach for solving EVPIs, on an engineering problem. The engineering problem is to compute the expected value of performing physical experiments to gather information about random variables in a computational model the efficiency of an engine and transmission. The results are compared in terms of computational expense and solution accuracy. The results indicate that by recasting the EVPI as a parametric problem the computational expense is reduced drastically while maintaining solution accuracy.

3.1.3 Engineering Application: Expected Value of Perfect Information

In this research, information is considered to be perfect, and thus falls under the EVPI framework rather than the EVSI framework. However, it should be noted that this is simply to simplify the exposition; the concepts described herein can be readily extended to more complex frameworks. We demonstrate one such extension to the Expected Value of Partial Perfect Information in the following section.

We begin with some notation. Let $\boldsymbol{x}$ be a vector denoting the design variables (those which the designer can control), which must belong to the nonempty feasible region (set) X. Let $\boldsymbol{\theta}$ be a vector of parameters with probability distribution $p(\boldsymbol{\theta})$ that are outside of the control of the designer. Let Θ be the feasible subset of parameter vectors in the parameter space.

The value (or profit, benefit, etc.) model $f : X \times \Theta \to \mathbb{R}$ maps the Cartesian product of the decision space and parameter space to the payoff (profitability) of the design. In other words, the value function relates design details and random variables to value of the design. Again, we note that in a typical EVPI problem, the set of information gathering alternatives is discrete. Equation 3.2 is an extension to the case with an continuous range of alternatives.

If the designer chooses not to gather information, she will chose $\boldsymbol{x}$ such that expected value is maximized

$$\max_{\boldsymbol{x}\in X} \int f(\boldsymbol{x},\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} \tag{3.2}$$

The solution to Eq. 3.2 is the expected value of designing the artifact without gathering information. The argument that maximizes Eq. 3.2 is an instance of $\boldsymbol{x}$, i.e., a design concept. If instead, she chooses to gather information, she will optimize the design (make the design decision) with respect to the revealed value of the random parameter, denoted $\boldsymbol{\theta}_R$.

$$\max_{\boldsymbol{x}\in X} f(x,\boldsymbol{\theta}_R) \tag{3.3}$$

The optimal solution to the design problem (choice of $\boldsymbol{x}$) in this case is dependent on $\boldsymbol{\theta}$. Thus, the expected reward if the uncertain parameter were revealed is

$$\int \Big(\max_{\boldsymbol{x}\in X} f(\boldsymbol{x},\boldsymbol{\theta})\Big)p(\boldsymbol{\theta})d\boldsymbol{\theta} - C \tag{3.4}$$

where C is the cost of gathering information. The EVPI is how much the expected reward improves if the uncertain information were known perfectly accurately. n

$$EVPI = \int \Big(\max_{\boldsymbol{x}\in X} f(\boldsymbol{x},\boldsymbol{\theta})\Big)p(\boldsymbol{\theta})d\boldsymbol{\theta} - \max_{\boldsymbol{x}\in X}\Big(\int f(\boldsymbol{x},\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}\Big) - C \tag{3.5}$$

Unlike in the decision problem, the outcome of Eq. 3.5 is not a design concept (choice of $\boldsymbol{x}$). Instead, the aim is to approximate the value of gathering perfect information. If $EVPI > 0$, the decision that maximizes value to the firm is to gather information. In that case, the optimal $\boldsymbol{x}$ will depend on the outcome of

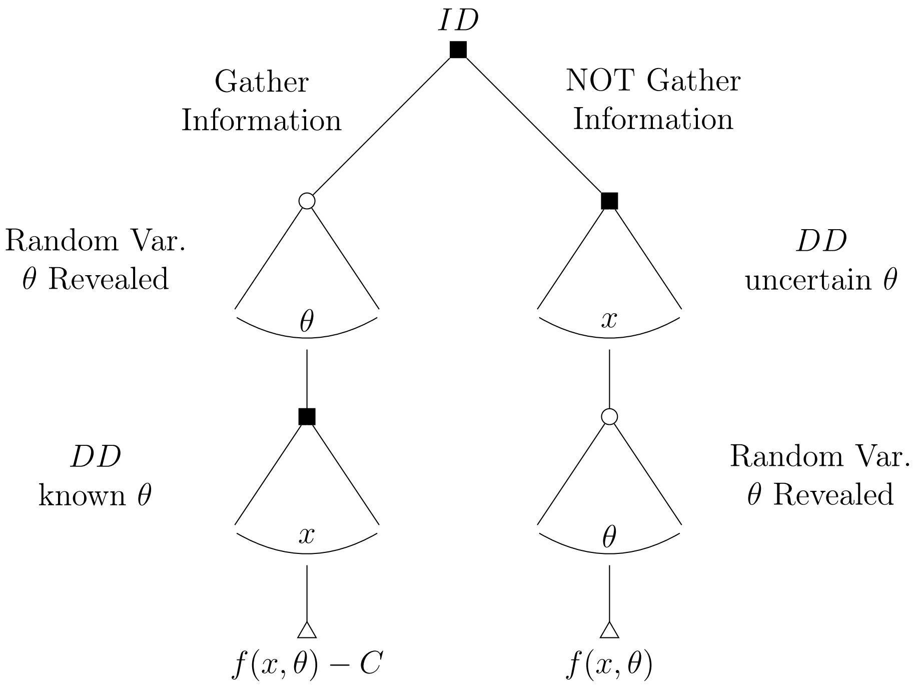


Figure 3.1: Decision Tree for information decision (ID). The arcs represent branches where a continuum of outcomes or actions are available. Decision nodes represent design decisions (DD) and are filled for emphasis.

the information gathering action, which is unknown at this stage. Figure 3.1 is an illustration of the information decision (ID) problem as a decision tree.

Traditional methods of optimization under uncertainty can be used to numerically approximate Eq. 3.5

$$EVPI \approx \frac{1}{N}\sum_{i=1}^{N}\Big(\max_{\boldsymbol{x}\in X} f(\boldsymbol{x}, \boldsymbol{\theta}_i)\Big) - \max_{\boldsymbol{x}\in X}\Big(\frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}, \boldsymbol{\theta}_i)\Big) - C \qquad (3.6)$$

where $\boldsymbol{\theta}_i \sim P(\boldsymbol{\theta})$ and N is the number of samples to draw to approximate the expectation. Note that the first term, in Eq. 3.6 involves an optimization for each of the N instances of $\boldsymbol{\theta}_i$ [67]. Many algorithmic approaches for solving EVI problems are aimed at scenarios where X is a discrete set [68, 69]. In this dissertation, we are instead focused on the case more relevant to engineering design where X is a continuous set with infinitely many alternatives. If each optimization is costly, a simple Monte-Carlo approach for approximating $EVPI$ may be prohibitively expensive. Consider the following. Let k be the number of iterations required to solve each optimization problem. The number of evaluations of $f(\boldsymbol{x}, \boldsymbol{\theta})$ required to solve Eq. 3.5 would be $2kN$. Even with relatively inexpensive engineering models and parallel computing, the cost of analyzing the decision problem may even exceed the cost of actually gathering the information (physical experiments, surveys, etc.). Thus, if information decisions are to be addressed algorithmically, there exists a need for a novel algorithmic approach to reduce the computational expense.

In this research, we propose that the iterated optimization in the approximation of $EVPI$ can be eliminated if recast as a parametric optimization problem as

$$\begin{aligned} y^*(\boldsymbol{\theta}) = \max_{\boldsymbol{x} \in X} \;\; & f(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} \;\; & \boldsymbol{\theta} \in \Theta \end{aligned} \tag{3.7}$$

The aim is to find a solution, denoted y^*, that maximizes the objective function f, for every $\boldsymbol{\theta} \in \Theta$. We can then substitute $y^*(\boldsymbol{\theta})$ into Equation 3.6

$$EVPI \approx \frac{1}{N}\sum_{i=1}^{N} y^*(\boldsymbol{\theta}_i) - \max_{\boldsymbol{x} \in X}\Big(\frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}, \boldsymbol{\theta}_i)\Big) - C \tag{3.8}$$

eliminating the “nested” optimization in the first term on the RHS. Recasting the problem in such a way eliminates the need to perform optimization within a Monte Carlo simulation. Instead of solving N optimization problems (where N is the number of Monte Carlo samples), we need only solve a single parametric optimization problem.

3.1.4 Engineering Application: Expected Value of Partial Perfect Information

In the previous section, information is considered to be perfect, and thus falls under the EVPI framework. However, the concepts described in this paper can be readily extended to account for Expected Value of Partial Perfect Information (EVPPI). Expected Value of Partial Perfect Information is an extension that considers gathering information on multiple random variables. Under EVPPI, the engineer chooses the set of parameters for which to gather perfect information that maximizes the engineer's expected reward.

The primary decision in an EVPPI problem is to determine for which random variables to gather information. This is in contrast to the EVPI problem, where the only information-gathering alternative is to gather information on a specific set of random variables.

To consider the information that may be gathered, let the subscript O (as in observation) denote the set of the random variables about which the engineer gathers perfect information, while the subscript C denote its complement, the set of uncertain parameters about which information is not gathered. Also, let the uncertain parameters for these sets be denoted respectively as $\boldsymbol{\theta}_O$ and $\boldsymbol{\theta}_C$, such that $\boldsymbol{\theta}_O \cup \boldsymbol{\theta}_C = y$ and $\boldsymbol{\theta}_O \cap \boldsymbol{\theta}_C = \varnothing$. Thus, O denotes the set or indices of the uncertain parameters, while $\boldsymbol{\theta}_O$ denotes the parameters themselves. Let the cost of gathering information about that set of parameters be denoted as C_O. For simplicity, assume that the uncertain parameters are independent, although they need not be. It is straightforward to extend this theory to the case where they are not, although the computation becomes more challenging.

In order to evaluate the value of gathering information on some set of parameters

O, the engineer must consider the possible outcomes of that information-gathering. Then, for each of those possible outcomes, choose $\boldsymbol{x}$ such that the expected value across the remaining uncertain parameters is maximized. The expected value of gathering perfect information on parameters O is formulated as follows

$$\int \max_{\boldsymbol{x}\in X}\left(\int f(\boldsymbol{x},\boldsymbol{\theta})p(\boldsymbol{\theta}_C)d\boldsymbol{\theta}_C\right)p(\boldsymbol{\theta}_O)d\boldsymbol{\theta}_O - C_O \tag{3.9}$$

An exhaustive approach for approximating Eq. 3.9 involves many nested search problems. For each possible value of $\boldsymbol{\theta}_O$ that can be received from gathering information, one must search for the $\boldsymbol{x}$ that maximizes the engineer's expected value across the remaining uncertain parameters $\boldsymbol{\theta}_C$, the optimization problem in the brackets. As a result, determining the expected value involves a nested Monte Carlo simulation. The "outer" simulation is for the different possible values of $\boldsymbol{\theta}_O$ that the designer may learn. The "inner" simulation is for, given that particular value of $\boldsymbol{\theta}_O$, the possible values of $\boldsymbol{\theta}_C$ that the engineer considers in selecting $\boldsymbol{x}$. This computation may be prohibitively expensive even for a small number of variables. Furthermore, the engineer must decide *across all* the possible combinations of parameters O that he can gather information about, since the choice is about which combination of parameters to select. Therefore, the full EVPPI decision problem is

$$\max_{O}\left(\int \max_{\boldsymbol{x}\in X}\left(\int f(\boldsymbol{x},\boldsymbol{\theta})p(\boldsymbol{\theta}_C)d\boldsymbol{\theta}_C\right)p(\boldsymbol{\theta}_O)d\boldsymbol{\theta}_O - C_O\right) \tag{3.10}$$

If O were the empty set , then this problem reduces to the traditional design decision from Eq. 3.5. Figure 3.2 shows the decision tree for this problem. In the figure, O_1 and O_2 represent different subsets of parameters about which information can be gathered. The total number of branches depends on the number of observable parameters.

Let N be the number of Monte Carlo samples necessary to obtain the parameter distribution, M be the number of available experiments (information-gathering alternatives), and K be the number of steps required to solve the search problem. The number of evaluations of $f\left(\boldsymbol{x},\boldsymbol{\theta}\right)$ required to solve Eq. 3.10 using a Monte Carlo Method would be $MK(N^2+1)$. Applying the parametric approach described in this paper eliminates the nested Monte Carlo simulation reducing the number of required function evaluations exponentially to $MK(N+1)$, an exponential decrease in computation.

This nested structure exists in many EVI formulations, beyond EVPI. For example, Ling et al. [70] note that the nested Monte Carlo simulation in their p-box approach for EVI problems may become prohibitively expensive for more complex problems.

3.2 Multiobjective Parametric Optimization

3.2.1 Mathematical Formulation

The Multiobjective Parametric Optimization (MPO) problem is stated as

$$\begin{aligned}
y^*(\boldsymbol{\theta}) = \min_{\boldsymbol{x}} \quad & \boldsymbol{f}(\boldsymbol{x},\boldsymbol{\theta}) \\
\text{subject to} \quad & g_i(\boldsymbol{x},\boldsymbol{\theta}) \leq 0 \; \forall \; i = 1,\ldots,q \\
& h_j(\boldsymbol{x},\boldsymbol{\theta}) = 0 \; \forall \; j = 1,\ldots,r \\
& \boldsymbol{x} \in X \subseteq \mathbb{R}^n \\
& \boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p \\
\text{where} \quad & \boldsymbol{f}(\boldsymbol{x},\boldsymbol{\theta}) = (f_1(\boldsymbol{x},\boldsymbol{\theta}), f_2(\boldsymbol{x},\boldsymbol{\theta}),\ldots,f_m(\boldsymbol{x},\boldsymbol{\theta}))
\end{aligned} \tag{3.11}$$

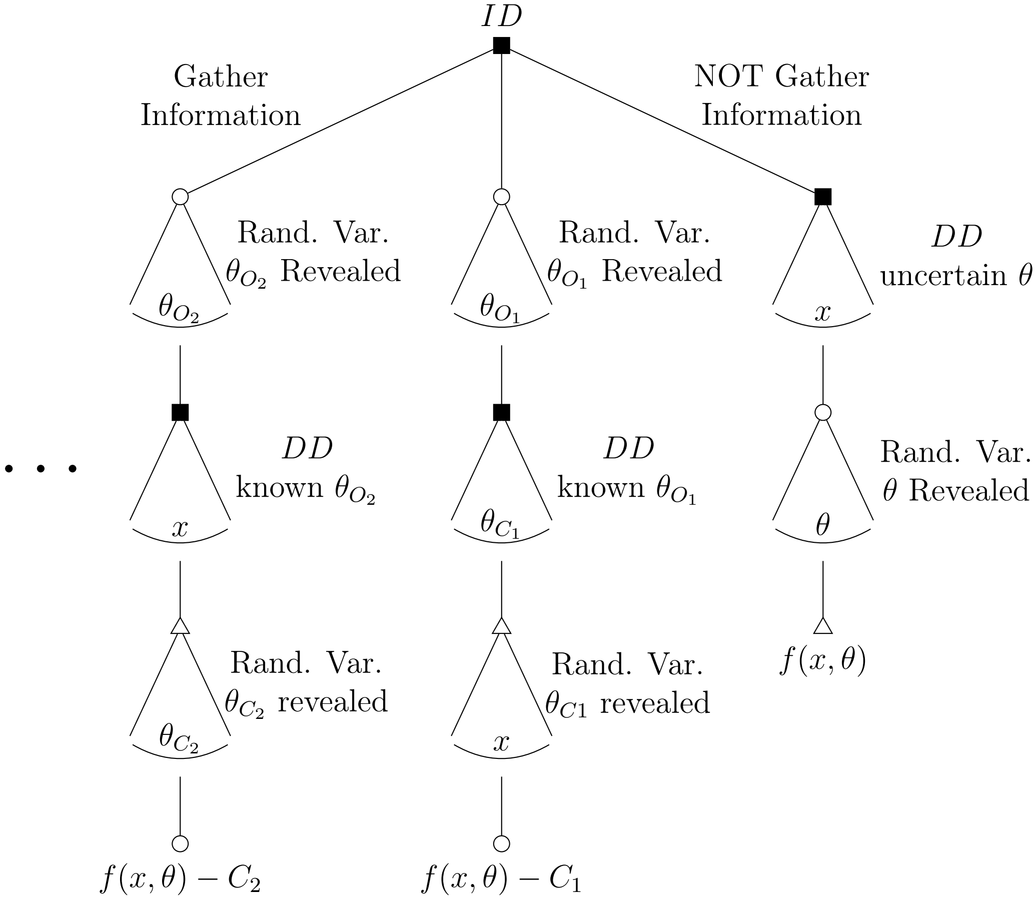

ID
Gather Information
NOT Gather Information
Rand. Var. θ_{O_2} Revealed
θ_{O_2}
Rand. Var. θ_{O_1} Revealed
θ_{O_1}
DD uncertain θ
x
DD known θ_{O_2}
x
DD known θ_{O_1}
θ_{C_1}
Rand. Var. θ Revealed
θ
Rand. Var. θ_{C_2} revealed
θ_{C_2}
Rand. Var. θ_{C1} revealed
x
$f(x, \theta)$
$f(x, \theta) - C_2$
$f(x, \theta) - C_1$

Figure 3.2: Decision Tree for information decision (ID) with partial perfect information. The arcs represent branches where a continuum of outcomes or actions are available. Decision nodes represent design decisions (DD) and are filled for emphasis.

where p and r are the number of inequality and equality constraints between the parameters and design variables, respectively. The MPO formulation can be thought of as an extension of the typical Parametric Optimization (PO) problem that allows for the consideration of multiple objectives.

To the best knowledge of the author, the MPO formulation is new to the literature and thus there exist no algorithmic approaches to solve this class of problem.

3.2.2 Engineering Relevance: Subsystem Capability Modeling

It is widely understood that a system composed of independently optimized subsystems is not itself likely to be optimal. Yet, many times it is important to build knowledge about the capabilities of a subsystem independent of system-level specifics. For example, this is true during initial research and development efforts or in sensitive military and competitive industrial design environments. In these scenarios compartmentalization of information is common and necessary.

Even within a specific systems development project, the need for concurrency often requires subsystem designers to make progress in the absence of full information about other interfacing subsystems. Multidisciplinary design optimization (MDO) methods are capable of coordinating disciplines and/or subsystem design efforts in a system-wide optimization scheme [71, 72, 73, 74, 75, 76, 77, 78]. However, these methods require all subsystems and their models to be defined. The MDO methods do not provide a ready pathway for characterizing individual subsystems independent of the system wide optimization process.

Finally, key requirements can change during a design project, potentially requiring redesign and repeated optimization of affected subsystems. These changes can be due to many factors, both external (e.g., customer change requests) and internal (e.g., one subsystem is unable to achieve targeted properties and other subsystems must adjust). A better understanding of the limits and capabilities of subsystems with respect to changes in requirements is needed.

One way to improve the communication of technical capabilities is to use formalized capability descriptions that are unambiguous and have well-defined semantics. In this research, we use the term *capability modeling* to describe the process of generating such a description.

The goal of capability modeling is to facilitate efficient communication between systems engineers and discipline engineers. Figure 3.3 is an illustration of the prototypical communication scenario between a systems engineer and discipline engineers during the systems definition and design phases of a project [79].

Systems engineers make decisions that require information about the capabilities of available technologies. As illustrated in Figure 3.3, discipline engineers should provide input about technical capabilities to systems engineers. In this work, we refer to the characteristics systems engineers need to know about to make decisions as the *attributes* of the subsystem. These can consist of, among other things, information about performance, cost, reliability, and relevant aspects of physical form (geometry, mass, etc.).

Of interest in this article is the case in which discipline engineers are able to use models to evaluate the attributes of a given technology. Figure 3.4 is a notional illustration of this scenario. Discipline engineers use a model to relate low-level design details to technology attributes, $\boldsymbol{f} : X \to Y$. In practice, $\boldsymbol{f}$ may be a composite of several individual models and simulation codes. Every feasible implementation of the technology in question is described by a vector of design variables, denoted $\boldsymbol{x}$, and the set of all physically realizable implementations is referred to as the feasible set, denoted X. The attributes of each design, $\boldsymbol{x} \in X$, are represented as a vector, denoted $\boldsymbol{y}$, and the set of attribute vectors corresponding to all feasible

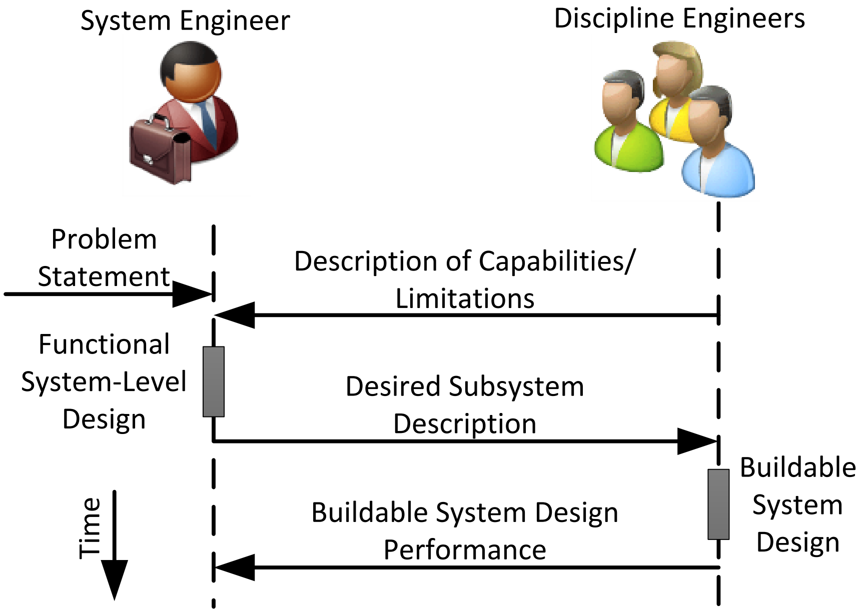

System Engineer
Discipline Engineers
Problem Statement
Description of Capabilities/ Limitations
Functional System-Level Design
Desired Subsystem Description
Buildable System Design
Time
Buildable System Design Performance

Figure 3.3: Idealized interaction between system-level and discipline-level engineers. [c]

implementations is referred to as the feasible attribute set, denoted A.

In this context, the goal of capability modeling is to generate a compact representation (not containing irrelevant information) of the feasible attribute set. Discipline engineers can identify the relevant subset of the feasible attribute set using dominance analysis.

Conceptually, the most straightforward representation of the a subsystem would be to describe the set of all feasible realizations. However, in principle, this set can be unbounded and, more practically, many members of the set may be demonstrably inferior to others. A more effective approach would be to focus only on the subset of realizations that are **potentially desirable** to the decision maker, i.e., the systems engineer. The concept of *parameterized Pareto dominance* (PPD) provides a means

[c] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

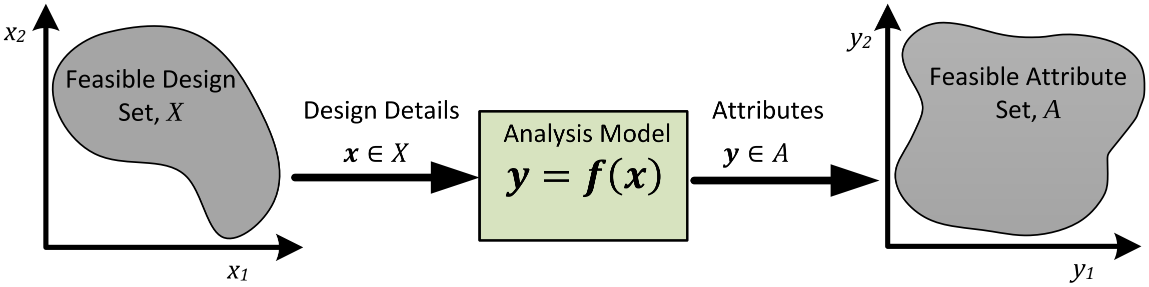


Figure 3.4: Illustrative example of the relationship between design details, $\boldsymbol{x}$, and their corresponding attribute values, $\boldsymbol{y}$. [d]

for identifying such potentially desirable members, collectively referred to as the *parameterized Pareto frontier* (PPF) [48] The criterion is a generalization of the classical Pareto dominance (PD) rule to handle situations in which some attributes have an unknown rank ordering. An unknown rank ordering means that one cannot determine which values of the attribute are most preferred based on the available information. Complete information (from system *and* subsystem levels) may reveal that the attribute value should be minimized, maximized or follows a non-monotonic ordering (e.g., the best value is somewhere in the middle of its range, with both lower and higher values being less desirable). Discipline engineers need to be able to perform dominance reasoning despite having attributes with unknown rank ordering if they are to generate practical capability models of their subsystem.

3.2.3 Engineering Application: Design of a Liquid Metal Magnetohydrodynamic Pump

In this section we motivate the use of parametric optimization for modeling the capabilities of a structural cooling subsystem. The design incorporates a liquid metal (LM) working fluid and driven by a *magnetohydrodynamic* (MHD) pump [80, 81, 82,

[d] Reprinted with permission from “P3GA: An Algorithm for Technology Characterization”, Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

83, 84]. In this scenario, the LM-MHD circuit is integrated into a larger structure and used to transport thermal energy from a heat source to a heat sink. The configuration of the cooling system is depicted in Figure 3.5.

A fluid circuit fully filled with a liquid metal is used to transport thermal energy from a hot reservoir, into which heat is transferred at some defined power P_{hot}, to a cold reservoir, which conducts heat out to a heat sink maintained at a temperature T_{cold}. The design objectives are to maximize the temperature of the hot reservoir, T_{hot}, while simultaneously minimizing channel mass of the circuit, denoted m_{chan}.

Ultimately, the best subsystem design will depend on the amount of heat being transferred to the hot reservoir, P_{hot}. In this scenario, the details of the larger system into which it will incorporated have not been revealed. Specifically, the discipline engineer does have any information about P_{hot}. the subsystem cannot be meaningfully optimized. Although a number of representative "reference systems" could be considered to enable local optimization, such an approach only provides a limited understanding regarding the performance capabilities and limitations of the MHD-driven LM subsystem.

In this scenario, P_{hot} is a parameter from the perspective of the cooling subsystem designer since it influences the optimal design yet the designer does not have direct control over it (it is to be determined by the systems engineer).

Let $\boldsymbol{x}$ be the design variables controlled by the cooling subsystem designer. The cooling subsystem designer knows that, all else being equal, a lower hot reservoir temperature, denoted $T_{hot}(\boldsymbol{x}, P_{hot})$, and cooling subsystem channel mass, denoted

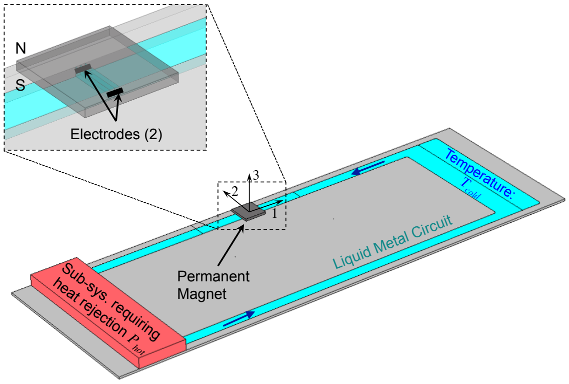


Figure 3.5: System schematic illustrating the details of the LM-MHD pump. [e]

$m_{chan}(\boldsymbol{x})$, are preferred. The parameterized optimization problem is formulated as

$$\begin{aligned} y^*(P_{hot}) = \min_{\boldsymbol{x}} \quad & (T_{hot}(\boldsymbol{x}, P_{hot}), m(\boldsymbol{x})) \\ \text{subject to} \quad & P_{hot}^L \leq P_{hot} \leq P_{hot}^U \\ & x \in X \end{aligned} \tag{3.12}$$

where the L and U superscripts denote the upper and lower bounds of the corresponding variable, respectively. The result of this optimization, $y^*(P_{hot})$, is a relation between the parameter variables and the best achievable values of the objectives, T_{hot} and m_{chan}. A systems designer can use this relation to explore how system-level decisions affect the cooling subsystem performance.

3.2.4 Engineering Application: Design of a Liquid Magnetohydrodynamic Pump Continued

In Eq. 3.12 the parameter space is defined simply by upper and lower bounds. In many engineering applications, it may be useful consider more complicated con-

[e] Reprinted with permission from “Parameterized Design Optimization of a Magnetohydrodynamic Liquid Metal Active Cooling Concept”, Hartl, D.J., Galvan, E., Malak, R.J., Baur, J.W., 2016 J. Mech. Des. 138(3) Copyright 2015 by ASME.

straints, such as those involving both design variables and parameters. Consider the scenario where the parameter is actually a *response* of the design variables $\boldsymbol{\theta}(\boldsymbol{x}) : X \to \Theta$. This formulation may be useful when two or more subsystems must agree on a response.

Consider, for example, a scenario where the LM-MHD is being used for thermal management rather than simply cooling. In this case, it is not preferred to minimize T_{hot}, but rather should be some yet unknown set point determined though system-level considerations. In this case, the problem would be stated as

$$\begin{aligned} y^*(T_{hot}, P_{hot}, l_{chan}) = \min_{\boldsymbol{x} \in X} \quad & m(\boldsymbol{x}, P_{hot}, l_{chan}) \\ \text{subject to} \quad & T_{hot} = T_{hot}(\boldsymbol{x}, P_{hot}, l_{can}) \\ & P_{hot}^{lb} \leq P_{hot} \leq P_{hot}^{ub} \\ & l_{chan}^{lb} \leq l_{chan} \leq l_{chan}^{ub} \end{aligned} \tag{3.13}$$

In this formulation, the parameter T_{hot} is a system response, and the solution y^* maps the parameters to the optimum achievable m.

3.3 Conclusions and Chapter Summary

In this dissertation, we define demonstrate how some common systems design challenges can be modeled as parametric optimization problems. Specifically, the parametric optimization problems will be motivated by (1) the problem of computing the expected value of information, and (2) capability modeling. The aim in modeling the problem this way is to allow for the development of tailored solution techniques that can exploit the parametric structure of the problem.

In this chapter, we demonstrated the use of PO to compute the Expected Value of Information (EVI). Under the EVI framework, one considers how much one is willing to pay to have access to some information. Specifically we considered the Expected

Value of Perfect Information (EVPI), where one is able to pay for "clairvoyance," that is, to know the uncertain parameters perfectly. We demonstrated how the computational expense of computing EVPI may exceed the cost of actually gathering information. In the extension to partial perfect information (EVPPI), the cost of gathering information grows exponentially with the amount of information that may be gathered. We demonstrated how, by recasting the EVI type problems, one can reduce the complexity of the calculation, in the case of EVPPI, the result is an exponential decrease.

Next, we extend the traditional definition of PO to the case with multiple objectives. We term this formulation multiobjective parametric optimization (MPO). To the best knowledge of the author, the MPO formulation is new to the literature and no algorithms exist for this class of problem.

The motivation for the MPO formulation is capability modeling. Engineering projects typically involve many discipline engineers, each with specialized knowledge about the system under development. The systems engineers must make decisions about system architecture definition, resource and requirement allocation, etc., despite not being an expert in each relevant discipline. Thus, the accurate communication of technical capabilities from discipline engineers to systems engineers can be an important factor in the success of a systems engineering project. We demonstrate how the solution to an MPO is the parametric Pareto frontier, which serves as a minimal representation of the capabilities of a subsystem. We provide an engineering example: the Liquid Metal Magnetohydrodynamic Pump (LM-MHD).

We demonstrated how the LM-MHD subsystem cannot be optimized independent of the subsystem into which it will be incorporated, since the optimal LM-MHD depends on the properties of the system (these properties are parameter attributes). The MPO formulation allows the designer to optimize the system, while remaining agnostic about the overall system. This can lead to a better understanding of the underlying physics in the model and a model of the capabilities of the LM-MHD.

4. PERFORMANCE ASSESSMENT OF APPROXIMATION ALGORITHMS FOR PARAMETRIC OPTIMIZATION

4.1 Considerations

It is important in any optimization field to be able to quantitative evaluate and compare different optimization algorithms. The performance assessment of algorithms allows us to simplify judgment between algorithms and helps us better understand the algorithm's behavior. The appropriate framework for comparison depends on the nature of the algorithms of interest and the problems which those algorithms aim to solve.

Broadly, there are two approaches for assessing algorithmic performance (ii) analytically, and (ii) empirically. In analytical approaches, it is common to estimate the performance in an asymptotic sense, in other words, how does the algorithm behave with arbitrarily large inputs? This is referred to as algorithm complexity. In optimization, analytical techniques can be used to develop solution quality and complexity guarantees. However, the results of analytical analysis are often limited to the "worst" case scenario. Sometimes, we are interested in the "average" case. Another limitation is that many algorithms, especially heuristic algorithms, are inaccessible to analytical analysis. Empirical analysis of algorithms, on the other hand, considers the "average" performance of the algorithm and can be used to obtain insights into any algorithm, including heuristic algorithms. Rather than using mathematical reasoning to assess performance, empirical analysis takes a scientific approach. Particularly, repeated experiments and statistical methods are used to establish confidence in the performance of an algorithm. Table 4.1 is a summary of the difference in the analytical and empirical approaches to performance assessment.

Because the algorithms presented in this dissertation (P3GA and p-NSGAII) are heuristic, we must rely on an empirical approach to performance assessment. However, the performance methods herein can be applied to any parametric optimization algorithm.

Table 4.1: Comparison between analytical and empirical approaches to performance assessment.

Analytical	Empirical
Mathematical Reasoning	Scientific Reasoning
Performance Guarantees	Detailed Analysis
Limited Application	Universal Application

The empirical analysis of algorithms requires a carefully chosen test set. Close attention must be paid to the size, difficulty, and structural properties, and relevance of the problem. Real wold problems are likely to be the most relevant; however, they tend to be expensive to evaluate (limiting the number of experiments that can be performed) and the true solution may not be known (limiting the assessment of solution quality). In this research, we develop artificial test problems that have similar characteristics (multi-modality, discontinuity, etc.) as many engineering problems. The artificial problems are cheap to evaluate, allowing us to conduct a detailed repeated analysis, and the true solution is known.

For optimization algorithms, the notion of performance of involves the quality of the solution found and the effort it takes to find it. Computational effort is typically measured as run-time (wallclock or CPU-time) or using the number of function evaluations (NFE). For single-objective optimization algorithms, comparing solution quality is straightforward: one only needs to consider the objective value, smaller or larger is better. This however is not the case for parametric optimization. Given two solutions to a PO problem S and S', how can the solutions be compared with respect to the objectives *and* parameters? One possibility is that is every point in S is parametrically dominated by some point in S'. If this is the case, we can say that S' dominates S. If this is not the case, however, the solutions are incomparable using parameterized dominance analysis.

Intuitively, one might suggest a measure of quality that captures how useful the approximation is to the designer. However, this indicator would necessarily be subjective and problem dependent since it must depend on the preferences of the designer and on how the data will be used. Thus, it is unclear what objective quality means with respect to parametric solutions.

This challenge is not unique to parametric optimization. It is discussed extensively in the multiobjective optimization literature [42, 41]. The approach taken in the MO literature is to focus on what statements can be made about the solution, without assuming any additional decision maker preferences. In this section, we leverage many of the results in the MO community to develop a useful quality measure for solutions to PO. Because, we are interested in the general PO problem, which may contain multiple objectives and multiple parameters, we will focus on the problem of assessing the quality of an approximation of the global parametric Pareto frontier (PPF).

4.1.1 Approximation Sets

In parametric optimization (single or multiobjective), a range of solutions is sought. From a decision perspective, each solution to the PO problem is mutually incomparable, meaning we cannot determine which solution is better without further preference information. The full set of solutions to the PO problem is the parameterized Pareto frontier (PPF). Herein, we will refer to approximations of the PPF as approximation sets.

It is important to note the distinction between term *approximation sets* as we have defined them here and *function approximation.* The problem of approximating function arises in many branches of math, computer science, engineering, etc. and is the process of determining a function that closely matches a target function (a relation between inputs and outputs). The approximation set is of the non-dominated designs rather than a relation. Thus, rather than *explicit* there is an *implicit* relationship between the attribute values. Consider again the notional example in Figure 3.4. One may find a function approximation of $f(x)$. However, we are interested in the approximation set of the non-dominated set of the feasible attribute set A.

Consider a system with a vector function $\boldsymbol{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that maps the feasible set of alternatives $X \subseteq \mathbb{R}^n$ to the attribute space $\mathbb{R}^m$. Let $\Theta \subseteq \mathbb{R}^p$ be the set of parameter vectors, denoted $\boldsymbol{\theta}$. The attribute space is the combined space of objective and parameter attributes $A = \{(\boldsymbol{y}, \boldsymbol{\theta}) : \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}), \boldsymbol{x} \in X, \boldsymbol{\theta} \in \Theta\}$. Suppose preference direction is known for each attribute y_i for $i = 1, \ldots, r$. Given two attribute vectors $\boldsymbol{a}', \boldsymbol{a}'' \in A$, we say $\boldsymbol{a}'$ Strictly dominates $\boldsymbol{a}' \succ\succ \boldsymbol{a}''$ If $\boldsymbol{a}'$ is better in all objective attributes and equal in all parameter attributes, weakly dominates $\boldsymbol{a}' \succ \boldsymbol{a}''$, if $\boldsymbol{a}'$ is not worse in all objectives, better in at least one, and equal in all parameter attributes, incomparable $\boldsymbol{a}' \nparallel \boldsymbol{a}''$ neither weakly dominates.

Definition 11. *(Approximation Set) Let $S \subset A$ be a set of attribute vectors. S is an approximation set if no solution in S weakly dominates another. Let Ω be the set of all approximation sets to a parametric optimization problem.*

Generally, we make no assumptions about the mathematical form of the approximation set; it may be a continuous or discrete representation. Given these definitions, the general results and theorems presented in [42] hold for the parametric case. The proofs can be carried out in the same way.

4.1.2 Unary Quality Indicators

The aim in this section is to investigate the use of unary quality indicators, denoted $I($, for PO approximation sets. A unary quality indicator is a mapping from the approximation set to the set of real numbers. Finding a satisfactory unary indicator would allow us to leverage tradition empirical performance evaluation techniques for the parametric case.

Given the definitions in , the general results regarding unary indicators developed in the MO literature also apply to the parametric case. Specifically, given two approximation sets S and S' in Ω

(a) There exists no unary quality measure that can indicate *whether* S is better than S'

(b) A unary quality measure may exist that can indicate *that* S is better than S'

See [42] for a rigorous definition of these statements and mathematical proof. In other words, if S is better than S', that is every member in S' is weakly dominated by some member in S, there may exists a unary indicator such that $I(S) \succ I(S')$. However, there cannot exist a unary indicator such that $I(S) \succ I(S')$ only if S is better than S'.

This limitation of unary indicators is significant but it gives us a useful metric for evaluating the usefulness different indicators. Our aim in selecting a unary indicator for PO solutions should be that it is able to detect that S is better than S'. There exist in the literature several unary quality indicators for Pareto frontier approximation sets such as generational distance and inverse generational distance, spread, distance from reference set, etc. [41, 62, 26, 85, 86, 87, 88, 89]. However, very few satisfy the condition that they be able to indicate *that* one solution is better than another. Specifically, only the hypervolume indicator [90] and mean Hausdorff distance [91] have been shown to have this property. We will discuss the applicability of these indicators to the problem at hand.

4.1.2.1 Hypervolume Indicator

The hypervolume indicator is widely used approach for measuring the quality of Pareto frontiers approximation sets. The hypervolume indicator is defined as follows

Definition 12. *(Hypervolume Indicator) The hypervolume indicator measures the volume of the dominated space of all solutions in the approximate space, truncated by a reference point.*

Several authors have used this metric to measure the quality of discrete Pareto frontier approximations [92, 93, 94]. This approach is well suited for comparing. If the true solution is known (as is the case for test problems in Section 4.3), quality can be measured relative to the hypervolume dominated by the true frontier. Figure 4.1 is an illustration of the hypervolume measure of two incomparable Pareto approximation sets S and S'.

Applying the hypervolume indicator to the case with parameters is problematic. Consider that the hypervolume indicator is truncated by a reference point. In the parametric case, it is unlikely that a single reference point can truncate an approximation set. Instead, the space dominated by the approximation set must be truncated by a $p-$dimensional surface, where p is the number of parameters. Figure 4.2 is an illustration of the hypervolume indicator in (a) the Pareto case, and (b) the parametric case. Notice that in the parametric case, it is necessary to bound the space dominated by the approximation set with a surface.

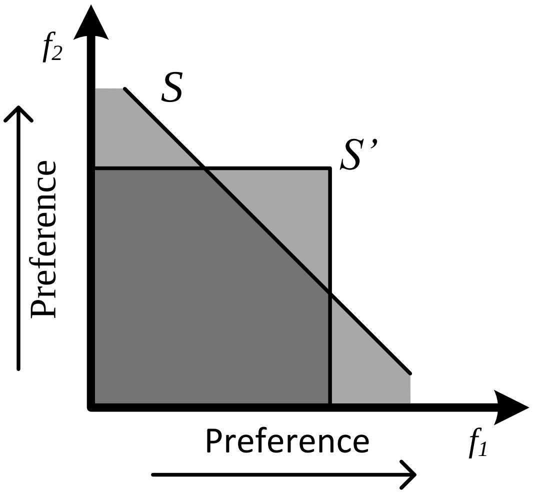
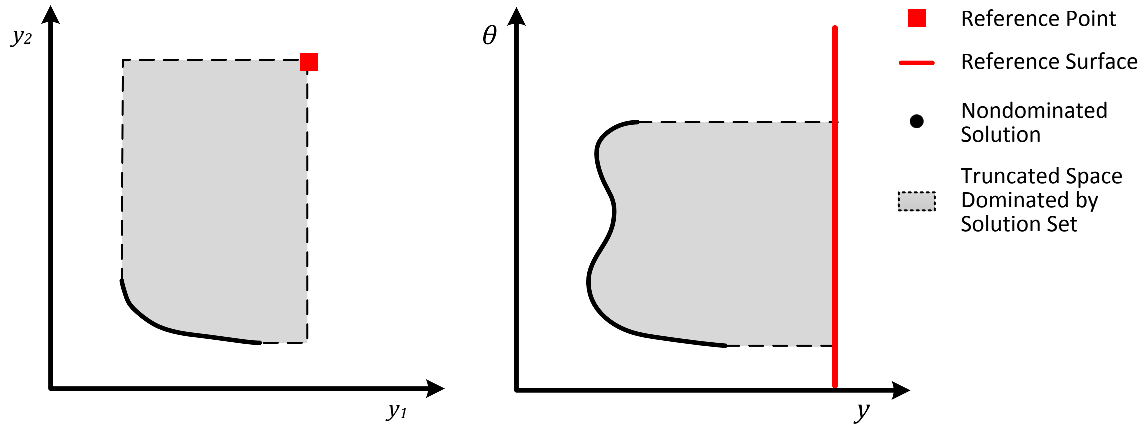


Figure 4.1: Illustration of hypervolume measure on two Pareto approximation sets S and S'. The approximation sets have equal hypervolume measures but are incomparable.

Another more considerable difficulty arises when the approximation set is not continuous. In Figure 4.3, we illustrate the hypervolume indicator, in (a) the Pareto case, and (b) the parametric case when the approximation sets are discrete. In case (a) the hypervolume indicator captures the space dominated by the approximation set. We can see that an additional member in the dominated space (shaded area) would not improve the hypervolume measure. On the other hand, adding a member in the non-dominated pace would. Furthermore, it has the desirable characteristic of increasing with spread of solutions and closeness to the true solution.

In case (b), one of the attributes is now a parameter. The space dominated by the approximation set is now 1-dimensional (a line). Again, an additional member in the dominated space (the dotted line) would not improve the hypervolume measure. However, the hypervolume measure is no longer a good indicator of spread among solutions. In Figure 4.3, the solutions could be moved arbitrarily closer (as long as



(a) Classic Pareto Dominance. Minimize y_1 and y_2

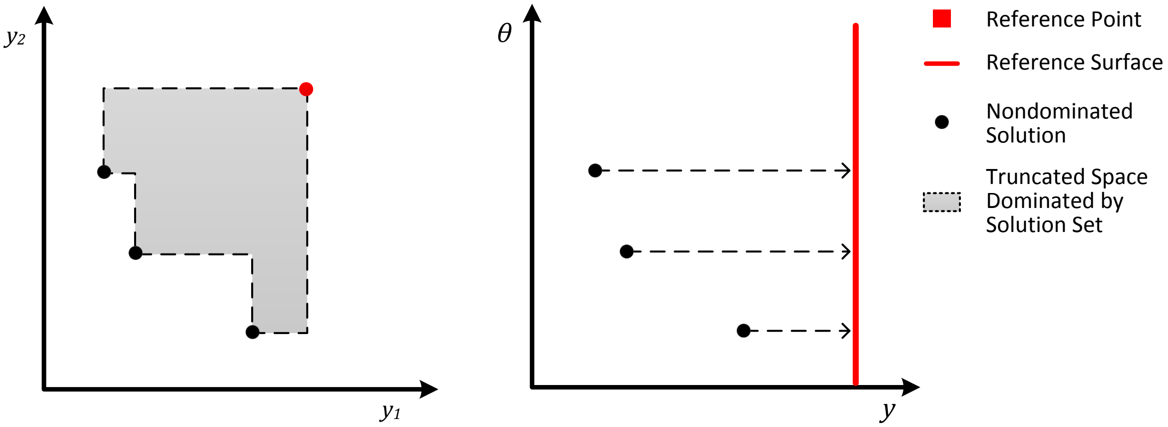
(b) Parametric Pareto Dominance. Minimize y and parameter θ

Figure 4.2: Application of the definition of hypervolume indicator for continuous sets in the case of (a) Pareto dominance and (b) parameterized Pareto dominance.

they do not overlap) or further away without changing the hypervolume measure.

4.1.2.2 Mean Hausdorff Distance Quality Indicator

The hypervolume indicator has the desirable property that one need not know the global solution to compare two approximation sets. However, in the case where the true solution is known, one way to measure the quality of an approximation set is by some measure of the distance to the true solution. Mathematically, this is task falls under *shape comparison*. Shape comparison is a common area of study in the computer graphics community, specifically in the task of shape comparison or shape matching [95, 96, 97]. There are several approaches for shape comparison in the literature focusing on different types of dissimilarity between the surfaces. In shape matching (classifying 3D objects), most comparison metrics consider *invariant* properties to characterize (compare) objects since operations such as translation and rotation do not fundamentally change the shape of an object [98, 99]. Many methods



(a) Classic Pareto Dominance. Minimize y_1 and y_2

(b) Parametric Pareto Dominance. Minimize y_1 and parameter y_2

Figure 4.3: Application of the definition of hypervolume indicator for discrete sets in the case of (a) Pareto dominance and (b) parameterized Pareto dominance

are additionally *scale invariant*, which enables a size independent classification of an object. Intuitively, invariant comparison methods are not well suited for comparing Pareto surfaces since we are typically concerned with differences in scaling, rotation, translation, etc.

A more appropriate shape comparison metric for our aim is the *geometrical distance* between the shapes (i.e., surfaces). To this end, we introduce two generic metrics commonly used to measure the distance between two surfaces: (1) the Hausdorff distance and (2) the mean Hausdorff distance. The Hausdorff distance is a measure of the distance between two subsets of a metric space [100]. Conceptually, the Hausdorff distance between two sets is the largest of all the distances between points on the sets where one of the points is the closest point to the other set. The mean Hausdorff distance is the integral of the Hausdorff distance of an entire surface, normalized by the hypervolume of the surface. Figure 4.4 is an illustration of the

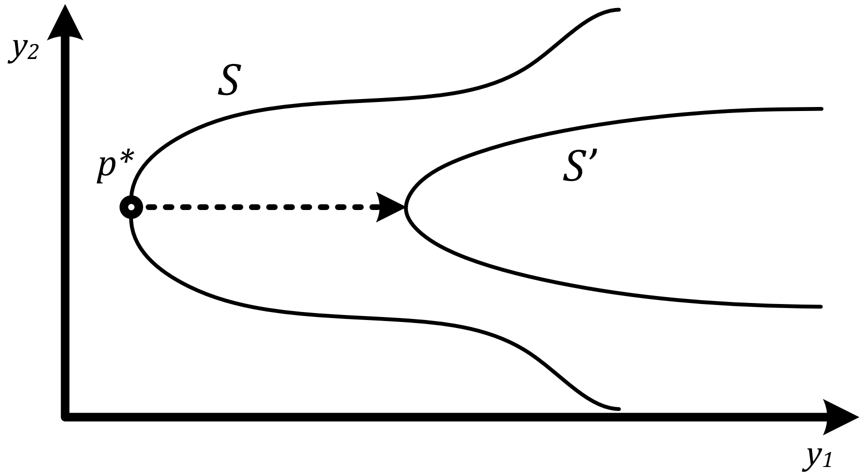


Figure 4.4: Illustration of the Hausdorff distances between the surfaces $\boldsymbol{S}$ and $\boldsymbol{S'}$ in $\mathbb{R}^{\mathbf{2}}$.

Hausdorff distance between two surfaces S and S' in $\mathbb{R}^2$.

The Hausdorff distance is the maximum expected error of the approximation set while the mean Hausdorff error gives the average error over the entire surface. The Hausdorff distance and mean Hausdorff distance have been used to measure the error associated with the simplification of 3-dimensional triangular meshes [101, 102].

Recently, the mean Hausdorff distance indicator Δ_p has been proposed as a robust quality indicator in the Pareto frontier approximation sets [91]. The Δ_p indicator satisfies the condition that if S is better than S' then $\Delta_p(S) \succ \Delta_p(S')$. Importantly, the Δ_p indicator is also compatible with the notion of parametric optimization. That is, it improves with solution spread. In this research, we propose the use of mean Hausdorff distance as a robust quality indicator for approximation sets for parametric optimization.

To define the two-sided mean Hausdorff distance, it is useful to first define the Hausdorff distance. Given a point p and a surface S, the distance $e(p, S)$ is defined as

$$e(p, S) = \min_{p' \in S} d(p, p') \tag{4.1}$$

where $d(\cdot)$ is the Euclidean distance between two points in E^n and n is the number of dimensions. The Hausdorff distance between surfaces S_1 and S_2 is then

$$E(S_1, S_2) = \max_{p \in S_1} e(p, S_2) \tag{4.2}$$

The mean Hausdorff distance is the surface integral of the Hausdorff distance divided by the area of S_1

$$E_m(S_1, S_2) = \frac{1}{|S_1|} \int_{S_1} e(p, S_2) ds \tag{4.3}$$

where $|S|$ denotes the total hypervolume of the surface S. The Hausdorff distance is not guaranteed to be symmetrical, i.e., $dS, S') \neq d(S', S)$. The *two-sided mean Hausdorff distance* is the maximum of $E_m(S, S')$ and $E_m(S', S)$. Note that this definition is applicable to continuous and/or discrete data sets (surfaces). The δ_p indicator is the two-sided mean Hausdorff distance between the true solution and the approximation.

4.2 Approximating Mean Hausdorff Distance

In this section we present an algorithm for measuring the geometric distance between two n-dimensional surfaces represented using simplices. This type of representation desirable since any generic n-dimensional surface can be represented using simplices. Therefore, the concepts described in this paper can be applied to other surface representations. For example, it is trivial to apply the techniques here in to surfaces that are continuous, discontinuous, discrete, or a mix. In the case of continuous frontiers, the simplex approach described herein can be used for numerical integration along the continuous function.

A surface (mesh) S will be represented by a point cloud P in $\mathbb{R}^n$ and by the set T of simplices describing how the vertices of P form the surface. Each simplex in surface S is of dimension $n-1$, therefore, a surface in $\mathbb{R}^3$ would be represented by triangles, in $\mathbb{R}^4$ we would use tetrahedrons, etc. We denote the two surfaces that we will be comparing S and S'.

The implementation of the algorithm relies on the definitions in the previous section. We will focus on the computation of the one sided mean Hausdorff distance, i.e., $d_m(S, S')$, since the two sided mean Hausdorff distance is simply the maximum of $d_m(S, S')$ and $d_m(S', S)$. The algorithm outlined in the following sections is an extension to the general n-dimensional case of the algorithms concepts presented in [101, 102].

4.2.1 Point to Surface Distance

The minimum distance between some point p belonging to surface S (note that p may not belong to the point cloud P) to and the surface S' is the minimum distance between p and all of the simplices $\mathcal{T} \in T'$. The minimum distance between p and some simplex is calculated using the approach presented in [103].

We first simplify the problem by shifting the point p and simplex $\mathcal{T}$ such that p is at the origin. The resulting $n-1$ simplex $\mathcal{T}$ in $\mathbb{R}^n$ is defined has vertices $V = (V_1, V_2 \dots, V_n)$. The smallest set containing V is called the *convex hull*

$$C(V) = \left\{ X : X = \sum_{i=1}^{n} V_i w_i, \sum_{i=1}^{n} w_i = 1, w \geq 0 \right\}. \tag{4.4}$$

In both cases the vector $w = (w_1, w_2, \dots, w_n)$ are the barycentric coordinates of the point X in V. The simplex $\mathcal{T}$ is the convex hull of its vertices, i.e., $C(V)$. The interior of $\mathcal{T}$ is the set of points whose weights in V are all positive. We can easily minimize

$|X|$, the Euclidean distance between the origin and the simplex $\mathcal{T}$, as follows

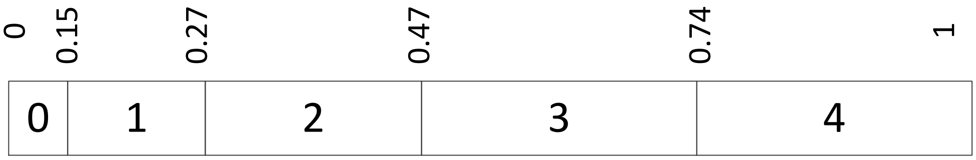
$$\begin{aligned} \min \quad & \|X\|^2 = w^T C^T V w \\ \text{subject to} \quad & \sum_{i=1}^{n} w_i = 1, \\ & w \geq 0. \end{aligned} \tag{4.5}$$

4.2.2 Hausdorff Distance Evaluation

Although $e(p, S')$ can be computed for any point p, it is necessary to resort to sampling to obtain the maximum for $p \in S$. Each simplex $\mathcal{T} \in T$ is sampled, and the minimum distance between each sample and the surface S' is computed. Surface sampling is achieved using the *abacus model* of a simplex which is used to subdivide an $n-$simplex into k^n $n-$simplices, all with the same volume and shape characteristics [104].

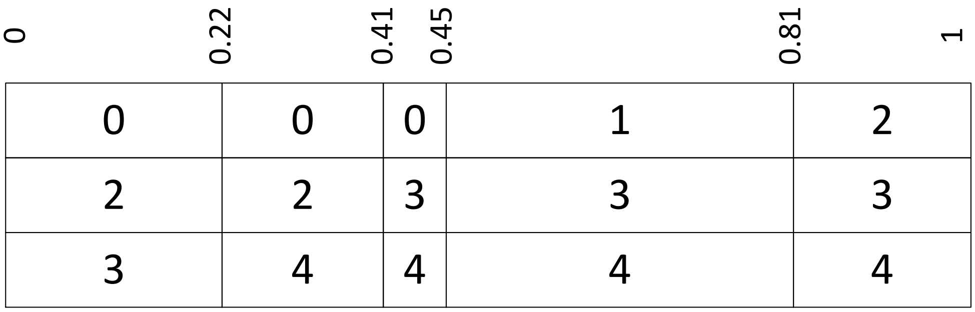
Using the notation from the previous section, a point X in a $n-1-$simplex in R^ncan be represented by a sequence of n numbers w_i in a unit interval, where w_i are the barycentric coordinates of X. The barycentric coordinates can be viewed graphically by drawing the interval as a rectangle with regions colored from 0 to $n-1$. Figure 4.5 illustrates this for a point in $\mathbb{R}^5$, $w = (0.15, 0.12, 0.2, 0.27, 0.27)$. The dividing lines are the cumulative sums of the barycentric coordinates, $(B_0 = 0, B_1, , B_n, B_{n+1} = 1)$.

Next we divide the rectangle in Figure 4.5 into k rectangles of equal size and stack them one on top of the other expanding each rectangle by a factor of k in the horizontal direction (see Figure 4.6. The coordinates of the dividing lines are found by multiplying the dividing lines from Figure 4.5 by k and discarding the integer portion. Then we extend the lines vertically and cut each rectangle into equally



0	1	2	3	4

Figure 4.5: Representation of the unit interval with colored sections. The coordinates of the dividing lines are displayed above the rectangle.



0	0	0	1	2
2	2	3	3	3
3	4	4	4	4

Figure 4.6: The rectangles are divided into equal regions. Each region keeps the color of the original divided lines multiplied by k.

many regions (see Figure 4.6).

If the number of regions in each row is $j+1$, the dividing lines are the numbers $(C_0 = 0, C_1, , C_j, C_{j+1} = 1)$. The widths of the resulting stacks are $c_i = C_{i+1} - C_i$, for $0 \leq i \leq j$. The resulting numbers form the following matrix called the color scheme

$$\chi = \begin{pmatrix} \chi_{1,0} & \cdots & \chi_{1,j} \\ \vdots & \ddots & \vdots \\ \chi_{1,0} & \cdots & \chi_{1,j} \end{pmatrix} \tag{4.6}$$

The numbers c_i can be used as barycentric coordinates for the point X in our original simplex. Define the notation

$$V_{\chi_1\chi_2\cdots\chi_k} \triangleq (V_{\chi_1} + V_{\chi_2} + \cdots + V_{\chi_k})/k \tag{4.7}$$

Defining points this way results in barycentric coordinates that are integer multiples of $1/k$. The i^{th} column in χ is the point $V_{\chi_1\chi_2\cdots\chi_k}$ of the new simplex. This approach takes a point X in $\mathcal{T}$ and produces a smaller simplex within it that contains the point. In order to obtain all of the $k^{(n-1)}$ color schemes corresponding to the desired subdivided simplices, the approach developed by Goncalves was used [105].

The desired sampling density for a simplex $\mathcal{T} \in T$ is determined by the user defined step size δ, which is a percentage of the diagonal bounding box of the surfaces S and S'. The number of samples that result from subdividing an $n-1$ simplex in $\mathbb{R}^n$ into k'^{n-1} smaller simplices is $(n+k'-1)!/(k'-1)!$. The desired sampling density (samples per hypervolume) is

$$\frac{1}{\delta^n} = \frac{(n+k'-1)!}{|\mathcal{T}|(k'-1)!n!} \tag{4.8}$$

Rewriting so that the k' terms are on the LHS

$$\prod_{i=0}^{n-1} k' + i = \frac{n!|\mathcal{T}|}{\delta^d} \tag{4.9}$$

Equation 4.9 can be solved using Newton's method. Since the sampling frequency k' found in Equation 4.9 is not typically an integer, we choose $k = \lfloor k' \rfloor$ or $k = \lfloor k' \rfloor + 1$ for each simplex, with probability ρ and $1-\rho$, respectively. The probabilities are chosen such that the expected value of the resulting sampling density

$$E\left(\frac{(n+k'-1)!}{|\mathcal{T}|(k'-1)!n!}\right) = \frac{1}{\delta^n} \tag{4.10}$$

Solving for ρ

$$\rho = \frac{\lfloor k' \rfloor}{n} + 1 - \frac{(n-1)!|\mathcal{T}|\lfloor k' \rfloor!}{\delta^n(\lfloor k' \rfloor + n - 1)!} \tag{4.11}$$

4.2.3 Mean Hausdorff Distance

The computation of the mean Hausdorff error is straightforward given that the distances (errors) at each sample have been computed. The integral of the error over the entire surface is computed by summing the contributions of all of the simplices formed by n samples in $\mathbb{R}^n$. Denote the simplex $t = (x_1, x_2, \ldots, x_n)$ where $x_1, x_2, \ldots, x_n$ are samples inside of a simplex $\mathcal{T} \in T$ and $(e_1, e_2, \ldots, e_n)$ are the errors (distances) associated with each sample. The error over T can be approximated by linearly interpolating the values between the errors. The value of the integral is $\frac{|t|}{n}\sum_{i=1}^{n} e_i$, where $|t|$ is the hypervolume of t. Figure 5 depicts the approximation in the 2-dimensional case. As a result of the sampling method described in the previous section, the samples form smaller simplices with the same hypervolume $|\mathcal{T}|/k^{n-1}$.

The hypervolume of $\mathcal{T}$ with vertices $V = (V_1, V_2 \ldots, V_n)$ is computed by first translating the simplex so that one of the vertices is at the origin, then finding the vector V_0 which is orthogonal to the remaining vectors $(V_1, V_2 \ldots, V_{n-1})$, and then normalize such that $|V_0| = 1$. The volume of V is then

$$|\mathcal{T}| = \frac{\det(V_1 - V_0, V_2 - V_0 \ldots, V_{n-1} - V_0)}{(n-1)!} \tag{4.12}$$

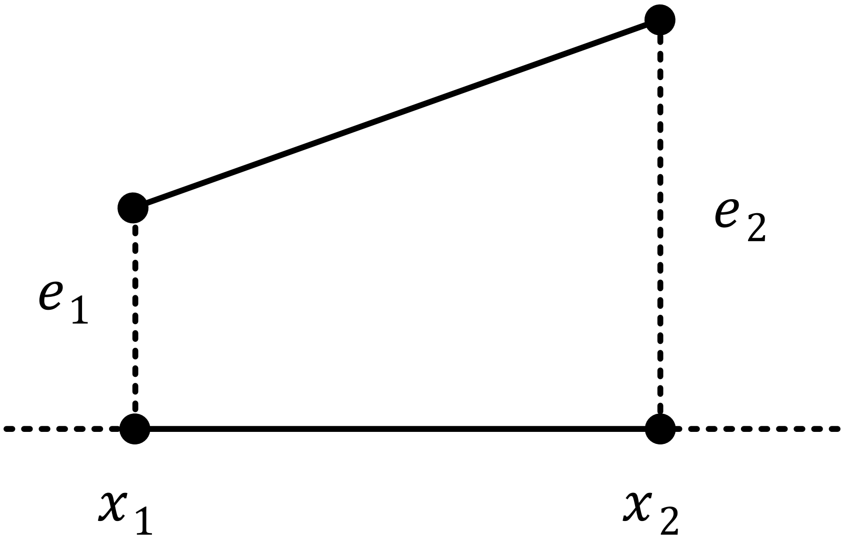


Figure 4.7: Approximation of the integral of the error in the 2 dimensional case.

4.2.4 Uniform Grid

The algorithm, if implemented naively, can become prohibitively computationally expensive even for small, low dimensional models. For each sample point p we would have to measure the point to simplex distance to every simplex in the surface S'. A sensible approach to reducing the computational burden is to attempt to reduce the number of required point to simplex computations. In the proposed algorithm, we extend the uniform grid approach used in [101, 102] to the n-dimensional case.

A uniform n-dimensional grid, D, is created around the joint bounding box of the surfaces S and S'. Each hyper-cube cell in the grid has side length Δ set by the user as a fraction of the side length of the average regular simplex $\bar{\mathcal{T}}'$. The simplex $\bar{\mathcal{T}}'$ is defined as the regular simplex having hypervolume equal to the average simplex $\mathcal{T}' \in T'$. Let $\bar{C}$ be the cell containing p and $D_l(\bar{C})$, $l \in \mathbb{N}$ be the set of cells that are a distance $l\Delta$ from p. See Figure 4.8 for an illustration for the 2-dimensional case where the cells $D_1(C)$ are shaded. First, the distance from p to all simplices that

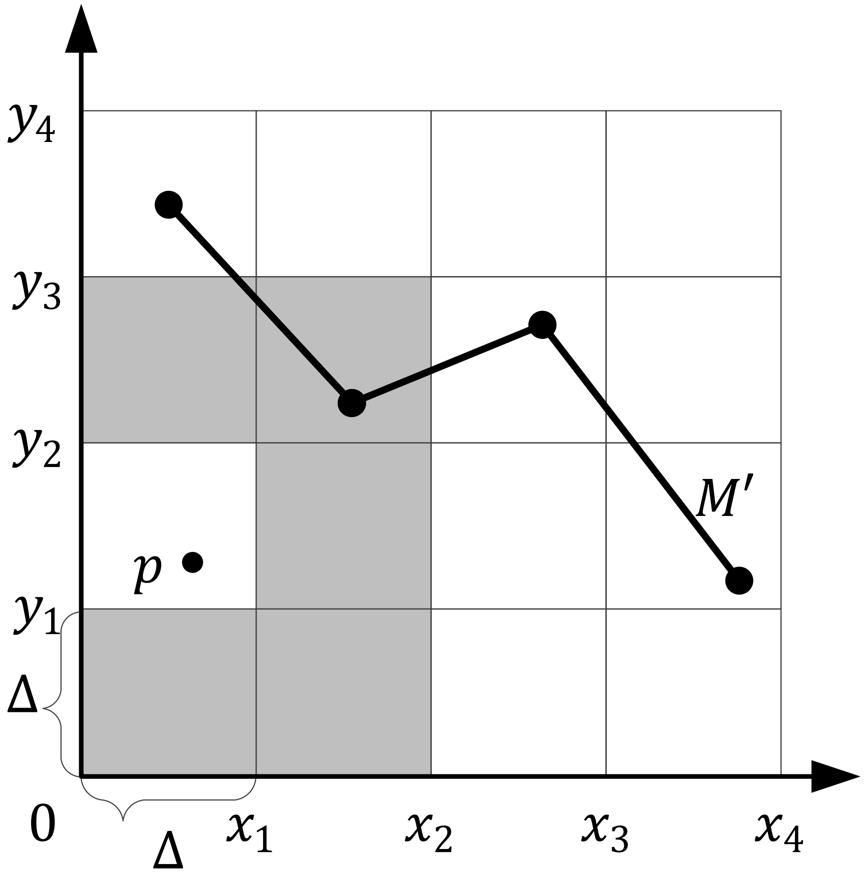


Figure 4.8: Unit grid in $\mathbb{R}^2$ with side length Δ. The cells $D_1(C)$ are shaded. Note that the surface M' in $\mathbb{R}^2$ is composed of 1-simplices (lines).

intersect the cells $D_l(C)$ for $l = 0$ is computed and the minimum, e_p, is retained. This process is repeated for increasing integer values of l (updating e_p at each iteration), until $l\Delta$ is larger than the current value of e_p since the remaining simplices in M' must be further away than e_p.

Consider again the simplex $\mathcal{T}$ with vertices $V = (V_1, V_2 \ldots, V_n)$. The constraint $lb \leq x \leq ub$ defining the cell $C \in D$ can be converted to barycentric coordinates as

$lb \leq xV \leq ub$. The intersection of the cell and the simplex is then the set

$$X = \left\{ x :, \ lb \leq xV \leq ub, \sum_{i=1}^{n} x_i = 1, x \geq 0 \right\} \tag{4.13}$$

The feasibility (whether the set is empty or not) of the set can be easily determined using linear programming techniques.

A straightforward approach leads to the complexity $O(N_c N'_{\mathcal{T}})$ where N_c and $N_{\mathcal{T}'}$ are the number of cells in grid D and the number of simplices in surface S', respectively. Since N_c grows exponentially with the number of dimensions, even this procedure can become very complex if implemented naively. To reduce the complexity of the approach we first reduce the memory requirement of storing the cell grid. Rather than storing $2^n N_c$ points (where 2^n is the number of vertices a hypercube in $\mathbb{R}^n$) in memory to define the grid, we only store the axis values of the grid, $A = (A_1, A_2, \ldots, A_n)$. The values along the i^{th} axis are $A_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,m})$, $\Delta = a_{i,j} - a_{i,j-1}$, for $1 \leq j \leq m$. Note that, in general, m is not the same for all dimensions $1 \leq i \leq n$. Now we can index the cells in D from 1 to m along each dimension. Figure 4.9 is an example in the 2-dimensional case.

To reduce the number of times we check for feasibility (i.e., the intersection between a cell and a simplex as in Equation 4.13) we first solve a much simpler feasibility problem by taking advantage of the grid-indexing scheme. The cell C can only intersect the simplex $\mathcal{T}$ if it also intersects the bounding box of $\mathcal{T}$. Let the bounding box of $\mathcal{T}$ be $B = (B_u, B_l)$, where B_u and B_l are the upper and lower bounds, respectively. Next we find the axis indices $J_i = \{j : \ a_{i,j} \leq B_{u,i}, a_{i,j-1} \geq B_{u,i}\}$, for $1 \leq i \leq n$. The indices of the set of the cells D that intersect B are then all of the combinations of the indices in $J = (J_1, J_2, \ldots, J_n)$, call this set R. The number of cells in R is typically very small compared to N_c. Finally, we determine whether the

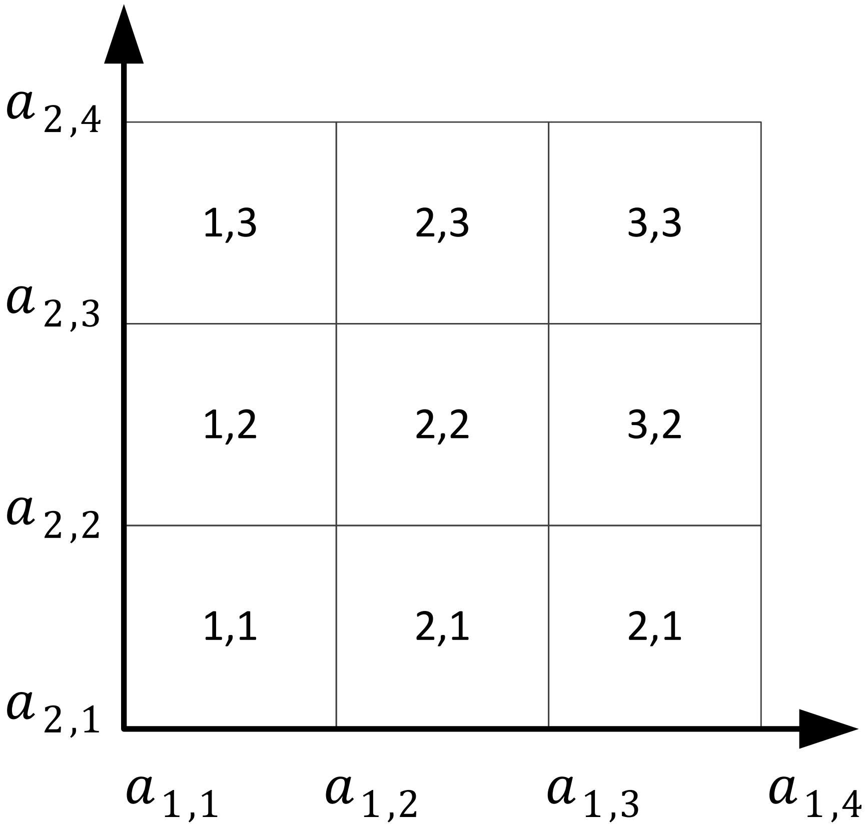


Figure 4.9: Grid indexing in the 2−dimensional case.

intersection between each cell in R and $\mathcal{T}$ is empty by solving Equation 4.13.

4.3 Parametric Test Problem Builder

In the case of approximation algorithms, it is particularly important to evaluate algorithmic performance relative to test problems. The test problems allow useful evaluation of characteristics such as convergence and precision. However, because parametric optimization is not a common problem in the literature, there does not exist a standard suite of test problems.

A critical part of this research is the development of problems that exhibit that are likely to cause difficulty to the optimizer in converging to the true solution, such as [106, 107]

- convexity or non-convexity in the non-dominated frontier,
- discontinuity in the non-dominated frontier,
- non-uniform distribution of solutions,
- multi-modality, and
- deception.

Test suites based on these factors, such as the DTLZ and ZDT problem suites, are widely used to evaluate the effectiveness of multiobjective algorithms [108, 90, 109, 110]. These factors are intended to cause difficulty in converging to the frontier and to reflect the difficulties encountered in real-world problems. In this research, parametric test problems are developed to exhibit a combination of the difficulty factors using the bottom-up approach described in [111]. Let $X \subseteq \mathbb{R}^n$ be the set of feasible decision vectors, denoted $\boldsymbol{x}$. Let $\boldsymbol{f} : X \to \mathbb{R}^m, \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$ be the vector valued objective function. Further, let the parameters be a response of the design variables $\boldsymbol{\theta} : \mathbb{R}^n \rightarrow \mathbb{R}^p, \boldsymbol{\theta}(\boldsymbol{x}) = (\theta_1(\boldsymbol{x}), \ldots, \theta_p(\boldsymbol{x}))$. The general mathematical form of the benchmark problems developed in this section is

$$\begin{aligned} y^*(\boldsymbol{\theta}) = \min_{x} \quad & \boldsymbol{f}(x) \\ \text{subject to} \quad & \boldsymbol{\theta} = \boldsymbol{\theta}(x) \\ & x_i^L \leq x_i \leq x_i^U \; \forall i = 1, \ldots, n \end{aligned} \tag{4.14}$$

For m objectives and p parameters, let

$$\boldsymbol{x} \equiv (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{m-1}, \boldsymbol{x}_m, \boldsymbol{x}_{m+1}, \ldots, \boldsymbol{x}_{m+p}) \tag{4.15}$$

be a complete decision variable vector partitioned into $m+p$ non-overlapping groups. That is $\boldsymbol{x}_i$ for $i = 1, \ldots, m+p$ are independent vectors.. The test problems are generated as

$$\begin{aligned} y^*(\theta) = \min_{\boldsymbol{x}} \quad & (f_1(\boldsymbol{x}_1), f_2(\boldsymbol{x}_2), \ldots, f_{m-1}(\boldsymbol{x}_{m-1}), f_m(\boldsymbol{x})) \\ \text{subject to} \quad & \boldsymbol{x}_i \in \mathbb{R}^{|\boldsymbol{x}_i|} \\ \text{where} \quad & f_m(\boldsymbol{x}) = g(\boldsymbol{x}_m) h(f_1(\boldsymbol{x}_1), f_2(\boldsymbol{x}_2), \ldots, f_{m-1}(\boldsymbol{x}_{m-1}), g(\boldsymbol{x}_m), \boldsymbol{\theta}) \\ & \boldsymbol{\theta} = (\theta_1(\boldsymbol{x}_{m+1}), \theta_2(\boldsymbol{x}_{m+2}), \ldots, \theta_p(\boldsymbol{x}_{m+p})) \end{aligned} \tag{4.16}$$

The parametric Pareto-optimal front is described by solutions which are global minimum of $g(\boldsymbol{x}_m)$, denoted g^*. The so called "difficulty function" $g(\boldsymbol{x}_m)$ involves features that make it difficult for a search algorithm to converge to the global minimum. The difficulty of the problem is associated with the difficulty factor $k = |\boldsymbol{x}_m|$. The parametric Pareto-optimal front is described as

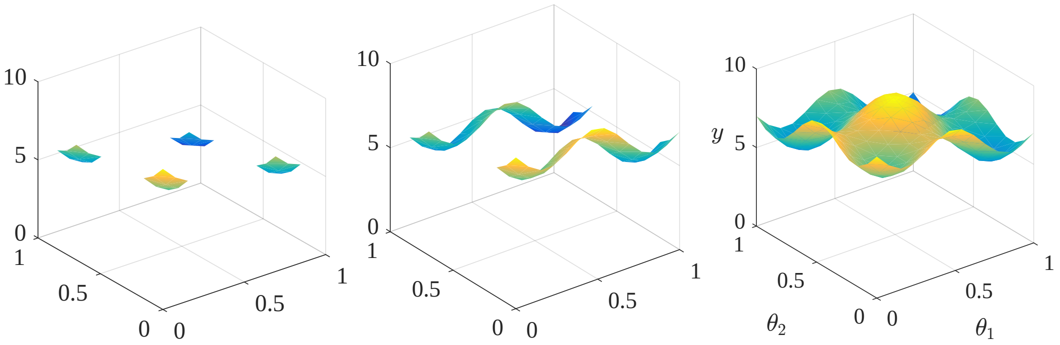
$$f_m = g^* h(f_1, f_2, \ldots, f_{m-1}, \boldsymbol{\theta}) \tag{4.17}$$

Since g^* is a constant number, the h function describes the parametric Pareto-optimal surface. Also, note that in the case without parameters, Eq. 4.16 simplifies to the test problem generator in [111].

4.3.1 Test Problem A

$$
\begin{aligned}
&f_i(\boldsymbol{x}_i) = \frac{1}{2}\sin(20x_i) \ \forall \ i = 1, \ldots, m-1 \\
&\theta_j(\boldsymbol{x}_{m+j}) = \frac{1}{2}\sin(20x_{m+j}) \ \forall \ j = 1, 2, \ldots, p \\
&f_m(\boldsymbol{x}) = (1 + g(\boldsymbol{x}_m))h(x_1, x_2, \ldots, x_{m-1}, \theta) \\
&g(\boldsymbol{x}_m) = \sum_{x_i \in \boldsymbol{x}_m} (x_i - 1/2)^2 \\
&h(f_1, f_2, \ldots, f_{m-1}, \theta) = 3(m+p) - \frac{\sum_{i=1}^{m-1} 2f_i + \sin(3\pi f_i) - \sum_{j=1}^{p} 2\theta_j + \sin(3\pi\theta_j)}{g(\boldsymbol{x}_m)} \\
&0 \leq x_i \leq \pi \ \forall \ i
\end{aligned}
\tag{4.18}
$$

Test problem A is developed to have variable density along $f_1, f_2, \ldots, f_{m-1}$. The parameter value is a non-linear response of the design variables. Difficulty function in test problem A is from DTLZ2 in [111]. This test problem will test the ability of the algorithm to converge to a non-convex frontier. The parameter is constrained only by upper and lower bounds. Figure 4.10 is an illustration of test problem A in 3 dimensions for 0-2 parameters. Notice that whether the solution is discontinuous depends on the number of parameters.



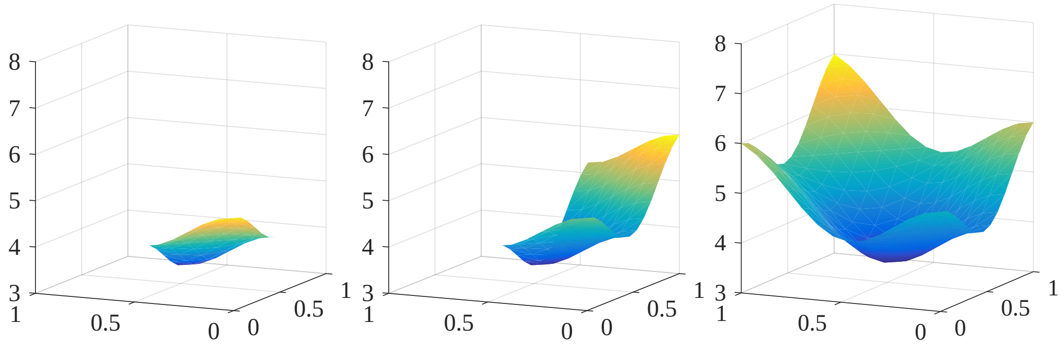
(a) 0 Parameters (b) 1 Parameter (c) 2 Parameters

Figure 4.10: Parametric Pareto frontier for test problem A with 3 dimensions

4.3.2 Test Problem B

$$
\begin{aligned}
&f_i(\boldsymbol{x}_i) = 1/2 + \sin(2\pi\cos(5\pi x_i)) \ \forall \ i = 1, \ldots, m-1 \\
&\theta_j(\boldsymbol{x}_{m+j}) = 1/2 + \sin(2\pi\cos(5\pi x_{m+j})) \ \forall \ j = 1, 2, \ldots, p \\
&f_m(\boldsymbol{x}) = (1 + g(\boldsymbol{x}_m))h(x_1, x_2, \ldots, x_{m-1}, \theta) \\
&g(\boldsymbol{x}_m) = 1 + m + p + \sum_{x_i \in \boldsymbol{x}_m} \cos(i\pi x_i) \\
&h(f_1, f_2, \ldots, f_{m-1}, \theta) = m + p + \sum_{i=1}^{m-1} f_i^2 + \cos\left(\pi(1 - f_i)\right)^2 + \sum_{j=1}^{p} \theta_j^2 + \cos\left(\pi(1 - \theta_j)\right)^2 \\
&0 \leq x_i \leq 1 \ \forall \ i
\end{aligned}
\tag{4.19}
$$

Test problem B is developed to have more challenging characteristics than test problem A. Test problem B features variable density along $f_1, f_2, \ldots, f_{m-1}$. The parameter value is a non-linear response of the design variables. The difficulty and h functions are novel. The difficulty function is multimodal resulting in many local solutions to this problem. Figure 4.11 is an illustration of test problem A in 3 dimensions for 0-2 parameters. The difficulty function g also features local Pareto attractors.



(a) 0 Parameters (b) 1 Parameter (c) 2 Parameters

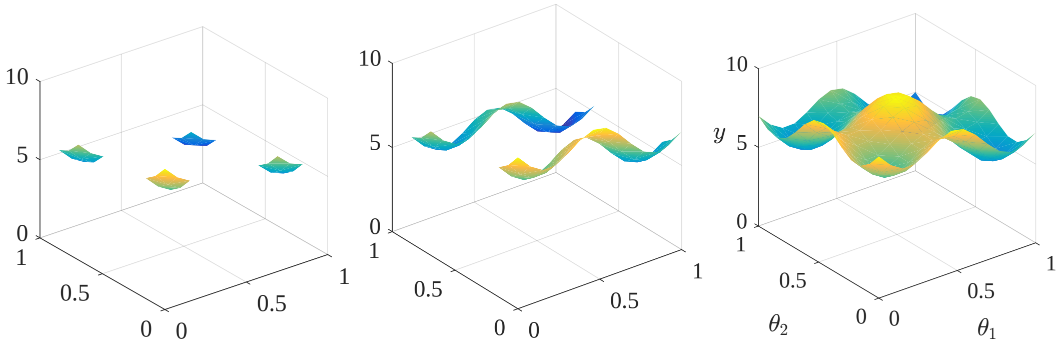
Figure 4.11: Parametric Pareto frontier for test problem B with 3 dimensions

4.3.3 Test Problem C

$$
\begin{aligned}
&f_i(\boldsymbol{x}_i) = \frac{1}{2}\sin(20x_i)\ \forall\ i = 1, \ldots, m-1 \\
&\theta_j(\boldsymbol{x}_{m+j}) = \frac{1}{2}\sin(20x_{m+j})\ \forall\ j = 1, 2, \ldots, p \\
&f_m(\boldsymbol{x}) = (1 + g(\boldsymbol{x}_m))h(x_1, x_2, \ldots, x_{m-1}, \theta) \\
&g(\boldsymbol{x}_m) = 1 + \frac{9}{k}\sum_{x_i \in \boldsymbol{x}_m} x_i \\
&h(f_1, f_2, \ldots, f_{m-1}, \theta) = 3(m+p) - \sum_{i=1}^{m-1} 2f_i + \sin(3\pi f_i) - \sum_{j=1}^{p} 2\theta_j + \sin(3\pi\theta_j) \\
&0 \leq x_i \leq 1\ \forall\ i
\end{aligned}
\tag{4.20}
$$

Test problem C is a variation of test problem A. The difference is the difficulty function g and the h function has been simplified. The difficulty function is from

DTLZ7 in [111]. Figure 4.12 is an illustration of test problem B in 3 dimensions for 0-2 parameters.



(a) 0 Parameters (b) 1 Parameter (c) 2 Parameters

Figure 4.12: Parametric Pareto frontier for test problem C with 3 dimensions

4.3.4 Test Problem D

$$\begin{aligned}
&f_i(\boldsymbol{x}_i) = \frac{1}{2}\sin(20x_i)\ \forall\ i = 1, \ldots, m-1 \\
&\theta_j(\boldsymbol{x}_{m+j}) = \frac{1}{2}\sin(20x_{m+j})\ \forall\ j = 1, 2, \ldots, p \\
&f_m(\boldsymbol{x}) = (1 + g(\boldsymbol{x}_m))h(x_1, x_2, \ldots, x_{m-1}, \theta) \\
&g(\boldsymbol{x}_m) = \sum_{x_i \in \boldsymbol{x}_m} x_i^{1/4} \\
&h(f_1, f_2, \ldots, f_{m-1}, \theta) = 10\sum_{i=2}^{m-1} f_i \\
&0 \leq x_i \leq 1\ \forall\ i
\end{aligned} \tag{4.21}$$

Test problem D is developed to test the ability of the algorithm to generate evenly spaced parametric Pareto optimal solutions. Test problem D features variable density along $f_1, f_2, \ldots, f_{m-1}$. The difficulty function is the same as for test problem

C. Figure 4.13 is an illustration of test problem B in 3 dimensions for 0-2 parameters.

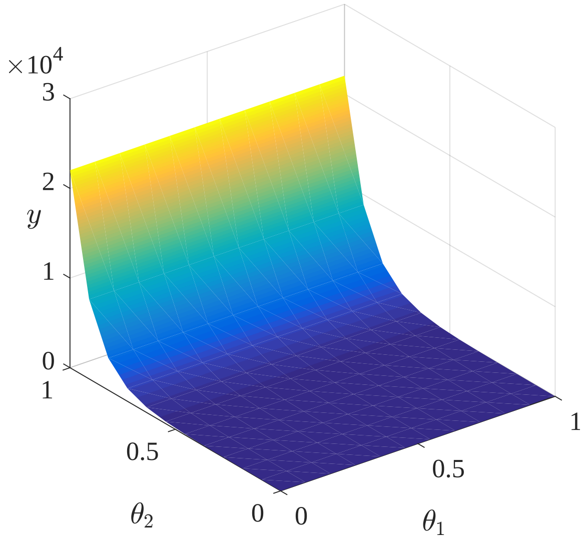


Figure 4.13: Parametric Pareto frontier for test problem D with 3 dimensions: 1 objective and 2 parameters.

4.4 Conclusions and Chapter Summary

In this chapter, we have presented an approach for quantitative evaluation and comparison of algorithms for parametric optimization. Because the algorithms to be presented in this dissertation are heuristic, we focused on an empirical approach to performance evaluation. An empirical approach uses scientific reasoning and detailed analysis to evaluate and compare algorithms. Key challenges are the selection of suitable test problems and performance metrics.

We developed test problems for which we can arbitrarily scale the number of objectives and parameters and even the difficulty of the problems. The solutions to the test problems are known with certainty, which simplifies performance assessment. The test problems allow useful evaluation of characteristics such as convergence and precision. The problems were carefully developed to exhibit features that are likely to cause difficulty to the optimizer and are common to many engineering problems, e.g., non-convexity, discontinuity, multi-modality, etc.

The selection of a suitable performance metric is also a critical step in algorithm evaluation. From an analysis perspective, a unary indicator of quality is desirable since it allows for the determination of out-performance: determining with statistical significance whether one technique outperforms the other. We showed how some general results on unary indicators hold for the parametric case. Finally, we considered two unary indicators: hypervolume indicator and the mean Hausdorff distance. We showed, that in the case of discrete approximation sets involving parameters, the hypervolume metric is not able to reflect the solution “spread.” We suggest the use of mean Hausdorff distance for performance assessment of approximation sets involving parameters. The mean Hausdorff distance has desirable properties in that it can indicate *that* one approximation is better than another and it can reflect solution “spread,” even in the case with parameters.

5. ALGORITHMS FOR PARAMETRIC OPTIMIZATION

5.1 Heuristic Algorithms for Engineering Design

In optimization, is not generally possible to develop algorithms that (1) find truly optimal solutions (2) quickly (3) for any instance. For engineering problems in particularly, we typically must sacrifice one of these three. Possibly the most common approach is to relax condition (1), and instead find solutions that are "good enough." This relaxation is the justification for many types of heuristics and meta-heuristics: genetic algorithms, simulated annealing, etc.

At the cost of solution accuracy, the benefit of a heuristic approach is (1) they are easy to implement since they require little knowledge about the mathematical structure of the problem (2) and can be applied to a wide range of problems. This makes them attractive as a first step in introducing parametric optimization to systems design.

5.1.1 Adapting Multiobjective Techniques for Parametric Optimization

The examples used in this section involve one objective and one parameter. However, the results herein also apply to the multi-parametric multiobjective case.

5.1.1.1 Method of Alternating Preference Directions

Perhaps an intuitive approach to solving parametric optimization is to assume a preference direction for the parameter attributes, and solve as usual using multiobjective techniques. Then, repeat this procedure with alternating preference directions. However, the result of this approach would not in general capture the full PPF. It should be clear from Figure 2.7 (where we compare PF and PPF) that PPF is not a union of PFs with alternating preference directions. However, in the interest of completeness and clarity, we address the issue directly as follows. Consider a nonempty set of alternatives A where $\boldsymbol{a} = (a_1, a_2, \ldots, a_M)$ denotes a vector of M attributes. Let D denote the nonempty set of indices for the objective attributes and P denote the set of indices corresponding to the parameter attributes such that $D \cup P = \{1, 2, \ldots, m\}$ and $D \cap P = \emptyset$. We assume minimization in the objective attributes without loss of generality.

An alternative $\boldsymbol{a}^* \in A$ lies on the PPF if and only if there does not exist another alternative $\boldsymbol{a'} \in A$ such that $a'_i \leq a^*_i \; \forall i \in D$ and $a'_i < a^*_i \; \exists i \in D$ and $a'_i = a^*_i \; \forall i \in P$. Consider three alternatives on the PPF $\boldsymbol{a}^*$, $\boldsymbol{a}^{**}$, and $\boldsymbol{a}^{***}$ such that $a^*_i = a^{**}_i = a^{***}_i$ $\forall i \in D$, and $a^*_k < a^{**}_k < a^{***}_k$ $k \in P$ and $a^*_i = a^{**}_i = a^{***}_i$ $\forall i \in \{P \backslash k\}$.

From the definition of PD, an alternative $\boldsymbol{a}^* \in A$ lies on the PF if and only if there does not exist a vector $\boldsymbol{a}^{'} \in A$ such that $a^{'}_i \leq a^*_i \; \forall i \in D \cup P$ and $a^{'}_i < a^*_i$ $\exists i \in D \cup P$.

If we apply PD to the alternatives, then $\boldsymbol{a}^{**}$ and $\boldsymbol{a}^{***}$ would be dominated since $a^*_i \leq a^{**}_i, a^{***}_i \; \forall i \in \{D \cup P\}$ and $a^*_k < a^{**}_k, a^{***}_k$. If we reverse the preference direction in the j^{th} objective, alternatives $\boldsymbol{a}^*$ and $\boldsymbol{a}^{**}$ would be dominated. Reversing the preference direction in any other objective would have no effect on dominance since $a^*_i = a^{**}_i = a^{***}_i \; \forall i \in \{D \cup P \backslash k\}$.

Thus, *any* combination of preference direction for PD will always dominate alternative $\boldsymbol{a}^{**}$. If the *preference order* in the k^{th} objective is unknown, $\boldsymbol{a}^{**}$ may ultimately be the most preferred alternative As a result, straightforward multiobjective optimization techniques such as such as the weighted sum method, ε-constrained method [112], or MOGAs such as NSGAII cannot in general solve a parametric problem by simply alternating the preference directions.

5.1.1.2 Method of Iterating over Constrained Parameters

Another intuitive approach is to solve the parametric optimization problem by constraining the parameter to a specific value, then solving the resulting search problem. One could iterate over many different parameter (or combinations of parameter) values to approximate the parametric Pareto frontier. This would be an iterative approach to parametric optimization. However, many questions naturally arise: How do we select the parameter combinations? Once we have constrained the parameter value, what algorithm should be used to solve the resulting search problem? Would the resulting solutions be parametric Pareto optimal? Can we leverage the information gained during each iteration?

The most straightforward iterative approach is to select equidistant values. This approach is not guaranteed to result in a well-distributed spread of solutions. For example, consider the simple parametric optimization problem

$$\begin{aligned} y^*(\theta) = \min_x \quad & x \\ \text{subject to} \quad & \left(\frac{\theta - 20}{20}\right)^8 + \left(\frac{x-1}{1}\right)^8 \leq 1 \\ & 0 \leq \theta \leq 20 \end{aligned} \tag{5.1}$$

Sampling θ equidistantly and solve using single-objective optimization techniques produces the results illustrated in Figure 5.1.

The problem of evenly distributing points along the solution frontier is also common in multiobjective optimization. Note the similarity between Figures and , which is an illustration of the ε-constraint method for MO optimization. Methods such as the Normal Boundary Intersect (NBI) method by Das and Dennis [24] or the normal constraint method [113] have been created to address the problem. Unfortunately,

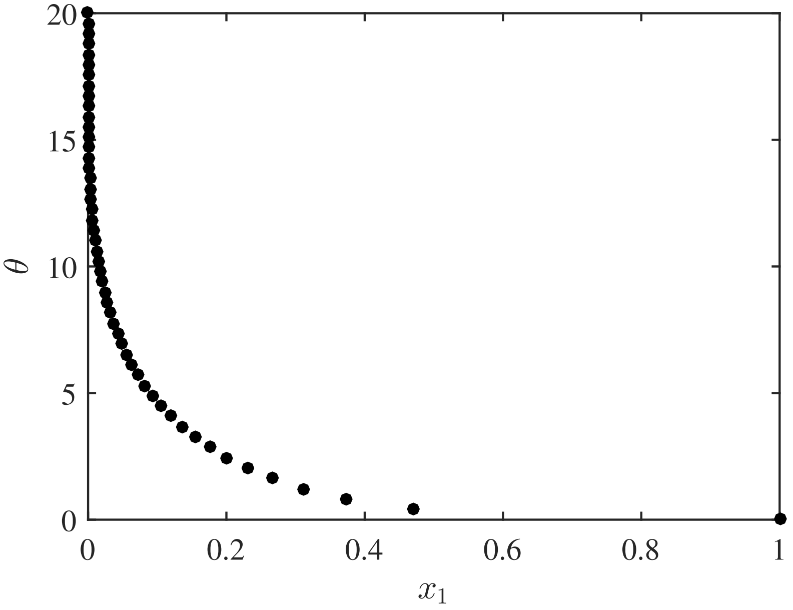


Figure 5.1: Solution to Equation 5.1 by sampling θ equidistantly.

applying such methods to the case with parameters is problematic.

Consider the illustrative example in Figure 5.2, which has one objective attribute y and one parameter θ. The true non-dominated frontier (parametric Pareto frontier) is depicted as the bold line. To apply NBI to the parametric case, we create further problems where the preference direction of the parameter value attributes. These subproblems are denoted $\text{NBI}_{\min\ \theta}$ and $\text{NBI}_{\max\ \theta}$ in the figure. Recall from Section 2.4.2 that the NBI method places points on the boundary of the search space rather than the Pareto frontier. As a result, an NBI approach to parametric problems may contain irrelevant solutions (not truly parametric Pareto optimal) if the search space is non-convex, see Figure 5.2.

In the Pareto case, the non-optimal solutions are typically filtered using the Pareto dominance rule. This is problematic in the case with parameters since it is unlikely that any of the solutions will have equal parameter values, a necessary

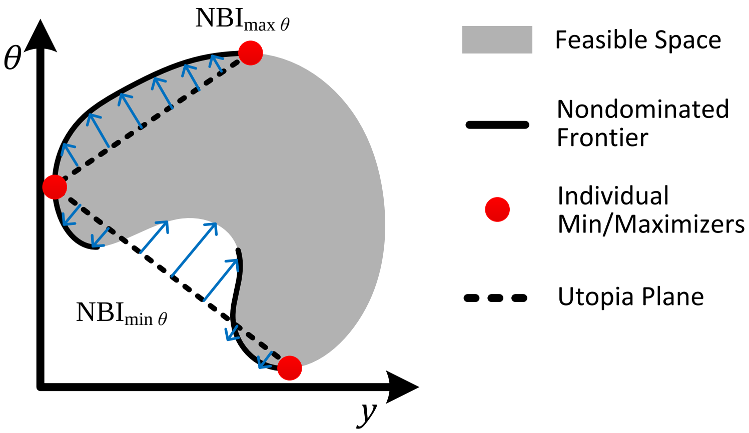


Figure 5.2: Application of NBI to a parametric optimization problem. Two NBI subproblems are created NBI$_{\min\ \theta}$ and NBI$_{\max\ \theta}$ where the parameter value is minimized and maximized, respectively. The solid line represents the true parametric Pareto frontier for the example problem. For this example problem, the NBI method would find solutions not on the optimal frontier.

condition for parametric dominance. A NBI approach to parametric optimization would resolve the issue of unequally spaced solutions but would create a new issue of finding non-optimal solutions.

5.2 Parametric NSGAII

In the previous section, we saw that there is no obvious way to extend traditional techniques for multiobjective optimization to the case with parameters. As a first step, we developed an extension to Nondominated Sorting Genetic Algorithm II (NSGAII) for the parametric case, which we call parametric NSGAII (p-NSGAII). In p-NSGAII, the parametric problem is solved by constraining the parameter value, then solving the resulting multiobjective problem with NSGAII. We elected to use NSGAII to solve the multiobjective subproblems because of its performance, flexibility, and because it is widely used as a standard for comparison in the multiobjective community.

A practical consideration is that genetic algorithms are known to converge considerably slowly in the when such constraints are present [114]. Thus, if we include a nonlinear equality constraint at each application of NSGAII, we should expect a considerable loss in performance. We instead constrain each application of NSGAII to ranges of parameter values instead. The p-NSGAII subproblem is

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{s.t.} \quad & \varepsilon_i^L \leq \theta_i \leq \varepsilon_i^U \ \forall \ i = 1, \dots, p \\ & g_i(\boldsymbol{x}, \boldsymbol{\theta}) \leq 0 \ \forall \ i = 1, \dots, q \\ & h_i(\boldsymbol{x}, \boldsymbol{\theta}) = 0 \ \forall \ i = 1, \dots, r \\ & \boldsymbol{x} \in X \subseteq \mathbb{R}^n \end{aligned} \tag{5.2}$$

where $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}) = (f_1(\boldsymbol{x}, \boldsymbol{\theta}), f_2(\boldsymbol{x}, \boldsymbol{\theta}), \dots, f_m(\boldsymbol{x}, \boldsymbol{\theta}))$. Another benefit of this approach is that we are able to find solutions at arbitrary parameter values, improving spread in the approximation set. The subproblem in Equation 5.2 is a multiobjective optimization problem, which can be solved with NSGAII.

Genetic algorithms such as NSGAII can be *seeded* (initialized with a "good" population) to improve performance. In p-NSGAII, we seed each subproblem with the results from a previous search. It was found anecdotally that the p-NSGAII algorithm performance greatly improved when seeded with the final generation of neighboring subproblem. The details of NSGAII are provided in the following section.

5.2.1 Non-dominated Sorting Genetic Algorithm II

The multiobjective subproblem in p-NSGAII is solved using Non-dominated Sorting Genetic Algorithm II (NSGAII) [26]. NSGAII is a general-purpose multiobjective genetic algorithm for locating solutions on the Pareto frontier. A high-level flow-chart of the NSGAII algorithm is presented in Figure 5.3.

In order to assign fitness to the population members, NSGAII incorporates the Nondominated Sorting method. In this method, before the selection operator is performed, the population is ranked on the basis of a member's "non-domination". Let $NonDominated(\cdot)$ be a function that finds the non-dominated members of a set. The pseudo code for the non-dominated sorting procedure is presented in Algorithm 1.

Algorithm 1 Nondominated Sorting

```
procedure (P)
    $j \leftarrow 1$
    $P_j \leftarrow \emptyset$
    while $P \neq \emptyset$ do
        $P' \leftarrow \boldsymbol{NonDominated}(P)$
        $P_j \leftarrow P'$
        $P \leftarrow P \backslash P'$
        $j \leftarrow j + 1$
```

The non-dominated sorting technique is widely used and has been shown to promote population diversity and improve convergence [26, 86, 115]. In addition to being converged closely to the non-dominated frontier, solutions must also be sparsely spaced along the non-dominated frontier. The diversity preservation operator assigns higher fitness to members that lie in less "crowded" regions of the space. Several crowding distance metrics have been proposed in the literature. The crowding-distance technique used in NSGA-II sorts the set of solutions according to each objective function. The solutions are assigned distance values equal to the absolute difference of the objective values of the two adjacent solutions; an example is illustrated in Figure 5.4.

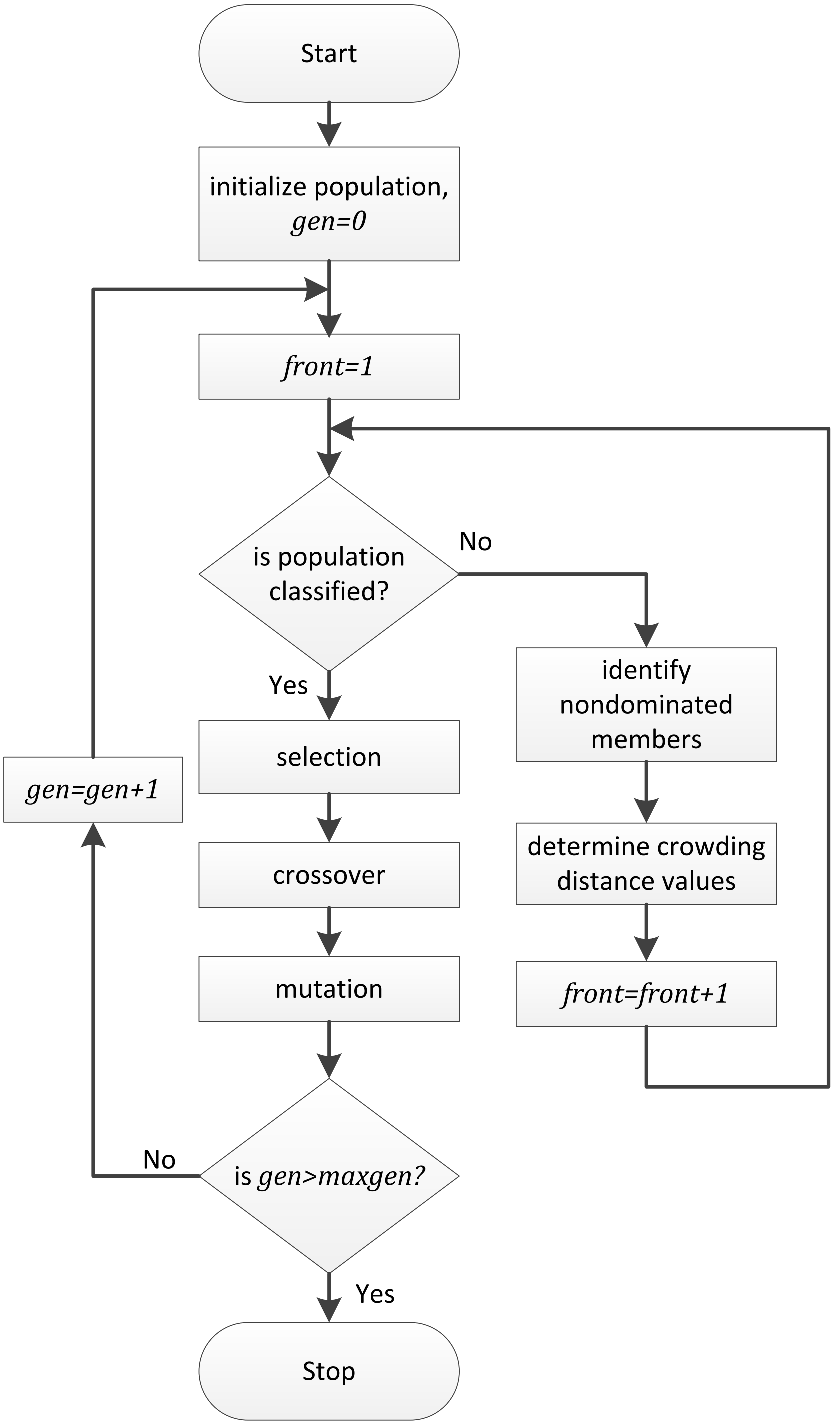

Start
initialize population, *gen=0*
front=1
is population classified?
No
Yes
identify nondominated members
determine crowding distance values
front=front+1
selection
crossover
mutation
gen=gen+1
is *gen>maxgen?*
No
Yes
Stop

Figure 5.3: Flowchart of the NSGA-II algorithm

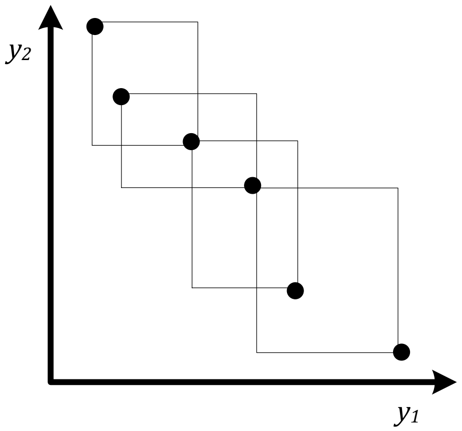


Figure 5.4: An example of NSGA-II crowding distance metric. Objectives are to minimize y_1 and y_2 [38].

Once the population has been sorted into non-dominated ranks, each rank is arranged in descending order of magnitude of the crowding distance values. Then the Crowded Tournament Selection operator proposed by Deb is used to determine the parents for the offspring [62]. The Crowded Tournament Selection operator is defined as follows. A solution i wins a tournament with another solution j if any of the following conditions are true:

- If solution i has higher rank than j.
- If solution i and j have equal rank but solution i has better crowding distance than j.

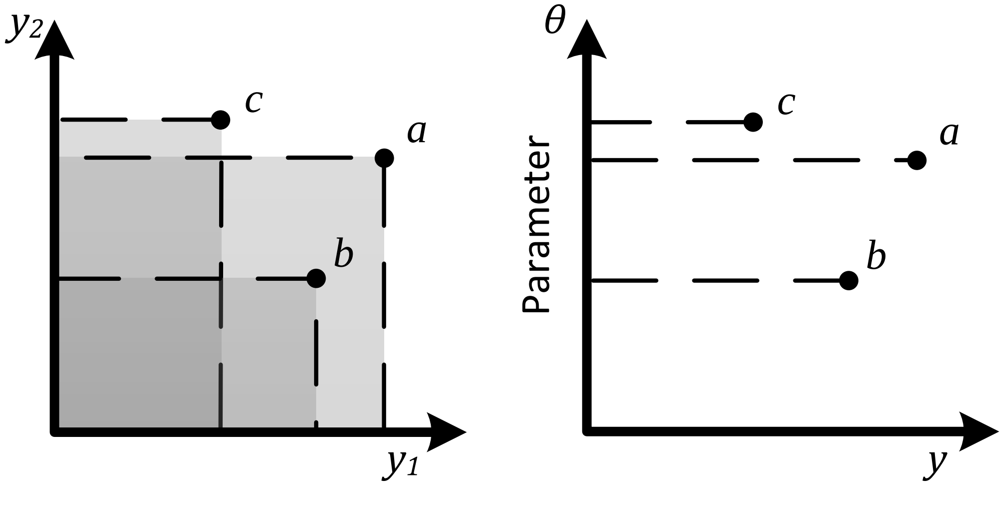
The first condition ensures that the chosen solution lies on the better non-dominated frontier. The second condition resolves the tie that occurs when both solutions are on the same non-dominated frontier. The solution in the less crowded area wins. Once the parents have been selected, the usual recombination and mutation operators are used to create offspring.

5.3 Predictive Parametric Pareto Genetic Algorithm (P3GA)

Although in p-NSGAII we seed each application of NSGAII, due to its sequential iterative nature "knowledge sharing" is limited to previous neighboring subproblems. That is, knowledge about the search space can only be shared unidirectionally by neighboring members. A more efficient algorithm would eliminate the need for iteration and would share knowledge of all solution found along the frontier.

Most MOGAs rely on the concept of dominance to assign higher fitness values to members closer to the PF. In these algorithms, two members are compared on the basis of whether one dominates the other or not. We adapt the MOGA scheme to the case with parameters by replacing the concept of Pareto dominance with parameterized Pareto dominance (PPD). Recall from Section 2.5 that under the PPD rule, a dominated alternative is one for which there exists an alternative with a more preferred objective value and equal parameter value. Under the PPD rule a non-dominated alternative will be necessarily closer to the solution of the parametric optimization problem. Thus, the PPD definition is compatible with the MPO problem formulation.

However, the complicating factor is that **alternatives must have equal parameter values for dominance to occur**. Because genetic algorithms are variants of randomized search algorithms, this is unlikely to occur. In other words, a straightforward application of PPD to the population members is unlikely to dominate ***any*** members. Figure 5.5 is an illustration of this difficulty. Fig. 5.5a depicts PD analysis in the case where preferences are monotonically decreasing for attributes y_1 and y_2. The space dominated by each member is shaded and outlined by the dashed lines. In the PD case in Figure 5a, member a is dominated by b while members c and b are mutually non-dominating. Fig. 5.5b depicts PPD in the case where the designer



(a) Minimize f_1 and f_2 (b) Minimize f_1, parameter f_2

Figure 5.5: Illustration of dominance analysis on randomly generated data using (a) the Pareto dominance rule and (b) the parameterized Pareto rule. [f]

has monotonically decreasing preference for attribute y_1 but has not established a preference order in y_2 (parameter). In b, **none** of the members can be said to be dominated since PPD can only occur along the dashed lines (parameter value equality). If no members are classified as dominated, there is no selection bias towards the frontier, which would result in poor search performance.

To overcome this difficulty, we instead perform dominance analysis using points that are *predicted* to be feasible rather than current members of the population. In the proposed algorithm, we use a machine learning technique to make this prediction using the current population as training data. A non-dominated member is one that is not parametrically Pareto dominated by any point that is predicted to be feasible. The following section contains a high-level description of the proposed algorithm and a more detailed explanation of the predicted dominance concept.

[f] Reprinted with permission from “P3GA: An Algorithm for Technology Characterization”, Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

Because of the reliance on predicted PPD, the proposed algorithm is called, Predictive Parameterized Pareto Genetic Algorithm (P3GA). In P3GA, we take the attribute space image of the current members of the population as training data for a machine learning algorithm: support vector domain description (SVDD). The resulting SVDD is then the predicted feasible set, i.e., the shaded region illustrated in Fig. 5.6. Using the SVDD approach, the p-nondominated members are those that are not dominated by a predicted feasible site, the blue shaded region in the figure.

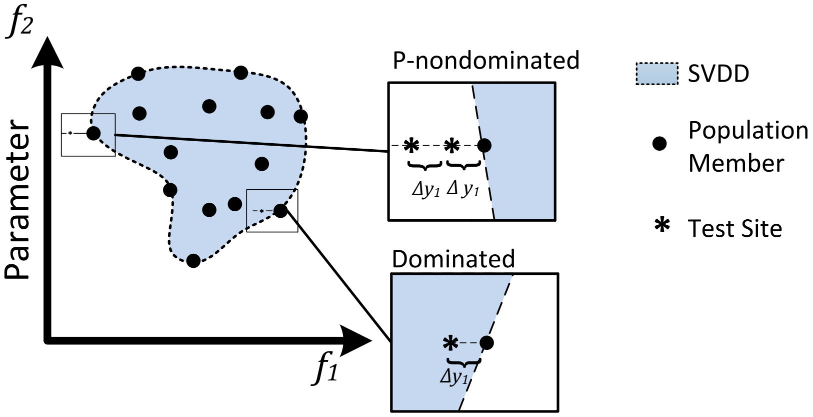


Figure 5.6: Illustration of p-dominance. Objective is to minimize f_2 and f_1 is a parameter. [g]

Figure 5.7 is a flow chart of the proposed algorithm. The algorithm closely parallels the general procedure of many MOGAs in the literature. In fact, P3GA draws extensively from the Nondominated Sorting Genetic Algorithm II (NSGAII) developed by K. Deb [26], i.e., the concepts of nondominated sorting and Crowded Tournament Selection. The novelty of P3GA is in the use of PPD as the dominance criterion and the concept of *predicted dominance*. Because many aspects of the

[g] Reprinted with permission from “P3GA: An Algorithm for Technology Characterization”, Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

algorithm are well documented in the MOGA literature, the following subsections detail only the novel portions of the algorithm: **(1)** determining the predicted feasible set and **(2)** predicted PPD dominance. These correspond to the shaded processes in Figure 5.7.

5.3.1 Predicting the Feasible Set using Support Vector Domain Description

The Support Vector Domain Description (SVDD) [116] is used to approximate the solution based on observed data. The SVDD method is a non-probabilistic machine learning technique for predicting the boundary of a data set in a Euclidean space. The SVDD method bears close resemblance to Support Vector Machines (SVMs) [117, 118]. The principal difference being that SVDD is a one-class classifier while SVMs are two-class classifiers. For a detailed description of the SVDD, see [116, 4]. Under the SVDD method, one finds the minimum-radius hyper-sphere that contains a set of training data. Let $\boldsymbol{x}_i$ denote a vector in the design variable space $\mathcal{X}$—the input space. Given n data points, $X = \{\boldsymbol{x}_i | i = 1, 2, \ldots, n\}$, the minimum radius hyper-sphere containing every data point with centroid $\boldsymbol{a}$ and radius r

$$\begin{aligned} \underset{r,\boldsymbol{a}}{\text{Minimize}} \quad & r^2 + c\sum_i \xi_i \\ \text{subject to} \quad & \|\boldsymbol{x}_i - \boldsymbol{a}\|^2 \leq r^2 + \xi_i, \;\; \xi_i \geq 0 \; \forall i \end{aligned} \tag{5.3}$$

where ξ_i are slack variables that allow for the possibility of outliers in the training set. The parameter c defines how to trade-off between hyper-sphere volume and errors. Any point at a distance equal to or less than r from the hyper-sphere center is inside of the domain description. However, because a hyper-sphere is typically a poor representation of the domain, a kernel function is used to non-linearly remap the training data into a higher-dimensional feature space where a hyper-sphere is a

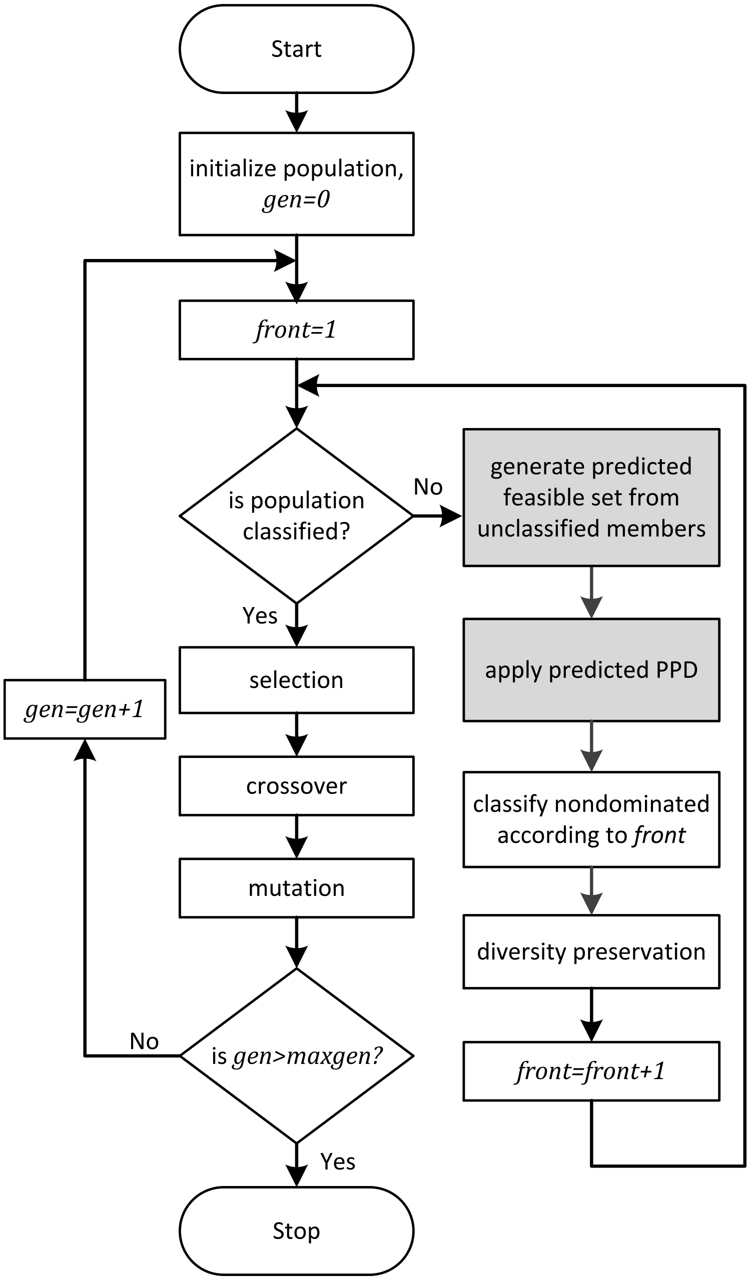

Start
initialize population, *gen=0*
front=1
is population classified?
No
generate predicted feasible set from unclassified members
apply predicted PPD
classify nondominated according to *front*
diversity preservation
front=front+1
Yes
selection
crossover
mutation
gen=gen+1
No
is *gen>maxgen?*
Yes
Stop

Figure 5.7: Flow chart of the Predictive Parameterized Pareto Genetic Algorithm (P3GA). The shaded processes correspond to the novel concepts implemented in P3GA. [i]

[i] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

good model. Through the so-called kernel trick, the data points are mapped to the feature space, without computing the mapping explicitly. The result is an implicit mapping to a feature space of unknown, and possibly infinite, dimensionality. There are several valid kernel functions common in the literature [119]. The proposed algorithm uses the Gaussian kernel function

$$K_G(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j) = e^{-q\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}, \tag{5.4}$$

where $\Phi(\cdot)$ is the nonlinear mapping from the data space to the feature space. The q parameter determines how "tightly" or "loosely" the domain description is fit around the training data. The constraint in Eq.5.3 then becomes

$$\|\Phi(\boldsymbol{x}_i) - \boldsymbol{b}\|^2 \leq r^2 + \xi_i \ \forall i, \tag{5.5}$$

where $\boldsymbol{b}$ is the centroid of the feature space hyper-sphere. Rewriting in terms of the kernel function, the Wolfe dual problem can be developed from Eq. 5.5 as

$$\begin{aligned} \underset{\beta_i}{\text{Maximize}} \quad & \sum_i \beta_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i,j} \beta_i \beta_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{subject to} \quad & 0 \leq \beta_i \leq c \ \forall i \\ & \sum_i \beta_i = 1 \end{aligned} \tag{5.6}$$

For a detailed description of the method for formulating the Wolfe dual problem see [120]. For each data point, $\boldsymbol{x}_i$ for $i = 1, 2, \ldots, n$, are three possible classifications:

- It is inside the hyper-sphere, which is indicated by $\beta_i = 0$,
- It is on the boundary of the hyper-sphere, which is indicated by $0 < \beta_i < c$,
- It is an outlier outside of the hyper-sphere, which is indicated by $\beta_i = c$.

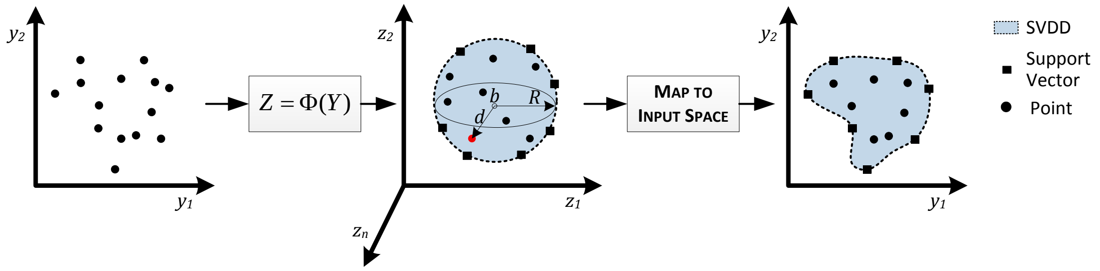
Data on the boundary of the hyper-sphere are called support vectors and are essential to the domain description representation. The outliers are not part of the domain description. Choosing $c \geq 1$ yields no outliers since $\sum_i \beta_1 = 1$ and $0 < \beta_i < c \; \exists i$, and therefore $\beta_i \neq c \; \forall i$. The squared distance of the feature space image of a point, $\boldsymbol{z}$, to the centroid of the hyper-sphere is

$$r^2(\boldsymbol{z}) = K(\boldsymbol{z}, \boldsymbol{z}) - 2 \sum_i \beta_i K(\boldsymbol{x}_i, \boldsymbol{z}) - \sum_{i,j} \beta_i \beta_j K(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{5.7}$$

A new test point, $\boldsymbol{z}$, is inside the domain description if the distance from the feature space image of test point to the hyper-sphere centroid and is less than the radius of the hyper-sphere. The expression for classification, En.5.7, is a simple algebraic expression that is fast to evaluate. In fact for the Gaussian kernel function, the first term is equal to 1, and the last term can be pre-computed since it is independent of $\boldsymbol{z}$.

A final consideration is the SVDD method is able to tighten the description by using negative examples—labeled outliers. The aim in this case is to find the minimum radius hyper-sphere that includes the positive examples (target-data) while excluding the negative examples (outlier-data). Consider that target-data are enumerated by indices i, j and the outlier-data by l, m. Further, the target-data are labeled $y_i = 1$ and outlier-data are $y_l = -1$. The search problem becomes

$$\begin{aligned} \underset{r,\boldsymbol{a}}{\text{Minimize}} \quad & r^2 + c_1 \sum_i \xi_i + c_2 \sum_l \xi_l \\ \text{subject to} \quad & \|\boldsymbol{x}_i - \boldsymbol{b}\|^2 \leq r^2 + \xi_i, \\ & \|\boldsymbol{x}_l - \boldsymbol{b}\|^2 \geq r^2 + \xi_l, \\ & \xi_i, \xi_l \geq 0 \; \forall i, l. \end{aligned} \tag{5.8}$$



(a) 2-D Input Space (b) n-D Feature Space (c) 2-D Input Space

Figure 5.8: Illustration of kernel based SVDD. The SVDD is a test to determine whether a point is inside of the boundary. [j]

Again, by the method of Lagrange multipliers, we can obtain

$$\begin{aligned} L = &\sum_i \beta'_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_l \beta'_l K(\boldsymbol{x}_l, \boldsymbol{x}_l) - \sum_{i,j} \beta'_i \beta'_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &+ 2\sum_{l,j} \beta'_l \beta'_j K(\boldsymbol{x}_l, \boldsymbol{x}_j) - \sum_{l,m} \beta'_l \beta'_m K(\boldsymbol{x}_l, \boldsymbol{x}_m) \end{aligned} \tag{5.9}$$

where $\beta'_i = y_i \beta_i$ (the index i again enumerates both target and outlier-data). See [121] for a detailed exposition of the SVDD method with negative examples.

To prevent weighting variables with large magnitudes more than those with lower ones in this comparison, the training data is centralizing (scale all data to a -1 to 1 range), which improves the SVDD model. An important benefit of the SVDD method is that it can be constructed incrementally and decrementally [122]. This allows for a relatively inexpensive update procedure to be used when new members are added or removed from the SVDD. Figure 5.8 is an illustration of the SVDD method on a 2-dimensional data set in the thermodynamic conditions space.

[j] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

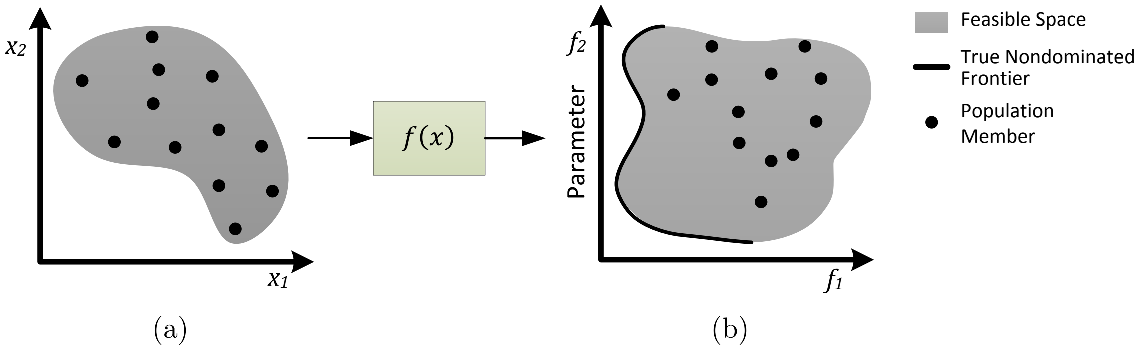


Figure 5.9: Random population members in (a) the design space and their (b) attribute space image. The shaded region corresponds to the true feasible space in both domains. Preference is to minimize f_1 and f_2 is a parameter attribute. [k]

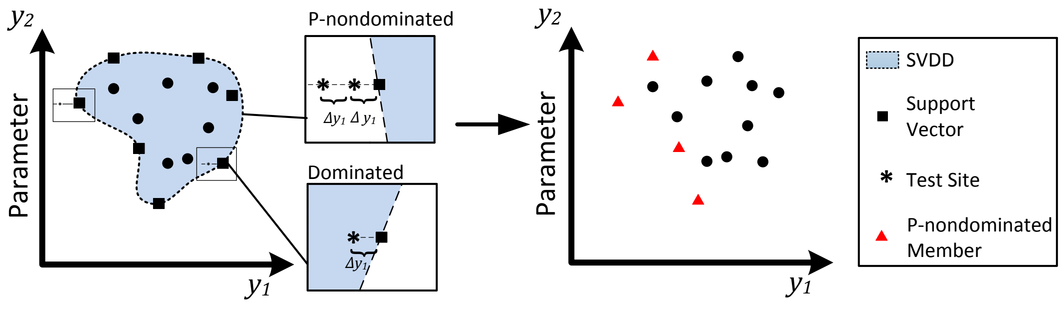
5.3.2 Predicted Dominance

In the proposed algorithm, the first step is to initialize the population by randomly selecting design alternatives in the feasible design space. These are mapped to the attribute space by an analysis function(s), see Figure 5.9. In Figure 5.9b, the feasible attribute space (shaded region) and true PPF (solid line) are unknown to the designer at this point, i.e., before solving the PMO problem. The next step is to use dominance analysis to identify designs that are likely closer to the true PPF, the bold line in Figure 5.9b. Next, we perform dominance analysis using points that are *predicted* to be feasible (p-dominance) rather than current members of the population.

Under the concept of predicted PPD, a non-dominated member is one that is not parametrically Pareto dominated by any member that is predicted to be feasible. For the remainder of this article, we will refer to this concept as *p-dominance*.

In the proposed algorithm, we take the attribute space image of the current members of the population (the points in Figure 5.9b) as training data for the SVDD.

[k] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.



(a) (b)

Figure 5.10: An illustration of predictive PPD. Objective is to minimize y_1, while y_2 is a parameter attribute.[1]

The resulting SVDD is then the predicted feasible set: the shaded region illustrated in Figure 5.10a. Any population member in the interior of the SVDD must be p-dominated. If every population member is part of the domain description, ($C \geq 1$), only members on the boundary of the SVDD (support vectors) can potentially be p-nondominated. Thus, the non-support vectors can be ignored when performing dominance analysis. This is significant since for most data sets only a small portion of the data will lie on the hyper-sphere boundary and be classified as support vectors.

Using the SVDD approach, the p-nondominated members are the support vectors that are not dominated by a **predicted** feasible site. To test this, we sample the SVDD in the space that dominates each support vector. Figure 5.10a is an illustration of this sampling procedure for two support vectors in the population. These test sites are in the space that dominates the support vectors. For each test point, we compute the distance from the point to centroid in the feature space image using Eq. (7). If the distance for any site is less than or equal to the radius of the hyper-sphere, i.e.,

[1] Reprinted with permission from "P3GA: An Algorithm for Technology Characterization", Galvan, E., Malak, R.J., 2016 J. Mech. Des. 137(1) Copyright 2015 by ASME.

predicted to be feasible, the support vector is classified as p-dominated. The test space is bounded by the minimum bounding hyper-rectangle (the envelope containing the training data). In the illustrative case, the result would be the non-dominated members in Figure 5.10b.

It is important to note that without the use of p-dominance, **none of the members shown in Figure 5.10 would be dominated**. This is because, as defined earlier, PPD requires that members have **equal** parameter values in order for dominance to occur; none of the members in the illustrated population have equal parameter values.

One approach to finding the p-nondominated members is to simply sample the space that dominates each support vector. If a any of the samples is inside of the SVDD, the support vector is classified as p-dominated. Under this approach, if there are n support vectors, u dominator attributes, and v samples along each dominator attribute, determining the non-dominated members requires at most nv^u evaluations of Eq. 5.7. In practice the number of evaluations required is much lower than this upper limit since many of the support vectors are classified as dominated after the first sample, see the *dominated* member in Figure 5.10a. However, this approach may still become prohibitively expensive for many dimensional problems.

Rather than densely sampling the (potentially many dimensional) space that dominates a point, we can instead leverage gradient information about the SVDD. Let $\boldsymbol{z}_k$ be the population member of interest. The direction of largest decrease in Eq. 5.7, the squared distance of the feature space, is

$$\nabla_k = -2 \sum_i \beta_i \frac{\partial K(x_i, \boldsymbol{z}_k)}{\partial \boldsymbol{z}_i} \tag{5.10}$$

If $\nabla_k < 0$ in each objective direction, $\boldsymbol{z}_k$ must be p-dominated since there exists predicted feasible members in the space that dominates $\boldsymbol{z}_k$. However, we cannot yet be confident that $\boldsymbol{z}_k$ is not p-dominated by disjoint region of the SVDD. Because the SVDD method allows disjoint regions and non-convexity, $\boldsymbol{z}_k$ may be p-dominated by a non-neighboring region. As a result, we must search the space that dominates $\boldsymbol{z}_k$ for predicted feasible sites. Let D be the set of indices corresponding to the objectives. The general outline of the method is

$$z_{k+1} = \boldsymbol{z}_k - \alpha_k \mathcal{D} \nabla_k, k \geq 0, \tag{5.11}$$

where $\alpha_k \geq 0$, and $\mathcal{D} = (d_{ij})$ is a diagonal matrix such that $d_{i,j} = 1$ if $i = j \; \forall i, j \in D$, and $d_{i,j} = 0$ otherwise. The matrix $\mathcal{D}$ ensures that we only step in the direction that parametrically dominates $\boldsymbol{z}_k$. In the method of gradient descent, α_k is chosen so that the descent property is satisfied:

$$R^2(\boldsymbol{z}_{k+1}) < R^2(\boldsymbol{z}_k) \tag{5.12}$$

Because the purpose of the search is not to find an extreme of the function, this approach is not suitable. For P3GA, we take a fixed step size. A step size of $\alpha_k = \sqrt{m} * 0.05$, where m is the number of objectives, was found to be produces very similar results to densely sampling the dominating space. The procedure is repeated until the member is found to be p-dominated or the search steps outside of the population bounding box.

5.4 Conclusions and Chapter Summary

In this chapter, we adapted multiobjective techniques for the general mutltiobjective parametric case. First, we considered some challenges in a straightforward application. We proved that the method alternating preference directions cannot, in general, find all solutions to an MPO problem. Next, we considered an iterative approach, where the MPO problem is decomposed into several MO subproblems with fixed parameters. We showed how constraining the subproblems to equidistant parameter values might lead to an uneven spread in solutions. Because an even spread of solutions is an important consideration in MO algorithms, we considered taking an approach similar to the Normal Boundary Intersect (NBI) method. We showed how an adaptation of the NBI method to PO could result in non-optimal solutions that cannot be filtered.

We developed an extension of the Nondominated Sorting Genetic Algorithm II (NSGAII) for the parametric case, which we call parametric NSGAII (p-NSGAII). In this iterative approach to MPO, the parameters are constrained to ranges rather than fixed values of the parameters. As a result, the algorithm is able to find solutions at arbitrary values of the parameter. To improve algorithm performance, each application of NSGAII is seeded with the results from the previous application. This allows each new application to "learn" from its predecessor. However, because of the sequential nature of p-NSGAII, this knowledge sharing is limited to neighbors and is unidirectional.

The predictive parametric Pareto genetic algorithm (P3GA) resolves this limitation by replacing the notion of dominance in the general genetic MO strategy with parametric Pareto dominance. Because it is unlikely that randomly generated alternatives will be parametrically dominated, P3GA instead relies on the concept of predicted dominance. By eliminating iteration, In P3GA, knowledge about one region of the search space can be communicated to any other solution, improving algorithm performance.

6. COMPARISON OF NOVEL ALGORITHMS

6.1 Experimental Setup

In this section, we compare p-NSGAII and P3GA on several test problems that exhibit features likely to cause difficulty in converging to the true PPF. We use the test problems developed in Section 4.3 and measure quality using the mean Hausdorff distance, described in Section 4.1.2.2.

Recall that the test problems were developed to allow the number of objectives and parameters to be scaled arbitrarily. This allows us to investigate how the performance of each algorithm changes with dimensionality. The number of decision variables is fixed at $n = 5 + m + p$, where m and p are the number of objective attributes and parameters, respectively. We compare the algorithms on test problems up to $m + p = 6$.

For each test problem, 30 runs are performed using either algorithm. The P3GA algorithm is run with the following

Generations:	$G =$	200
Population Size:	$P =$	100
Crossover Rate:		0.08
Mutation Rate:		0.01
Kernel function parameter:	$q =$	15
Max Num. of Training Data:		500

Thus, each P3GA run involves $PG = 2,000$ function evaluations. Recall that p-NSGAII involves solving a number of subproblems with NSGAII. For p_d discretizations along each parameter attribute, the number of subproblems is p_d^p. The to-

Table 6.1: p-NSGAII parameter settings as a function of the number of parameters p. The number of function evaluations (NFE) is greater than or equal to $20,000$, the NFE used in P3GA.

p	Pop. Size P	Num. Gen. G	p_d	NFE
0	100	200	N/A	20,000
1	23	58	16	21,344
2	23	58	4	21,344
3	32	80	2	20,480
4	23	58	2	21,344
5	16	40	2	20,480

tal number of function evaluations (NFE) for the p-NSGAII approach is PGp_d^p. To maintain a similar level of computational effort across algorithms, each application of NSGAII must about $20,000$ function evaluations. For this comparison, we let the number of subproblems be dependent on the number of parameters according to $p_d = \lfloor 16^{1/p} \rfloor$. The population size is set to the smallest size such that $2.5P^2\lfloor 20^{1/p}\rfloor^p \geq \frac{20{,}000}{p_d^p}$. The rule results in parameter settings in Table 6.1. We made no attempt to determine the best parameter settings. In both algorithms a crossover rate of 0.8 and mutation rate of 0.01 is used.

The p-NSGAII algorithm was implemented in C. The underlying NSGAII algorithm is original implementation of NSGAII with constraint handling by K. Deb [26]. The principal difference between NSGAII and P3GA is predicted parametric Dominance. To improve the quality of the comparison, we replaced Pareto dominance in the original implementation of NSGAII with predicted parametric Dominance in Section 5.3.2.

6.2 Results and Discussion

Figure 6.1 to 6.3 show a comparison between the performance of p-NSGAII and P3GA using the test problems developed in 4.3. The Δ_p values are normalized using the best and worst Δ_p values (lower is better) found in any test. Each figure corresponds to one of the scalable test problems A-C. For each test problem, the numbers of objective and parameter attributes were varied, with each of variations corresponding to a subplot. For example, in Figure 6.1, the top left subplot represents an experiment on test problem A with $m = 6$ objectives and $p = 0$ parameters. The top right subplot is another experiment where with $m = 1$ objective and $p = 5$ parameters. Note that the case with no parameters $p = 0$, is the multiobjective problem and p-NSGAII behaves as NSGAII. The aim is to show how the performance of each algorithm varies with the number of objective and parameters.

The boxplot indicates the average Δ_p for each method, p-NSGAII and P3GA. The notch indicates the 95% confidence interval. The whiskers indicate the upper and lower quantiles. Outliers are depicted as red crosses. For ease of visualization, the subplots corresponding to experiments where P3GA significantly (.95 confidence) outperformed p-NSGA2 are shaded green and outlined in bold. Those shaded in blue correspond to cases where p-NSGA2 significantly outperformed P3GA. The unshaded subplots correspond to the cases where there was no statistically significant difference.

The aim of test problem A is to test the ability of the algorithm to converge to a non-convex frontier, see Figure 4.10. The problem has uniform density along $f_1, f_2, \ldots, f_{m-1}$ and the parameter is constrained only by upper and lower bounds. Test problem A is a relatively straightforward MPO. In this test, p-NSGAII outperforms P3GA in the non-parametric case with 4 or less objectives. P3GA outperformed p-NSGAII in all parametric cases with more than 2 objectives and 2 parameters.

Test problem B is developed to have more challenging characteristics than test problem A. Test problem B features variable density along $f_1, f_2, \ldots, f_{m-1}$. The parameter value is a non-linear response of the design variables. The difficulty function

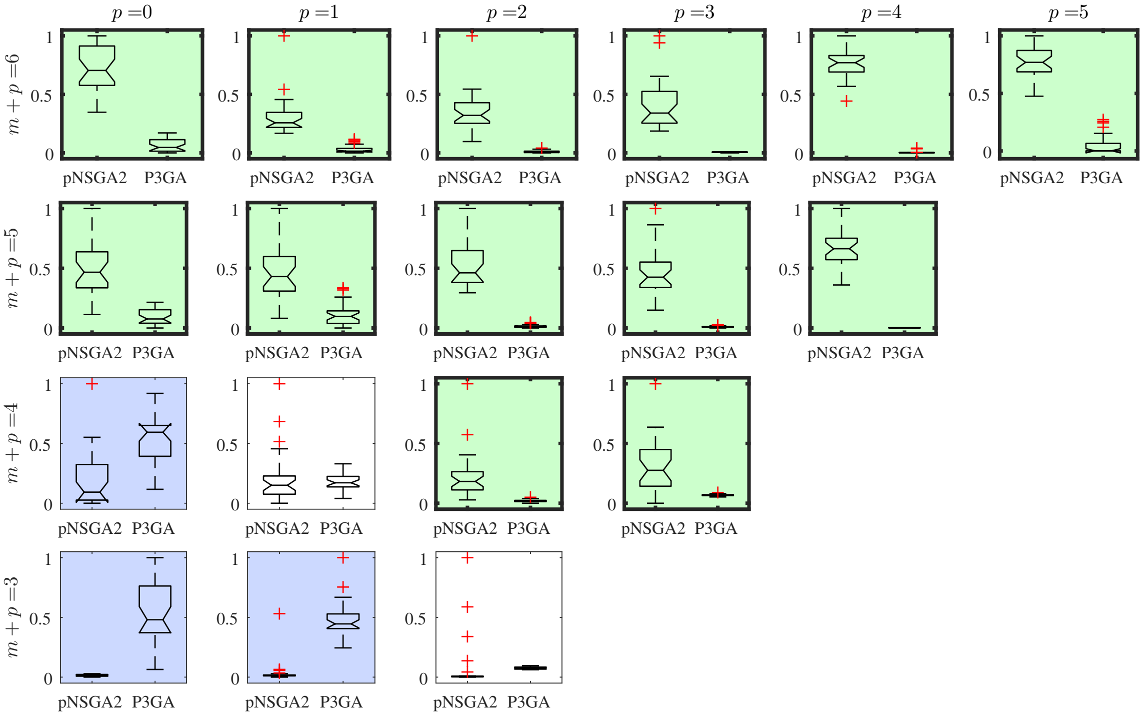


Figure 6.1: Comparison between p-NSGAII and P3GA on test problem A with varying numbers of objectives and parameters, each variation corresponds to a subplot. p-NSGAII outperforms P3GA in the non-parametric case with 4 or less objectives. P3GA outperformed p-NSGAII in all parametric cases with more than 2 objectives and 2 parameters.

is multi-modal, resulting in a number of local parametric Pareto frontiers. This test problem will test the ability of the algorithm to converge to a discontinuous frontier. For these tests, P3GA outperforms p-NSGAII for every parametric problem.

Test problem C is a variation of test problem B. The difference is the difficulty function g creates a "flatter" search region, which is often challenging for genetic algorithms. The problem does not feature local parametric Pareto attractors. In this case, p-NSGAII outperforms P3GA in the non-parametric case with 4 or less objectives. P3GA outperformed p-NSGAII in all parametric cases with more than 2 objectives and 1 parameter.

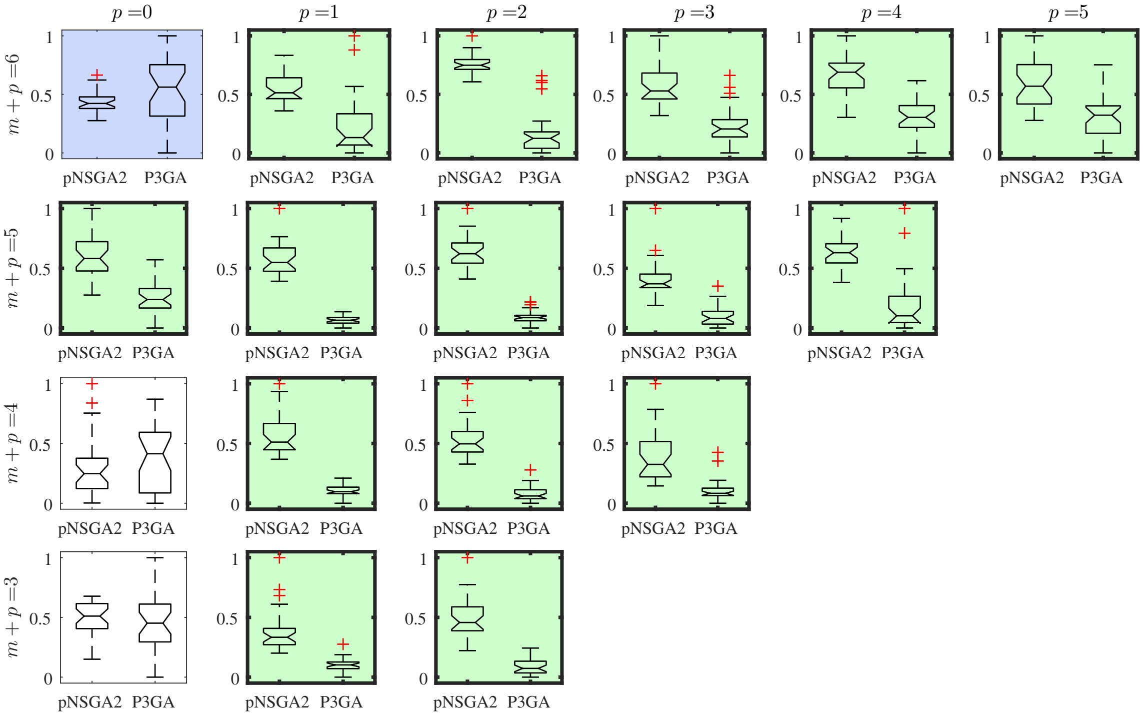


Figure 6.2: Comparison between p-NSGAII and P3GA on test problem B with varying numbers of objectives and parameters, each variation corresponds to a subplot.p-NSGAII only outperforms P3GA in the non-parametric case with 6 objectives.

The final problem considered in this study is test problem D. The aim of the problem is to test the ability of the algorithm to generate evenly spaced parametric Pareto optimal solutions in the case where the true frontier is a steep curve. The test problem features variable density along $f_1, f_2, \ldots, f_{m-1}$ and the difficulty function is the same as for test problem C. The purpose of this study is to gain an understanding of understanding of the issues that may arise in a constraint based approach to parametric problems, such as p-NSGAII. The p-NSGAII approach performs poorly in cases where the frontier is steep along the parameter attribute. This issue can be seen in Figure 6.5, which is an illustration of the solutions found by (a) p-NSGAII and (b) P3GA for one trial on test problem D. For p-NSGAII to more accurately (in terms of Δ_p) capture the true frontier, the number of discretizations along the

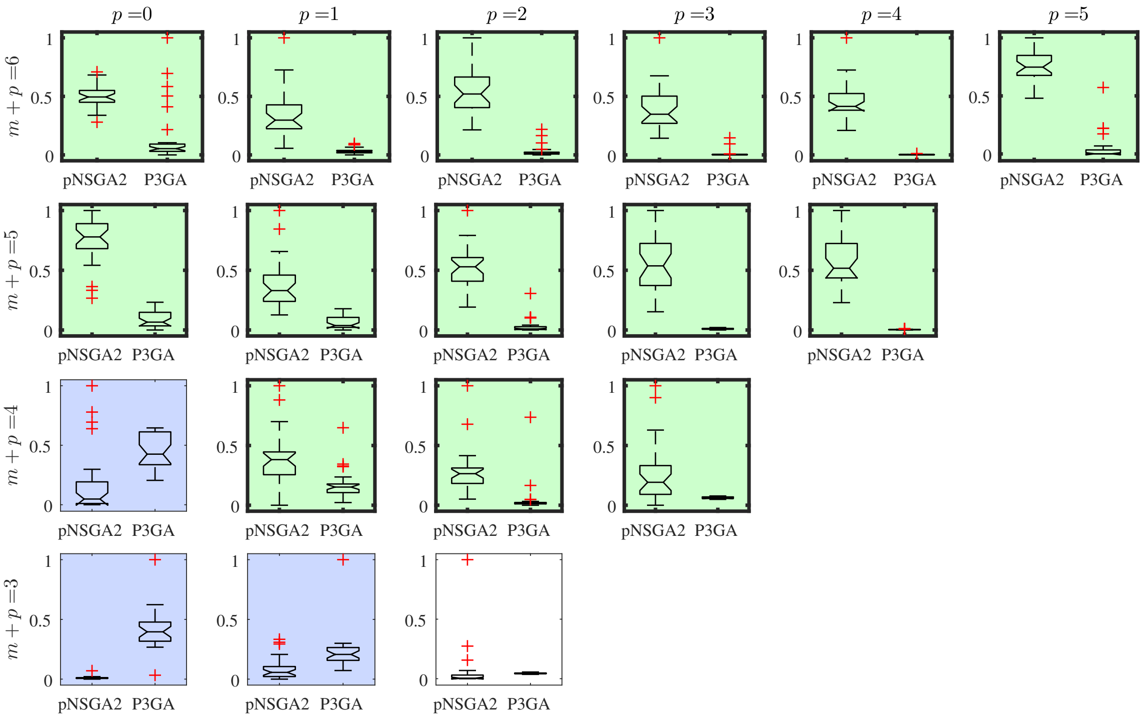


Figure 6.3: Comparison between p-NSGAII and P3GA on test problem C with varying numbers of objectives and parameters, each variation corresponds to a subplot. p-NSGAII outperforms P3GA in the non-parametric case with 4 or less objectives. P3GA outperformed p-NSGAII in all parametric cases with more than 2 objectives and 1 parameter.

parameter dimension, p_d, must be increased. However, this would increase the total number of function evaluations (NFE) dramatically since for p-NSGAII the NFE is PGp_d^p, where p is the number of parameters.

The results in Figures 6.1 to 6.3 indicate that P3GA generally performs better than p-NSGAII as the number of parameters and/or objectives increases. This is perhaps unsurprising since the performance of NSGAII is known to decrease dramatically with the number of objectives [37]. The method of p-dominance used in P3GA mitigates this effect to some degree. To better understand the ability of p-dominance to order solutions in terms of objective function values, we perform the same exper-

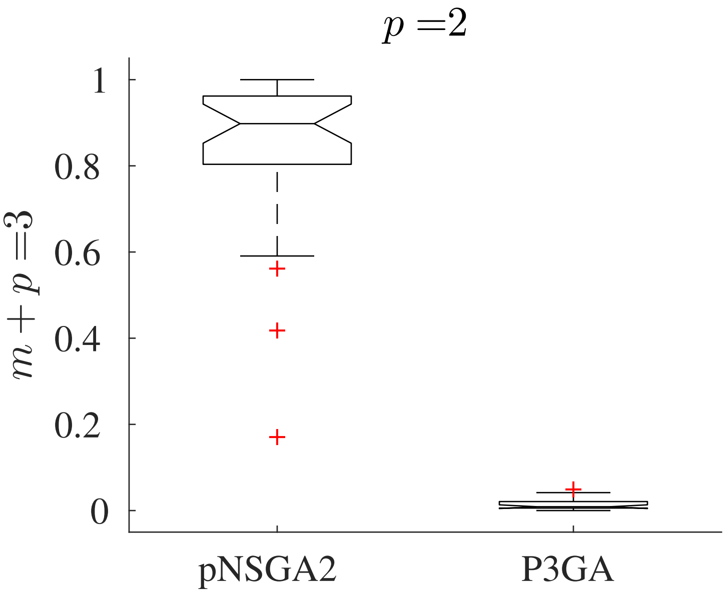
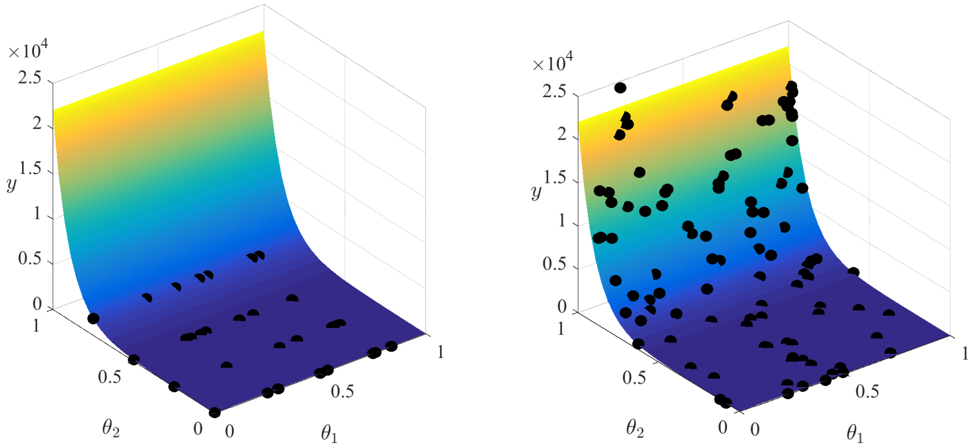


Figure 6.4: Comparison between p-NSGAII and P3GA on test problem D with one objective and two parameters. P3GA outperforms p-NSGAII.



(a) p-NSGAII

(b) P3GA

Figure 6.5: Illustration of Test Problem D.

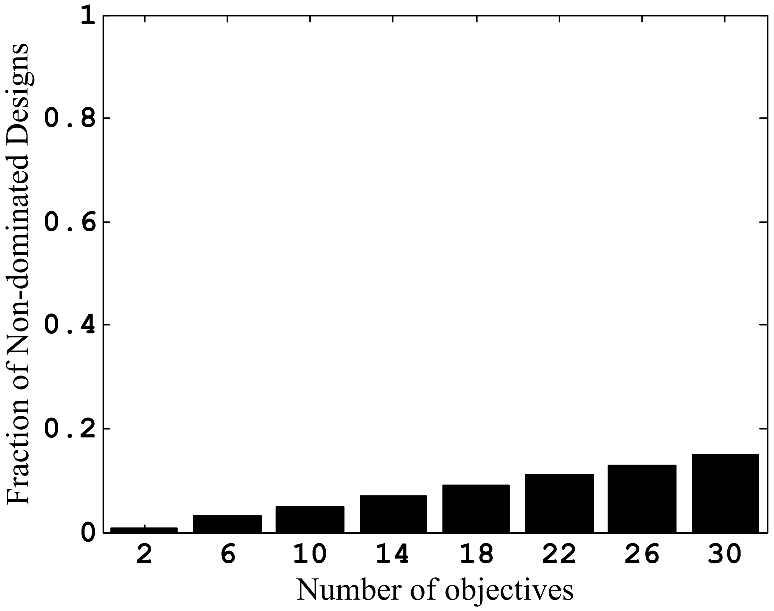


Figure 6.6: Mean fraction of predictive Pareto non-dominated designs from a set of 200 uniformly distributed designs in an M-dimensional unit hypercube.

iment reported in Figure 2.5 using p-dominance. We randomly generate 200 designs in an M-dimensional unit hyper-cube for $M = 2, 6, \ldots, 30$. The average percentage of non-dominated designs over 10 runs is presented in Figure 6.6. The results show that using p-dominance, only a small fraction of solutions are non-dominated, even in very high dimensions.

The ability of p-dominance to better order solutions in many objective problems does not explain why P3GA outperforms p-NSGAII for the parametric problems with low number of objectives, e.g., $m = 3$ and $p = 3$. The advantage of P3GA on parametric problems is likely a result of a combination of several factors: **(i)** improved ability to dominate solutions in many objective spaces **(ii)** improved knowledge sharing among solutions throughout the search space, and **(iii)** ability to maintain a more even spread of solutions along the parameter dimensions.

The results presented in Figures 6.1 to 6.4 are normalized to improve visualization. However, some performance information is lost when the data is normalized. We present the data without normalization in Tables 6.2 to 6.4. For all test problems and either algorithm, performance overall performance (as measured by Δ_P) tends to decrease with the dimensionality of the problem, i.e., $m + p$. This is unsurprising since as the dimensionality of the frontier increases, the search problem becomes more difficult. However, we should not interpret increasing values of $m + p$ at higher dimensions to mean that neither algorithm is able to converge to the true solution when dimensionality is increased. To illustrate this point, we repeated the some experiments with increased number of function evaluations. The results of these experiments are reported in Table 6.5. The total number of function evaluations (NFE) is increased to 40,000. The results presented are for test problems A-C, with $p = 5$ and $m + p = 6$. Doubling the NFE results in about a doubling in performance for P3GA. The p-NSGAII method sees some increase in performance for test problem C but none for A and B.

The most significant trend that can be seen in the data is that the performance of p-NSGAII **decreases** with the number of parameters. This result is to be expected due to the iterative nature of the p-NSGAII approach. A less intuitive trend is that the performance of P3GA **improves** with the number of parameters. The results in Figure 6.2 for test problem B provide some deep insight into the performance of P3GA and p-NSGAII in this respect. Test problem B is the only case where p-NSGAII outperforms P3GA for a high (more than 4) dimensional problem. To gain a better understanding of the issue at hand we provide an illustration of the solutions found by p-NSGAII and P3GA for test problem B in Figures 6.2 and 6.2, respectively. As expected, the performance of p-NSGAII decreases with the number of parameters. In essence, the ε constrained subproblems partition the search

Table 6.2: Comparison between p-NSGAII and P3GA on test problem A with varying numbers of objectives and parameters. Total number of function evaluations is NFE=20,000.

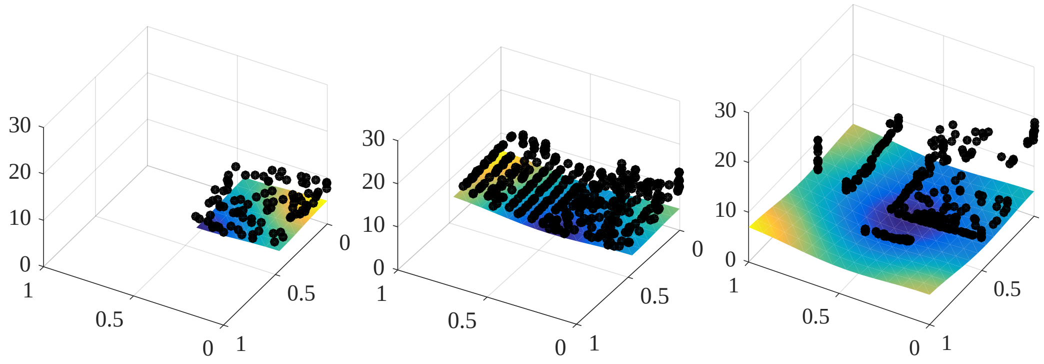
$m+p$	p	P3GA	p-NSGAII
3	0	0.41 ± 0.002	0.10 ± 0.000
3	1	0.18 ± 0.000	0.06 ± 0.000
3	2	0.16 ± 0.000	0.16 ± 0.002
4	0	1.62 ± 0.006	0.75 ± 0.006
4	1	0.67 ± 0.002	0.74 ± 0.005
4	2	0.38 ± 0.000	1.21 ± 0.008
4	3	0.32 ± 0.000	0.62 ± 0.003
5	0	2.34 ± 0.009	6.75 ± 0.030
5	1	1.34 ± 0.005	2.83 ± 0.010
5	2	0.72 ± 0.001	4.93 ± 0.018
5	3	0.53 ± 0.000	2.90 ± 0.011
5	4	0.46 ± 0.000	10.23 ± 0.028
6	0	2.56 ± 0.013	15.47 ± 0.044
6	1	1.72 ± 0.008	7.00 ± 0.035
6	2	1.18 ± 0.003	10.16 ± 0.050
6	3	0.80 ± 0.001	8.42 ± 0.044
6	4	0.70 ± 0.002	17.43 ± 0.031
6	5	1.52 ± 0.019	15.18 ± 0.030

Table 6.3: Comparison between p-NSGAII and P3GA on test problem B with varying numbers of objectives and parameters. Total number of function evaluations is NFE=20,000.

$m + p$	p	P3GA	p-NSGAII
3	0	1.45 ± 0.005	1.51 ± 0.003
3	1	1.15 ± 0.004	3.09 ± 0.014
3	2	0.99 ± 0.003	3.09 ± 0.010
4	0	3.38 ± 0.009	3.20 ± 0.009
4	1	2.25 ± 0.008	9.17 ± 0.030
4	2	1.91 ± 0.009	7.60 ± 0.022
4	3	1.64 ± 0.009	4.09 ± 0.021
5	0	6.49 ± 0.012	9.55 ± 0.019
5	1	3.87 ± 0.010	16.63 ± 0.004
5	2	2.97 ± 0.013	14.56 ± 0.003
5	3	2.84 ± 0.020	9.35 ± 0.035
5	4	3.83 ± 0.040	10.17 ± 0.002
6	0	26.10 ± 0.013	21.73 ± 0.004
6	1	13.09 ± 0.011	24.91 ± 0.006
6	2	8.68 ± 0.054	23.22 ± 0.002
6	3	8.98 ± 0.050	18.50 ± 0.006
6	4	9.95 ± 0.046	20.26 ± 0.005
6	5	10.29 ± 0.005	16.27 ± 0.005

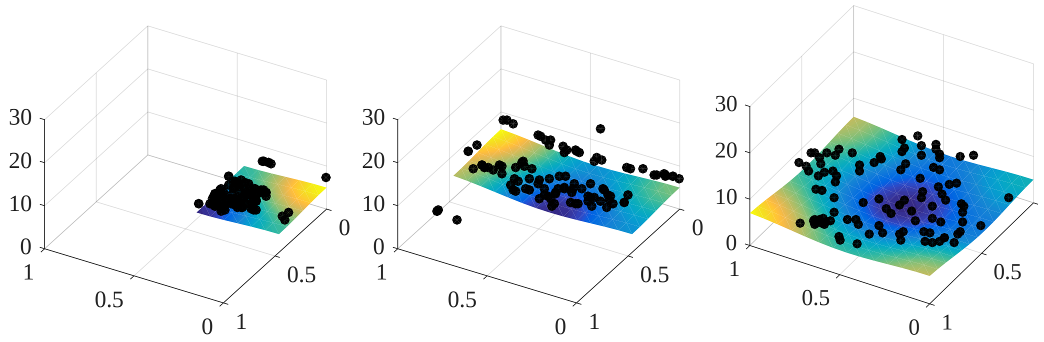
space. Then, the algorithm finds the PF for each partitioned subspace. This creates degenerate solutions in some of the ε constrained subproblems, that is, the PF within an ε constraint subproblem may be a point. These degenerate solutions create the "clusters" around the ε constraint values in Figure 6.2c.

To better understand why the performance of P3GA may increase with the number of parameters, it is useful to consider the notional scenario illustrated in Figure 6.9. The scenario depicts a situation where predicted dominance fails to accurately identify the non-dominated members. In the scenario, both attributes are to be minimized. We consider the case without parameters to simplify exposition. When



(a) 0 Parameters (b) 1 Parameter (c) 2 Parameters

Figure 6.7: Illustration of the solutions found by p-NSGAII for test problem B for a single trial . The performance of p-NSGAII **decreases** with the number of generations.



(a) 0 Parameters (b) 1 Parameter (c) 2 Parameters

Figure 6.8: Illustration of the solutions found by P3GA for test problem B for a single trial. The performance of P3GA **improves** with the number of generations.

Table 6.4: Comparison between p-NSGAII and P3GA on test problem C with varying numbers of objectives and parameters. Total number of function evaluations is NFE=20,000.

$m+p$	p	P3GA	p-NSGAII
3	0	0.62 ± 0.002	0.10 ± 0.000
3	1	0.37 ± 0.002	0.18 ± 0.001
3	2	0.20 ± 0.000	0.24 ± 0.004
4	0	1.87 ± 0.006	0.85 ± 0.011
4	1	0.78 ± 0.003	1.32 ± 0.007
4	2	0.56 ± 0.007	1.59 ± 0.009
4	3	0.35 ± 0.000	0.74 ± 0.006
5	0	2.53 ± 0.011	13.23 ± 0.035
5	1	1.53 ± 0.005	3.93 ± 0.018
5	2	0.87 ± 0.006	4.87 ± 0.017
5	3	0.55 ± 0.000	4.15 ± 0.018
5	4	0.49 ± 0.000	7.05 ± 0.027
6	0	9.68 ± 0.145	28.16 ± 0.059
6	1	1.88 ± 0.008	10.89 ± 0.069
6	2	1.50 ± 0.012	12.07 ± 0.048
6	3	0.98 ± 0.008	10.11 ± 0.052
6	4	0.68 ± 0.001	15.09 ± 0.054
6	5	2.22 ± 0.046	26.53 ± 0.051

Table 6.5: Comparison between p-NSGAII and P3GA on test problem A to C with $p = 5$ and $m + p = 6$ Total number of function evaluations is NFE=40,000.

Test Problem	P3GA	p-NSGAII
A	0.65 ± 0.001	19.37 ± 0.032
B	5.90 ± 0.021	16.50 ± 0.029
C	0.67 ± 0.001	12.57 ± 0.031

the domain description is generated around the initial population, the boundary members (support vectors) are even more widely spaced in the objective space. As a result, predictive Pareto dominance fails to rank the solutions, since almost all solutions are classified as dominated.

In other words, in cases where the solution space is small relative to the search space, the SVDD technique may generate a poor prediction near the true solution. See [123] for another example of this phenomenon. In the notional example in 6.9, the prediction is too “loose” near the solution and as a result, few of the population members are identified as non-dominated (highlighted in red). If we were to “zoom” into the predicted frontier, we can see how this may result in incorrectly labeling population members.

This limitation of the P3GA is relevant to test problem B, but only in the case with few parameters. As can be seen in Figures and , the size of the solution relative to the search space increases with the number of parameters. Conversely, the size of the search space *sometimes* decreases with the number of objectives, as is the case with test problem B. This drawback could potentially be addressed by introduction an approach for eliminating population members that are far from the non-dominated frontier from the domain description. This is left to future work.

Another important consideration in assessing the performance of a search algorithm is the internal wallclock time. The internal wallclock time of an algorithm is the human perception of time required for the internal computations involved in the search, that is those computations other than function evaluations. Internal computation time is typically not reported for genetic algorithms since it tends to be small relative to the external computation time (the time required to perform the function evaluations). The internal wallclock time of the p-NSGAII is on the order of minutes (many of the runs were completed in less than one minute). However, because P3GA involves a machine learning step, the computation time is not negligible.

The internal wallclock of P3GA is reported in Figures 6.10 to 6.12. The mean wallclock time is reported in minutes with varying numbers of objective and pa-

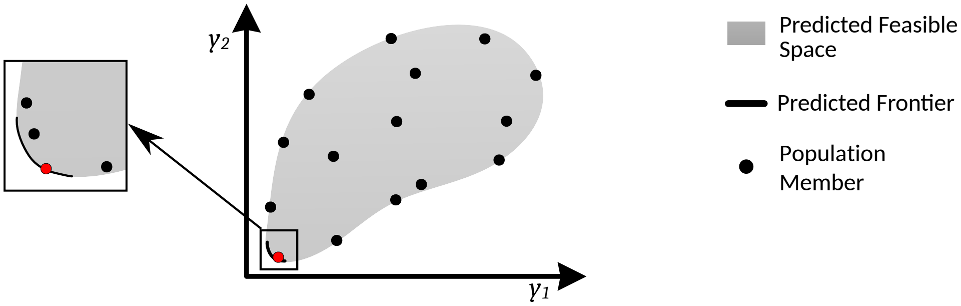


Figure 6.9: Situation where predicted Dominance fails to identify non-dominated members. The predicted non-dominated member lies on the predicted frontier and is highlighted in red. If we focus near the predicted frontier, we can see that any members near that predicted non-dominated member (in red) would be incorrectly labeled as dominated.

rameters. Each symbol corresponds to a fixed number of parameters. The x-axis corresponds to the total number of objectives and parameters, i.e., the dimensionality of the solution space. For test problem A, most problems are solved in about 10 minutes but no clear trend can be seen with varying number of objective and parameters. For test problems B and C, however, the wallclock time increases with the number of objectives. An interesting trend is that wallclock time *decreases* if one objective becomes instead a parameter. Because the time to build the SVDD should not be affected by whether a dimension is a parameter or not, we speculate that computation savings are in the computation of p-dominance. When one objective becomes a parameter, the search space when performing p-dominance, Eq. 5.3.2, decreases in dimensionality, leading to computational savings.

6.3 Conclusions and Chapter Summary

In this chapter, we compare p-NSGAII and P3GA on several test problems that exhibit features likely to cause difficulty in converging to the true frontier. We

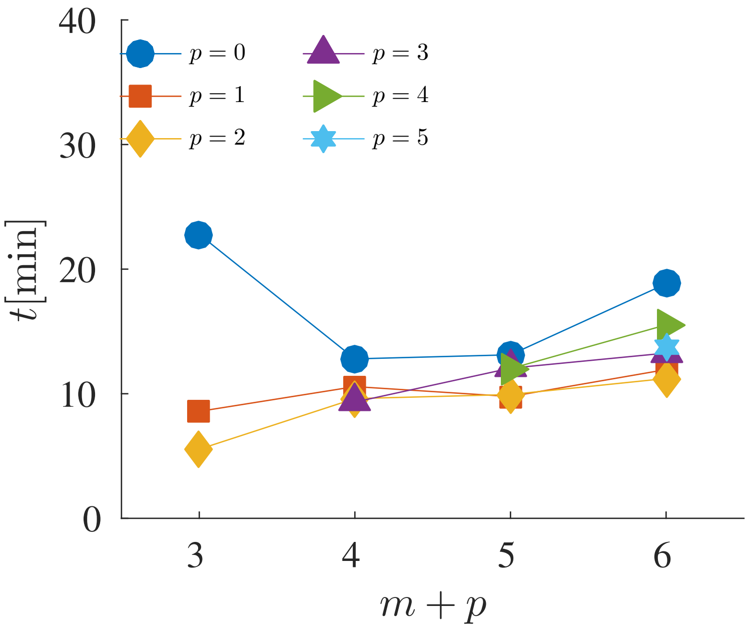


Figure 6.10: Mean wall-clock time in minutes of the P3GA algorithm on test problem A with varying numbers of parameters and objectives. Each symbol corresponds to a fixed number of parameters. The x-axis corresponds to the total number of objectives and parameters, i.e., the dimensionality of the solution space.

use the test problems intended to cause difficulty in converting to the true frontier, especially for genetic algorithms. The performance was measured in terms of solution quality (measured using mean Hausdorff distance), number of function evaluations, and wallclock time.

The p-NSGAII algorithm was implemented in C, leveraging the original implementation of NSGAII by K. Deb [26]. To improve the quality of the comparison, P3GA was implemented by replacing Pareto dominance in the original implementation of NSGAII with predicted parametric Dominance in Section 5.3.2.

The results indicate that P3GA generally performs better than p-NSGAII (in terms of Δ_p) as the number of parameters and/or objectives increases. For low dimensional cases without parameters (i.e., traditional multiobjective problems) the

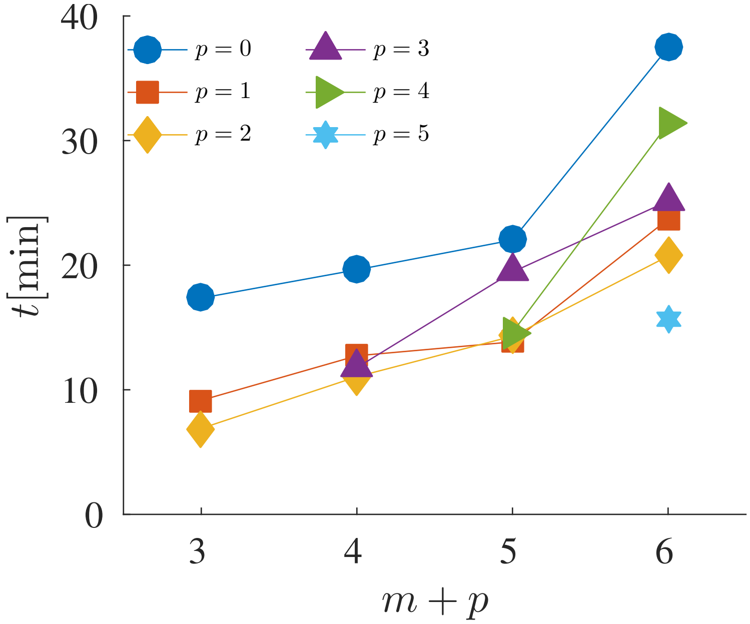


Figure 6.11: Mean wall-clock time in minutes of the P3GA algorithm on test problem B with varying numbers of parameters and objectives. Each symbol corresponds to a fixed number of parameters. The x-axis corresponds to the total number of objectives and parameters, i.e., the dimensionality of the solution space.

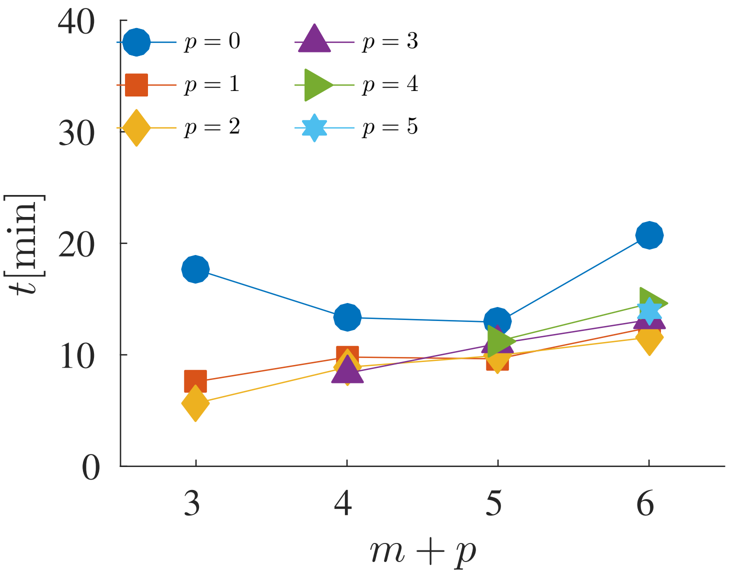


Figure 6.12: Mean wall-clock time in minutes of the P3GA algorithm on test problem C with varying numbers of parameters and objectives. Each symbol corresponds to a fixed number of parameters. The x-axis corresponds to the total number of objectives and parameters, i.e., the dimensionality of the solution space.

p-NSGAII method tended to outperform P3GA. This is expected since without parameters, p-NSGAII reduces to the original NSGAII, which is developed to solve multiobjective problems. For those problems that included parameters, P3GA tended to outperform.

Some of the performance gain for P3GA can be explained by the a ability of p-dominance to better order solutions in many objective. However, this would not explain why P3GA outperforms p-NSGAII for the parametric problems with low number of objectives. The advantage of P3GA on parametric problems is likely a result of a combination of several factors: **(i)** improved ability to dominate solutions in many objective spaces **(ii)** improved knowledge sharing among solutions throughout the search space, and **(iii)** ability to maintain a more even spread of solutions along the parameter dimensions.

A drawback of P3GA is an increase in internal wallclock time. The computation time of genetic algorithms is typically negligible. However, because P3GA implements a machine learning technique, computation time is significantly increased. For the test problems considered in this study, the internal wallclock time increased with the number of objectives but decreased if an objective attribute becomes a parameter.

7. INFORMATION GATHERING CASE STUDY

7.1 Multi-ratio Transmission Model

To illustrate the use of parametric optimization for the information gathering problem, we consider the design of a vehicle's multi-ratio transmission with uncertain information on its engine characteristics. Aside from transmitting power from the engine to wheels, an important function of a multi-ratio transmission is to allow the engine to run at a more efficient state, thus improving fuel economy. Therefore, at a high level, we wish to design a multi-ratio transmission that maximizes engine efficiency.

The engineering problem is to compute the expected value of performing physical experiments to gather information about the uncertain information in a computational model the efficiency of an engine and transmission. In Section 3.1 we developed a method of using parametric optimization to reduce the computational expense of computing the expected value of perfect information. In this chapter, we compare the proposed approach against a "naive approach:" simple Monte Carlo.

The engineering models in this chapter are from [124], developed by Vermillion. Engine efficiency is modeled as the ratio of brake power to fuel power such that

$$\eta_e = \frac{\Pi_e}{\dot{m}_f \Delta H_c^0} \tag{7.1}$$

where $\dot{m}_f$ is the fuel mass flow rate into the engine and ΔH_c^0 is the lower heating value of the fuel [125]. The ratio of fuel mass flow rate, $\dot{m}_f$, and brake power, Π_e, is called the Brake Specific Fuel Consumption (BSFC) such that

$$g_e = \frac{\dot{m}_f}{\Pi_e} \tag{7.2}$$

The BSFC is a measure of fuel economy and is determined experimentally. Using Eq. 7.2, engine efficiency is redefined as

$$\eta_e = \frac{1}{g_e \Delta H_c^0} \tag{7.3}$$

and since ΔH_c^0 is constant for a given fuel type, an expression for g_e is needed. In [126], Golverk models the BSFC map using a second order polynomial:

$$g_e(\omega_e, T_e) = K_1 + K_2\omega_e + K_3 T_e + K_4 \omega_e^2 + K_5 \omega_e T_e + K_6 T_e^2 \tag{7.4}$$

where K_i for $i = 1, \ldots, 6$ are experimentally determined coefficients, and T_e and ω_e are the engine torque and speed outputs, respectively; these coefficients vary with engine type and model.

The conveyance of power through the transmission can be written as

$$2T + \Xi_t \xi_d T_e = 0 \text{ and } -\Xi_t \xi_d \omega + \omega_e = 0 \tag{7.5}$$

where T and ω are the torque and speed demanded by a single wheel. Manring *et al.* model the torque demand at a single wheel as the following [27]

$$T = \frac{1}{2} m R^2 \frac{d\omega}{dt} + \left(\frac{\rho C_d A}{4} R^3 \right) \omega^2 + \frac{1}{2} C_r m g R \tag{7.6}$$

where m is vehicle mass, R is tire radius ρ, is the density of air, C_d is the dimensionless

Table 7.1: Fixed simulation parameters.

Symbol	Description	Value	Units
A	Frontal area	2.00	m^2
C_d	Drag coefficient	0.40	dimensionless
C_r	Rolling resistance coefficient	0.01	dimensionless
m	Vehicle mass	1347	kg
R	Tire radius	0.305	m
ξ_d	Differential speed ratio	3.55	dimensionless

drag coefficient for the vehicle, A is the effective frontal areal of the vehicle, C_r is the dimensionless rolling-resistance coefficient, and g is the gravitational constant. Tire angular speed and acceleration are assumed known a priori from a driving schedule.

Working from Eq. 7.6 back to Eq. 7.3, engine efficiency is indeed dependent on the adjustable gear ratio Ξ_t, as the parameters in 7.1, which are assumed to be fixed.

In the design of the multi-ratio transmission, we assume it is to be a five speed transmission that has the following shifting schedule:

$$\Xi_t = \begin{cases} \xi_1 & : 0.0 \leq \hat{\omega} < 0.4 \\ \xi_2 & : 0.4 \leq \hat{\omega} < 0.6 \\ \xi_3 & : 0.6 \leq \hat{\omega} < 0.8 \\ \xi_4 & : 0.8 \leq \hat{\omega} < 1.0 \\ \xi_5 & : 1.0 \leq \hat{\omega} < \infty \end{cases}$$

where $\hat{\omega} = \omega/73.3$. Our figure of merit for the quality of a set of gear ratios is the average engine efficiency when put through a driving schedule characterized by the following angular speed and acceleration profiles:

$$\omega = 18.325 + 0.733t \text{ and } \frac{d\omega}{dt} = 0.733 \tag{7.7}$$

where $t \in (0, 100)$. As mentioned earlier, the coefficients of the BSFC map in Eq. 7.3 are determined experimentally, so a meta-level decision must be made as whether to design the transmission given some belief on what these parameters may be or to conduct a series of physical experiments to calculate them.

7.2 Single-Objective Parametric Optimization Problem

As mentioned in the previous section, the coefficients of the BSFC map in Eq. 7.3 are determined experimentally, so a meta-level decision must be made as whether to design the transmission given some belief on what these parameters may be or to conduct a series of physical experiments to calculate them. To compute the EVPI on these coefficients, we must solve

$$\underset{\boldsymbol{k}}{E}\left[\, f\left(\boldsymbol{x}, \boldsymbol{k}\right)\right] - C \tag{7.8}$$

$$= \int_{\boldsymbol{k}^L}^{\boldsymbol{k}^U} \left(\max_{\boldsymbol{x} \in X} f\left(\boldsymbol{x}, \boldsymbol{k}\right) \right) p\left(\boldsymbol{k}\right) d\boldsymbol{k} - C \tag{7.9}$$

where $\boldsymbol{k} = (k_1, k_2, k_3, k_4, k_5)^T$, is a vector corresponding to the random coefficients that can be determined experimentally, $p\left(\boldsymbol{k}\right)$ is the probability density function corresponding to the random vector $\boldsymbol{k}$, $f\left(\boldsymbol{x}, \boldsymbol{k}\right)$ is a profit model, $\boldsymbol{x} = (x_1, x_2, x_3, x_4, x_5)^T$ is a vector corresponding to the design of the transmission, specifically, the gear ratios, and C is the cost of conducting the experiment.

7.3 Frontier Sampling

Once the parametric optimization problem has been solved, the next step is to sample the solution $\boldsymbol{y}^*(k)$ according to the distribution $P(K)$ to recover the solution to Equation 3.5.

If the solution to Equation 7.2 is continuous, such as would be the case for the mp-

QP algorithm presented in Section 2.2.5. Sampling the solution is straightforward. However, if the solution to Equation 7.2 is an approximation setas is the case for p-NSGAII and P3GA, sampling becomes more challenging.

Consider the notional example illustrated in Figure 7.1. The feasible range of the function $f(\boldsymbol{x}, \boldsymbol{k})$ is the shaded region, the true solution to the parametric problem is the dashed line. The output of an approximation such as P3GA would be a set of points distributed along the true solution.

There are a number of potential techniques to address this challenge. For example, importance sampling can be used to estimate the expected value of the desired distribution. Using importance sampling, we can estimate the properties of the desired distribution, Equation 3.5, with the samples generated from a different distribution, such as the samples generated by P3GA [127]. However, importance sampling requires that we know the original distribution. Since P3GA is a randomized search, there are no guarantees on the distribution of the sample set. As an initial step in this research, we interpolated between the non-dominated points using Kriging interpolation. The Kriging model then serves as a surrogate of $\max_{x \in X_f}\ \mathrm{f}\,(x, y)$ in Equation 3.5.

Interpolation is a curve-fitting method in which the model passes through all the data points. The model uses the relative location of the data points to each other in its fitting process. Interpolation assumes that the closer the input data points are to each other, the more positively correlated their outputs are. Kriging is a geostatistical interpolation method that predicts unknown values from the observed input/output relationships [128, 129]. Kriging is intended for spatially distributed data and considers both the distance and the degree of variation between the observed data points to predict unknown values. Kriging can be interpreted as a Gaussian process where, given the observed samples, one takes a multivariate Gaussian with

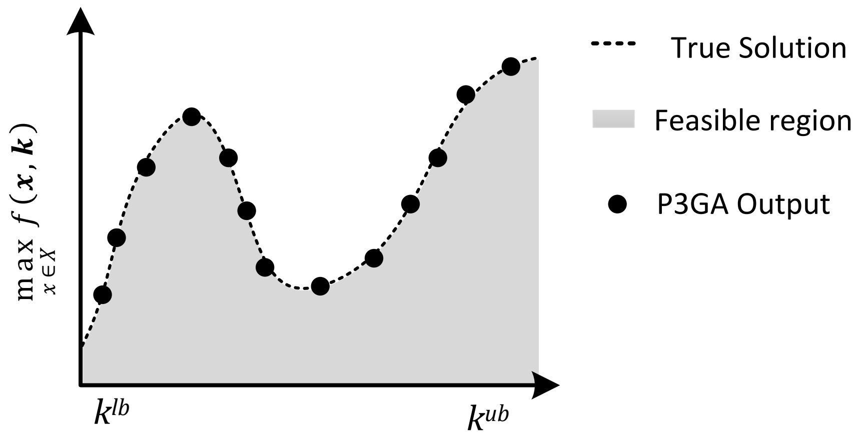


Figure 7.1: Notional illustration of P3GA output for a parametric optimization problem.[m]

some desired kernel and samples from that Gaussian. Then, the unknown points are predicted by combining the Gaussian prior with the Gaussian likelihood function. Kriging interpolation is accomplished using different methods depending on the assumptions made about the sample distributions. In this research, we use "Ordinary Kriging" throughout [130]. Using this Kriging method, the predicted value of a new unobserved input is a weighted linear combination of all the previously observed outputs. The following equations describe the Ordinary Kriging model:

$$\hat{Y}(\boldsymbol{x}_{n+1}) = \sum_{i=1}^{n} \lambda_i \cdot Y(\boldsymbol{x}_i) = \lambda^T \cdot \boldsymbol{Y} \tag{7.10}$$

where $\lambda = (\lambda_1, \ldots, \lambda_n)^T, \sum_{i=1}^{n} \lambda_i = 1, \boldsymbol{Y} = (Y(\boldsymbol{x}_i), \ldots, Y(\boldsymbol{x}_n))^T$, and $\boldsymbol{x}_{n+1}$ denotes the unobserved input, $\hat{Y}(\boldsymbol{x}_{n+1})$ denotes the predictor for the input, $\boldsymbol{x}_i$ are the n previously observed output, λ_i are called the Kriging weights and capital letters denote random variables that are determined through a fitting process. For more a

[m] Reprinted with permission from "A Parallel Approach for Computing the Expected Value of Gathering Information", Galvan, E., Hsiao, C., Vermillion, S., & Malak, R., 2015. SAE Int. J. Mater. Manf., 8(2):271-282, Copyright 2015 by SAE International.

more detailed description and analysis of ordinary Kriging see [130].

We use the DACE Kriging tool developed for MATLAB by Lophaven *et al.* to model the data sets in the example [131]. We use the linear correlation function and a 1^{st} order polynomial regression function in all cases. The interpolation model predicts the value of one attribute given the values of the others. Thus, the model predicts the value of $\max_{x \in X} f(\boldsymbol{x}, \boldsymbol{k})$ for a given $\boldsymbol{k}$, using the P3GA solution as a training data.

With the Kriging model in place, it is straightforward to sample the model according to $P(K)$ to approximate the solution to Equation 3.5. This sampling procedure is computationally inexpensive since it does not involve evaluating the engineering analysis function.

7.4 Experimental Setup

To evaluate the effectiveness of the proposed approach, we measure its performance against a "naive approach." Under the naive approach we approximate the EVPI as

$$\frac{1}{N}\sum_{i=1}^{N}\left(\max_{x \in X} f(\boldsymbol{x}, \boldsymbol{k}_i)\right) - C \tag{7.11}$$

where N is the number of Monte Carlo samples, and $k_i \sim P(K)$. We use $N = 10$, 20, 40, 60, 90, and 100 to capture the effect of increased Monte Carlo samples on solution quality and computational expense. To solve each search problem in the Monte Carlo simulation, we used Matlab's build in gradient optimizer *fmincon* with default options. Each search is terminated when the default termination criteria was met. For completeness, we also conducted a similar experiment with Matlab's single objective genetic algorithm, *ga*. Since Monte Carlo simulations are stochastic, each experiment was conducted 40 times to gather statistical information on the performance of the approach.

For the parametric approach, we executed P3GA 40 times. Each run was conducted using the following parameters

Population Size:	$P =$	100
Crossover Rate:		0.08
Mutation Rate:		0.01
Kernel function parameter:	$q =$	15
Max Num. of Training Data:		500

We made no attempt to determine the best parameter settings. To measure the effect of increased function evaluations, the experiments were repeated for Generations G =20, 40, 60, 80, 100, 120, 140, 150. For P3GA, the number of function evaluations for each run is PG.

The results, of both the proposed approach and the naive approaches are compared to the results of a brute force approach. For the brute force approach, Eq. 7.11 was solved $N = 5000$ using *fmincon*. For each sample, 50 random restarts were performed to mitigate the effects of any local optima. This result was used as a baseline to which we compare all other results. Note that this baseline is in terms of "value or profit" to the designer and is not associated with any particular gear design. Error is measured as the absolute difference between the calculated EVPI using either approach and this baseline.

7.5 Results and Discussion

For visualization purposes, the error is normalized against the highest error observed in any experiment. Figures 7.2 and 7.3 depict the results of the experiment. Figure 7.2 is a comparison between the Parametric approach and the Naive approach

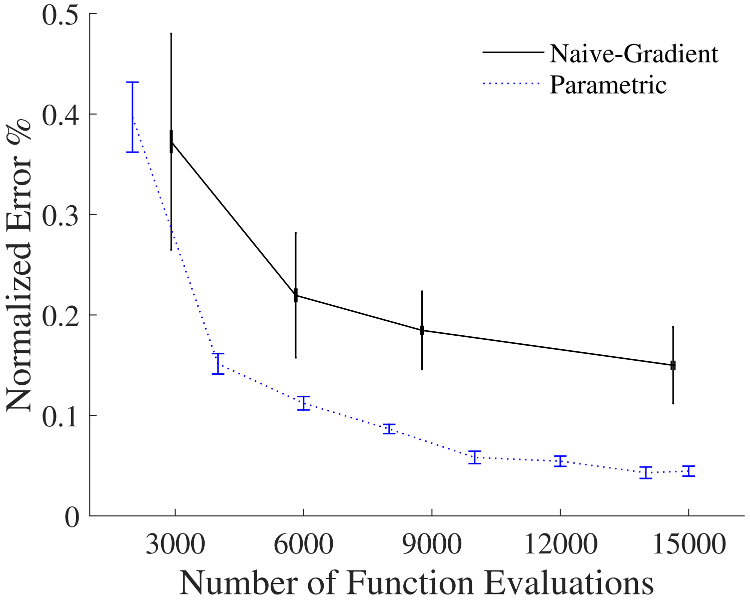


Figure 7.2: Comparison between the Parametric approach and the Naive approach using gradient based optimization. The parametric approach converges to a solution with less error. [n]

using gradient based optimization. The parametric approach converges significantly faster and has lower variance.

It is possible that the faster convergence of the parametric approach is due to the genetic nature of the algorithm, since genetic algorithms are known to outperform gradient based algorithms on specific classes of problems. In these experiments this is unlikely to be the case since *fmincon* converged in an average of 25 steps, a relatively low number. For completeness however, we also compare the parametric approach against a naive approach using a genetic algorithm: MATLAB's *ga* function with default settings. The results are depicted in Figure 7.3.

In this comparison the parametric approach drastically outperforms the naive

[n] Reprinted with permission from "A Parallel Approach for Computing the Expected Value of Gathering Information", Galvan, E., Hsiao, C., Vermillion, S., & Malak, R., 2015. SAE Int. J. Mater. Manf., 8(2):271-282, Copyright 2015 by SAE International.

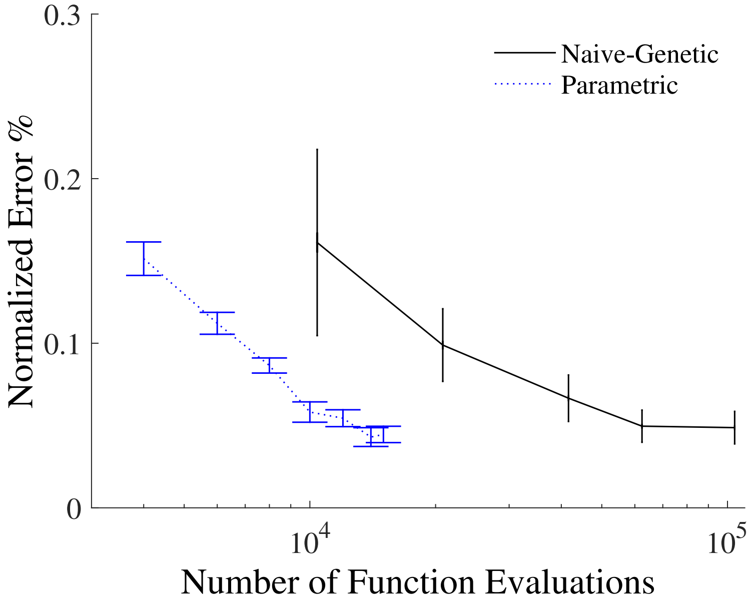


Figure 7.3: Comparison between the Parametric approach and the Naive approach using single objective genetic optimization. The parametric approach converges dramatically faster to a solution with less error. [o]

genetic approach. This result indicates that even though a genetic algorithm is not particularly well suited to solve the search problem $\max_{x \in X} f(\boldsymbol{x}, \boldsymbol{k})$. The parametric approach still shows computational advantages.

Our focus in this study is on the methods and their comparisons rather than the design results *per se*. However, it is beneficial to discuss the design results to verify the models. For this simulation, if the designer would chose not to gain information, the optimal gear ratio, $\max_{x \in X} \; E[f(\boldsymbol{x}, \boldsymbol{k})]$, is $x = (2.17, \; 1.40, \; 1.00, \; 0.78, \; 0.61)$. If the designer would choose to gather information, the optimal gear ratio would then depend on the outcome of the information gathering action.

[o] Reprinted with permission from “A Parallel Approach for Computing the Expected Value of Gathering Information”, Galvan, E., Hsiao, C., Vermillion, S., & Malak, R., 2015. SAE Int. J. Mater. Manf., 8(2):271-282, Copyright 2015 by SAE International.

7.6 Conclusions and Chapter Summary

In this chapter, we have presented an approach for reducing the computational expense in approximating the EVPI. The proposed parametric approach the EVPI problem is decomposed into two problems. In the first, we approximate the solution of the design decision for several values of the random parameter. Then we sample the solution according to the distribution of the random parameter to approximate the EVPI.

We demonstrated the performance of the parametric approach on engineering example. We consider the design of a vehicle's multi-ratio transmission with uncertain information on its engine characteristics. Aside from transmitting power from the engine to wheels, an important function of a multi-ratio transmission is to allow the engine to run at a more efficient state, thus improving fuel economy. Therefore, at a high level, we wish to design a multi-ratio transmission that maximizes engine efficiency.

The engineering problem is to compute the expected value of performing physical experiments to gather information about the uncertain information in a computational model the efficiency of an engine and transmission. In Section 3.1 we developed a method of using parametric optimization to reduce the computational expense of computing the expected value of perfect information. In this chapter, we compared the proposed approach against a "naive approach:" simple Monte Carlo.

The results from the engineering cast study indicate that, in this case, we were able to generate an accurate approximation of the EVPI significantly faster than the naive approach. The results also indicate that the parametric approach is likely beneficial even in problems where genetic algorithms tend to perform poorly. The parametric approach proposed in this study can easily be extended to apply to EVPPI problems.

For this case study we used P3GA to solve the parametric optimization problem. However, the output of P3GA is unlikely to reflect the probability distribution $P(\boldsymbol{k})$. As a first step in this research, we fit a Kriging model to the output. This Kriging model was sampled according to $P(\boldsymbol{k})$ to approximate Equation 3.5. No investigation was performed to determine whether this is the best approach.

8. CAPABILITIES REPRESENTATION CASE STUDY

8.1 Magnetohydrodynamic Active Cooling Subsystem Model

In this section, we demonstrate the use of multiobjective parametric optimization for the purpose of capability modeling. We revisit the the design of a of a structural cooling subsystem incorporating in Section 3.2.4. The structurally embedded cooling subsystem considered herein was largely inspired by the industrial research developments of [81]. The engineering models are based on the models developed in [132] by Hartl.

A fluid circuit fully filled with a liquid metal is used to transport thermal energy from a hot reservoir, into which heat is transferred at some defined power P_{hot}, to a cold reservoir, which conducts heat out to a heat sink maintained at a temperature T_{cold}. The aim is to regulate the temperature of the hot reservoir, T_{hot}. Ultimately, the best subsystem design will depend on the desired T_{hot} and the amount of heat being transferred to the hot reservoir, P_{hot}.

The longer and less wide transport channels have common length and width l_{chan} and w_{chan}, respectively. Both the channels and reservoirs are assumed to have a common depth $d_{chan} = d_{res}$. The channel aspect ratio r_{chan} is a key dimensionless design variable in this work and is simply expressed as $r_{chan} = w_{chan}/d_{chan}$.

Both transport channels pass through a DC-driven MHD pump, illustrated in Fig. 8.1. The pumps consist of two aligned permanent magnets sandwiched between two thin ferromagnetic plates. Two electrodes spanning the depth of the channel allow DC current to pass through the conductive fluid, where the highest current density exists between the two electrode plates. A critical dimension is the length of the electrodes in the direction of the channel, l_{elec}. A volumetric *Lorentz force* is

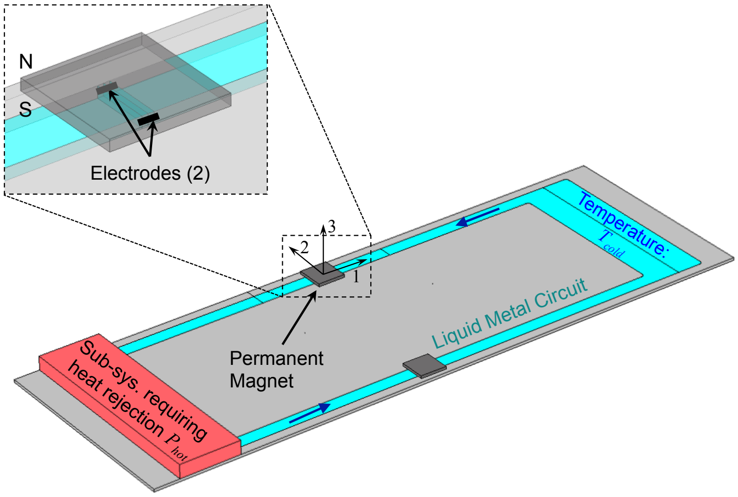


Figure 8.1: MHD-driven cooling subsystem considered in the analysis-driven design studies of the current work; inset illustrates detail of the DC MHD pump. [p]

generated by the interaction of the current and the magnetic field, driving the fluid in a direction orthogonal to both (i.e., in the direction of the channel). This effect is illustrated in Fig. 8.1b, where realistic in-plane Lorentz force rotations induced by the nonuniform nature of the current density are shown.

8.1.1 Coupled Algebraic Model

The general multiphysical mathematical model has been described in detail [132, 133]. For this cases study we consider only the reduced fidelity coupled algebraic model. In this reduced-fidelity algebraic model, spatial homogeneity in the electromagnetic fields in the region of the MHD pump (i.e., between the electrodes) is assumed. It is assumed that all current flows directly and uniformly between the two electrodes and that the magnetic field is both vertically aligned perpendicular to the current flux and is also spatially constant between the electrodes. This results in a

[p] Reprinted with permission from "Parameterized Design Optimization of a Magnetohydrodynamic Liquid Metal Active Cooling Concept", Hartl, D.J., Galvan, E., Malak, R.J., Baur, J.W., 2016 J. Mech. Des. 138(3) Copyright 2015 by ASME.

constant and unidirectional Lorentz force of

$$f_L = \frac{-B\, I_{app}}{A_{elec}} \tag{8.1}$$

between the electrodes while $f_L = 0$ elsewhere. Here B is the local magnetic flux aligned in the out-of-plane direction and I_{app} is the total current traveling between two electrodes. Each electrode is assumed to have a square face exposed to the channel with the area $A_{elec} = d_{chan} l_{elec}$. The local magnetic flux vector B outside of a rectangular permanent magnet and in its poled direction can be approximated knowing the unidirectional remnant magnetic flux B_r and the dimensions of the magnet [134, 135]

$$\begin{aligned} B \;&= \frac{2\, B_r}{\pi} \left[\arctan\left(\frac{l_{mag} w_{mag}}{2x\sqrt{4x^2 + l_{mag}^2 + w_{mag}^2}} \right) \right. \\ &\left. - \arctan\left(\frac{l_{mag} w_{mag}}{2(d_{mag} + x)\sqrt{4(d_{mag} + x)^2 + l_{mag}^2 + w_{mag}^2}} \right) \right], \end{aligned} \tag{8.2}$$

where the combined effects of the two aligned magnets (see Fig. 8.1) is assumed to be additive in the region between them. Here x is one half the distance between the two magnet faces, which we take to be one half the channel depth such that $x = d_{chan}/2$.

It has been shown that the Lorentz volumetric force of Equation (8.1), if uniform and aligned with the direction of axial flow in a fluid channel, will act to drive the fluid as if it were a localized pressure gradient [136]. Such a flow field can be idealized as steady, unidirectional, and fully developed. Given the low velocities considered herein, it can also be assumed strictly laminar.

To model the relationship between the Lorentz force, channel configuration, pressures, and mass flow rate, the rectangular liquid metal fluid circuit is divided into six regions of constant cross-section. Assuming linear pressure variations, constant or null Lorentz forces, the absence of transition effects between regions, and a known reference pressure p_{ref} at some point allows us to write the following system of algebraic equations:

$$Q = \left(-\frac{p_{i+1} - p_i}{L_i} + f_L^i \right) g(w_i, d_i, \mu) \; \forall \; i = 1 \ldots 6, \tag{8.3}$$

where

$$p_1 = p_7 = p_{ref}, \qquad f_L^i = 0 \;\; \forall \;\; i \neq 1.$$

This yields six linear equations (8.3) for the following six unknowns:

$$\{p_2, p_3, p_4, p_5, p_6, Q\}.$$

The function $g(w, d, \mu)$ captures the geometry of a channel, in this case rectangular with width w and depth d and is given in [137] as

$$g(w, d, \mu) = \frac{wd^3}{12\mu} \left[1 - 6\frac{d}{w} \sum_{n=1}^{\infty} \frac{1}{a_n^5} \tanh(a_n \frac{w}{d}) \right], \tag{8.4}$$

where μ is the dynamic viscosity, $a_n = (n - \frac{1}{2})\pi$ and the Fourier series converges by $n \sim 5$. Note that $f_L = 0$ in all regions except in between the parallel electrodes (i.e., Region 1). The fluid circuit model is graphically described in Fig. 8.2.

For the modeling of cooling, all conductivity except through the two reservoir walls is neglected and only advective heat transfer is considered. Assuming perfect mixing in the reservoirs, it is straightforward to consider a simple lumped thermal model to describe the couplings between power, temperature, and fluid flow in the system. Noting that the volumetric flow rate Q is identical for all sections and that

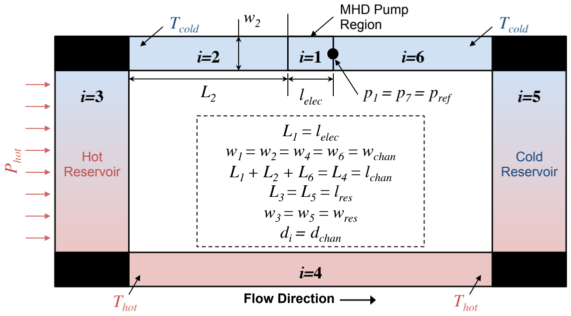


Figure 8.2: Graphical description of the reduced-fidelity fluid-thermal engineering model considered herein. [q]

all power into the hot reservoir is assumed to be moved to the cold reservoir, we have simply

$$T_{hot} = P_{hot}/(\rho Q c_p) + T_{cold} \tag{8.5}$$

where ρ and c_p are the density and heat capacity, respectively, of the working fluid and T_{cold} and P_{hot} are application-defined inputs. The hot reservoir temperature T_{hot} represents one of the most important response variables associated with the considered system. Throughout this work, we will consider only $T_{cold} = 300\,\mathrm{K}$, though this environmental variable represents the kind of uncontrolled input intended for consideration as a parameter.

Finally, other key design/response relationships should be defined. The applied current I_{app} that generates the driving Lorentz force via (8.1) can be approximated in terms of the applied voltage V_{app} via

$$I_{app} = V_{app}/R_{MHD} = V_{app}\,\sigma\, l_{elec}/r_{chan}, \tag{8.6}$$

[q] Reprinted with permission from “Parameterized Design Optimization of a Magnetohydrodynamic Liquid Metal Active Cooling Concept”, Hartl, D.J., Galvan, E., Malak, R.J., Baur, J.W., 2016 J. Mech. Des. 138(3) Copyright 2015 by ASME.

where R_{MHD} is the resistance across the electrodes expressed in terms of the electrical conductivity of the working fluid σ and geometric parameters. The electrical power P_{MHD} needed to drive the MHD pump is then

$$P_{MHD} = I_{app}\, V_{app} = V_{app}^2\, \sigma\, l_{elec}/r_{chan}. \tag{8.7}$$

The efficiency of the MHD-driven cooling subsystem is quantified via the MHD power-specific effective thermal conductivity $\hat{k}$, defined here as

$$\hat{k} = P_{hot}/(P_{MHD}\, \Delta T), \tag{8.8}$$

where ΔT is simply the difference in hot and cold reservoir temperatures ($\Delta T = T_{hot} - T_{cold}$). Assuming the reservoirs to be of fixed configuration (i.e., determined by external agents), the mass of the designed system is expressed only in terms of the channel mass m_{chan}, given as

$$m_{chan} = 2\rho(l_{chan}\, d_{chan}\, w_{chan}) = 2\rho(l_{chan}\, r_{chan}\, d_{chan}^2). \tag{8.9}$$

8.1.2 Model Calibration

For this case study, a number of potential design variables are held fixed to simplify the problem. The geometric design parameters assigned fixed values throughout this work are listed in Table 8.1. Note that, as per Fig. 8.2, the reservoir depth and channel depth are taken to be equal. The magnet and working fluid (i.e., liquid metal) materials are also taken to be pre-defined and their problem-essential properties known. The appropriate values are given in Table 8.2.

Table 8.1: Fixed geometric design parameters used for all MHD-driven cooling subsystem analyses.

Fixed Parameter	Value	Fixed Parameter	Value
l_{mag}	10 mm	l_{res}	70 mm
w_{mag}	10 mm	w_{res}	20 mm
d_{mag}	1.0 mm	l_{elec}	2.0 mm
		l_{chan}	200 mm

Table 8.2: Relevant material properties used for MHD-driven thermal transfer analysis case.

Property	Value	Property	Value
(Galinstan)			
ρ	6440 kg/m^3	μ	2.40E-3 Pa-s
c_p	300 J/kg/K	σ	3.57E6 S/m
(Magnet)			
B_r	1.2 T		

8.2 Multiobjective Parameterized Design Optimization Problem

We presume the designer of the LM-MDH active cooling subsystem has control over three design variables: fluid channel depth, d_{chan}), fluid channel aspect ratio, r_{chan}, and the voltage applied across the MHD electrodes, V_{app}. The vector of design variables is

$$\boldsymbol{x} = (d_{chan}, r_{chan}, V_{app}) \tag{8.10}$$

We distinguish between x, which describes the geometry of thermal transport subsystem and $\boldsymbol{x}$, the vector of design variables. The model parameters that are outside of the control of the subsystem designer are the power provided by the heat source

P_{hot}, and the goal temperature of the hot reservoir T_{hot}. The vector of parameters is

$$\boldsymbol{\theta} = (P_{hot}, T_{hot}) \tag{8.11}$$

In this subsystem, P_{hot} is in input to the cooling subsystem to be determined by an external agent (e.g., the systems engineer). On the other hand, T_{hot} is a response of the subsystem model; that is, different designs result in different values of T_{hot}. The design objectives are to minimize the channel mass, $m_{chan}(\boldsymbol{x})$, and maximize the effective thermal conductivity, $\hat{k}(\boldsymbol{x}, \boldsymbol{\theta})$. The aim of the design problem is to find optimal (multiobjective) design for the cooling subsystem as a function of the multiple parameters.

The design of the cooling system must satisfy two constraints: the current applied to the MHD electrodes must be below an allowable maximum ($I_{app}(\boldsymbol{x}) \leq I_{app}^{max}$), and the hot reservoir temperature, while defined as a parameter to be explored, must remain below an allowable maximum ($T_{hot}(\boldsymbol{x}, P_{hot}) \leq T_{hot}^{max}$).

Given the above description and the relationships of Section 8.1.1, the parameterized optimization problem for capability modeling of the cooling system is

$$\begin{aligned} y^*(T_{hot}, P_{hot}) = \min_{\boldsymbol{x}} \quad & \big(m_{chan}(\boldsymbol{x}), \quad -\hat{k}(\boldsymbol{x}, T_{hot}, P_{hot})\big) \\ \text{subject to} \quad & I_{app}(\boldsymbol{x}) \leq I_{app}^{U} \\ & T_{hot}(\boldsymbol{x}, P_{hot}) \leq T_{hot}^{U} \\ & P_{hot}^{L} \leq P_{hot} \leq P_{hot}^{U} \\ & x_i^L \leq x_i \leq \ x_i^U \ \ i = 1, \ldots, 4 \end{aligned} \tag{8.12}$$

where again $\boldsymbol{x} = (d_{chan}, r_{chan}, V_{app})$, and the parameter attributes are $\boldsymbol{\theta} = (P_{hot}, T_{hot})$. The superscripts L and U denote the upper and lower bounds of the variables, respectively. The particular limits for the optimization cases described are:

$$\boldsymbol{x}^L = (0.1\,\mathrm{mm}, 1, 0.001\,\mathrm{V}), \boldsymbol{x}^U = (1.0\,\mathrm{mm}, 10, 0.01\,\mathrm{V})$$

$$P_{hot}^L = 1.0\,\mathrm{W}, \quad P_{hot}^U = 10.0\,\mathrm{W}, \quad I_{app}^U = 10\,\mathrm{A}, \quad T_{hot}^U = 498\,\mathrm{K}.$$

8.3 Building a Continuous Model of Subsystem Capabilities

While the problem in Equation 8.12 is to approximate the set of all solutions to a multiobjective parametric optimization (MPO) problem, the purpose of doing so is to characterize the capabilities of the cooling subsystem. The solution to the MPO problem is a set of discrete points that lie on the global parametric Pareto frontier. Here in, we use the term parametric Pareto frontier (PPF) to refer to the approximation of the global PPF.

The PPF could potentially be used in a number of different ways to aid system design. For example, the PPF could be used in "design by shopping" [138] where the decision maker forms his or her preferences by viewing a set of "good" solutions. Because the PPF is composed of parametrically Pareto efficient solutions, it serves as a minimal notion of goodness.

Another possibility is to use the PPF to enable real-time design negotiation. One can envision a scenario where the PPF is used a low order approximation of what alternatives are possible. The PPF could be used in design meeting between system and subsystems engineers to quickly evaluate the feasibility or quality of design alternatives.

Investigating the different use cases of the capability model is beyond the scope of this dissertation. In this section we explore one possibility: visualizing the capability models to improve the requirements. The discrete points that comprise the capability model are analogous to the Pareto frontier data generated when solving a multiobjective (MO) problem. In the MO case, it is often desirable to generalize beyond the discrete Pareto frontier data and create a continuous approximation of the frontier [55, 56, 139]; that is, one is motivated to find solutions not represented in the discrete set. Such generalization is likewise desirable in the parametric case; presently, to improve visualization.

In this work, we generalize the data using Ordinary Kriging interpolation [130] to fit a model of the parameterized Pareto frontier in the attribute space, i.e., the combined space of design objectives and parameter attributes. Other fitting approaches are possible, but care must be taken to ensure the fitted model captures key features of the data. Linear regression models, which presume the data contains noise that should be averaged out, tend to perform poorly on frontier data [139]. Thus, the fitting model must be of appropriate order and mathematical form or should be an interpolating relationship as is the case with Kriging. The details of Ordinary Kriging are discussed in Section . In this section we again use the DACEfit toolbox Kriging model [131] to interpolate between the data. We use the linear correlation function and a second order polynomial regression function. Let $a^k = (y_1^k, \ldots y_m^k, \boldsymbol{\theta}^k)$ for $k = 1, \ldots, N$ be vectors on the parameterized Pareto frontier, where $y_1, \ldots, y_m$ denotes objective attribute values and $\boldsymbol{\theta} \in \mathbb{R}^p$ denotes the vector of p parameter values. We fit the Kriging model such that

$$\hat{y_j^k} = f_{Krig}(y_1^k, \ldots, y_{j-1}^k, y_{j+1}^k, \ldots, y_m^k, \boldsymbol{\theta}^k) \tag{8.13}$$

where the choice of j is arbitrary. Given a specific instance of the parameter vector $\boldsymbol{\theta}$, one can use the Kriging model as a constraining relationship that implicitly defines a Pareto frontier. Several variations on this technique are possible. See [139] for a detailed description and comparison of methods. Model accuracy should be assessed via cross-validation [140], residual error analysis [141], mean absolute squared prediction Error [142], R^2, or similar. In this article, we assess the accuracy of the model using leave one out cross-validation with error measured as mean absolute percent deviation

$$LOOCV = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{y}_j^k - y_j^k}{y_j^k}\right| \tag{8.14}$$

where $\hat{y}_j^k$ is the response of the Kriging model fit with all N data points except p.

8.4 Experimental Setup

For this case study, we solved Equation 8.12 using P3GA and p-NSGAII. In the case of P3GA, the following parameters were used

Generations:	$G =$	100
Population Size:	$P =$	100
Crossover Rate:		0.08
Mutation Rate:		0.01
Kernel function parameter:	$q =$	10
Max Num. of Training Data:		∞

The total number of function evaluations for the P3GA is $PG = 10,000$. Recall that the p-NSGAII requires that the user specify the number of discretizations, p_d along each parameter attribute. This parameter may be critical to the performance of the algorithm in terms of solution quality and computation (as measured by number of function evaluations). Since selecting the appropriate p_d value is critical to the performance of p-NSGAII, we performed this case study with two different parameter settings. The following parameters were used in the initial case study, here in referred to as p-NSGAII-A

Generations:	$G =$	100
Population Size:	$P =$	100
Crossover Rate:		0.08
Mutation Rate:		0.01
Discretizations :	$p_d =$	4

Because Equation 8.12 has $p = 2$ parameters, the total number of function evaluations the p-NSGAII-A case is $GPp_d^2 = 160,000$ (considerably more than for P3GA). A second attempt was made with increased number of discretizations

Generations:	$G =$	100
Population Size:	$P =$	20
Crossover Rate:		0.08
Mutation Rate:		0.01
Discretizations :	$p_d =$	10

The total number of function evaluations for the p-NSGAII-B approach is $GPp_d^2 = 200,000$. Note that the population size was decreased in this case to maintain a similar order of magnitude in the total number of function evaluations. Another consideration is that increasing the number of discretizations decreases the solution space for each p-NSGAII subproblem. As a result, we can reasonably expect that a smaller population is needed to represent the search space.

Unlike test problems A-D, the true solution to Equation 8.12 is unknown. As a result, the performance assessment techniques presented in Chapter 4 cannot be used to evaluate the solution quality. Furthermore, since the data is 4 dimensional (2 objectives and 2 parameters), visualizing the data is also challenging.

Table 8.3: Kiging Model *LOOCV* values corresponding to each algorithm. The low *LOCCV* values give confidence in the predictive performance of the model.

Algorithm	*LOOCV* [%]
p-NSGAII-A	0.033
p-NSGAII-B	0.247
P3GA	0.865

8.5 Results and Discussion

The results from a single run of each case, P3GA, p-NSGAII-A, and p-NSGAII-B, are illustrated in Figures 8.3 to 8.5. The matrix of subplots contains scatter plots of the data. The axes along the diagonal are a histogram of the data corresponding each dimension of the data. The P3GA appears to generate a more even (less clustered and more wide spread) set of points that p-NSGAII. While P3GA is able to find points evenly throughout the search space, the p-NSGAII approach clusters the solutions around the ε constraint values. As can be seen in comparing Figures 8.4 and 8.5, increasing the number of discretizations along the parameter increases the number of clusters.

A Kriging model was fit to each data set; the *LOOCV* percentages are reported in Table 8.3. The cross-validation error values in each model are less that 1 percent. The low error gives us confidence in the predictive performance of the Kriging models.

Figure 8.6 is a plot of the Kriging models fit to the P3GA and p-NSGAII-B data sets. To allow visualization of the 4 dimensional plot in 2 dimensions, we took “slices” along the parameter dimensions. The “slices” are taken along the parameter values. For example, the bottom column in Figure 8.6 corresponds to the parameter value $P_{hot} = 10$.

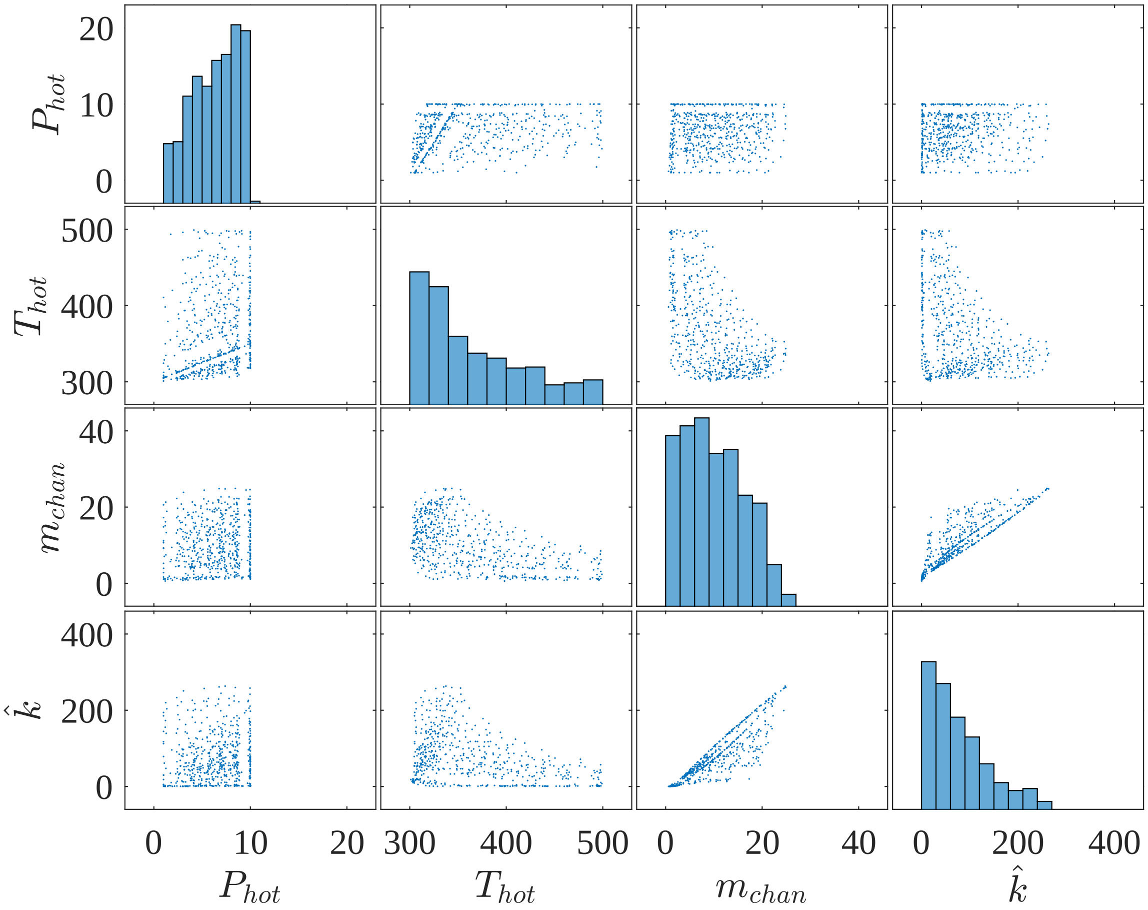


Figure 8.3: Scatter plot matrix of the parametric Pareto frontier data generated using P3GA for the LM-MHD cooling subsystem. The data appears fairly evenly spread throughout the space.

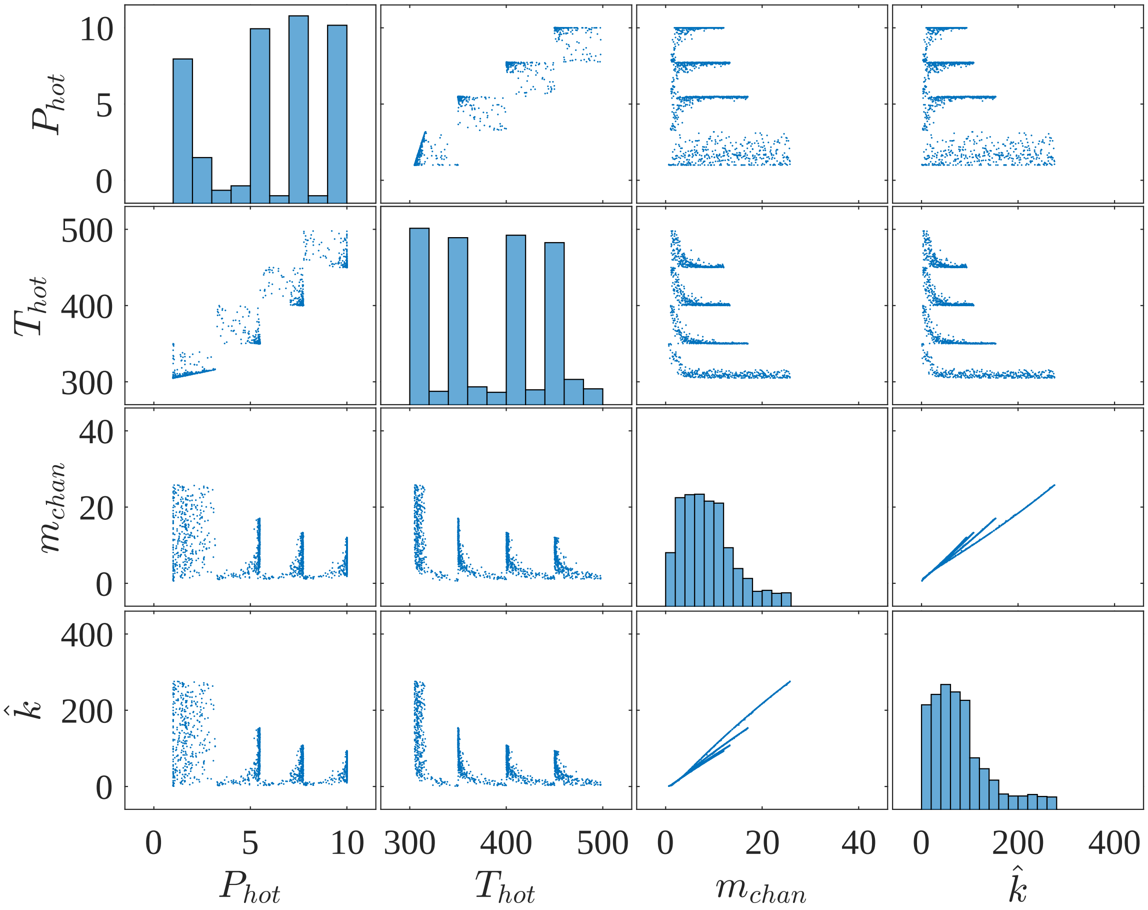


Figure 8.4: Scatter plot matrix of the parametric Pareto frontier data generated using p-NSGAII-A for the LM-MHD cooling subsystem. The data appears to cluster at the ε constraint values.

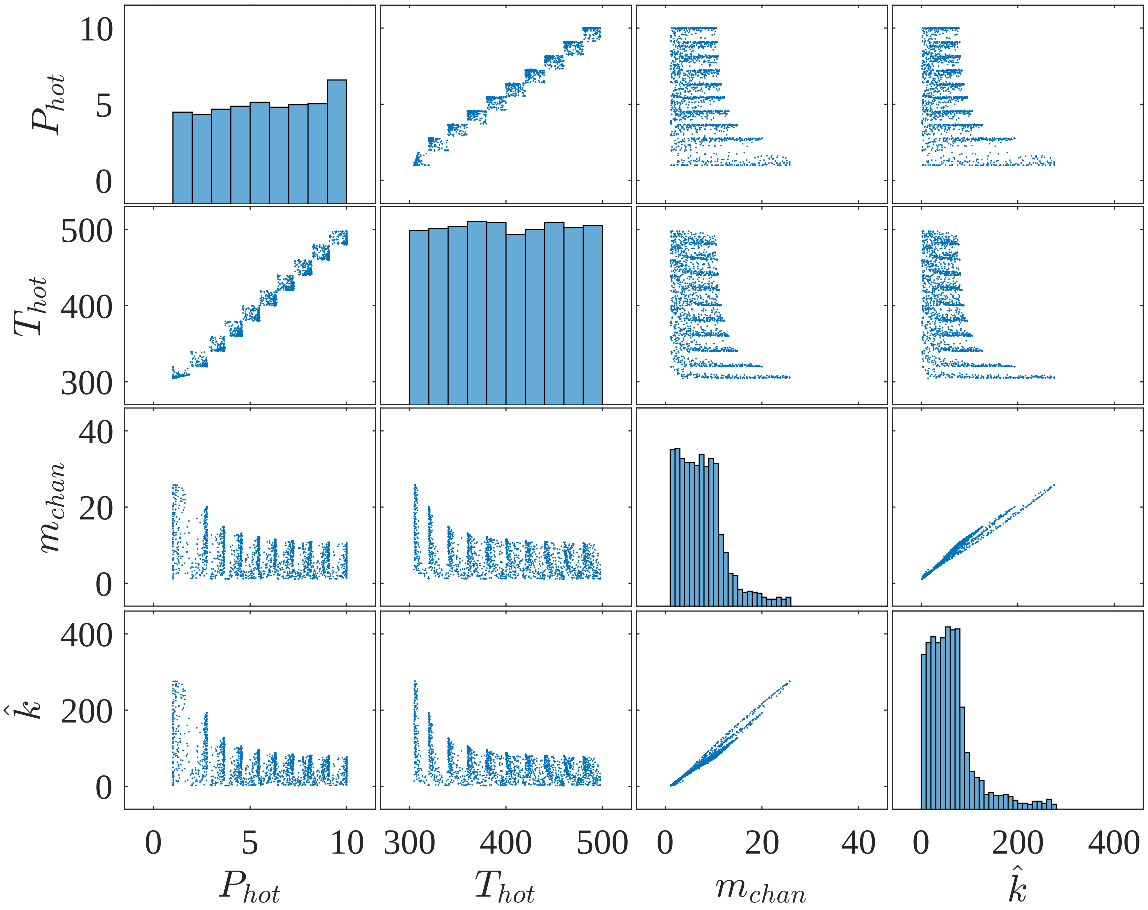


Figure 8.5: Scatter plot matrix of the parametric Pareto frontier data generated using p-NSGAII-B for the LM-MHD cooling subsystem. The data appears to cluster at the ε constraint values.

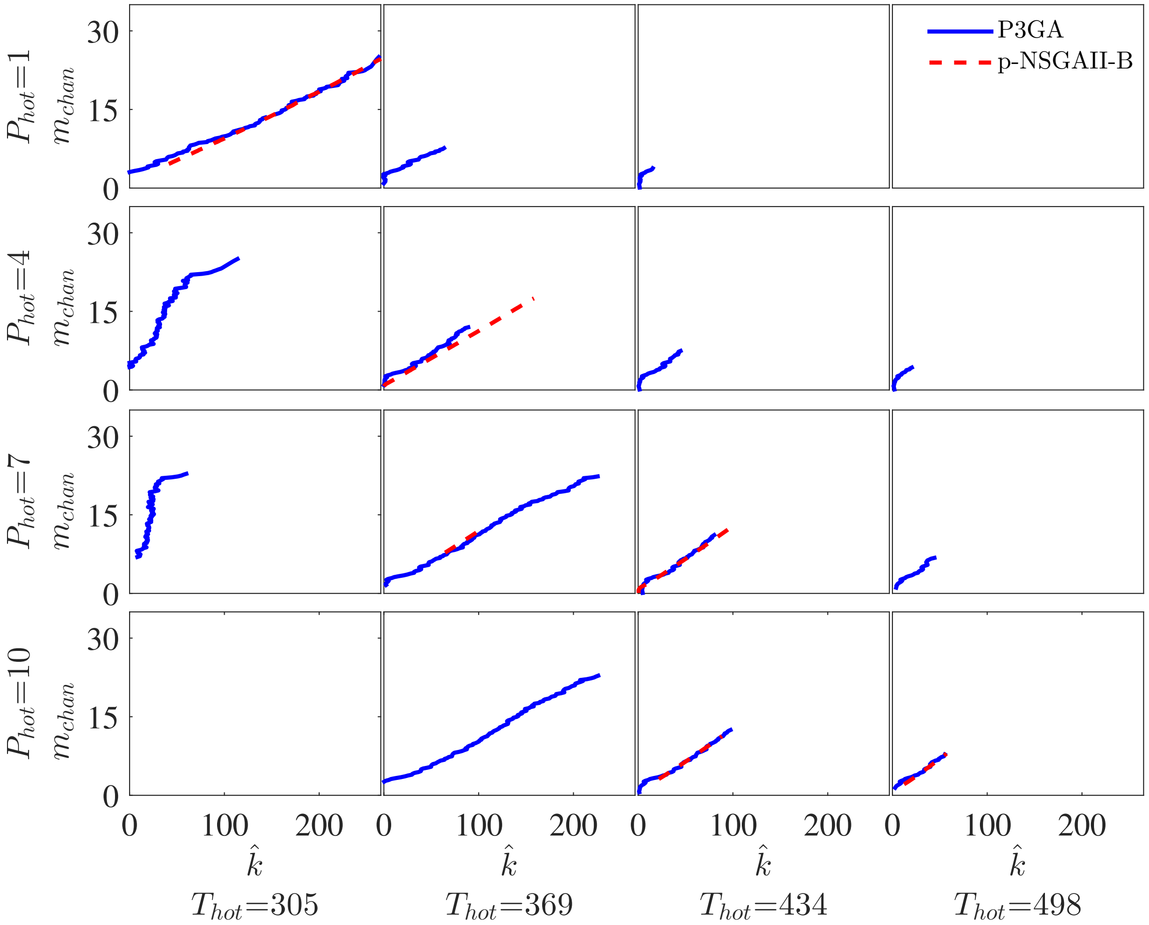


Figure 8.6: Plot matrix of "slices" of the Kriging models fit to the PPF data generated by P3GA and p-NSGAII. The Kriging model fit to the P3GA PPF data appears to predict a wider range of solutions.

8.6 Conclusions and Chapter Summary

In this chapter we have demonstrated the use of parametric optimization in the process of modeling the capabilities of a subsystem. The use of parametric optimization allows designers to build knowledge about the capabilities and limitations of a subsystem without requiring knowledge about the system-level considerations. The parametric problem is to find the optimum of a function (with one or several objectives) as a function of some parameters. These parameters may be unknown (at least to the subsystem designer) system level specifics or interface parameters (e.g., shared variables) between different subsystems. Using the parametric techniques presented in this dissertation, the solution to the parametric optimization problem is a discrete set of solutions that lie on the parametric Pareto frontier (PPF).

The motivating problem in this case study is the design of a novel Magnetohydrodynamic active cooling subsystem. A fluid circuit fully filled with a liquid metal is used to transport thermal energy from a hot reservoir, into which heat is transferred at some defined power P_{hot}, to a cold reservoir, which conducts heat out to a heat sink maintained at a temperature T_{cold}. The aim is to regulate the temperature of the hot reservoir, T_{hot}. However, P_{hot} and T_{hot} are not yet known to the designer. The parametric problem is to find the designs, $\boldsymbol{x}$, that simultaneously minimize channel mass $m_{chan}(\boldsymbol{x})$ and maximizes effective thermal conductivity $\hat{x}(\boldsymbol{x}, T_{hot}, P_{hot})$ as a function of the parameters.

The parametric problem is solved using P3GA and p-NSGAII. For this case study, P3GA appears to generate a more even (less clustered and more wide spread) set of points that p-NSGAII. The p-NSGAII approach clusters the solutions around the ε constraint values. Increasing the number of discretizations (ε constraint values) increases the number of clusters. A Kriging model was fit to the data generated using either approach. Using LOOCV, the Kriging model was determined to be a “good” fit, less than a mean of 1% prediction error for each model. To allow visualization of the 4 dimensional Kriging models (2 objectives and 2 parameters), we took 2 dimensional “slices” of the model at evenly spaced parameter values. The Kriging model fit to the P3GA PPF data appears to predict a wider range of solutions.

9. SUMMARY OF CONTRIBUTIONS AND LIMITATIONS

9.1 Review of Research

We define parametric optimization as the process of finding the optimal solution as a function of one or more parameters. From a decision perspective the difference between parametric optimization the single and multi-objective is the available preference information. In SO, all preference information is available and each alternative can be ranked. In MO, some preference information is missing, specifically how to trade-off between some attributes relevant to decision making (multiple conflicting objectives. Finally, PO can be thought of as the most general case where no preference ordering is available for at least one attribute.

The general notion of parametric optimization, as we have described it, has been used in several fields, including economics and more recently model based controls engineering. The aim of this research is *to introduce the concept of parametric optimization as a useful approach for systems design challenges.* Specifically, the use of parametric optimization in systems design is motivated by (1) the problem of computing the expected value of information (EVI), and (2) capability modeling.

The design of engineered systems requires careful management of uncertainty. A class of alternatives that is often overlooked, is information activities to reduce the uncertainty, e.g., prototype development, increasing the sample size of an experiment, conducting more expensive tests, etc. These information gathering alternatives can be performed at a cost. The first step in deciding whether or not to perform an information gathering action is to compute its value. The Expected Value of Information (EVI) framework provides a rigorous decision theoretic approach for such "information decisions." The EVI is value one is willing to pay to gain access to some information. A practical challenge in EVI is the computational expense in approximating its value. The approximation of EVI can involve many nested optimizations, which may grow exponentially with the amount of information to be gathered, such that the cost of computing EVI may even exceed the cost gathering information. Consequently, techniques for reducing the computational burden in EVI are of prime concern. In this research, we proposed to model the EVI problem using parametric optimization. The value of recasting the problem EVI problem this way is a potential reduction in computational complexity.

The second motivating application is capability modeling. Engineering projects typically involve many individuals, each of who contributes knowledge to the project and none of whom has complete knowledge about the system under development. Although this specialization of knowledge is necessary and even desirable from certain perspectives, it can create challenges for system-level decision making. The systems engineers must make decisions such as to define the system architecture and to allocate resources or requirements despite not being an expert in each relevant discipline. Thus, the accurate communication of technical capabilities from discipline engineers to systems engineers can be an important factor in the success of a systems engineering project. The attributes subsystem attributes that are relevant for decision making are the *objective attributes* (those with an application independent defined preference relation, e.g., minimize cost, mass, power consumed etc.) and *parameter attributes* (those where the preference relation depends on the application, e.g., in a suspension subsystem, the best value of spring constant depends on the application). The efficient set of alternatives (the subset of alternatives that could possibly be most preferred) is the parametric Pareto frontier (PPF). We demonstrate how the solution to the MPO formulation is the PPF.

Existing techniques for parametric optimization are exact but limited to problems with specific mathematical structures: linear, quadratic, convex nonlinear. This is a significant limitation for engineering design challenges where the models are often highly nonlinear (dynamical systems) or non-analytical (finite element analysis, simulations involving conditionals). If parametric optimization is to be used in general systems design problems, there is a need for optimization techniques that can handle a wide range of mathematical forms. As a first step towards this end, we investigate the development of heuristic algorithms for parametric optimization. The solution generated by a heuristic algorithm is an approximation of the true solution. An appropriate performance evaluation strategy is needed.

We developed empirical approach for quantitative evaluation and comparison of algorithms for parametric optimization. Key challenges in the development of an empirical approach for performance assessment are the **(1)** development of suitable test problems and **(2)** performance metrics. We developed test problems for which we can arbitrarily scale the number of objectives and parameters and even the difficulty of the problems. The problems were carefully developed to exhibit features that are likely to cause difficulty to the optimizer and are common to many engineering problems, e.g., non-convexity, discontinuity, multi-modality, etc. Towards **(3)**, the selection of a suitable performance metric, we consider unary indicators. From an analysis perspective, unary indicators are desirable since it allows for the statistical analysis over performance. We suggest the use of mean Hausdorff distance for performance assessment of approximation sets involving parameters. The mean Hausdorff distance has desirable properties in that it can indicate *that* one approximation is better than another and it can reflect solution “spread,” even in the case with parameters.

For the algorithm development, we considered the adaptation of multiobjective techniques to the parametric case. We proved that the method alternating preference directions cannot, in general, find all solutions to an MPO problem. Taking an iterative approach to MPO, we developed an extension of the Nondominated Sorting Genetic Algorithm II (NSGAII) for the parametric case, which we call parametric NSGAII (p-NSGAII). In this iterative approach to MPO, the parameters are constrained to ranges rather than fixed values of the parameters. As a result, the algorithm is able to find solutions at arbitrary values of the parameter. To improve algorithm performance, each application of NSGAII is seeded with the results from the previous application. This allows each new application to "learn" from its predecessor. However, because of the sequential nature of p-NSGAII, this knowledge sharing is limited to neighbors and is unidirectional.

The predictive parametric Pareto genetic algorithm (P3GA) resolves this limitation by replacing the notion of dominance in the general genetic MO strategy with parametric Pareto dominance. Because it is unlikely that randomly generated alternatives will be parametrically dominated, P3GA instead relies on the concept of predicted dominance. By eliminating iteration, In P3GA, knowledge about one region of the search space can be communicated to any other solution, improving algorithm performance.

To assess the performance if the proposed algorithms, p-NSGAII and P3GA were compared on the proposed test problems and performance was measured in terms of solution quality (measured using mean Hausdorff distance), the number of function evaluations (NFE), and wallclock time. The results indicate that P3GA generally performs better than p-NSGAII (in terms of Δ_p) as the number of parameters and/or objectives increases. For low dimensional cases without parameters (i.e., traditional multiobjective problems) the p-NSGAII method tended to outperform P3GA. This is expected since without parameters, p-NSGAII reduces to the original NSGAII, which is developed to solve multiobjective problems. For those problems that included parameters, P3GA tended to outperform.

We demonstrated the performance of the parametric approach on engineering example. We consider the design of a vehicle's multi-ratio transmission with uncertain information on its engine characteristics. Aside from transmitting power from the engine to wheels, an important function of a multi-ratio transmission is to allow the engine to run at a more efficient state, thus improving fuel economy. Therefore, at a high level, we wish to design a multi-ratio transmission that maximizes engine efficiency.

The engineering problem is to compute the expected value of performing physical experiments to gather information about the uncertain information in a computational model the efficiency of an engine and transmission. We compared the proposed parametric approach against a "naive approach:" simple Monte Carlo.

The results from the engineering cast study indicate that, in this case, we were able to generate an accurate approximation of the EVPI significantly faster than the naive approach. The results also indicate that the parametric approach is likely beneficial even in problems where genetic algorithms tend to perform poorly. The parametric approach proposed in this study can easily be extended to apply to EVPPI problems.

Finally, we demonstrated how the use of MPO capability modeling using a real world engineering example: a Liquid Metal Magnetohydrodynamic Pump (LM-MHD). We demonstrated how the LM-MHD subsystem cannot be optimized independent of the subsystem into which it will be incorporated, since the optimal LM-MHD depends on the properties of the system (these properties are parameter attributes). The MPO formulation allows the designer to optimize the system, while remaining agnostic about the overall system. This can lead to a better understanding of the underlying physics in the model and a model of the capabilities of the

LM-MHD.

The LM-MHD parametric problem is solved using P3GA and p-NSGAII. For this case study, P3GA appears to generate a more even (less clustered and more wide spread) set of points that p-NSGAII. The p-NSGAII approach clusters the solutions around the ε constraint values. For visualization, a Kriging model was fit to the data generated using either approach. Using LOOCV, the Kriging model was determined to be a "good" fit, less than a mean of 1% prediction error for each model. Visually, the Kriging model fit to the P3GA data appear to predict a wider range of solutions.

9.2 Limitations and Future Work

A limitation of this research is in the breadth of engineering problems investigated. Although we attempted to draw from a variety of engineering disciplines to investigate the performance of the possessed approaches, the investigation is practically limited. Whether or not parametric optimization is truly useful in systems engineering, future work should include repeated real-world applications in a wide range of disciplines.

Existing techniques for parametric optimization are limited to problems with specific mathematical structures (linear, quadratic, convex nonlinear, etc). However, many models in engineering design are considerably more complex, e.g, dynamical systems, Finite Element Analysis, process simulations, etc. As a result, we were motivated to develop a general purpose heuristic algorithms for parametric optimization since (1) they are easy to implement, requiring little knowledge about the problems mathematical structure, and (2) can be applied to a wide range of problems. A drawback however, is they are limited in their performance generally finding only an approximation of the true optimal.

In approximating the solution rather than finding it exactly, a number of new problems arise with were not rigorously addressed in this dissertation: (1) how to best use the approximation and (2) what is the penalty we pay for using an approximation? Future work may include (i) a development of exact solution techniques for the MPO case, (ii) extensions of exact PO techniques to more general mathematical forms, and (iii) an investigation into how to best use approximation sets. Such an investigation would likely be dependent on the application. For example, the case of approximating EVI, it may be desirable to develop an approximation such that the distribution along the parameter attribute reflects the distribution of interest. This approach, would not be useful for capability modeling.

In terms of capability modeling, we discussed at length its development but did not focus much on its use. Clearly, an accurate unambiguous (having well-defined semantics) model of the capabilities of a subsystem is desirable. Such a model can inform the mental model of designers, improving decision making throughout the design process. However, we do not investigate the possibility of directly making design decisions form the capability model. For example, once the system-level considerations have been determined (establishing a preference order over the parameter attributes), can the capability model be used to "retrieve" the optimal design? In our discussion of parametric optimization, we concerned ourselves with the approximation of $y^*(\theta)$, the optimal objective values as a function of the parameters. To retrieve the optimal design from a capability model, we must have $x^*(\theta)$, the optimal design variables as a function of the parameters. Future work may include techniques for finding $x^*(\theta)$ for engineering design problems.

Another possible use of capability models is at the system-level. One can imagine a scenario where the systems engineer has available capability models for every relevant subsystem. How can the systems engineer search these combined models for the optimal system architecture, design requirements, etc.

REFERENCES

[1] G. A. Hazelrigg, *Fundamentals of Decision Making for Engineering Design and Systems Engineering*. Upper Saddle River, NJ: Pearson Education, 2012.

[2] A. Sage and J. Armstrong Jr., *Introduction to Systems Engineering*. Hoboken, NJ: Wiley and Sons, 2000.

[3] D. M. Buede, *The Engineering Design of Systems*. New York, NY: John Wiley & Sons, 2000.

[4] R. J. Malak and C. J. Paredis, “Using parameterized pareto sets to model design concepts,” *Journal of Mechanical Design*, vol. 132, no. 4, p. 041007, 2010.

[5] N. Saunders and A. P. Miodownik, *CALPHAD (Calculation of Phase Diagrams): A Comprehensive Guide: A Comprehensive Guide*, vol. 1. New York, NY: Elsevier, 1998.

[6] P. Milgrom and I. Segal, “Envelope theorems for arbitrary choice sets,” *Econometrica*, vol. 70, no. 2, pp. 583–601, 2002.

[7] C. Carlsson and P. Korhonen, “A parametric approach to fuzzy linear programming,” *Fuzzy Sets and Systems*, vol. 20, no. 1, pp. 17–30, 1986.

[8] P. Milgrom and C. Shannon, “Monotone comparative statics,” *Econometrica: Journal of the Econometric Society*, pp. 157–180, 1994.

[9] H.-J. Zimmermann, “Fuzzy programming and linear programming with several objective functions,” *Fuzzy sets and systems*, vol. 1, no. 1, pp. 45–55, 1978.

[10] S. Chanas, “The use of parametric programming in fuzzy linear programming,” *Fuzzy Sets and Systems*, vol. 11, no. 1, pp. 229–241, 1983.

[11] T. C. Wagner and P. Y. Papalambros, "Selection families of optimal engine designs using nonlinear programming and parametric sensitivity analysis," tech. rep., SAE International, 1997.

[12] J. Sobieszczanski-Sobieski, J. S. Agte, and R. R. Sandusky, "Bilevel integrated system synthesis," *AIAA Journal*, vol. 38, no. 1, pp. 164–172, 2000.

[13] H. T. Jongen and G.-W. Weber, "On parametric nonlinear programming," *Annals of Operations Research*, vol. 27, no. 1, pp. 253–283, 1990.

[14] T. Gal and H. J. Greenberg, *Advances in sensitivity analysis and parametric programming*, vol. 6. New York, NY: Springer Science & Business Media, 2012.

[15] E. Pistikopoulos, A. Galindo, V. Dua, E. S. Kikkinides, L. Papageorgiou, W. Jorisch, K.-W. Benz, W. Neumann, M. Köhler, W. Fritzsche, *et al.*, *Multi-Parametric Model-Based Control: Theory and Applications, Volume 2.* New York: Wiley, 2007.

[16] J. Guddat, F. G. Vazquez, and H. T. Jongen, *Parametric optimization: singularities, pathfollowing and jumps.* Wiesbaden, Germany: Springer, 1990.

[17] A. B. Berkelaar, K. Roos, and T. Terlaky, "The optimal set and optimal partition approach to linear and quadratic programming," in *Advances in Sensitivity Analysis and Parametic Programming*, pp. 159–202, Springer, 1997.

[18] A. V. Fiacco, "Sensitivity analysis for nonlinear programming using penalty methods," *Mathematical Programming*, vol. 10, no. 1, pp. 287–311, 1976.

[19] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms.* Chichester: Wiley & Sons, 2001.

[20] R. Cirillo, *The Economics of Vilfredo Pareto.* New York, NY: Frank Cass and Company Limited, 1979.

[21] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives*. Cambridge, UK: Cambridge University Press, 2 ed., 1993.

[22] Y. Y. Haimes, L. Ladson, and D. A. Wismer, "Bicriterion formulation of problems of integrated system identification and system optimization," *IEEE Transactions on Systems Man and Cybernetics*, no. 3, p. 296, 1971.

[23] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[24] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[25] N. Drechsler, R. Drechsler, and B. Becker, "Multi-objective optimisation based on relation favour," in *Evolutionary Multi-Criterion Optimization*, pp. 154–166, Springer, 2001.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[27] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm," in *Congress on Evolutionary Computation*, vol. 1, pp. 831–836, May 2002.

[28] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *IEEE World Congress on Computational Intelligence First IEEE Conference on Evolutionary Computation*, vol. 1, pp. 82–87, June 1994.

[29] C. Poloni, A. Giurgevich, L. Onesti, and V. Pediroda, “Hybridization of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics,” *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 403–420, 2000.

[30] D. Sarkar and J. M. Modak, “Pareto-optimal solutions for multi-objective optimization of fed-batch bioreactors using nondominated sorting genetic algorithm,” *Chemical Engineering Science*, vol. 60, no. 2, pp. 481–492, 2005.

[31] A. D. Nandasana, A. K. Ray, and S. K. Gupta, “Applications of the non-dominated sorting genetic algorithm (NSGA) in chemical reaction engineering,” *International Journal of Chemical Reactor Engineering*, vol. 1, 2003.

[32] S. Kannan, S. Baskar, J. D. McCalley, and P. Murugan, “Application of NSGA-II algorithm to generation expansion planning,” *IEEE Transactions on Power Systems*, vol. 24, no. 1, pp. 454–461, 2009.

[33] N. Agrawal, G. P. Rangaiah, A. K. Ray, and S. K. Gupta, “Design stage optimization of an industrial low-density polyethylene tubular reactor for multiple objectives using NSGA-II and its jumping gene adaptations,” *Chemical Engineering Science*, vol. 62, no. 9, pp. 2346–2365, 2007.

[34] S. Shan and G. G. Wang, “An efficient pareto set identification approach for multiobjective optimization on black-box functions,” *Journal of Mechanical Design*, vol. 127, p. 866, 2005.

[35] S. Kukkonen and J. Lampinen, “Ranking-dominance and many-objective optimization,” in *IEEE Congress on Evolutionary Computation*, pp. 3983–3990, IEEE, 2007.

[36] E. Galvan, M. Bose, and R. Malak, “A predictive pareto dominance based algorithm for many-objective problems,” in *10th World Congress on Structural and Multidisciplinary Optimization (WCSMO), Orlando, FL, May*, pp. 19–24, 2013.

[37] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, “Evolutionary many-objective optimization: A short review.,” in *IEEE congress on evolutionary computation*, pp. 2419–2426, 2008.

[38] E. Galvan, “A genetic algorithm approach for technology characterization,” Master’s thesis, Texas A&M University, College Station, TX, August 2012.

[39] L. S. Batista, F. Campelo, F. G. Guimaraes, and J. A. Ramirez, “A comparison of dominance criteria in many-objective optimization problems,” in *IEEE Congress on Evolutionary Computation*, pp. 2359–2366, 2011.

[40] D. W. Corne and J. D. Knowles, “Techniques for highly multiobjective optimisation: some nondominated points are better than others,” in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 773–780, ACM, 2007.

[41] J. D. Knowles, L. Thiele, and E. Zitzler, “A tutorial on the performance assessment of stochastic multiobjective optimizers,” tik report, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, 2006.

[42] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, “Performance assessment of multiobjective optimizers: An analysis and review,” *Evolutionary Computation, IEEE Transactions on*, vol. 7, no. 2, pp. 117–132, 2003.

[43] M. Emmerich, N. Beume, and B. Naujoks, “An emo algorithm using the hypervolume measure as selection criterion,” in *Evolutionary Multi-Criterion Optimization*, pp. 62–76, Springer, 2005.

[44] C. Igel, N. Hansen, and S. Roth, “Covariance matrix adaptation for multi-objective optimization,” *Evolutionary Computation*, vol. 15, no. 1, pp. 1–28, 2007.

[45] D. Brockhoff and E. Zitzler, “Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods,” in *IEEE Congress on Evolutionary Computation*, pp. 2086–2093, IEEE, 2007.

[46] K. Bringmann and T. Friedrich, “Approximation quality of the hypervolume indicator,” *Artificial Intelligence*, vol. 195, no. 0, pp. 265–290, 2013.

[47] J. Bader and E. Zitzler, “Hype: An algorithm for fast hypervolume-based many-objective optimization,” *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011.

[48] R. J. Malak, *Using Parameterized Efficient Sets to Model Alternatives for Systems Design Decisions.* Ph.d., Georgia Institute of Technology, Atlanta, GA, 2008.

[49] K. Ikeda, H. Kita, and S. Kobayashi, “Failure of pareto-based moeas: does non-dominated really mean near to optimal?,” in *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 2, pp. 957–962, 2001.

[50] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, “Combining convergence and diversity in evolutionary multiobjective optimization,” *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[51] D. K. Sobek II, A. C. Ward, and J. K. Liker, "Toyota's principles of set-based concurrent engineering," *Sloan Management Review*, vol. 40, no. 2, pp. 67–83, 1999.

[52] A. C. Ward, *A Theory of Quantitative Inference Applied to a Mechanical Design Compiler*. Ph.d., Massachusetts Institute of Technology, 1989.

[53] E. Davis, "Constraint propagation with interval labels," *Artificial Intelligence*, vol. 32, no. 3, pp. 281–331, 1987.

[54] G. A. Hazelrigg, *Systems Engineering: An Approach to Information-based Design*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[55] S. Ferguson, A. Gurnani, J. Donndelinger, and K. Lewis, "A study of convergence and mapping in preliminary vehicle design," *International Journal of Vehicle Systems Modelling and Testing*, vol. 1, no. 1, pp. 192–215, 2005.

[56] C. A. Mattson and A. Messac, "Concept selection using s-pareto frontiers," *AIAA Journal*, vol. 41, no. 6, pp. 1190–1198, 2003.

[57] C. A. Mattson and A. Messac, "Pareto frontier based concept selection under uncertainty, with visualization," *Optimization and Engineering*, vol. 6, no. 1, pp. 85–115, 2005.

[58] A. Gurnani, S. Ferguson, K. E. Lewis, and J. Donndelinger, "A constraint-based approach to feasibility assessment in preliminary design," *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, vol. 20, no. 4, pp. 351–367, 2006.

[59] K. T. Ulrich, "Estimating the technology frontier for personal electric vehicles," *Transportation Research Part C*, vol. 13, pp. 448–462, 2005.

[60] C. Huang, J. Galuski, and C. L. Bloebaum, *Multi-objective pareto concurrent subspace optimization for multidisciplinary design*, vol. 45. Reston, VA: American Institute of Aeronautics and Astronautics, 2007.

[61] T. Goel, R. Vaidyanathan, R. T. Haftka, W. Shyy, N. V. Queipo, and K. Tucker, "Response surface approximation of pareto optimal front in multi-objective optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 46, pp. 879–893, 2007.

[62] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms.* Chichester, UK: Wiley & Sons, 2001.

[63] R. Malak and E. Galvan, "Using predictive modeling techniques to solve multilevel systems design problems," in *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, 2010.

[64] R. J. Malak, L. Tucker, and C. J. J. Paredis, "Compositional modeling of fluid power systems using predictive tradeoff models," *International Journal of Fluid Power*, vol. 10, no. 2, pp. 45–55, 2009.

[65] H. Raiffa and R. Schlaifer, *Applied Statistical Decision Theory.* Boston, MA: Harvard University, 1961.

[66] J. W. Pratt, H. Raiffa, and R. Schlaifer, *Introduction to Statistical Decision Theory.* Cambridge: MIT Press, 1995.

[67] A. Brennan, S. Kharroubi, A. O'Hagan, and J. Chilcott, "Calculating partial expected value of perfect information via Monte Carlo sampling algorithms," *Medical Decision Making*, vol. 27, no. 4, pp. 448–470, 2007.

[68] A. Brennan, S. Kharroubi, A. O'Hagan, and J. Chilcott, "Calculating partial expected value of perfect information via monte carlo sampling algorithms,"

Medical Decision Making, vol. 27, no. 4, pp. 448–470, 2007.

[69] J. B. Oostenbrink, M. J. Al, M. Oppe, and M. P. Rutten-van Mölken, "Expected value of perfect information: an empirical example of reducing decision uncertainty by conducting additional research," *Value in Health*, vol. 11, no. 7, pp. 1070–1080, 2008.

[70] J. M. Ling, J. M. Aughenbaugh, and C. J. Paredis, "Managing the collection of information under uncertainty using information economics," *Journal of Mechanical Design*, vol. 128, no. 4, pp. 980–990, 2006.

[71] J. Sobieszczanski-Sobieski and R. T. Haftka, "Multidisciplinary aerospace design optimization: survey of recent developments," *Structural optimization*, vol. 14, no. 1, pp. 1–23, 1997.

[72] H. M. Kim, N. F. Michelena, P. Y. Papalambros, and T. Jiang, "Target cascading in optimal system design," *Journal of Mechanical Design*, vol. 125, no. 3, pp. 474–480, 2003.

[73] N. Michelena, H. Park, and P. Y. Papalambros, "Convergence properties of analytical target cascading," *AIAA Journal*, vol. 41, no. 5, pp. 897–905, 2003.

[74] H. M. Kim, D. K. Kumar, W. Chen, and P. Y. Papalambros, "Target exploration for disconnected feasible regions in enterprise-driven multilevel product design," *AIAA Journal*, vol. 44, no. 1, pp. 67–77, 2006.

[75] D. K. Kumar, W. Chen, and H. M. Kim, "Multilevel optimization for enterprise-driven decision-based product design," *Decision Making in Engineering Design*, pp. 203–215, 2006.

[76] I. Kroo and V. Manning, "Collaborative optimization: status and directions," in *Presented at the 8th AIAA/NASA/ISSMO Symposium on Multidisciplinary*

Analysis and Optimization, vol. 6, p. 8, 2000.

[77] X. Gu, J. E. Renaud, L. M. Ashe, S. M. Batill, A. S. Budhiraja, and L. J. Krajewski, “Decision-based collaborative optimization,” *Journal of Mechanical Design*, vol. 124, no. 1, pp. 1–13, 2002.

[78] E. J. Cramer, J. Dennis, Jr, P. D. Frank, R. M. Lewis, and G. R. Shubin, “Problem formulation for multidisciplinary optimization,” *SIAM Journal on Optimization*, vol. 4, no. 4, pp. 754–776, 1994.

[79] A. W. Wymore, *Model-based systems engineering: an introduction to the mathematical theory of discrete systems and to the tricotyledon theory of system design*, vol. 3. Boca Raton, FL: CRC press, 1993.

[80] F. Kadid, R. Abdessemed, and S. Drid, “Study of the fluid flow in a MHD pump coupling finite element-finite volume computations,” *Journal of Electrical Engineering*, vol. 55, no. 11-12, pp. 301–305, 2004.

[81] R. Wilcoxon, N. Lower, and D. Dlouhy, “A compliant thermal spreader with internal liquid metal cooling channels,” in *Semiconductor Thermal Measurement and Management Symposium, 2010. SEMI-THERM 2010. 26th Annual IEEE*, pp. 210–216, IEEE, 2010.

[82] H.-J. Kang and B. Choi, “Development of the mhd micropump with mixing function,” *Sensors and Actuators A: Physical*, vol. 165, no. 2, pp. 439–445, 2011.

[83] M. Hodes, R. Zhang, R. Wilcoxon, and N. Lower, “Cooling potential of galinstan-based minichannel heat sinks,” in *3th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, pp. 297–302, IEEE, 2012.

[84] D. J. Hartl, G. J. Frank, and J. W. Baur, “Embedded magnetohydrodynamic liquid metal thermal transport: validated analysis and design optimization,” *Journal of Intelligent Material Systems and Structures*, p. 1045389X16657429, 2016.

[85] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[86] E. Zitzler and L. Thiele, “Multiobjective optimization using evolutionary algorithms: A comparative case study,” in *International Conference on Parallel Problem Solving from Nature*, pp. 292–301, Springer, 1998.

[87] H. Esbensen and E. S. Kuh, “Design space exploration using the genetic algorithm,” in *ISCAS ’96., Connecting the World., IEEE International Symposium*, vol. 4, pp. 500–503, May 1996.

[88] C. M. Fonseca and P. J. Fleming, “On the performance assessment and comparison of stochastic multiobjective optimizers,” in *International Conference on Parallel Problem Solving from Nature*, pp. 584–593, 1996.

[89] S. Sayın, “Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming,” *Mathematical Programming*, vol. 87, no. 3, pp. 543–560, 2000.

[90] J. M. Bader, *Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods*. PhD thesis, ETH Zürich, 2010.

[91] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello, “Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective op-

timization," *Evolutionary Computation, IEEE Transactions on*, vol. 16, no. 4, pp. 504–522, 2012.

[92] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimisation," in *Congress on Evolutionary Computation*, vol. 2, pp. 878–885, IEEE, 2003.

[93] C. Erbas, S. Cerav-Erbas, and A. Pimentel, "Multiobjective optimization and evolutionary algorithms for the application mapping problem in multiprocessor system-on-chip design," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 358–374, 2006.

[94] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 4, pp. 257–271, 1999.

[95] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, 2002.

[96] R. Veltkamp, "Shape matching: Similarity measures and algorithms," in *SMI 2001 International Conference on Shape Modeling and Applications*, pp. 188–197, IEEE, 2001.

[97] M. Gromov, *Metric structures for Riemannian and non-Riemannian spaces*, vol. 152. Birkhuser Boston, 2006.

[98] J. Rugis and R. Klette, "A scale invariant surface curvature estimator," *Advances in Image and Video Technology*, pp. 138–147, 2006.

[99] A. Davies and P. Samuels, *An introduction to computational geometry for curves and surfaces.* Oxford University Press, Inc., 1996.

[100] J. Munkres, *Elements of algebraic topology*, vol. 2. Menlo Park, CA: Addison-Wesley, 1984.

[101] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "Mesh: Measuring errors between surfaces using the hausdorff distance," in *IEEE International Conference in Multimedia and Expo*, vol. 1, pp. 705–708, August 2002.

[102] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[103] P. Wolfe, "Finding the nearest point in a polytope," *Mathematical Programming*, vol. 11, no. 1, pp. 128–149, 1976.

[104] H. Edelsbrunner and D. R. Grayson, "Edgewise subdivision of a simplex," *Discrete & Computational Geometry*, vol. 24, no. 4, pp. 707–719, 2000.

[105] E. N. Gonalves, R. M. Palhares, R. H. C. Takahashi, and R. C. Mesquita, "Algorithm 860: Simples—an extension of freudenthal's simplex subdivision," *ACM Transactions on Mathematical Software*, vol. 32, no. 4, pp. 609–621, 2006.

[106] K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, pp. 205–230, September 1999.

[107] K. Deb, J. Horn, and D. E. Goldberg, "Multimodal deceptive functions," *Complex Systems*, vol. 7, no. 2, pp. 131–154, 1993.

[108] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 2, pp. 284–302, 2009.

[109] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: test cases, approximations, and applications," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004.

[110] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 5, pp. 477–506, 2006.

[111] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, A. Abraham, L. Jain, and R. Goldberg, "Scalable test problems for evolutionary multiobjective optimization evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, Advanced Information and Knowledge Processing, pp. 105–145, Springer Berlin Heidelberg, 2005.

[112] M. Laumanns, L. Thiele, and E. Zitzler, "An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method," *European Journal of Operational Research*, vol. 169, no. 3, pp. 932–942, 2006.

[113] A. Messac, A. Ismail-Yahaya, and C. A. Mattson, "The normalized normal constraint method for generating the pareto frontier," *Structural and Multidisciplinary Optimization*, vol. 25, no. 2, pp. 86–98, 2003.

[114] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 311–338, 2000.

[115] G. Ying, S. Lei, and Y. Pingjing, "Study on multi-objective genetic algorithm," in *3rd World Congress on Intelligent Control and Automation*, vol. 1, pp. 646–650, 2000.

[116] D. M. Tax and R. P. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, no. 11, pp. 1191–1199, 1999.

[117] V. Vapnik, *The Nature of Statistical Learning Theory.* New York, NY: Springer, 1995.

[118] B. Scholkopf and J. A. Smola, *Learning with Kernels.* Cambridge, MA: MIT Press, 2002.

[119] B. Schölkopf, "The kernel trick for distances," in *TR MSR 2000-51, Microsoft Research*, Citeseer, 1993.

[120] P. Wolfe, "A duality theorem for nonlinear programming," *Quarterly of Applied Mathematics*, vol. 19, no. 3, p. 239, 1961.

[121] D. M. Tax and R. P. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[122] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *Neural Information Processing Systems*, pp. 409–415, 2000.

[123] E. Galvan and R. Malak, "A predictive pareto dominance based algorithm for many-objective problems," in *10th World Congress on Structural and Multidisciplinary Optimization (WCSMO), Orlando, FL. 19-24 May*, International Society for Structural and Multidisciplinary Optimisation (ISSMO), 2013.

[124] E. Galvan, C. Hsiao, S. Vermillion, and R. Malak, "A parallel approach for computing the expected value of gathering information," *SAE International Journal of Materials and Manufacturing*, vol. 8, no. 2015-01-0436, pp. 271–282, 2015.

[125] S. D. Vermillion, "Modeling a hydraulic hybrid drivetrain: Efficiency considerations," Master's thesis, University of Missouri - Columbia, Columbia, MO, 2011.

[126] A. A. Golverk, "Mathematical calculation of the performance map of internal combustion engine," in *International Congress and Exposition*, SAE Technical Paper Series, (Detroit, MI), pp. 1–5, Publ by SAE, February 1992.

[127] J. Bucklew, *Introduction to rare event simulation.* New York, NY: Springer-Verlag, 2004.

[128] M. L. Stein, *Interpolation of spatial data: some theory for kriging.* New York, NY: Springer, 1999.

[129] N. Cressie, "The origins of kriging," *Mathematical Geology*, vol. 22, no. 3, pp. 239–252, 1990.

[130] W. C. M. v. Beers and J. P. C. Kleijnen, "Kriging interpolation in simulation: A survey," in *Proceedings of the 36th Conference on Winter simulation*, 2004.

[131] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, "DACE- a MATLAB Kriging toolbox, version 2.0," Technical Report IMM-TR-2002-12, Technical University of Denmark, 2002.

[132] D. J. Hartl, E. Galvan, R. J. Malak, and J. W. Baur, "Parameterized design optimization of a magnetohydrodynamic liquid metal active cooling concept," *Journal of Mechanical Design*, vol. 138, no. 3, p. 031402, 2016.

[133] D. J. Hartl, G. H. Huff, H. Pan, L. Smith, R. L. Bradford, G. J. Frank, and J. W. Baur, "Analysis and characterization of structurally embedded vascular antennas using liquid metals," in *SPIE Smart Structures and Materials+*

Nondestructive Evaluation and Health Monitoring, pp. 980333–980333, International Society for Optics and Photonics, 2016.

[134] Y. Wu, C. McKee, and P. Armstrong, “Galileo epd,” Technical Report 86-01, University of Kansas, 1986.

[135] M. MacCaig, *Permanent Magnets in Theory and Practice.* New York, NY: Halsted Press a Division of Wiley and Sons, 1977.

[136] M. Qin and H. H. Bau, “When MHD-based microfluidics is equivalent to pressure-driven flow,” *Microfluidics and Nanofluidics*, vol. 10, no. 2, pp. 287–300, 2011.

[137] C. Pozrikidis and D. Gartling, “Fluid dynamics: Theory, computation, and numerical simulation,” *Applied Mechanics Reviews*, vol. 55, p. 55, 2002.

[138] R. Balling, “Design by shopping: A new paradigm?,” in *Proceedings of the Third World Congress of structural and multidisciplinary optimization (WCSMO-3)*, vol. 1, pp. 295–297, 1999.

[139] R. R. Parker, E. Galvan, and R. J. Malak, “Technology characterization models and their use in systems design,” *Journal of Mechanical Design*, vol. 136, no. 7, p. 071003, 2014.

[140] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *International joint Conference on Artificial Intelligence*, vol. 14, pp. 1137–1145, 1995.

[141] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response surface methodology: process and product optimization using designed experiments*, vol. 705. New York, NY: John Wiley & Sons, 2009.

[142] L. L. Scharf, *Statistical signal processing*, vol. 98. Reading, MA: Addison-Wesley Reading, MA, 1991.