CONTROL OF A BENCHMARK STRUCTURE USING GA-OPTIMIZED FUZZY LOGIC CONTROL

A Thesis

by

DAVID ADAM SHOOK

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

MASTER OF SCIENCE

December 2006

Major Subject: Civil Engineering

CONTROL OF A BENCHMARK STRUCTURE USING GA-OPTIMIZED FUZZY LOGIC CONTROL

A Thesis

by

DAVID ADAM SHOOK

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

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Paul N. Roschke
Committee Members,	Harry Jones
	Sergiy Butenko
Head of Department,	David Rosowsky

December 2006

Major Subject: Civil Engineering

ABSTRACT

Control of a Benchmark Structure Using GA-Optimized Fuzzy Logic Control.
(December 2006)
David Adam Shook, B.S., Texas A&M University
Chair of Advisory Committee: Dr. Paul N. Roschke

Mitigation of displacement and acceleration responses of a three story benchmark structure excited by seismic motions is pursued in this study. Multiple 20-kN magnetorheological (MR) dampers are installed in the three-story benchmark structure and managed by a global fuzzy logic controller to provide smart damping forces to the benchmark structure. Two configurations of MR damper locations are considered to display multiple-input, single-output and multiple-input, multiple-output control capabilities. Characterization tests of each MR damper are performed in a laboratory to enable the formulation of fuzzy inference models. Prediction of MR damper forces by the fuzzy models shows sufficient agreement with experimental results.

A controlled-elitist multi-objective genetic algorithm is utilized to optimize a set of fuzzy logic controllers with concurrent consideration to four structural response metrics. The genetic algorithm is able to identify optimal passive cases for MR damper operation, and then further improve their performance by intelligently modulating the command voltage for concurrent reductions of displacement and acceleration responses. An optimal controller is identified and validated through numerical simulation and full-scale experimentation. Numerical and experimental results show that performance of the controller algorithm is superior to optimal passive cases in 43% of investigated studies.

Furthermore, the state-space model of the benchmark structure that is used in numerical simulations has been improved by a modified version of the same genetic algorithm used in development of fuzzy logic controllers. Experimental validation shows that the state-space model optimized by the genetic algorithm provides accurate prediction of response of the benchmark structure to base excitation.

DEDICATION

To my loving and supporting family.

ACKNOWLEDGEMENTS

I gratefully acknowledge support of the National Center for Research on Earthquake Engineering, Taipei, Taiwan. In addition, this research is supported in part by the National Science Foundation (Grant No. OISE-0553917, Dr. Anne Emig, Project Officer).

NOMENCLATURE

ANFIS	Adaptive Neuro-Fuzzy Inference System
FLC	Fuzzy Logic Controller
FIS	Fuzzy Inference System
GA	Genetic Algorithm
LVDT	Linear Variable Displacement Transducer
ER	Electrorheological
MR	Magnetorheological
NSGA-II	Non-Dominated Sorting Genetic Algorithm II
NSGA-II CE	Non-Dominated Sorting Genetic Algorithm II with Controlled Elitism
PAES	Pareto-Archived Evolution Strategy
RMS	Root-Mean Squared
RMSE	Root-Mean-Squared-Error
SPEA	Strength Pareto Evolutionary Algorithm
V	Volts

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

Page

Page

Page

INTRODUCTION

General

Since the first ages of civilization technological advancement has progressed rapidly during certain periods and has been caught in doldrums through others. In more recent years, engineers have continued to inspire the imagination and surmount prior limitations. One such example is construction of the Tower 101 building in Taipei, Taiwan (Fig. 1), which has a height of 508 m and is currently the tallest building in the world. Although it was once thought to be impractical to construct such a tall building in a location that is often subject to seismic temblors and is susceptible to numerous typhoons each summer, Tower 101 has shown the resilience of technological progress and innovation. As architects and designers continue to push the limits of structural steel and other modern materials, simple solutions of the past must make way for innovations of the future. New technologies are continually being employed by structural engineers in their practice. Smarter, not stronger, is the language of the populous.

Fig. 1. Tower 101 in Taipei, Taiwan

This thesis follows the style of the *Journal of Structural Engineering,* ASCE.

Structural Control Systems

One recent innovation for constructed facilities such as buildings and bridges that has garnered much attention is the concept of structural control, which was first proposed by Yao (1972). Structural control can be defined as any system that dissipates unwanted energy in a structure that is imparted to it by internal or external perturbations.

Numerous domestic and foreign structures are incorporating this technology into new and retrofit construction projects. Engineers from the United States have pioneered these technologies as exampled by the Citicorp building in New York City, New York. Here a passive tuned mass damper was installed during construction of the building in 1977. A variety of other North American examples include the base isolation system installed in the University of Southern California University Hospital located in Los Angles, California, and the tuned water damping system installed in the One Wall Centre tower in Vancouver, British Columbia, Canada. A more recent example of such technology includes the largest mechanical damping system in the world, which was installed in Tower 101 in 2004. Each of these full-scale implementations of structural control devices share one key similarity: passive control. That is, the control device is not altered by any external system during operation.

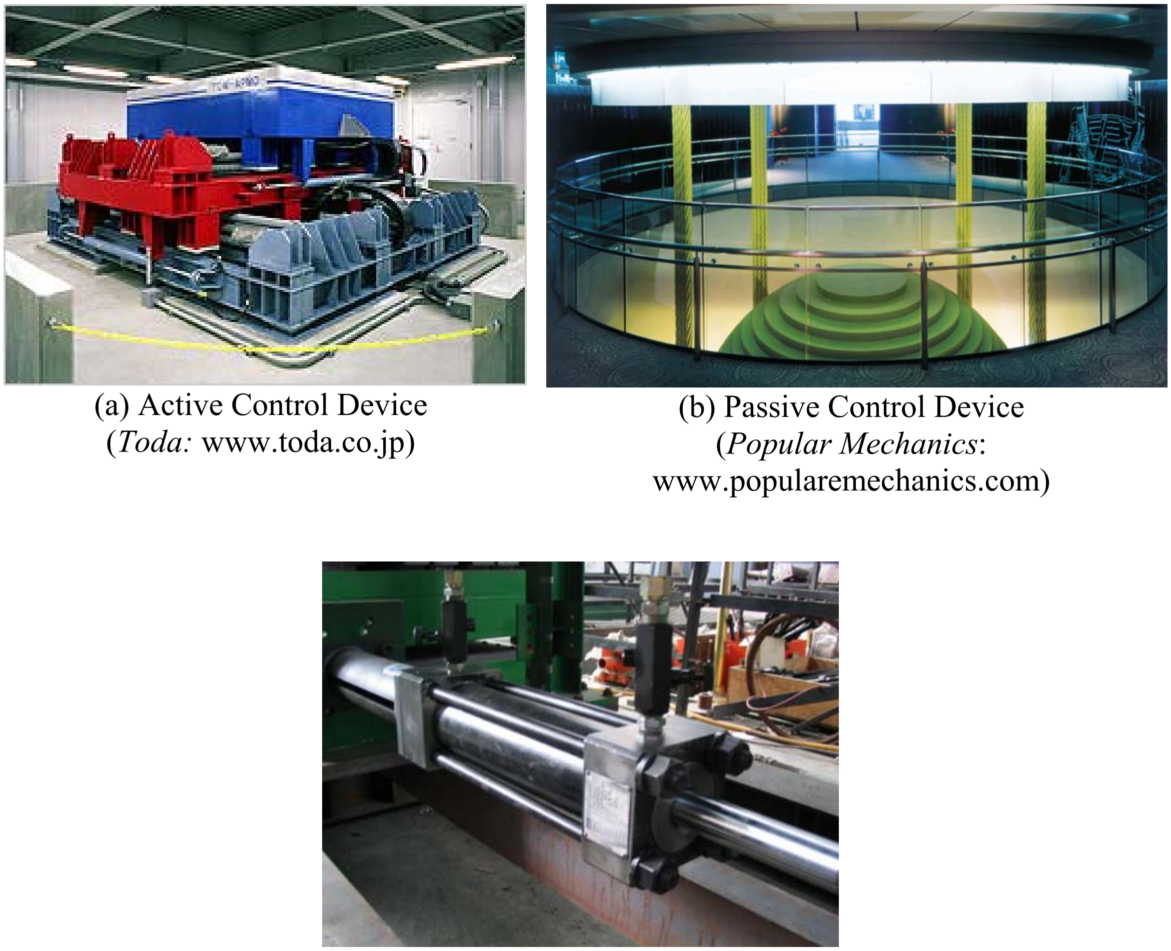
In Asian countries researchers have ventured into active and semi-active control systems where the properties of the implemented damping device are altered in real time for the intelligent reduction of motion experienced by the structure. An abbreviated listing of currently installed semi-active and active structural control systems is shown in Table 1. For a more complete list of full-scale installations consult Spencer and Nagarajaiah (2003).

Table 1. Example Implimentations of Structural Control Devices

Structure	Location	Control System
Yokohama Land Mark Tower	Yokohama, Japan (1993)	Active mass damper
T.C. Tower	Kau-Shon, Taiwan (1997)	Active mass damper
Laxa Osaka	Osaka, Japan (1999)	Semi-active mass damper
Shin-Jei Building	Taipei, Taiwan (1999)	Active mass damper
Keio University Engineering Bldg.	Tokyo, Japan (2000)	Variable-orifice damper
Harumi Island Triton Square	Tokyo, Japan (2001)	Coupled building control

Overview of Structural Control

In the following section a brief synopsis of structural control devices is provided. These structural control devices are classified into three groups: active, passive, and semi-active. An example of each of these control devices is shown in Fig. 2.



(a) Active Control Device
(*Toda:* www.toda.co.jp)

(b) Passive Control Device
(*Popular Mechanics*:
www.popularemechanics.com)

(c) Semi-Active Control Device

Fig. 2. Example Control Devices

Active control techniques utilize mechanical systems to impose actuated forces on a structure for the purpose of mitigating excitation. These systems employ a variety of technologies, but generally include hydraulic actuators such as the one shown in Fig. 2a. Here an active mass damping system is used to mitigate wind loads in a tall structure. To provide optimal control these devices often produce significant forces on a structure to counteract external disturbances. However, when internal forces acting on a structure such as a tall building become large, the optimal control forces required become exceptionally large as well. Thus, very large actuators are needed which, in turn, require exceedingly large amounts of power. In the event of a loss of power (e.g. in an earthquake), beneficial effects of the actuators would be nullified. Moreover, dynamic instability can occur with active control and possibly aggravate response of a building instead of reducing it.

By contrast, passive control approaches utilize inert force-resisting mechanical systems that absorb the energy of a structure and thereby mitigate excitation. Passive control devices employ a variety of technologies, and are effective in a number of situations. A commonplace example of a passive control system is the set of shock absorbers that are commonly installed in automobiles. A structural engineering example is the pendulum and tuned mass dampers that are installed in Tower 101 as shown in Fig. 2(b). Passive devices can be categorized into three basic types: springs, viscous dampers, and energy absorbing structural elements. When correctly tuned to the motion of the structure, these devices can effectively absorb energy from unwanted perturbations. These techniques are very reliable and generally require no power. Moreover, dynamic instability is not a concern since no energy is imparted to the structure; rather, energy is absorbed. Passive devices are often only optimized for a single scenario, and thus lack adaptive capabilities. As a consequence if structural characteristics change beyond the operational range of the passive device, it becomes ineffective in fulfilling its intended purpose.

The final category of structural control is termed semi-active. This approach utilizes variable force-resisting mechanical systems that absorb energy and thereby mitigate undesirable motion. Semi-active control systems employ a variety of technologies including its more common form, variable viscous damping. This variable component, of an otherwise passive system, highlights a key difference between passive and semi-active control, namely controllability. With a modest amount of power semi-active control combines the adaptability of active control with the reliability of passive control. Furthermore, dynamic instability is not a concern for semi-active control approaches since these semi-active devices only modulate their level of resistance and do not impart energy into the structure.

In a recent publication by Spencer and Nagarajaiah (2003) an overview of current technologies in the realm of structural control are discussed. Evidence justifying the need for semi-active control systems is provided. This paper calls for the continued development of both technology and controller algorithms. Toward this goal, the current study intends to advance the development of effective structural control algorithms and improve optimization methods.

Justification for Proposed Control System

In what follows, control of a multi-degree of freedom structure is pursued. Several key components are required for the implementation of such control systems. An overview of the rationale and background for each component is provided below to give the reader a foundation before providing details of their development and integration.

Control Device

For this study semi-active devices are selected as a means to mitigate structural responses to excitation. Two general categories of semi-active devices include variable orifice dampers and rheological dampers. Variable orifice dampers typically consist of a piston in a casing that is filled with a fluid such as water or oil (see Fig. 3). Controllability is derived from constricting the flow of fluid during motion of the piston. Recently, these types of viscous dampers have been determined to be effective in mitigating wind excitations as noted by Kim and Adeli (2005) and Reigles and Symans (2005). Generally, mechanical properties of the damper are tuned for optimal alleviation of acceleration responses of the structure where wind excitation is most taxing. As a result, the reduction displacement responses are often a secondary priority. Moreover, constricting the flow of the fluid through an orifice can require more power than a simple backup battery system could provide. This is crucial in the event of a power failure. Although variable orifice dampers often require less power than active control applications, it is a concern of engineers.

Rheological dampers offer a similar resistance to motion, yet resistance is derived from the modulation of fluid properties rather than the alteration of orifice size. Two commonly employed fluids that exhibit such a rheological phenomenon are electrorheological (ER) and magnetorheological (MR) dampers. Both fluids alter their resistance to motion when current is applied. This idea was initially proposed by

Winslow (1948) with the invention of electromagnetic (ER) fluid. ER fluid lacks significant resistance to motion and requires a large amount of voltage; consequently, it has been largely abandoned by engineers who instead are opting for use of MR fluid.

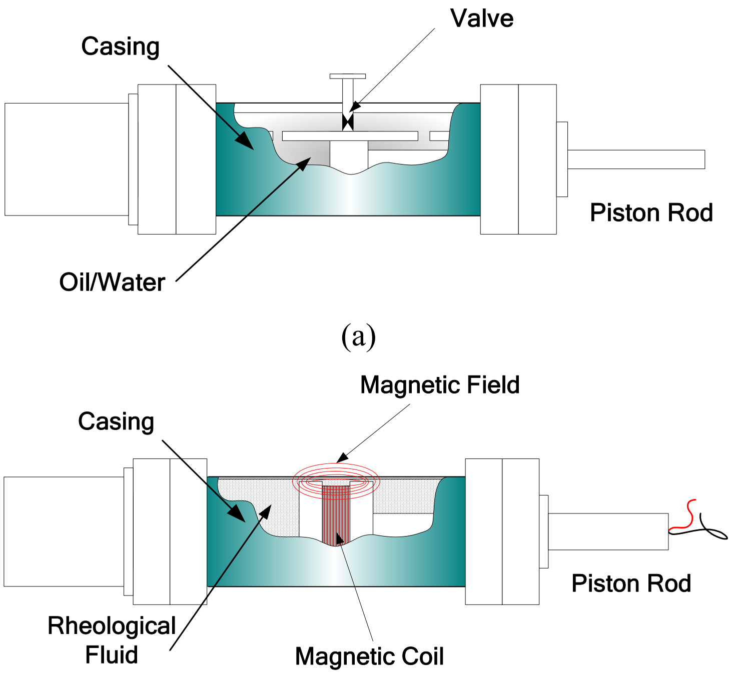
Dampers that utilize MR fluid for the modulation of resistance are termed MR dampers. An example of this type of device is shown in Fig. 3(b). MR dampers typically consist of a piston in a steel casing that is filled with MR fluid. MR fluid consists of base oil, magnetizable particles (iron), and a stabilizer. An electrical current is applied to the MR fluid through a coil wrapped around the piston head which produces a magnetic field. The magnetized particles align with the magnetic field created by the coil when a current is applied. Upon alignment of the rheological particles the effective density of the oil-based liquid changes drastically. Since the particles generally align within a few milliseconds, the fluid converts from a free flowing oil-based liquid to a high density semi-solid almost instantaneously. This conversion process is also reversible when the magnetic field is removed. Thus, by continuously varying the current the MR fluid can produce a sweeping range of effective densities, thus providing the damper with significant controllability.

Two key advantages of rheological dampers in comparison with variable orifice dampers include reaction time of the controlling event and lower power requirements. Primarily for these reasons, rheological dampers, or more explicitly, MR dampers, are selected for control system implementation in this study.

In the previous text justification for the semi-active components of control has been provided. The next step in development of an efficient vibration control system is to exploit the beneficial characteristics of MR dampers. The following section presents the controller algorithm that is to be developed.

Controller Algorithm

For active and semi-active systems a controller is necessary to manage the mechanical equipment. A controller is any algorithm that adjusts characteristics of a control device. Since a semi-active control device is selected for use in this study, namely an MR damper, a controller is also necessitated.



(a)

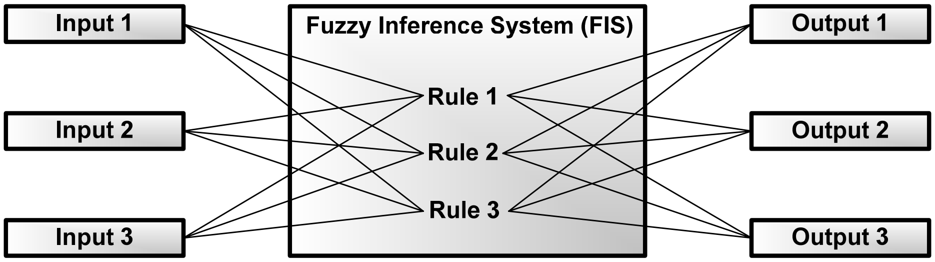
(b)

Fig. 3. Sketches of (a) Variable Orifice and (b) Rheological Dampers

Numerous algorithms have been developed prior to and since the inception of the concept of structural control. Control algorithms have a sweeping range of applications and capabilities. In the past, much of the structural control research community gravitated towards use of what is termed here as "traditional" control methods. Examples of these controllers include Proportional, Integration, and Derivative (PID), Linear Quadratic Gaussian (LQG), and H-Infinity (H_∞). Traditional control algorithms can provide adequate control management for certain classes of problems, but this is often at the expense of a strong robust nature. This is due to the sensitivity of traditional controllers to characteristics of the structure itself such as mass, stiffness, and damping. Thus, if the structural properties vary from those used to develop the control algorithm, the effectiveness of many of these control algorithms diminishes significantly as discussed by Casciati *et al.* (1994) and Deoskar *et al.* (1996).

In more recent studies, controllers that make use of fuzzy logic have gained acceptance in the research community for their robust nature and ability to account for uncertainties. Fuzzy sets were first introduced by Zadeh (1965) as a means of effective control when considering uncertainties. A fuzzy inference system (FIS) is a compilation of IF [ ], THEN [ ] statements, that, when combined in unison, provide a unique form of control. Avoiding complex and computationally expensive state observers or estimators that are often used in traditional control, fuzzy logic seeks a more simplistic approach to save computational time and account for non-linear relationships with relative ease. Logic statements or 'rules' relate any provided input signal to desired output signals. Fuzzy logic shows its multifaceted nature in that it can be developed for both numerical modeling of systems and as a controller of a system. In this study fuzzy logic is used for both numerical modeling of MR dampers and management of MR damper command signals.

Fuzzy inference systems are adaptable for control of multiple-input, single-output (MISO) and multiple-input, multiple-output (MIMO) systems. Nonlinear relationships are required for mapping specified inputs to desired outputs for both MR damper modeling and development of a fuzzy logic controller (FLC). Fuzzy logic is inherently proficient at nonlinear relationships, thus making it an even more viable candidate for MR damper modeling and controller implementation. A sketch of a nonspecific fuzzy inference system is shown in Fig. 4.



Rule 1: IF [Input 1, Input 2, Input 3], THEN [Output 1, Output 2, Output 3]
Rule 2: IF [Input 1, Input 2, Input 3], THEN [Output 1, Output 2, Output 3]
Rule 3: IF [Input 1, Input 2, Input 3], THEN [Output 1, Output 2, Output 3]

Fig. 4. Nonspecific Fuzzy Inference System

Controller Inputs

Dynamic motion of the structure must be characterized to provide fuzzy logic controllers with information in order to identify voltage singals to modulate MR damper resistance levels. In general there are three commonly used metrics for characterization of dynamic motion: displacement, velocity, and acceleration. Although displacement is an ideal characterization of building motion, it is fiscally expensive to acquire accurate displacement data in real time for a civil engineering structure such as a tall building or bridge. For example, displacements are generally measured with linear velocity displacement transducers (LVDTs). They are often unreliable for use in actual structures in the event of a strong seismic motion that could damage such devices. Furthermore, they usually require some fixed location for installation which is often impossible to acquire in real civil engineering structures. Velocity transducers are simply accelerometers fitted with a state-estimator. Thus, accuracy of the signal depends on the accuracy of real-time time integration of acceleration signals. This is not a computationally efficient means of response observation. Accelerometers, which measure acceleration, are relatively inexpensive and are more reliable than LVDTs or velocity transducers. Therefore, for this study, building characterization is quantified by acceleration feedback from specified degrees of freedom.

Since accelerometers are very sensitive to excitation, a common issue with their use is their inherent "noisy" signal or poor signal quality. That is, they are prone to a poor signal to noise ratio. A key advantage for the of fuzzy inference systems is their ability to provide acceptable and even optimal output signals despite significant noise in input signals. Many algorithms based on traditional control theory are hampered by unwanted noise that is common in readings from most accelerometers. What is more, many traditional control methods require multiple building characterization types such as displacement and acceleration to compute control signals, whereas fuzzy logic can be optimized for any set of input or output signals.

Optimization Algorithm

The system proposed in this study for optimal control of a civil engineering structure utilizes MR dampers which are managed by a fuzzy logic controller that employs acceleration feedback for characterization of building motion. This system is inherently nonlinear due to the nonlinearity associated with the MR dampers response to motion. The complexity of this system requires automation to develop an optimal FLC. A heuristic optimization process that is reasonably free of user guidance is desired since understanding the complex relationships involved such a system is a daunting task. For this reason a novel approach to optimization is considered here through use of a genetic algorithm (GA). A robust trial-and-error approach to optimization is studied that allows the user to optimize a FLC by training with a specified data set and adjustment of a relatively few user-defined parameters.

Use of genetic algorithms for optimization, since first bring proposed by Holland *et al.* 1975, has striven towards a blind optimization process that identifies unexpected relationships where other optimization processes can not. The optimization is to be robust and consider numerous potential solutions. With these thoughts in mind, genetic algorithms have been employed to identify optimal solutions for a variety of optimization objectives. The general flow of logic of a rudimentary genetic algorithm can be described as shown in Fig. 5.

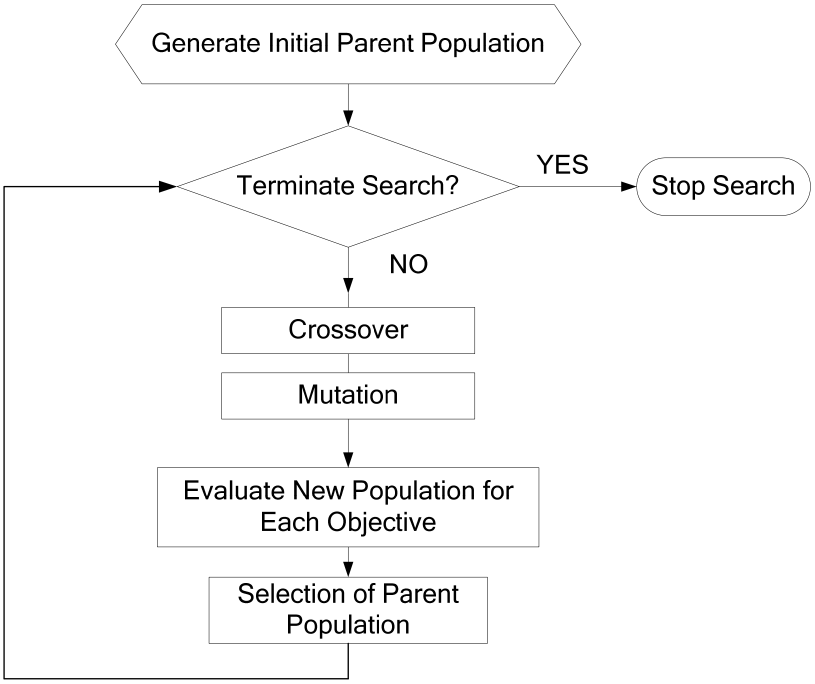

Generate Initial Parent Population
Terminate Search?
YES
Stop Search
NO
Crossover
Mutation
Evaluate New Population for Each Objective
Selection of Parent Population

Fig. 5. Typical Flow of Logic of a Nonspecific Genetic Algorithm

A population of solutions is considered at each generation as opposed to a single solution as exampled by neural network optimization techniques. By this fundamental principle, genetic algorithms are much more explorative than many prior optimization processes.

Recent advances in GA methodology have been exploited by numerous researchers. For example, Schaffer (1985) incorporated the idea of non-domination. In later studies Fonseca and Fleming (1993) and Goldberg (1989) considered a truly multi-objective optimization with the employment of Pareto fronts. The ability to optimize on multiple objectives simultaneously was crucial in bringing GA into the robust automated optimization process it is today. In recent years researchers of genetic algorithms such as Deb (2001) have brought GA-based approaches to the forefront of multi-objective optimization.

Summary of Proposed Control System

Thus, for the proposed control system MR dampers that are managed by a FLC are used to provide optimal control forces to a civil engineering structure. The FLC must be able to consider multiple concurrent structural response optimization objectives. Thus, a GA is employed for its robust and multi-objective optimization capabilities. FLC performance is then experimentally substantiated through full-scale testing on a benchmark structure.

Numerical Modeling and Simulation

Since no online learning algorithms are pursued in the current study, two entities require numerical modeling in advance of experimental testing to produce a fuzzy logic controller. To numerically simulate dynamic responses of the building to external forces, a feedback control loop is required as shown in Fig. 6**.** As described in a later section, a state-space model of the benchmark structure is created using a modified version of the genetic algorithm originally used in FLC development. Second, the MR dampers are modeled using a neuro-fuzzy approach. Both the state-space model of the benchmark structure and fuzzy models of the MR dampers are created using results from experimental tests that isolate each entity to investigate their dynamic characteristics independently of each other. Subsequently, these models are incorporated into the numerical simulation such that the MR dampers are assumed to be “force producers” as can be shown by basic mechanics of free bodies and (see Fig. 7). Notation of excitation and response forces in Fig. 7 are derived from Chopra (2006). Also, F_{MR} is the resistance force of the MR damper and V denotes voltage specified for damper operation.

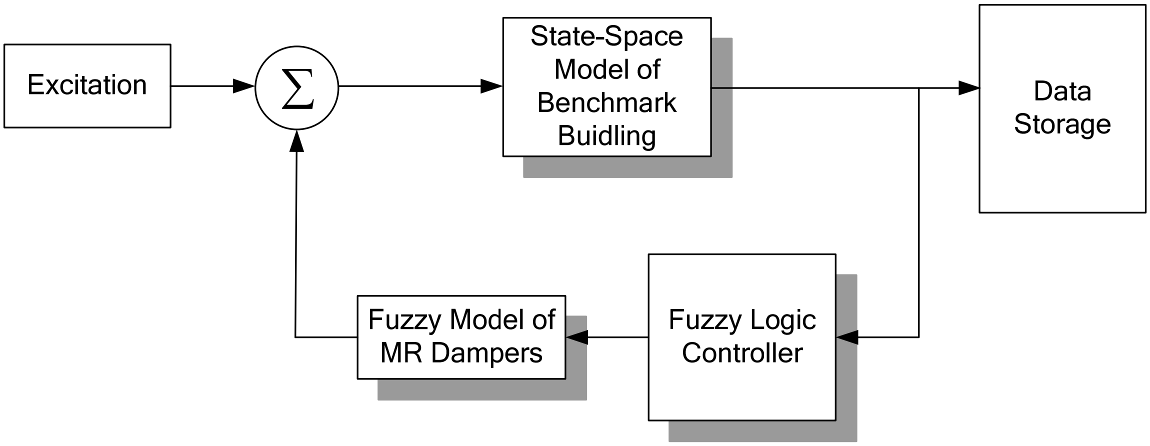

Excitation
∑
State-Space Model of Benchmark Buidling
Data Storage
Fuzzy Model of MR Dampers
Fuzzy Logic Controller

Fig. 6. Flow Chart of Numerical Simulation

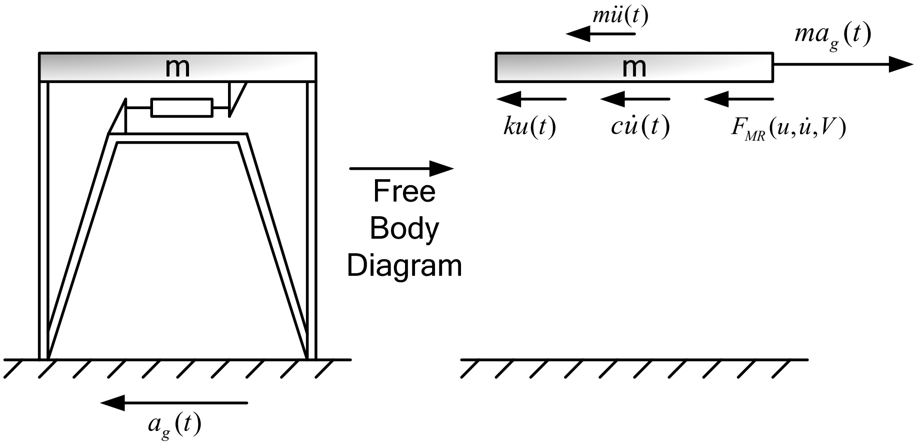


Fig. 7. Free Body Diagram of Simple Structure

Experimental Studies

In this study experimental substantiation of GA-optimized FLCs is conducted with employment of two 20 kN MR dampers and a 9 m tall benchmark structure. Experimental trials were conducted at the National Center for Research on Earthquake Engineering (NCREE) located in Taipei, Taiwan, with the aid of local researchers. A seismic simulator located at NCREE is used for experimental testing of the FLCs for a variety of near- and far-field excitations. Furthermore, performance tests of MR dampers were conducted by an NCREE researcher to aid in the formulation of numerical models representing the MR dampers.

Software

MATLAB 7.2, Simulink, and a complementary set of toolboxes are used to conduct numerical simulation and computations in the work that follows. Moreover, a genetic algorithm is used which is based on work by Deb and Goel (2001), and further advanced by Kim and Roschke (2006b). Significant alterations of these algorithms are described in what follows and are used in the simulations. Since a GA is computationally intensive a cluster-based supercomputer is employed for GA calculations to expedite optimization.

1. REVIEW OF LITERATURE AND RELEVANT TOPICS

1.1. General

Current and prior research to this study are noted and discussed in the following section. The compilation of relevant research is vital to understanding the intellectual merit of the proposed control system. What is more, many key topics are derived from prior work which is disseminated in what follows.

1.2. Modeling of a MR Damper

Any method used in MR damper modeling requires several key components, the most important of which is the ability to generate non-linear relationships that accurately describe the response of the damper to motion and voltage. Resistance of an MR damper is modulated by application of a range of voltages as discussed in subsequent sections. Generally, numerical models consist of a set of input variables and they output one or more results that correspond to the device that is being modeled. The methods of mapping these relationships differentiate the various modeling techniques.

Several, often competing, factors are to be considered when designing a numerical model of an MR damper. The first and most important is accuracy. Accurate models of dampers are critical for future controller formulation. Robustness is also a vital aspect of a numerical model. That is, the model must be adept in accounting for new scenarios and uncertainties. Finally, the model should be as computationally fast as possible. In subsequent development of FLC controllers numerous computational cycles will be required; as such rapid calculation of MR damper forces is needed.

Modeling techniques for MR dampers can be summarized into two categories: analytical and non-analytical. A well known analytical technique that accurately models MR damper behavior through a set of seven governing differential equations has previously been proposed by Spencer *et al.* (1997). This proposed method is a modified version of the classic Bouc-Wen model initially proposed by Wen (1976). Validation of the Bouc-Wen model involves using several types of motion and voltage time-histories both of which are sinusoidal and random. All presented data show validity of a highly accurate model. Also, three error metrics have been proposed to quantify the accuracy of force prediction of the Bouc-Wen model. A significant shortcoming of the Bouc-Wen model is computational efficiency. Although a highly accurate model of the MR damper is formulated, computational time is quite expensive as described by Schurter and Roschke (2000). Since heuristic optimization, as employed in this study, requires numerous expensive computational cycles for FLC development, this type of modeling is not selected for this study.

Other researchers have since developed numerous modified Bouc-Wen models that have more efficient computational algorithms. A modified Bouc-Wen model has been proposed by Lin *et al.* (2005) for a 3 kN MR damper in which on-line parameter identification is conducted. Here several models of the 3 kN MR damper operating at discrete voltage levels are generated. Then cubic interpolation is used to identify MR damper characteristics with specified voltages lie between identified models. Although this model is not as accurate as the one proposed by Spencer *et al.* (1997), it is very computationally efficient and suitable for heuristic optimization of a control system. As an alternative to on-line parameter identification, a genetic algorithm has been proposed by Giulea *et al.* (2004) for the identification for Bouc-Wen parameters.

More recently Jiménez and Álvarez-Icaza developed an alternative modeling method, termed the LuGre friction model (2005). This modified LuGre friction model, original proposed by Canudas *et al.* (1995), has been simplified and utilizes a set of linear approximations to model an inherently non-linear device. The LuGre friction model is reported to be more computationally efficient than the Bouc-Wen model as proposed by Spencer *et al.* (1997). The modified LuGre approach models damper behavior with a set of two differential equations. When results are compared to the Spencer *et al.* (1997) Bouc-Wen model a similar degree of accuracy can be noted.

Other researchers have incorporated fuzzy logic in modeling of MR dampers as referenced by Schurter and Roschke (2000), Likhitruangsilp and Roschke (2003), Atray and Roschke (2004), Oh *et al.* (2004), and Kim *et al.* (2006). In these studies a neuro-fuzzy approach to MR damper modeling has been proposed that shows high accuracy in tandem with efficient computational effort. Previous neuro-fuzzy modeling studies from this group of researchers utilized an Adaptive Neuro-Fuzzy training of Sugeno-type (ANFIS) for fuzzy training of the fuzzy model as initially developed by Jang (1993). ANFIS is an automated optimization process which, through trial and error, formulates a set of fuzzy rules that relate input and output arguments. A similar approach is utilized in the current study, but modifications from prior modeling efforts are incorporated.

1.3. Control System Background and Optimization

A number of control strategies exist in the research community today. They can be generally summarized into two approaches: traditional and non-traditional control. Traditional control is an umbrella term that describes numerous control algorithms including commonly used algorithms such as PID, LQR, skyhook, or H_∞. These types of controllers are prevalent in many of the research communities interested in control. Less commonly utilized are non-traditional controllers. Examples of this classification include fuzzy logic and neural network controllers. Generally speaking, traditional controllers directly use structural characteristics such as mass and stiffness in their formulation. Non-traditional controllers incorporate a wide variety of algorithms but, in general, are logic-based algorithms which incorporate weights and bias. These control strategies are generally not derived, but formulated through an iterative process and make use of error metrics.

Other methods such as modal control strategies (Cho *et al.*, 2005) have been recently applied to management of MR dampers in seismic scenarios. Here researchers seek to control specific modes of the structure that are prevalent in response to seismic excitation. For optimal control they assume linear elastic analysis of the structure. The developed controller utilizes displacement, velocity, and acceleration feedback for controller operation. A Kalman filter is usually employed to estimate these states. However, use of a Kalman filter is computationally expensive in real-time application of a control system. Although this control system has limitations, it does perform well for its intended purpose of controlling a set of selected modes. Alternatively, a non-traditional model control strategy has been developed by Rao and Datta (2006) with employment of an artificial neural network. The neural network is optimized to mitigate response of a few select modes instead of the total response of the structure. Results also show effective reductions to of building response.

Skyhook control has been employed by Nagarajaiah and Narasimhan (2003) for control of a numerical benchmark base-isolated structure that is augmented with MR dampers. Here assessments of both semi-active and active control schemes are pursued.

The skyhook controller is determined to be not quite as effective as a clipped optimal controller for the semi-active case.

Many structural engineering researchers have used widely known control strategies such as Lyapunov and energy based methods for semi-active control of MR dampers (Sahasrabudhe and Nagarajaiah, 2005; Yoshida and Dyke, 2005; Renzi and Serino, 2004, Dyke *et al.*, 1996). These controllers have been shown to be effective in numerical and physical testing. Each of these strategies relies on the accurate determination of structural characteristics such as mass and stiffness prior to controller formulation. These methods have been shown to be effective when highly accurate models of the structure can be attained. When accurate models cannot be attained the effectiveness of the controller often diminishes.

Prior efforts for optimization of fuzzy logic controllers have employed automated strategies such as neural networks and neuro-fuzzy optimization algorithms (Schurter and Roschke, 2001). In the cited study artificial neuro-networks are employed for modeling of MR damper behavior and control system development separately. Since the employed neuro-fuzzy optimization process (ANFIS) is limited to one output researchers were limited in the number of control devices they could include in the global control scheme. Schurter and Roschke (2001) optimized a single and multiple degree of freedom control system with superior results for the MDOF case. Researchers were also limited to the control capabilities of an LQR controller since LQR controller output is selected as a means to calculate controller error.

GA optimization for fuzzy logic controller optimization has been conducted by several groups of researchers. Ahlawat and Ramaswamy (2004a and 2004b) optimized controllers for wind and seismic excitations in separate studies with positive results from of each. Researchers were limited to numerical simulations in evaluating their GA optimized fuzzy controller. Researchers optimized the proposed control system considering only two objectives. In the cited study feedback of velocity and acceleration are used for building characterization. In a similar effort a semi-active control strategy has been proposed by Yan and Zhou (2004) which incorporates optimization of a FLC by a genetic algorithm. Here only one optimization objective is considered.

Recently Kim and Roschke (2006a and 2006b) further advanced GA-optimization of fuzzy logic controllers related to structural control applications. In each of these applications control of a hybrid base isolation system augmented by one or more MR dampers is pursued. Kim and Roschke (2006b) pursued concurrent minimization of four structural response objectives utilizing a non-dominated sorting genetic algorithm (NSGA-II) in each case. Favorable results are observed, especially when considering performance of traditional methods such as skyhook control.

Prior efforts to experimentally investigate the effectiveness of structural systems that incorporate MR damper technology have been primarily limited to small scale structures. However, one recent example of a large scale test utilized a decentralized control strategy to mitigate the seismic response of a four degree of freedom steel frame building (Renzi and Serino, 2004). Four MR dampers are attached to the structure using a bracing configuration that spanned two floors. Results show that MR dampers can be very effective in reducing both displacement and acceleration response of the structure to seismic excitations. Other researchers utilized a 24,000 kg single degree of freedom structure for large scale testing of a hybrid base isolation system (Kim *et al.*, 2006). A friction pendulum system augmented by a 20-kN MR damper is employed to effectively mitigate response of the structure to a suite of scaled earthquakes. Soda *et al.* (2003) also conducted a large scale experiment with linear roller bearings augmented by a 40 kN MR damper as a base isolation system for a three degree of freedom structure. Favorable reductions in seismic response of the structure were obtained. These three studies consist of the majority of large scale tests involving MR dampers in structural control applications that have been reported in the literature. Clearly further large-scale experimental investigations are needed to bring MR damper technology into full fruition as an effective control device.

1.4. GA System Identification

Past endeavors to use a GA as a means for system identification of civil engineering structures have entertained few large-scale experimental studies available for verification of accuracy and single objective optimization as exampled by Perry *et al.* (2006). In this study a GA algorithm is used to accurately identify only the acceleration response of a structure. Displacement and velocity are not predicted since they would be difficult to attain from a full-scale civil engineering structure. Few large scale verification studies have been attempted by prior researchers that employ this type of parametric identification. Other popular methods include frequency-domain system identification (Jin *et al.*, 2005) and subspace methods (Overschee and DeMoor, 1996) which utilize state-space formulations. The frequency-domain based methods offer rapid convergence towards a solution, but generally exhibit difficulties with data that have a high frequency component due to instrumentation noise (Perry *et al.*, 2006)

1.5. Summary

In the previous narrative concepts and results from existing literature have been discussed to aid the reader in understanding the currently proposed study. With discussion of current literature provided, focus now turns to the benchmark structure that is to be tested, modeled, and controlled in a series of large-scale laboratory tests.

2. OVERVIEW OF BENCHMARK STRUCTURE

2.1. General

In the following text the proposed benchmark structure and related equipment used in experimental testing are described. The test structure and equipment reside at the National Center for Research on Earthquake Engineering (NCREE) located in Taipei, Taiwan, and are shown in Fig. 8.

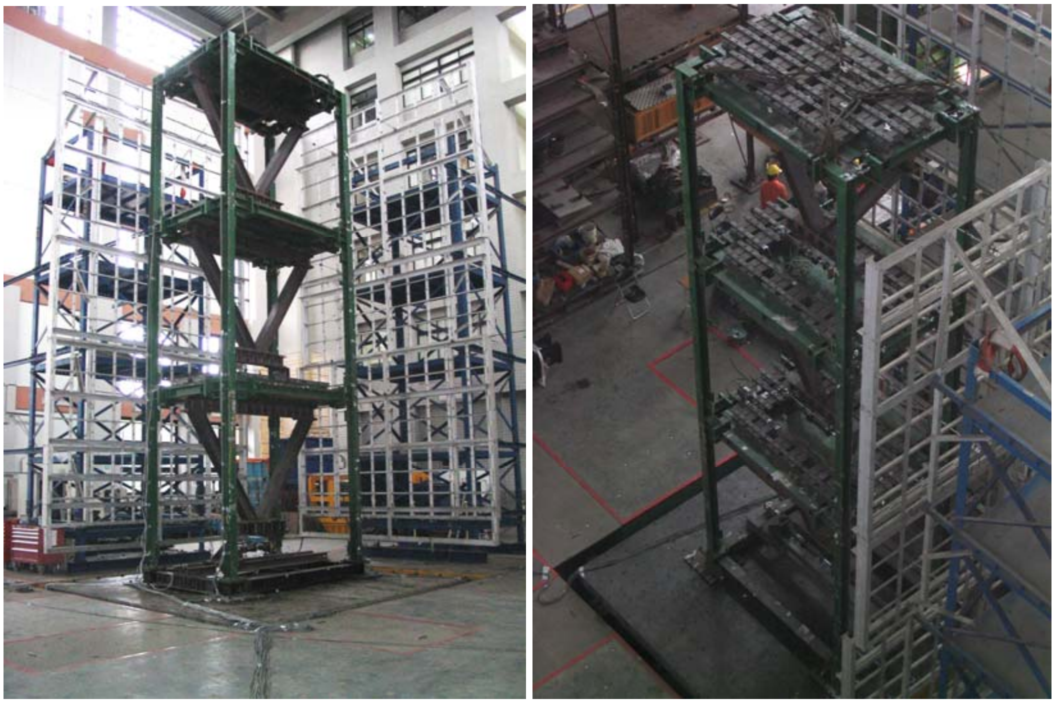


Fig. 8. Benchmark Structure

2.2. Test Structure

The benchmark structure is 9 m tall and has a total mass of approximately 18,440 kg. Details pertaining to the benchmark geometry and mass are provided in Table 2. All columns, beams, and braces are composed of H150×150×7×10 rolled shapes of grade A36 steel. Density of the steel is assumed to be 7,850 kg/m^3. Moreover, lead weights are placed on each floor to increase the mass to stiffness ratio. This is important since the benchmark structure should exhibit similar response characteristics as a real civil engineering structure. Since many tall structures have a fundamental frequency of approximately 1 Hz, the target frequency for the benchmark structure is similar. As described in Table 2 the approximated lumped mass of each floor includes all components attached to the benchmark structure such as floor beams, columns, floor plates, lead weights, and etc. It can be observed in Table 2 and in Fig. 8 that more lead weights are attached to the 3^{rd} floor than to the 1^{st} or 2^{nd} floors. Note also that some lead weights are removed from the 1^{st} and 2^{nd} floors to make room for the installation of MR dampers. This can also be observed in Table 2.

Table 2. Benchmark Structure Information

Parameter	Value
Floor Height	3 m
Floor Dimensions	2 m × 3 m
Column, Beam, and Chevron Size (A36)	H150×150×7×10
Estimated Lumped Floor Masses	
1st Floor	5,800 kg
2nd Floor	5,800 kg
3rd Floor	6,840 kg
Total	18,440 kg
Mass of One ‘Rack’ of Lead Weights	250 kg
Number of Lead Weight Racks Per Floor	
1^{st} Floor	10
2^{nd} Floor	10
3^{rd} Floor	14
Floor Plate Thickness	25 mm
Density of Steel	7,850 kg/m^3
Density of Lead	11,340 kg/m^3
Modulus of Elasticity of Steel	210 GPa
Yield Strength of Steel	250 MPa

The structure is idealized into three degrees of freedom as illustrated in Fig. 9. Since a state-space model of the structure is necessary for numerical simulation purposes, it must also be composed. Eqs. (1), (2), and (3) list the mass, stiffness, and damping coefficient matrices, while Eqs. (5), (6), and (7) are used describe the state-space model of the structure (Franklin *et al.*, 2002):

$$M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix} \tag{1}$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix} \tag{2}$$

$$C = \alpha M + \beta K \tag{3}$$

$$\alpha = 2\frac{\zeta \omega_1 \omega_2}{\omega_1 + \omega_2} \qquad \beta = 2\frac{\zeta}{\omega_1 + \omega_2} \tag{4}$$

$$\dot{x} = Ax + Bu \tag{5}$$

$$y = Cx + Du \tag{6}$$

$$\begin{gathered} A = \begin{bmatrix} 0_3 & I_3 \\ -M^{-1}K & -M^{-1}C \end{bmatrix} \qquad B = \begin{bmatrix} 0_3 \\ M^{-1} \end{bmatrix} \\ C = \begin{bmatrix} \mathrm{M^{-1}K} & \mathrm{M^{-1}C} \end{bmatrix} \qquad D = \begin{bmatrix} M^{-1} \end{bmatrix} \end{gathered} \tag{7}$$

where *M*, *K*, and *C* denotes the mass, stiffness, and damping coefficient matrices, respectively, *m* and *k* denote mass and stiffness, respectively, of an individual floor, *ω* is a fundamental frequency defined by an eigenvalue analysis, *ζ* is the values, *u* is a vector of pertebation inputs, and *A*, *B*, *C*, and *D* are state-space coefficient matrices.

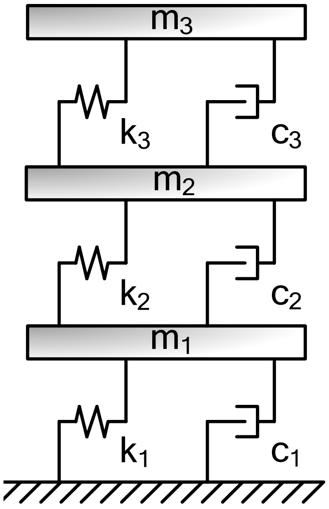


Fig. 9. Idealization of Benchmark Structure

As shown in Eqs. (3) and (4), Raleigh damping is employed for calculation of the damping coefficient matrix (Chopra, 2006). Identification of stiffness and damping parameters is pursued later in the text with the aid of a genetic algorithm. GA identified values are presented in Table 3 to provide a complete description of the state-space model in the current section of the manuscript. State-space modeling is a linear modeling technique and is utilized for numerical simulations to be conducted in MATLAB/Simulink (2006). As discussed later, these GA-optimized values do not hold a unique physical meaning since state-space formulations are non-unique. Yet, by employment of these values in the state-space model highly accurate predictions of the response of the benchmark structure are achieved.

Table 3. Identified Stiffness and Damping Values

Parameter	GA Identified Value
k_1	1,172.82 kN/m
k_2	1,750.02 kN/m
k_3	1,998.30 kN/m
ζ	0.00525

Identified mass and stiffness values are used to compute eigenvalues and eigenvectors as shown in Tables 4 and 5. Here it can be observed that the benchmark structure exhibits a fundamental frequency of approximately 1 Hz, thus the addition of lead weights to the structure is justified. Note that the eigenvectors have been normalized to unity. A sketch of the mode shapes is shown in Fig. 10.

Table 4. Benchmark Eigenvalues

	Fundamental Frequencies (Hz)
Mode 1	1.05
Mode 2	3.25
Mode 3	4.99

Table 5. Benchmark Eigenvectors

	Mode 1	Mode 2	Mode 3
Floor 3	-1	-0.682	-0.423
Floor 2	-0.851	0.290	1
Floor 1	-0.558	1	-0.631

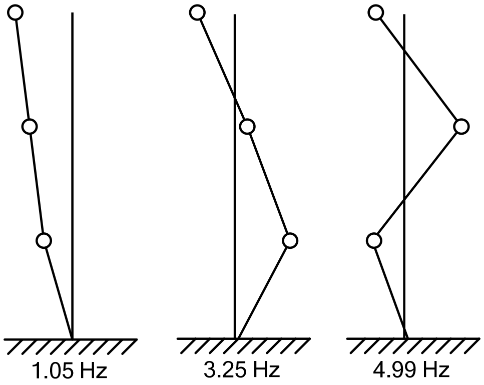


Fig. 10. Mode Shapes of Benchmark Structure

2.2.1. Experimental Setup

Tests conducted using the MTS seismic simulator at NCREE require use of the control room and experimental hardware as shown in Fig. 11(a). The control room is used by the NCREE staff to control the shake table and record all data collected from installed transducers. The FLC is implemented into a computer that manages the dSPACE data acquisition and control system shown in Fig. 11(b). The FLC specifies unique voltages to be applied to one or more MR dampers. The dSPACE hardware sends this voltage to a voltage controlled current source (VCCS) where the voltage is used to specify proportional amplitude of current to be applied to the MR dampers as shown in Fig. 12(a). The VCCS requires 24 V of power and can be operated using two automobile batteries connected in series. However, NCREE researchers choose to use a standard power supply as shown in Fig. 12(b). To measure the motion of the benchmark structure an array of transducers are installed on each floor of the structure. To monitor displacements, velocities, and accelerations of each floor LVDTs, velocity transducers, and accelerometers are installed, as shown in Fig. 13(a). To monitor the displacement experienced and force produced by the MR damper a LVDT and load cell are installed as shown in Fig. 13.

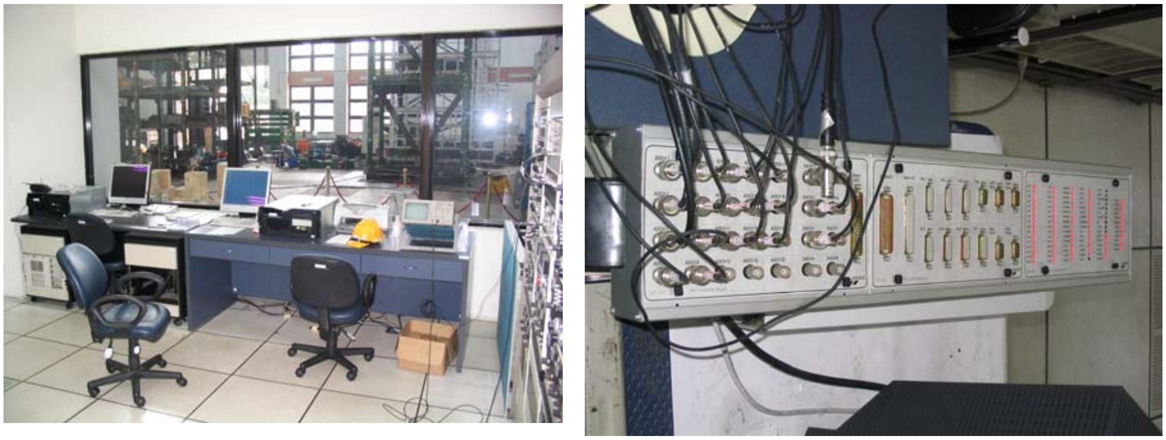

(a) (b)

Fig. 11. (a) NCREE Control Room and (b) dSPACE Hardware

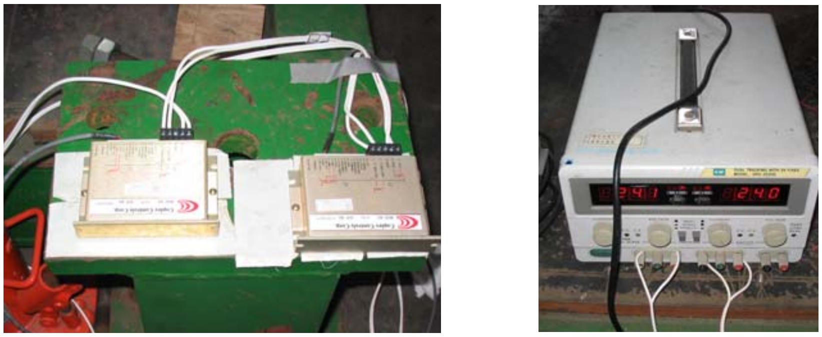

(a) (b)

Fig. 12. (a) VCCS and (b) Power Supply

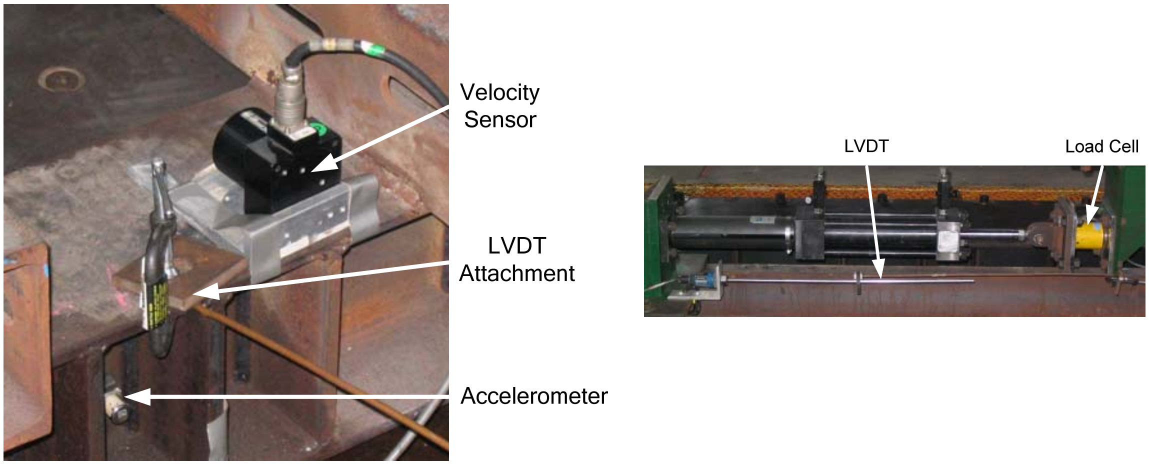


Fig. 13. Benchmark Transducer Photos

An experimental setup of the MIMO case is shown in Fig. 14. Individual components of the experiment are realized in this schematic diagram that describes their relationships. Accelerations measured on each floor by accelerometers are sent to the dSPACE hardware system where FLC computations are performed. A sketch relating inputs and outputs of a MIMO case FLC is rendered in Fig. 15 with employed units. Then the FLC-specified voltages are sent to the MR dampers via one or more VCCSs. Data from all transducers and voltages sent to the MR dampers are collected by the hardware at NCREE for a complete set of experimental results.

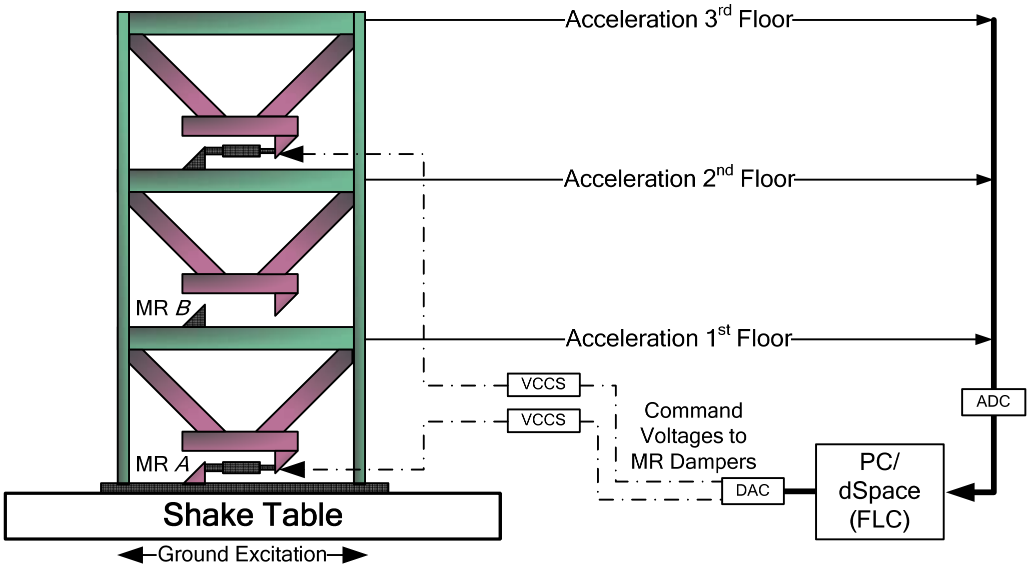


Fig. 14. Experimental Setup

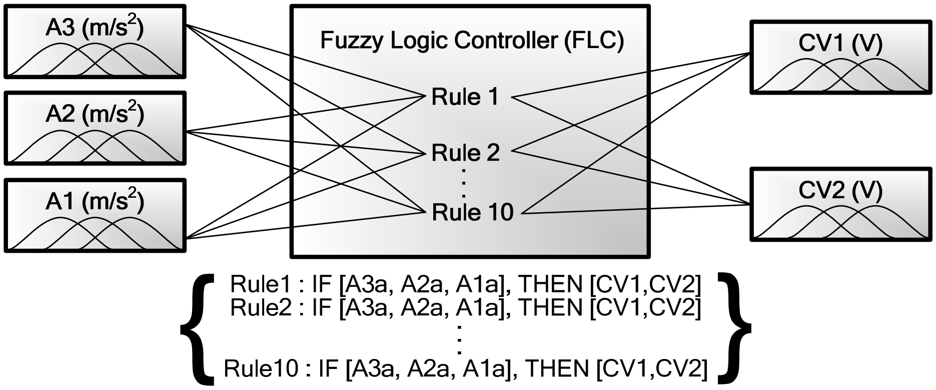
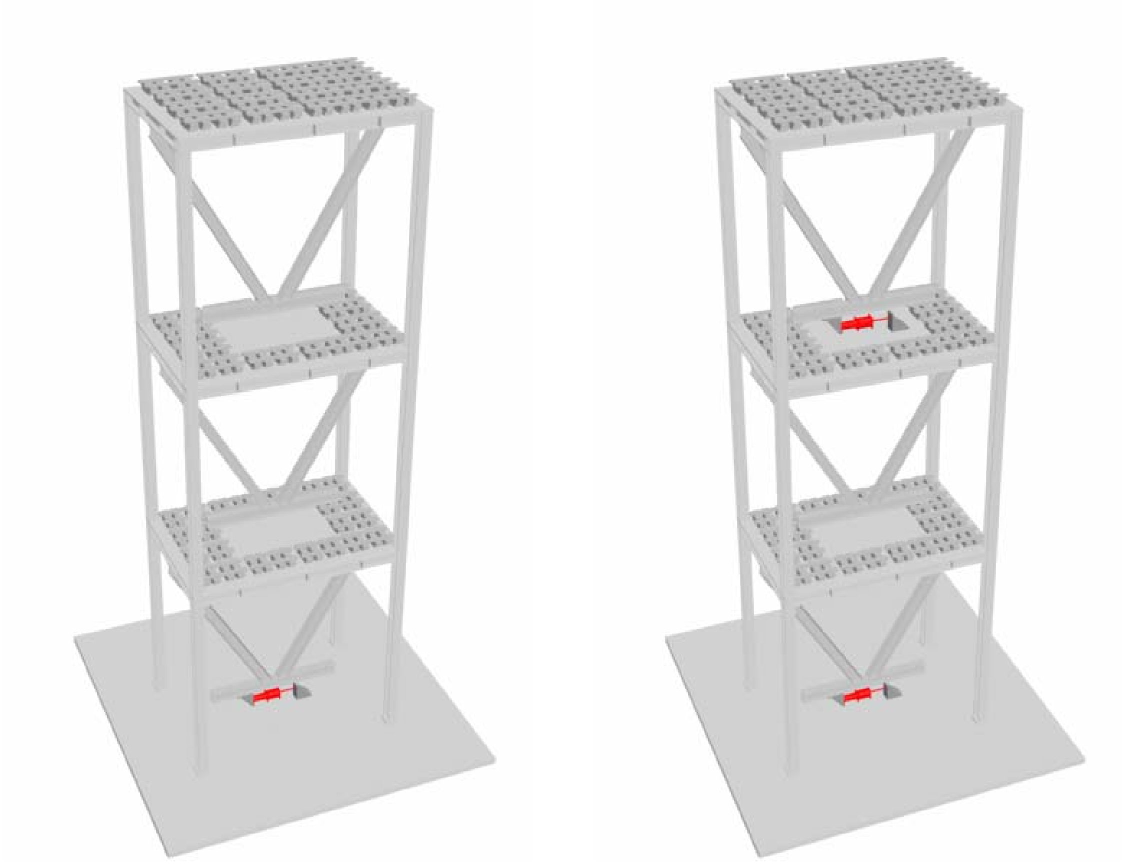

A3 (m/s2)
A2 (m/s2)
A1 (m/s2)
Fuzzy Logic Controller (FLC)
Rule 1
Rule 2
Rule 10
CV1 (V)
CV2 (V)
Rule1 : IF [A3a, A2a, A1a], THEN [CV1,CV2]
Rule2 : IF [A3a, A2a, A1a], THEN [CV1,CV2]
Rule10 : IF [A3a, A2a, A1a], THEN [CV1,CV2]

Fig. 15. FLC Sketch of MIMO Case

2.2.2. MR Damper Locations in Benchmark Structure

The benchmark structure contains three potential locations for MR damper installation. Two MR damper configurations are studied to show the adaptability of fuzzy logic control and genetic algorithm optimization. The first case involves multiple-input, single-output (MISO) control and the second case involves multiple-input, multiple-output (MIMO) control. In the MISO case MR damper *A* is attached to the inverted chevron brace between the ground and 1st floor. In the MIMO case MR damper *A* is attached to the inverted chevron brace between the ground and 1st floor and the MR damper *B* is attached to the inverted chevron brace between the 2nd and 3rd floors. Both MISO and MIMO cases are graphically rendered in Fig. 16 to aid in understanding their installations.



(a) (b)

Fig. 16. Rendering of MISO (a) and MIMO (b) Structures

2.3. Training Excitations

A variety of excitations need to be accounted for in the training process of the FLC. Artificial earthquake motions are created as described in what follows. Generation of this artificial earthquake was first proposed by Nagarajaiah and Narasimhan (2005). All parameters for this artificial earthquake are obtained from Kim and Roschke (2006b).

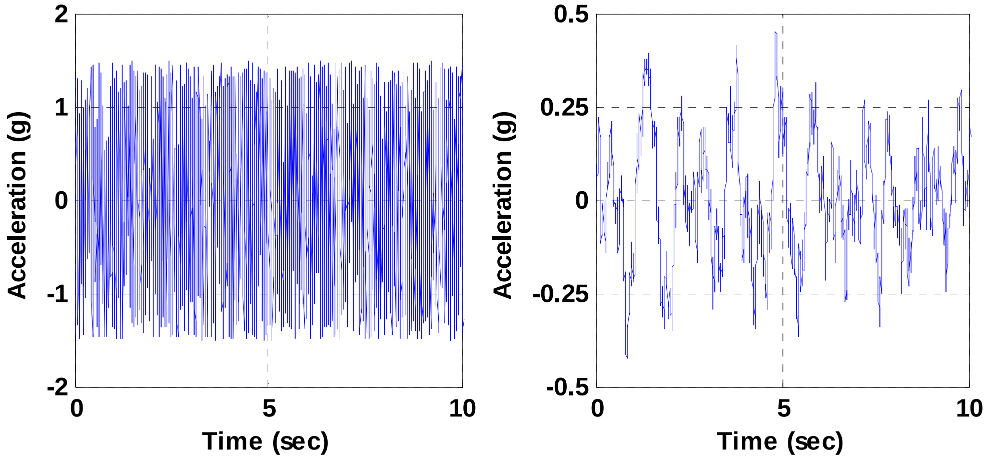
Generated artificial seismic excitations consist of amplitude and frequency content located in near-field seismic records. A set of near-field seismic records are compiled and studied to produce parameters required by a shaping filter. Near-field seismic records can be fundamentally differentiated from far-field seismic records by several metrics including their peak velocity (Chopra, 2006). Near-field excitation characteristics are used for training data since they exhibit higher velocities than far-field tremblers.

Creation of the artificial earthquake occurs in three stages. In the first step random data points are generated and passed through a shaping filter as follows:

$$F(s) = \frac{4\zeta_g \omega_g s}{s^2 + \zeta_g \omega_g s + w_g^2} \tag{8}$$

where s is the generated white noise in the time domain. Values for the shaping filter are ω_g = 2π radian/sec and ζ_g = 0.3. Here, frequencies with near-field characteristics are extracted from the random data. In Fig. 17, a 10 sec portion of the initial random data and post shaping filter data are shown. The total time span of the excitation is 30 sec.

Data retrieved from the shaping filter are processed through a shaping envelope, see Fig. 18. This is done to give a realistic growth and decay of excitation, as is typical for most seismic records. Exponential and logarithmic decay functions are used for time intervals from *a* to t_b, and t_c to *d*, respectively. Values for the shaping envelope are as follows: t_b = 7, t_c = 12, t_d = 30, and α = 0.3, (see Fig. 18).



(a) (b)

Fig. 17. (a) Initial Random Data, (b) Post-Shaping Filter Data

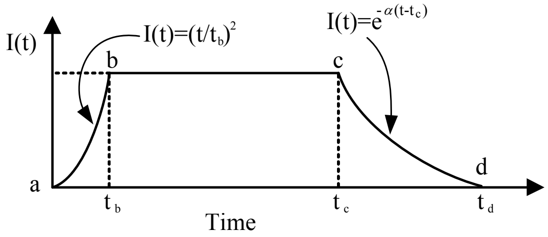
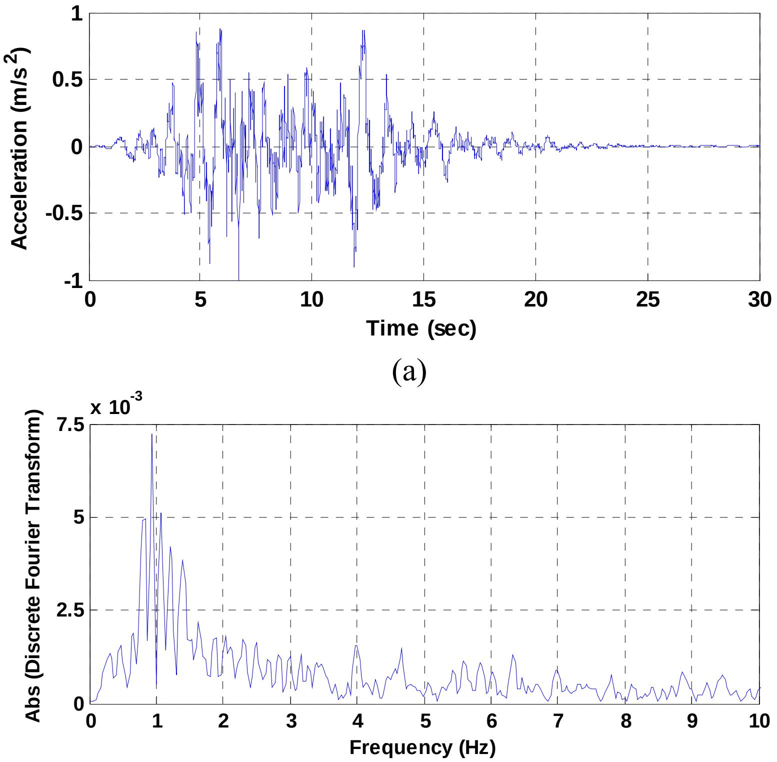


Fig. 18. Shaping Envelope

Due to inter-story drift limitations of 30 mm as mentioned earlier, the excitation is scaled to 100 gal to ensure that the structure remains linearly elastic during experimental testing. Fig. 19 displays the excitation created by the artificial earthquake generator with respect to time and frequency content. Notice that the majority of excitation is near 1 Hz.



(a)

(b)

Fig. 19. Artificial Seismic Base Excitation (a) Time History and (b) FFT

2.4. Seismic Records

In addition to the artificial record, four recorded ground motions are used for a variety of applications in the current study. Acceleration time histories of these recorded events are shownin Fig. 20. Selected excitations consist of a variety of near- and far-field seismic events. Records include excitations from the El Centro (1940), Kobe (1995), and Chi-Chi (1999) earthquakes. Two stations are selected from the Chi-Chi earthquake: TCU076 and TCU082. The amplitude of the excitations are limited such that the materials of the benchmark structure remain linear and in the elastic range. It has been determined through experimentation that inter-story drifts beyond 30 mm incite material yielding of the steel in the columns. Therefore, for the uncontrolled case the peak acceleration of most temblors must be scaled to no more than 100 gal. With the addition of two MR dampers excitations can be scaled to as high as 300 gal depending on the frequency content of the seismic record.

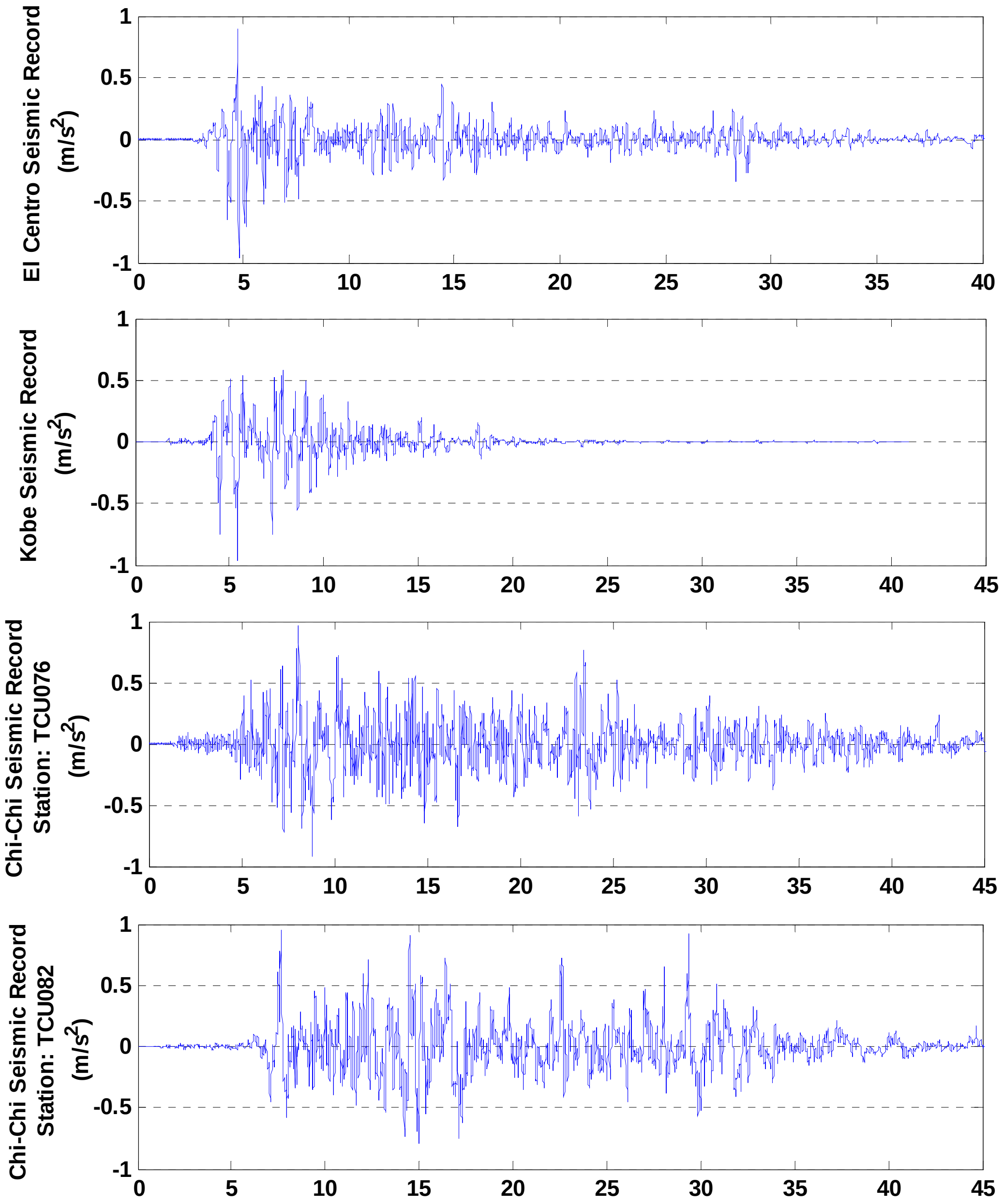


Fig. 20. Recorded Ground Motion Records (100 gal)

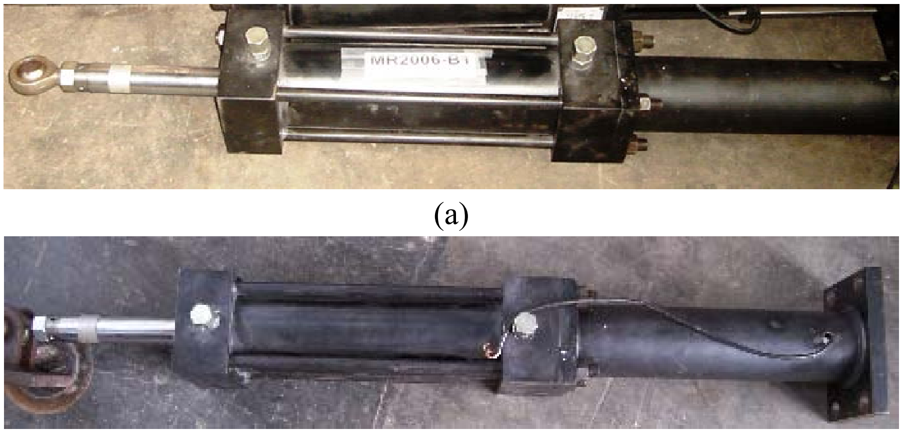
2.5. Summary

A test structure has been identified for implementation of a structural controller in numerical simulation and experimental testing. Furthermore, an artificial excitation has been established for training purposes with the GA optimization. Now generation of neuro-fuzzy models of MR dampers *A* and *B* are discussed.

3. MAGNETORHEOLOGICAL DAMPERS

3.1. General

MR dampers provide a significant amount of controllability for a structural engineer. Their controllability is derived from rheological properties exhibited by iron particles suspended in an oil-based MR fluid when a magnetic field is applied to the damper. The magnetic field is the result of current being applied to the coil. Amplitude of the current is specified by a voltage originating from a dSPACE data acquisition and control system. Thus, the terms voltage and current are used interchangeably in what follows. In the next subsections, the rationale for employment of neuro-fuzzy modeling techniques and details pertaining to experimental characterization tests of MR dampers *A* and *B* are discussed. MR dampers *A* and *B* are each approximately 1 m in length (see Fig. 21).



(a)

(b)

Fig. 21. MR Dampers (a) *A* and (b) *B*

3.2. Fuzzy Logic for MR Damper Modeling

Fuzzy logic is a non-analytical method of modeling nonlinear behavior. A fuzzy inference system (FIS) is composed of a set of rules that contain input and output relationships. For neuro-fuzzy modeling Sugeno-type FISs are used where membership functions compose the antecedent portion and polynomial equations compose the subsequent portion (see Fig. 22).

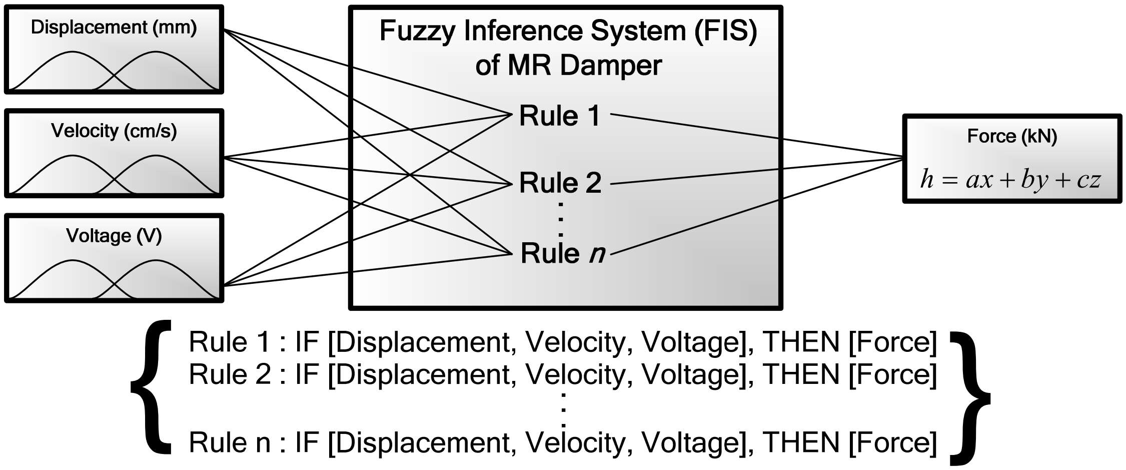

Displacement (mm)
Velocity (cm/s)
Voltage (V)
Fuzzy Inference System (FIS) of MR Damper
Rule 1
Rule 2
Rule n
Force (kN)
$h = ax + by + cz$
Rule 1 : IF [Displacement, Velocity, Voltage], THEN [Force]
Rule 2 : IF [Displacement, Velocity, Voltage], THEN [Force]
Rule n : IF [Displacement, Velocity, Voltage], THEN [Force]

Fig. 22. Fuzzy Inference System for MR Damper

For this study generalized bell-shaped membership functions are used as inputs and a polynomial is used for the output. To model damper behavior inputs must be selected that yield accurate MR damper behavior. Displacement is important since casing and piston alignment are typically imperfect which affects damper resistance. Velocity is also used since resistance of the piston head through the viscous MR liquid is best described by velocity. Voltage is an important variable since it can drastically change the effective density of the MR fluid. Acceleration is not used since ample information is provided by displacement, velocity, and voltage data.

3.2.1. Advantages to Fuzzy Logic Modeling of MR Dampers

MR dampers have been employed as semi-active variable viscous dampers for a number of structural engineering research studies. A common problem associated with MR dampers is their non-linear response to motion, especially when current is applied to the coil. To model the non-linear nature of MR dampers researchers have derived accurate, but computationally expensive differential equations (Spencer *et al.*, 1997). This differs from fuzzy logic which requires almost negligible computational time. To contrast this computational difference Fig. 23 shows the computational processing time required for a phenomenological model of a SD-1000 MR damper proposed by Spencer *et al.* (1997) and a fuzzy logic model as proposed by Schurter and Roschke (2000). Schurter and Roschke (2000) discovered that the ratio of the computational time between the phenomenological model and fuzzy model is approximately 150:1. Since phenomenological models are computationally taxing it is advantageous to use the fuzzy logic model, especially since thousands of numerical simulations are required for a single GA optimization used later in development of a FLC. Furthermore, MR damper *A* is a two stage model and no phenomenological model has been proposed with such a configuration.

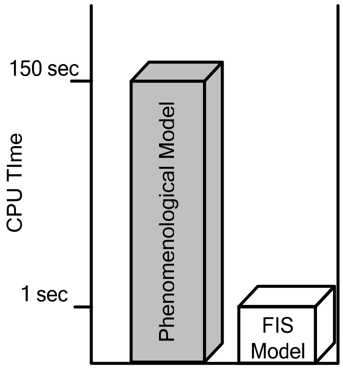


Fig. 23. Comparison of CPU Time for MR Damper Models

Moreover, fuzzy logic has an inherent component of robustness that allows the numerical model to account for some uncertainties. Although accuracy of the FIS is still very dependent on the training data, FISs are more reliable than typical analytical models when motions occur that are not accounted for in training. This robustness is primarily derived from the continuous membership and linear functions that fully describe the inputs and outputs, respectively.

Despite the advantages of using FIS numerical models, an important disadvantage must be considered. In general, FISs are not quite as accurate as these other modeling techniques. That is, there is a compromise between accuracy and computational speed for numerical modeling methods. FISs reside in the faster yet modestly less precise spectrum of numerical models. The reduction in precision is not considered to be a problem since the FIS has been shown to accurately mimic experimental results. Thus, a good compromise between computational time and numerical precision is ascertained.

3.2.2. Design of MR Damper

MR dampers *A* and *B* are both two-stage MR dampers as demonstrated in Fig. 24. Two coils rather that one are installed around each piston head to increase the resistance of the damper. Unfortunately, due to a manufacturing mistake the coils are wound in the same direction and thus the magnetic fieldes cancel each other in the region between the two coils. If they had been wound in opposite directions, even more resistance could be achieved. Also, it should be noted that for MR damper *A* one of the two coils is believed to be malfunctioning. As shown later (see pages 38 and 39), this alters the resistance capacity of the damper; however, this malfunction is not fatal to its operation. Displacement and voltage ranges for MR dampers *A* and *B* are listed in Table 6.

Table 6. MR Damper Characteristics

Characteristic	Range
Displacement	± 140 mm
Voltage	0 – 1.2 Volts

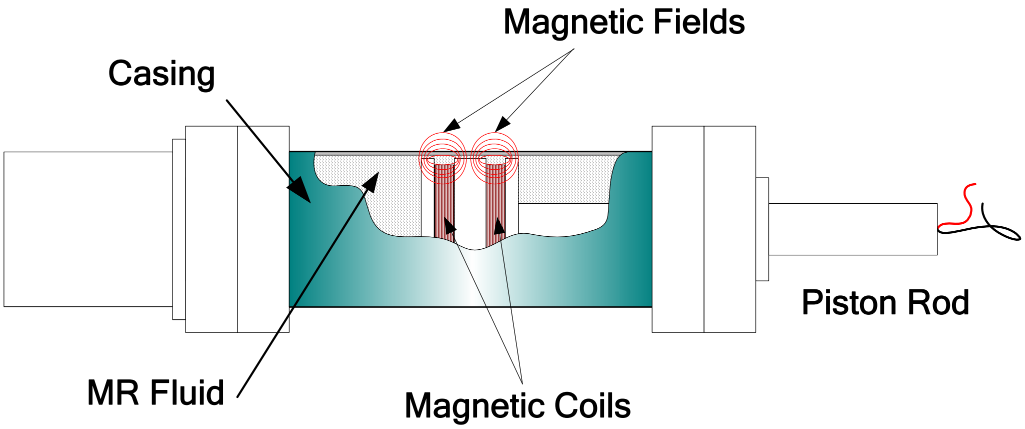


Fig. 24. Schematic of MR Dampers *A* and *B*

3.3. Experimental Characterization Tests

In preparation for installation in the benchmark structures, a series of characterization tests are performed on MR dampers *A* and *B* (see Fig. 21). The objective of the characterization tests is to experimentally quantify the response of each damper to a variety of motions and applied currents.

Performance tests of each MR damper are conducted by NCREE researchers who provided data that are used to produce a fuzzy model of each device. Acquiring robust and thorough data from experimental tests is important when preparing training data. Prior approaches by Schurter and Roschke (2000) employed a philosophy of acquiring data beyond the operational use of the damper. Thus, if a maximum velocity of 30 cm/s is expected of the damper in operation, then a maximum velocity range of 45 cm/s would be pursued in performance testing. This approach is helpful and can aid in producing a robust FIS. Yet, care must be taken during the training of the FIS to ensure that the fuzzy rules are accurate in these higher ranges of operation where training data may be sparse.

ANFIS is employed as the neuro-fuzzy optimization algorithm to generate fuzzy models of the MR dampers. ANFIS gives consideration to every data point provided in the training data. Thus, regions of data with relatively few data points can yield unreliable results in the neuro-fuzzy model. For these reasons a different approach to generation of training data is employed for this study. As previously mentioned, experimental testing should provide data well beyond the expected operational ranges of the damper, and the data should be plentiful inside these operational ranges to ensure a reliable set of training data for the FIS. The expected operational ranges for these dampers during testing of the benchmark structure are ±35 mm for displacement and ±30 cm/s for velocity. Thus, training data are not required to extend beyond these ranges, but should be plentiful over the entire ranges. Testing data were provided by NCREE researchers and did not include all ranges of operation.

Two random displacement time histories termed ‘RD1’ and ‘RD2’ are used for performance testing. RD1 and RD2 are tested with passive voltages ranging from passive-off (0 V) to passive-on (1.2 V). Voltage is increased in increments of 0.2 V for a total of 14 damper tests for each MR damper. Although this is not an ideal composition of performance tests, they are the only data made available by NCREE researchers. Ideally a suite of random and sinusoidal displacements test would have been conducted that could fully explore the operational ranges of the MR dampers. Furthermore, random and passive voltage signals would be included to completely investigate the operational ranges of the dampers.

Time history plots of RD1 and RD2 are shown in Figs. 25 and 26. Velocity time histories are calculated from the displacement time histories by the 4th order backward difference formula as follows:

$$\dot{x} = \frac{3x_{i-4} - 16x_{i-3} + 36x_{i-2} - 48x_{i-1} + 25x_i}{12\Delta t} \tag{9}$$

where x and $\dot{x}$ denote displacement and velocity, respectively, and Δ_t denotes the time step. A 4th order backward difference approach is used since the data collected contain a modest amount of noise as is common with all data acquisition systems. If a lower order difference equation had been used the computed velocity data would be inaccurate. No filtering is required.

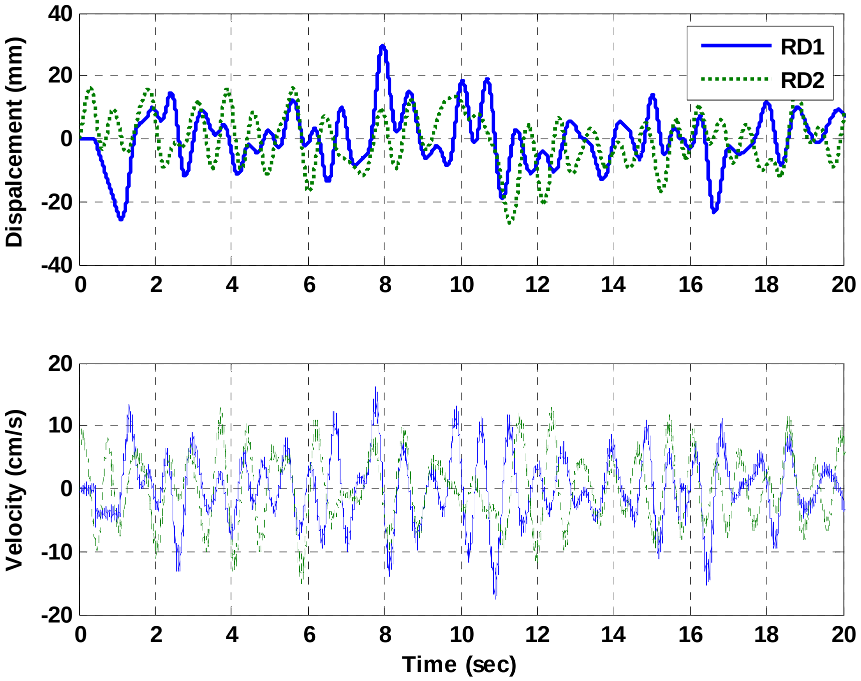


Fig. 25. RD1 and RD2 Time Histories

As shown in Fig. 26 an even distribution of points at or beyond the operational range of the MR dampers is not achieved. This lack of data has a negative affect on subsequent accuracy of the fuzzy models. Yet in spite of this difficulty the ANFIS algorithm produces sufficiently accurate models of both MR dampers.

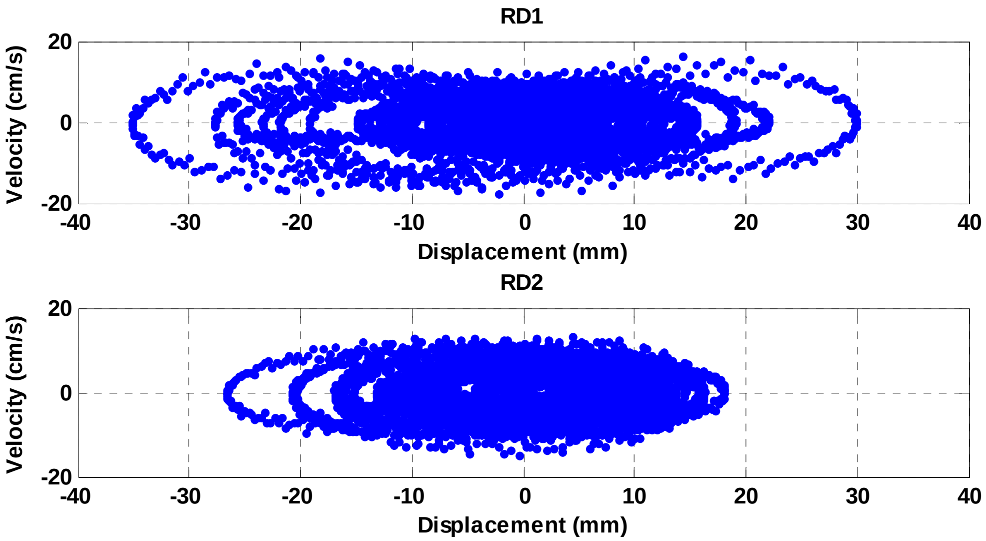


Fig. 26. Displacement vs. Velocity for RD1 and RD2

Figs. 27 and 28 show the controllability of both MR dampers. Controllability is judged by observing the passive-off (0 V) and passive-on (1.2 V) resistances of the MR damper to excitation. Clearly, the resisting force of the passive-on case is significantly higher than for the passive-off. This means that a significant variation of resistance can be achieved by modulating the voltage to each MR damper. This is one of the favorable characteristics of an MR damper.

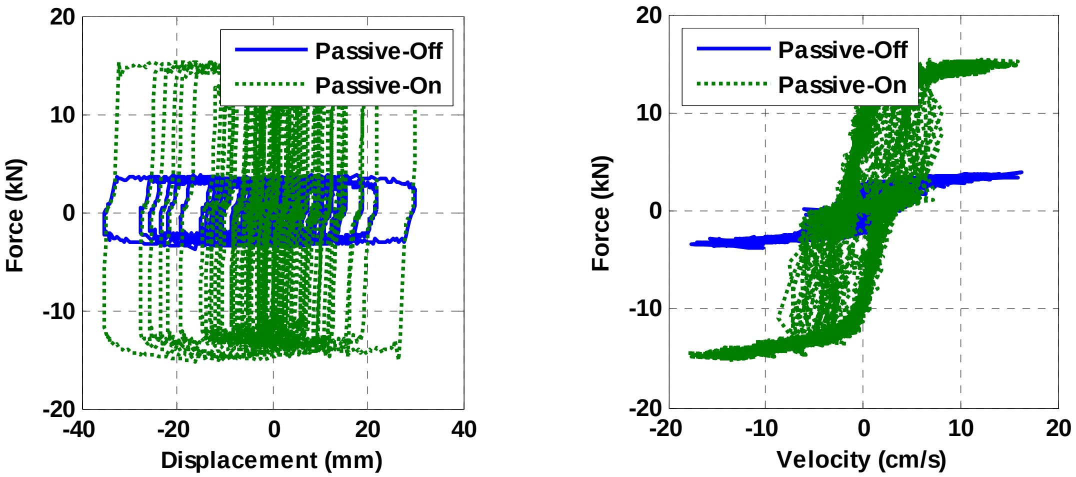


Fig. 27. Hysteretic Behavior of MR Damper *A* Over RD1

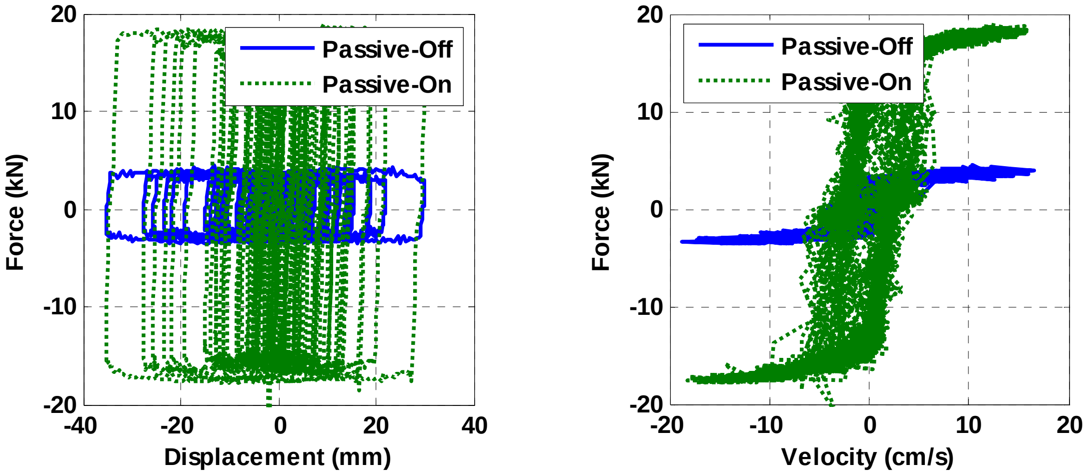


Fig. 28. Hysteretic Behavior of MR Damper *B* Over RD1

3.4. Fuzzy Modeling of a MR Damper

In the following text information concerning fuzzy models of both MR dampers is reported. Neuro-fuzzy optimization is a hybridization of fuzzy and neural network theory in which Sugeno-type fuzzy inference systems are used. In this study an Adaptive Neuro-Fuzzy Inference System (ANFIS) algorithm is utilized for parametric identification of the Sugeno-type FIS. Key factors are discussed below that are critical in the development of fuzzy models.

3.4.1. Selection of Training Data

The selection of training data is crucial in producing an accurate fuzzy model of a MR damper. ANFIS, through an iterative learning process, attempts to optimize fuzzy membership functions and the fuzzy rule base to accurately model provided output data. Thus, accuracy of the ANFIS generated fuzzy model is primarily dependent on the training data with which it is presented. ANFIS requires both input and output data to build a numerical model; thus, input and output parameters must be selected. As discussed in Section 3.2, displacement, velocity, and applied voltage are selected as inputs. The output is the resistance force exerted by the damper. For this study two sets of performance tests have been conducted corresponding to data sets RD1 and RD2. Each data set has been evaluated over a suite of passive voltages ranging from 0 to 1.2 volts. RD1 is used to form ANFIS training data since it exhibits the best distribution and greatest range of data points. The initial training data set used for neuro-fuzzy modeling of MR dampers *A* and *B* are shown in Figs. 29 and 30. Since these training data sets are created by concatenating several tests, Figs. 31(a) and 31(b) should be observed to see how evenly distributed the data points are in relation to displacement and velocity.

Note that numerous sparse regions of data can be seen in Figs. 31(a) and 31(b). In these regions the ANFIS algorithm is required to guess and interpolate what the data should be since fuzzy logic is composed of continuous functions of the entire input and output domains. This interpolation often results in noticeable error. To help increase accuracy of the FIS over a reasonable region of displacements, the original training data are modified or 'trimmed.' These new training data sets are voided of any data points that contain displacement magnitudes greater than ± 25 mm as shown in Figs. 31(a) and 31(b) as denoted by the shaded regions. Although significantly more regions could be voided, restraint is exercised to maximize the operational range of the FIS.

FIS rules are commonly described by graphical means. A three-dimensional surface where two inputs and one output can be visually inspected simultaneously is the most common representation. This fuzzy surface is often used to verify if the FIS is realistic by expert knowledge. Inspection of the FIS surface can yield important discoveries. For example, two fuzzy surfaces are shown in Fig. 32. Fig. 32(a) is a fuzzy surface trained on all data shown in Fig. 31 which contains sparse displacement data points that have magnitudes greater than ±25 mm. Note the significant gradient in the surface near the 30 mm displacement and -20 cm/s velocity coordinates. This directly corresponds to the lack of data located in this region as shown in Fig. 31**.** Fig. 32(b) is a fuzzy surface trained on the same data, but with sparse data points beyond the ± 25 mm limit removed. That is, by removing data points that lay outside of the desired range the large gradient has been corrected. Modest alterations in the training data have significant effects on the FIS surface if regions of sparsely populated data exist.

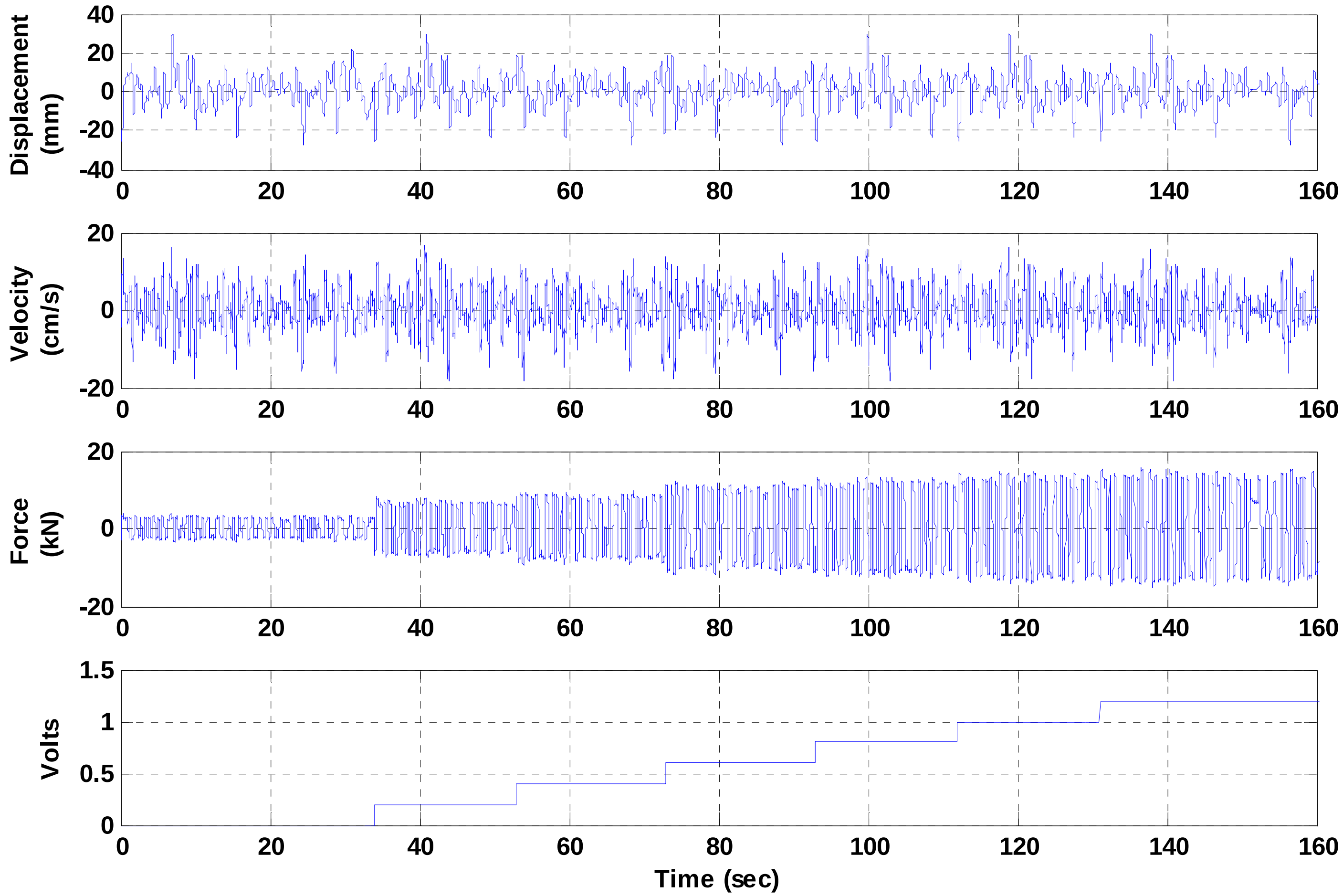


Fig. 29. MR Damper A Training Data

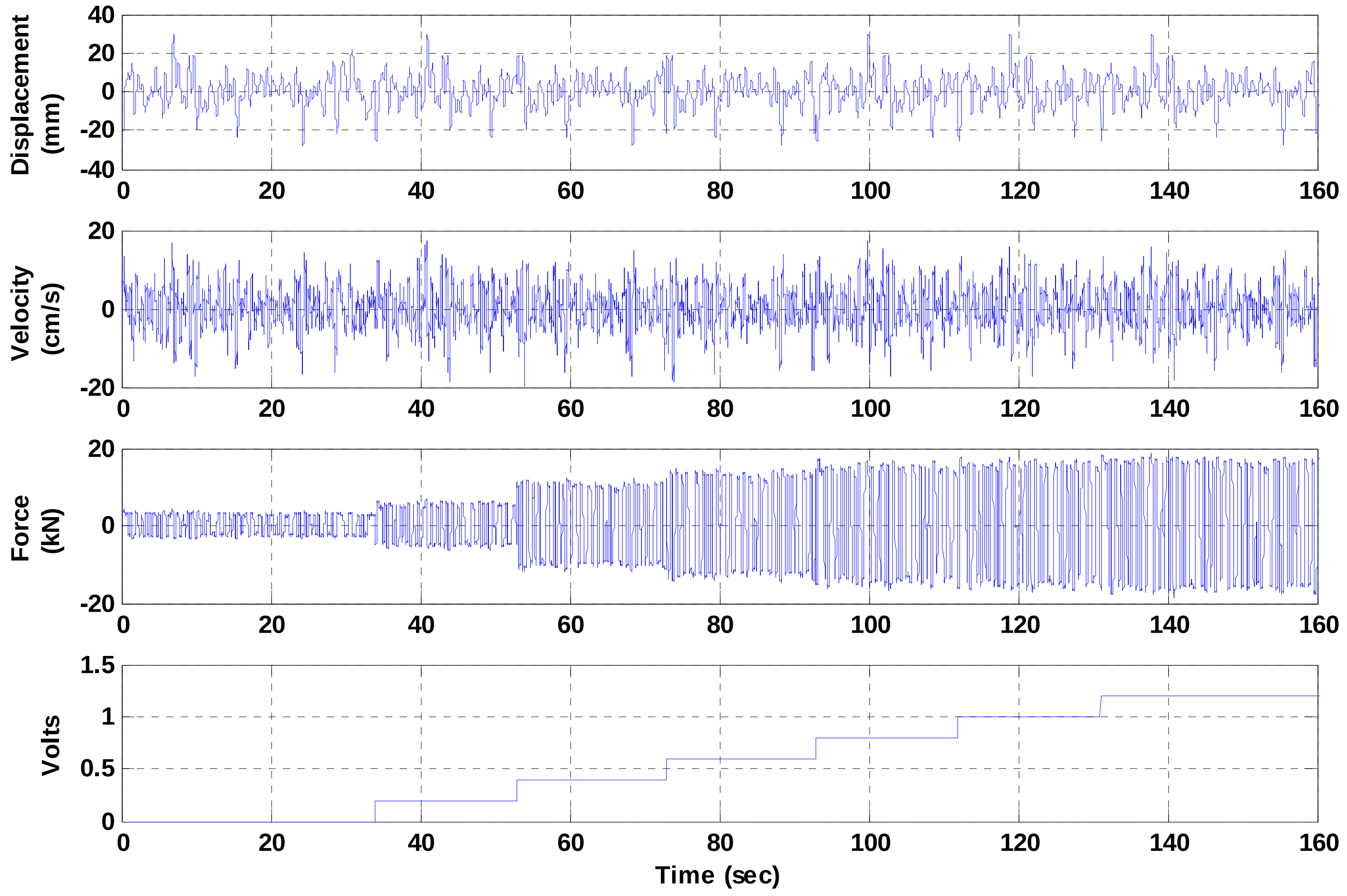
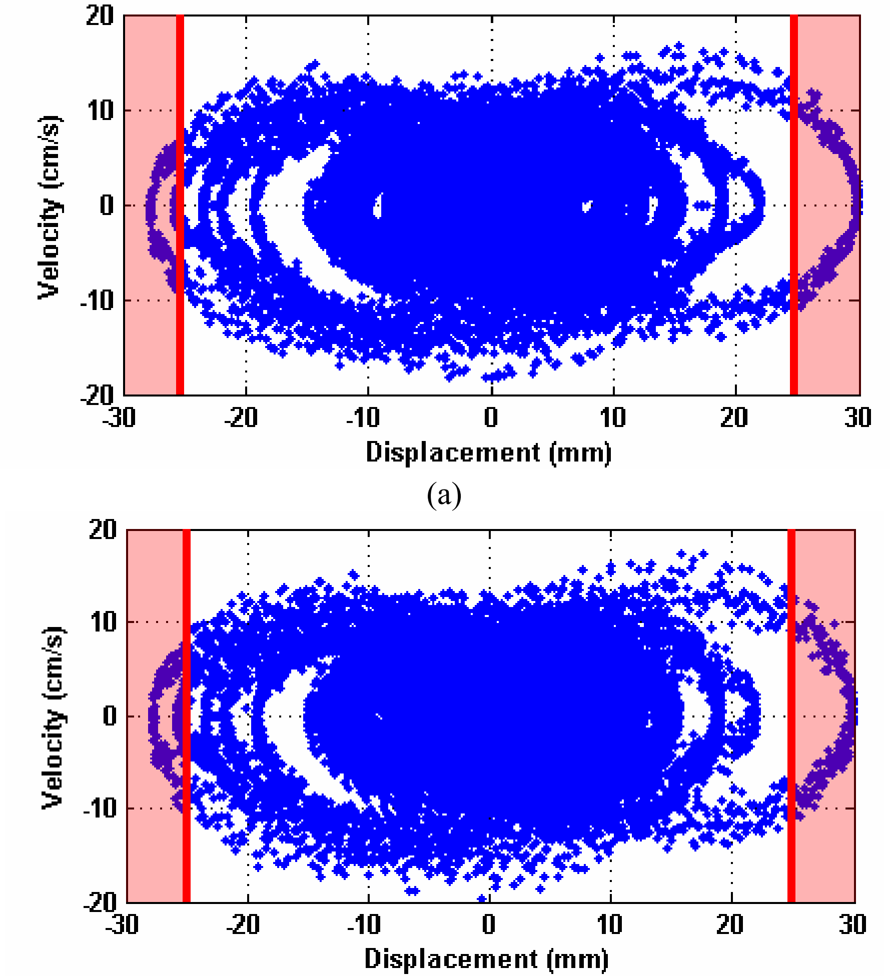


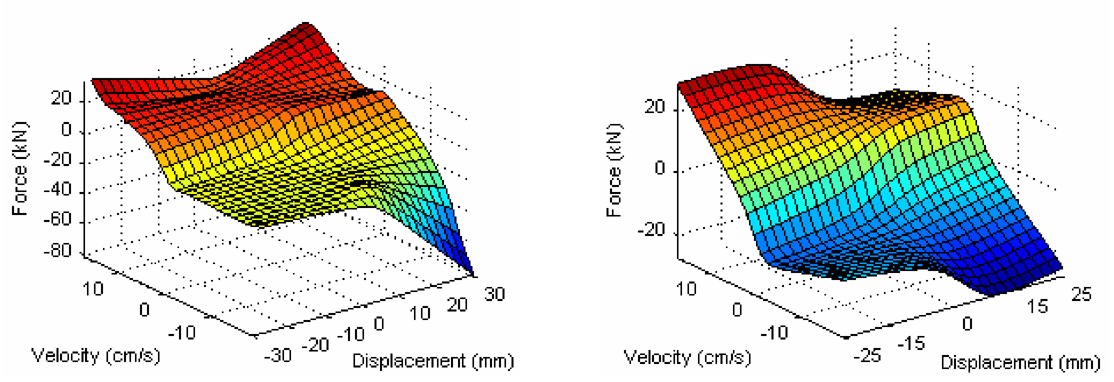
Fig. 30. MR Damper B Training Data



(a)

(b)

Fig. 31. Trimmed (a) *A* and (b) *B* Training Data



(a)

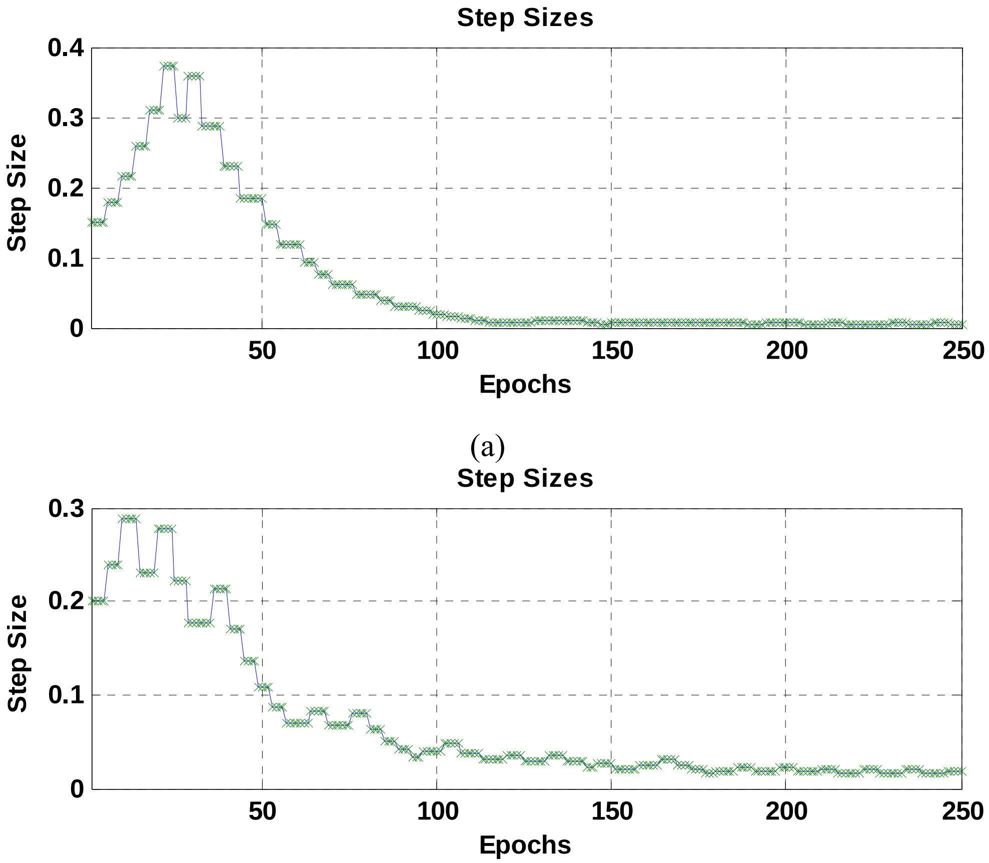
(b)

Fig. 32. Resulting Fuzzy Surfaces (a) With and (b) Without Sparse Regions of Data

3.4.2. Optimal ANIFS Training Parameters

ANFIS uses both training data and user defined parameters in optimization procedures. User defined parameters include: number of membership functions per input, membership function type, step size, increasing and decreasing step size, and the number of epochs used in training. For this study only generalized bell-shaped membership functions are used since they have shown through trial and error to be the most adaptive method for use with experimental data that often contains unwanted high frequency content (i.e. noise). Other types of membership functions, such as triangular, often yield poor results when using laboratory data. Encoding of the ANFIS algorithm is conducted in MATLAB/Simulink (2006) and is provided in Appendix A.

Step size is an important tuning parameter for ANFIS optimization. The step size is a value that determines the magnitude of alteration of fuzzy parameters at each epoch. An epoch is a set of iterations. Inside a single epoch all alterations are determined by the presiding step size. After several iterations the step size is decreased or increased depending on the error calculated by ANFIS. MATLAB/Simulink has specified defaults of 0.01, 0.9, and 1.1 for initial, decreasing, and increasing step sizes, respectively. In the computations that follow, the initial step size is specified to be 0.15 for ANFIS training of MR damper *A* and 0.2 for ANFIS training of MR damper *B*. Generally, a slight increase in step size during early epochs followed by a gradual reduction is ideal over the series of epochs. This trend is observed in ANFIS training results of both MR dampers as shown in Fig. 33. Here execution of ANFIS yields an early increase in step size that is followed by a gradual reduction to almost zero.



(a)

(b)

Fig. 33. Step Size Alterations of ANFIS Training for MR Damper (a) *A* and (b) *B*

During ANFIS training membership functions are shifted and re-sized to optimize the model for accurate prediction of desired outputs. Thus, it is important to observe the location and width of membership functions before and after training, as shown in Figs. 34 and 35. Significant shifts can be seen in all three input sets of membership functions, especially for voltage.

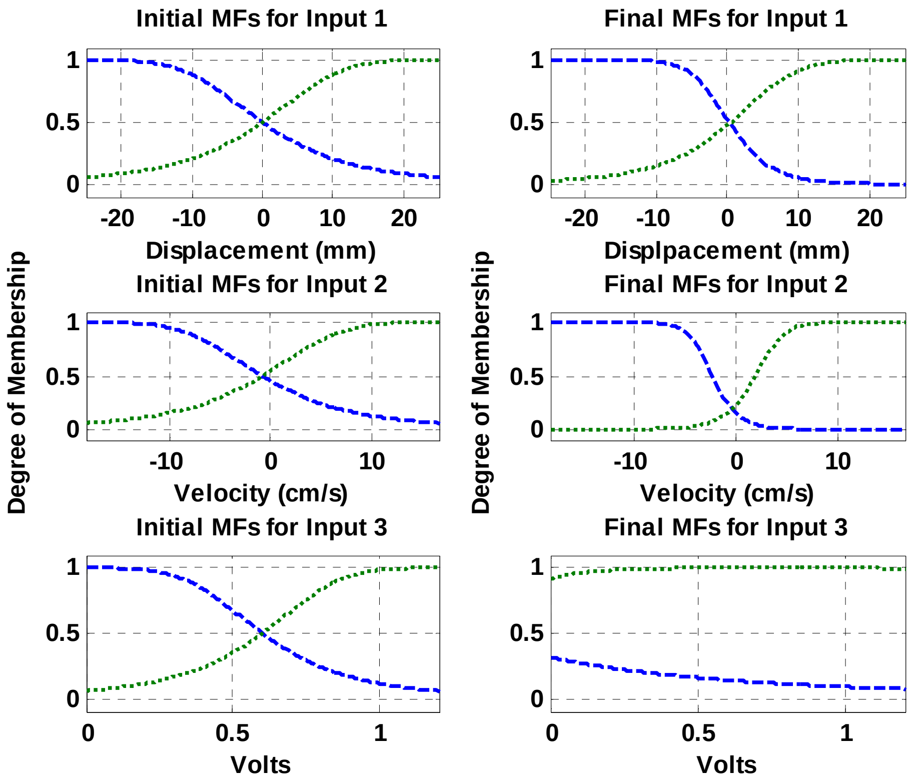


Fig. 34. Membership Functions Before/After Training: MR Damper *A*

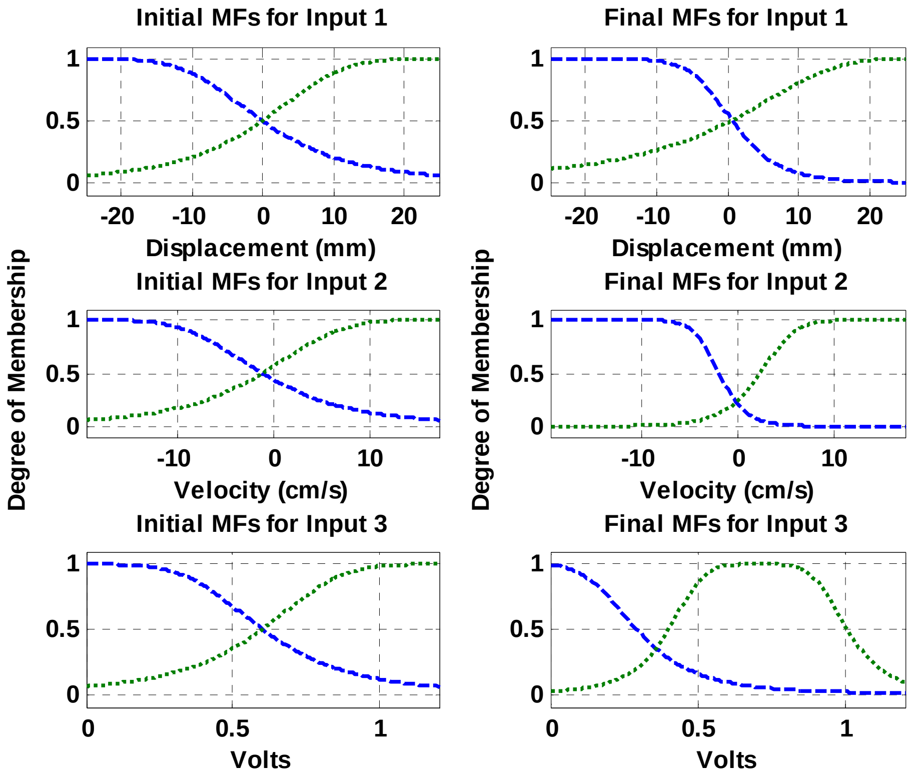


Fig. 35. Membership Functions Before/After Training: MR Damper *B*

Two membership functions are used for each input variable. This is not the ideal composition of membership functions for neuro-fuzzy modeling of a MR damper. Ideally, three or four membership functions would be used for velocity and or voltage inputs as shown by Schurter and Roschke (2000). With the addition of more membership functions the potential accuracy of the neuro-fuzzy model is increased. These additional membership functions allow the neuro-fuzzy model to be more closely fitted to the training data. However, since the training data in this application contains significant regions of sparse data points a lower accuracy model is chosen so that the neuro-fuzzy algorithm does not produce FISs similar to the one shown in Fig. 32(a).

3.4.3. Validation of FIS for MR Damper

The primary source for validation of the fuzzy models is through comparison with experimental data collected during characterization tests, but not used for training. To quantify the accuracy of the fuzzy models a modified set of metrics have been employed that were initially proposed by Spencer *et al.* (1997). These metrics relate error of fuzzy models to time, displacement, and velocity. They are defined as follows:

$$E_t = \frac{\varepsilon_t}{\sigma_F} \ , \ E_x = \frac{\varepsilon_x}{\sigma_F} \ , \ E_{\dot{x}} = \frac{\varepsilon_{\dot{x}}}{\sigma_F} \tag{10}$$

$$\varepsilon_t^2 = \int_0^T (F_{\mathrm{exp.}} - F_{FIS})^2 \, dt \tag{11}$$

$$\varepsilon_x^2 = \int_0^T (F_{\mathrm{exp.}} - F_{FIS})^2 \left| x \right| dt \tag{12}$$

$$\varepsilon_{\dot{x}}^2 = \int_0^T (F_{exp.} - F_{FIS})^2 \left| \dot{x} \right| dt \tag{13}$$

$$\sigma_F^2 = \int_0^T (F_{exp.})^2 \, dt \tag{14}$$

where $F_{exp.}$ denotes measured experimental force, F_{FIS} denotes fuzzy prediction of force, x is displacement of the damper piston, and t denotes time. Performance of the fuzzy models according to the above metrics for sets of validation data is tabulated in Table 7.

Table 7. Performance of Fuzzy Models of MR Dampers *A* and *B*

	Fuzzy Model of MR *A*			Fuzzy Model of MR *B*		
	E_t	E_x	$E_{\dot{x}}$	E_t	E_x	$E_{\dot{x}}$
RD1 (0 Volts)	0.039	0.092	0.057	0.024	0.057	0.033
RD1 (0.2 Volts)	0.090	0.203	0.129	0.046	0.107	0.053
RD1 (0.4 Volts)	0.109	0.248	0.162	0.077	0.180	0.101
RD1 (0.6 Volts)	0.141	0.325	0.205	0.087	0.204	0.118
RD1 (0.8 Volts)	0.149	0.339	0.218	0.099	0.220	0.135
RD1 (1.0 Volts)	0.173	0.382	0.254	0.108	0.243	0.147
RD1 (1.2 Volts)	0.100	0.214	0.146	0.118	0.264	0.157
Avg. Error	**0.114**	**0.258**	**0.167**	**0.080**	**0.182**	**0.106**
RD2 (0 Volts)	0.081	0.201	0.121	0.115	0.288	0.256
RD2 (0.2 Volts)	1.053	2.478	1.578	0.233	0.585	0.447
RD2 (0.4 Volts)	0.426	1.013	0.662	0.663	1.550	0.949
RD2 (0.6 Volts)	0.230	0.540	0.348	0.392	0.915	0.568
RD2 (0.8 Volts)	0.231	0.521	0.347	0.435	1.022	0.608
RD2 (1.0 Volts)	0.164	0.372	0.246	0.286	0.672	0.392
RD2 (1.2 Volts)	0.155	0.356	0.231	0.222	0.528	0.295
Avg. Error	**0.334**	**0.783**	**0.505**	**0.335**	**0.794**	**0.502**

Generally these results show that the fuzzy models are not as accurate as the Bouc-Wen model proposed by Spencer *et al.* (1997). As discussed above, this is an acceptable margin of error since the Spencer *et al.* (1997) Bouc-Wen model is computational expensive in comparison with the fuzzy model. Furthermore, a less than idea set of training data was used in neuro-fuzzy training of the model, adversely affecting the accuracy of the model.

Along with a quantitative analysis of the MR dampers, a visual inspection is also pursued. Figs. 36 through 39 display performance of the fuzzy models in comparison to collected experimental data. Here good agreement with collected experimental data can be observed. Thus, the neuro-fuzzy model is deemed to be suitable for numerical simulations.

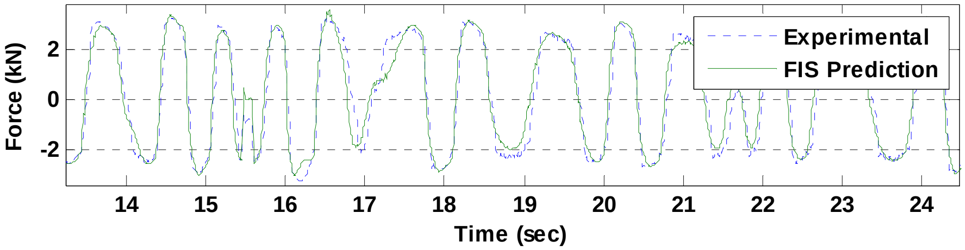


Fig. 36. Data Set 1 (0 V): MR Damper *A*

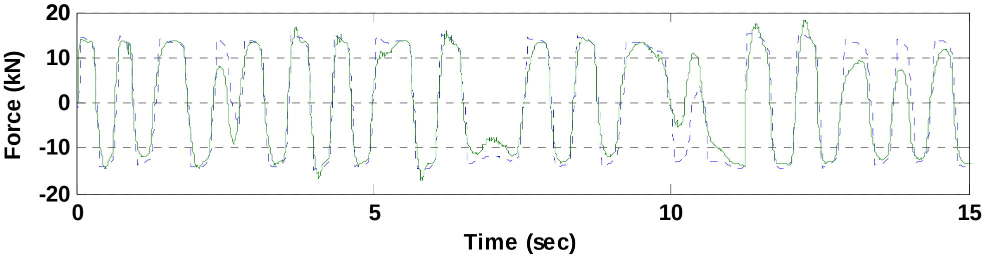


Fig. 37. Data Set 2 (1.2 V): MR Damper *A*

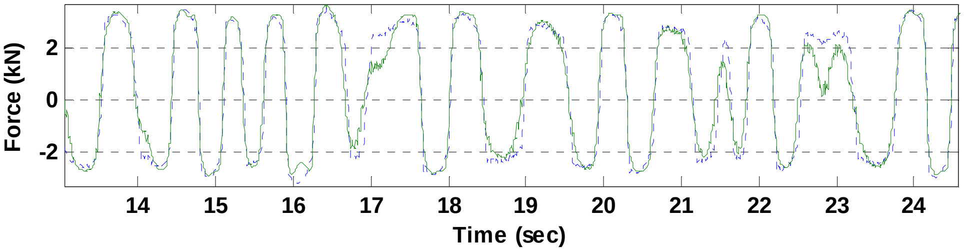


Fig. 38. Data Set 2 (0 V): MR Damper *B*

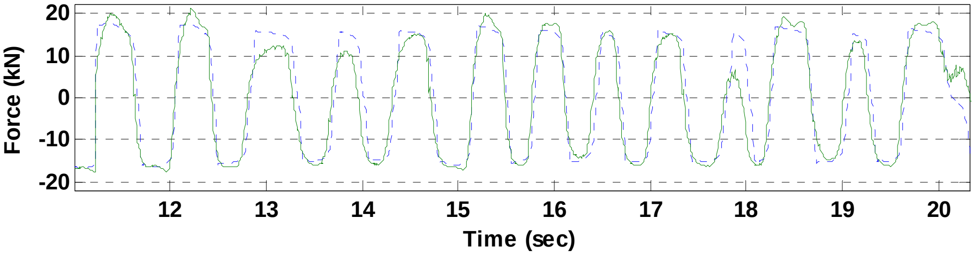
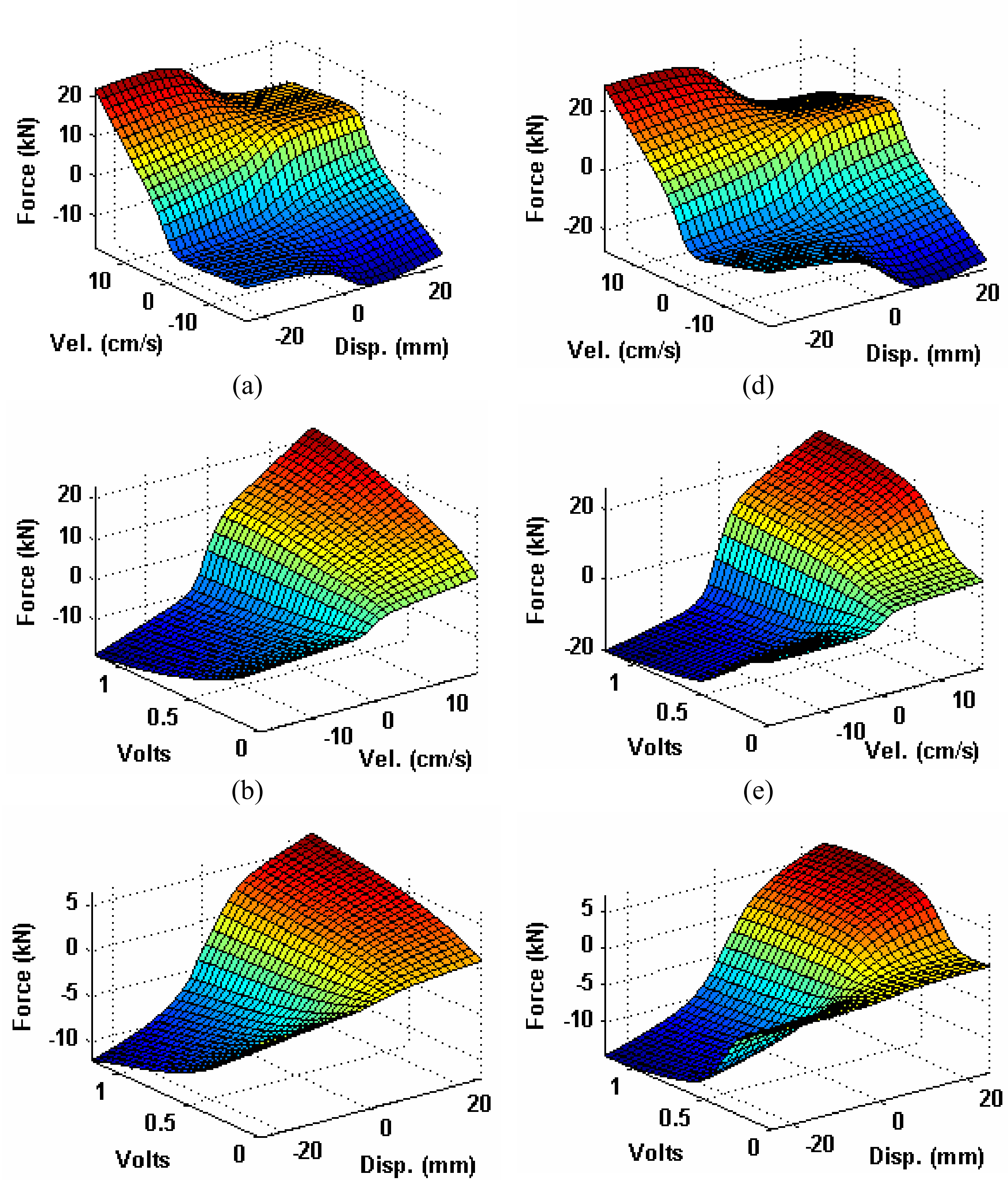


Fig. 39. Data Set 2 (1.2 V): MR Damper *B*

A final inspection of the fuzzy surfaces bears evidence of the ability of fuzzy logic to be evaluated with expert knowledge. As shown in Fig. 40, FIS surfaces have realistic magnitudes and reasonable geometries. It is important to note the differences between the fuzzy models for MR dampers *A* and *B*. Although the dampers have similar mechanical configurations, the resulting numerical models of the dampers are significantly different. The primary difference between the two dampers is the failed coil in MR damper *A*. It is important to note that the difference is accounted for in the fuzzy model of MR damper *A* by observation of Fig. 40. This is especially evident in contrasting Figs. 40(b) versus (e) and (c) versus (f)**.** Here with respect to increasing voltage a nonlinear increase in force is observed in Figs. 40(e) and (f) as opposed to a more linear increase in force as shown in Figs. 40(b) and (c). This phenomenon can be attributed to the position of membership functions shown in Figs. 34 and 35. Here membership functions composing the voltage input of MR damper *A* are generally monotonic for the entire range of voltage input after neuro-fuzzy training. This stands in contrast to MR damper *B* where voltage input membership functions show non-monotonic characteristics. In the monotonic case of MR damper *A* a more linear relationship between voltage and force is established, whereas a more nonlinear relationship is evident in MR damper *B*. Classical modeling methods would have significant difficulty in accounting for such a modification, yet neuro-fuzzy modeling is successful in capturing these important characteristics of MR dampers *A* and *B*.



(a) (d)

(b) (e)

(c) (f)

Fig. 40. FIS Surface Plots: (a, b, c) MR Damper *A* and (d, e, f) MR Damper *B*

3.4.4. Ensuring Realistic Results of Fuzzy Model

A common occurrence resulting from neuro-fuzzy modeling of MR dampers is prediction of a modest force output when displacement and velocity inputs are exactly zero. This error generally does not affect the performance of the FIS when evaluated over experimental results and is often negligible. However, this error can create instability in numerical simulations that are carried out in conjunction with a state-space model of a structure as shown in Fig. 6 with free body assumptions as shown in Fig. 7. If the excitation is zero then the total response of the state-space model is zero. Due to free-body diagram assumptions the MR damper force predicted by the fuzzy model is added into the excitation just before entering into the state-space model. Thus, if the FIS is outputting some non-zero force, this force results in an output from the state-space model being non-zero despite the fact that the external excitation is zero. This error in fuzzy prediction of MR dampers forces often leads to significant inaccuracies in numerical simulations and even instability of the simulation.

To correct this error two approaches can be investigated. The first approach consists of adding a number of artificial data points to the training data that consist of exactly zero displacement and velocity. Corresponding values of the voltage are given nonzero magnitudes since these input data points should not alter the FIS output if displacement and velocity are exactly zero. This alteration of training data is acceptable since the additional training data makes physical sense. Despite the inclusion of new data, neuro-fuzzy training of the FIS produced similar results to the original training data set. Thus, this methodology, although ideal, is abandoned.

The second approach consists of modifying the FIS model directly. This alteration is permissible if the modified FIS still predicts MR damper forces accurately as judged by a comparison with all collected experimental data. Alteration to the FIS model consists of adding one membership function to each input and output as noted in Fig. 41. Here the solid, bold line represents the added membership functions. For displacement and velocity inputs the membership function is restricted to have a small width such that it only affects the region of inputs near zero. The membership function added to the voltage input is specified to be at or near 1.0 over the entire range of the input. This is done since the FIS output should be zero for all voltage levels if displacement and velocity inputs are exactly zero.

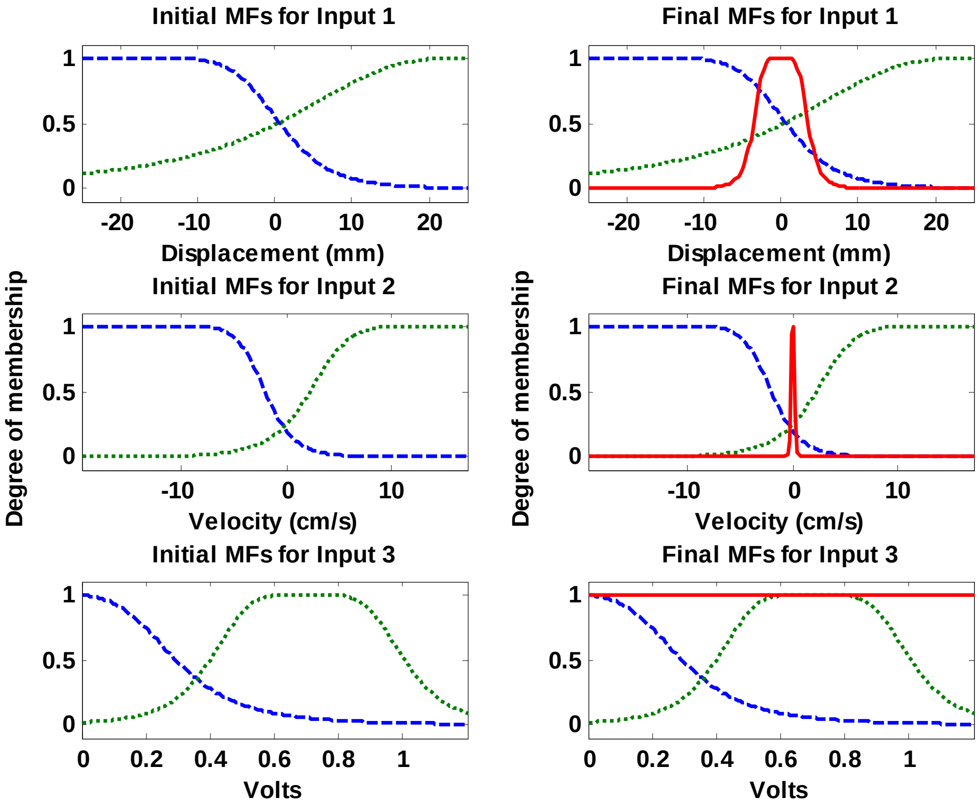
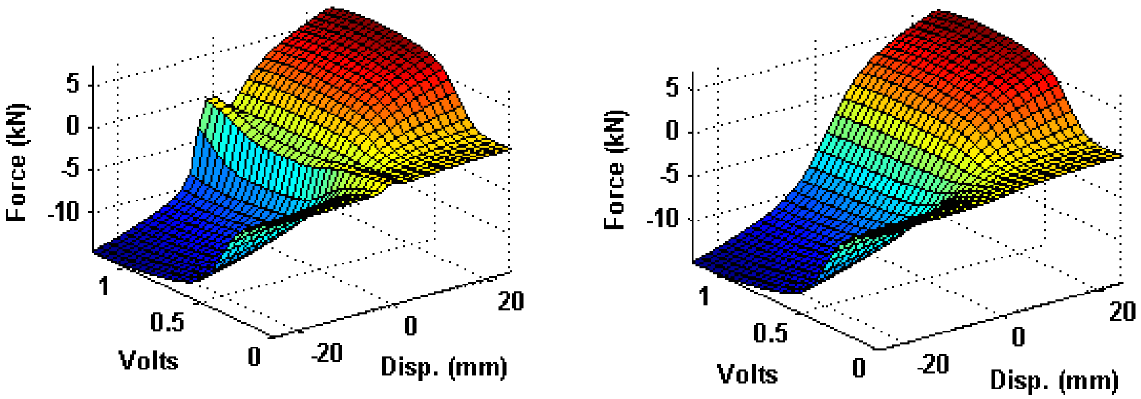


Fig. 41. Membership Functions with Addition of Seat for MR Damper *B*

The new antecedent membership functions are united by a single rule with a constant subsequent for the output. Recall that here Sugeno-type fuzzy inference systems are used for MR damper modeling and ANFIS optimization. Hence, the output membership is a polynomial, not a Gaussian or generalized bell function. A polynomial consisting of only a constant is chosen and set to 0.0 since the outputted force should be 0.0 when each of these membership functions is activated by the inputs. The width of each membership function is determined by its effect on the fuzzy surface. Occasionally the addition of a membership function can cause a sudden irregularity in the surface and thus an unrealistic model as shown in Fig. 42. To prevent this occurrence observation of the FIS surface is required. Note that the FIS surface does not pass through zero at 1.2 volts, but that the total FIS output is approximately zero after inclusion of additional membership functions.



(a) (b)

Fig. 42. Fuzzy Surfaces with (a) Unrealistic Surface and with (b) Realistic Surface

3.5. Investigation of Benchmark Structure and MR Damper Relationship

It is important to understand the relationship between the benchmark structure and the installed MR dampers in both MISO and MIMO configurations. To approximate the interaction between the structure and MR damper(s) numerical simulations are conducted to observe the response of the structure to a single excitation, but with a variety of constant, passive voltages specified to the damper(s). Numerical simulations are conducted with the structure excited by the artificial earthquake. Numerical simulations are conducted with the same excitation signal, but the voltage specified to the MR damper(s) is increased from 0 V to 1.2 V in increments of 0.1 V. By observing the resulting displacement and acceleration responses an optimal reduction of these objectives for passive operation of the damper(s) is identified to occur at 1.1 V and 1.0 V, respectively, for both MISO and MIMO cases. This information reveals that, for passive control, both displacement and acceleration are reduced to their optimal values at or near the passive-on case for this excitation record. What is more, optimal voltage levels may change with varying excitation records. Although it is possible that a passive controller with voltages at or near the passive-on case may provide optimal control, without a thorough investigation of semi-active control such an assumption is not justified. Therefore, a controller superior to the passive-on case is pursued through application of fuzzy logic control.

3.6. Summary

Fuzzy models of two MR dampers have been generated from experimental tests using the neuro-fuzzy optimization algorithm ANFIS. Several methods are utilized to validate each FIS as a viable model of the behavior of its MR damper using both visual inspection and quantitative metrics. Thus, the fuzzy models can be used in development of fuzzy logic controller for mitigation of structural response.

4. DEVELOPMENT OF MULTI-OBJECTIVE GENETIC ALGORITHM

4.1. General

With the test structure identified and accurate models of the MR dampers having been created, construction of a fuzzy logic controller (FLC) is pursued. Since the relationships between the inputs and outputs of a FLC are very complex, an automated process is needed to produce optimal FLCs that can account for these relationships. As mentioned earlier genetic algorithms (GA), first proposed by Holland (1975), have been used in a variety of applications with success in solving problems involving minimization of multiple objectives (Sarkar and Jayant, 2004; Guo *et al.*, 2004). Genetic algorithms mimic natural selection through improvement by fitness and generation cycles. Schaffer (1985) realized the potential for genetic algorithms and hypothesized a multi-objective GA. Early efforts towards the optimization of fuzzy logic by GA were lead by Karr (1991). More advanced Pareto based multi-objective optimization algorithms were proposed by Goldberg (1989) and later by Fonseca and Fleming (1993).

Commonly, genetic algorithms only account for one or two objectives, but in this application the following four structural response characteristics are selected to be the objectives for GA optimization of FLCs:

- Peak inter-story drift of all floors
- Peak absolute acceleration of all floors
- Root-mean-squared of inter-story drift of all floors
- Root-mean-squared of absolute acceleration of all floors

All four of these objectives are vital to the ability of a fuzzy logic controller to minimize structural responses that often incite damage in the structure. Peak inter-story drift is chosen for structural safety in cases of seismic and high wind excitations to help prevent an overloading scenario that could lead to collapse. Root-mean-squared (RMS) of inter-story drift is chosen to minimize the possibility of low-cycle fatigue that can develop quickly in the immediate response of a tall structure to seismic excitation or an extended imposition of strong winds. Peak and RMS accelerations generally are correlated to the comfort of inhabitants and much debate has centered on how to best quantify acceleration in relation to the comfort of the inhabitants. Therefore, both peak and RMS accelerations are accounted for in this study. Furthermore, nonstructural components such as water and electrical machinery are frequently damaged by high accelerations. Considering both comfort and cost, acceleration is a key characteristic of structural response that needs to be minimized.

For discussion, optimization of FLCs is focused on this in this section though similar discussions and arguments could be made for GA system identification of the state-space model of the benchmark structure discussed more in detail in Section 6.

4.2. Overview of NSGA-II CE

A non-dominated multi-objective algorithm (NSGA-II CE) is employed for optimization (Deb and Goel, 2001) of the FLCs. An overview of the algorithm is shown in Fig. 43. The initial population is composed of completely randomized values. A non-random initial set of FLCs is not suggested in order to avoid any user preference or weight to a considered fuzzy relationship. Rather, the approach is to use the GA, through a trial and error process, to identify optimal relationships between inputs and outputs with minimal user assistance. For purposes of optimization a chromosome is created to represent the FLC. A chromosome is a string of real-valued, floating point parameters that can completely describe a single FLC. Encoding of the GA optimization algorithm is conducted in MATLAB/Simulink (2006) and is provided in Appendix B.

4.3. Makeup of Chromosome

The chromosome consists of all floating point values for parameters, not a binary alphabet. The use of real-valued, floating point quantities for parameters over binary digits has gained use since Janikow and Michalewicz (1991) discovered that a real-valued chromosome used in a GA is more efficient in computational time and has a faster convergence rate than traditional binary chromosomes. For this reason real-valued, floating point quantities are used.

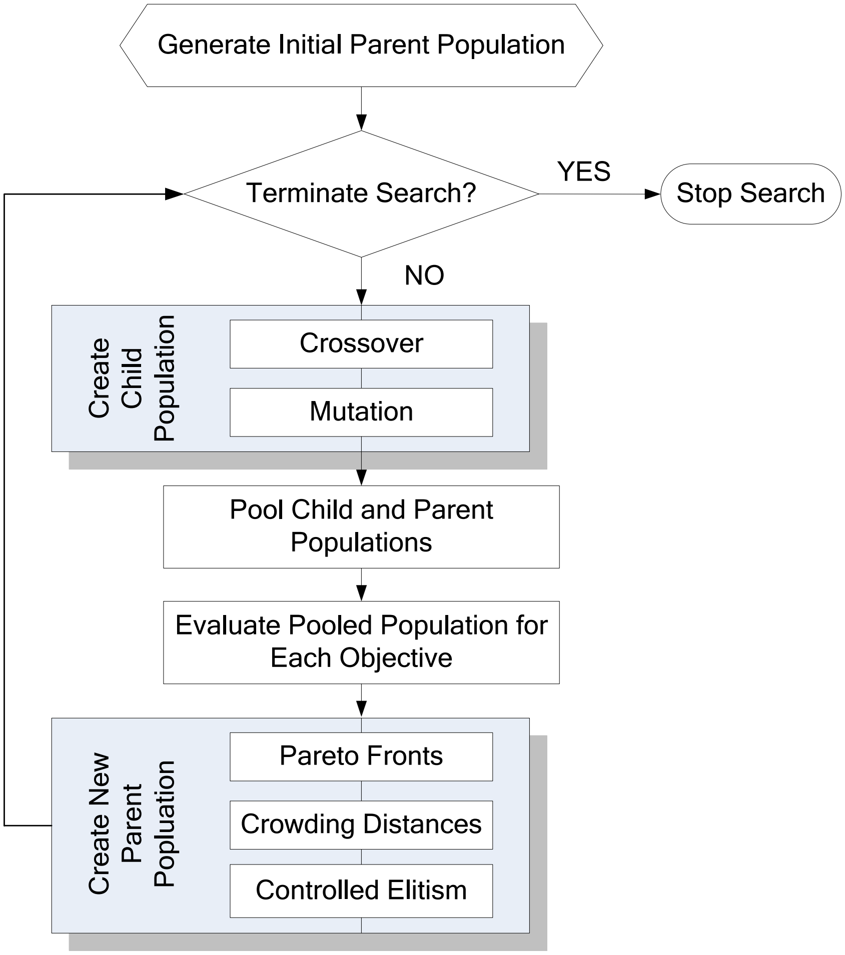

Generate Initial Parent Population
Terminate Search?
YES
Stop Search
NO
Create Child Population
Crossover
Mutation
Pool Child and Parent Populations
Evaluate Pooled Population for Each Objective
Create New Parent Popluation
Pareto Fronts
Crowding Distances
Controlled Elitism

Fig. 43. NSGA-II CE Flow Chart

Each chromosome is subdivided into four sections: FLC rules, objectives, rank, and crowding distance (see Fig. 44). A FLC is completely described by a set of fuzzy rules that are used to relate input and output variables. As described in what follows the fuzzy logic controllers are of the Mamdani-type and, thus, the inputs and outputs are described by membership functions as displayed by Fig. 15.

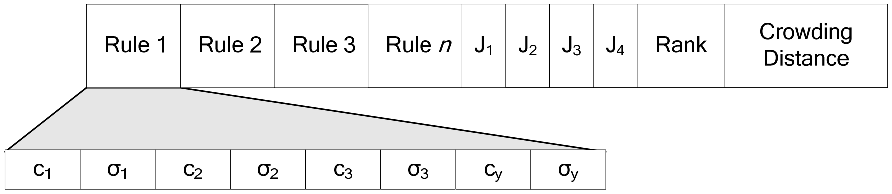


Fig. 44. Example Chromosome

Each rule relates input and output variables by their membership functions. Here Gaussian curve membership functions are employed that can be completely described by the location of its center (c) and its width (σ) as follows:

$$y(x) = e^{\left(-\frac{(x-c)^2}{2\sigma^2}\right)} \tag{15}$$

An example membership function is shown in Fig. 45. The membership function always has a peak value of unity, but the width and location of its center are variable. Ten rules are used to relate inputs and outputs in this study. Since each input and output is assigned one membership function per rule, 40 and 50 membership functions are incurred for the MISO and MIMO cases, respectively. Moreover, two variables (c and σ) are needed to fully describe each membership function; thus, 80 and 100 variables are to be adjusted by the GA for the MISO and MIMO cases, respectively. Initially more rules are believed to be more effective in creating a valuable FLC, but it has been concluded that use of only a few rules produces a more effective and efficient controller.

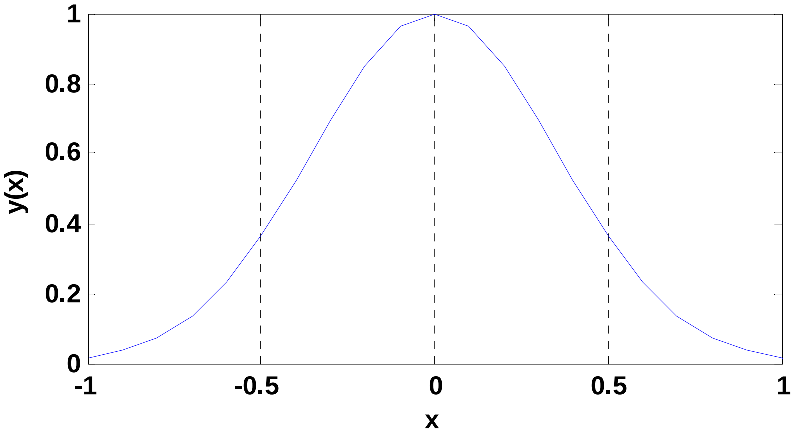


Fig. 45. Example of Membership Function

The values of these variables comprise the first portion of the chromosome and are the building blocks of each FLC. The second and third sections of the chromosome contain information about objectives and objective status.

4.4. Objectives

Effectiveness of each FLC that is created by the GA is evaluated by a quantitative structural response metric that is termed an objective. For this application the FLC is evaluated for effectiveness as the controller for the three story structure that is subjected to a series of seismic loads (see Fig. 8). Results of each FLC are classified according to its ability to reduce inter-story drift ($\hat{x}$) and absolute acceleration ($\ddot{u}$). More specifically each controller is evaluated over four objectives: peak drift (*J₁*), peak acceleration (*J₂*), RMS of displacement (*J₃*), and RMS of acceleration (*J₄*). These objectives are defined as follows:

$$J_1 = \max_j \left\{ \frac{\max_t \left| \hat{x}_{j,controlled} \right|}{\max_t \left| \hat{x}_{j,passive-off} \right|} \right\} \tag{16}$$

$$J_2 = \max_j \left\{ \frac{\max_t \left| \ddot{u}_{j,controlled} \right|}{\max_t \left| \ddot{u}_{j,passive-off} \right|} \right\} \tag{17}$$

$$J_3 = \max_j \left\{ \frac{\sigma(\hat{x}_{j,controlled})}{\sigma(\hat{x}_{j,passive-off})} \right\} \tag{18}$$

$$J_4 = \max_j \left\{ \frac{\sigma(\ddot{u}_{j,controlled})}{\sigma(\ddot{u}_{j,passive-off})} \right\} \tag{19}$$

where σ(*x*) denotes the computation of RMS of quantity *x*, *j* denotes the floor of consideration, and *t* denotes time. The objectives are stored near the end of the chromosome but before the rank and crowding distance (see Fig. 44). To account for all four objectives, a multi-objective GA is necessitated. This leads to a number of optimization considerations that are outlined below.

4.5. Crossovers and Mutations

Fundamental aspects of the evolutionary search process of a genetic algorithm are the concepts of crossovers and mutations. These two operators modify parent chromosomes to produce the next generation of solutions. They allow for exploration of new solutions while maintaining continuity of current solutions.

4.5.1. Crossovers

Crossovers occur when two parent chromosomes exchange equal amounts of information to produce two children as shown in Fig. 46. A parent chromosome is a chromosome from the prior generation. A child chromosome is a newly formed chromosome in the current generation. In this study all individuals in the parent population are subjected to a crossover to produce the child population. The parent and child populations have the same number of chromosomes at the conclusion of the crossover operation.

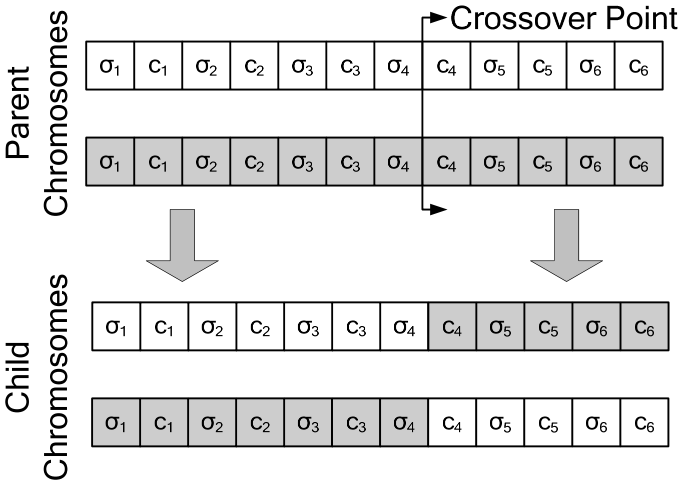


Fig. 46. Crossover Operation

4.5.2. Dynamic Mutations

Data are altered at random after the crossover operations are complete to keep the generation cycle from becoming static; this is called mutation. In this study a very explorative GA is required since four objectives are considered concurrently. Thus, a dynamic mutation algorithm is employed which alters the magnitude of the mutation according to the following:

$$\Delta = y\left(g\left(1-\frac{a}{m}\right)\right)^{b} \tag{20}$$

Here Δ is the value added to the current mutation candidate. Also, *y* is the maximum amount of change allowed by the FLC ranges specified by the user, *g* is a random number between 0 and 1, *a* is the current generation, *m* is the maximum number of generations, and *b* is a user defined shape factor. A larger value of *b* provides a more exploitive mutation process and a low value of *b* produces a more explorative mutation process. In this study a shape factor of 1 is selected. In Fig. 47 Eq. (20) has been evaluated over many generations. In this study 20% of each child population is mutated before being pooled with the parent population. In a single mutation only one value of the chromosome is altered.

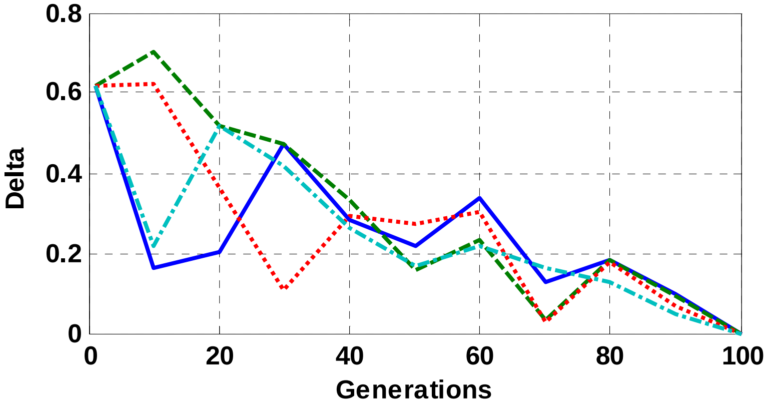


Fig. 47. Delta Computations for Many Generations

4.6. Fronts and Crowding Distances

Recent developments in the field of GA optimization have caused a number of engineering and scientific fields to give consideration to its inherent simplicity and capacity to be tailored for unique situations. A key reason for the increased interest in GA optimization is the ability to optimize on several objectives simultaneously. Several widely accepted multi-objective genetic algorithms include: Pareto-Archived Evolution Strategy (PAES) (Knowles and Corne, 1999), Strength Pareto Evolutionary Algorithm (SPEA) (Zitzler and Thiele, 1998), and Non-dominated Sorting Genetic Algorithm (NSGA-II CE) (Deb and Goel, 2001).

Two central factors of all GAs are the exploitation of current solutions and exploration of new solutions. PAES and SPEA suggest using an archived set of optimal solutions, and each have an algorithm for entrance into this optimal archived set of solutions. The archived sets of solutions for PAES are safe from elimination unless a newly produced child solution proves superior to both parents that spawned the child. This creates a very exploitive nature for PAES, but limits its explorative aspects.

SPEA specifies that the first non-dominated front of each generation be included in the archived set of optimal solutions. Thus the archived set of solutions can become exceedingly large. Child populations are derived from this large population which gives

SPEA a strong explorative nature, yet a much weaker exploitive nature. To avoid an excessively large archive population, a pruning algorithm is employed to manage the size of the archived set of solutions. Flowcharts describing PAES and SPEA are shown in Fig. 48.

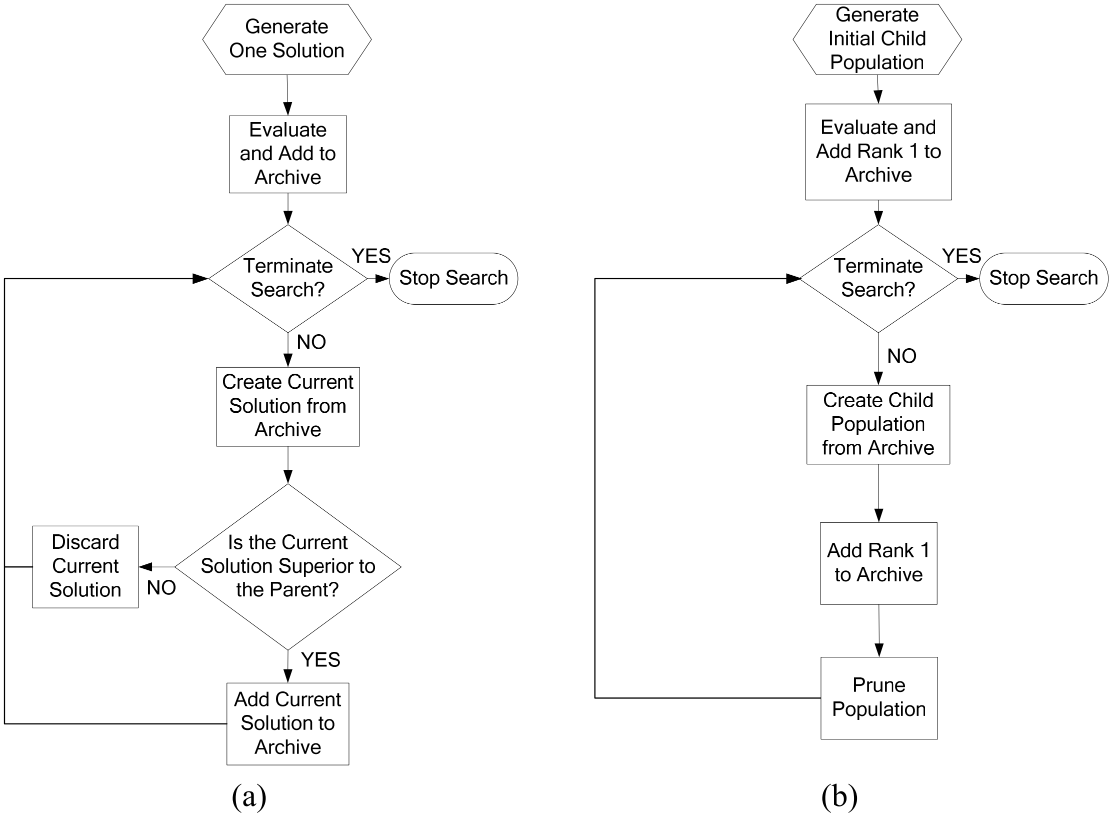

Generate One Solution
Evaluate and Add to Archive
Terminate Search?
YES
Stop Search
NO
Create Current Solution from Archive
Is the Current Solution Superior to the Parent?
NO
Discard Current Solution
YES
Add Current Solution to Archive
(a)
Generate Initial Child Population
Evaluate and Add Rank 1 to Archive
Terminate Search?
YES
Stop Search
NO
Create Child Population from Archive
Add Rank 1 to Archive
Prune Population
(b)

Fig. 48. Flowcharts (a) PAES and (b) SPEA Algorithms

NSGA-II CE allows for the entire population to be evaluated at each generation; thus, none of the solutions are truly safe from elimination. This, in turn, reduces the exploitive nature but greatly increases the explorative nature of NSGA-II CE.

4.6.1. A Description of Pareto Fronts

The following is an outline of the definition of a Pareto front. A "rank" is an integer value assigned to a front. A front is a group of non-dominated individuals that reside in the current population. The 1st front is completely non-dominated. That is, each individual of the front must have no superior when all objectives are considered. Superiority is determined by checking that no other individuals dominate the individual.

The striped region in Fig. 49 illustrates this concept. For example no other FLC can have superior results for "Objective 2" at "Objective 1" than the considered FLC. Once the 1st front is determined by locating all non-dominated solutions, its FLCs are set aside and the next front of "non-dominated" solutions is determined. They are denoted as the 2nd front and so on.

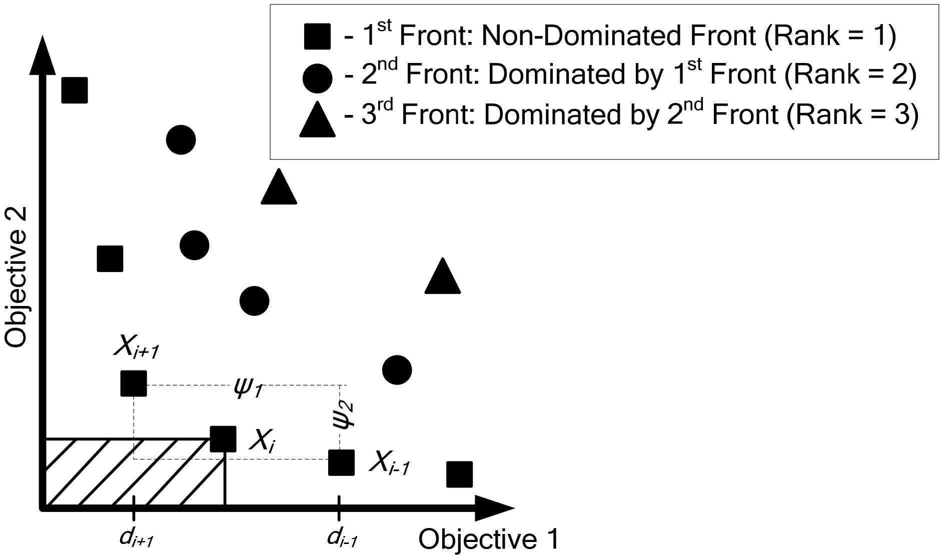

■ - 1st Front: Non-Dominated Front (Rank = 1)
● - 2nd Front: Dominated by 1st Front (Rank = 2)
▲ - 3rd Front: Dominated by 2nd Front (Rank = 3)
Objective 2
X_{i+1}
ψ_1
ψ_2
X_i
X_{i-1}
d_{i+1}
d_{i-1}
Objective 1

Fig. 49. Pareto Fronts and Crowding Distances

4.6.2. A Description of Crowding Distances

Crowding distances are a way to differentiate members of an individual front. A crowding distance is a value that describes the density of solutions surrounding a single solution in a single Pareto front. In multi-objective optimizations these densities are summed as follows:

$$\psi_k = \sum_{j=1}^{J} (d_{i-1} - d_{i+1}) \tag{21}$$

$$\varphi = \sum_{k=1}^{K} \psi_k \tag{22}$$

where φ denotes the crowding distance of an individual with respect to all Pareto fronts, ψ denotes the crowding distance of a single individual with respect to members of the kth Pareto front, X denotes the i^{th} individual of a common Pareto front, J is the total number of optimization objectives, and K denotes the number of Pareto fronts defined later in

Eqn. (23). Prior to computation of crowding distances the entire pool of solutions are sorted in into Pareto fronts. Then ψ is computed for each individual of each Pareto front. Note that the crowding distance is cumulative among all objectives. Thus, a crowding distance with respect to members of a common Pareto front is attained. In Fig. 49 the crowding distances ψ_1 and ψ_2 of a single solution with respect to 'Objective 1' and 'Objective 2' are shown, respectively.

Between two solutions (i.e. FLCs) within the same rank, the solution with the larger crowding distance is considered superior. This may seem counter intuitive since the optimal solution is not necessarily far away from the other solutions. However, this logic is implemented to encourage strong diversity of the population. Moreover, a minimum number of solutions in each front are specified to keep all solutions of a population from evolving into a single front. Exploration of new solutions and diversity of current solutions are key issues when working with a GA.

4.6.3. Longitudinal and Lateral Diversity

Two types of diversity are considered in this study: longitudinal and lateral diversity as depicted in Fig. 50. PAES, SPEA, and NSGA-II CE give consideration to longitudinal diversity with the use of clustering and/or crowding distance. Lateral diversity is satisfied by maintaining a population consisting of multiple Pareto fronts. Often, GA optimization converges toward a local Pareto front, while the global front still remains unknown. Requiring lateral diversity aids in preventing solutions from converging too quickly to a local Pareto front. PAES gives minimal consideration to lateral diversity, but SPEA gives significant consideration with its archived set of non-dominated fronts from numerous generations. NSGA-II CE proposes using controlled elitism through a forced selection of Pareto front sizes to ensure that individuals from a specified number of fronts are maintained in the population. This differs from SPEA which maintains an archived set of non-dominated solutions from all or many generations. Considering numerous previous non-dominated fronts from prior generations in forming a child population is relatively inefficient in maintaining lateral diversity in comparison with NSGA-II CE.

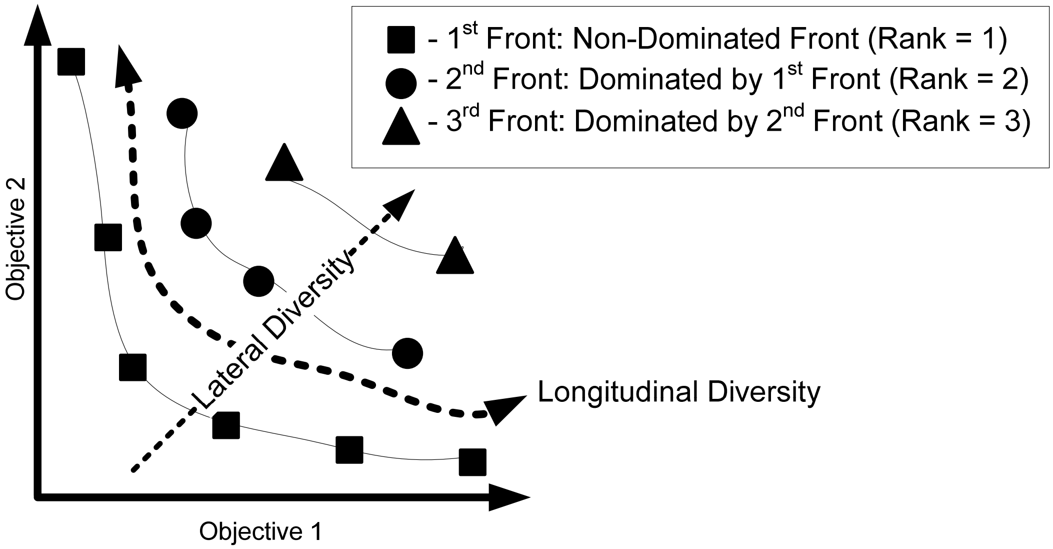

■ - 1st Front: Non-Dominated Front (Rank = 1)
● - 2nd Front: Dominated by 1st Front (Rank = 2)
▲ - 3rd Front: Dominated by 2nd Front (Rank = 3)
Objective 2
Objective 1
Lateral Diversity
Longitudinal Diversity

Fig. 50. Types of Diversity

4.7. Controlled Elitism

To avoid early convergence to a localized solution a more explorative GA is required. A vital component to an explorative GA is diversity in the population. Deb and Goel (2001) have determined that forcing dominated solutions to coexist in a population is an effective way to avoid early convergence, especially in a multi-objective GA which is termed controlled elitism. This philosophy differed from the original NSGA-II (Deb *et al.* 2002) which did not include controlled elitism. A geometric distribution is used to specify the number of solutions allowed in each front as follows:

$$n_i = N\frac{1-r}{1-r^K}r^{i-1} \tag{23}$$

where n_i represents the number of individuals allowed in rank i. The total number of individuals in the population is N, K is the desired number of fronts for the population, and r is a user defined parameter. Typically 2 to 5 fronts are desirable; in this study 4 fronts are specified.

A large value of r produces a uniform distribution (most diversity) and a smaller value of r produces a more exponential distribution (minimal diversity). An r value of 0.5 is used for this study. Fig. 9 shows the population distribution among fronts for a population (N) of 100 and K of 4 with different r values. Thus, a large r is more explorative of new solutions and a small r is more exploitive of current solutions.

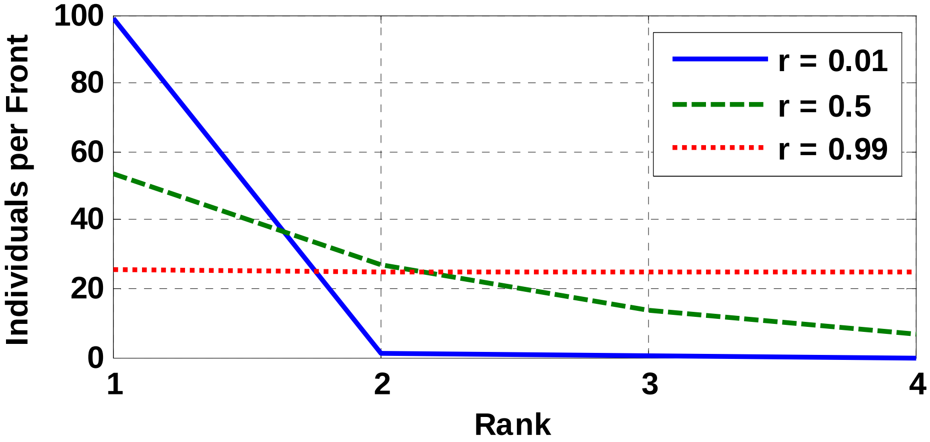


Fig. 51. Population Size for Each Pareto Front for Different *r* Values

4.8. Optimization Examples

To better understand NSGA-II CE and to validate its improvement from its predecessor, NSGA-II, two commonly used optimization problems are considered. The first optimization problem is termed 'F6' in optimization literature and consists of a very complex scenario where identifying the global minima is difficult due to a significant number of local minima as proposed by Schaffer *et al.* (1989). The 'F6' function contains only two variables and is a single optimization objective. The second optimization problem is titled 'ZDT4' and is augmented by the Griewangk functional as proposed by Deb (1999). This problem can have any number of variables and contains two competing objectives as discussed by Deb and Goel (2001). For this study 10 and 100 variables are demonstrated. Initially, all GA examples in this section utilize a population of 100, undergo 200 generation cycles, are given a crossover rate of 95%, and have no mutations. Mutations are suppressed to allow for a more clear comparison between NSGA-II and NSGA-II CE algorithms. Each algorithm beings with the same initial randomized set of FLCs.

4.8.1. F6 Optimization Problem

The F6 problem is defined as follows:

$$F(x_1, x_2) = 0.5 + \frac{\sin^2(\sqrt{x_1^{\,2} + x_2^{\,2}}) - 0.5}{1.0 + 0.001 \cdot (x_1^{\,2} + x_2^{\,2})^2} \qquad (24)$$

where x_1 and x_2 are variables to be defined. If x_1 and x_2 are evaluated over a range of ± 20 Figs. 52 and 53 can be generated. The F6 surface has numerous local minima near the global minima which is located at exactly (0,0) where the F6 function is exactly zero. Note that the global minimum is located at the origin and is equal to zero but that there are numerous local minima in the vicinity of the origin. Thus, 'F6' is considered to be a difficult optimization problem.

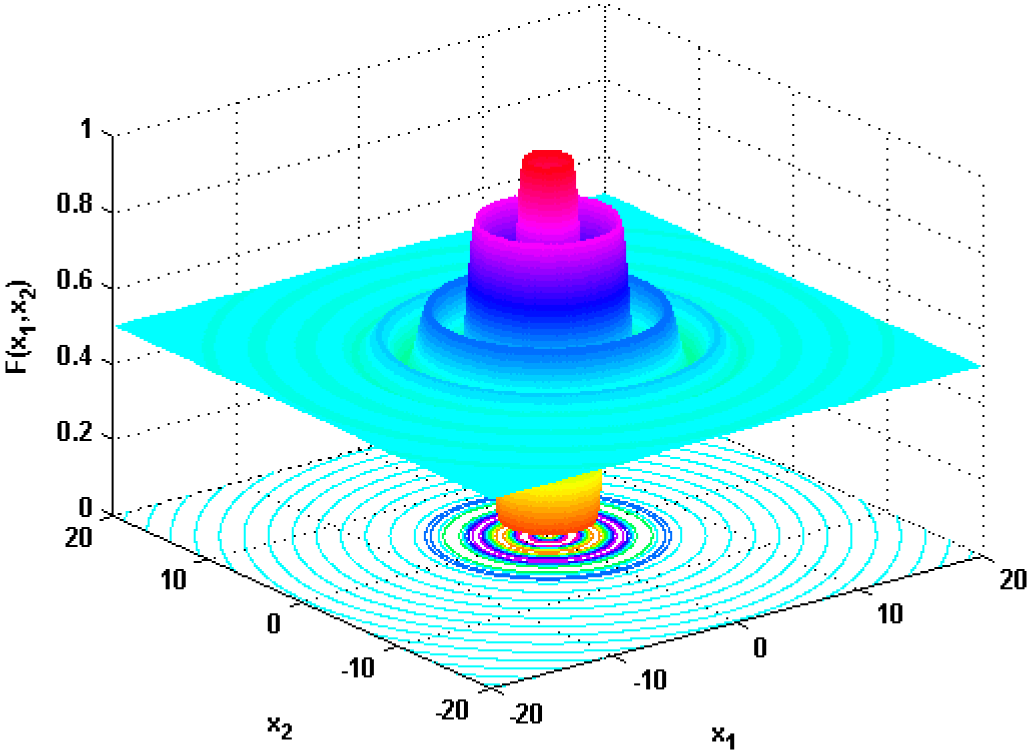


Fig. 52. F6 Surface

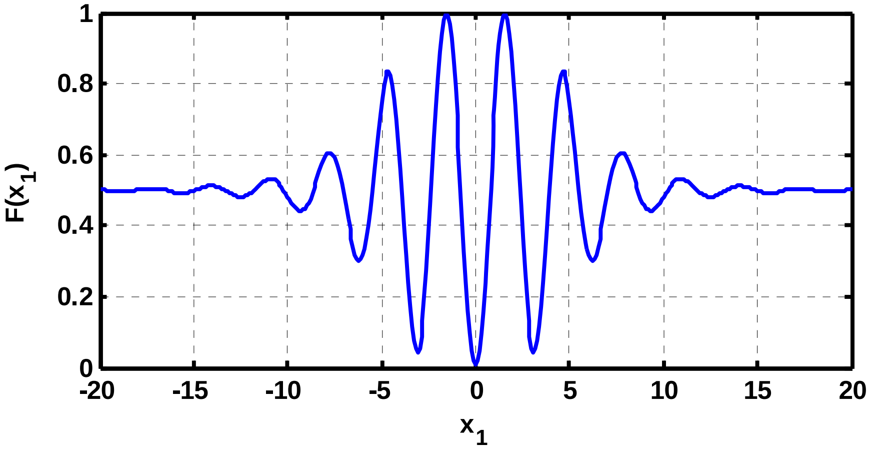


Fig. 53. F6 2D Slice

For the F6 optimization problem both NSGA-II and NSGA-II CE are not able to locate the global minimum when mutations are suppressed. This is due numerous local minima near the global minimum. To observe the performance of each algorithm with the aid of mutations a mutation rate of 20% is now included for each algorithm. After conducting GA optimization 10 times for each algorithm it is found that NSGA-II does not locate the global minimum within 200 generations. However, NSGA-II CE identifies the global minimum within approximately 20 generations. As shown in Fig. 54, NSGA-II CE is able to quickly realize the global minimum. Thus, for the F6 optimization problem NSGA-II CE has been determined to be superior to NSGA-II, but only with the aid of mutations.

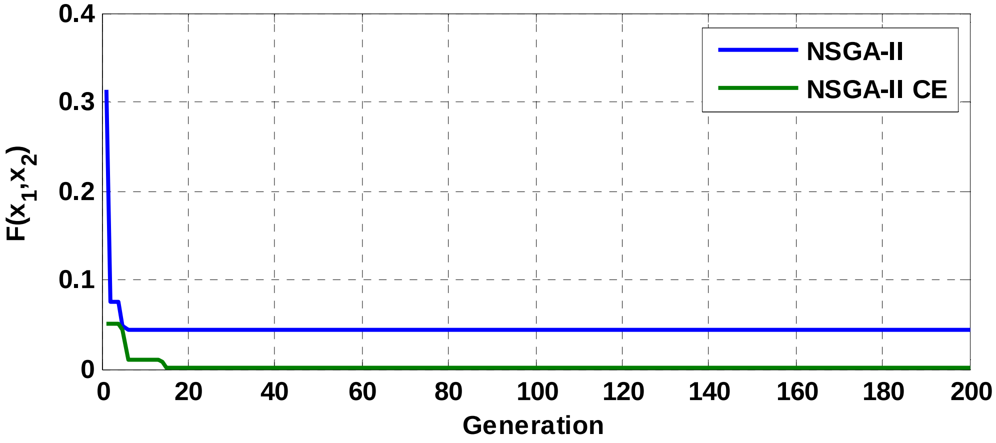


Fig. 54. Results of F6 Optimization

4.8.2. ZDT4 Optimization Problem

The ZDT4 optimization problem augmented with the Griewangk functional is defined by:

$$f_1(x) = x_1 \tag{25}$$

$$f_2(x) = g(x)[1 - \sqrt{x_1/g(x)}] \tag{26}$$

$$\begin{gathered} g(x) = 91 + \sum_{i=2}^{10}[x_i^2 - 10\cos(4\pi x_i)] \\ x_1 \in [0,1], x_2 \in [-5,5], i = 2,...,10. \end{gathered} \tag{27}$$

where $f_1(x)$ and $f_2(x)$ define the two objectives, x_1 and x_i are variables to be defined by GA, $g(x)$ is the Griewangk functional, and i denotes the number of variables to include. This optimization problem possesses less complexity between variables than the F6 problem, but is scalable to any number of variables. For this problem two numerical examples are presented with 10 and 100 variables, respectively.

To better understand the Griewangk functional Fig. 55 is created to show the case when i is set equal to 2. In this case a surface can be generated that describes a two variable application of the Griewangk functional. Clearly, the global minimum is not as difficult to locate as for the F6 functional, but with competing objectives and numerous variables included; this multi-objective optimization problem becomes exceedingly difficult. Also, it is difficult to quantify results since this is a multi-objective problem. Therefore, the most globally optimal Pareto front discovered by each algorithm is used as a comparison.

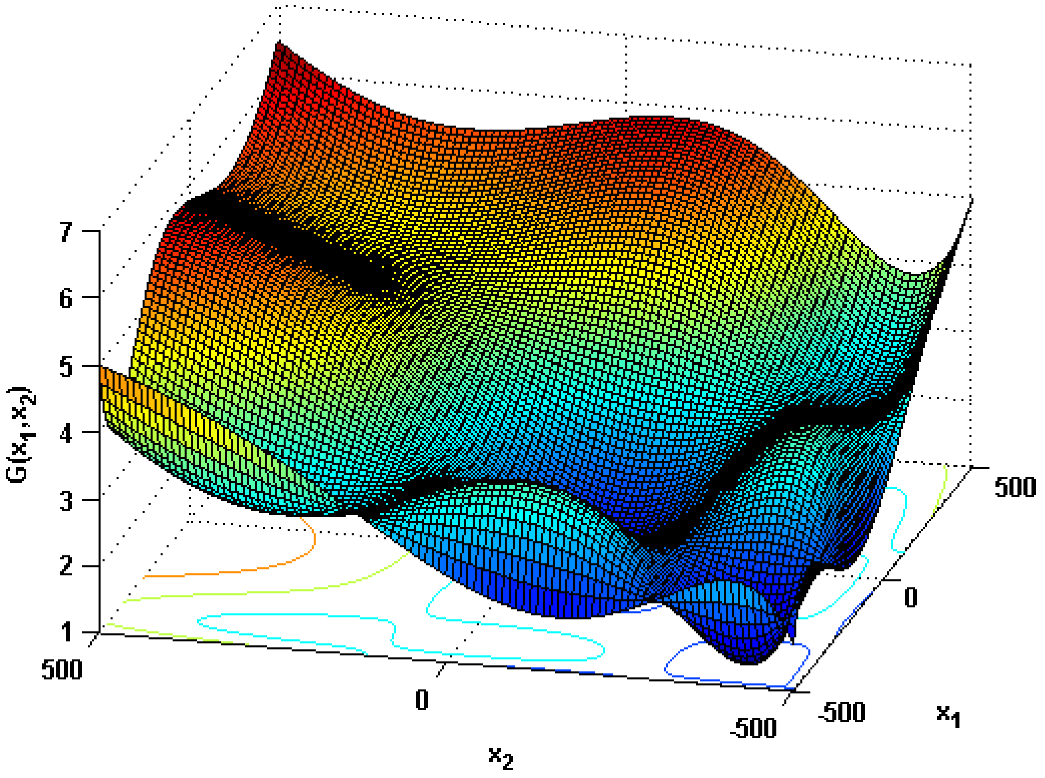
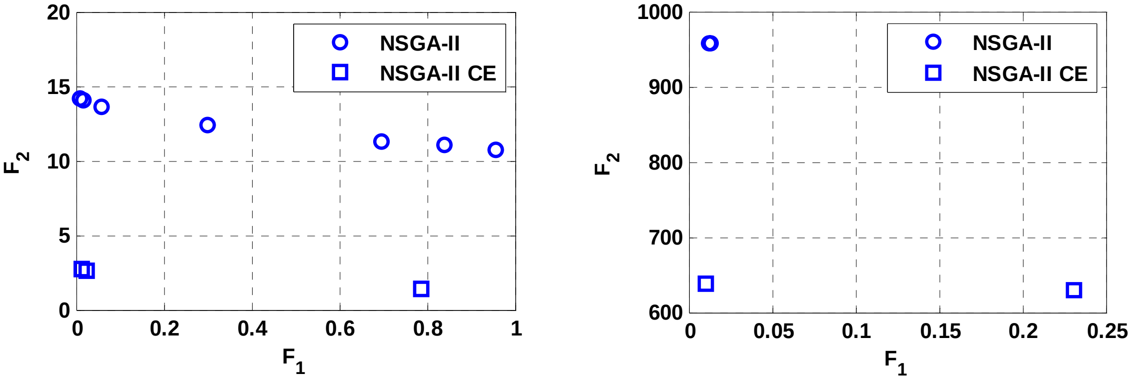


Fig. 55. Griewangk Problem with 2 Variables

Results of NSGA-II and NSGA-II CE optimization for ZDT4 are shown in Fig. 56. Here, it is evident that NSGA-II CE provides a more globally optimal set of solutions than the original NSGA-II with either 10 or 100 variables. The 100 variable case is of primary interest in this study since fuzzy logic controllers for the MISO and MIMO cases consist of 80 and 100 variables, respectively. As evidenced by this study and prior work of Deb *et al.* (2002), NSGA-II CE is shown to be superior to previous optimization methods that are based on the NSGA-II algorithm.



(a) (b)

Fig. 56. Results of ZDT4 Optimization with (a) 10 and (b) 100 Variables

4.9. Summary

A genetic algorithm has been outlined, discussed, and explored. With an accurate model of the MR damper having been identified and the GA optimization method being formulated, optimization of the state-space model of the benchmark structure is now presented.

5. GENETIC ALGORITHM SYSTEM IDENTIFICATION

5.1. General

A GA algorithm for system identification the benchmark building has been developed that incorporates the same NSGA-II CE algorithm employed for development of fuzzy logic controllers. In this case the goal for is to determine stiffness and damping coefficients used in a state-space representation of the structure. In the current study, a robust approach to optimization is presented since prediction of displacement, velocity, and acceleration are all considered concurrently through multi-objective optimization. Furthermore, full-scale tests are conducted for verification of the GA system identification results over a suite of excitations.

Four variables are estimated by the GA to predict the response of the structure to excitation including stiffness matrix coefficients and the critical damping ratio. With this information the following can be defined:

$$M\ddot{U} + C\dot{U} + KU = -M\ddot{U}_g \tag{28}$$

$$M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix} \tag{29}$$

$$C = \alpha M + \beta K$$

$$\alpha = 2\frac{\zeta\omega_1\omega_3}{\omega_1 + \omega_3},\ \beta = 2\frac{\zeta}{\omega_1 + \omega_3}$$

where U is a vector of displacements of the building floors relative to its base, $\ddot{U}_g$ is the ground acceleration, M is the mass coefficient matrix, K is the stiffness coefficient matrix, C is the damping coefficient matrix, ζ is the critical damping ratio, and ω_1 and ω_2 are 1st and 3rd fundamental frequencies of the structure. For the damping matrix α and β are Rayleigh damping coefficients (Chopra, 2006) that are determined by an eigenvalue analysis. The masses of the structure (m_1, m_2, and m_3) are assumed to be known and are calculated using assumed geometries and densities of the structure (see Table 2).

Assuming that the ends of the columns do not rotate, the translational stiffness of each column about its weak axis is initially calculated to be approximately 500 kN/m per column using the following equation:

$$k = 12\frac{EI}{L^3} \tag{30}$$

where E is Young's modulus of elasticity, I is the second area moment of inertia about the weak axis, and L is the unsupported length of the column. Thus, an effective stiffness per floor of approximately 2,000 kN/m can be assumed since there are four columns acting together. This calculation is used as a reference point in deciding the range of the search space of the GA. The search space for the GA is taken to be ±50% of the calculated k value. Damping is estimated by computing Raleigh damping coefficients. Here damping is assumed to be accurately estimated by using modes 1 and 3 (Chopra 2006). A range of 0.001 – 0.01 is assumed for critical damping, ζ, since damping effects should be modest in this all-steel structure. In summary, the variables for the GA system identification algorithm to identify include k_1, k_2, k_3, and ζ.

Training data for the GA consists of full-scale experimental results that are collected during testing of the uncontrolled benchmark structure when no dampers are installed although the chevron braces are attached to the frame. While excitations from numerous historic earthquakes are tested on the uncontrolled benchmark structure, data collected from an El Centro 100 gal earthquake are used for training. This excitation is selected since results of GA optimization are observed to be more accurate than results that are obtained using other excitations. Displacement and acceleration data are collected through an array of LVDTs and accelerometers, respectively, which are attached to each floor of the benchmark structure and discussed in Section 3.

Three objectives are used to quantify simulated results. Each objective is the root-mean-squared-error (RMSE) computed when compared to actual full-scale experimental results with respect to displacement, velocity, and acceleration response. These objectives are expressed mathematically as follows:

$$J_1 = \max_j \left\{ \sigma(u_{j,\text{exp.}} - u_{j,sim.}) \right\} \tag{31}$$

$$J_2 = \max_j \left\{ \sigma(\dot{u}_{j,\text{exp.}} - \dot{u}_{j,sim.}) \right\} \tag{32}$$

$$J_3 = \max_j \left\{ \sigma(u_{j,\text{exp.}} - \ddot{u}_{j,sim.}) \right\} \tag{33}$$

where *j* denotes the floor under consideration, u, $\dot{u}$, and $\ddot{u}$ denote displacement, velocity, and acceleration relative to the base of the structure, respectively.

GA optimization results (see Table 3) reveal that GA optimized stiffness values differ considerably from those of the closed-form calculations from Eq. (30). It should be noted that GA-optimal values for k_1, k_2, k_3, and ζ are not unique since the state-space formulation is non-unique. These values have an *uncertain physical meaning*. Figs. 57 and 58 show estimated structural responses of the 3rd floor from the GA-optimized structural model of the benchmark structure that is excited by 100 gal El Centro and 200 gal Kobe earthquakes, respectively. Results shown in Fig. 57 correspond to the uncontrolled case with no MR dampers installed, while results shown in Fig. 58 correspond to GA-optimized FLC control of the MIMO structure. Measured and fuzzy predictions for MR damper *B* are also shown. Note that accuracy of these results relies on the accuracy of the GA-optimized state-space model and also on the ANFIS optimized neural fuzzy models of the MR dampers.

To further affirm the state-space model with experimental results, a fast Fourier transform is taken of the acceleration response of the 3rd floor from both numerical simulation and experimental testing (see Figs. 59 and 60). The sinusoidal excitation is a simple test where a few cycles of a sine wave are generated at 1.38 Hz to excite the structure. In both the simple sinusoidal case and a more complex seismic case the GA-optimized state-space model are shown to be highly accurate in reproducing experimental results. It can be observed that the first three modes are modeled accurately since the peak frequencies show strong agreement. Table 4 shows the eigenfrequencies of the benchmark structure according to the GA-optimized state-space model. Note that each plot has been normalized to unity for the ordinate axis.

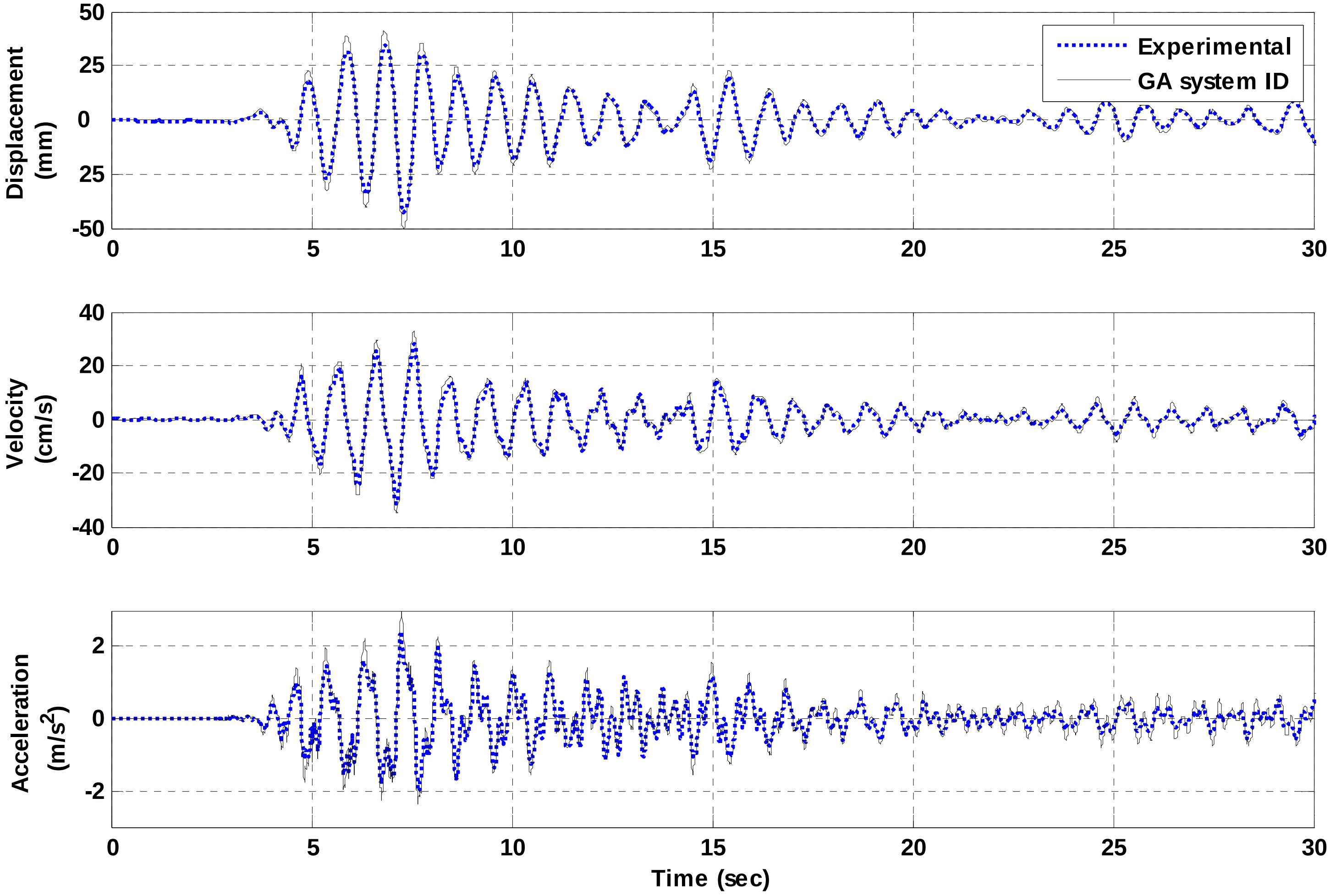


Fig. 57. GA Estimation of Uncontrolled 3rd Floor Response Resulting from El Centro (100 gal) Excitation

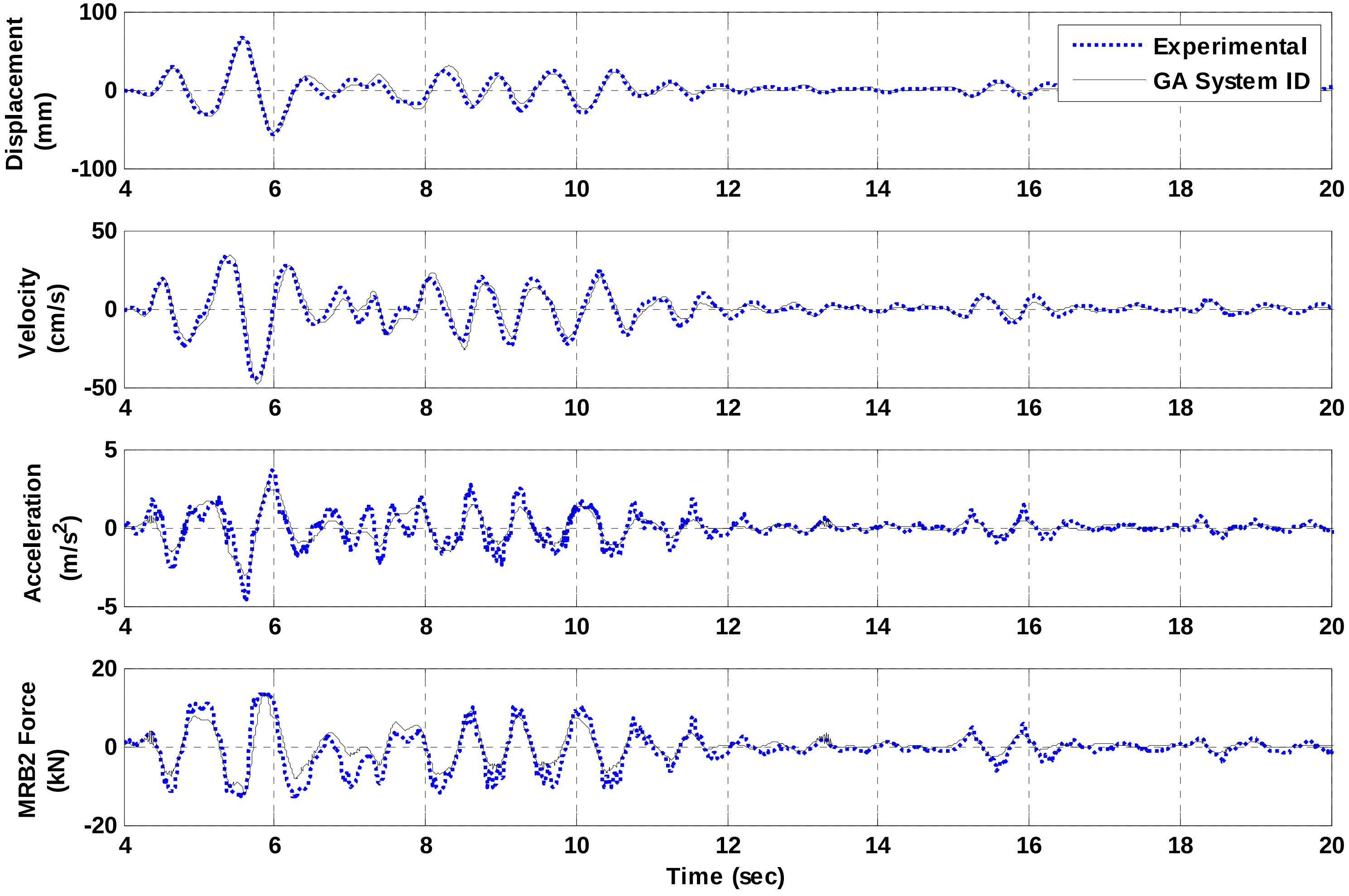


Fig. 58. GA Estimation of MIMO Controlled 3rd Floor Response Resulting from Kobe (200 gal) Excitation

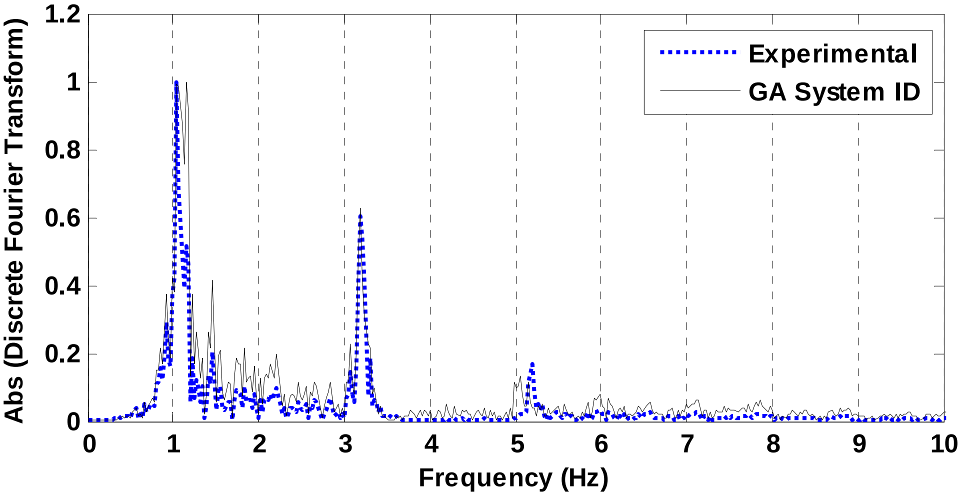


Fig. 59. FFT of Acceleration Responses for 3rd Floor: El Centro 100 gal

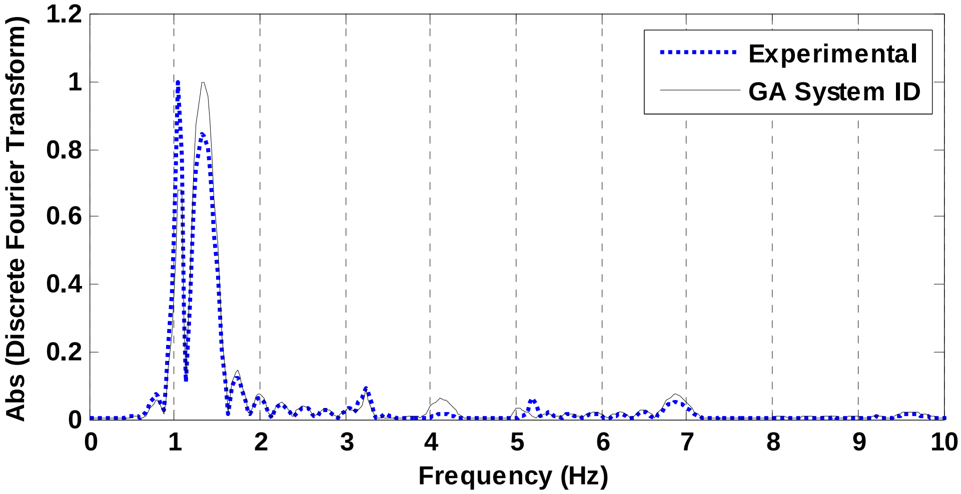


Fig. 60. FFT of Acceleration Responses for 3rd Floor: Sinusoidal Excitation

5.2. Summary

The previous text discusses GA optimization of the state-space model of the benchmark structure. Four optimization objectives are included to accurately quantify error in the state-space prediction when compared to experimental results. Results show that a highly accurate state-space model has been achieved. Next, results of GA optimization of the FLC are provided. Furthermore, numerical and experimental substantiation of the FLC are given.

6. RESULTS OF GA OPTIMIZATION AND EXPERIMENTAL TESTING

6.1. General

Results of GA optimization of fuzzy logic controllers are described in this section. First results of GA optimization are shown and a fuzzy logic controller is identified for both the MISO and MIMO control cases. Then numerical simulations are carried out to show the effectiveness of the identified fuzzy logic controllers for a suite of excitations containing near- and far-field excitations. Finally, experimental validation of controllers is provided from large-scale experimental tests.

6.2. Results of GA Optimization

In what follows results of GA optimization of FLCs for the MISO and MIMO cases are provided. Figs. 61 and 62 show the Pareto fronts that result from GA optimization of MISO and MIMO controllers, respectively. Here, each symbol represents the performance of one FLC when considering two of the four performance objectives. Since four objectives are considered six plots are required to describe the GA-optimized results for each damper configuration. Figs. 63 and 64 show rank 1 of generations 1, 50, and 200; thus, the progression from early, primitive controllers to more advanced controllers is evident. Longitudinal diversity is maintained as can be observed by the final population of generation 200. This diversity in the GA pool of fuzzy logic controllers demonstrates that by favoring solutions with increased crowding distances a pool of FLC controllers displaying longitudinal diversity can be maintained. Lateral diversity of solutions can be observed in Figs. 61 and 62 by noting the number of solutions in each front of generation 200. Thus, it is shown that NSGE-II CE is successful in maintaining both lateral and longitudinal diversity. Furthermore, a balance of optimization towards the reduction of displacement and acceleration responses can be observed. Hence, the benefit of concurrent optimization is shown.

In several cases controllers from early generations had already attained a substantial reduction in a single objective and, therefore, only minimal improvement is observed. For example, controllers that reduced displacement well in early generations commonly specify significant amounts of voltage that closely approximates the passive-on case. Still other controllers are outputting small amounts of voltage and imitate the passive-off case. GA-optimization is able to combine these characteristics to define a smart controller that combines the favorable attributes of each case into a single controller.

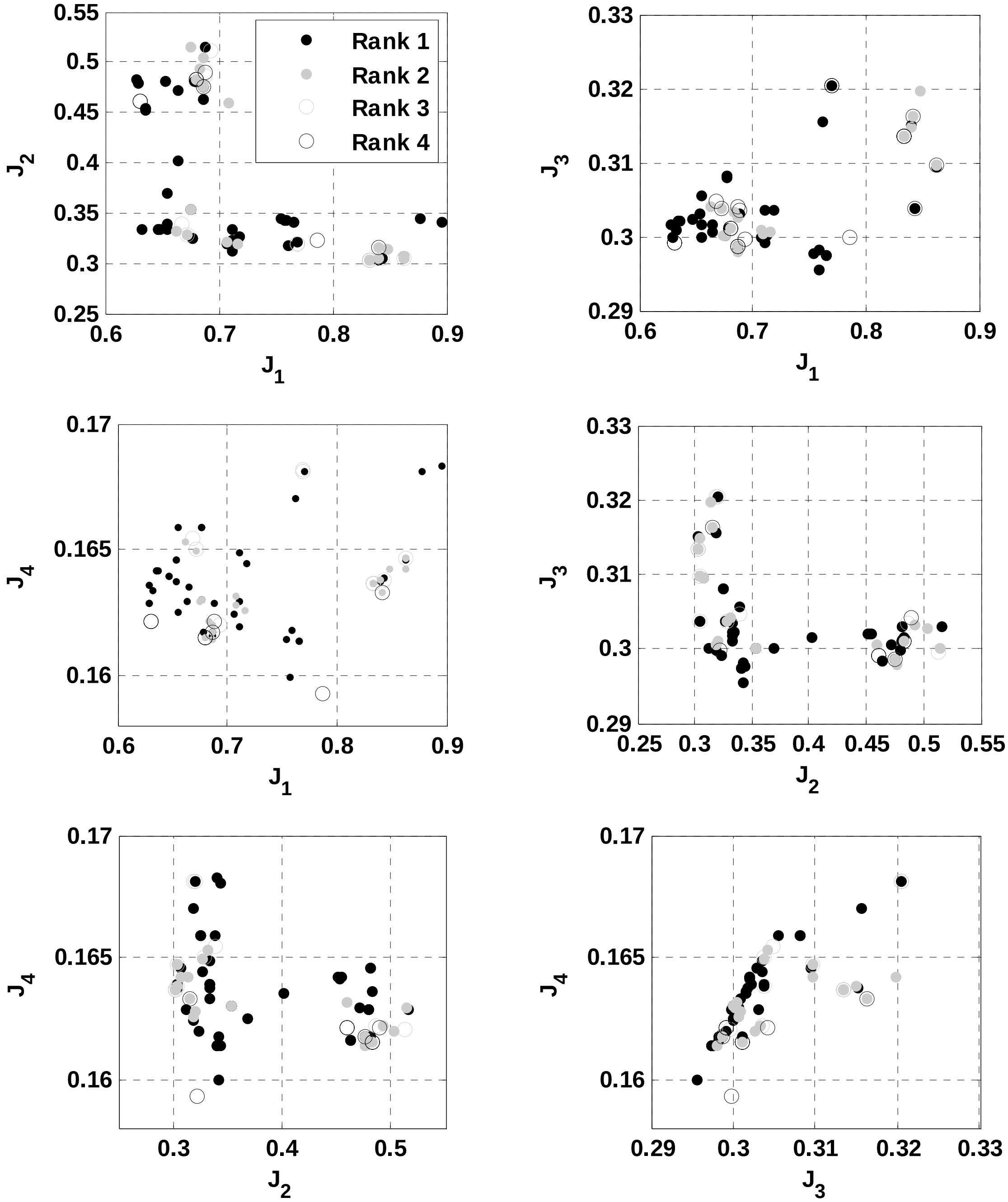


Fig. 61. Pareto Fronts Resulting from MISO GA Optimization

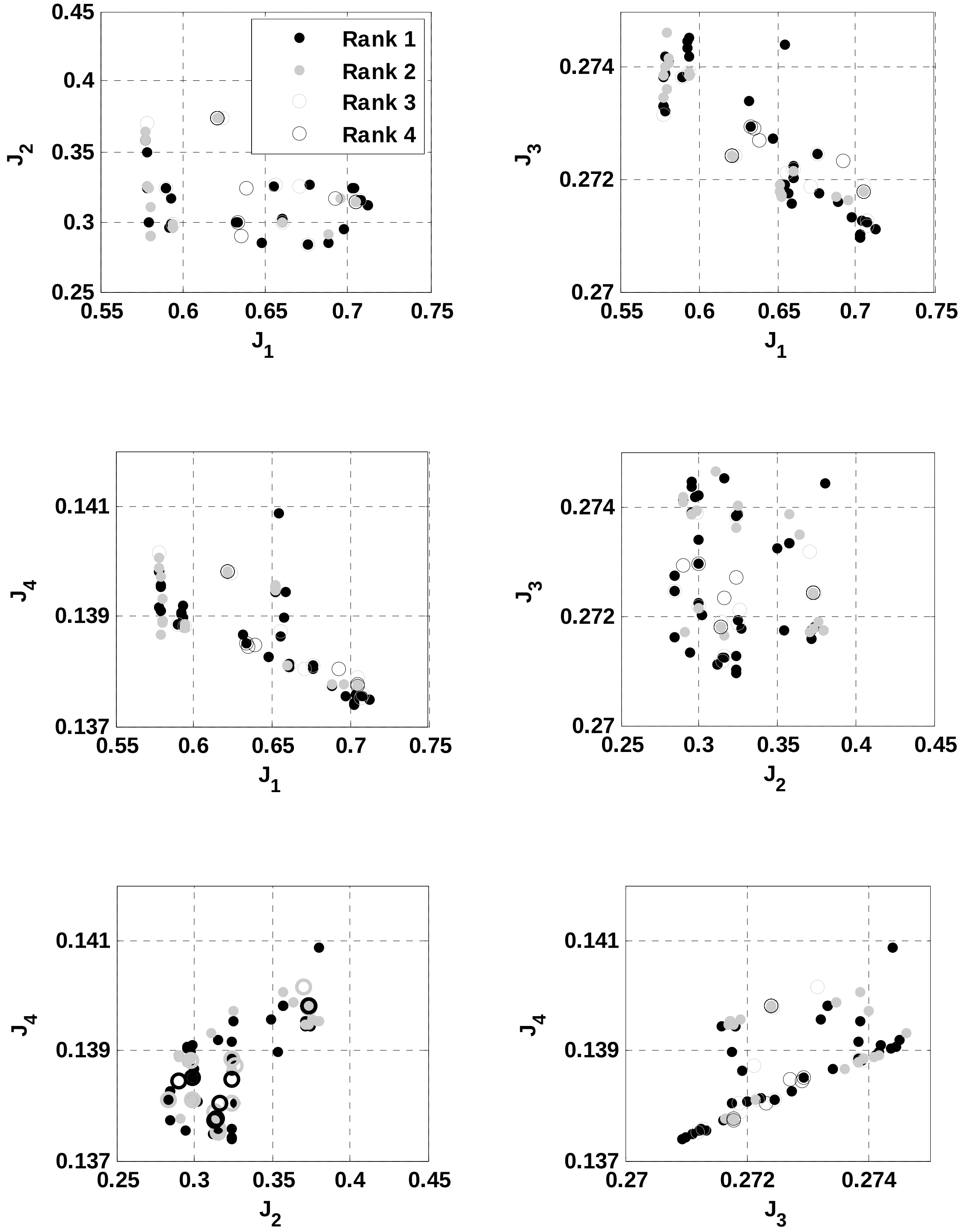


Fig. 62. Pareto Fronts Resulting from MIMO GA Optimization

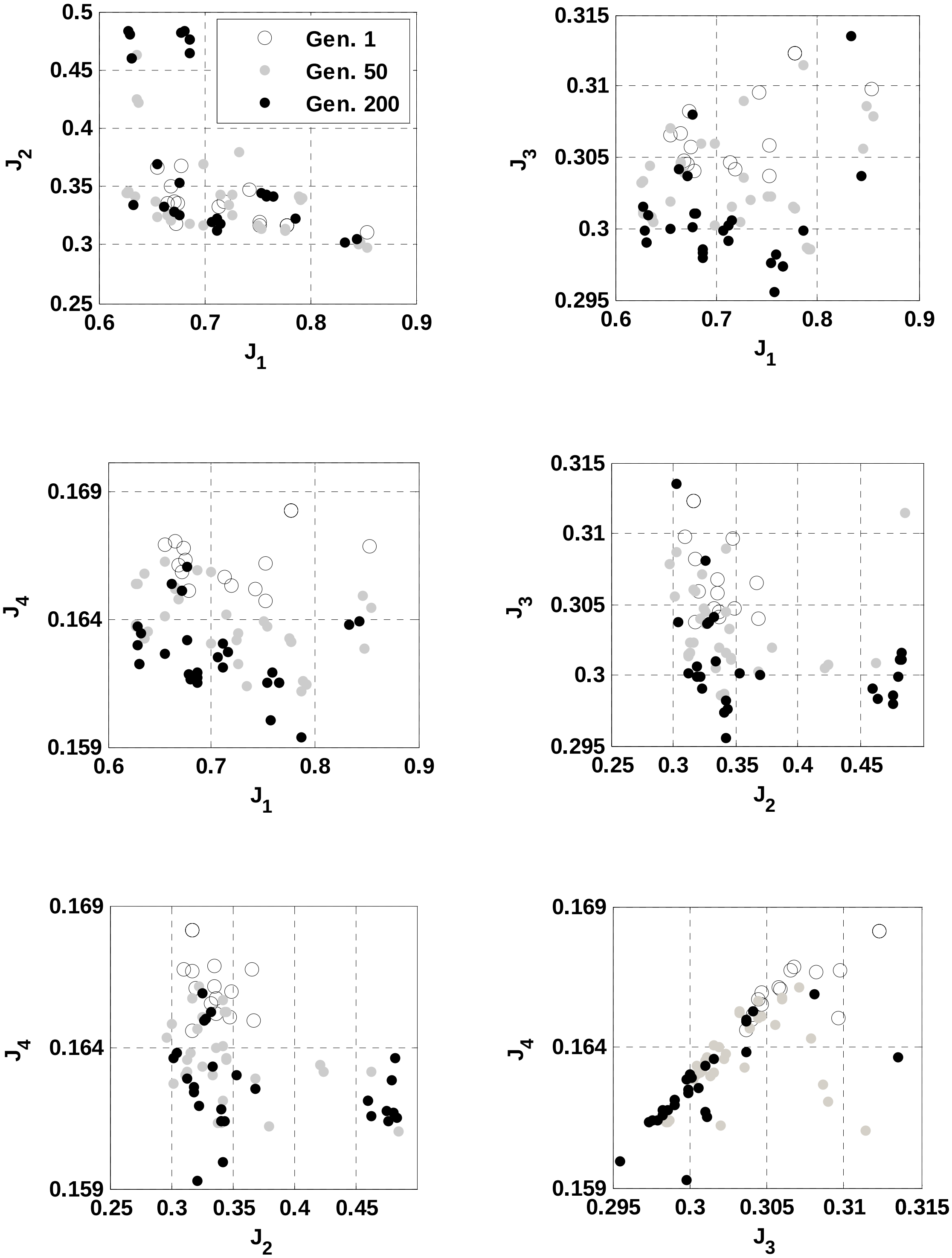


Fig. 63. Several Generations Resulting from MISO GA Optimization

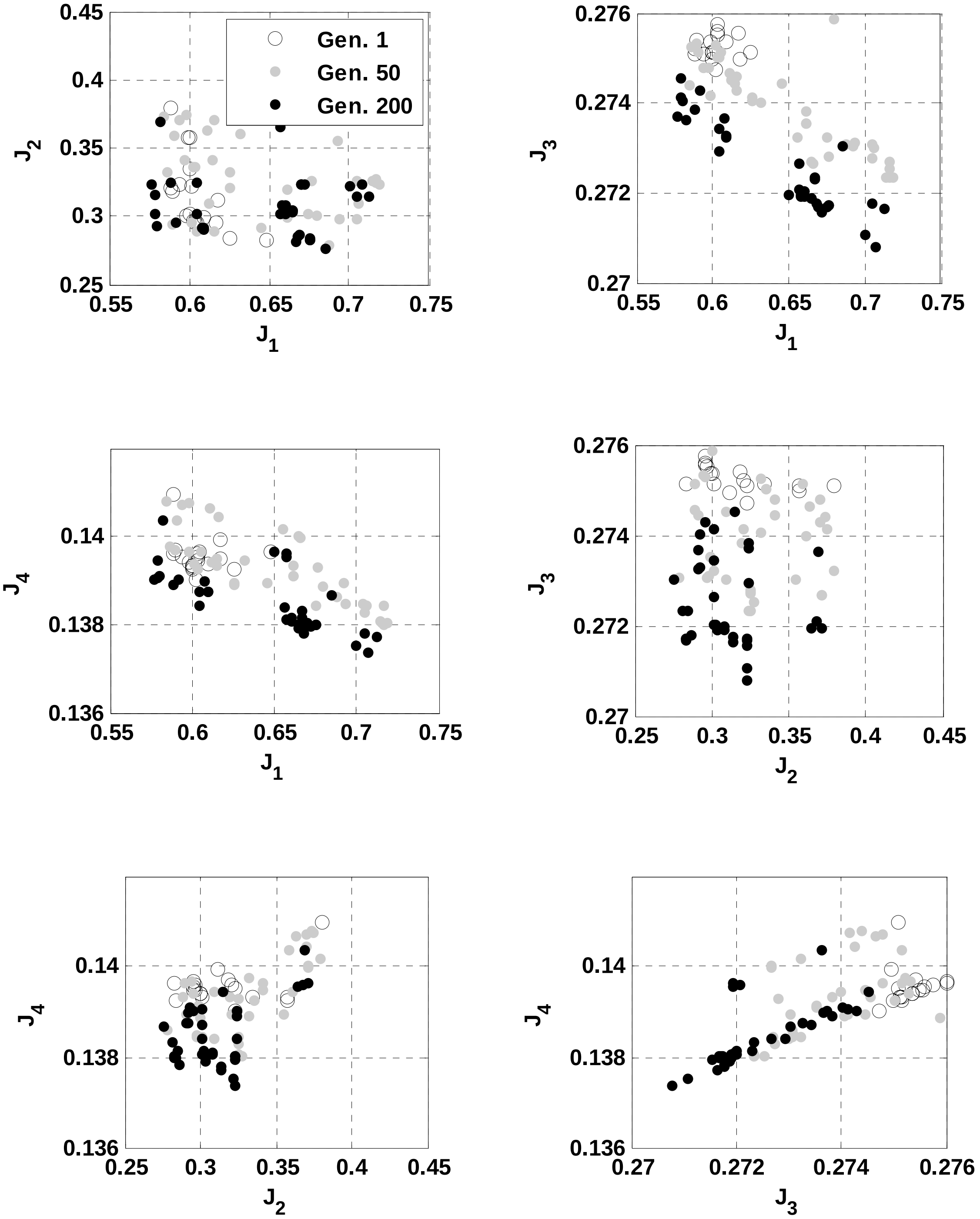


Fig. 64. Several Generations Resulting from MIMO GA Optimization

By observation of results from GA optimization two optimal controllers are identified that correspond to the MISO and MIMO cases (see Figs. 65 and 66). The controllers are selected by equal merit of all four objectives; that is, the controller might not perform with global optimal results for all objectives. Yet, the identified controllers possess favorable results considering all four objectives.

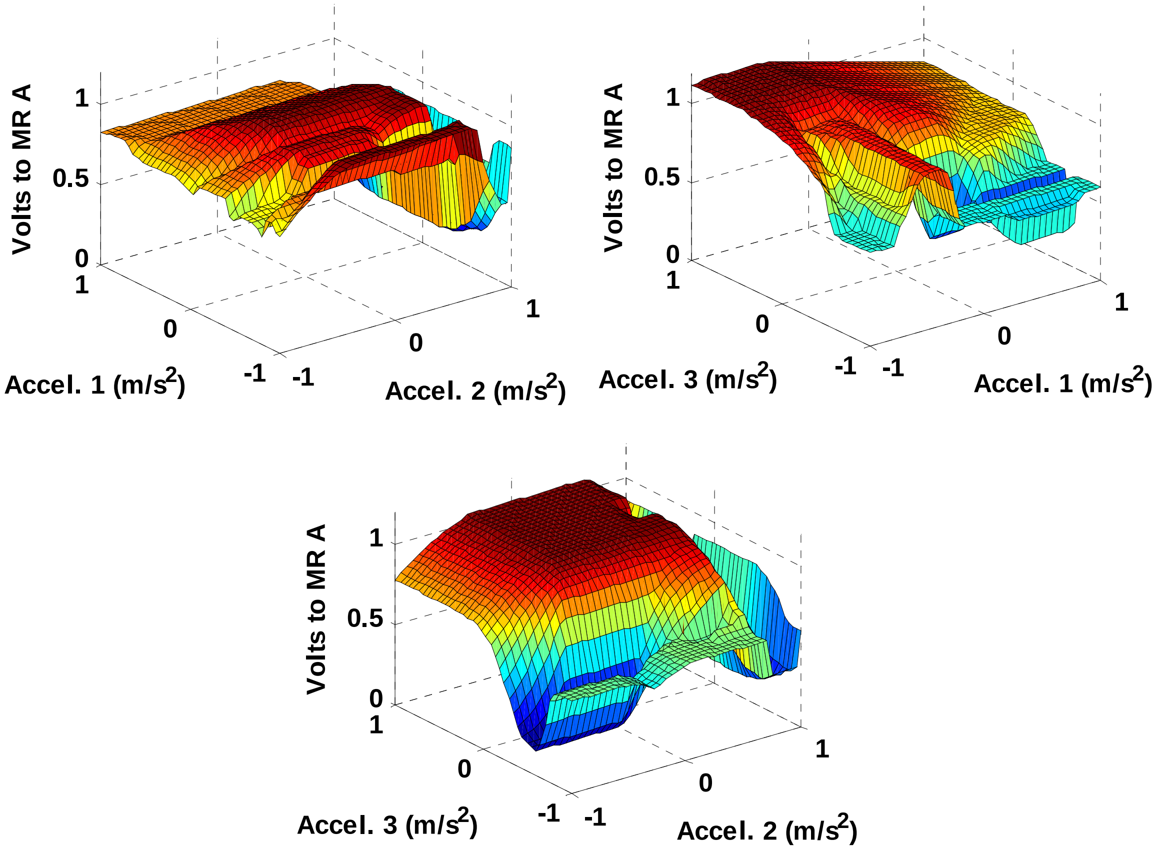


Fig. 65. MISO Control Surfaces

Fuzzy logic controllers can be partially described by a set of surfaces where input and output relationships are shown graphically. Figs. 65 and 66 display all potential fuzzy surfaces of MISO and MIMO controllers. Although these three-dimensional surfaces aid in understanding each controller, it is not possible to fully realize their complete combination visually. However, this set of surfaces is adequate for a control engineer to understanding its fundamental aspects.

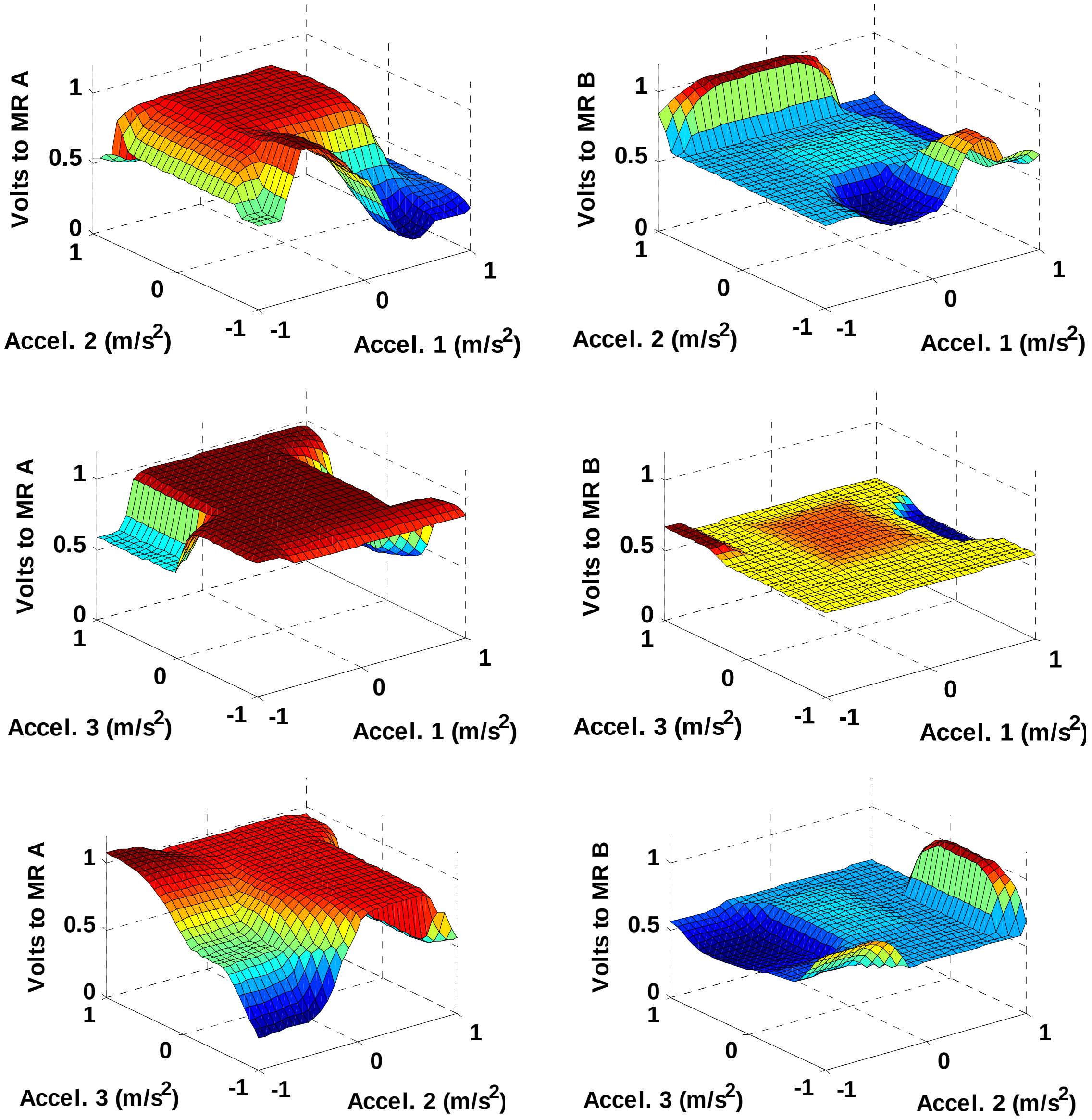
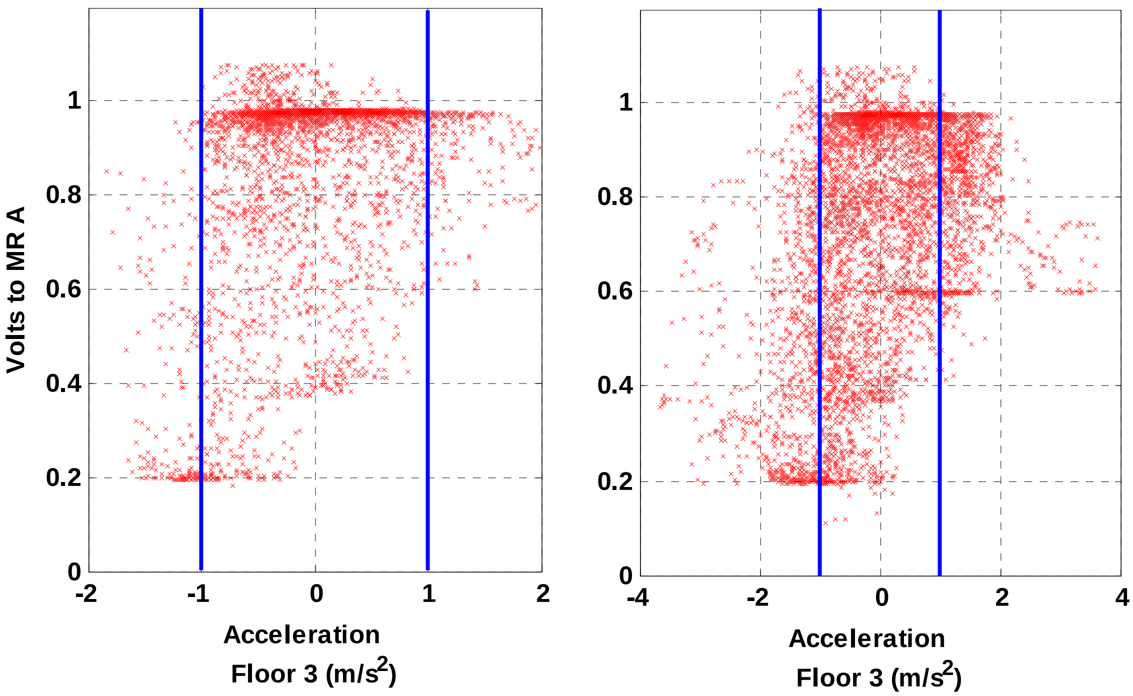


Fig. 66. MIMO Control Surfaces

As observed in Section 3.5, an optimal passive operating voltage of approximately 1.0 V exists with consideration for displacement and acceleration responses. Although this information is not imparted to the FLCs by the user, GA optimization is able to identify these optimal passive voltages without user guidance. This can be verified by observation of the prominent monotonic sections near 1.0 V of all FLC surfaces shown in Figs. 65 and 66 for both MISO and MIMO cases for MR damper *A*. In the MIMO case an optimial passive operational voltage of appoximatly 0.5 V is identified. Moreover, GA optimization is able to improve upon all of these passive voltages by altering the command signal as demonstrated by the non-monotonic section of the fuzzy surfaces in these figures. Thus, GA-optimization ascertained that for some input cases a passive voltage provides optimal reduction to displacement and acceleration responses, but in many other cases the intelligent modulation of voltages can improve the response of the structure beyond any single passive case.

To observe FLC output signals under experimental operation Fig. 67 has been generated to observe, in a simplified sense, FLC input/output relationships from experimental testing of MISO case under a 100 gal TCU076 earthquake and MIMO under a 200 gal TCU082 earthquake.. Here the third input of the FLC, acceleration of the 3rd floor, is plotted against the voltages specified for MR damper *A* for both MISO and MIMO cases. For the MISO case in Fig. 67(a) a higher consentration of specified voltages are located near 1.0 V. This is expected due to the monotonic regions of the fuzzy surfaces near 1.0 V as shown in Fig. 65(a). A more even distribution of specified voltages is observed in the MIMO case in Fig. 65(b). It is important to monitor these relationships to further understand GA-optimized FLCs and how effective each region of the FLC under is varing excitions. The input ranges of the FLCs are chosen to be ± 1 m/s^2. For the 100 gal case this range is reasonable, but for the 200 gal case the accelerations extend significantly beyond the input ranges of the FLC. To formulate a more robust FLC, extention of the input ranges would help facilitate a more robust controller when higher amplitude tremblors occur. Yet, as shown later, good results have been obtained with input ranges being limited to ± 1 m/s^2.

In what follows results and discussion relating to the numerical and experimental evaluation of GA-optimized FLCs are provided. In the first two sections results pertinent to numerical simulations and experimental trials are examined through tabulated performance metrics and visual inspections of plotted data. In the third section results from numerical simulation and experimental trials are pooled to observe overall performance of the FLC controllers. Note that the GA-optimized FLC was trained on an artificial seismic record as discussed in Section 3.3. In the following sections only seismic records that were not part of the training records are used for investigation of FLC performance.



(a) (b)

Fig. 67. Input/Output of Controllers in (a) 100 gal and (b) 200 gal excitations

6.2.1. Results of Numerical Simulations

An evaluation of MISO and MIMO performance is first conducted using numerical simulations. Numerical simulations are computed by use of MATLAB/Simulink (2006) code that is listed in Appendix C. In Figs. 68 through 82 time histories of floor displacement and acceleration as well as hysteresis plots of MR damper force are provided to allow a visual inspection of the controlled, passive-off, and FLC results from numerical simulation of the Kobe, El Centro, and Chi-Chi earthquakes (denoted as stations TCU076 and TCU082). The Kobe and El Centro seismic records are generally classified as near-field excitations while the two Chi-Chi seismic records utilized are generally classified as far-field excitation records. All displacement and acceleration values are shown relative to the base of the structure.

Figs. 68 through 71, and 74 through 77 show time history results considering four earthquakes that are simulated at various intensity levels. For all floors the passive-off case shows great improvement upon the uncontrolled case for both displacement and acceleration responses as shown by the Kobe excitation in Figs. 68 and 69. FLC controlled cases show further improvement upon the passive-off case with respect to displacement responses for both 100 and 200 gal excitation levels. Although not noted in the time histories of the 2nd and 3rd floors, the inter-story drifts of all cases were minimized such that plastic deformations do not occur. This linear deformation is later substantiated by experimental trials.

Acceleration reductions are notable, but are generally not as significant as for the displacements. This is a very positive result of the FLC since acceleration responses are not aggravated as is common in many structural control applications. As observed in Figs. 69 and 75 the 1st floor acceleration response of the FLC case and uncontrolled case are similar, but when considering the 3rd floor response the reduction is more substantial. This top floor reduction of acceleration response is crucial in structural engineering applications were high accelerations on the top floor of a building often cause damage to non-structural components such as air conditioning ducts, electrical conduits, and water supply lines. What is more, in modest excitations personnel located on the top floor of a building might become sick due to a sudden increase in acceleration. Thus, the FLC is effective in reducing the global acceleration response of the building as considered in the optimization objectives of Eqs. 12 and 14.

As an aid to understanding the operational characteristics of MR damper *A* during the simulated earthquakes, Figs. 72, 73, 78, and 79 show hysteresis plots of force versus displacement and force versus velocity. From these figures the effect of FLC control on the magnitude of the displacement and velocity is apparent by their reduction. Conversely, the MR forces corresponding to FLC control clearly increase over those predicted during passive operation. For the displacement versus force figures it is interesting to note that the shapes of the passive and FLC curves are rotated approximately 90 degrees from each other. Safety limitations for the MR damper stroke are considered in its peak displacement as noted earlier in Table 6. Here the peak displacements experienced by the MR damper are well within the ± 240 mm stroke limitation. The 'classical' *S*-shape of the force-velocity curve is readily apparent for the passive-off hysteresis curves. By contrast, the initial slope of the force-velocity curves is much steeper than for the passive-off case; however the horizontal portion of the FLC curves is not well-developed because the velocity is reduced in comparison with the passive-off operation. The full capacity of the MR damper is not fully realized in numerical simulation since the peak MR damper force is approximately 15 kN except in the 200 gal Chi-Chi (TCU082) case as shown in Fig. 79. Thus, for the majority of cases reserve capacity of the MR damper is retained.

Data presented in Table 8 correspond to numerical simulation results for both MISO and MIMO cases of control as outlined in Eqs. 11-14. Results from two control cases are shown: passive-on ("P-ON") and fuzzy logic controlled ("FLC"). Passive-off information is implicitly included since both passive-on and FLC results are normalized by the corresponding passive-off value; thus, if the value of the performance index is greater than 1, passive-off is the superior case. Uncontrolled data are not provided since they do not substantiate effectiveness of the controller and they could not be attained for excitations greater than 100 gal because of the possibility of large strains causing damage to the steel columns. Italicized values denote FLC performance that is superior to passive-off control while bolded values denote results superior to both passive results. The optimal passive case of 1.0 V being applied to the MR dampers is not included since it was not tested during the experimental trials.

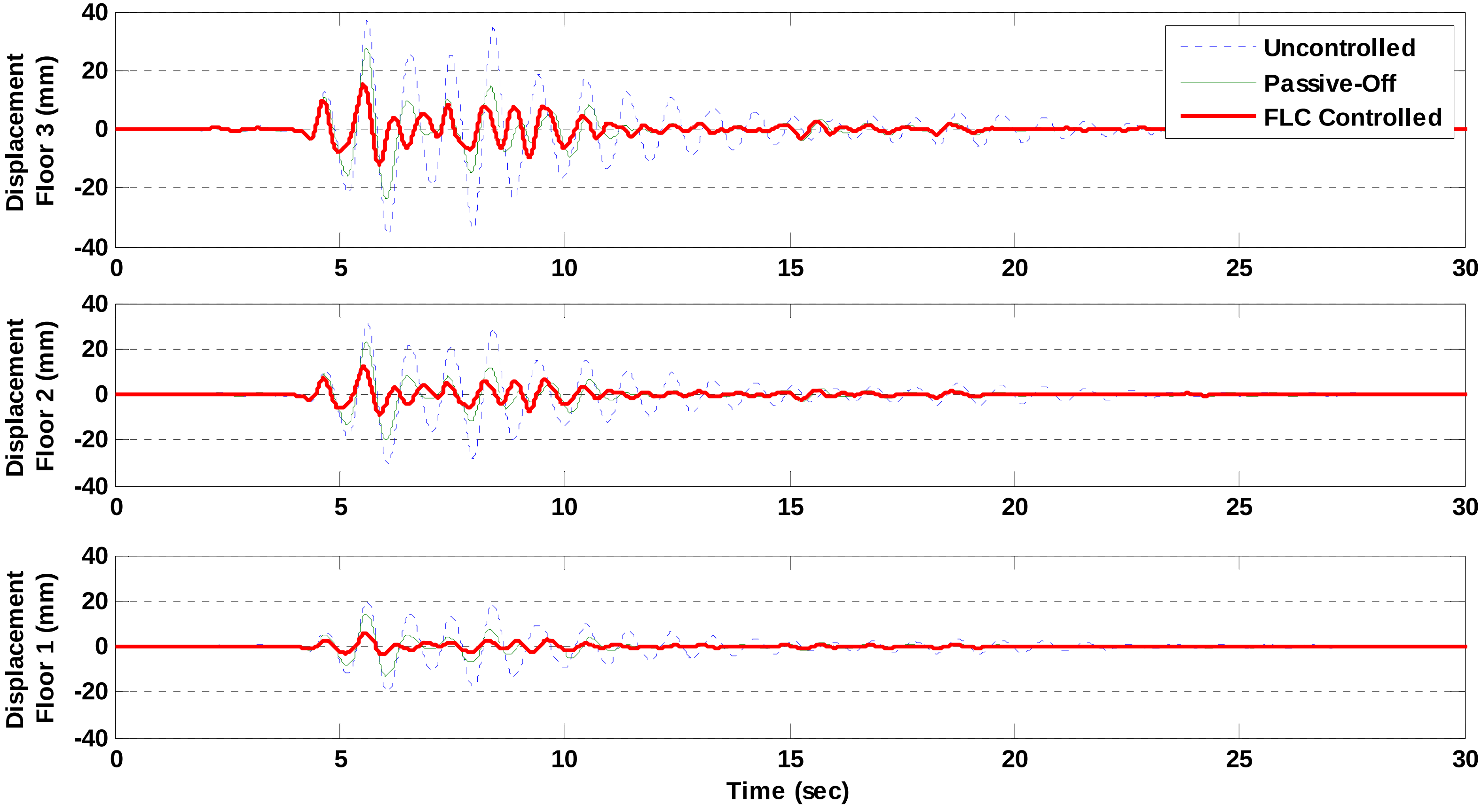


Fig. 68. Numerical Simulation of MISO: Displacement from 100 gal Kobe Earthquake

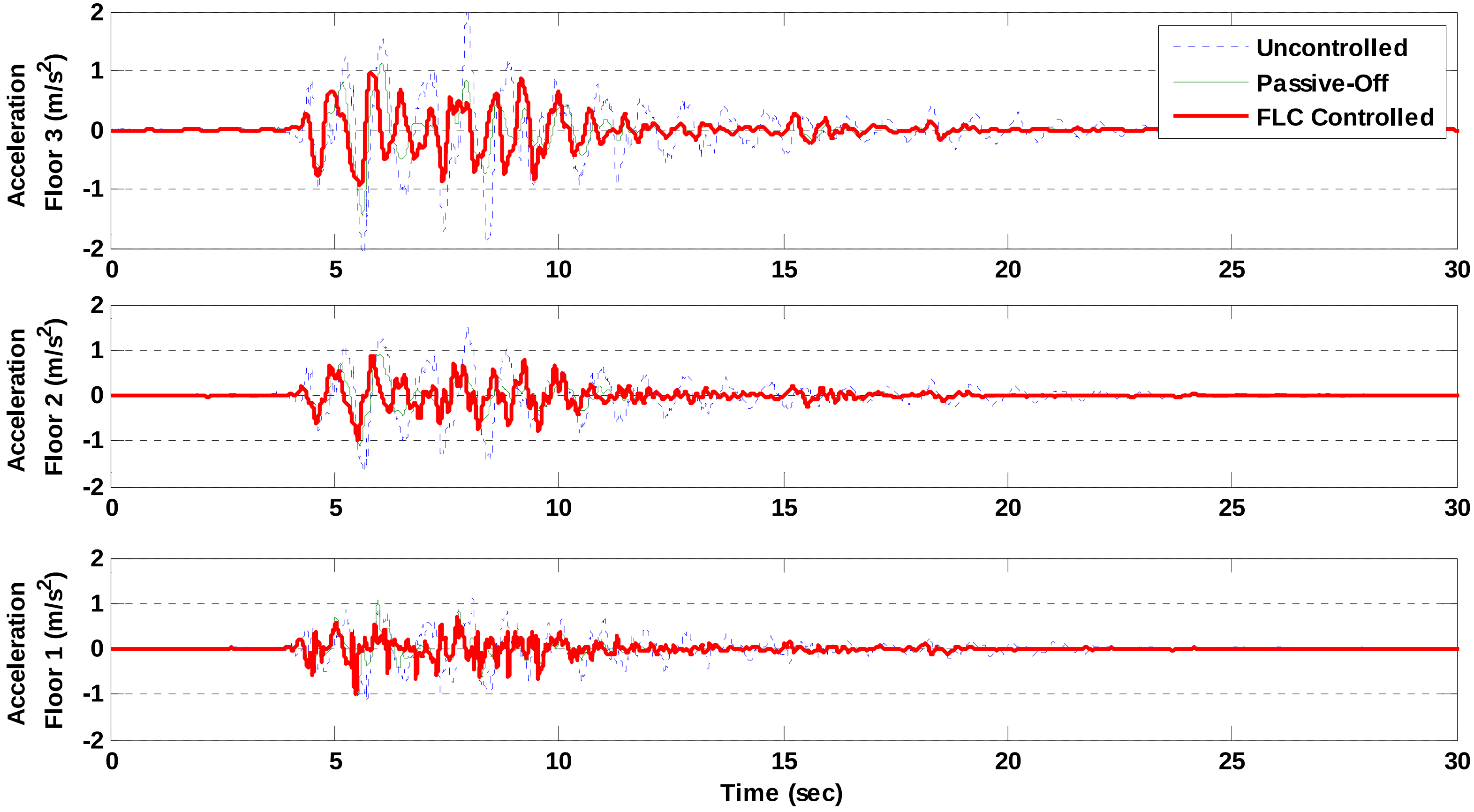


Fig. 69. Numerical Simulation of MISO: Acceleration from 100 gal Kobe Earthquake

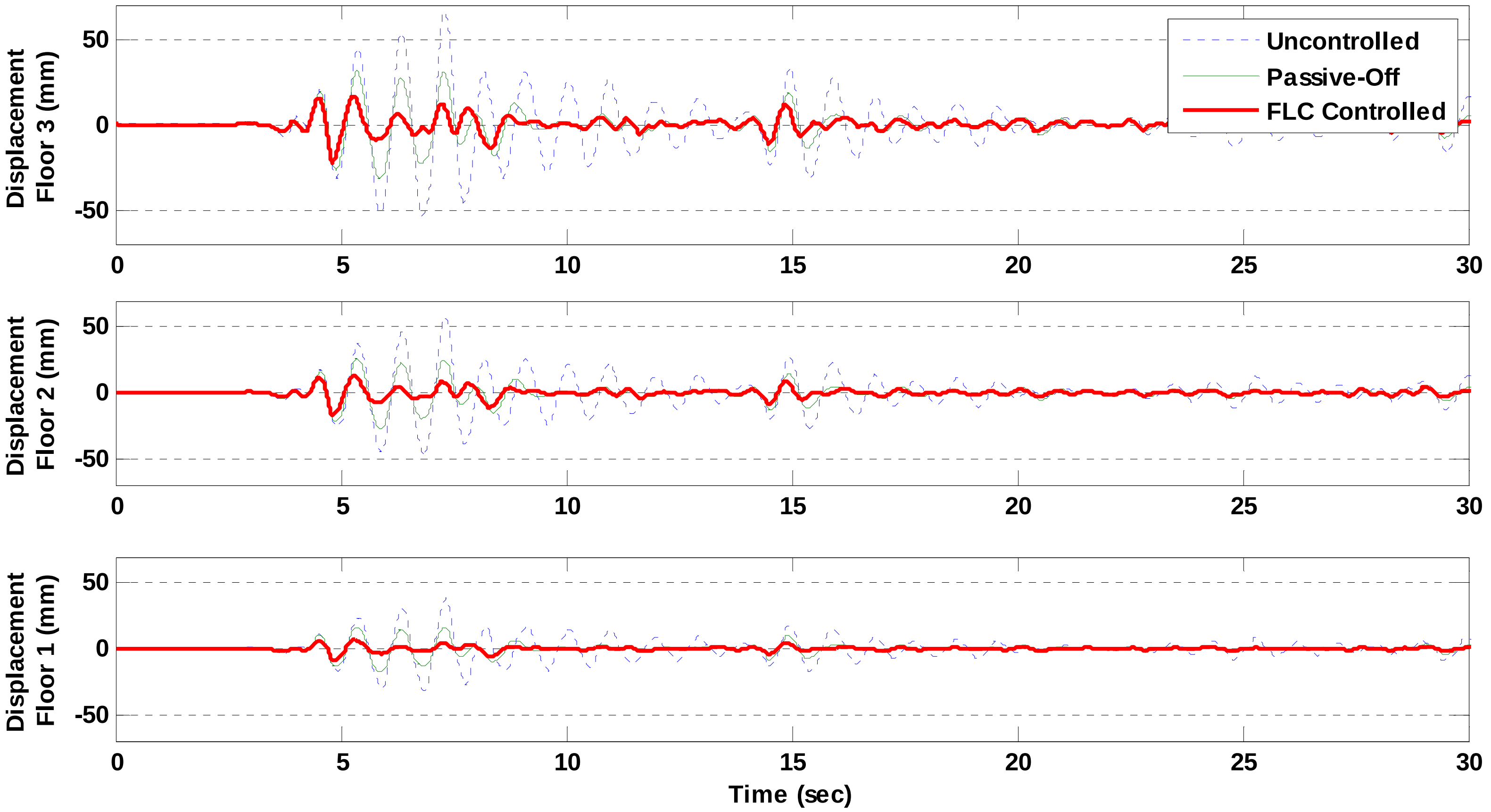


Fig. 70. Numerical Simulation of MIMO: Displacement from 200 gal El Centro Earthquake

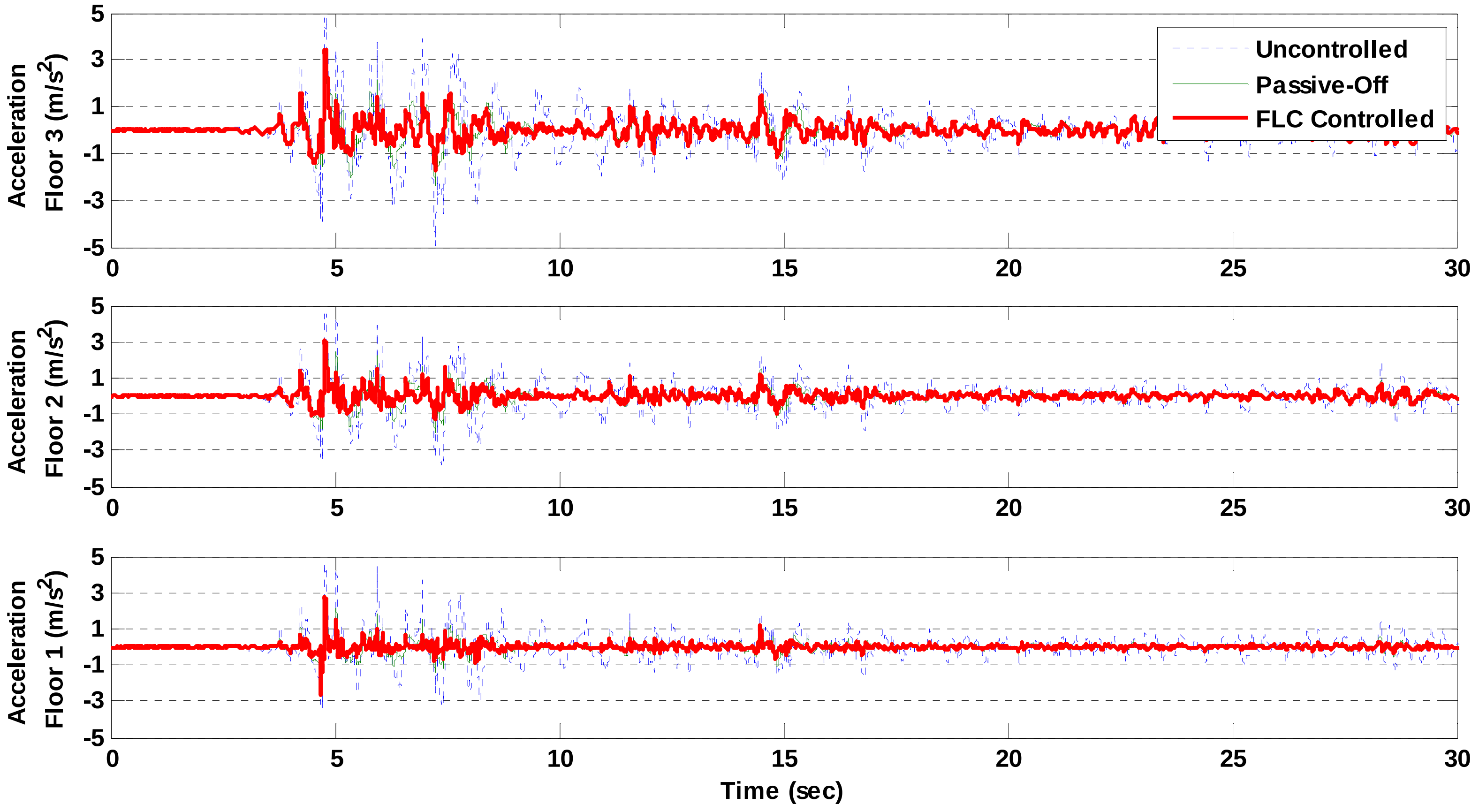


Fig. 71. Numerical Simulation of MIMO: Acceleration from 200 gal El Centro Earthquake

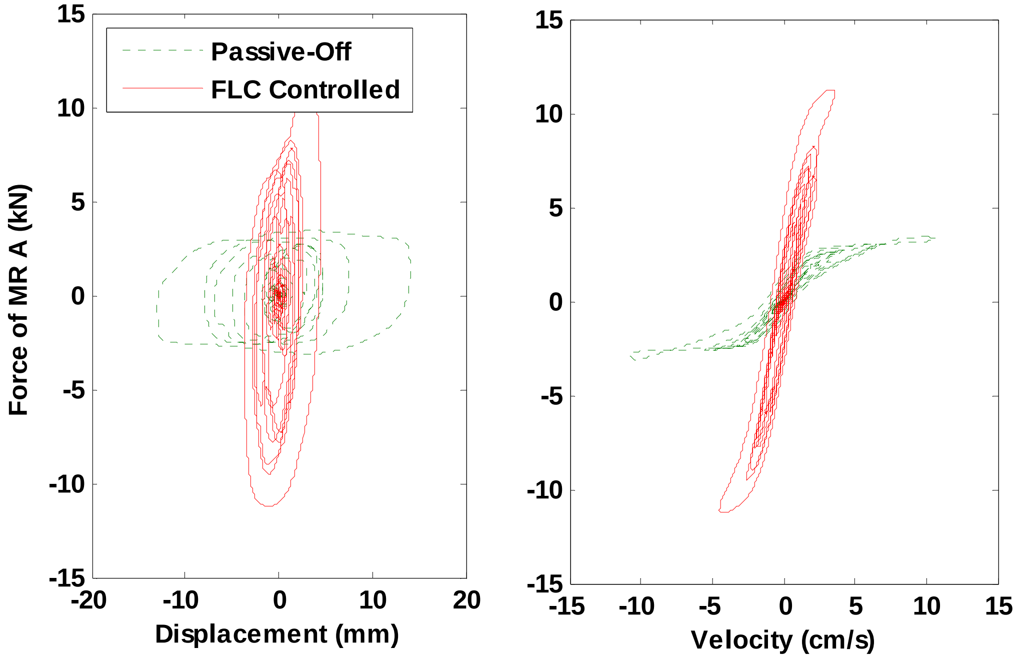


Fig. 72. Numerical Simulation of MISO: Hysteresis from 100 gal Kobe Earthquake

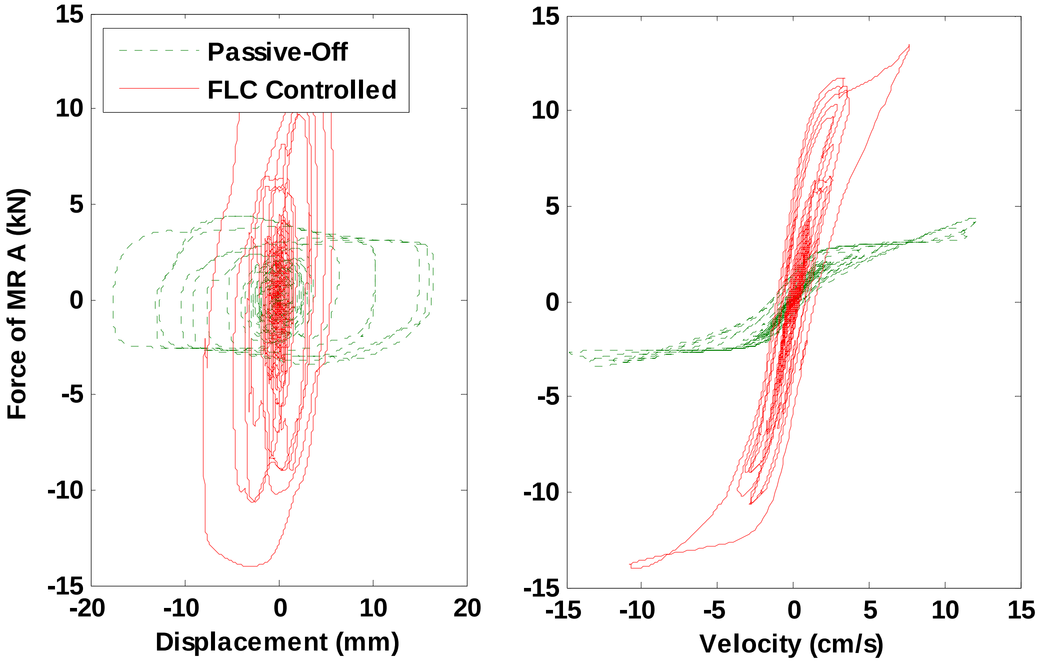


Fig. 73. Numerical Simulation of MIMO: Hysteresis from 200 gal El Centro Earthquake

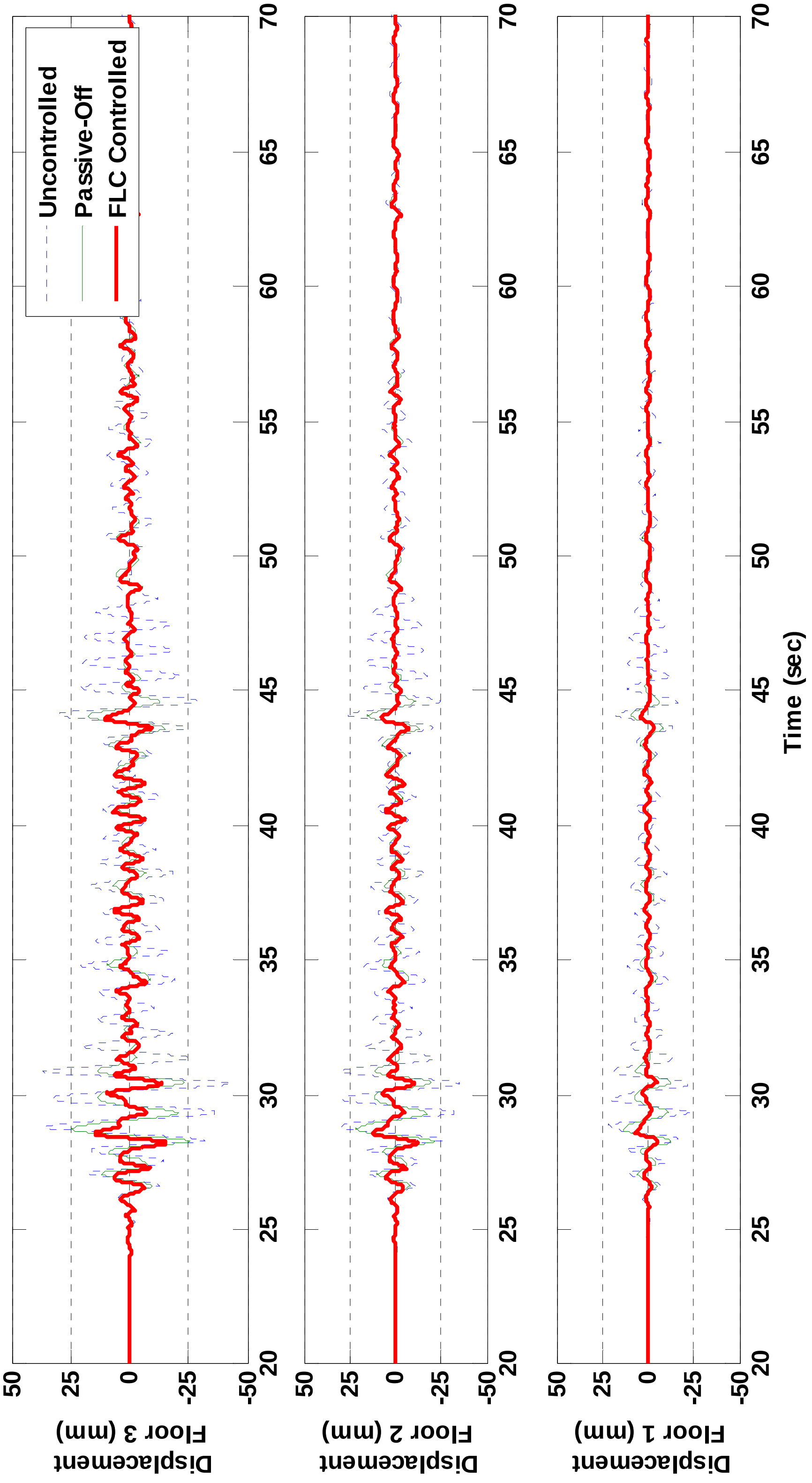


Fig. 74. Numerical Simulation of MISO: Displacement from 100 gal TCU076 Earthquake

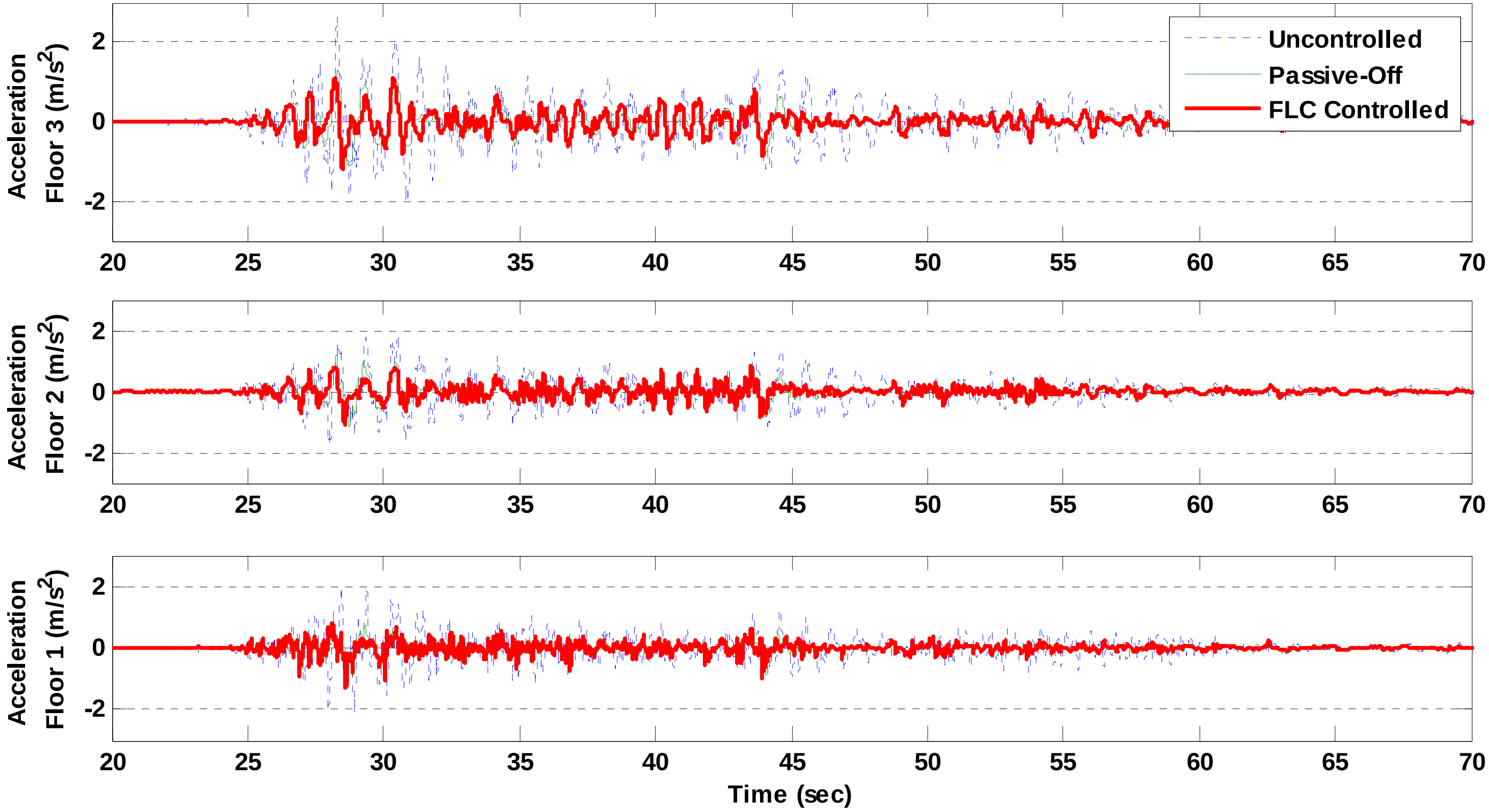


Fig. 75. Numerical Simulation of MISO: Acceleration from 100 gal TCU076 Earthquake

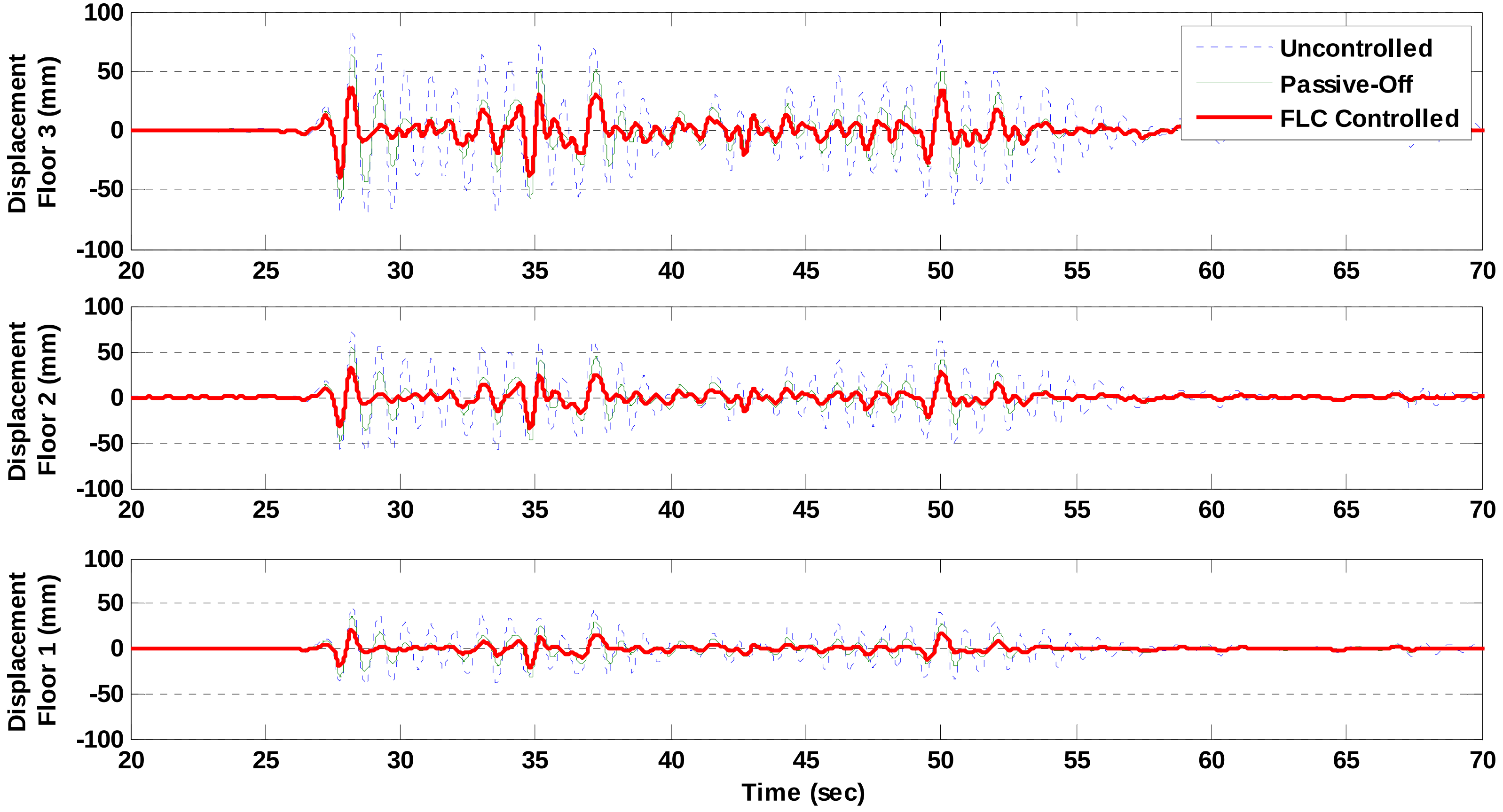


Fig. 76. Numerical Simulation of MIMO: Displacement from 200 gal TCU082 Earthquake

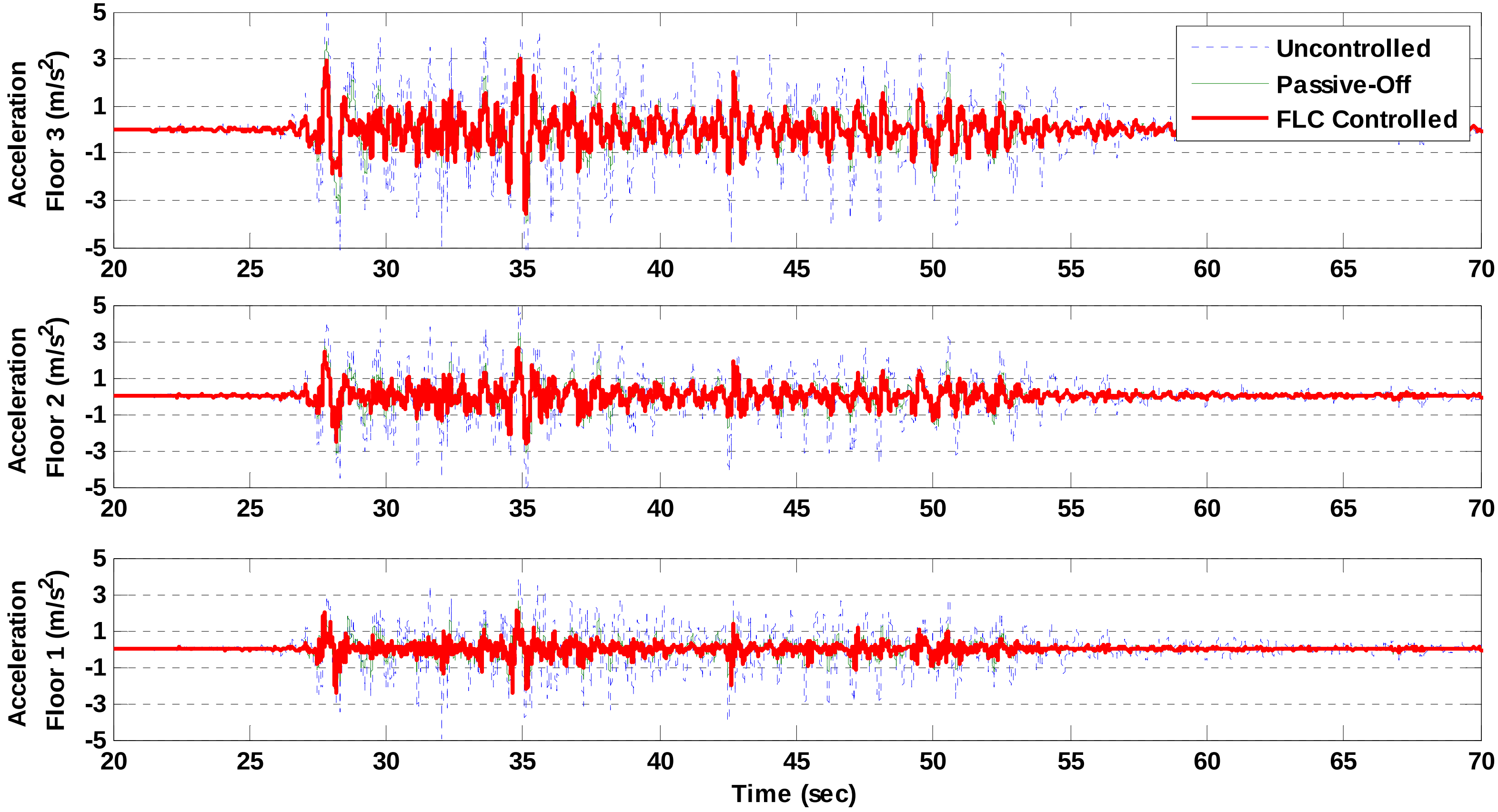


Fig. 77. Numerical Simulation MIMO: Acceleration from 200 gal TCU082 Earthquake

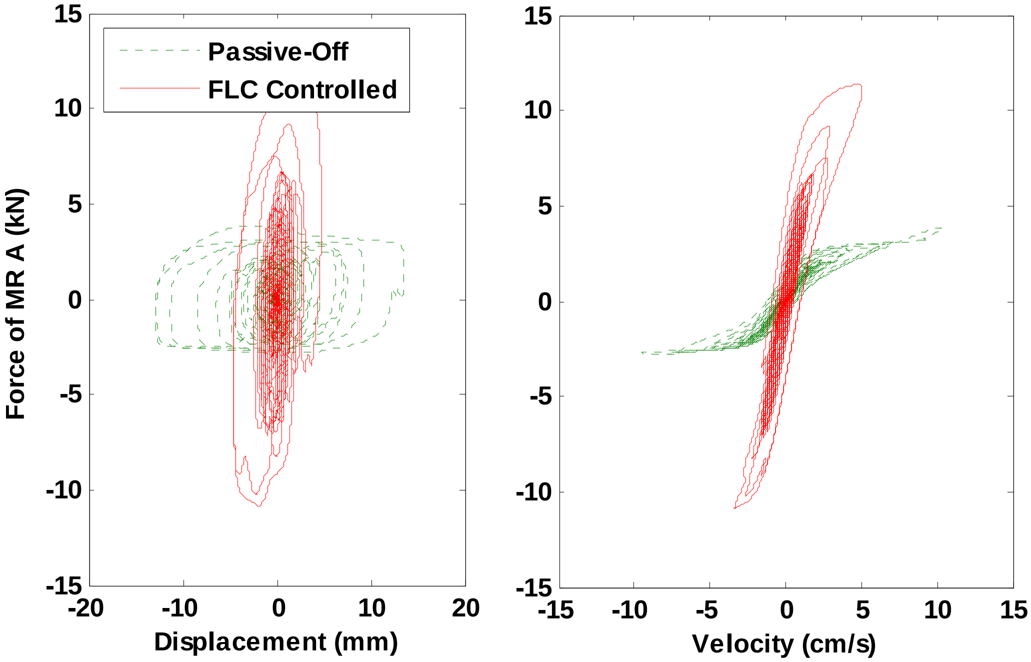


Fig. 78. Numerical Simulation of MISO: Hysteresis from 100 gal TCU076 Earthquake

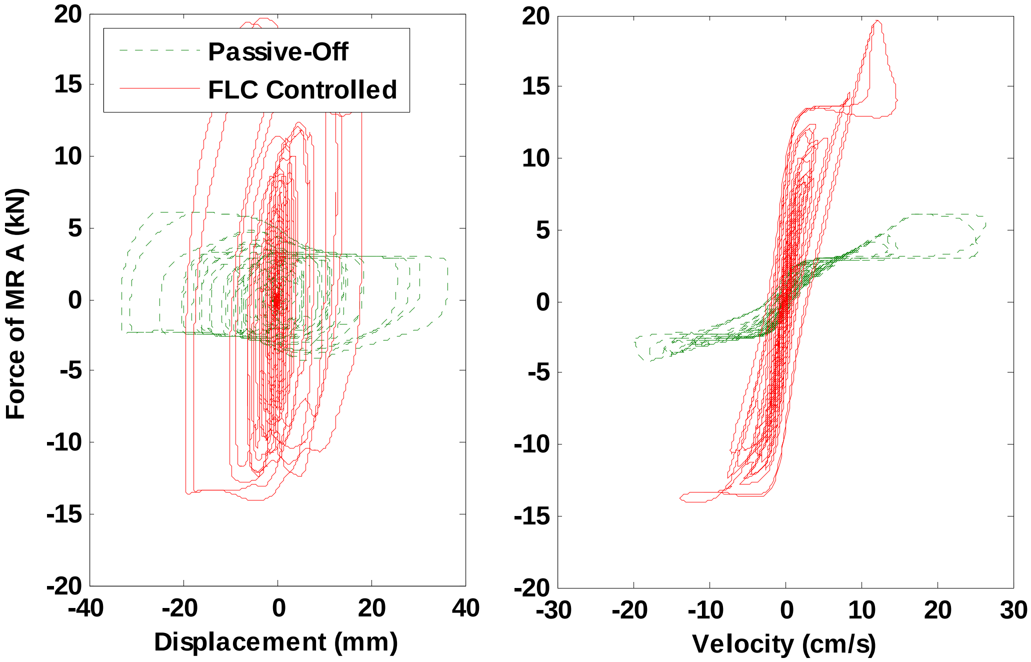


Fig. 79. Numerical Simulation of MIMO: Hysteresis from 200 gal TCU082 Earthquake

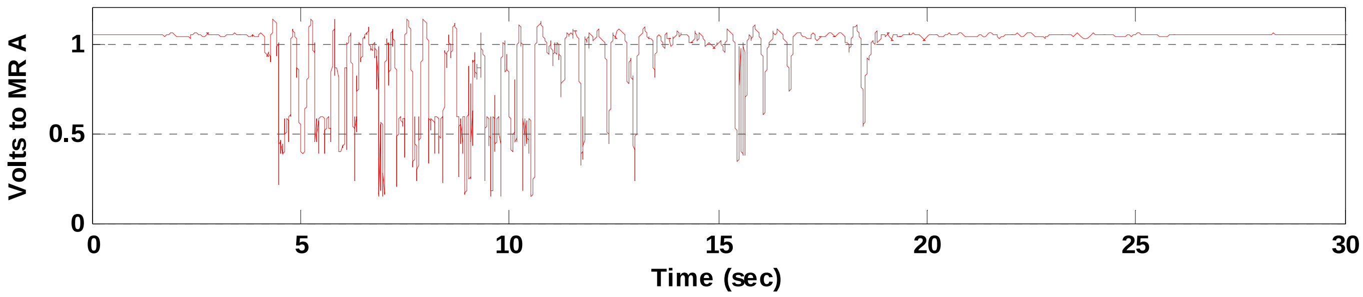


Fig. 80. Numerical Simulation of MISO: Voltage from 100 gal Kobe Earthquake

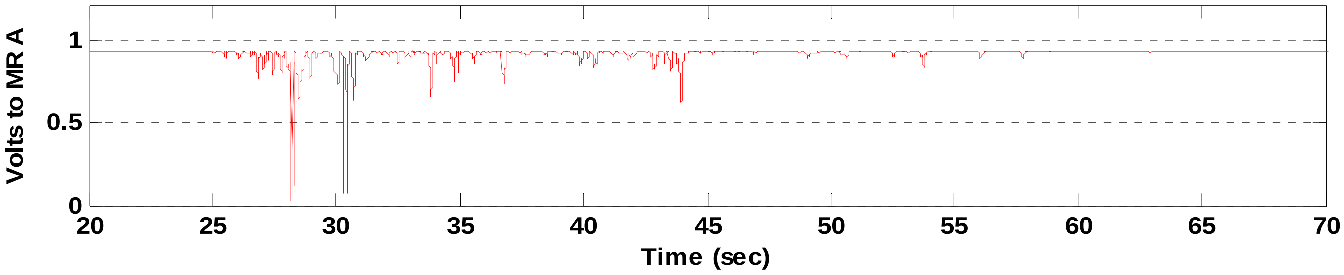


Fig. 81. Numerical Simulation of MISO: Voltage from 100 gal TCU076 Earthquake

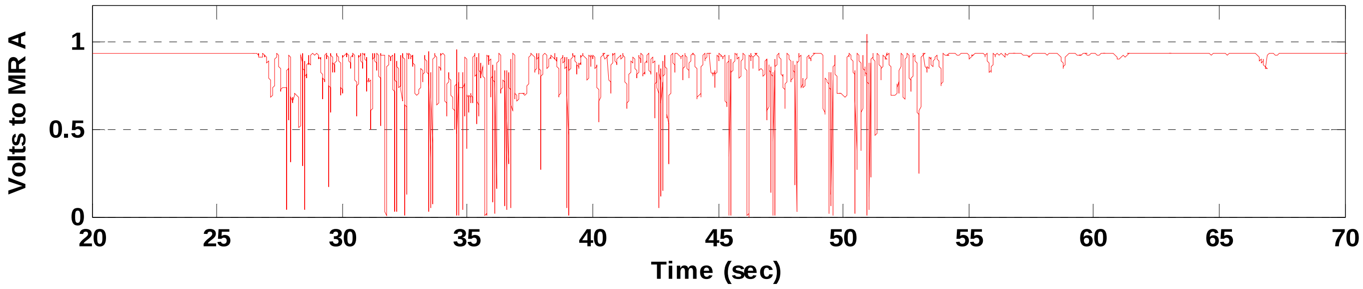


Fig. 82. Numerical Simulation of MIMO: Voltage from 200 gal TCU082 Earthquake

Table 8. Numerical Evaluation of FLC Controllers

		MISO Configuration				MIMO Configuration			
		J_1	J_2	J_3	J_4	J_1	J_2	J_3	J_4
El Centro (100 gal)	P-ON	0.85	1.30	0.65	1.07	0.76	1.29	0.93	0.98
	FLC	**0.79**	1.27	**0.64**	1.01	**0.74**	1.34	**0.92**	**0.97**
Kobe (100 gal)	P-ON	0.53	0.74	0.63	1.08	0.49	0.86	0.92	0.98
	FLC	*0.55*	**0.68**	**0.61**	1.01	*0.50*	1.17	**0.89**	**0.93**
Chi-Chi, TCU076 (100 gal)	P-ON	0.58	0.99	0.59	1.01	0.55	0.96	0.91	0.95
	FLC	*0.61*	1.01	*0.60*	**0.96**	*0.56*	1.52	**0.89**	*0.96*
Chi-Chi, TCU082 (100 gal)	P-ON	0.63	1.13	0.66	1.10	0.53	1.08	0.98	1.02
	FLC	**0.59**	1.10	*0.66*	1.03	*0.56*	1.72	**0.95**	**0.99**
El Centro (200 gal)	P-ON	-	-	-	-	0.68	1.16	0.83	0.89
	FLC	-	-	-	-	**0.62**	1.68	**0.80**	*0.95*
Kobe (200 gal)	P-ON	-	-	-	-	0.58	1.23	0.74	0.82
	FLC	-	-	-	-	*0.72*	1.49	**0.73**	*0.92*
Chi-Chi, TCU076 (200 gal)	P-ON	-	-	-	-	0.58	1.38	0.73	0.85
	FLC	-	-	-	-	*0.64*	1.20	**0.72**	*0.92*
Chi-Chi, TCU082 (200 gal)	P-ON	-	-	-	-	0.53	1.08	0.98	1.02
	FLC	-	-	-	-	*0.56*	1.72	**0.95**	**0.99**

From these tabulated results of numerical simulation performance of the fuzzy logic controllers are superior to both passive cases for many cases over a variety of excitations, objectives, and MR damper configurations. The fuzzy logic controller for the MIMO case is superior to both passive cases concerning objective J_3 over all excitations while maintaining favorable performance over the remaining objectives. For almost all cases fuzzy logic control is superior to both passive cases for at least one or more objectives. This shows that multi-objective optimization is effective in reducing all objectives.

6.2.2. Results of Experimental Trials

In this subsection experimental evidence of FLC performance is reported. Experimental testing of the FLC controllers utilizing the benchmark structure with installed MR dampers was conducted at the NCREE laboratory in June 2006.

Figs. 83 through 97 present results from experimental testing of the benchmark structure with an installed control system. Several time histories and hysteretic plots are shown with uncontrolled, passive-off, and FLC controlled cases for 100 gal excitation and passive-off and FLC controlled for 200 gal excitations. No uncontrolled 200 gal experimental trials were conducted due to potential plastic deformations that could occur in the benchmark columns under such loads. Excitation levels greater than 100 gal were investigated after installation of the MR dampers due to their additive resistance to motion. Corresponding with results of numerical simulations, FLC controlled displacement reductions are considerable when compared to the uncontrolled case, but acceleration reduction are not as large. Yet, an increase in accelerations above the uncontrolled case is not observed.

In experimental trials a higher frequency component of acceleration is observed that was not realized in numerical simulations. This is very apparent in the acceleration response of the 1st floor of the benchmark structure excited by the Chi-Chi (TCU076) earthquake as shown in Fig. 90 when compared to the numerical simulations shown in Fig. 75. Although this high frequency content is not numerically simulated and not incorporated into GA optimization of the FLC, it has been experimentally shown that the FLC of MR dampers can effectively mitigate this unexpected response.

Results from experimental testing of MISO and MIMO controllers validate performance of the fuzzy logic controller. When experimental results are compared to results of numerical simulation reasonably good agreement is indicated, but some non-trivial discrepancies exist. Differences of this magnitude are expected in real-world applications and yet are valid concerns in controller assessment. Despite this difficulty, the clemency of fuzzy logic shows that it can accommodate non-trivial errors during optimization and still perform favorably under a variety of excitations. Experimental results show FLC robustness with consideration of frequency content and the magnitude of excitation in the MIMO control case.

FLC specified voltages for MR damper *A* shown in Figs. 80, 81, and 82 correspond to numerical simulation, while Figs. 95, 96, and 97 correspond to MR damper voltages from experimental tests. In earlier efforts penalty functions were employed in GA optimization to ensure that the voltage specified by the FLC is small when the response of the building, as characterized by measured acceleration, is minimal. This was found to have an adverse effect on GA optimization of FLCs, resulting in poor performance of GA-generated FLCs. Although ideal, this philosophy was abandoned and penalty functions negated to ensure optimal performance of the FLC under high amplitude excitations. If these controllers were to be implemented in a real civil engineering structure a switching algorithm would need to be installed to bring the controller online when acceleration responses became significant; also, it would need to remain in an offline mode for the remainder of its service.

Experimental results of the response of the MR dampers to motion under seismic loadings are shown in Figs. 87, 88, 93, and 94. Here peak displacements of approximately ± 20 mm and peak velocities of approximately ± 20 cm/s are observed except for the 200gal TCU082 excitation shown in Fig. 94. This gives experimental evidence to the validity of selected in ranges for the neuro-fuzzy model of the MR dampers in Section 4 of ± 25 mm for displacement and ± 20 for velocity. Hence, the 'trimming' of training data is justified since experimental trials show that the MR damper generally does not experience displacement or velocities outside of the modeled range for 100 or 200 gal excitations. In numerical simulations where inputs to the neuro fuzzy model extend beyond the modeled ranges (as shown in Fig. 79) the inputs are truncated at the saturation limits of the neuro fuzzy model.

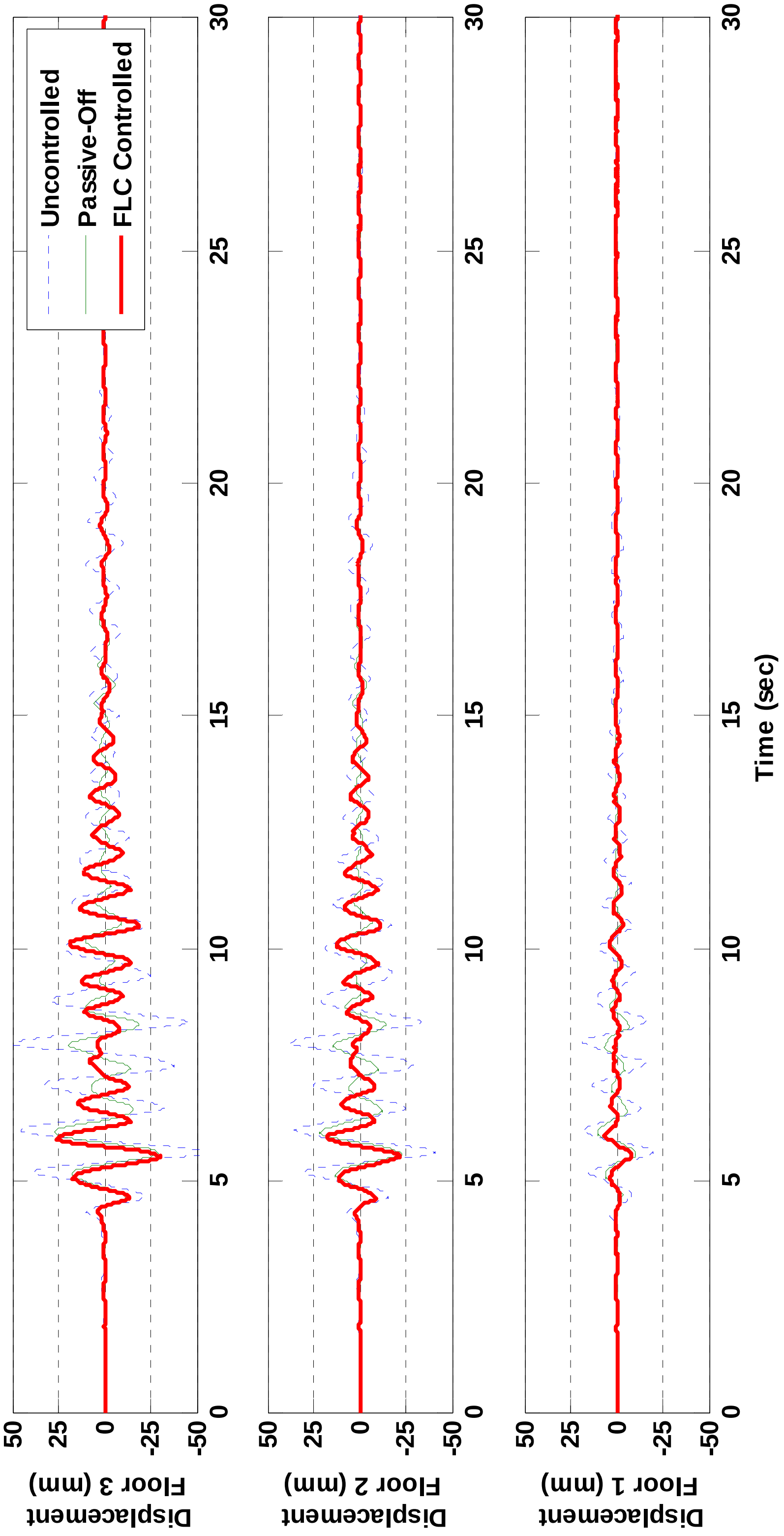


Fig. 83. Experimental Results of MISO: Displacement from 100 gal Kobe Earthquake

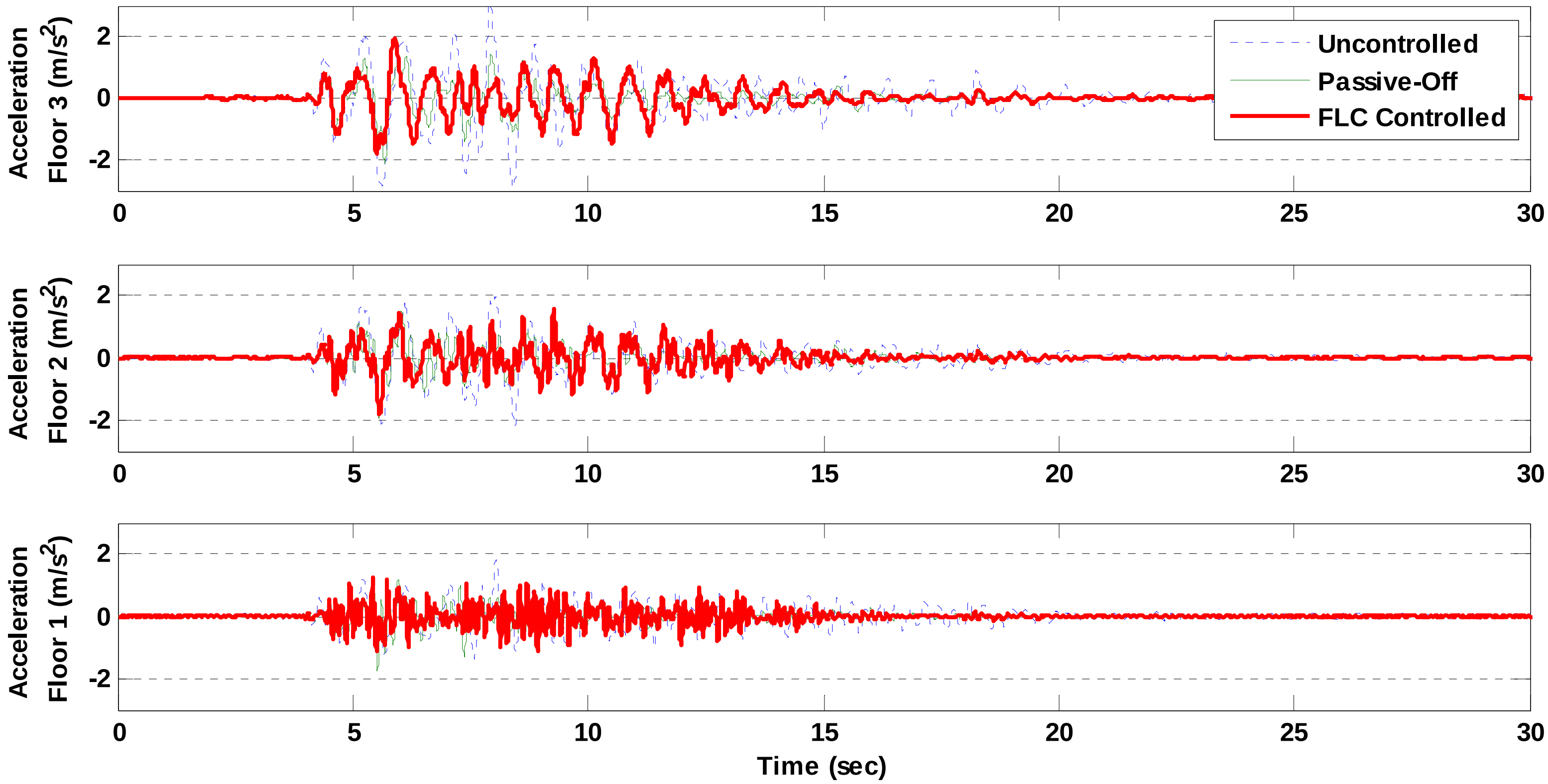


Fig. 84. Experimental Results of MISO: Acceleration from 100 gal Kobe Earthquake

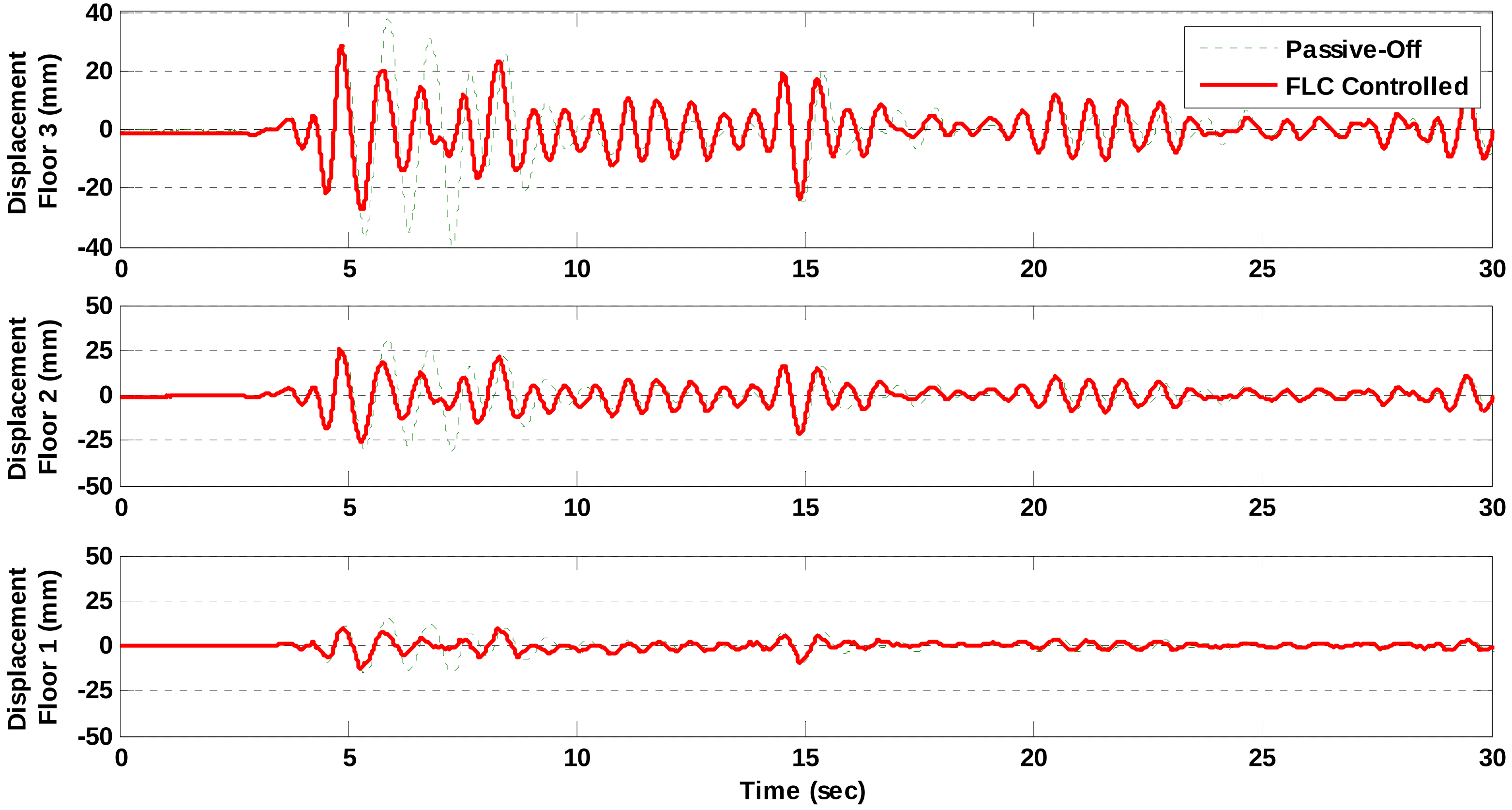


Fig. 85. Experimental Results of MIMO: Displacement from 200 gal El Centro Earthquake

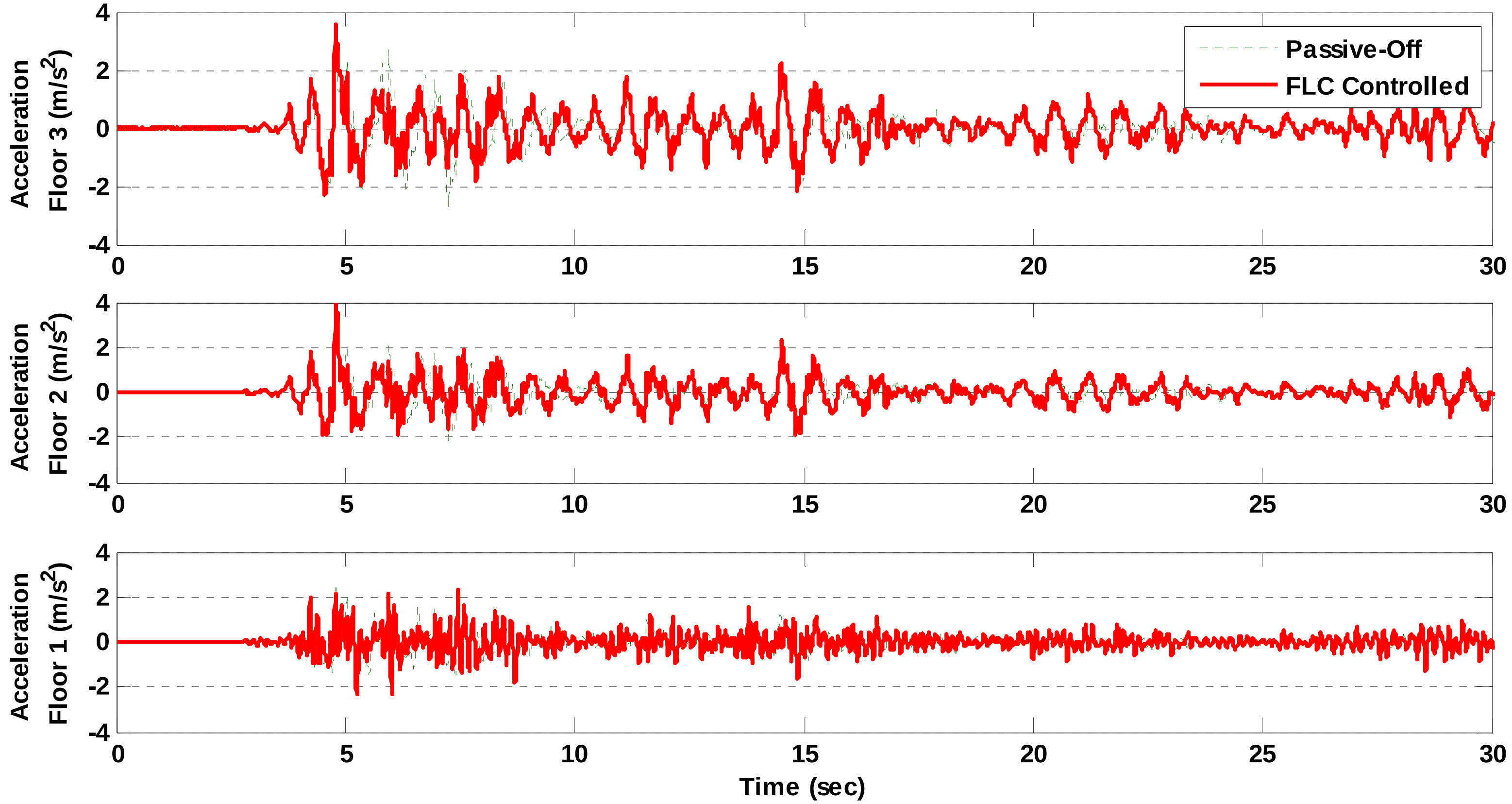


Fig. 86. Experimental Results of MIMO: Acceleration from 200 gal El Centro Earthquake

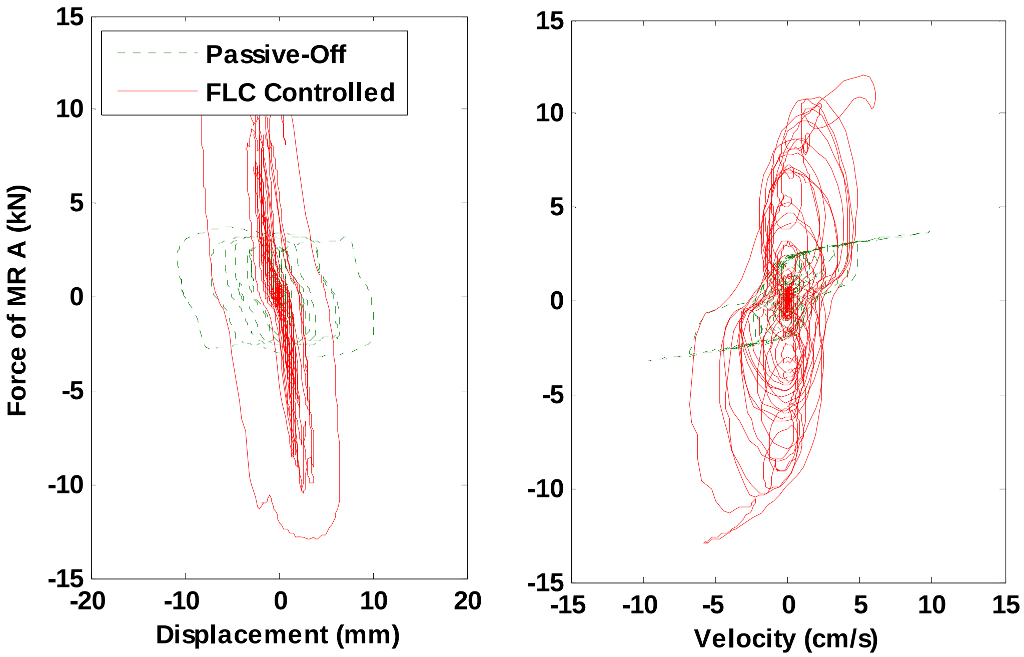


Fig. 87. Experimental Results of MISO: Hysteresis from 100 gal Kobe Earthquake

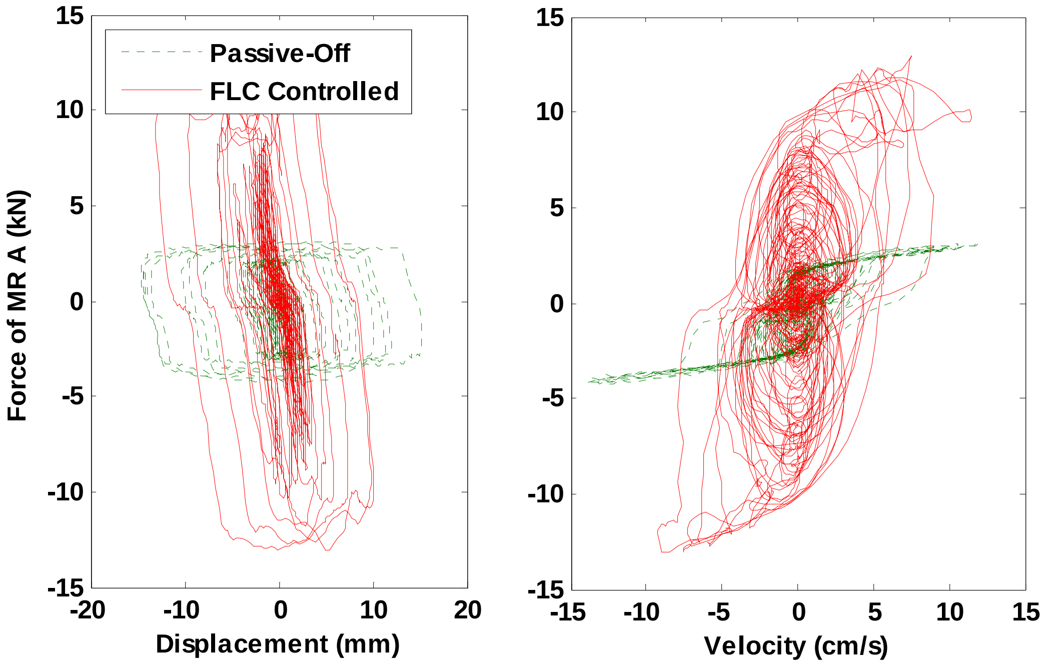


Fig. 88. Experimental Results of MIMO: Hysteresis from 200 gal El Centro Earthquake

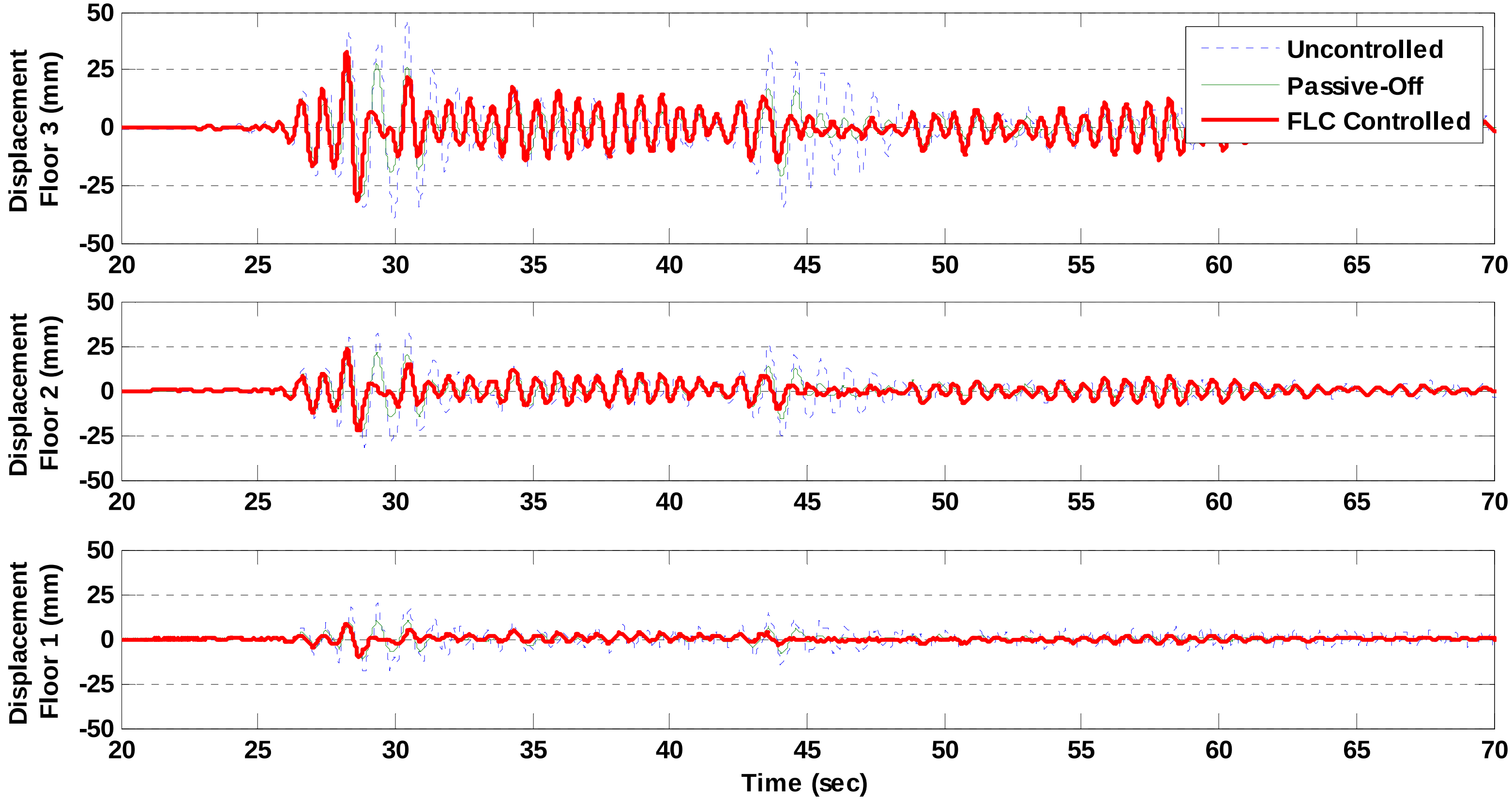


Fig. 89. Experimental Results of MISO: Displacement from 100 gal TCU076 Earthquake

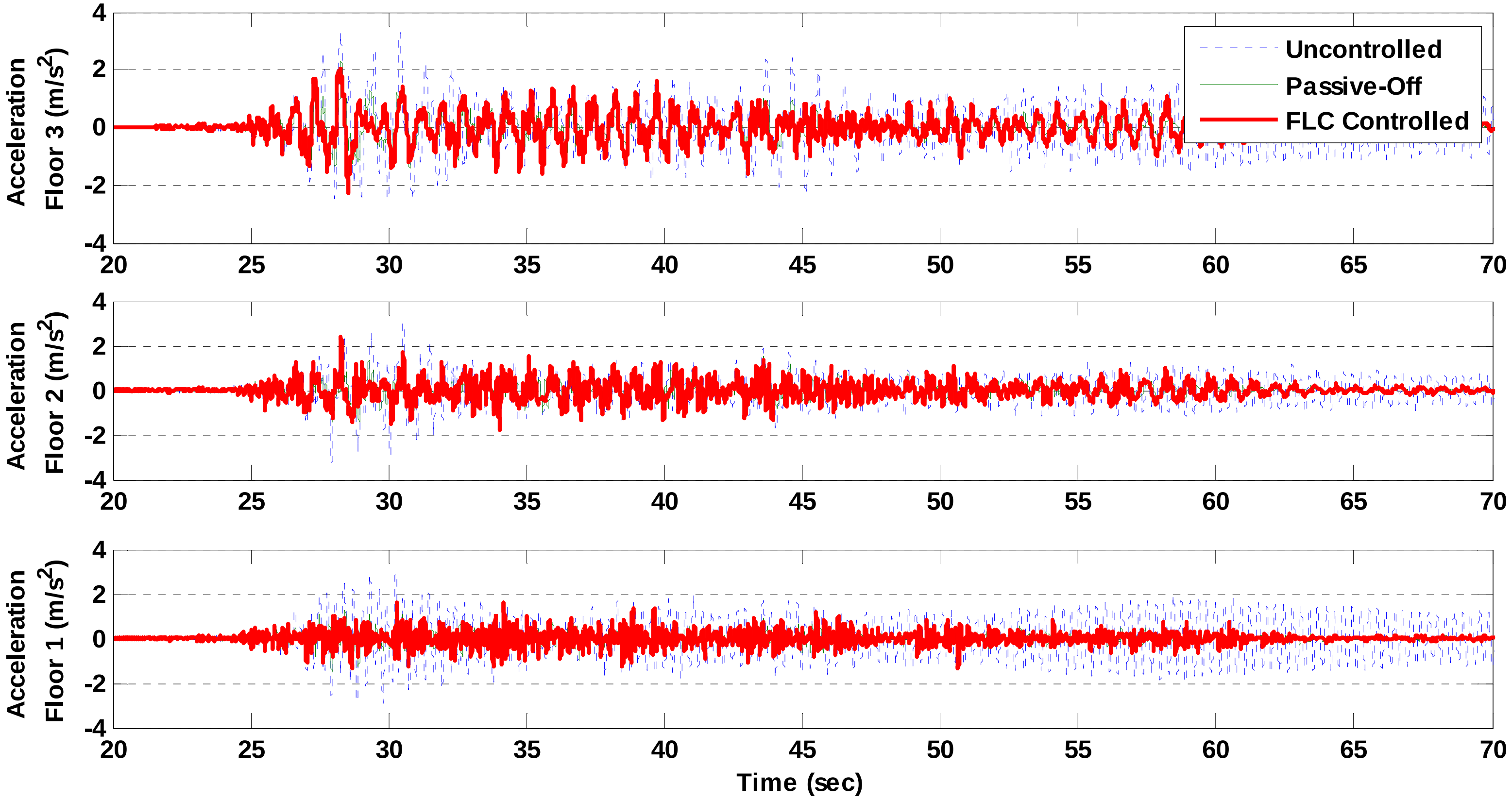


Fig. 90. Experimental Results of MISO: Acceleration from 100 gal TCU076 Earthquake

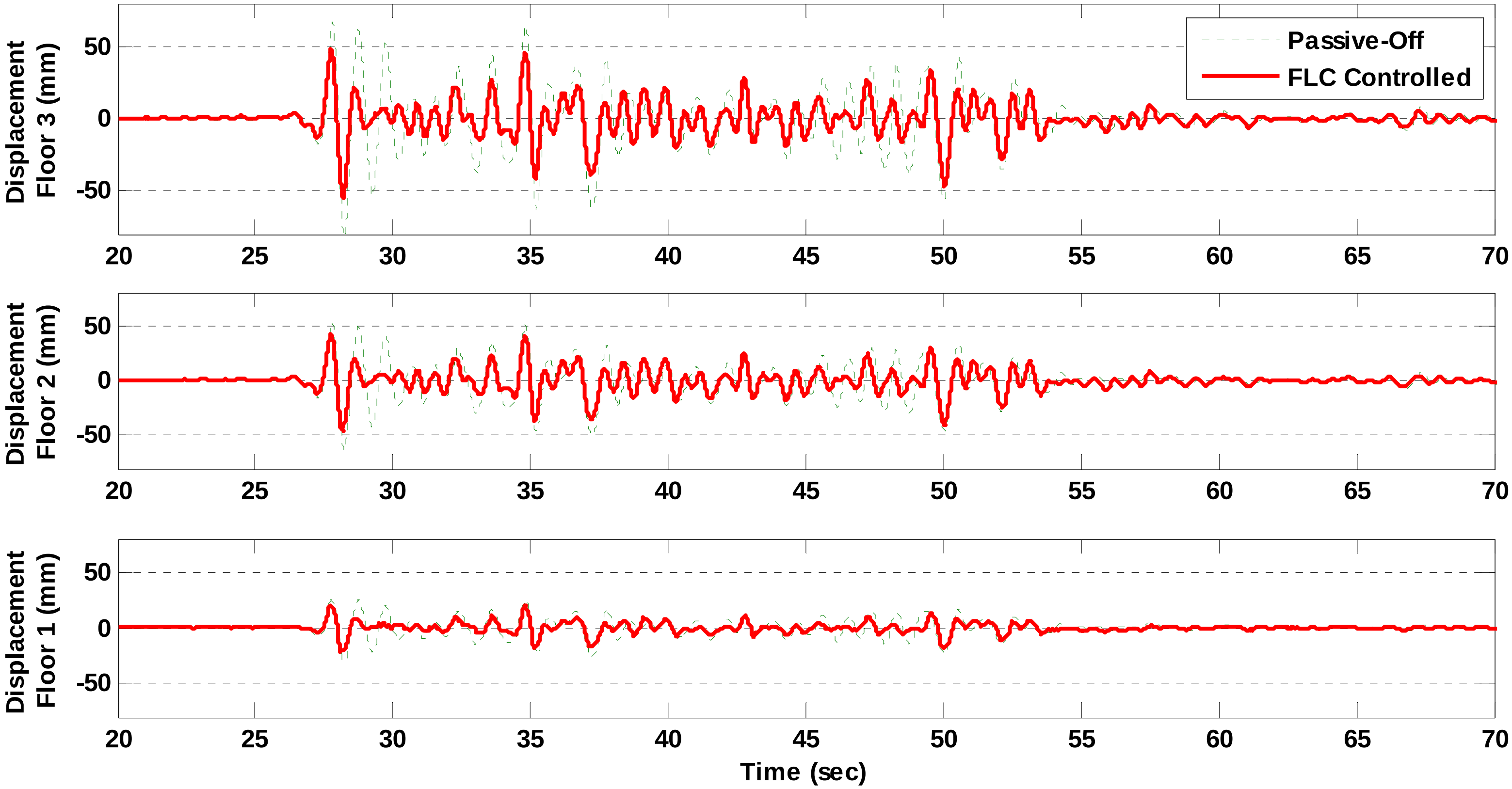


Fig. 91. Experimental Results of MIMO: Displacement from 200 gal TCU082 Earthquake

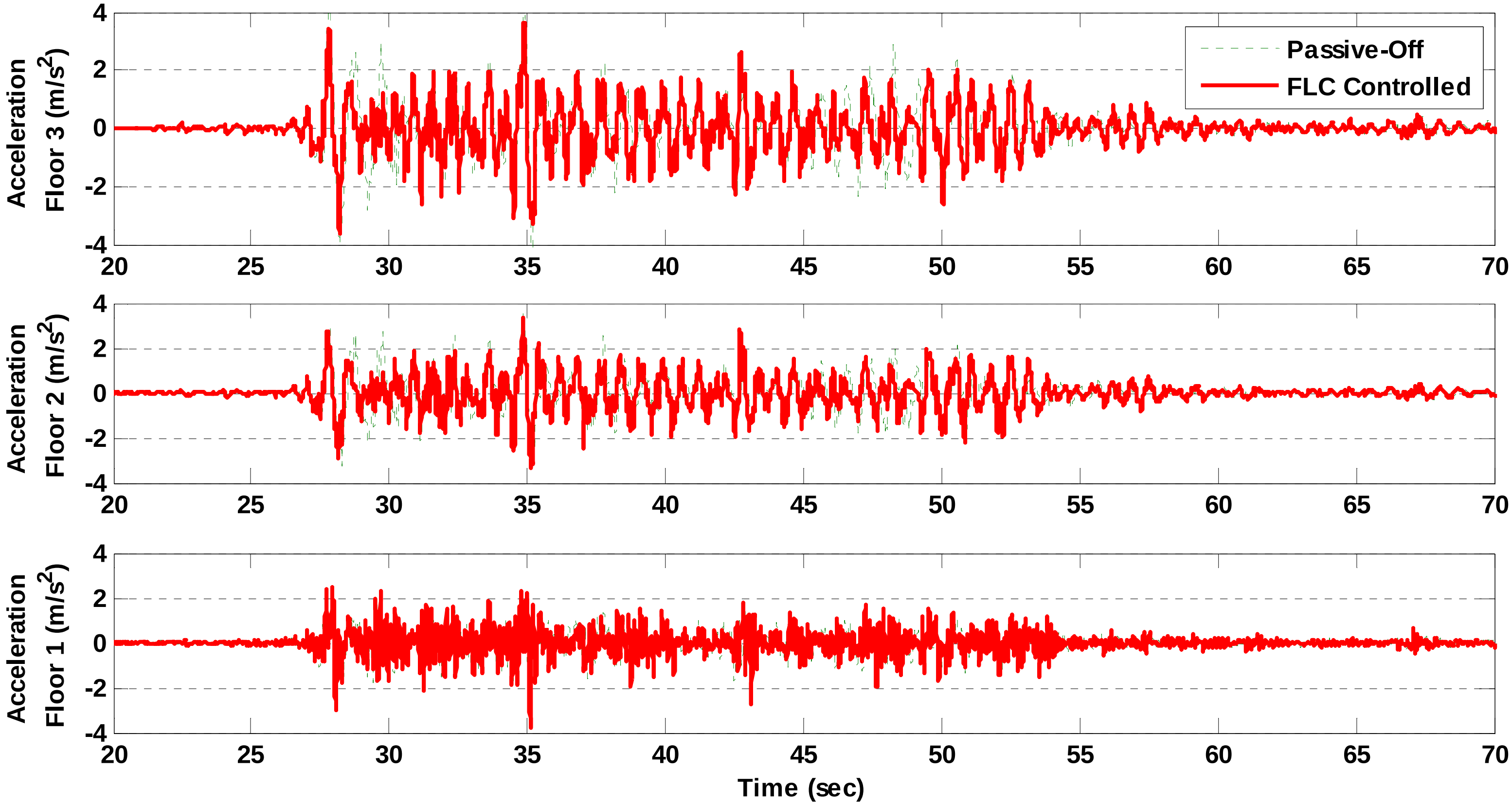


Fig. 92. Experimental Results of MIMO: Acceleration from 200 gal TCU082 Earthquake

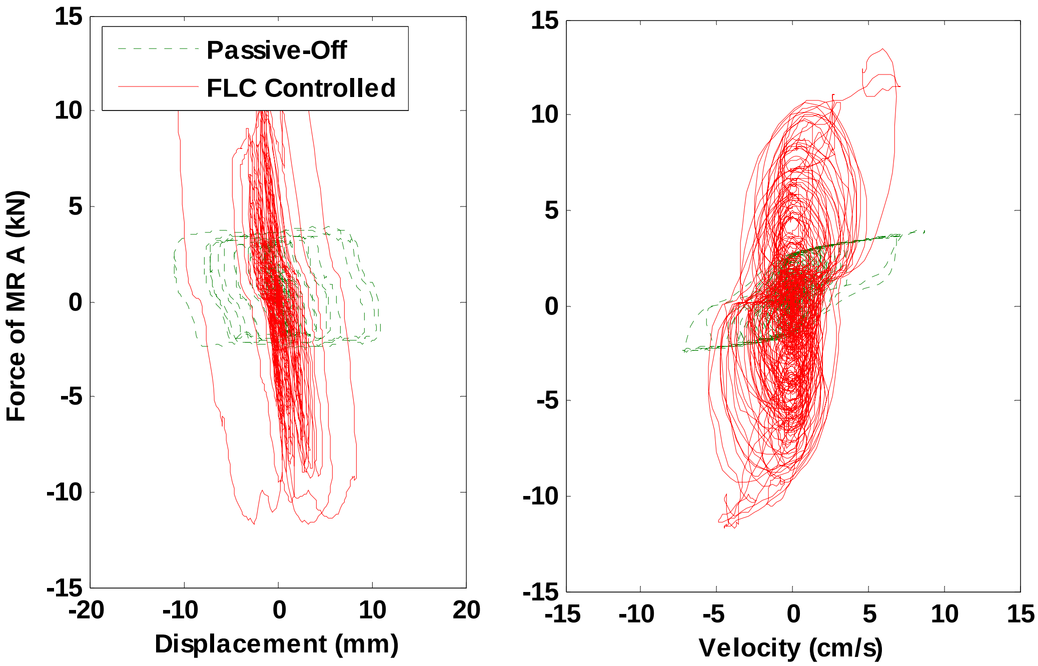


Fig. 93. Experimental Results of MISO: Hysteresis from 100 gal TCU076 Earthquake

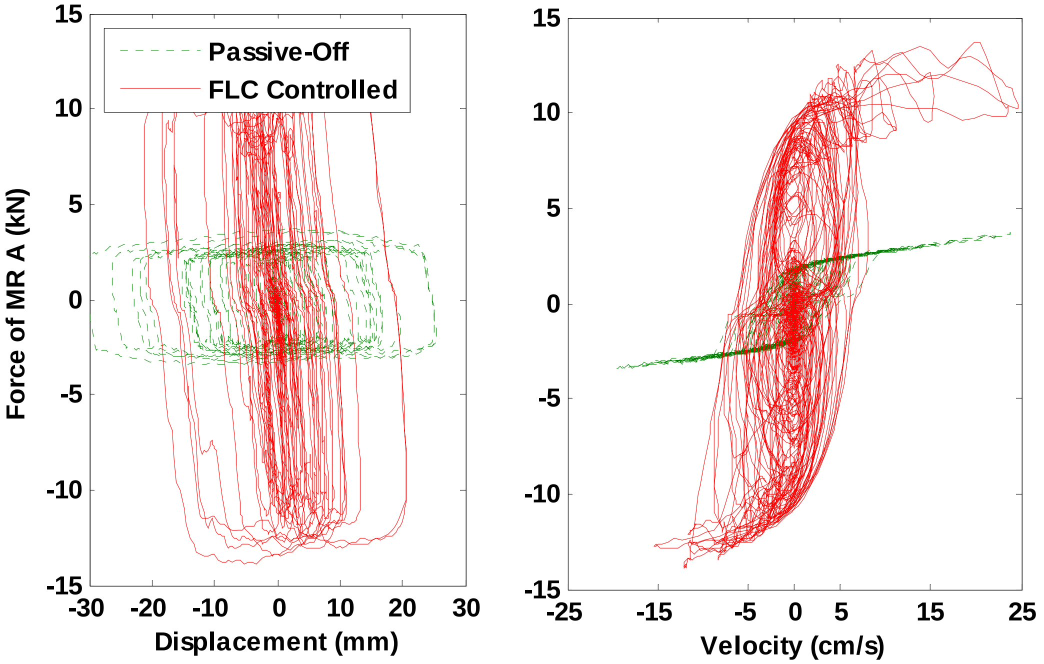


Fig. 94. Experimental Results of MIMO: Hysteresis from 200 gal TCU082 Earthquake

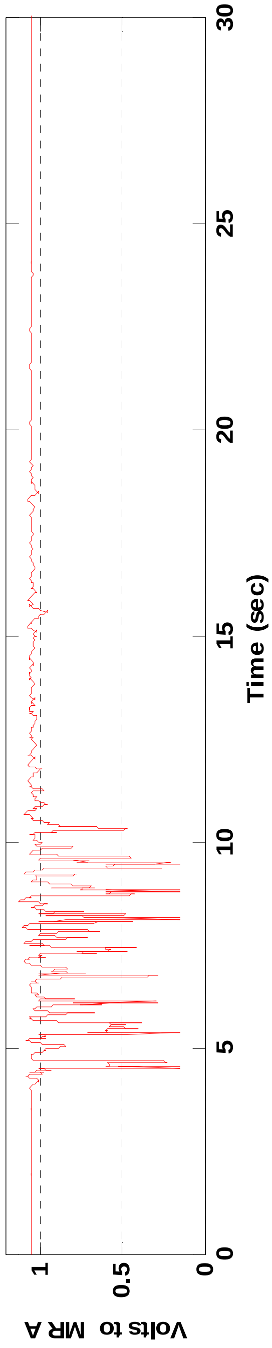


Fig. 95. Experimental Results of MISO: Voltage from 100 gal Kobe Earthquake

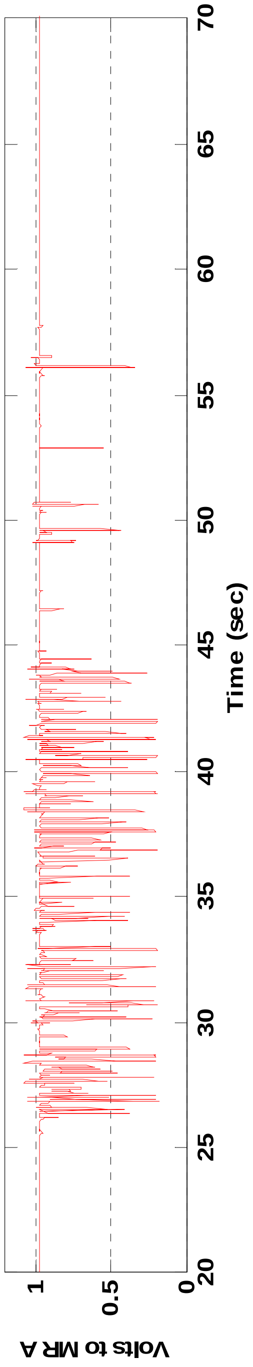


Fig. 96. Experimental Results of MISO: Voltage from 100 gal TCU076 Earthquake

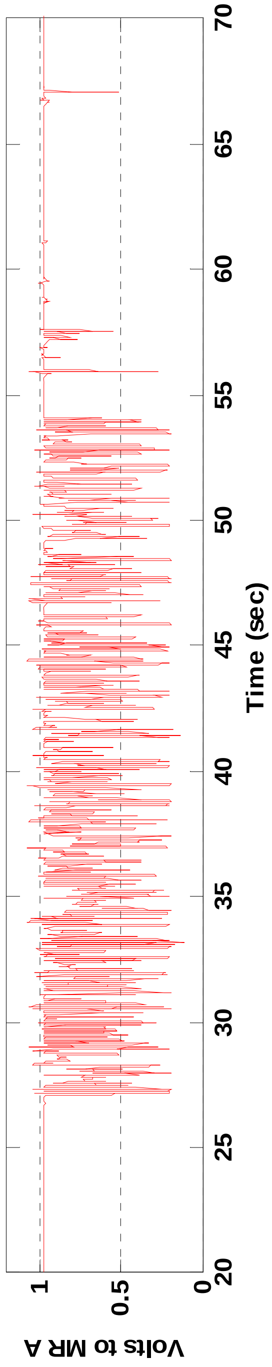


Fig. 97. Experimental Results of MIMO: Voltage from 200 gal TCU082 Earthquake

As can be observed in Table 9, FLC performance with respect to the 100 gal excitation level is favorable for most performance objectives when compared to both passive cases. FLC performance digresses as excitation levels increase above 100 gal.

This can be attributed to training of the FLC on a 100 gal artificial earthquake. Yet, FLC management is not negated since results are always superior to the passive-off case and are, at times, superior to the passive-on case.

When comparing numerical simulations and experimental results reasonable agreement can be observed. This is due to extensive modeling efforts using both neuro-fuzzy theory for generation of MR damper models and multiobjective optimization of a state-space model of the benchmark building. Despite these efforts some non-negligible errors do exist in numerical simulations. Such errors are reasonable and do not severally hinder the performance of FLCs in experimental trials. Metrics in this table display the robust nature of the FLC performance.

Table 9. Experimental Evaluation FLC Controllers

		MISO Configuration				MIMO Configuration			
		J_1	J_2	J_3	J_4	J_1	J_2	J_3	J_4
El Centro (100 gal)	P-ON	0.99	0.98	0.89	1.13	1.26	1.62	1.15	1.30
	FLC	1.05	**0.96**	**0.85**	1.11	1.19	1.33	1.18	1.31
Kobe (100 gal)	P-ON	0.93	1.09	1.01	1.37	0.70	1.14	1.17	1.55
	FLC	**0.92**	**0.92**	**0.94**	1.28	**0.59**	1.11	1.02	1.36
Chi-Chi, TCU076 (100 gal)	P-ON	1.01	1.06	0.84	1.20	0.72	0.97	1.04	1.36
	FLC	1.04	1.06	*0.97*	1.39	**0.63**	**0.95**	**0.98**	1.30
Chi-Chi, TCU082 (100 gal)	P-ON	0.43	0.64	0.40	0.69	0.82	1.64	1.08	1.53
	FLC	*0.43*	**0.59**	**0.38**	**0.63**	**0.72**	1.54	**0.96**	1.36
El Centro (200 gal)	P-ON	-	-	-	-	0.68	1.46	0.80	1.12
	FLC	-	-	-	-	*0.78*	1.41	**0.77**	1.06
Kobe (200 gal)	P-ON	-	-	-	-	0.58	0.70	0.53	0.86
	FLC	-	-	-	-	*0.72*	*0.87*	*0.56*	**0.84**
Chi-Chi, TCU076 (200 gal)	P-ON	-	-	-	-	0.58	0.76	0.51	0.88
	FLC	-	-	-	-	*0.64*	*0.81*	**0.48**	**0.84**
Chi-Chi, TCU082 (200 gal)	P-ON	-	-	-	-	0.53	0.79	0.55	0.95
	FLC	-	-	-	-	*0.59*	*0.87*	**0.51**	**0.83**

6.2.3. General Summary of FLC Performance

Substantiation of the FLC controllers has been provided using numerical simulations and experimental trials. Although experimental tests are critical for showing the validity of FLCs in future real world implementations, substantiation of the FLC controller algorithm is found equally in numerical simulation and experimental trials. These results are further summarized in Table 10 where the percentages of cases for which GA-optimized FLCs are superior to both passive cases are listed considering both numerical simulations and experimental trials. A 'case' corresponds to each objective of each excitation. That is, there are 16 cases resulting from the MISO configuration and 32 cases resulting from the MIMO configuration.

In Table 10 cases corresponding to the MISO configuration show greater consistency than the MIMO configuration when comparing simulation and experimental results for the singular passive cases. Although this can be attributed to a number of factors it is believed to primarily be derived from errors in modeling of MR dampers. Since two MR dampers are incorporated in the MIMO case the potential for error increases in comparison with the MISO configuration. It is believed that with a more ideal set of training data a much more accurate model of the MR damper could be attained as shown by Kim *et al.* (2006). When considering percentages in the 'Overall' column in Table 10**,** both passive-off and passive-off cases show favorable FLC performance. It can also be observed that FLC performance is superior to both passive voltages for 43% of all cases. Thus, the effectiveness of the GA optimization of FLC controllers is demonstrated.

Table 10. Summary of Controller Performance

	MISO		MIMO		Overall
	Simulation	Experimental	Simulation	Experimental	
Cases FLC is superior to passive-off	*63%*	*63%*	*84%*	*63%*	*68%*
Cases FLC is superior to passive-on	69%	63%	47%	72%	63%
Cases FLC is superior to passive-off and passive-on	**38%**	**50%**	**44%**	**41%**	**43%**

6.3. Summary

Overall results of numerical simulation and experimental testing show that GA optimization of MISO and MIMO control systems manifest improvement over both passive cases for a wide range of seismic records and configurations. GA-optimized FLC results show a reduction in displacement from the passive-on case without increasing the acceleration response. Also, for all numerical and experimental cases less voltage is specified to the MR damper(s) and the peak force produced by the MR dampers is reduced in comparison with the passive-on case. Difficulties of FLC performance can be observed at higher PGA excitations such as the 200 gal excitation records corresponding to the numerical simulation of the MIMO case. Such difficulties in control can be improved upon by selection of a more robust artificial excitation or alteration in FLC fuzzy limits prior to optimization. Despite these concerns, results similar to passive-on operation of the damper(s) are recorded for 200 gal excitation cases. Thus, the GA-optimized FLC is suitable for 100 and 200 gal excitations, but optimal for 100 gal excitations. Moreover, the FLC is robust with respect to frequency content due to favorable results corresponding to both near- and far-field excitations. Although difficulties appear when the level of the excitation amplitude is increased, GA-optimized FLCs are shown to be robust over a range of seismic intensities. GA-optimized FLC results show a clear reduction in displacement from the passive-on and passive-off cases without aggravation of the acceleration response.

7. CONCLUSIONS

All the research topics in this thesis converge towards the single goal of creating an effective structural control system for the mitigation of seismic excitations in civil engineering structures. To accomplish this several areas of study are included and consist of the following:

- Dynamic analysis of MDOF structures using state-space formulation,
- Identification of MR damper properties using neuro-fuzzy optimization techniques,
- Formulation of a multi-objective genetic algorithm,
- Identification of MDOF system properties using a genetic algorithm,
- Creation and identification of GA-optimized fuzzy logic controllers, and
- Experimental validation of identified controllers.

Discussion and dissemination of results of the above mentioned topics are presented in this thesis in seven correlated sections.

Section 1 gives reasons for the necessity of structural control systems for the mitigation of hazardous loads such as earthquakes. Then a control device, controller algorithm, and optimization method are identified. Finally an overview of software utilized in this study is provided.

Section 2 provides discussion of related studies pertaining to this investigation. A background of well-known studies and current literature is provided to give the reader a foundation for the current study.

Section 3 discusses the benchmark structure and its state-space formulation. Details pertaining to both material and geometry related to the benchmark structure are provided. The experimental setup is discussed and details are given to better describe the tests conducted and transducers employed. In the later portion of Section 3 a summery of the artificial earthquake that is created is provided. The artificial excitation is created to give a single excitation that contains characteristics of numerous earthquake records and at the same time avoids any preference to a single excitation or characteristic.

Section 4 provides details pertaining to MR dampers *A* and *B* that are used in the present study. Experimental performance tests that are conducted in Taiwan at NCREE are outlined and commented on. Furthermore, modeling of the MR dampers with a neuro-fuzzy optimization method called ANFIS is elaborated upon. Ideal characteristics of neuro-fuzzy training data are provided and linked to deficiencies in the current experimental tests.

Section 5 furnishes an overview of the genetic algorithm used in this study and its multi-objective capabilities. Comparisons with other major GA algorithms are provided and a rationale for employment of NSGA-II CE is given. Furthermore, several common optimization examples are used to explain the employment of NSGA-II CE as opposed to its forerunner NSGA-II.

Section 6 supplies details pertaining to system identification of the benchmark structure by utilization of the NSGA-II CE algorithm. Data collected from experimental tests are used in optimization and validation of the optimal state-space model.

Section 7 renders results of GA optimization for both MISO and MIMO cases. Here Pareto fronts are presented to show progression and final results of GA optimization. In addition, numerical substantiation of fuzzy logic controller is provided through evaluation of the controller performance over a suite of earthquake excitation records. Finally, these numerical endeavors are validating by experimental testing of the controller implemented on the benchmark structure.

7.1. Overall Effectiveness of GA Optimization

Numerical and experimental results of GA-optimized fuzzy logic controllers demonstrate improvement upon both passive-on and passive-off cases over a suite of excitations, but do not offer substantial reductions from these passive cases. The ability of a controller to significantly improve upon the optimal passive case is a common difficulty of many researchers in semi-active structural control.

This deficiency in controller effectiveness is believed to be primarily derived from physical limitations. A structure with an array of semi-active devices contains physical limitations that no controller can overcome or circumvent. For example, there exists a global minimization of structural responses that corresponds to an optimal modulation in the resisting force of the semi-active devices. This minimization of structural responses may, for example, only be 5% lower than any passive operation of the semi-active devices. The effectiveness of a controller can only be judged within this context. This context is difficult to define for most systems including the currently studied benchmark problem. It is believed that FLC management of the MR dampers in both MISO and MIMO cases of this study are optimal, but are only modestly superior to optimal passive operation of the MR dampers.

With such information in mind the potential of FLC performance could be greatly improved if the building/damper system were altered such that FLC management of MR dampers has a great effect on the response of the structure to excitation. For example, in a new construction scenario where MR dampers are to be installed the number, location, and individual characteristics of each MR can be optimized. Here, more avenues can be included in FLC generation to produce optimality with respect to structural response mitigation and fiscal cost can be attained.

In this study, the above-mentioned damper and structure parameters are assumed to be fixed and then a controller is derived for the provided system. A more ideal application of a genetic algorithm would be to consider variations in damper characteristics and, potentially, structural design considerations. The current study has the goal of providing the basis of a controller formulation if such a turn-key approach to structural control is devised.

Furthermore, new heuristic optimization methods need to be explored and developed for optimization of FLCs. As demonstrated in Section 4.8, deficiencies in NSGA-II CE exist as exampled by the 'F6' optimization example. Continued improvement of current algorithms and consideration of new methods will assist in generating faster and robust optimization algorithms.

The improved integration of structural control methods and devices is one of continued innovation in the structural engineering community. The advancement of semi-active structural control techniques will help bring today's imagination into tomorrow's reality for structural engineers.

7.2. Future Work

With a comprehensive study of the MISO and MIMO systems considered in this thesis having been conducted, more complex systems can also be pursued. Such configurations include control of torsional motion and base-isolated structures. Each of these systems possesses unique characteristics and difficulties for optimal control, yet they are related to realistic applications of MR dampers. Such prominence is compelling researchers to develop and test control schemes for these applications.

Torsional response of structures is often one of significant interest to structural design engineers due to the inherently weak resistance of most structures to torsional motion. A small number of large-scale torsional tests have been conducted with semi-active control systems. Implementation of torsional control systems concerning the NCREE benchmark structure could consist of inclusion of eccentric masses to incite significant torsional response of the benchmark structure for the MR dampers to mitigate as shown in Fig. 98. Early experimental tests show significant promise for such GA-optimized fuzzy logic control systems. Furthermore, a base-isolated benchmark problem has been issued by Nagarajaiah *et al.* (2002) that considers torsional response for an asymmetric building. Such applications are also to be considered in future efforts by the author.

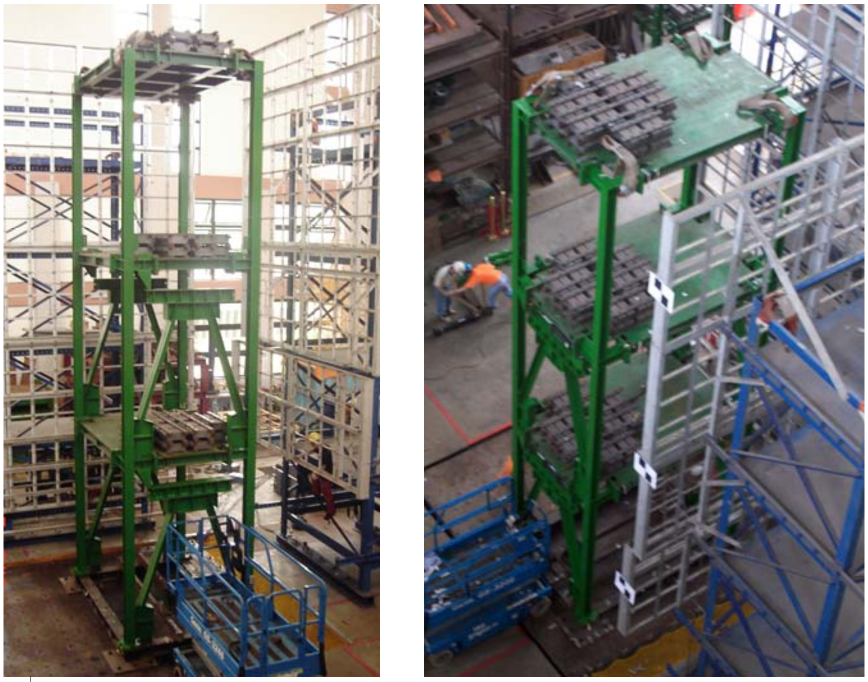


Fig. 98. Torsional Benchmark Structure with Eccentric Masses

A second application of future consideration is control of base-isolated structures. Here MR dampers would augment already existing passive devices such as high damping rubber bearings or friction pendulum systems as shown in Fig. 99. Bidirectional excitation should be also considered for controller development since earthquake motion is not usually aligned with a specific axis of a structure. Additionally, benchmark simulations created by Narasimhan *et al.* (2002) consider base isolation with MR damper augmentation as well.

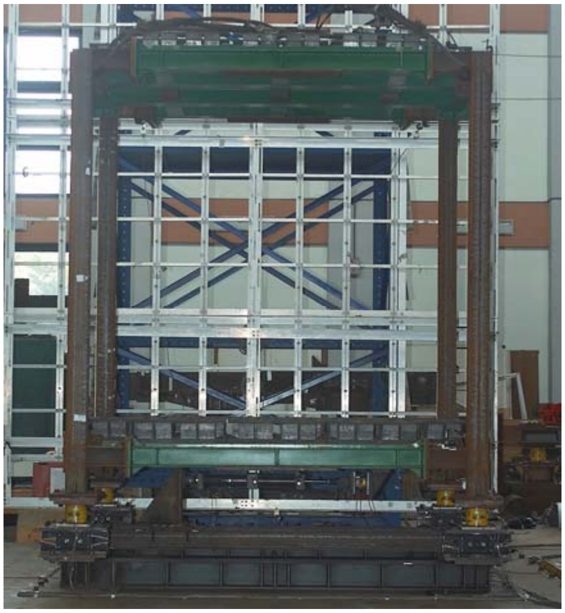


Fig. 99. Base Isolated Benchmark Structure

Creation and experimental validation of GA-optimized fuzzy logic controllers for these and other structural systems are crucial to the development of semi-active control systems. Continued efforts towards realistic and optimal control systems are to remain at the forefront of research and are the focus of future studies by the author.

REFERENCES

Ahlawat, A. S., and Ramaswamy, A. (2004a). “Multi-objective optimal fuzzy logic control system for response control of wind-excited tall building.” *Journal of Engineering Mechanics*, 130(4), 524-530.

Ahlawat, A. S., and Ramaswamy, A. (2004b). “Multi-objective optimal fuzzy logic controller driven active and hybrid control systems for seismically excited nonlinear buildings.” *Journal of Engineering Mechanics*, 130(4), 416-423.

Atray, V. A., and Roschke, P. N. (2004). “Neuro-fuzzy control of railcar vibrations using semi-active dampers.” *Computer-Aided Civil and Infrastructure Engineering*, 19(2), 81-92.

Canudas de Wit, C., Olsson, H., and Ǻstrőm, K. J. (1995). “A new model for control of systems with friction.” *IEEE Transaction on Automatic Control*. 40(3), 419-425.

Casciati, F., Faravelli, L, and Yao, T. (1994). Fuzzy logic in active structural control, *Proceedings of The 1994 1st International Joint Conference of NAFIPS/IFIS/NASA*, San Antonio, TX, USA, 268-272.

Cho, S. W, Kim, B. W., Jung, H. J., and Lee, I. W. (2005). “Implementation of modal control for seismically excited structures using magnetorheological dampers.” *Journal of Engineering Mechanics*, 131(2), 177-184.

Chopra, AK. (2006). *Dynamics of structures, 4th edition*. Prentice-Hall, Upper Saddle River, NJ, USA.

Deb, K. (1999). “Multi-objective genetic algorithms: Problem difficulties and construction of test problems.” *Evolutionary Computation*, 7(3), 205-230.

Deb, K. (2001). *Multi-Objective optimization using evolutionary algorithms*, John Wiley and Sons, Ltd., West Sussex, England.

Deb, K. and Goel, T. (2001). “Controlled-elitist non-dominated sorting genetic algorithms for better convergence.” *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, Zurich, Switzerland, 67-81.

Deb, K., Pratap, A., Agrawal, S., and Meyarivan, T. (2002). “A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II.” *IEEE Transaction on Evolutionary Computation,* 6(2), 182-197.

Deoskar, H. S., Tappeta, R. V., and Spencer Jr., B. F. (1996). “Effect of uncertainty on an active mass damper system.” *Proceedings of the 7th Specialty Conference*, Worcester Polytechnic Institute, MA, USA, 426-429.

Dyke, S. J., Spencer Jr., B. F., Sain, M. K., and Carlson, J. D. (1996). “Modeling and control of magnetorheological dampers for seismic response reduction.” *Smart Materials and Structures,* 5, 565-575.

Fonseca, C. M., and Fleming, P. J. (1993). “Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization, in genetic algorithms.” *Australian Electronics Engineering*, 27(2), 416-423.

Franklin, G., Powell, J, and Emami-Naeini, A. (2002). *Feedback Control of Dynamic Systems, 4th edition*. Prentice-Hall, Upper Saddle River, NJ, USA.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley Publishing Co., New York, NY, USA.

Guo, H. Y., Zhang, L., Zhang, L. L., and Zhou, J. X. (2004). “Optimal placement of sensors for structural health monitory using improved genetic algorithms.” *Smart Materials and Structures*, 13, 528-534.

Holland, J. (1975). *Adaptation in natural and artificial systems*, The University of Michigan Press, Ann Arbor, Michigan, 48016, USA.

Jang, J.S.R. (1993). “ANFIS: Adaptive-network-based fuzzy inference system.” *IEEE Transactions on Systems, Man and Cybernetics*, 23(3), 665-685

Janikow, C. Z., and Michalewicz, Z. (1991). “Experimental comparison of binary and floating point representations in genetic algorithms.” *Proceedings of the 4th International Conference on Genetic Algorithms*, San Diego, CA, USA, 31-45.

Jiménez, R., and Álvarex-Icaza, L. (2005). “LuGre friction model for a magnetorheological damper.” *Structural Control and Health Monitoring*, 12, 91-116.

Jin, G., Sain, M., and Spencer Jr., B. F. (2005). “Frequency domain system identification for controlled civil engineering structures.” *IEEE Transactions on Control Systems Technology*, 13(6), 1055-1062.

Karr, C. L. (1991). “Design of an adaptive fuzzy logic controller using a genetic algorithm.” *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Deigo, CA, USA, 450-457.

Kim, H., and Adeli, H. (2005). “Wind-induced motion control of 76-story benchmark building using the hybrid damper-TLCD system.” *Journal of Structural Engineering*, 131(12), 1794-1802.

Kim, H. S., and Roschke, P. N. (2006a). “Design of fuzzy logic controller for smart base isolation system using genetic algorithms.” *Engineering Structures*, 28(1), 84-96.

Kim, H. S., and Roschke, P. N. (2006b). “Fuzzy control of base-isolation system using multi-objective genetic algorithm.” *Computer-Aided Civil and Infrastructure Engineering*, 21, 436-449.

Kim, H. S., Roschke P. N., Lin, P. Y., and Loh, C. H. (2006). “Neuro-fuzzy model of hybrid semi-active base isolation system with FPS bearings and a MR damper.” *Engineering Structures*, 28(7), 947-958.

Knowles, J., and Corne, D. (1999). “The Pareto archived evolution strategy: A new baseline algorithm for multi-objective optimization.” *Proceeding of the 1999 Congress on Evolutionary Computation*, Washington, DC, USA, 98-105.

Lin, P. Y., Chung, L. L., and Loh, C. H. (2005). “Semiactove Control of Building Structures with Semiactive Tuned Mass Damper.” *Computer-Aided Civil and Infrastructure Engineering*, 20, 35-51.

Likhitruangsilp, V., and Roschke, P. N. (2003). “Fuzzy modeling of behavior of a 300kN magnetorheological damper.” *Proceeding of the Structures Congress and Exposition*, Seattle, Washington, USA 1003-1014.

MATLAB/Simulink, The Mathworks, Natick, MA, 2006.

Nagarajaiah, S., and Narasimhan, S. (2003). “Controllers for benchmark base isolated building with linear and friction isolation system.” *Proceedings of the 16th Engineering Mechanics Conference*, ASCE, Univ. of Wash., Seattle, July, CDROM.

Nagarajaiah, S., and Narasimhan, S. (2005). "Smart base-isolated benchmark building. Part II: Phase I sample controllers for linear isolation systems." *Structural Control and Health Monitoring*, 13, 589-604.

Narasimhan, S., Nagarajaiah, S., Johnson, E. A., and Gavin H. P. (2002). "Smart base isolated benchmark building Part I: Problem definition." *Journal of Structural Control and Health Monitoring*, 13, 573-588.

Oh, J., Roschke, P. N., Lin, P. Y., Carlson, J. D., and Sunakoda, K. (2004) "Experimental behavior and neuro-fuzzy modeling of 30-ton magnetorheological damper." *KSCE Journal of Civil Engineering*, 8(2), 213-219.

Overschee, P., and DeMoor, B. (1996). *Subspace identification of linear systems: Theory, implementation, applications*. Kluwer Academic Publishers, Norwell, MA, USA.

Perry, M. J., Koh, C. G., and Choo, Y. S. (2006). "Modified genetic algorithm strategy for structural identification." *Computers and Structures*, 84, 529-540.

Rao, M. M., and Datta, T. K. (2006). "Modal seismic control of building frames by artificial neural network." *Journal of Computing in Civil Engineering*, 20(1), 69-73.

Reigles, D. G, and Symans, M. D. (2005). "Supervisory fuzzy control of a base-isolated benchmark building utilizing a neruo-fuzzy model of controllable fluid viscous dampers." *Structural Control and Health Monitoring*, 13, 724-747.

Renzi, E., and Serino, G. (2004). "Testing and modeling a semi-actively controlled steel frame structure equipped with MR dampers." *Structural Control and Health Monitoring,* 11(3), 189-221.

Sahasrabudhe, S., and Nagarajaiah, S. (2005). "Experimental study of sliding base-isolated building with magnetorheological dampers in near-field earthquakes." *Journal of Structural Engineering*, 131(7), 1025-1034.

Sarkar, D., and Jayant, M. M., (2004). "Pareto-optimal solutions for multi-objective optimization of fed-batch bioreactors using nondominated sorting genetic algorithm." *Chemical Engineering Science*, 60, 481-492.

Schaffer, J. D., (1985). "Multiple objective optimization with vector evaluated genetic algorithms." *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Pittsburgh, PA, USA, 93-100.

Schaffer, J. D., Caruana, R. A., Eshelman, L. J., and Das, R. (1989). “A study of control parameters affecting online performance of genetic algorithms for function optimization.” *Proceedings of 3rd International Conference on Genetic Algorithms*, Fairfax, Virginia, USA 51-60.

Schurter, K. C., and Roschke P. N., (2000). “Fuzzy modeling of a magnetorheological damper using ANFIS,” *Proceeding of the 9th IEEE International Conference on Fuzzy Systems*, San Antonio, TX, May, 2000, 122-127.

Schurter, K. C., and Roschke P. N., (2001). “Neuro-fuzzy control of structures using acceleration feedback.” *Smart Materials and Structures*, 10, 770-779.

Soda, S., Kusumoto, H., Chatani, R., Iwata, N., Fujitani, H., Shiozaki, Y., and Hiwatashi, T. (2003). “Semi-active seismic response control of base-isolated building with MR damper.” *Smart Materials and Structures: Damping and Isolation,* 5052, 460-467.

Spencer Jr., B. F., and Nagarajaiah, S., (2003). “State of the art of structural control.” *Journal of Structural Engineering*, 129(7), 845-856.

Spencer Jr., B. F., Dyke, S. J., Sain, M. K., and Carlson, J. D. (1997). “Phenomenological model for magnetorheological dampers.” *Journal of Engineering Mechanics*. 123(3), 230-238.

Wen, Y. K. (1976). “Method of random vibration of hysteretic systems,” *Journal of Engineering Mechanics*, 102(2), 249-263.

Winslow W.M. (1948). “Method and means for translating electrical impulses into mechanical force.” US Patent No. 2,417,850.

Yan, G., and Zhou, L. L., (2004). “Fuzzy logic and genetic algorithms for intelligent control of structures using MR dampers.” *Smart Structures and Materials*, 5391, 555-565.

Yao, J. T. P. (1972). “Concept of structural control.” *Journal of Structural Engineering*. 98(ST7), 1567-1574.

Yoshida, O., and Dyke, S. J. (2005). “Response control of full-scale irregular building using magnetorheological dampers.” *Journal of Structural Engineering*, 131(5), 734-742.

Zadeh, L.A. (1965). "Fuzzy sets." *Information and Control*, 8, 338-353.

Zitzler, E., and Thiele., L, (1998). “Multi-objective evolutionary algorithms a comparative case study and the strength Pareto approach,” *IEEE Transactions on Evolutionary Computation*, 3(4), 257-271.

APPENDIX A

MATLAB CODE FOR NEURO-FUZZY TRAINING

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ANFIS Model of 20kN MR Damper A
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This M-file will generate TWO ANFIS models, thus the for loop being i=1:2
clear;clc;
for i = 1:2
    tic
    %Load in Test Data to be used in training
    load TrnDataANFIS
    time = dataset(1,:);
    disp = dataset(2,:);
    vel = dataset(3,:);
    volts = dataset(5,:);
    force = dataset(4,:);

    %Create Training and Validation Data
    input = [disp' vel' volts'];
    output = [force'];
    data = [input output];

    % Define training and checking data
    numPts = length(data);
    interval = 1;
    trnData = data(1:interval:numPts,:);
    chkData = data(interval:interval:numPts,:);
    clear data
    numPtsused = length(trnData)

    %Number of membership functions
    if i == 1
        numMFs = [2 2 2];
    elseif i == 2
        numMFs = [2 4 3];
    end

    %Type of membership function
    mfType = 'gbellmf';

    %For now provide a fis to be trained called in_fismat
    %genfis1 is a basic fis provided by matlab
    in_fismat = genfis1(trnData,numMFs,mfType);

    %Define training options (trnOpt)
    %Training epochs
    epoch_n = 200;
    %Training error goal
    error_goal = 0;
    %Training step size
    ss = 0.2;
    %Training step size decrease rate
    ss_dec_rate = 0.8;
    %Training step size increase rate
    ss_inc_rate = 1.2;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %ANFIS training
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    [trn_out_fismat trn_error step_size chk_out_fismat chk_error] = ...
        anfis(trnData, in_fismat, [epoch_n nan ss ss_dec_rate ss_inc_rate], ...
        [1,1,1,1], chkData);
    writefis (chk_out_fismat,'fis_anfis');

    %Plot Error Curves
    errorplot = figure('Color',[1 1 1])
    subplot(2,1,1)
```

```
[a, b] = min(chk_error);
plot (1:epoch_n, trn_error, '-', ...
    1:epoch_n, chk_error, ':', ...
    b, a, 'go');
xlabel ('Epochs');
ylabel ('RMSE');
title ('Error Curves');
legend ('Training error', 'Checking error', 0);
grid on
subplot(2,1,2)
plot (1:epoch_n,step_size, '-', 1:epoch_n,step_size, 'x')
axis ([-inf inf -inf inf])
xlabel ('Epochs'); ylabel ('Step Size'); title ('Step Sizes')
grid on
if i == 1
    hgsave(errorplot,'errorplot_222_gbel');
elseif i == 2
    hgsave(errorplot,'errorplot_243_gbel');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ANFIS Results
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
force_anfis = evalfis([disp' vel' volts'],trn_out_fismat);
force_anfis = force_anfis';

%Plot given data and ANFIS output to see how good ANFIS optimized FIS is
dataplot = figure('Color',[1 1 1])
subplot(3,1,1)
plot(time,force,time,force_anfis);
xlabel('Time (s)')
ylabel('Force (N)')
title('Validation')
legend('Training Data','ANFIS Output')
subplot(3,1,2)
plot(disp,force,disp,force_anfis);
xlabel('Displacement (mm)')
ylabel('Force (N)')
legend('Training Data','ANFIS Output')
subplot(3,1,3)
plot(vel,force,vel,force_anfis);
xlabel('Velocity (mm/s)')
ylabel('Force (N)')
legend('Training Data','ANFIS Output')
if i == 1
    hgsave(dataplot,'dataplot_222_gbel');
elseif i == 2
    hgsave(dataplot,'dataplot_243_gbel');
end

%New Membership Functions
mfplot = figure('Color',[1 1 1])
subplot (321);
plotmf (in_fismat, 'input', 1);
subplot (322);
plotmf (chk_out_fismat, 'input', 1);
delete (findobj(gcf, 'type', 'text'));
subplot (321);
title ('Initial MFs for Input 1');
subplot (322);
title ('Final MFs for Input 1');
subplot (323);
plotmf (in_fismat, 'input', 2);
subplot (324);
plotmf (chk_out_fismat, 'input', 2);
delete (findobj(gcf, 'type', 'text'));
subplot (323);
title ('Initial MFs for Input 2');
subplot (324);
title ('Final MFs for Input 2');
```

```
    subplot (325);
    plotmf (in_fismat, 'input', 3);
    subplot (326);
    plotmf (chk_out_fismat, 'input', 3);
    delete (findobj(gcf, 'type', 'text'));
    subplot (325);
    title ('Initial MFs for Input 3');
    subplot (326);
    title ('Final MFs for Input 3');
    if i == 1
        hgsave(mfplot,'mfplot_222_gbel');
    elseif i == 2
        hgsave(mfplot,'mfplot_243_gbel');
    end

    %Plot FIS surface
    fisplot = figure
    gensurf(trn_out_fismat)
    if i == 1
        hgsave(fisplot,'fisplot_222_gbel');
    elseif i == 2
        hgsave(fisplot,'fisplot_243_gbel');
    end

    if i == 1
        writefis(trn_out_fismat,'222FIS_gbel');
    elseif i == 2
        writefis(trn_out_fismat,'243FIS_gbel');
    end
    toc/60
end
```

APPENDIX B

MATLAB CODE FOR FLC OPTIMIZATION

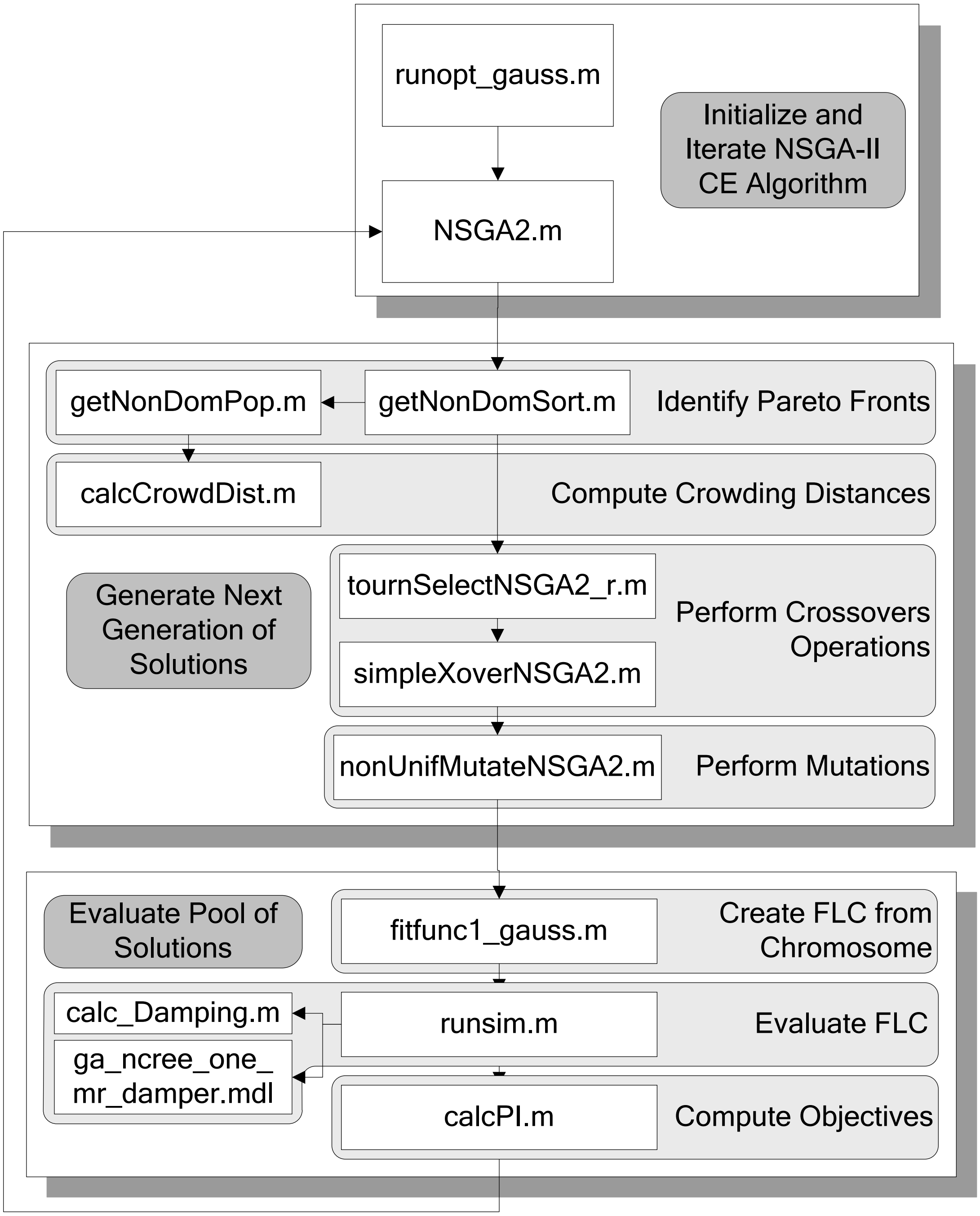


Fig. B1. Flowchart of NSGA-II CE M-files

In what follows M-files and Simulink diagrams used in excecution of NSGA-II CE optimization are provided.

B. 1.   runtop_gauss.m

```
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% runopt_gauss.m
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% M-file running Genetic Algorithm Optimization Toolbox
% Intended for control of NCREE Benchmark.  The controller uses a fuzzy inference system.
% There are 3DOFs in the NCREE structure.  Each story has a
% Chevron brace that attaches the damper between two floors.
% Input to the controller consists of accelerations from the 1st, 2nd, 3rd,
% There is no time delay (see Kyle Schurter's thesis).
% Output from the controller is a voltage signal to the single MR damper
% located bewteen the ground and the 1st floor.  Gaussian membership
% functions are used for the input and output variables.
% Here only one MR Damper is installed between the ground and 1st floor
%
% Created by : Hyun-Su Kim
%             09-06-2004
% Modified by: PR    18 December 2004
% Modified by: PR    11 Feb 2005
% Modified b : PR    31 May 2005
% Modified b : PR    4  June 2005
% Modified by: DS    8  August 2005
%             DS   18  May 2006
glbvar;

% Define the degrees of freedom of the structure
nDOF = 3 ;

% Initialize the number of the iteration to zero.
nIter = 0;
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Define the earthquake name and time history (see Simulink)
load ArtEQ
eq = [feq(1,:);feq(2,:)*9.81];

% Extract the time vector from the input file.
t = eq(1,:);
save eq eq
% Prepare time increment and total simulation time for Simulink
dt = t(2) - t(1) ;
Duration = max(t(:)) ;
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Enter the number of the run from the keyboard.  There are only two
% possibilities:  1 = first time run, 2 = any restart run.
numRun = input('Input number of run [Default = 1]: ');
if isempty(numRun)
  numRun = 1;
end
if numRun ~= 1
      disp('If you loaded previous results, strike any key to continue. Otherwise, press [Ctrl+C], load previous results and try
again!');
  pause
  initPop = endPop;
  bResume = 1;
else
  initPop = [];
  bResume = 0;
end
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Define Fitness Function
fitnessFunc = 'fitfunc1_gauss';
% Declare the number of variables for each rule
% For 3 inputs and 1 output there are 8 variables for each rule,
% 2 for each of the 3 inputs and 2 for the 1 output.
nVariableNum = 8 ;
% Declare the total number of rules for each input and output variable.
% Here Kyle Schurter's thesis is followed: use fewer rules, but more
% variables.
nRules = 10 ;
```

```
% Declare the number of input variables: 'x' rules times variables
numVar = nVariableNum*nRules ;
% Set the total population size.  This is somewhat arbitrary.
popSize = 20 ;
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
%Number of generations for GA optimization
genNum = 200; % for one running
%Number of muation generations for GA optimization
mutGenNum = 300; %finalGenNum-genNum*(numRun-1);
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% load MR FIS - PY 20kN (march 2006 performance testing)
MRfis = readfis('B1_Seat_222b.fis');
MR_disp_sat_min = MRfis.input(1).range(1);
MR_disp_sat_max = MRfis.input(1).range(2);
MR_vel_sat_min = MRfis.input(2).range(1);
MR_vel_sat_max = MRfis.input(2).range(2);
MR_volt_sat_min = MRfis.input(3).range(1);
MR_volt_sat_max = MRfis.input(3).range(2);
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% The range of accelerations that is to be used for training the controller
% Units are m/s^2.
min_acc = -1 ;
max_acc =  1 ;
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% The range of each input variable is given below.
acc_range     = [0 0.4] ;
c_acc_range   = [min_acc max_acc] ;
volt_range    = [0 0.4] ;
c_volt_range  = [MR_volt_sat_min MR_volt_sat_max] ;
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Intial chromosomes
initRange = [ ...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range;...
   acc_range; c_acc_range; acc_range; c_acc_range; acc_range; c_acc_range; volt_range; c_volt_range] ;
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Check to see that the number of rules times the number of variables per
% rule agrees with the dimensions of initRange
if numVar ~= size(initRange,1)
    disp ('Check the size of the initRange and the number of rules, etc.');
    stop
end
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Set the mutation rate for child individuals that are created by the mutation operator.
mutRate = 0.3;
% Compute the number of population members in the next generation that are obtained from mutation.
mutNum =  mutRate*popSize;
% Set the rate of child individuals that are created by the crossover
% operator.  Again, some guidance was obtained from the Matlab GA user
% manual.
xoverRate = 1;
% Calculate the number of population members for the next generation
xoverNum =  floor(xoverRate*popSize/2.);
% Set a tolerance rate.
tolerance = 1e-6;
% Set a switch for displaying values to the screen during calculations.
dispOnOff = 1;
% Define the number of multiobjective functions
numObj = 4;
%=== Start GA function ================================================
[x,endPop,bPop,traceInfo] = NSGA2(initRange, fitnessFunc,[],initPop,...
    [popSize tolerance dispOnOff numObj bResume],'maxGenTerm',genNum,...
  'tournSelectNSGA2',[2],['simpleXoverNSGA2'],[xoverNum 0],'nonUnifMutateNSGA2',[mutNum mutGenNum 3]);
```

B. 2. NSGA2.m

```
function [x,endPop,bPop,traceInfo] = NSGA2 (bounds,evalFN,evalOps,startPop,opts,...
   termFN,termOps,selectFN,selectOps,xOverFNs,xOverOps,mutFNs,mutOps)
% GA run a genetic algorithm
% function [x,endPop,bPop,traceInfo] = NSGA2(bounds,evalFN,evalOps,startPop,opts,
%                                termFN,termOps,selectFN,selectOps,
%                                xOverFNs,xOverOps,mutFNs,mutOps)
%
% Output Arguments:
%   x          - the non-dominated individuals of the first front
%   endPop      - the final population
%   bPop        - a trace of the non-dominated individuals of the first
%   front
%   traceInfo    - a matrix of the objectives of the non-dominated individuals
%               of the first front for each generation
%
% Input Arguments:
%   bounds      - a matrix of lower and upper bounds on the variables
%                 (see initRange in runopt)
%   evalFN      - the name of the evaluation .m function
%   evalOps      - options to pass to the evaluation function ([NULL])
%   startPop     - a matrix of solutions that can be initialized
%               from initialize.m
%   opts        - [pop_size epsilon display obj_num resume_flag]
%               pop_size is the basic population size.
%               epsilon is tolerance to compare values.
%               display is 1 to output progress 0 for quiet.
%               obj_num is number of numti-objectives.
%               if resume_flag is 0, start with random new population,
%               resume_flag is 1, resume with startPop
%               ([100 1e-6 1 2 0])
%   termFN      - name of the .m termination function (['maxGenTerm'])
%   termOps      - options string to be passed to the termination function
%               ([100]).
%   selectFN     - name of the .m selection function (['tournSelectNSGA2'])
%   selectOpts   - options string to be passed to select after
%               select (tournament number) ([2])
%   xOverFNS     - a string containing blank seperated names of Xover.m
%               heuristicXover is not available !!!!
%               But, now heuristicXover is available only in case of Nagoya approach
%               because fitness is evaluated right after crossover operation
%               To use this function, fitness should be evaluated right
%               after crossover operation or parent pool and children
%               pool should be separated to keep fitness of parents.
%               Because this function use fitness of parent.
%               files (['simpleXoverNSGA2'])
%   xOverOps     - A matrix of options to pass to Xover.m files with the
%               first column being the number of that xOver to perform
%               similiarly for mutation ([2 0])
%   mutFNs      - a string containing blank separated names of mutation.m
%               files (['nonUnifMutateNSGA2'])
%   mutOps      - A matrix of options to pass to Xover.m files with the
%               first column being the number of that xOver to perform
%               similiarly for mutation ([4 100 3])

% Binary and Real-Valued Simulation Evolution for Matlab
% Copyright (C) 1996 C.R. Houck, J.A. Joines, M.G. Kay
%
% C.R. Houck, J.Joines, and M.Kay. A genetic algorithm for function
% optimization: A Matlab implementation. ACM Transactions on Mathmatical
% Software, Submitted 1996.
%
% This program is free software; you can redistribute it and/or modify
% it under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 1, or (at your option)
% any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details. A copy of the GNU
% General Public License can be obtained from the
% Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

%%$Log: ga.m,v $
%Revision 1.10  1996/02/02  15:03:00  jjoine
% Fixed the ordering of imput arguments in the comments to match
% the actual order in the ga function.
%
%Revision 1.9  1995/08/28  20:01:07  chouck
% Updated initialization parameters, updated mutation parameters to reflect
% b being the third option to the nonuniform mutations
%
%Revision 1.8  1995/08/10  12:59:49  jjoine
%Started Logfile to keep track of revisions
%
% Modified by Hyun-Su Kim,  2004/10/17
% Modified by Paul Roschke, 5 December 2004   25 January 2005
%  This file calls on a public-domain GA toolbox (gaot) and uses an NSGA-2
%  algorithm that was developed from papers by
%  Kalyanmoy Deb and Tushar
%  Goel, "Controlled Elitist Non-dominated Sorting Genetic Algorithms for
%  Better Convergence,"
%  Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan, "A Fast
%  Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective
%  Optimization: NSGA-II"

% Load global variables
glbvar ;

% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% The variable 'nargin' is the number of function arguments in the call
% to this routine.  It is a standard Matlab function.  See "help nargin"
n = nargin;
% Check for numerous cases of input parameters for the NSGA2 function.
if n < 2 | n == 6 | n == 10 | n == 12
   disp ('Insufficient arguments')
end
% Number of input arguments is 2 and there are no input evaluation options
if n < 3
   evalOps = [];
end
% Number of input arguments is 4, so assign some input evaluation option
% values.
if n < 5
   opts = [100 1e-6 1 2 0];
end
if isempty(opts)
   opts = [100 1e-6 1 2 0];
end
% Number of input arguments is less than six, so enter
% default value for termination.
if n < 6
   termOps = [100];
   % Define the Matlab function maxGenTerm.m file name
   termFN = 'maxGenTerm';
end
% Set the default mutation filename information
if n < 12
   mutFNs = ['nonUnifMutateNSGA2'];
   mutOps = [4 termOps(1) 3];
end
% Assign default crossover information
if n < 10
   xOverFNs = ['simpleXoverNSGA2'];
   xOverOps = [2 0];
end
% Default select options string, only i.e. roullete wheel
if n < 9
```

```
  selectOps = [];
end
% Set default for selection function
if n < 8
  selectFN = ['tournSelectNSGA2'];
  selectOps = [2];
end
% Set default for termination information
if n < 6
  termOps = [100];
  termFN = 'maxGenTerm';
end
% If no starting population is given
if n < 4
  startPop = [];
  % Set a resume flag
  opts(5) = 0;
end
% If no starting population is given, generate randomly
if isempty(startPop)
  % Set resume flag
  opts(5) = 0;
end

% Define a string variables that will be sent to be run in Matlab as
% a command line entry by the 'eval.m' function.
e1str = ['[c1 c1(numVar+1:numVar+numObj)]=' evalFN '(c1,[gen numObj evalOps]);'];

% Return a matrix of strings from the original string that is blank
% separated.  See list in incoming parameters to this function.
xOverFNs = parse(xOverFNs);
mutFNs   = parse(mutFNs);

%=== Set Initial values ==================================================
popSize      = opts(1);   %Number of individuals in the population (See runopt.m)
epsilon      = opts(2);   %Threshold for two fittness to differ
display      = opts(3);   %Display progress
numObj       = opts(4);   %Number of objectives
resumeFg     = opts(5);   %Resume Flag
numVar       = size(bounds,1);                %Number of variables
xZomeLength  = numVar+numObj+2; %Length of the xzome=numVars+multi_objs+rank+dist
endPop       = zeros(popSize,xZomeLength); %A secondary population matrix
c1           = zeros(1,xZomeLength);   %An individual
c2           = zeros(1,xZomeLength);   %An individual
numXOvers    = size(xOverFNs,1);       %Number of Crossover operators
numMuts      = size(mutFNs,1);         %Number of Mutation operators
bFoundIn     = 1;                      %Number of times best has changed
%See very bottom of this routine.
gen          = 1;                      %Current generation number
done         = 0;                      %Done with simulated evolution
% nargout is a special Matlab function that gives the number of function output
% arguments. Inside the body of a user-defined function, NARGOUT returns the
% number of output arguments that were used to call the function.
collectTrace = (nargout > 3);                %Should we collect info every gen

%=== Start NSGA2 ==========================================================
% Use the next set of statements to loop until the expression is not true.
% A 'true' value occurs when the expression is '1' and a 'false' value
% occurs when the expression is '2.'  So, if 'done' is set to '0' (see
% above) the 'not done' is made to be one and the expression is 'true.'
% Later, when 'done' is set to unity, then 'not zero' is zero, and the
% value of the expression is 'false' and the looping ceases.  (Go figure.)
while (~done)

  %=== Start Next Generation ===
  if gen == 1
    % Check the 'bResume' flag from the runopt.m definition to see if this
    % is a restart from a previous run.  If resumeFg = 0, this is the first
    % run and initialization of a population is required.
    if resumeFg == 0
```

```
      startPop = initializeNSGA2 (popSize, bounds, evalFN, [], [1e-6 numObj]);
   end
else
   % Redifine previous generation endPop as new startPop
   startPop = endPop;
end

%+++++++Define Parent Population (ie startPop)+++++++
%=== Selection operator ("r") ===
% Intead of a normal tourneyment style selection an "r" value
% implimentation has been made.

% Determine Number of individuals allowed in each rank by geometric
% distribution (Deb and Goel).  "numInRank" is a vector with the limit of the
% number of individuals per rank.
n = size(startPop,1)/2;
K = 4;
r = 0.5;
[numInRank] = getNumIndividualsPerRank(n,K,r)

% Location of Crowding Distance in startPop
e = size(startPop,2);
% Setup zero matrix to fill up with best of parent population
newPop = zeros(n,e+1);
% Start counter for newPopmatrix
newPopRow = 1;
% Define "check" for while loop
check = 1;
% Define rank for ranks in loop
rank = 1;

% Get non dominated fronts
[sortedPop] = getNonDomSort(startPop, numObj)
save sortedPop sortedPop

% Create Position Vector.  This is needed since individuals will be
% relocated to other matricies (ie extraPop)
for i = 1:size(sortedPop,1)
   pos(i) = i;
end
% Add Position Vector to sortedPop
sortedPop = [sortedPop,pos'];
   % Set up matrix for extraPop
extraPop = [];

% Loop for each rank of startPop
while check > 0
   % Print current rank in command window
   rank = rank
   % Remove old leftOverPop
   leftOverPop = [];
   % Get rid of previous Front information
   currentFrontPop = [];
   % If an extra rank is needed, fill it up with individuals from
   % extraPop and ranks larger than K here
   if rank > K
      disp('K Case')
      % Calculate number needed to full up population from extraPop
      numNeededPop = size(newPop,1) - lastRowNewPop;
      % Define last row of currentFrontPop
      lastRowNewPop = newPopRow + numNeededPop - 1;
               % Loop over sortedPop and include ranks above K for final
      % review for addition to newPop
      % Set up counter for next Loop
      counterK = 1;
      for k = 1:size(sortedPop,1)
         % Check rank > k and add to extraPop
         if sortedPop(k,e-1) > K
            leftOverPop(counterK,:) = sortedPop(k,:);
            % Increase counter by 1
            counterK = counterK +1;
        end
    end
    % Add ranks larger than K to extraPop
    extraPop = [extraPop;leftOverPop];

    % Use tournment selection for fill up newPop with "left overs"
    newPop(newPopRow:lastRowNewPop,:) = tournSelectNSGA2_r(extraPop,[gen numNeededPop]);
else
    % Loop for each rank and take individuals as specified by "r"
    % to fill up newPop
    % Count number of individuals in current front
    numSortedPop = size(sortedPop,1);
    % Counter for loop
    counter = 1;
    % Loop over all of sortedPop to fine current front of interest
    for i = 1:numSortedPop
        if sortedPop(i,e-1) == rank
            currentFrontPop(counter,:) = sortedPop(i,:);
            counter = counter + 1;
        end
    end
    % Get number of individuals in currentRank
    nCurrentFront = size(currentFrontPop,1);

    % Make sure the current Front has a population great than 0.
    % If it doesn't then skip to end of if statement.
    if nCurrentFront ~= 0
        % Define last row of new rank
        lastRowNewPop = newPopRow + nCurrentFront - 1;

        %%%%%%% Possible Cases for "r" implimentation %%%%%%%
        % Case 1: Equal number of individuals in current non
        %          dominated front as allowed by "r"
        % Case 2: More individuals in current non dominated front
        %          than allowed by "r"
        % Case 3: Less individuals in current non dominated front
        %          than allowed by "r"
        %%%% Case 1 %%%%
        if nCurrentFront == numInRank(rank)
            % If size of Front and number allowed by "r" the same then use all
            % of current front
            disp('Case 1')
            newPop(newPopRow:lastRowNewPop,:) = currentFrontPop;
            currentFrontPop = [];
        %%%% Case 2 %%%%
        elseif nCurrentFront > numInRank(rank)
            % If size of Front is more than allowed by "r" then use the
            % "tournSelectNSGA2" and take the indivuald with larger
            % crowding distances by a random tournament.
            % This makes sure we good distribution of individuals in front.
            % We don't take just the largest crowding
            % distance b/c that would focus to much on the "outer" solns.
            disp('Case 2')
            % Define last row of new front to be added to newPop
            lastRowNewPop = newPopRow + numInRank(rank) - 1;
            % Add new front to newPop
            newPop(newPopRow:lastRowNewPop,:) = tournSelectNSGA2_r(currentFrontPop,[gen numInRank(rank)]);

            % Since this is case 2, extra individuals exist.  Find
            % their positions in "pos" vector added above
            newRemovePos = newPop(newPopRow:lastRowNewPop,e+1);
            % Reset killRow varaible
            killRow = [];
            % Loop for each chromosome to remove
            for m = newRemovePos'
                % Loop over each row to locate the chromosome to be
                % removed
                for j = 1:size(currentFrontPop,1)
                    % If statement used to find location in
```

```
                        % currentFrontPop matrix so it can be removed
                        if currentFrontPop(j,e+1) == m
                            % killRow contains location in
                            % currentFrontPop of chromosome to be
                            % removed
                            killRow = [killRow;j];
                        end
                    end
                end
                % Remove all used chromosomes
                currentFrontPop(killRow,:) = [];
                % Redefine left over chromosomes to be added to
                % extraPop
                leftOverPop = currentFrontPop;
            %%%% Case 3 %%%%
            elseif nCurrentFront < numInRank(rank)
                disp('Case 3')
                % Since there are less individuals than specified by
                % "r" take all avaible for current non dominated front
                % and add to newPop
                newPop(newPopRow:lastRowNewPop,:) = currentFrontPop;
                % Now change numInRank for current rank to the number
                % actually added to newPop and increase the next rank
                % by the number of extra individuals specified by "r".
                % This will change the actual number of individuals in
                % each rank so updated numInRank appropriately.
                if rank ~= K
                    extra = numInRank(rank) - nCurrentFront;
                    numInRank(rank+1) = numInRank(rank+1) + extra;
                    numInRank(rank) = numInRank(rank) - extra;
                end
            end
            % Update new row for next iteration of while loop
            newPopRow = lastRowNewPop + 1;
        end
        numInRank
        extraPop = [extraPop;leftOverPop];
    end
    % End while loop when newPop is full or stop loop if to many
    % individuals added (ie logic error occured)
    % check to see if last row is still zeros
    if newPop(end,1) ~= 0
        check = 0;
    elseif lastRowNewPop > n
        disp('Error in "r" value selection, see NSGA2.m')
        stop
    end

    % Update rank for next iteration of while loop
    rank = rank + 1;
    newPop = newPop
end
% Redefine startPop and remove position vector
startPop = newPop(:,1:end-1);

%+++++++Define Children Population (ie startPop)+++++++
%=== Crossover operator ===
% The basic crossover rate is set in runopt to be something like 0.8.
% Then, xoverNum is calculated and passed through the call to NSGA2.  The
% parameters are passed into this routine through xOverOps.  Note that
% numXOvers is usually one in the next loop because we only pass in the
% tournSelectNSGA2 operator.
% Number of children in chldPop
chldPop = startPop;
chldSize = size(chldPop,1);
for i = 1:numXOvers,
    % Loop through the number in the first parameter in xOverOps.
    % This number is usually something like 40 for a population of 100.
    for j = 1:xOverOps(i,1),
        % Randomly select a parent; first generate a random number between
```

```
      % 0 and 1.  Multiply by the population, etc.
      a = round(rand*(chldSize-1)+1);
      % Randomly select a second parent in the same manner.
      b = round(rand*(chldSize-1)+1);
      % Remove any trailing whitespace characters from the name in the
      % string of the crossover function.
      xN = deblank(xOverFNs(i,:));
      % Execute the 'xN' function(s) [usually one function] which is defined
      % above to be 'simpleXoverNSGA2'.  For 'bounds' see initRange in
      % runopt.
      [c1 c2] = feval (xN, chldPop(a,:), chldPop(b,:), bounds, [gen xOverOps(i,:) numObj]);
      % Move the 'c1' and 'c2' chromosomes into the endPop.
      chldPop(a,:) = c1;
      chldPop(b,:) = c2;
   end
end
%=== Mutation operator ===
% [mutNum mutGenNum 3]
% Here only one mutation operator is used: nonUnifMutateNSGA2
% So, numMuts = 1 and only execute once the next 'for' loop
for i = 1:numMuts,
   % Loop through the number of mutations;
   % Incoming parameters:  mutOps = [mutNum mutGenNum 3]
   % From runopt: mutNum = mutRate*popSize;
   % So, typically, mutRate*popSize = 0.2 x 100 = 20
   for j = 1:mutOps(i,1),
      % Randomly select a parent; first generate a random number between
      % 0 and 1; multiply by the population, etc.
      a = round(rand*(chldSize-1)+1);
      % Remove any trailing whitespace characters from the name in the
      % string of the mutation function.
      % Execute the 'nonUnifMutateNSGA2'function which is defined above.
      % For 'bounds' see initRange in runopt.
      % Incoming parameters mutOps from runopt are [mutNum mutGenNum 3].
      % Also from runopt: mutNum =  mutRate*popSize.  numObj is the number
      % of objectives in this problem.
      c1 = feval (deblank(mutFNs(i,:)), chldPop(a,:), bounds, [gen mutOps(i,:) numObj]);
      % Add the mutated chromosome to the population
      chldPop(a,:) = c1;
   end
end
% Define endPop with "exploited" startPop and "explored" population
% chldPop
startPopsize = size(startPop);
chldPopsize = size(chldPop);
endPop = [startPop;chldPop];

%=== Evaluate fitness of all the chromosomes in the new generation
endSize = size(endPop,1)
for i = 1:endSize
   % Transfer the new population (endPop) into the temporary 'c1' array.
   c1 = endPop(i,:);
   % Execute the e1str string as a Malab command (see above for definition)
   % This is the location for eventual parallelization.
   eval(e1str);
   % Move the 'c1' population back into endPop, but this time the
   % performance indices are known.
   endPop(i,:) = c1;
end
disp('stop')
endPop
save endPop endPop
%=== End of NSGA2 ========================================================
% Wrap up this generation and prepare for the next generation.
% Save non-dominated individuals from endPop.
nonDomPops = getNonDomPop (endPop, [numObj 2]);
% Determine the number of non-dominated individuals in the endPop
numNonDom = size(nonDomPops,1);
% The array traceInfo that is formed below gives the multiobjective
% information for rank 1 fronts for each generation.  This can be plotted
```

```
   % out after a run is complete to look at the advance of the front with
   % the number of the generation.
   % See definition of 'collectTrace' above (nargout > 3)
   if collectTrace
      % Put the current generation number into the traceInfo array
      traceInfo(gen,1) = gen;
      % Save the objectives of all non-dominated individuals.
      for itmp = 1:numNonDom
         for jtmp = 1:numObj
            traceInfo(gen,1+(itmp-1)*numObj+jtmp) = nonDomPops(itmp,numVar+jtmp);
         end
      end
   end
   % Write output to screen display; see definition as opts(3) above.
   % In runopt.m the line 'dispOnOff = 1' sets the display to 'on.'
   if display
      % Update the display with the current generation number.  The first
      % argument in the next command, '1', is a file identifier specifying
      % that the output is to be written to the screen.  '\n' requests a line
      % feed.
      fprintf(1,'\n Current Gen: %d, Num of Non-dominated Pop: %d\n',gen,numNonDom);
   end
   % Generate a column vector with the generation number.  Length of the
   % vector is equal to the length of the non-dominated population.
   genVec = ones(numNonDom,1)*gen;
   % Place the column of generation numbers to the left of the array of
   % non-dominated individuals and add the entire matrix to the bottom of
   % the bPop array.  That is, keep a running list of the best population
   % members.
   bPop(bFoundIn:bFoundIn+numNonDom-1,:) = [genVec nonDomPops];
   % Update the counter of the number of non-dominated population members
   % that have been saved.
   bFoundIn = bFoundIn + numNonDom;
   % Create a new array 'x' that containes the non-dominated population
   x = nonDomPops;
   %if gen==10 | gen==20 |gen==30 |gen==40 |gen==50 |gen==60 |gen==70 |gen==80 |gen==90 |gen==100 |gen==110
|gen==120 |gen==130 |gen==140 |gen==150 |gen==160 |gen==170 |gen==180 |gen==190 |gen==200
   % According to the list below, save important information in a
   % 'garesults 10' type file at frequent intervals of generations
   % so that a restart of the optimization process can be made if necessary.
   save currentGenResponse
   if gen==1 | gen==5 | gen==10 | gen==15 | gen==20 | gen==25 | gen==30 | gen==40 | gen==50 | gen==60 | gen==70 |
gen==80 | gen==90 | gen==100 | gen==110 | gen==120 | gen==130 | gen==140 | gen==150 | gen==160 | gen==170 |
gen==180 | gen==190 | gen==200
      sName = sprintf ('%s %d', 'garesults', gen);
      save (sName, 'traceInfo', 'x', 'bPop', 'endPop', ...
         'uncon_peak_drift_Max', 'uncon_peak_acc_Max', 'uncon_peak_baseShear_Max', ...
         'uncon_rms_drift_Max',  'uncon_rms_acc_Max',  'uncon_rms_baseShear_Max' );
   end
   % Check to see if the GA optimization is complete. Run termFN.m function
   % which is usually 'maxGenTerm'.  The logic is simple:  'done' is true if
   % the current 'gen' is greater than or equal to the maximum number of
   % generations.
   done = feval (termFN, [gen termOps], bPop, endPop);
   % Iterate the generation number in preparation for another run.
   gen = gen + 1;
end
%=== End of GA ==========================================================

% Save a results file that summarizes the final population.
% After the run is complete a careful inspection of the rank one
% population individuals is made and one of these chromosomes is selected
% to serve as the controller based on engineering judgment of which one
% (or more) of the objectives is important.
% Also the uncontrolled values are saved in case of a restart, because in a
% restart run the initializeNSGA2 is not executed.
% Format:  save ( filename, x, y, z,.... etc variables)
save ('garesults', 'traceInfo', 'x', 'bPop', 'endPop', ...
   'uncon_peak_drift_Max', 'uncon_peak_acc_Max', 'uncon_peak_baseShear_Max', ...
   'uncon_rms_drift_Max',  'uncon_rms_acc_Max',  'uncon_rms_baseShear_Max' );
```

B. 3. getNonDomSort.m

```
function [rankedPop] = getNonDomSort(curPop, numObj)
% Get ranking and crowding distance for all individuals in the population.
% Rank of the first Pareto front is 1.
%
% function [rankedPop] = getNonDomSort(curPop, numObj)
% rankedPop - this output matrix is the final ranked population
% curPop  - the current population and is the input population to this
% routine.
% numObj - number of objectives in a multiple objective problem
%   Modified by PR   13 December 2004

%
% Find the length of a chromosome; i.e. numVar + numObj + rank + dist
e = size(curPop,2);
% Find the number of individuals in the population
n = size(curPop,1);
% Set up a vector of zeros for the rank of each population member.
rankVec = zeros(n,1);
% Set up a vector of zeros for the crowding distance of each population
% member.
distVec = zeros(n,1);
% Initialize the number of the rank to unity (for starting)
nRank = 1;
% Initialize the size of the population. This variable changes as the
% population is added or subtracted.
nCurPopNum = n;
% Set up a temporary array that can be modified during the sort process
tmpPop = curPop;

% Cycle through the following procedure until all of the current population
% members have been eliminated.
while (nCurPopNum > 0)
  % Place the non-dominated members of the population in array nonDomPop
  % and make a vector 'pos' that gives the row number of each population
  % member in the old population.
  [nonDomPop pos] = getNonDomPop (tmpPop, [numObj 2]);
  % Determine the number of non-dominated members in the new population.
  numNonDomPop = size (nonDomPop,1);
  % For the nonDomPop calculate the crowding distance
  [nonDomPop] = calcCrowdDist (nonDomPop, numObj);
  % Set up an empty array that will hold the 'old' position number.
  oldPos = [];
  % Loop through each of the non-dominated rows
  for itmp = 1:numNonDomPop
    % Put the row location of each non-dominated individual in the rankVec
    % in the oldPos array.
    % getPosInOldPop.m is a subfunction that is written below.
    % 'rankVec' begins as an array of zeros. 'n' is the number of
    % individuals in the sorted population, and
    % 'pos' is the row number in the original array.
    oldPos(itmp) = getPosInOldPop(rankVec, n, pos(itmp));
  end
  % Set rank and distance into their arrays
  rankVec(oldPos(:),1) = nRank;
  distVec(oldPos(:),1) = nonDomPop(:,e);

  tmpPop(pos(:),:) = [];
  nRank = nRank + 1;
  nCurPopNum = size(tmpPop,1);
end
% Put the incoming curPop (current population) array into the new array
% rankedPop.  Then add the rankVec and distVec columns to the array.
% That is, at the end of this routine, the new population is really just
% the old population, but with the ranking and distance vectors added.
rankedPop = curPop;
rankedPop(:,e-1) = rankVec;
rankedPop(:,e)   = distVec;
```

```
%====== Sub Function=================================================================
function oldPos = getPosInOldPop (rankVec, nTot, nPos)

% rankVec begins as a column of zeros.
% nTot
% nPos
% Calling:     oldPos(itmp) = getPosInOldPop(rankVec, n, pos(itmp));
% This subfunction determines the number of the chromosome in the original
% unsorted array of chromosomes.
% Modified by PR    10 December 2004

  % Initialize an index
  nCumNum = 0;
  % Loop through all of the population being considered and sorted.
  for itmp = 1:nTot
    % Look for a non-dominated element
    if rankVec(itmp,1) == 0
      % Increase the number of the index by one if a non-dominated element
      % is found.
      nCumNum = nCumNum+1;
      if nCumNum == nPos
        oldPos = itmp;
        break;
      end
    end
  end
```

B. 4. getNonDomPop.m

```
function[newPop pos] = getNonDomPop (oldPop, options)
% getNonDomPop.m selects non-dominated individuals from array 'oldPop.'
% This routine follows the discussion on page 4 of the paper by Deb,
% Agrawal, Pratap, and Meyarivan.
%
% function[newPop] = getNonDomPop(oldPop,options)
% newPop  - the nonDominated individuals
% pos     - position of nonDominated individuals in old population
% oldPop  - the current population
% options - options to getNonDomPop [number_of_objective num_of_additional_data]

% Determine the number of objectives in the multiobjective problem.
nobj = options(1);
% Bring in additional data from options(2)
naddData = options(2);
% Determine the length of a chromosome that is to be worked on, but do not
% include the rank and distance columns.  e = number of variables + numObjs
e = size(oldPop,2)-naddData;
% Determine the number of individuals in the population
n = size(oldPop,1);
% Set up a new array that will hold the number of individuals that dominate
% each of the members of the population.
donNum = zeros(n,1);

% Loop through each individual in the population.  The purpose of this
% entire set of statements is to determine for each individual how many other
% individuals lie in the 3rd quadrant from this individual.  Note that this
% 3rd quadrant requirement must hold for all of the objectives.
% This number is placed in the donNum array.
for itmp = 1:n
  % Loop through all other individuals in the population
  for jtmp = 1:n
    if jtmp ~= itmp

      % Assume that the jtmp chromosome dominates the itmp chromosome.
      bDominate = 1;
      % Loop through each objective function
      for ktmp = 1:nobj
        % Check to see if the itmp chromosome dominates the jtmp
        % chromosome for any of the objectives.  If this is true then
        % the jtmp chromosome does not dominate the itmp chromosome.
        if oldPop(itmp,e-nobj+ktmp) < oldPop(jtmp,e-nobj+ktmp)
          % Change the assumption that jtmp dominates.
          bDominate = 0;
          break;
        end
      end
      % For the case that jtmp dominates itmp for at least one objective:
      if bDominate == 1
        % Loop through each objective function
        for ktmp = 1:nobj
          % Check to see if the itmp chromosome is dominated by the jtmp
          % chromosome for all objective functions.
          if oldPop(itmp,e-nobj+ktmp) > oldPop(jtmp,e-nobj+ktmp)
          bDominate = 2;
          break;
          end
        end
      end
      % For the case that variable bDominate is still 2, the jtmp
      % chromosome dominates the itmp individual.
      if bDominate == 2 % individual(jtmp) dominate individual(itmp)
        % Increase the count of the number of dominant individuals for
        % itmp.
        donNum(itmp,1) = donNum(itmp,1)+1;
      end

    end
  end
end
% The number of 3rd quadrant dominant chromosomes should be known for each
% member of the population.

% Set up a counter for the number of non-dominated individuals that have
% been found.
numNewPop = 1;
% Loop through each member of the entire incoming population
for itmp = 1:n
  % If there are no individuals that dominate this chromosome:
  if donNum(itmp,1) == 0
    % Check to see if an identical individual exists.  Assume there is
    % none.
    bExist = 0;
    % Loop through all other members of the population
    for jtmp = 1:numNewPop-1
      % Compare all variables (but not the objectives and the rank and
      % distance).  Note that 'end' is a specialized Matlab name that goes
      % to the end of an array dimension.  (I.e. you do not need to use the
      % 'size' function.)
      if newPop(jtmp,1:(end-nobj-naddData))==oldPop(itmp,1:(end-nobj-naddData))
      bExist = 1;
      % Do not include this element in the non-dominated population.
      break;
      end
    end
    % If no identical individual has been found to exist
    if bExist == 0
      % Add the chromosome to the newPop of non-dominated individuals
      newPop(numNewPop,:) = oldPop(itmp,:);
      % Place the row number of the non-dominated individual in the 'pos'
      % array that is passed back to the calling routine.
      pos(numNewPop) = itmp;
      % Increase the counter of the number of non-dominated individuals.
      numNewPop=numNewPop+1;
    end
  end
end
```

B. 5. calcCrowdDist.m

```
function [calcPop] = calcCrowdDist(curPop, numObj)
% Calculate crowding distance of each individuals
%
% function [calcPop] = calcCrowdDist(curPop, numObj)
% calcPop  - population having calculated crowding distance
% curPop  - the current population
% numObj - number of objectives

% Determine the number of individuals in the incoming population
nPop = size(curPop,1);
% Determine the length of the chromosome, which is the number of changeable
% variables plus the number of objective functions.  Disregard the rank and
% distance variables.  The latter variable will be determined in this
% routine.
nCol = size(curPop,2) - 2;

% Set up a zero-vector that will hold the crowding distances.
crowdDist = zeros(nPop,1);
% Loop through each of the objective functions and sort each column
% according to the lowest objective function
for itmp = 1:numObj
  % Sort the column in each objective function.  Place the result in vector
  % 'ytmp' and place original row number of each entry in 'xtmp' so that you
  % can (see below) get the crowding distances back in the next to last column of the
  % current population array.
  [ytmp xtmp] = sort(curPop(:,nCol-numObj+itmp));

  % Set the crowding distance for the first and last point to large numbers
  crowdDist(xtmp(1))   = 99999;
  crowdDist(xtmp(nPop))= 99999;

  % For each of the other performance-evaluated points, add the difference
  % of the two neighboring performance-evaluated points to the current
  % crowdDist value and place the value in the correct location of the
  % original population.  Note that the crowdDist vector is cumulative and
  % that it is therefore summed over all of the objective functions.
  for jtmp = 2:nPop-1
    crowdDist(xtmp(jtmp)) = crowdDist(xtmp(jtmp))+(ytmp(jtmp+1)-ytmp(jtmp-1));
  end

end

calcPop = curPop;
% Enter the crowding distances in the last column of the current population
% array.
calcPop(:,nCol+2) = crowdDist(:,1);
```

B. 6. tournSelectNSGA2_r.m

```
function[newPop winners] = tournSelectNSGA2_r(oldPop,options)
% Performs a tournament selection.
% Modified by PR    13 December 2004
%            DS    6  June     2005 - Modifed for "r" value (see "Case 2"
%                                    in NSGA2.m)
%
% function[newPop] = tournSelectNSGA2(oldPop,options)
% newPop  - the new population selected from the oldPop
% oldPop  - the current population
% options - options to normGeomSelect [gen population_size_of_newPop]

% Binary and Real-Valued Simulation Evolution for Matlab
% Copyright (C) 1996 C.R. Houck, J.A. Joines, M.G. Kay
%
% C.R. Houck, J.Joines, and M.Kay. A genetic algorithm for function
% optimization: A Matlab implementation. ACM Transactions on Mathmatical
% Software, Submitted 1996.
%
% This program is free software; you can redistribute it and/or modify
% it under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 1, or (at your option)
% any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details. A copy of the GNU
% General Public License can be obtained from the
% Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

% Define options
gen = options(1);
% Size of newPop allowed by options 2
newPopSize = options(2);

% Set the number of tournaments to two.
% That is, compare two individuals from the population and select the best
% of the two.
tournSize = 2;
% Determine the chromosome length = variables + objectives + rank +
% distance.  Be careful for the -1, came from a pos vector at the end :)
e = size(oldPop,2)-1;
% Create an array of zeros that will hold the newPop after the tournament
% selection is complete
newPop = zeros(newPopSize,e+1);
%
% Create an array that is 'tournSize x n' and is filled with random numbers
% that range in magnitude from 0 to tournSize.  This is the schedule for the
% tournament.
nOldPop = size(oldPop,1);
tourns = round(rand(tournSize,nOldPop)*nOldPop);

% Make sure there aren no zeros in tourns and there isn't a tourny between
% the same two chromosomes.
for i = 1:nOldPop
    if tourns(1,i) == 0
        tourns(1,i) = 1;
    end
    if tourns(2,i) == 0
        tourns(2,i) = 1;
    end
    if tourns(2,i) < nOldPop
        if tourns(1,i) == tourns(2,i)
            tourns(2,i) = tourns(2,i)+1;
        end
    elseif tourns(2,i) == nOldPop
        if tourns(1,i) == tourns(2,i) & tourns(2,i) ~= 1
            tourns(2,i) = tourns(2,i)-1;
        end
    end
end

% Determine the winner of the tournaments
% From the above definition of 'e' notice that, for example, the notation
% 'e-1' means the same thing as 'end-1'.

for itmp = 1:newPopSize
    % Compare two chromosomes (if 2 is better than 1 use it)
    if oldPop(tourns(1,itmp),e-1) > oldPop(tourns(2,itmp),e-1)
        newPop(itmp,:) = oldPop(tourns(2,itmp),:);
    % Otherwise, if 1 is better than 2, use it
    elseif oldPop(tourns(1,itmp),e-1) < oldPop(tourns(2,itmp),e-1)
        newPop(itmp,:) = oldPop(tourns(1,itmp),:);
    % if rank is identical, longer distance is better
    else
        if oldPop(tourns(1,itmp),e) > oldPop(tourns(2,itmp),e)
            newPop(itmp,:) = oldPop(tourns(1,itmp),:);
        else
            newPop(itmp,:) = oldPop(tourns(2,itmp),:);
        end
    end
end
```

B. 7. simpleXoverNSGA2.m

```
function [c1,c2] = simpleXoverNSGA2(p1,p2,bounds,Ops)
% Simple crossover takes two parents P1,P2 and performs simple single point
% crossover in multiobjective optimization.
%
% function [c1,c2] = simpleXoverNSGA2(p1,p2,bounds,Ops)
% p1      - the first parent ( [solution string function value] )
% p2      - the second parent ( [solution string function value] )
% bounds  - the bounds matrix for the solution space
% Ops     - Options matrix for simple crossover [gen #SimpXovers 0 numObjs].

% Modified by Hyun-Su Kim, 2004-10-14
% Modified by PR  12 December 2004
% Binary and Real-Valued Simulation Evolution for Matlab
% Copyright (C) 1996 C.R. Houck, J.A. Joines, M.G. Kay
%
% C.R. Houck, J.Joines, and M.Kay. A genetic algorithm for function
% optimization: A Matlab implementation. ACM Transactions on Mathmatical
% Software, Submitted 1996.
%
% This program is free software; you can redistribute it and/or modify
% it under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 1, or (at your option)
% any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details. A copy of the GNU
% General Public License can be obtained from the
% Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

% Determine the number of objectives
numObj = Ops(4);
% Determine the number of variables by finding the length of the entire
% chromosome and subtracting the number of objectives and 2 for the rank
% and crowding distance columns.
numVar = size(p1,2)-numObj-2;
% Pick a cut point randomly from 1-number of vars
cPoint = round(rand * (numVar-2)) + 1;

% Create two children by switching the 'tail' end of both chromosomes.
c1 = [p1(1:cPoint) p2(cPoint+1:numVar+numObj+2)];
c2 = [p2(1:cPoint) p1(cPoint+1:numVar+numObj+2)];
```

B. 8. nonUnifMutateNSGA2.m

```
function [parent] = nonUnifMutateNSGA2 (parent, bounds, Ops)
% Non uniform mutation changes one of the parameters of the parent
% based on a non-uniform probability distribution in multiobjective optimization.  This Gaussian
% distribution starts wide, and narrows to a point distribution as the
% current generation approaches the maximum generation.
%
% function [newSol] = nonUnifMutateNSGA2(parent,bounds,Ops)
% parent  - the first parent ( [solution string function value] )
% bounds  - the bounds matrix for the solution space
% Ops     - Options for nonUnifMutate[gen #NonUnifMutations maxGen b numObj]

% Binary and Real-Valued Simulation Evolution for Matlab
% Copyright (C) 1996 C.R. Houck, J.A. Joines, M.G. Kay
%
% C.R. Houck, J.Joines, and M.Kay. A genetic algorithm for function
% optimization: A Matlab implementation. ACM Transactions on Mathmatical
% Software, Submitted 1996.
%
% This program is free software; you can redistribute it and/or modify
% it under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 1, or (at your option)
% any later version.
%
% This program is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details. A copy of the GNU
% General Public License can be obtained from the
% Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
% Modified  PR    12 December 2004

% Determine the number of the current generation
cg = Ops(1);
% Determine the maximum number of generations from mutGenNum in runopt.
mg = Ops(3);
% Determine the shape parameter (usually use 3)
b = Ops(4);
% Determine the number of objectives
numObj = Ops(5);
% Determine the range of each variable
df = bounds(:,2) - bounds(:,1);
% Determine the number of variables in the parent chromosome
numVar = size(parent,2) - numObj - 2;
% Randomly pick a variable (from 1 to number of vars) to mutate
mPoint = round(rand * (numVar-1)) + 1;
% Round a random number (between 0 and 1) to the nearest interger (0 or 1).
md = round(rand);
if md
  % Mutate toward the upper bound (see simple function delta.m)
  newValue = parent(mPoint)+delta(cg,mg, bounds(mPoint,2)-parent(mPoint), b);
else
  % Mutate towards lower bound (see simple function delta.m)
  newValue = parent(mPoint)-delta(cg,mg, parent(mPoint)-bounds(mPoint,1), b);
end
% Make the mutated child by placing the mutated chromosome into the parent
% as output from the function.
parent(mPoint) = newValue;
```

B. 9. nonUnifMutateNSGA2.m

```
function [chrom fitness] = fitfunc1_gauss (chrom, param)
% fitfunc1_gauss.m   Defines the fitness function for a four story building for
% use with NSGA-2 optimization.
% Four inputs (1st, 2nd, 3rd, and 4th floor accelerations) and one output (voltage to the damper)
% are used for the fuzzy logic inference system.
% Inputs for this fitfunc1_gauss.m function are
%    1. The chromosome chrom whose fitness is to be evaluated
%    2. A vector of parameters related to the chrom
% Outputs for this function are:
%    1. The chromosome string
%    2. The fitness of the chromosome
% Use Gaussian bell membership functions.
% There is a total 'x' rules and only one direction of motion.
% Each rule consists of 10 variables - 2 for each of four inputs and 2 for the output.
% Gaussian membership functions require 2 parameters to define their
% location and shape.  Also see runopt_gauss.m for the range of each variable.
% For one case, there is a total of 50 variables (5 rules x 10
% variables/rule).
% Modified PNR  1 Feb 2005

% Declare golbal variables so that they are available.
glbvar;

% Define the length of a subvector from the input vector 'chrom.'
% For this substring, exclude the objectives, rank, distance, and device
% locations (if any - not used for the building)
% from the original vector coming into this fitness function
% m-file.  All variables in the mutatable chromosome chrom are real-valued
% variables, not integers - although this can also be programmed but has
% not been done so far for simplicity.

% nChromLen = length(chrom)-param(2)-2-devNum;
nChromLen = length(chrom)-param(2)-2;
%=======================================================================================
% Use the getInteger function (see bottom of this m-file) to set the
% real-valued damper location to an integer; i.e. round down.

% Go to the story below the story of the current damper.  The MR damper
% lies between these two stories and is assumed to be connected with a
% chevron brace.
% Initialize a vector that shows the location of damper 1.  All are zeros
% except where the damper is located.  Also, account for action and
% 'reaction', namely, add another force on the story below in the opposite
% direction.  An exception occurs when the MR damper is located in the
% first story and the reaction forces are transmitted to the foundation.
%=======================================================================================

% Units: m, N, sec
% Define the range of the accelerations from glbvar
minAcc = min_acc;
maxAcc = max_acc;

% Bring in the number of variables (nVariableNum) for each rule from glbvar.m
% For triangles with 2 inputs and 1 output: nVariableNum = 6;
% Calculate the number of rules.
% See nVariableNum definition in runopt_tri.m.  It is passed through
% glbvar.m.
nRuleNum = nChromLen/nVariableNum;

%====== Start of making FLC FIS
  Input1MF  = []; % Gaussian Bell Shape MF
  Input2MF  = []; % Gaussian Bell Shape MF
  Input3MF  = []; % Gaussian Bell Shape MF
  Output1MF = []; % Gaussian Bell Shape MF
  ruleList  = []; % Value=[MF ID of Input1, MF ID of Output1, Weight, Connection] X number of Rules(number of row)

% Cycle through each of the rules that are to be created.  First look
% through all of the elements of the chromosome.  Except for repeated ones,
% first enter the number of unique membership functions that are in the
% chromosome.  Usually, there will be as many membership functions for each
% input and each output as there are rules.  That is, for 20 rules, there will
% be 20 membership functions for each of two inputs and 20 membership
% functions for each of two outputs.  Triangular membership functions are
% used.  Start with an empty set for each membership function list.
% Note:  some of the notation below is used in preparation for having
% accelerometers on all four floors.  For now, only the 2nd and 4th floors
% are assumed to have accelerations fed back to the controller.
  for m = 1:nRuleNum

    %=== Set up membership functions for Input1 (1st floor accelerations)
    [nMFNum1 nTmp] = size(Input1MF);
    bExistFg = 0;
    for n = 1:nMFNum1
      % Check for a previously existing membership function, which is not very
      % likely because of use of real-valued parameters, decimal
      % representation, etc.
      if (chrom(1+(m-1)*nVariableNum)==Input1MF(n, 1))...
              & (chrom(2+(m-1)*nVariableNum)==Input1MF(n, 2))
        imf1 = n;
        bExistFg = 1;
        break;
      end
    end
    if bExistFg == 0
      Input1MF(nMFNum1+1,:) = [chrom(1+(m-1)*nVariableNum) chrom(2+(m-1)*nVariableNum)];
      imf1 = nMFNum1+1;
    end

    %=== Set up membership functions for Input2 (2nd floor accelerations)
    [nMFNum2 nTmp] = size(Input2MF);
    bExistFg = 0;
    for n = 1:nMFNum2
      % Check for a previously existing membership function, which is not very
      % likely because of use of real-valued parameters, decimal
      % representation, etc.
      if (chrom(3+(m-1)*nVariableNum)==Input2MF(n, 1))...
              & (chrom(4+(m-1)*nVariableNum)==Input2MF(n, 2))
        imf2 = n;
        bExistFg = 1;
        break;
      end
    end
    if bExistFg == 0
      Input2MF(nMFNum2+1,:) = [chrom(3+(m-1)*nVariableNum) chrom(4+(m-1)*nVariableNum)];
      imf2 = nMFNum2+1;
    end

    %=== Set up membership functions for Input3 (3rd floor accelerations)
    [nMFNum3 nTmp] = size(Input3MF);
    bExistFg = 0;
    for n = 1:nMFNum3
      % Check for a previously existing membership function, which is not very
      % likely because of use of real-valued parameters, decimal
      % representation, etc.
      if (chrom(5+(m-1)*nVariableNum)==Input3MF(n, 1))...
              & (chrom(6+(m-1)*nVariableNum)==Input3MF(n, 2))
        imf3 = n;
        bExistFg = 1;
        break;
      end
    end
    if bExistFg == 0
      Input3MF(nMFNum3+1,:) = [chrom(5+(m-1)*nVariableNum) chrom(6+(m-1)*nVariableNum)];
      imf3 = nMFNum3+1;
    end

    %=== Set Output1 MF
    [nMFNumO1 nTmp] = size(Output1MF);
    bExistFg = 0;
    for n = 1:nMFNumO1
      if (chrom(7+(m-1)*nVariableNum)==Output1MF(n, 1))...
              & (chrom(8+(m-1)*nVariableNum)==Output1MF(n, 2))
        omf1 = n;
        bExistFg = 1;
        break;
      end
    end
    if bExistFg == 0
      %%Output1MF(nMFNumO1+1,:) = [chrom(5+(m-1)*nVariableNum) chrom(6+(m-1)*nVariableNum)];
      Output1MF(nMFNumO1+1,:) = [chrom(7+(m-1)*nVariableNum) chrom(8+(m-1)*nVariableNum)];
      omf1 = nMFNumO1+1;
    end
        % Add rules and combine to make an array
    % Variable 'wf' indicates the weight that is applied to the rule.
    % Variable 'andOr' is '1' for the 'AND' operator, and '0' for 'OR.'
    wf = 1;
    andOr = 1;
    ruleList(m,:) = [imf1, imf2, imf3, omf1, wf, andOr];
    %ruleList(m,:) = [imf1, imf2, imf3, imf4, omf1, wf, andOr];
    %ruleList(m,:) = [imf1, imf2, imf3, imf4, omf1, omf2, omf3, omf4, wf, andOr];
  end

  %%%%% All rules should now have been created.
  % Create a new fuzzy inference system with the filename flcFis.fis.
  flcFis = newfis('flcFis');
  % Add the acceleration variable (1st floor) as an input for the fis.
  % Also, convert two-parameter triangles to three-parameter triangles.
  flcFis = addvar(flcFis, 'input', 'Acceleration 1', [minAcc maxAcc]);
  [nMFNum1 nTmp] = size(Input1MF);
  for n = 1:nMFNum1
    mfName = sprintf('MF%d', n);
    flcFis = addmf(flcFis, 'input', 1, mfName, 'gaussmf', [Input1MF(n,1) Input1MF(n,2)]);
  end
    % Add the acceleration variable (2nd floor) as an input for the fis.
  % Also, convert two-parameter triangles to three-parameter triangles.
  flcFis = addvar(flcFis, 'input', 'Acceleration 2', [minAcc maxAcc]);
  [nMFNum2 nTmp] = size(Input2MF);
  for n = 1:nMFNum2
    mfName = sprintf('MF%d', n);
    flcFis = addmf(flcFis, 'input', 2, mfName, 'gaussmf', [Input2MF(n,1) Input2MF(n,2)]);
  end
  % Add the acceleration variable (3rd floor) as an input for the fis.
  % Also, convert two-parameter triangles to three-parameter triangles.
  flcFis = addvar(flcFis, 'input', 'Acceleration 3', [minAcc maxAcc]);
  [nMFNum3 nTmp] = size(Input3MF);
  for n = 1:nMFNum3
    mfName = sprintf('MF%d', n);
    flcFis = addmf(flcFis, 'input', 3, mfName, 'gaussmf', [Input3MF(n,1) Input3MF(n,2)]);
  end
  % Add the voltage variable for MR damper 1 as an output for the fis
  % Also, convert two-parameter triangles to three-parameter triangles.
  flcFis = addvar(flcFis, 'output', 'Voltage1', [MR_volt_sat_min MR_volt_sat_max]);
  [nMFNumO1 nTmp] = size(Output1MF);
  for n = 1:nMFNumO1
    mfName = sprintf('MF%d', n);
    flcFis = addmf(flcFis, 'output', 1, mfName, 'gaussmf', [Output1MF(n,1) Output1MF(n,2)]);
  end
  flcFis = addrule(flcFis, ruleList);
%====== End of making FLC FIS
% Run simulation file runsim.m for structure to be analyzed.  This m-file
% calls a Simulink file that carries out the actual simulation.
runsim;
fitness(1) = J1;
fitness(2) = J2;
fitness(3) = J3;
fitness(4) = J4;
% Increment the counter for the number of simulations that have been run.
nIter = nIter + 1
```

B. 10. runsim.m

```
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% runsim.m
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% This file sets up and runs the Simulink file ga_single_mr_damper.mdl
% which performs a set of analyses on a building using several forcing
% functions from earthquake loads.
%
% Script file to run simulations with a single MR damper.
% Written by Paul Roschke   18 December 2004   1 Feb 2005
% Modified by D. Shook      8  August   2005
% Units: N, m, sec
% Note:  there are two accelerometers used for feedback to the controller.
% Load global variables
glbvar ;
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Load FLC and set saturation limits for each input
% For now use acceleration feedback from 1st, 2nd, 3rd, and 4th floors
% Note:  notation prepares for having all 4 accels.
flcFis = readfis('ArtEQ_1MR_JALL_gen90_num17.fis');
sat_limits = getfis (flcFis, 'inrange');
FLC_inp1_sat_min = sat_limits(1,1);
FLC_inp1_sat_max = sat_limits(1,2);
FLC_inp2_sat_min = sat_limits(2,1);
FLC_inp2_sat_max = sat_limits(2,2);
FLC_inp3_sat_min = sat_limits(3,1);
FLC_inp3_sat_max = sat_limits(3,2);
clear sat_limits ;

% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% GA SysID Results
k1 = 1513225.8297356027;
k2 = 1120105.1448637175;
k3 = 2021016.0873949619;
zeta = 0.003393440499187;

% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% ABCD
m1 = 5800;
m2 = 5800;
m3 = 6840;
MassVec = [m1,m2,m3];
M = diag(MassVec);

K = [k1+k2 -k2     0;
      -k2     k2+k3 -k3;
       0     -k3     k3];
[C] = Calc_Damping_Torsion_3Floor(M,K,zeta);
nDOF = 3;
%  Build state space matrices for the building structure.
A = [zeros(nDOF) eye(nDOF); -inv(M)*K -inv(M)*C];
B = [zeros(nDOF); inv(M)];
C = [eye(nDOF) zeros(nDOF); zeros(nDOF) eye(nDOF); -inv(M)*K -inv(M)*C];
D = [zeros(nDOF); zeros(nDOF); inv(M)];
% Unit: Newtons
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
LocatorMatrix = [-1  1  0 ;
                  0 -1  1 ;
                  0  0 -1 ];
% +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% Run Simulink to execute analysis
sim('ga_ncree_one_mr_damper')
% Calculate Performance Indices
load response
calcPI
```

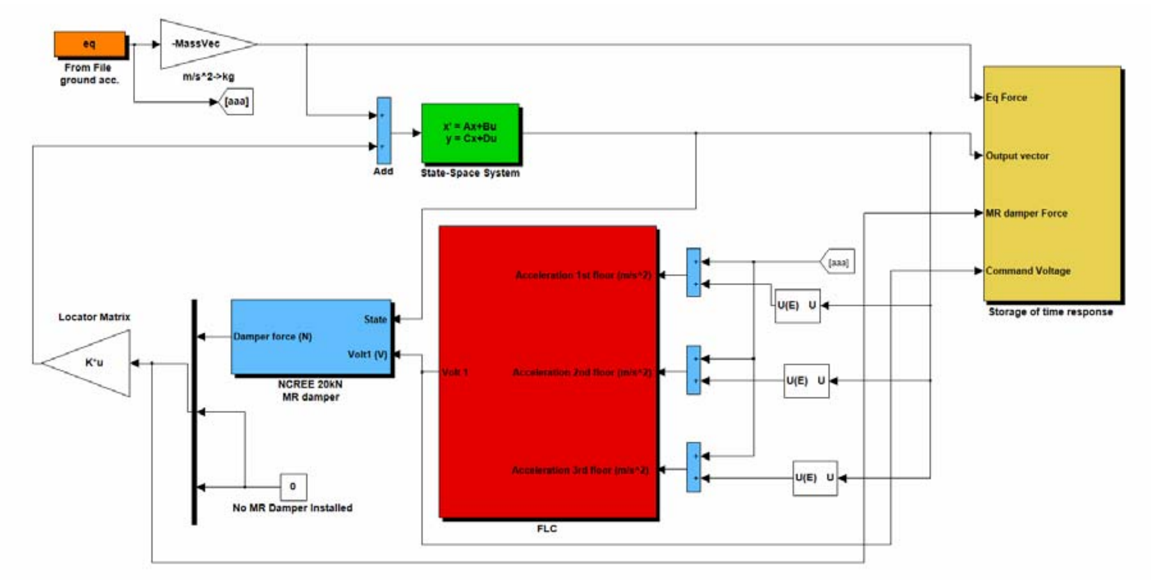
B. 11. Calc_Damping.m

```
% function [C] = Calc_Damping(M,K,zeta)

% Determine Eigen Values & Vectors
[phi,w] = eig(K,M);
wn = sqrt(w)/(2*pi);

% Transform M and K to modal coordinates
Mm = phi'*M*phi;
Km = phi'*K*phi;

% Calculate modal damping by Rayleigh damping
% Define eigen values to use in damping computations.  Use 1st and 3rd modes
w1 = w(1,1);
w2 = w(3,3);
% Solve for Alpha and Beta
alpha = zeta*2*w1*w2/(w1+w2);
beta = zeta*2/(w1+w2);

% Solveing for Modal Cm
Cm = alpha*Mm + beta*Km;

% Calculate C from Cm
C = inv(phi')*Cm*inv(phi);
```

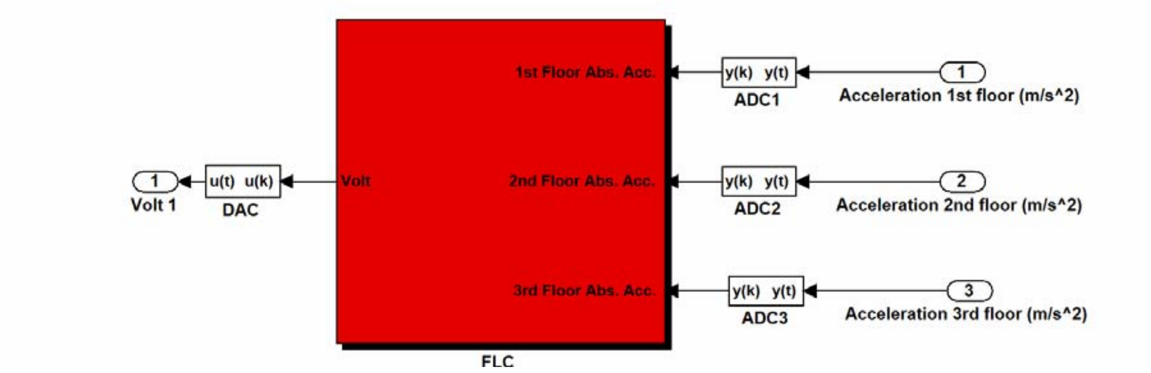
B. 12. calcPI.m

```
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% calcPI.m
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
% The order and column of saved variables in Simulink is as follows:
%%          t EQForce Disp  Vel    Accel     MR Force  Voltage EQ Accel
%% response: 1 2 3 4   5 6 7 8 9 10 11 12 13  14 15 16  17      18
% Written by Paul Roschke    19 December 2004   1 Feb 2005
% Uncontrolled peak and rms responses enter this routine through the global
% variables.
glbvar
load response
gen
% Cycle through each story level, looking for peak and rms values:
for itmp=1:nDOF
  if itmp==1
    con_peak_drift = max(abs(response(4+itmp, :)));
    con_peak_drift_Max = con_peak_drift;
    con_peak_acc  = max(abs(response(10+itmp, :)));
    con_peak_acc_Max = con_peak_acc;
    con_rms_drift = rms(response(4+itmp, :));
    con_rms_drift_Max = con_rms_drift;
    con_rms_acc  = rms(response(10+itmp, :));
    con_rms_acc_Max = con_rms_acc;
  else
    con_peak_drift = max(abs(response(4+itmp, :)-response(4+itmp-1, :)));
    con_peak_drift_Max = max(con_peak_drift_Max,con_peak_drift);
    con_peak_acc  = max(abs(response(10+itmp, :)));
    con_peak_acc_Max = max(con_peak_acc_Max,con_peak_acc);
    con_rms_drift = rms(response(4+itmp, :)-response(4+itmp-1, :));
    con_rms_drift_Max = max(con_rms_drift_Max,con_rms_drift);
    con_rms_acc  = rms(response(10+itmp, :));
    con_rms_acc_Max = max(con_rms_acc_Max,con_rms_acc);
  end
end
% Calculate Objectives
J1 = con_peak_drift_Max/uncon_peak_drift_Max;
J2 = con_peak_acc_Max/uncon_peak_acc_Max;
J3 = con_rms_drift_Max/uncon_rms_drift_Max;
J4 = con_rms_acc_Max/uncon_rms_acc_Max;
```

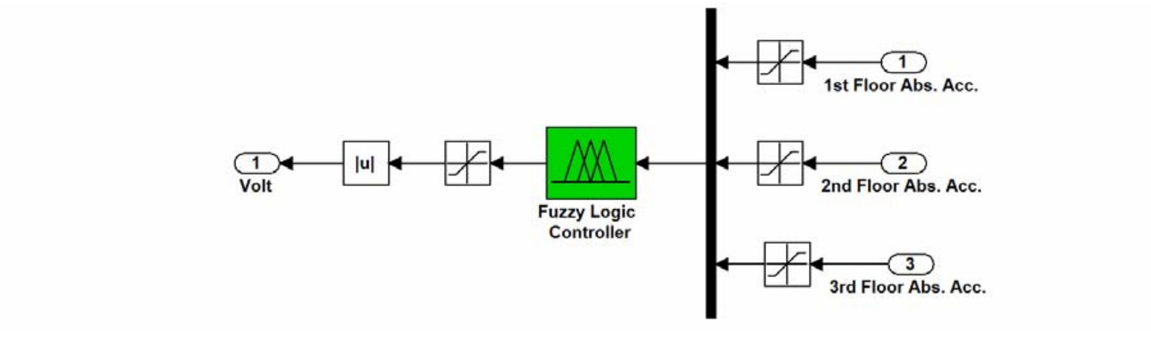
B. 13. ga_ncree_one_mr_damper.mdl



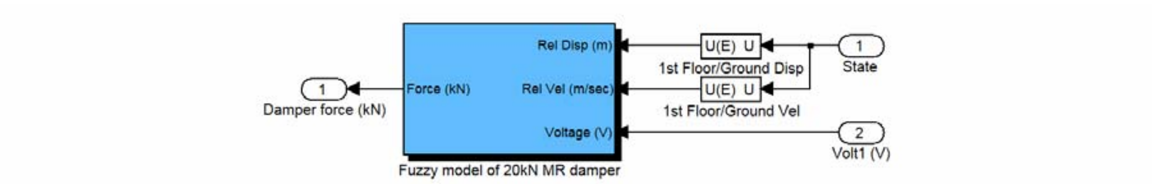
B. 14. ga_ncree_one_mr_damper/FLC



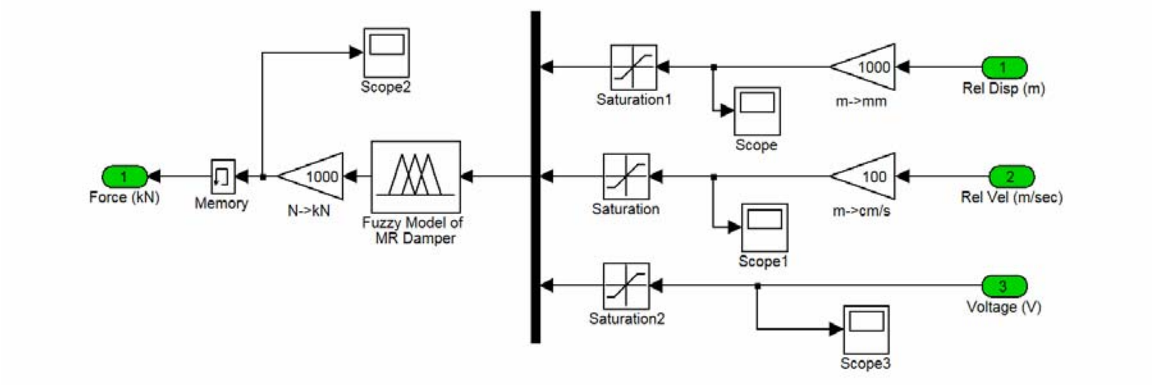
B. 15. ga_ncree_one_mr_damper/FLC/FLC



B. 16. ga_ncree_one_mr_damper/NCREE 20kN MR damper



B. 17. ga_ncree_one_mr_damper/NCREE 20kN MR damper/Fuzzy model of 20kN MR damper



APPENDIX C

MATLAB CODE FOR GA SYSTEM IDENTIFICATION

For GA system identification of the benchmark structure using the NSGA-II CE algorithm all files shown in Appendix B are required. Below are modified versions of files shown in Appendix B used to conduct GA system identification.

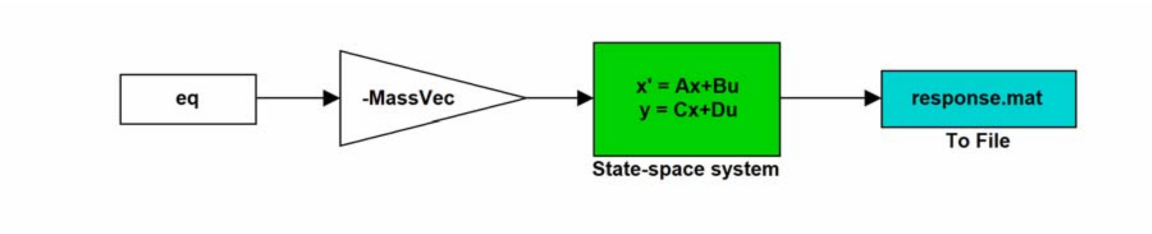
C. 1. fitfunc1_gauss.m

```
function [chrom fitness] = fitfunc1_gauss (chrom, param)

glbvar;
% This routine forms ABCD Matrices for simulation in Simulink
% Form Mass Matrix
m1 = 5800;
m2 = 5800;
m3 = 6840;
MassVec = [m1,m2,m3];
M = diag(MassVec);

% Define Stifness Values
k1 = chrom(1);
k2 = chrom(2);
k3 = chrom(3);
zeta = chrom(4);

stiff = [k1+k2 -k2     0;
    -k2     k2+k3 -k3;
    0     -k3     k3];
% Get Damping Properties

[damp,wn] = Calc_Damping(M,stiff,zeta);

%  Build state space matrices for the building structure.
A = [zeros(nDOF) eye(nDOF); -inv(M)*stiff -inv(M)*damp];
B = [zeros(nDOF); inv(M)];
C = [eye(nDOF) zeros(nDOF); zeros(nDOF) eye(nDOF); -inv(M)*stiff -inv(M)*damp];
D = [zeros(nDOF); zeros(nDOF); inv(M)];

% Run Simulink to execute analysis
tic
sim('GA_evalMCK)
toc

% Calculate Performance Indices
load response
calcPI
fitness(1) = J1;
fitness(2) = J2;
fitness(3) = J3;

% Increment the counter for the number of simulations that have been run.
nIter = nIter + 1
```

C. 2. calcPI.m

```
% calcPI.m
% This routine calculates the performance indices for the TAMU tower.
% The order and column of saved variables in Simulink is as follows:
%          t EQAccel Disp    Vel    Accel
% response: 1 2       3 4     5 6    7 8

glbvar
load response

% Make sure Simulink completed simulation.  It can stop early due to divergence of solution, note the STOP in the %
Simulink file
numPtsLab = length(t);
numPtsSim = length(response(1,:));

% If simulation completed do this
if numPtsLab == numPtsSim
   d1sim = response(2,:);
   d2sim = response(3,:);
   d3sim = response(4,:);
   v1sim = response(5,:);
   v2sim = response(6,:);
   v3sim = response(7,:);
   a1sim = response(8,:);
   a2sim = response(9,:);
   a3sim = response(10,:);
   % Fine RMS error of each response:
   d1RMSerror = rms(d1sim-d1)/rms(d1);
   d2RMSerror = rms(d2sim-d2)/rms(d2);
   d3RMSerror = rms(d3sim-d3)/rms(d3);
   v1RMSerror = rms(v1sim-v1)/rms(v1);
   v2RMSerror = rms(v2sim-v2)/rms(v2);
   v3RMSerror = rms(v3sim-v3)/rms(v3);
   a1RMSerror = rms(a1sim-a1)/rms(a1);
   a2RMSerror = rms(a2sim-a2)/rms(a2);
   a3RMSerror = rms(a3-a3sim)/rms(a3);
   % Determine J1 J2 J3
   J1 = max([d1RMSerror,d2RMSerror,d3RMSerror]);
   J2 = max([v1RMSerror,v2RMSerror,v3RMSerror]);
   J3 = max([a1RMSerror,a2RMSerror,a3RMSerror]);

   % Penalty function (no penalty)
   penalty = 0;
   J1 = J1 + penalty;
   J2 = J2 + penalty;
   J3 = J3 + penalty;

% If simulation didn't complete do this
else
   J1 = 2;
   J2 = 2;
   J3 = 2;
end

J = [J1 J2 J3]
```

C. 3. GA_evalMCK.mdl



APPENDIX D

MATLAB CODE FOR COMPARING FLCS

```
clc;clear;

% Use this file to evaluate an FLC on a number of EQ's
% Loop for each EQ record.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Read in FLC           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
flcFIS = readfis('220_ALLJ_gbell40gen.fis');
sat_limits = getfis (flcFIS, 'inrange');
FLC_inp1_sat_min = sat_limits(1,1);
FLC_inp1_sat_max = sat_limits(1,2);
FLC_inp2_sat_min = sat_limits(2,1);
FLC_inp2_sat_max = sat_limits(2,2);
FLC_inp3_sat_min = sat_limits(3,1);
FLC_inp3_sat_max = sat_limits(3,2);
max_volt = 1.2;

% Load MR FIS
MRfis1 = readfis('B1_seat_222.fis');
MR1_disp_sat_min = MRfis1.input(1).range(1);
MR1_disp_sat_max = MRfis1.input(1).range(2);
MR1_vel_sat_min = MRfis1.input(2).range(1);
MR1_vel_sat_max = MRfis1.input(2).range(2);
MR1_volt_sat_min = MRfis1.input(3).range(1);
MR1_volt_sat_max = MRfis1.input(3).range(2);

% load MR FIS for 2nd floor - Roschke 20kN
MRfis2 = readfis('B2_Seat_222.fis');
MR2_disp_sat_min = MRfis2.input(1).range(1);
MR2_disp_sat_max = MRfis2.input(1).range(2);
MR2_vel_sat_min = MRfis2.input(2).range(1);
MR2_vel_sat_max = MRfis2.input(2).range(2);
MR2_volt_sat_min = MRfis2.input(3).range(1);
MR2_volt_sat_max = MRfis2.input(3).range(2);

% Loop for several excitations
for i = 1:4
    if i == 1
        %load ELC100_ArtEQ_2MR
        load ELC200_ArtEQ_2MR
        eq = data;
        accel = eq(2,:)*9.81;
    elseif i == 2
        %load KOBE100_ArtEQ_2MR
        load KOBE200_ArtEQ_2MR
        eq = data;
        accel = eq(2,:)*9.81;
    elseif i == 3
        %load TCU76100_ArtEQ_2MR
        load TCU76200_ArtEQ_2MR
        eq = data;
        accel = eq(2,:)*9.81;
    elseif i == 4
        load TCU82100_ArtEQ_2MR
        %load TCU82200_ArtEQ_2MR
        eq = data;
        accel = eq(2,:)*9.81;
    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %             Load EQ             %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    time = eq(1,:);
    dt = time(1,2)-time(1,1);
    dt = 0.001;
    Duration = max(time(:));
    % Create an earthquake matrix that will be used in Simulink
    eq = [time; accel];
    % Save for Simulation
    save eq eq

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %         Locator Matrix         %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    LocatorMatrix = [-1  1  0 ;
                     0 -1  1 ;
                     0  0 -1 ] ;

    % Max Accel that instrudment can read
    sensor_min_acc = -20 ;
    sensor_max_acc =  20 ;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %   Build ABCD Matricies for 3DOF Benchmark  %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % New SysID with correct masses
    k1 = 1172818.1282418387;
    k2 = 1750017.1838405035;
    k3 = 1998294.2058354884;
    zeta = 0.00525031239033;

    % ABCD
    m1 = 5800;
    m2 = 5800;
    m3 = 6840;
    MassVec = [m1,m2,m3];
    M = diag(MassVec);

    K = [k1+k2 -k2     0;
        -k2     k2+k3 -k3;
        0     -k3     k3];
    [C] = Calc_Damping(M,K,zeta);
    nDOF = 3;

    %  Build state space matrices for the building structure.
    A = [zeros(nDOF) eye(nDOF); -inv(M)*K -inv(M)*C];
    B = [zeros(nDOF); inv(M)];
    C = [eye(nDOF) zeros(nDOF); zeros(nDOF) eye(nDOF); -inv(M)*K -inv(M)*C];
    D = [zeros(nDOF); zeros(nDOF); inv(M)];
    % Unit: Newtons

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %         Run Simulation         %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % Run Passive Off
    disp('Passive Off')
    passVolt = 0;
    select = 2;
    % Run Uncontrolled Bldg
    sim('simulateBldg')
    load response
    poff = response;
    time = poff(1,:);
    poffd1 = poff(5,:)*1000;
    poffd2 = poff(6,:)*1000;
    poffd3 = poff(7,:)*1000;
    poffv1 = poff(8,:);
    poffv2 = poff(9,:);
    poffv3 = poff(10,:);
    poffa1 = poff(11,:);
    poffa2 = poff(12,:);
    poffa3 = poff(13,:);
```

```
    % Run Passive On
    disp('Passive On')
    passVolt = max_volt;
    select = 2;
    % Run Uncontrolled Bldg
    sim('simulateBldg')
    load response
    pon = response;
    time = pon(1,:);
    pond1 = pon(5,:)*1000;
    pond2 = pon(6,:)*1000;
    pond3 = pon(7,:)*1000;
    ponv1 = pon(8,:);
    ponv2 = pon(9,:);
    ponv3 = pon(10,:);
    pona1 = pon(11,:);
    pona2 = pon(12,:);
    pona3 = pon(13,:);

    % Run FLC controlled
    disp('FLC Controlled')
    select = 1;
    sim('simulateBldg')
    load response
    con = response;
    dcon1 = con(5,:)*1000;
    dcon2 = con(6,:)*1000;
    dcon3 = con(7,:)*1000;
    vcon1 = con(8,:);
    vcon2 = con(9,:);
    vcon3 = con(10,:);
    acon1 = con(11,:);
    acon2 = con(12,:);
    acon3 = con(13,:);
    mrforce1 = con(15,:);
    mrvolt1 = con(18,:);

    % Build tables to compare data
    poffdmax = max([max(abs(poffd1)),max(abs(poffd2-poffd1)),max(abs(poffd3-poffd2))]);
    pondmax = max([max(abs(pond1)),max(abs(pond2-pond1)),max(abs(pond3-pond2))]);
    flcdmax = max([max(abs(dcon1)),max(abs(dcon2-dcon1)),max(abs(dcon3-dcon2))]);

    poffamax = max([max(abs(poffa1)),max(abs(poffa2)),max(abs(poffa3))]);
    ponamax = max([max(abs(pona1)),max(abs(pona2)),max(abs(pona3))]);
    flcamax = max([max(abs(acon1)),max(abs(acon2)),max(abs(acon3))]);

    poffdrms = max([rms(poffd1),rms(poffd2-poffd1),rms(poffd3-poffd2)]);
    pondrms = max([rms(pond1),rms(pond2-pond1),rms(pond3-pond2)]);
    flcdrms = max([rms(dcon1),rms(dcon2-dcon1),rms(dcon3-dcon2)]);

    poffarms = max([rms(poffa1),rms(poffa2),rms(poffa3)]);
    ponarms = max([rms(pona1),rms(pona2),rms(pona3)]);
    flcarms = max([rms(acon1),rms(acon2),rms(acon3)]);

    pon = [pondmax/poffdmax,ponamax/poffamax,pondrms/poffdrms,ponarms/poffarms];
    flc = [flcdmax/poffdmax,flcamax/poffamax,flcdrms/poffdrms,flcarms/poffarms];

    JResults{i,1} = [pon;flc];
    Jdisp = cell2mat(JResults)
    disp('<<<<<<<<<<<<------------>>>>>>>>>>>>>')
end
disp('J Results')
JResults = cell2mat(JResults)
disp('done')
```

VITA

David Adam Shook received his Bachelor of Science in civil engineering at Texas A&M University in May 2005. He then enrolled at Texas A&M University under the Master of Science program in the Zachry Department of Civil Engineering in June 2005 and received his Master of Science in December 2006. During his study David Shook worked under the guidance of Dr. Paul N. Roschke. His research involved optimization of fuzzy logic controllers that operate magnetorheological dampers. During his education David Shook received a research assistantship from Dr. Roschke in 2005. David Shook can be contacted at:

Contact Address:
9208 North County Road
Frisco, TX 75034
E-mail: olrock05@yahoo.com