A HYBRID METHOD FOR IDENTIFYING AND ESTIMATING KEY DYNAMIC PARAMETERS FOR EXCITERS, PSS AND GOVERNORS BASED ON EVENT-RECORDED MEASUREMENTS

by

YUNZHI CHENG

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2009

ACKNOWLEDGEMENTS

I would like to express deep gratitude to my supervisor professor, Dr. Wei-Jen Lee for his guidance, encouragement, patience, and support throughout the development of this dissertation and my academic program. I would also like to take this opportunity to convey my sincere gratitude to Dr. Lee for extending his support and guidance in a period during which certain circumstances almost led to the end of my dream to pursue a doctoral degree. I would be forever indebted to him for his unwavering moral support and sound advice during the period.

I wish to thank Dr. Raymond R. Shoults, Dr. William E. Dillon, Dr. Rasool Kenarangui, and Dr. Babak Fahimi for their instruction, serving on my dissertation committee, valuable suggestions and review. A special note of appreciation is also extended to Dr. Shun-Hsien Huang who is a planning engineer at the Electric Reliability Council of Texas (ERCOT) for his help on my research.

I also want to express appreciation to all members of Energy Systems Research Center (ESRC), the University of Texas at Arlington, for all their valuable assistances, discussions, and enjoyable association. Apart from that, I would like to thank Dr. Mandhir Singh Sahni and colleagues at PwrSolutions for their support and comments of this dissertation.

Finally, I would dedicate my dissertation to my parents, parents-in-law and my wife Weiping Xiao for serving as my continued support system and a constant source of inspiration for the tenure of my doctoral degree. I am very grateful to them for their love, encouragement and continuous support.

November 11, 2009

ABSTRACT

A HYBRID METHOD FOR IDENTIFYING AND ESTIMATING KEY DYNAMIC PARAMETERS FOR EXCITERS, PSS AND GOVERNORS BASED ON EVENT-RECORDED MEASUREMENTS

Yunzhi Cheng, PhD

The University of Texas at Arlington, 2009

Supervising Professor: Wei-Jen Lee

Following the power market deregulation, power systems have become more complex and are found to be consistently operating closer to their stability limits. Power system dynamic modeling and studies which provide significant insight into the dynamic characteristics of the system are bound to play increasingly critical roles. The dynamic simulation results are highly dependent on certain key parameters that govern the dynamics of the power system such as governors and/or exciters in the case of generation facilities. However, the dynamic parameters in the system database maintained by the Independent System Operator (ISO) are not accurate due to numerous reasons ranging from data submissions not corresponding to "as-built facilities" to data not being updated to reflect changes at the facility.

Such inconsistencies in the dynamic models utilized to represent actual system facilities have led to tremendous research in the field of dynamic parameter estimation. Numerous algorithms have been proposed for dynamic parameter estimation. The conventional gradient-based optimization approach suffers from an obvious and inherent dependency on the initial conditions and is found to have convergence problems when starting with a poor initial guess.

On the other hand, some inherently initial-value independent intelligent methods suffer from tremendous computation burden. This dissertation proposes a hybrid two-step method to achieve the accurate dynamic parameters in a balanced manner by making an optimal trade-off between convergence and computation speed. The concept of Particle Swarm Optimization (PSO) is employed to find an approximate solution at the first step, followed by a sensitivity analysis is run to achieve an accurate solution starting with the approximate solution obtained in the first step.

This dissertation describes how various categories are set up for the dynamic parameters and identifies the key parameters for parameter estimation to decrease the complexity of the problem and computation burden. While the approach documented in this dissertation is generic in terms of applicability to dynamic parameter estimation, the generator dynamic parameters have been utilized to illustrate the efficiency of the approach. All exciter and governor models in the Electrical Reliability Council of Texas (ERCOT) system are pre-scanned to identify the key parameters using the PSS/E response test.

The proposed hybrid method shows the validity and distinct advantages in the assumed test case. The exciter and governor parameters are successfully estimated using the proposed hybrid method. Reasonably accurate values can be achieved under some level of noise according to uncertainty analysis. Multi-core computation is utilized to dramatically decrease the computation burden.

The proposed hybrid method also successfully tunes the dynamic parameters of exciter and power system stabilizer (PSS) in a power plant to drive the trend of simulation results to match the recording information on file following a generator trip in ERCOT system.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background of Current Power System Deregulation and Problems

In 1882, Thomas Edison built up the first complete electrical power system with a generator, cable, and loads. It was a small and local system. Since then, electrical power system became larger through the network interconnection due to the development of higher AC/DC voltage. The modern Electrical power system is one of the largest and the most complicated man-made systems with billions of components [1]. Thousands of generators feed into a huge interconnected transmission and distribution network to serve the energy demand of billions of people. Maintaining the power system stability which deals with the ability of a system to withstand a wide variety of disturbances has been a challenge for power system researchers and engineers [2].

In recent years, the United States electric utility industry entered a phase of restructuring and deregulation. The traditional vertically integrated electric utility structure has been replaced by a horizontal structure with unbundled generation, transmission, and distribution companies in half of the states [3]. Deregulation is expected to encourage competition and decrease electricity price. However, deregulation results in a more intensive use of the transmission network which pushes the electrical power system closer to the stability limit. Thus, power system stability problem is more important and serious than ever.

1.2 Power System Parameter Estimation

Power systems have become more complex and closer to the stability limit since the deregulation of electricity markets. Dynamic modeling and studies which reveal the dynamic characteristic of a power system play more critical roles under these circumstances. The dynamic simulation results are heavily dependent on the parameters of the system components such as generators, exciters, governors, and loads. Unfortunately, many dynamic parameters in the system are not accurate because of the following reasons:

1) The dynamic parameters available to the Independent System Operator (ISO) are usually from generator owners. They collected the parameters from the manufacture data and/or on-site test before the first operation. Some parameters may drift over a long period of operation. In addition, it is very difficult to ask generator owners to check and update the parameters by performing frequent on-site tests.

2) Some parameters obtained from the manufacture data are the range of values and they can be changed during practical operation. The mean value sometimes is sent to the ISO from the generator owners.

3) Some parameters are not available from manufacture data or on-site. The ISO has to replace them by some generic value or using the same value from the other generation with the same/similar dynamic model.

Here is an example in the ERCOT system. As shown in Figure 1.1, CEC, DPEC, and DOW are three generators located in different places in the ERCOT system with ratings of 285MVA, 305MVA, and 227.5MVA, respectively. They use the same type of exciter model “ESAC1” which is a popular model. However, the exciter parameters of the three exciters are exactly the same according to the ERCOT dynamic data as shown in Figure 1.2. The accuracy of the parameters is highly suspect.

Thus, it is common to see some mismatches between simulation results from current dynamic stability analysis data and the real-time event recording information because of inaccurate dynamic parameters. In some cases, it will lead to conservative decision of the transfer limits such as available transfer capacity (ATC) and total transfer capacity (TTC) in the system, and then result in additional congestion in the power market and reduced asset utilization of the transmission network. On the other hand, the inaccurate simulation results may also lead to overestimation of the transfer capacity and mislead the operators of the system.

The impact of such over-estimations may vary from minor emergency notifications to cascading blackouts in the worse case scenario.

A large blackout of about 30GW loss of load occurred in the Western Systems Coordinating Council (WSCC) system on August 10, 1996 [4]. The dynamic simulation results based on the standard WSCC dynamic database illustrate no stability problem in the system which is in direct contradiction to the disturbance recordings as shown in Figure 1.3 and Figure 1.4. After that, WSCC established the Governor Modeling Task Force and the Load Modeling Task Force to address this issue. They modified some dynamic models such as exciters, governors and loads. A very good agreement was achieved between simulation results using the modified models and recordings, as shown in Figure 1.5 and Figure 1.6.

Eight years later, on June 14, 2004, a major disturbance resulted in approximately 1000MW loss of load in the Western Electricity Coordinating Council (WECC, the successor of WSCC) [5]. As usual, WECC simulated the dynamic events and compared the simulation results with the recorded data. The initial simulation successfully reproduced the system frequency performance as shown in Figure 1.7. However, the initial simulation failed to replicate the voltage profile in some areas. The actual voltage profile at the Palo Verde 500kV bus was much worse than that obtained via the simulation results as shown in Figure 1.8. As a result, WECC had to re-start the model validation work again. The work process lasted for more than one year and most of the work was manually and jointly performed by several utilities in WECC. Finally, the simulation results turned out to be closer to the recordings after some parameters of the Palo Verde generator were fine-tuned as shown in Figure 1.9.

The lessons learned in the WECC model validation effort can be summarized as follows:

1) Maintenance of the dynamic models and parameters is a long-term and on-going effort. The planners and operators of the system need to study and update them on a continuous basis.

2) Maintenance of the dynamic models and parameters is a local problem. The mismatch between the simulation and event recording usually can be fixed through the models and parameters of the local devices.

3) Since manual adjustment is time consuming and it is difficult to obtain an optimal solution, it is necessary to develop an automated procedure for the parameter estimation after disturbances.

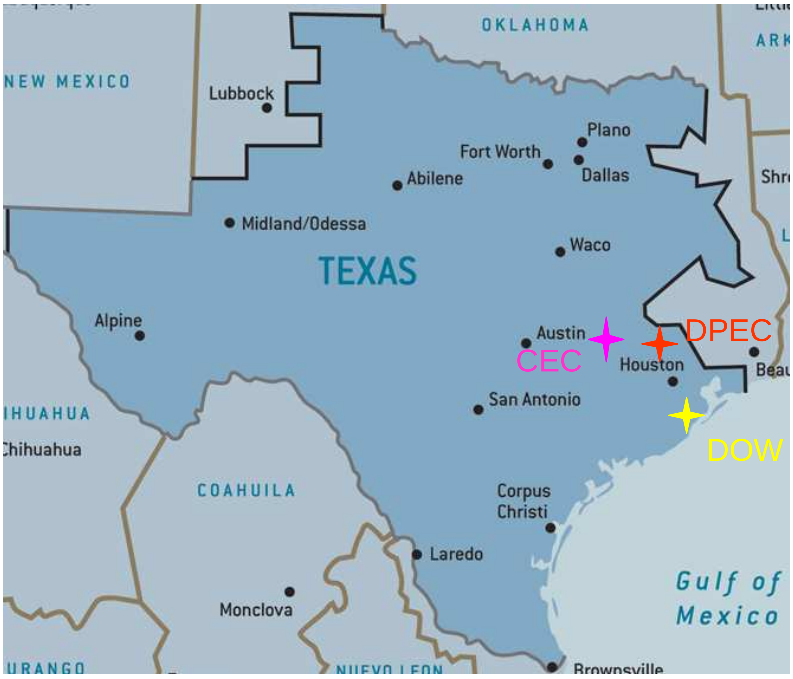


Figure 1.1 Locations of the Three Generators in ERCOT System

Model Name	Model Identifier	Type	TR (sec)	TB (sec)	TC (sec)	KA	TA (sec)	VA MAX	VA MIN	TE > 0 (sec)	KF
ESAC1A	49886 [DOW84000 18.000] 8	Stnd	0.0000	0.0000	0.0000	1642.0000	0.0120	10.5000	0.0000	0.9500	0.0320
ESAC1A	48645 [DPEC ST 18.000] 5	Stnd	0.0000	0.0000	0.0000	1642.0000	0.0120	10.5000	0.0000	0.9500	0.0320
ESAC1A	48594 [CEC ST-1 18.000] 4	Stnd	0.0000	0.0000	0.0000	1642.0000	0.0120	10.5000	0.0000	0.9500	0.0320

Model Name	Model Identifier	Type	TF > 0 (sec)	KC	KD	KE	E1	SE(E1)	E2	SE(E2)	VR MAX	VR MIN
ESAC1A	49886 [DOW84000 18.000] 8	Stnd	1.0000	0.2560	1.2800	1.0000	6.9500	0.0900	3.7100	0.0500	10.5000	0.0000
ESAC1A	48645 [DPEC ST 18.000] 5	Stnd	1.0000	0.2560	1.2800	1.0000	6.9500	0.0900	3.7100	0.0500	10.5000	0.0000
ESAC1A	48594 [CEC ST-1 18.000] 4	Stnd	1.0000	0.2560	1.2800	1.0000	6.9500	0.0900	3.7100	0.0500	10.5000	0.0000

Figure 1.2 Exciter Parameters of the Three Generators

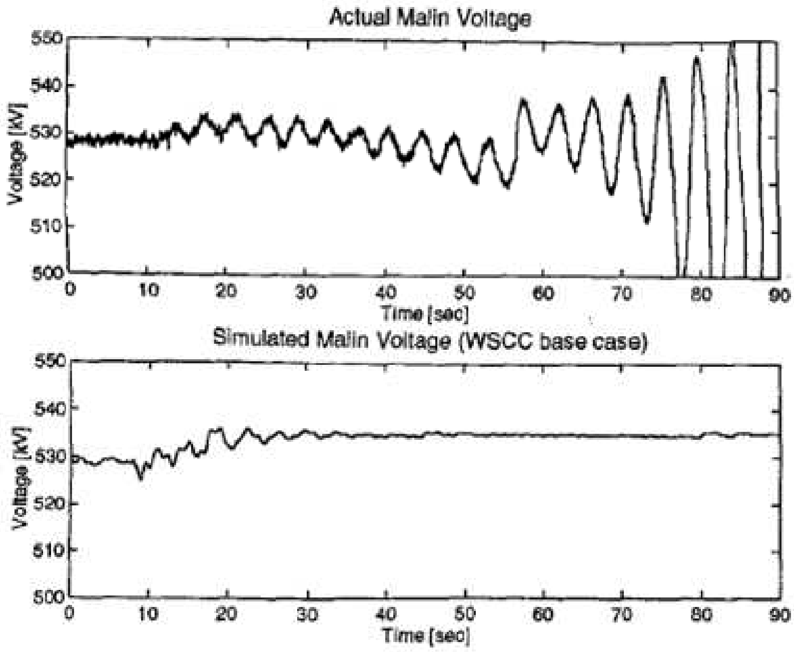


Figure 1.3 Voltage Profile - Recordings and the Simulation Using the standard Models (WSCC, 1996)

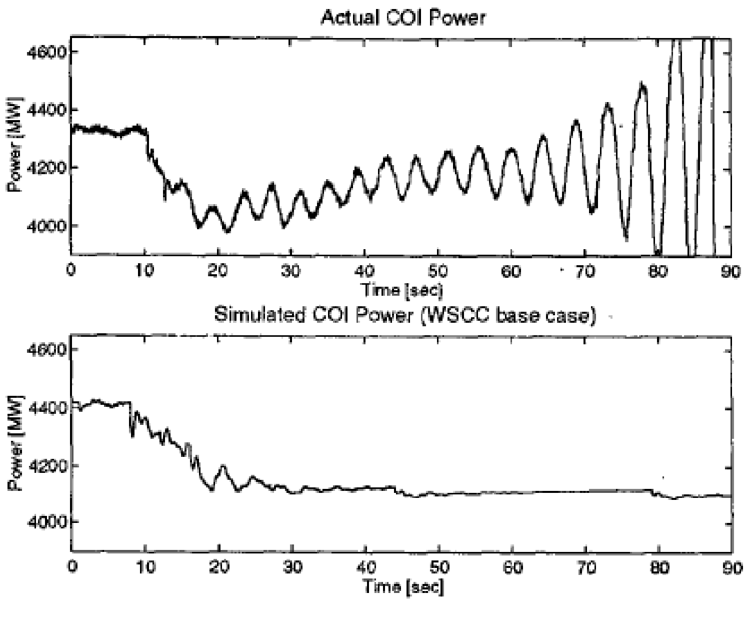


Figure 1.4 Active Power Profile - Recordings and the Simulation Using the standard Models (WSCC, 1996)

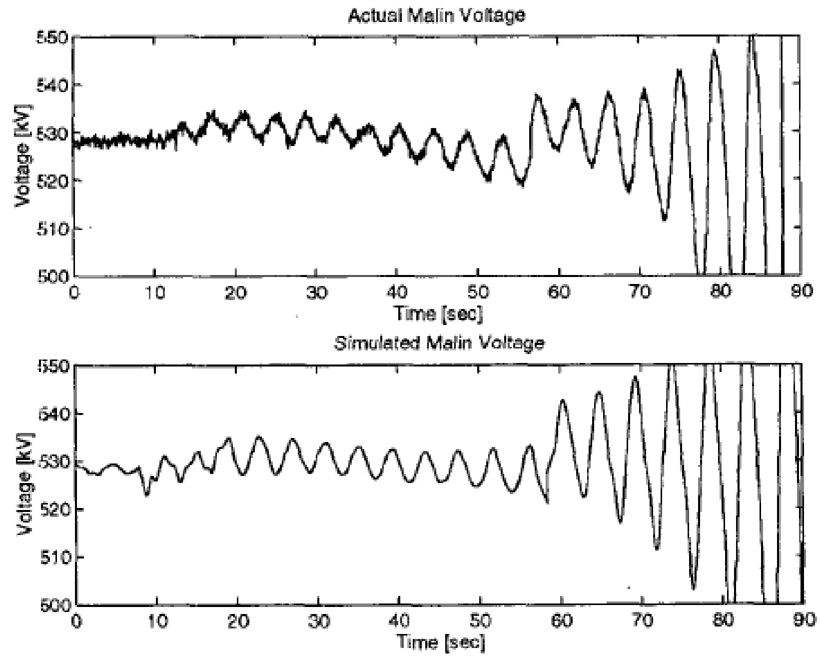


Figure 1.5 Voltage Profile - Recordings and the Simulation Using the Modified Models (WSCC, 1996)

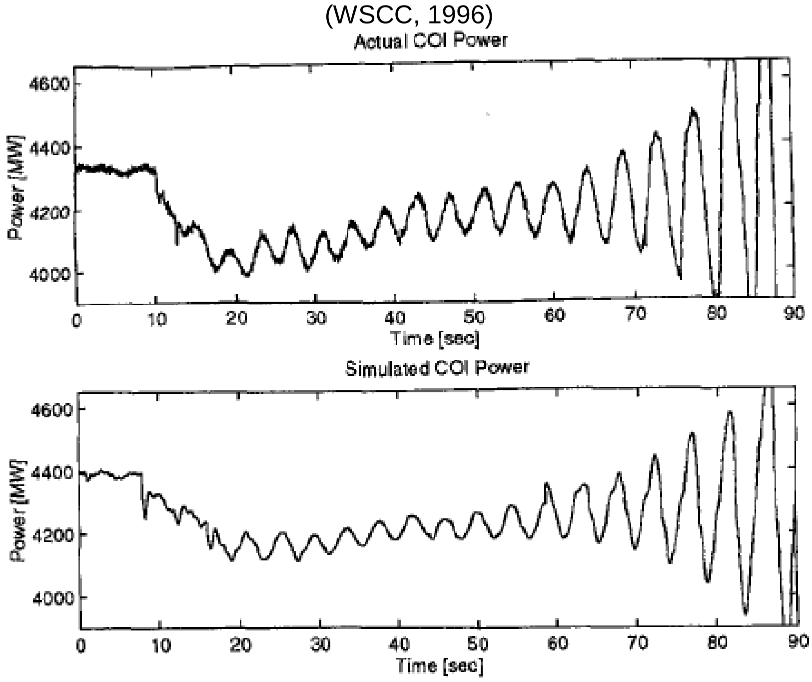


Figure 1.6 Active Power Profile - Recordings and the Simulation Using the Modified Models (WSCC, 1996)

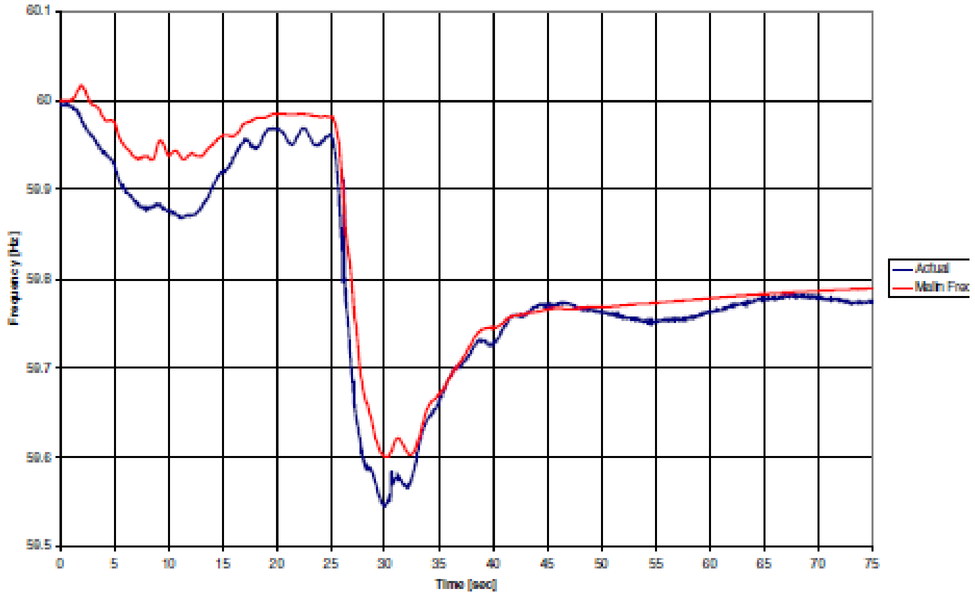


Figure 1.7 Frequency Profile - Recordings and the Simulation Using the existing Models (WECC, 2004)

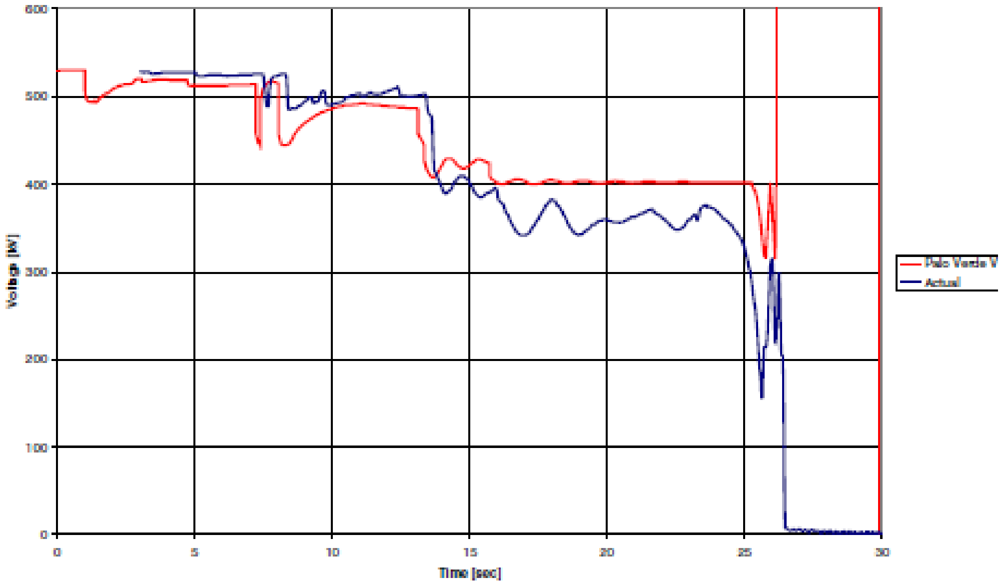


Figure 1.8 Voltage Profile - Recordings and the Simulation Using the existing Models (WECC, 2004)

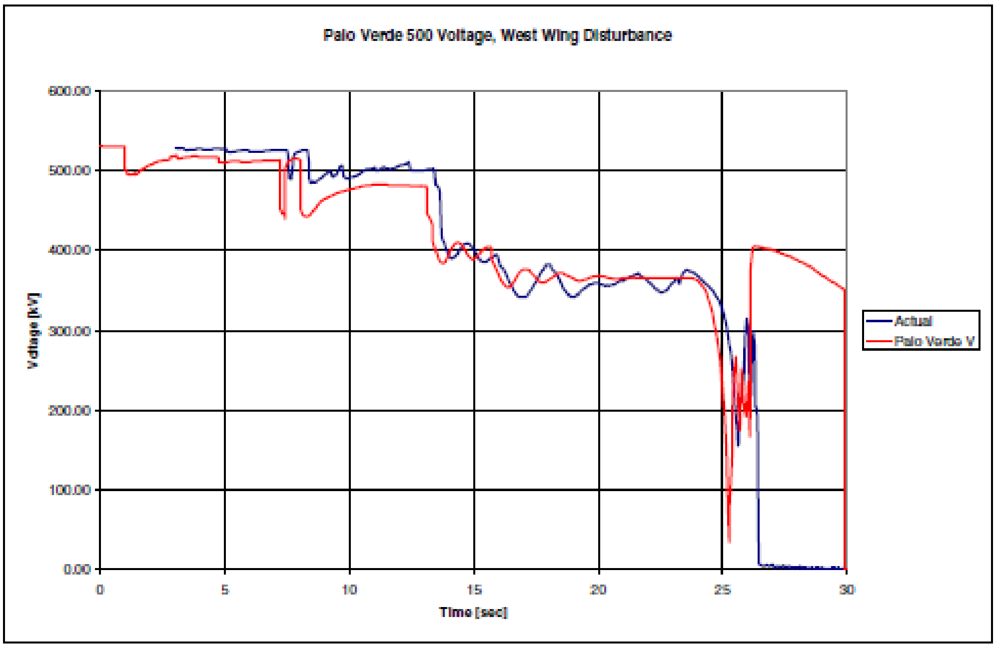


Figure 1.9 Voltage Profile - Recordings and the Simulation Using the modified Models (WECC, 2004)

1.3 The Proposed Method

Many approaches and techniques have been explored to achieve the objective of power system dynamic parameter estimation proposed in this dissertation. Some conventional gradient-based optimization methods were utilized to search for the accurate dynamic parameters. The main disadvantage of such a method is the property of initial-value dependency. A poor initial value may cause slow convergence or even divergence. Some intelligent methods based on population were explored to solve this problem since they are inherently initial-value independent. However, a substantial amount of dynamic simulation or historical information is needed for these kinds of methods.

To obtain a balance between the two approaches mentioned above, a hybrid method combining a new intelligent optimization method, particle swarm optimization (PSO), and conventional optimization method, sensitivity analysis (SA), is proposed in this dissertation.

The proposed hybrid method uses the recorded data from the on-line measurements following a large disturbance in the system as the input. As the first step the particle swarm optimization method is applied initially to find an approximate solution; following that a sensitivity analysis is executed using the approximate solution obtained through the first step as the initial condition to achieve the accurate parameters as the second step.

1.4 Assumptions and Contributions

1.4.1 Assumptions

The proposed method focuses on parameter estimation for the dynamic models used in the power system simulation. The dynamic model itself is assumed to be correct. This assumption is reasonable since the model was developed by the manufacturer and tested during the production process and at the on-site test-stage prior to the practical operation.

Some electrical quantities are very sensitive to the dynamic parameters. For example, E_{fd} (exciter field voltage) is very sensitive to exciter parameters and can be used to identify these parameters. However, it is not practical to capture the value of E_{fd} during routine operation. In addition, it is usually hard for the ISO to request performance data internal to the generators after deregulation. Therefore, only the on-line measurement data on the grid side and at generator-grid interface are utilized for dynamic parameter estimation in the proposed method.

1.4.2 Contributions

The proposed hybrid method provides the right balance and trade-off between convergence and computational speed. It is not dependent on the initial guess and exhibits superior performance in terms of simulation time. The improvement of solution accuracy and computation time by the PSO algorithm utilized in the proposed method has been identified by many researchers in power system optimization [6].

The proposed hybrid method is a model independent method. This dissertation focuses on dynamic parameter estimation of exciters, PSS, and governors. However, the method can easily be extended to dynamic parameter estimation of other device models in power systems such as dynamic load models. Commercially available power system simulation software, PSS/E, is utilized as the simulation engine in the proposed hybrid method so that the proposed method is applicable to and works well on large-scale systems.

There are more than ten parameters in most of models of exciters, power system stabilizers and governors. This dissertation analyzes and categorizes them into different groups. Those parameters which are adjustable and have a great impact on the dynamic simulation are defined as key parameters. Only the key parameters are needed to be incorporated in the estimation problem which decreases the complexity of the estimation problem and computation burden. All exciter and governor models used in the ERCOT system are pre-scanned to identify the key parameters using the PSS/E response test. It will benefit the model/parameter estimation work in the future.

1.5 Synopsis of Chapters

The organizational structure associated with this dissertation is as follows:

Chapter 1 introduces the general background of the power market deregulation, power system dynamic parameter estimation, illustrates the importance, motivation and objective of this dissertation.

Chapter 2 reviews the historical research approaches and techniques, discusses the conventional and intelligent optimization methods used in dynamic parameter estimation in elaborate detail.

Chapter 3 describes the proposed hybrid method in detail. This chapter focuses on sensitivity analysis and particle swarm optimization since they are the most important components of the proposed hybrid method.

Chapter 4 presents the process associated with key parameters Identification. The key parameters can be identified by sensitivity analysis using PSS/E response test and dynamic event simulation.

Chapter 5 uses an assumed test case in the ERCOT system to demonstrate the validity and the distinct advantages of the proposed hybrid method. The parameters of “EXAC1” exciter and “IEESGO” governor models are successfully estimated using the proposed hybrid method. A reasonably accurate range can be achieved in the presence of noise according to the uncertainties analysis. Multi-core computation is utilized with the proposed method to drastically decrease the computation time.

Chapter 6 applies the proposed hybrid method with the field recording information in ERCOT. The parameters of “EXAC1” exciter and ”PSS2A” power system stabilizer (PSS) are tuned using the proposed method according to the field recording information at the power plant. There is good agreement between the recording information and the simulation results with modified parameters obtained from the proposed method.

Chapter 7 presents the conclusions/recommendations drawn from the research associated with this dissertation and discusses the opportunity for further research.

CHAPTER 2

LITERATURE REVIEW

2.1 Parameter Estimation Approaches

Field test [7-8] and on-line measurements [9-12] are two typical approaches for power system dynamic parameter estimation.

Field test is a good means to gain direct insight for dynamic parameters of the power system devices. It is usually preformed before the commissioning of the new device to establish the initial models and parameters. However, parameters may drift after the continued operation or maintenance. Frequent field tests would be either impractical or even impossible due to the potential damage or high costs.

Another alternative approach for dynamic parameter estimation is on-line measurements during real-time operation of the devices. The dynamic behavior of the system can be captured through the on-line measurements after dynamic events such as a system disturbance. The Supervisory Control and Data Acquisition (SCADA) and Energy Management System (EMS) present the steady-state pre-event scenario for parameter estimation. The protection equipment and some newly developed Intelligent Electri c Devices (IED) such as Phasor Measurement Units (PMU) can provide the required information for parameter estimation with better resolution and time synchronization among measured dynamic data.

This dissertation focuses on the use of on-line measurements from the real-time operations following a large disturbance in the system. It provides the updated condition of the devices and avoids some unintended consequences caused by field tests.

2.2 Curve-fitting Technology for Parameter Estimation

Curve-fitting technology is the most popular method for dynamic parameter estimation. The curve from simulation results is adjusted to the desired curve (recorded data) by changing the parameters of the system model. It can be formulated as a nonlinear optimization problem as below:

$$Min \qquad J_{x \in P}(x) = (z - h(x))^T W (z - h(x)) \tag{2.1}$$

Where:

x is the parameter vector.

P is the parameter space.

z is the measurement vector corresponding to the recorded sample set of measured signals.

h(x) is the vector corresponding to the simulated set of the same signal as z.

W is the weight matrix. Usually the identity matrix is used as weight matrix.

There are many methods that can be utilized for the nonlinear optimization problem. Generally, they can be classified into two sets: Sensitivity Analysis and Intelligent Search.

2.2.1 Conventional Optimization Methods and Applications

Sensitivity Analysis (SA) is the most popular and conventional principle in solving nonlinear curve-fitting problems. SA minimizes the objective function that is the sum of the square of weighted residuals as in (2.1).

There are two kinds of SA: Gradient Search and Sensitivity Matrix.

2.2.1.1 Gradient Search

Gradient search is a simple optimization method based on the first derivative of the optimization objective function. Starting from an initial guess, the algorithm performs a two-step iteration as follow:

1. Evaluate the gradient $\nabla J(x^{(k)})$.

The finite-difference approximation is used when there is no explicit gradient expression available. The gradient vector is calculated by the formula:

$$\nabla J_i(x^{(k)}) = \frac{J(x^{(k)} + \Delta x_i) - J(x^{(k)})}{\Delta x_i} \tag{2.2}$$

Where:

Δx_i is a small disturbance (1%) for i-th parameter.

2. Determine the correction for each parameter by the formula (2.3) and step change by the formula (2.4)

$$\Delta x_i^{(k)} = -\frac{J(x^{(k)})}{\nabla J_i(x^{(k)})} \tag{2.3}$$

$$x^{(k+1)} = x^{(k)} + \alpha * \Delta x^{(k)} \tag{2.4}$$

Factor α in (2.4) is a damping ratio used to limit the step change. Usually the step change should not surpass 10% of $x^{(k)}$.

The flowchart of the gradient search algorithm for the non-linear curve-fitting problem is shown in Figure 2.1.

2.2.1.2. Sensitivity Matrix

The Weighted least squares (WLS) principle is utilized in the Sensitivity Matrix method. WLS is the most popular method for power system state estimation [13].

The optimality conditions of the first-order of the objective function at the minimum point can be expressed as:

$$g(x) = \frac{\partial J(x)}{\partial x} = -H^T(x)W(z - h(x)) = 0 \tag{2.5}$$

Where:

$H(x) = \left[\frac{\partial h(x)}{\partial x}\right]$ is called the Jacobi matrix, or Sensitivity matrix.

The nonlinear equation of (2.5) can be expanded into its Taylor series around the state vector $x^{(k)}$:

$$g(x) = g(x^{(k)}) + \frac{\partial g(x^{(k)})}{\partial x}(x - x^{(k)}) + \frac{1}{2!}\frac{\partial g^2(x^{(k)})}{\partial^2 x}(x - x^{(k)})^2$$

$$+ \frac{1}{3!}\frac{\partial g^3(x^{(k)})}{\partial^3 x}(x - x^{(k)})^3 + \cdots = 0 \qquad (2.6)$$

While neglecting the higher order of Taylor series, the nonlinear equation of (2.6) can be simplified as follows:

$$g(x) = g(x^{(k)}) + G(x^{(k)})(x - x^{(k)}) = 0 \qquad (2.7)$$

Where:

$G(x) = \frac{\partial g(x)}{\partial x} = H^T(x)WH(x)$ is called the gain matrix.

The optimized parameters can be explored by an iterative solution scheme known as the Gauss-Newton iteration which neglects the higher order of Taylor series as follows:

$$x^{(k+1)} = x^{(k)} + G(x^{(k)})^{-1} H^T(x^{(k)}) W (z - h(x^{(k)})) \qquad (2.8)$$

The sensitivity matrix H(x) can be assessed using an explicit expression in power system state estimation. However, there is no explicit expression for H(x) in most of curve fitting problems. In other words, H(x) may have to be computed by using a finite-difference approximation:

$$H_{i_j}(x^k) = \frac{h_i(x^k + \Delta x_j) - h_i(x^k)}{\Delta x_j} \qquad (2.9)$$

Where:

$h_i(x^k)$ is the i-th simulated sample at k-th iteration.

Δx_j is a small disturbance (1%) for j-th parameter.

The flowchart of the sensitivity matrix algorithm for non-linear curve-fitting problem is shown in Figure 2.2.

2.2.1.3. Performance of Sensitivity Analysis

The performance of sensitivity analysis is decided by two factors:

1) Initial guess

2) Complexity of the optimization problem

Sensitivity analysis starts the iteration from an initial guess, which makes the final solution highly dependent on the quality of the initial guess. It can quickly reach the final solution given a sufficiently accurate initial approximation. Without a good initial guess, it may require a tremendous number of iterations to reach the final solution, or be trapped into a local minimum, or even diverge during the iteration process because of a poor initial approximation.

The complexity of the optimization problem has also a great impact on the convergence behavior of the sensitivity analysis. The more complex is the optimization problem, the harder it becomes to reach convergence. This weakness limits the application of sensitivity analysis in power system optimization since many power system optimization problems are multi-dimensional and highly nonlinear.

2.2.1.4. Applications

In [8-12], sensitivity analysis (SA) methods (sensitivity matrix or gradient search) were utilized to find the best-fit parameters for the recorded curve during field tests or after a large perturbation.

The author proposed a framework based on gradient search to explore the dynamic parameters of exciters, power system stabilizers (PSS) and governors through the filed tests in [9]. AVR step response test and 0pf load (100% reactive power load) rejection test were performed for exciters and PSS parameter estimation. Partial load rejection test was performed for governor parameter estimation. For this method the convergence speed becomes very slow for only four parameters and poor initial condition, and it may not converge according to the summary by the author.

The trajectory sensitivity (gradient search) based estimation of synchronous generator and exciter system parameters was performed in [10]. The generator voltage and current are used to verify the generator reactance during single-phase fault event. The exciter field voltage (E_{fd}) is monitored for exciter parameter estimation using a line switch event. However, the method in this paper has three disadvantages: a good initial guess is needed; E_{fd} is not captured during routine power system operation; only single generator, single line to infinite bus power system is used.

The sensitivity matrix method was applied for exciter parameter estimation using field test results in [11]. The generator terminal voltage, exciter filed voltage (E_{fd}) and current (I_{fd}) were recorded by a remote digital recorder. A preliminary solution gained from the authors' experience is used as the initial guess to speed up the convergence.

The performance of these conventional methods on the parameter estimation is highly dependent upon the initial guess which is a common weakness for the above mentioned methods. A poor initial guess may result in a slow convergence or even divergence.

2.2.2 Intelligent Optimization Methods

To overcome the limitation of the initial guess in the conventional optimization method, many intelligent methods such as genetic algorithm and neural network have been explored for dynamic parameter estimation [14-16].

In [14], the authors employed the Hopfield Neural Network (HNN) for the parameter estimation of a static excitation system. The method effectively relieved the reliance on the initial guess. The neuron weighted matrix (W) and the output state (V) in the HNN algorithm were calculated using the information of state variables and inputs. However, those state variables in many cases are difficult to capture.

In [15], the genetic algorithm (GA) was proposed for dynamic parameter estimation when the gradient-based methods failed because of the inaccurate gradient. However, GA suffers from the need of a huge number of function evaluations since it is a population-based algorithm. The objective function in dynamic parameter estimation usually is evaluated by the dynamic simulation (stability program) package which is very time-consuming for a large power system.

In [16], the authors proposed an improved genetic algorithm based on differential evolution (GA-DE) to estimate the parameters of a static excitation system model of a Brazilian hydro power plant during field tests. The test input signals which contain many harmonic components such as white-noise, pseudorandom binary (PRBS), rectangular and step signals are added to field tests. The signal energy is limited to avoid undesirable operation of the generator during the filed tests. So it was observed that the highly noise-corrupted measurements will affect the behavior of the GA based identifier according to the conclusions of the paper.

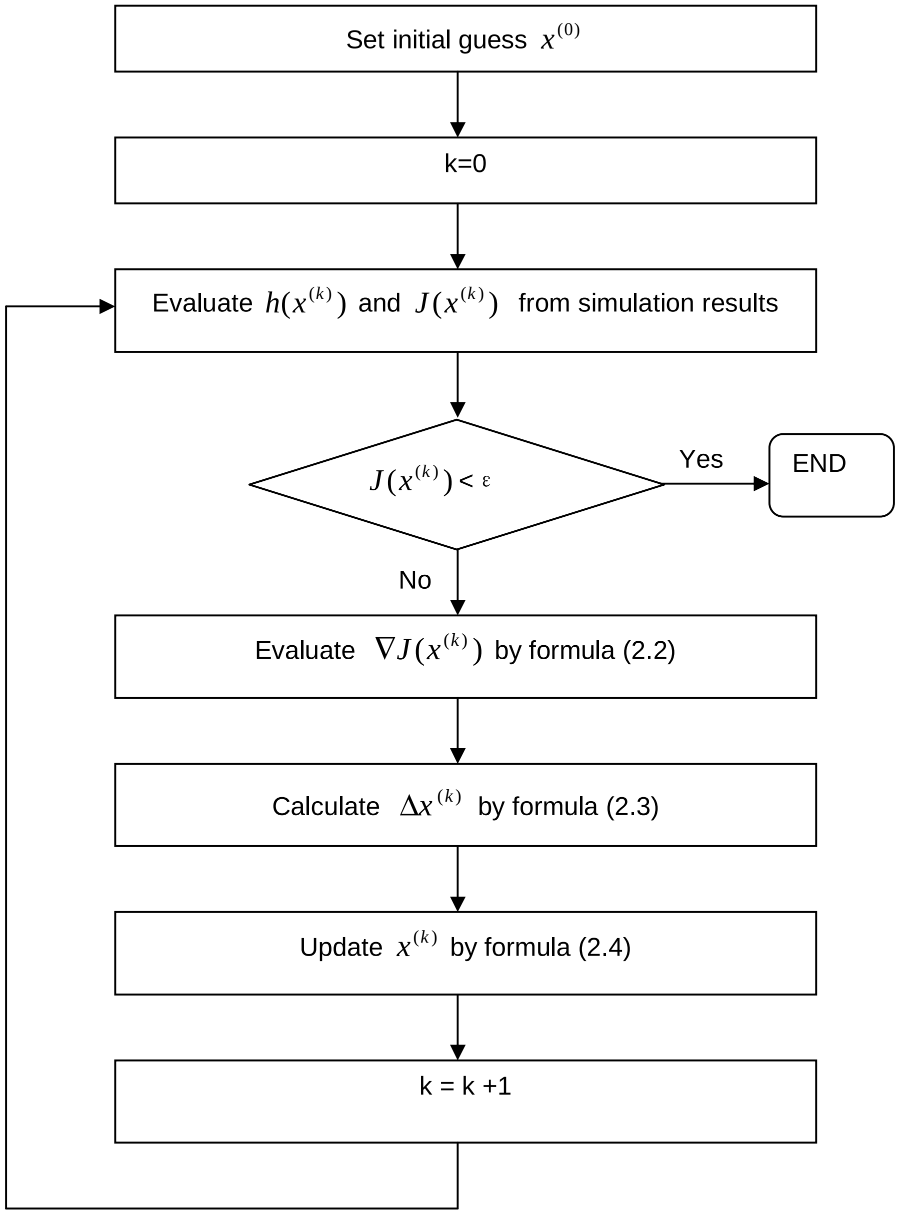


Figure 2.1 Flowchart of the Gradient Search Algorithm for Non-linear Curve-fitting Problem

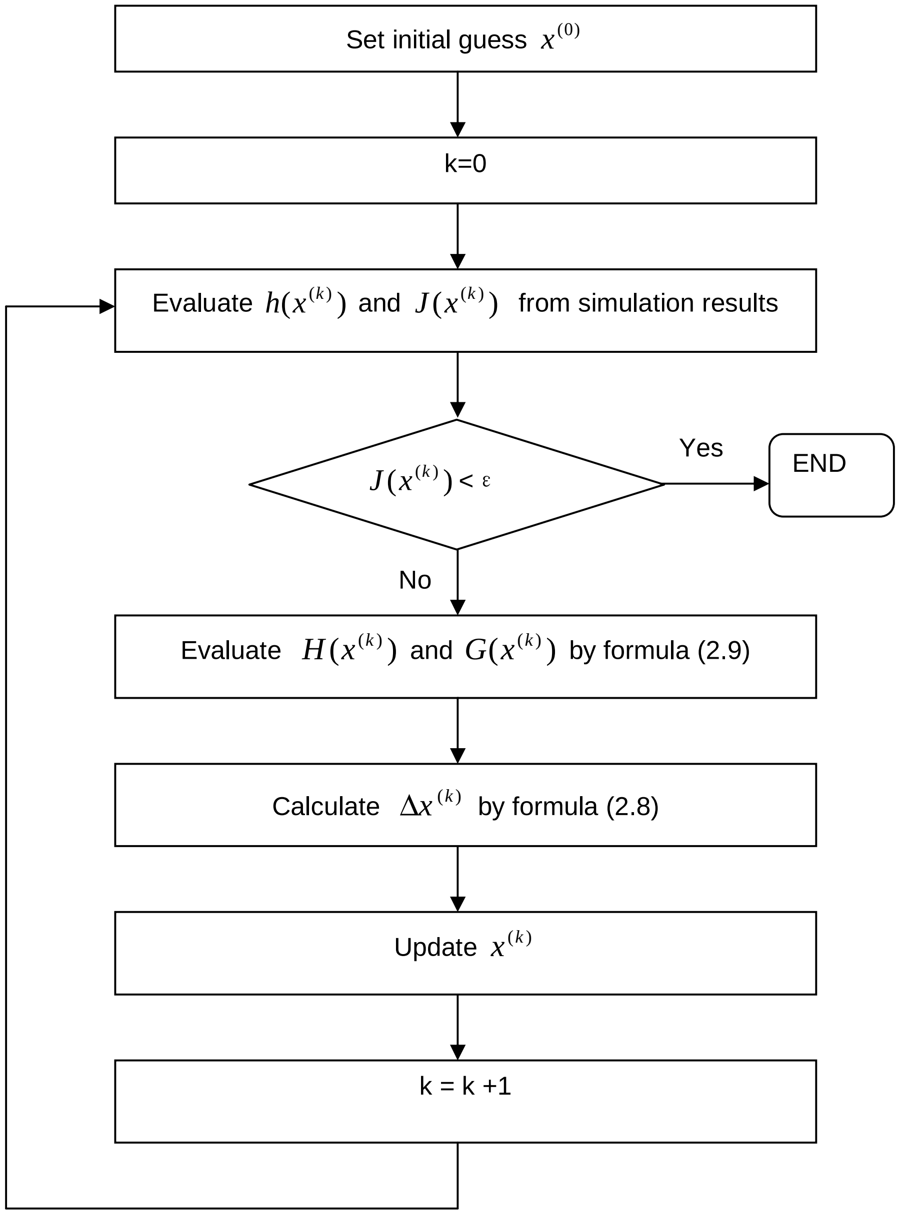


Figure 2.2 Flowchart of the Sensitivity Matrix Algorithm for Non-linear Curve-fitting Problem

2.3 Other Approaches for Parameter Estimation

In addition to the above mentioned methods, some stochastic models have been applied for parameter estimation.

In [17], the authors proposed an ARIMAX (Auto Regressive Moving Average with Integrator in the noise model and eXogenous input) method to estimate the parameters of a low-order dynamic model for a power system. The power system's natural frequency, damping factor and stiffness (power-frequency characteristic) are estimated by this method using long-time (1 hour) operational data. The performance of this method is highly affected by the signal-to-noise ratio (SNR) since the operational data (not the fault data) is used. And some high-order parameters are difficult to estimate with this method.

In [18], the authors employed an extended Kalman filter to calibrate the parameters of the dynamic models for a power system. Using the data from PMU following a fault, the parameters converged to the true value by a prediction-correction process. However, the noise effect is not studied in this paper. Theoretically, the extended Kalman filter is not an optimal estimator since the measurement and the state transition model are both non-linear [19]. So the estimation results are slightly less accurate than sensitivity analysis.

2.4 Review Conclusions

Both intelligent optimization and conventional optimization methods have been applied for power system dynamic parameter estimation. Conventional gradient based method can quickly converge to the optimal points with a good initial guess, and it may diverge or be trapped in local optimal when the initial guess is far away from the actual value. The population based intelligent methods are not affected by a poor initial guess. However, the computation burden caused by the large number of fitness evaluations is the main obstacle of the application in power system dynamic parameter estimation.

CHAPTER 3

THE PROPOSED TWO-STEP ALGORITHM

The objective of this dissertation is to make the simulation results match the measured curve by adjusting the parameters of dynamic devices in a power system. It can be formulated as a nonlinear curve-fitting problem. The mathematical optimization objective function is:

$$Min \quad J_{x \in P}(x) = (z - h(x))^T W (z - h(x)) \tag{3.1}$$

Where:

x is the parameter vector.

P is the parameter space.

z is the measurement vector corresponding to the recorded sample set of measured signals such as power output of the generator.

h(x) is the vector corresponding to the simulated set of the same signal as z.

W is the weight matrix. Usually the identity matrix is used as weight matrix.

As described in Chapter 2, the conventional optimization methods are highly dependent on the quality of the initial guess. The intelligent method is not affected by the initial guess but requires a tremendous number of fitness evaluations which are very time-consuming because of the need to solve the differential equations of the whole system. Therefore, this dissertation proposes a two-step hybrid method to achieve a balanced solution for this problem.

A new intelligent optimization method, particle swarm optimization (PSO) is firstly applied for the global search to find an approximate solution at the first step, then the sensitivity analysis is run starting from the approximate solution for local search to achieve the accurate parameters in the second step.

3.1 Particle Swarm Optimization

Many intelligent algorithms, such as Particle Swarm Optimization, Genetic Algorithm, Neural Network, etc have been adopted by power engineers to solve complicated optimization problems which were previously very difficult for conventional optimization algorithms such as sensitivity analysis. All of them can be applied to solve the non-linear curve-fitting problem discussed in this dissertation. However, neural network requires a tremendous amount of historical data for training purposes which is unavailable since the fault seldom occurs at the same location under the same operating condition.

PSO explores multiple solutions in parallel like GA. However, PSO utilizes a cooperative manner unlike GA, based on a competitive strategy. Additionally, PSO is a simpler algorithm which has fewer parameters and operations than GA. The main advantages of PSO over other optimization methods are: 1) derivative-free, 2) no good initial guess is required, 3) ability to escape local minima, 4) nature of plurality which easily fits into a parallel architecture, and 5) easy to implement and program.

As a stochastic population-based intelligent optimization algorithm, PSO was originally developed in 1995 by Kennedy and Eberhart as a novel intelligent optimization method [20-21]. The optimized solution is achieved through the mathematical simulation of the social behavior of bird flocks. Many power system researchers have applied PSO to power system optimization problems such as Economic Dispatch, Reactive Power Control, and Optimal Power Flow since 1999 [6]. The improvements of solution accuracy and computation efficiency were shown according to the research results. However, the computation burden is still significant compared to conventional derivative-based optimization methods. The application cases of PSO in power system parameter estimation, as far as we know, are focused upon induction motor [22] and dynamic equivalents [23].

In PSO, each particle represents a candidate solution and has two properties: position (x_i) and velocity (v_i). The velocity of a particle directs the flight of the particle. A population of particles, called a swarm, keeps flying around the search space until the stop criteria is satisfied.

There are many variants of PSO since its first development. This dissertation uses the standard PSO algorithm described in [24]. Each particle in the swarm is randomly initialized in the problem space. At each step, each particle is updated according to the formulas:

$$v_i^{(k+1)} = wv_i^{(k)} + c_1 * r_1 * (pbest_i^{(k)} - x_i^{(k)}) + c_2 * r_2 * (gbest^{(k)} - x_i^{(k)}) \tag{3.2}$$

$$x_i^{(k+1)} = x_i^{(k)} + v_i^{(k+1)} \tag{3.3}$$

Where:

k is the iteration index.

is the inertia weight.

$x_i^{(k)}$ is the position vector of i-th particle at k-th iteration.

$v_i^{(k)}$ is the velocity vector of i-th particle at k-th iteration.

c_1 and c_2 are two positive constants.

r_1 and r_2 are two random numbers in range [0, 1].

$pbest_i^{(k)}$ is the best position of i-th particle after k iterations.

$gbest^{(k)}$ is the best position of the whole swarm after k iterations.

The flowchart of the PSO algorithm for the non-linear curve-fitting problem is shown in Figure 3.1.

3.2 Sensitivity Analysis

The sensitivity matrix based on the least square principle is used at the second step of the proposed hybrid method. The initial guess of the sensitivity analysis comes from the PSO solution.

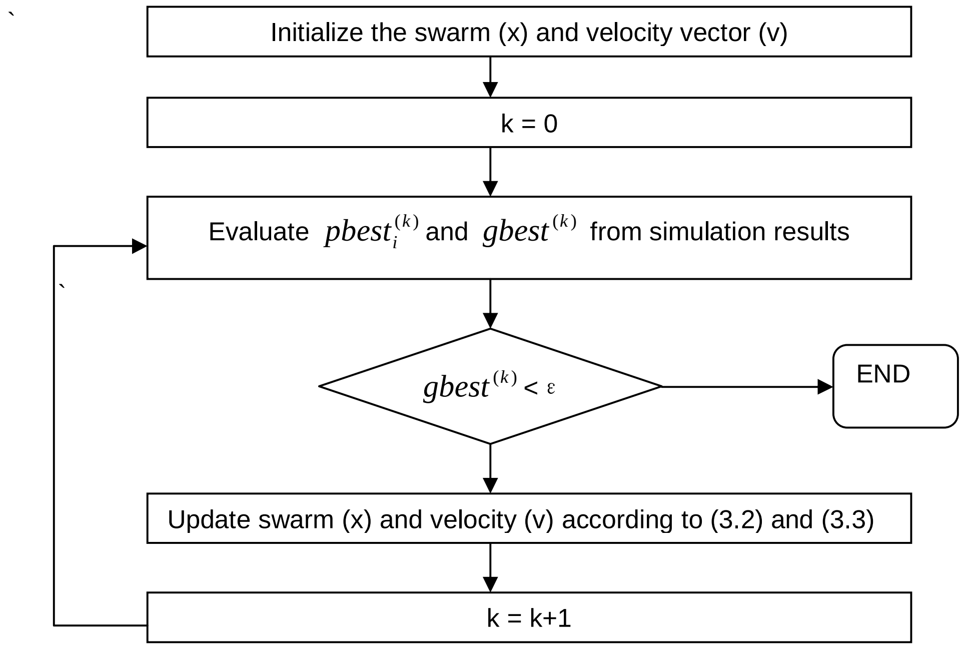


Figure 3.1 Flowchart of the PSO

3.3 The Proposed Procedure

The procedure of the two-step hybrid optimization method is listed below.

Step1. Particle Swarm Optimization (PSO)

The PSO algorithm is employed to find an approximate solution as the first step in the proposed hybrid method. The approximate solution is then used as the initial condition for sensitivity analysis in the second step.

Step2. Sensitivity Analysis (SA)

The sensitivity matrix method, based on WLS, is applied at the second step in the proposed hybrid method. SA searches the optimized parameters, starting with the approximate solution achieved by the PSO optimization in the first step.

The flow chart of the hybrid two-step method is shown in Figure 3.2. The tolerance[1] (PSO tolerance) is larger than the tolerance[2] (SA tolerance) since PSO is run to achieve an approximate solution while SA is run to achieve an accurate solution. The source codes are listed in Appendix A-1.

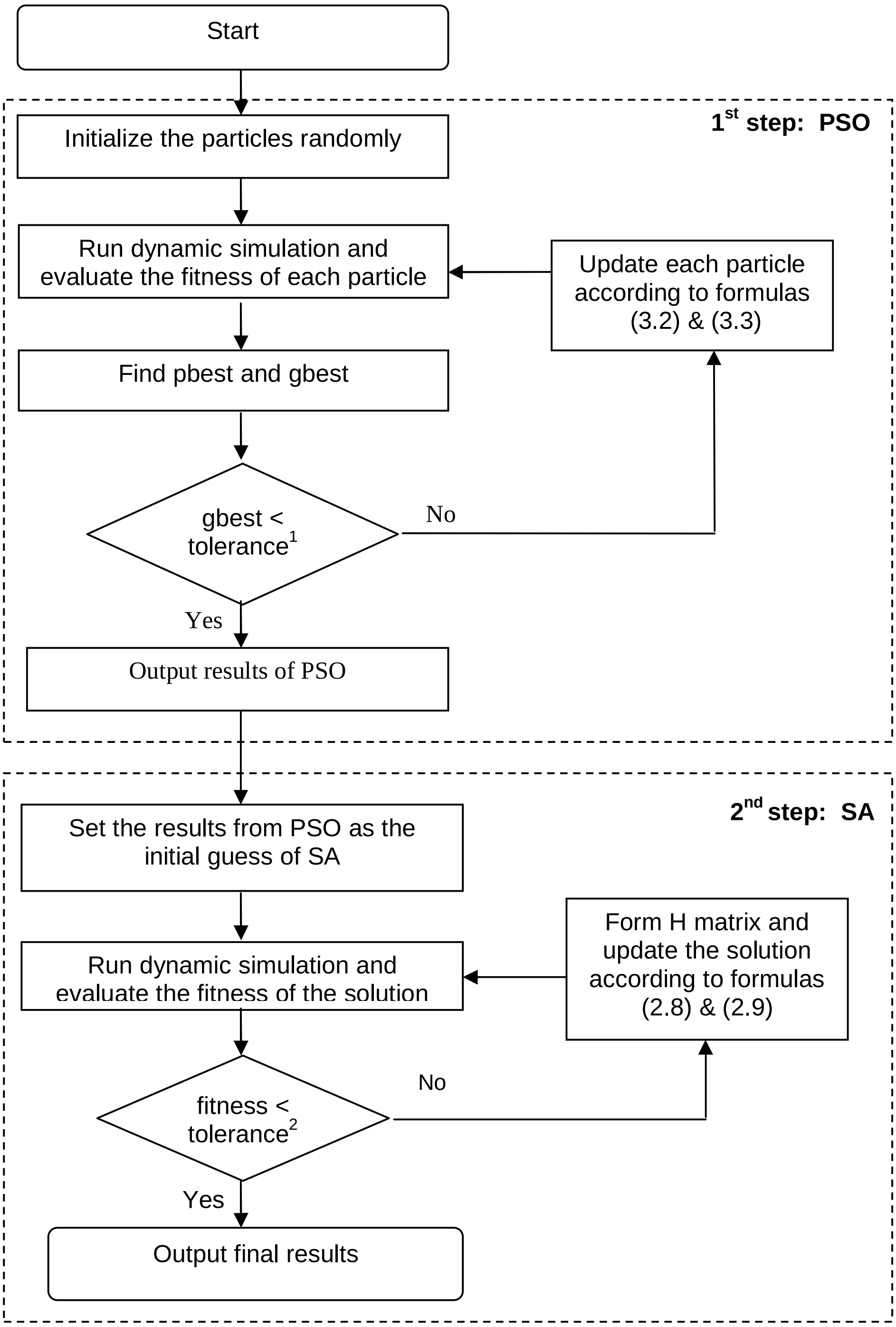


Figure 3.2 Flow Chart of the Proposed Hybrid Method

CHAPTER 4

KEY PARAMETERS IDENTIFICATION

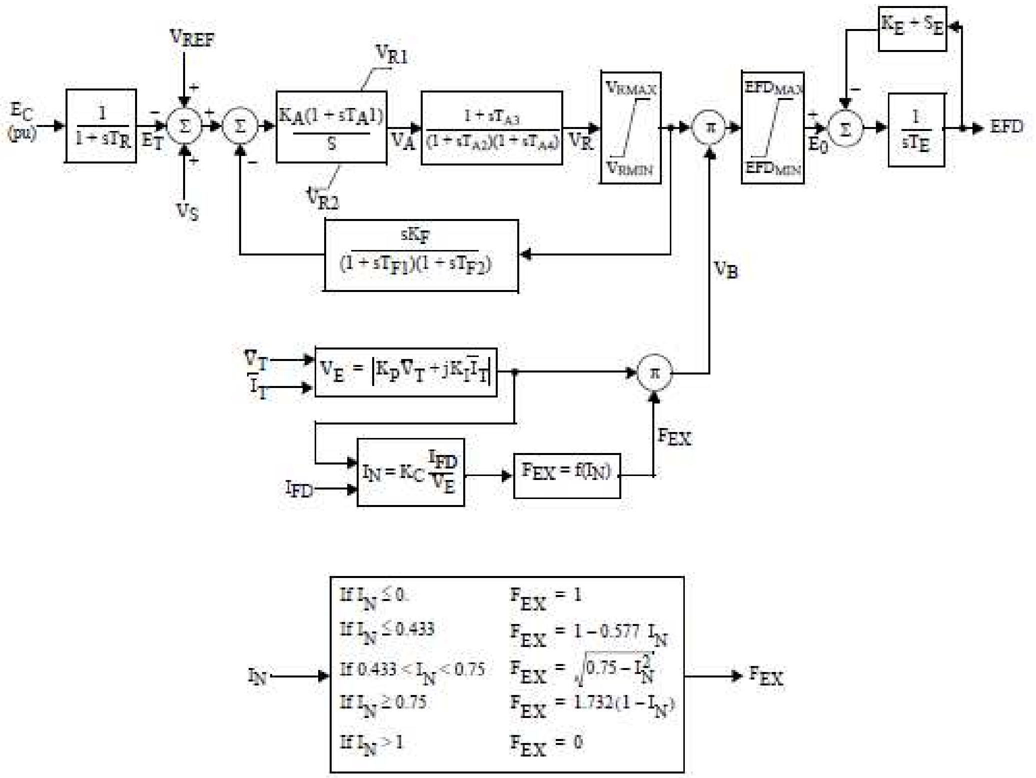
There are more than ten parameters in most of models of exciters, power system stabilizers and governors. Here are two examples of exciter models used in ERCOT system: "EXPIC1", a model of proportional/integral excitation system, has 24 dynamic parameters as shown in Figure 4.1. "EXAC1", a model of IEEE Type AC1 Excitation System, has 17 dynamic parameters as shown in Figure 4.2.

More parameters of the model make the estimation problem more complicated and less efficient. Fortunately, not all parameters are critical enough to be incorporated in the estimation problem. The parameters are generally categorized into four sets in terms of the purpose of estimation.

1) The parameters for hard limits and saturation curve such as V_{RMAX}, V_{RMIN}, E_1, $S(E_1)$, E_2, $S(E_2)$ etc. These parameters are usually set manually or seldom tuned. It is reasonable to assume that those parameters are accurate.
2) The zero parameters. Some parameters are set as zero to represent the absence of the corresponding sub-block in the main block of the model.
3) The parameters (except category 1 and 2) which have trivial impact on the result of dynamic events.
4) The parameters (except category 1 and 2) which have significant impact on the dynamic results.

Categories 1, 2 and 3 are excluded from the parameter estimation problem since they are not critical, or not used, or seldom tuned. Only the parameters in category 4 are defined as key parameters for estimation. Thus, the computation burden of the parameter estimation problem can be tremendously reduced.

CONs	#	Value	Description	CONs	#	Value	Description
J			T_R (sec)	J+12			T_{F2} (sec)
J+1			K_A	J+13			EFD_{MAX}
J+2			T_{A1} (sec)	J+14			EFD_{MIN}
J+3			V_{R1}	J+15			K_e
J+4			V_{R2}	J+16			T_e (sec)
J+5			T_{A2} (sec)	J+17			E_1
J+6			T_{A3} (sec)	J+18			SE_1
J+7			T_{A4} (sec)	J+19			E_2
J+8			V_{RMAX}	J+20			SE_2
J+9			V_{RMIN}	J+21			K_P
J+10			K_F	J+22			K_I
J+11			T_{F1} (>0.) (sec)	J+23			K_C



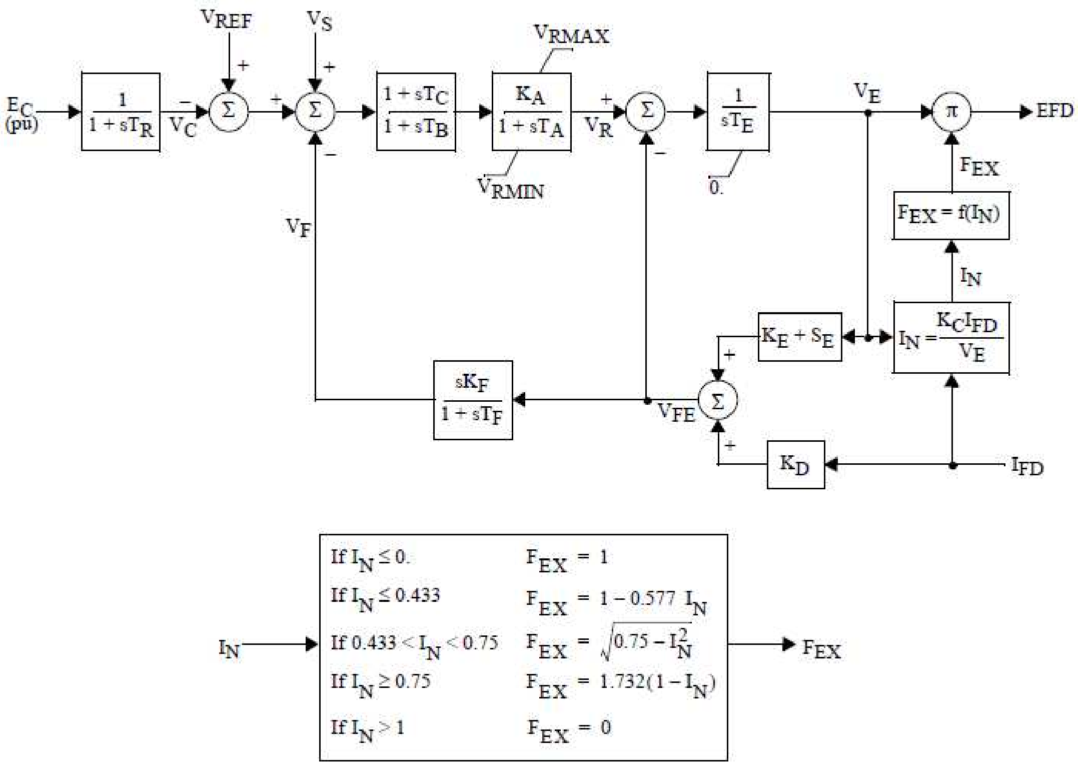
If ($K_P = 0$ and $K_I = 0$), then $V_B = 1$.

If $T_E = 0$, then $EFD = E_0$.

$V_S = VOTHSG + VUEL + VOEL$

Figure 4.1 the Parameter List and Model Diagram of EXPIC1 [25]

CONs	#	Value	Description	CONs	#	Value	Description
J			T_R (sec)	J+9			$T_F > 0$ (sec)
J+1			T_B (sec)	J+10			K_C
J+2			T_C (sec)	J+11			K_D
J+3			K_A	J+12			K_E
J+4			T_A (sec)	J+13			E_1
J+5			V_{RMAX}	J+14			$S_E(E_1)$
J+6			V_{RMIN}	J+15			E_2
J+7			$T_E > 0$ (sec)	J+16			$S_E(E_2)$
J+8			K_F				



$V_S = VOTHSG + VUEL + VOEL$

Figure 4.2 the Parameter List and Model Diagram of EXAC1 [26]

There are two kinds of data set that can be used to identify the importance of the parameters: response tests and dynamic event simulation. The response tests in PSS/E simulate the individual studied model in isolation mode, while the dynamic event simulation requires all of the models in the whole system for the studied dynamic event simulation. So the response test requires substantially less computation time than the dynamic event simulation since only the studied models are active in simulation.

4.1 Response Test in PSS/E

4.1.1 Response Test of Exciters

The exciter response test in PSS/E is used to verify excitation system data by the step response simulation of excitation systems in isolation [27]. There are two distinct step response tests in PSS/E: a response ratio test and an open-circuit setpoint step test. The original purpose of these two tests in PSS/E is to check the performance of exciters and governors; however, they can also be used for the exciter key parameter identification because of the property of the tests.

In the PSS/E response ratio test, the voltage regulator reference setting is automatically raised by a large amount, forcing the exciter system to reach its ceiling as rapidly as possible. The test is carried out for at least two seconds to allow the rotating exciter to reach its ceiling. The field voltage (E_{fd}) is recorded during the test.

In the PSS/E open-circuit response test, a step change of about five percent is applied to the voltage regulator references and the resulting responses of field voltage (E_{fd}) and generator terminal voltage (E_{term}) are observed. The normal simulation time is set as at least five seconds to allow the exciter to reach steady state.

The response ratio can reflect the model of the rotating machine exciters. However, it provides no information about the voltage regulator gains and time constants. These data can be captured by the open-circuit response test. Thus, the two kinds of response tests are jointly run to identify the exciter key parameters.

4.1.2 Response Test of Governors

The PSS/E governor response test simulates the response of the governing loops of units in isolation to a step change in load [27]. The original purpose of the test is to ensure the governor gain and time constant parameters correspond to a correctly tuned and well damped response. The governor response test can also be used for the governor key parameter identification since the test reflects the model of governors.

The test initializes each governor to a load level specified by the user and then simulates the response of the governors to a step change in load. The load is held constant (independent of frequency) after the step so that the response indicates the damping due to the turbine and governor loop only. The governors should be initialized to about 0.8 per-unit load and the load step should be approximately 0.1 per unit. The damping of hydro governing loops is usually decreased with increasing load and hence that response tests should normally be made near full load for these units.

4.2 Dynamic Events Simulation

The dynamic event simulation can provide directly the key parameters by sensitivity analysis since the objective of the parameter estimation is to drive the dynamic event simulation results to match the field recording data. However, there are two disadvantages when compared to the response tests in PSS/E:

1) All of the models in the whole system are required for the studied dynamic event simulation which means longer computation time.
2) Some of the key parameters may be missed because of the non-linear property of the power system.

4.3 Identification Method

According to the definition of the key parameters, the method of sensitivity analysis is applied for key parameter identification.

The outputs of the PSS/E response tests or the dynamic event simulation with the original parameters are set as the reference curve (denoted by ref vector). For any parameter which is not in category 1 or 2, a step change (20%) is added to it while other parameters are kept unaltered. The PSS/E response tests or the dynamic event simulation with the tuned parameter are run and recorded as chg vector. Then the mismatch between chg and ref is evaluated and denoted by: $J^* = (chg - ref)^T (chg - ref) / ref^T ref$. If the mismatch (J^*) is larger than a pre-defined tolerance (e.g. 1%), this parameter is identified as a key parameter. Otherwise, this parameter is not included as a key parameter.

4.4 Key Parameter Pre-Scan by PSS/E Response Test

There are 21 types of exciters and 9 types of governors in ERCOT system dynamic models. PSS/E response tests were performed for each type of model in the ERCOT system and sensitivity analysis was performed on each parameter. The key parameters are identified as shown in Table 4.1 and Table 4.2 according to the simulation results. The source codes are listed in Appendix A-2. The details of the sensitivity analysis of the PSS/E response test are listed in Appendix B.

Table 4.1 Key Parameters of Exciters in ERCOT

Model Name	Key Parameters
URST5T	T_{C2}, T_{B2}
ESST4B	K_{PR}, T_a, K_{PM},K_P
ESAC8B	K_A, K_E, K_P
ESST1A	T_C, T_B, T_{C1}, T_{B1},K_a
ESAC5A	K_E, T_E,K_F,T_{F1},T_{F2}
ESAC2A	K_A, K_B, T_E, K_F, T_F, K_E, K_C, K_D
ESAC1A	K_A, T_E,K_F, T_F, K_E
EXPIC1	K_A, T_{A1}, T_{A2}, K_P
EXST2A	K_E, T_E, K_P
EXST3	K_J, T_C, T_B, K_G, K_P, X_L
EXST1	T_R, K_C, K_F, T_F
EXDC2	K_A, T_A, T_B, T_C, K_E, K_F,T_{F1}
EXAC4	T_C, T_B, K_A, T_A
EXAC3	V_A, T_E, K_R, K_F, T_F, K_E, K_C, K_D, T_B, T_C, K_A
EXAC2	K_B, T_E, K_F, T_F, K_D, K_E
EXAC1	T_E, K_F, T_F, K_E, K_C, K_D

Table 4.1-*Continued*

IEEEX1	T_E, K_F, T_{F1}
SEXS	K, T_A
IEEET4	K_R, T_{RH}, K_V, T_E, K_E
IEEET3	T_E, K_F, T_F, K_P, K_I, K_E
IEEET2	T_A, K_E, T_E, K_F, T_{F1}, T_{F2}
IEEET1	T_E, K_E, T_F

Table 4.2 Key Parameters of Governors in ERCOT

Model Name	Key Parameters
GGOV1	K_{TURB}, LD_{REF}, T_{FLOAD}, T_{RATE}
GAST2A	W, Z, T_{RATE}, K_3, A, C, BF_2, T_R, T_C
TGOV3	K_1, T_5, K_2, K_3
IEEEG2	T_4
IEEEG1	T_4, K_1
IEESGO	T_3, T_4, T_5, T_6, K_1, K_2, K_3
HYGOV	R, r, T_r, T_g, TW,A_t, Dturb
GAST	R, T_1,T_2
TGOV1	R,T_1, D_t

CHAPTER 5

TEST CASE SIMULATION

In this chapter, we study the feasibility and effectiveness of the proposed method using the assumed test case and data.

5.1 Case Configuration and Data Set

A power plant with two identical generators in the ERCOT system is studied as a test case. The power plant connects with the ERCOT system through nine 345kV lines as shown in Figure 5.1. The exciter and governor types are "EXAC1" and "IEESGO" respectively. The detailed parameters of the models are listed as Table 5.1 and Table 5.2. The block diagram of "EXAC1" model is shown in Figure 4.2. The block diagram of the "IEESGO" model is shown in Figure 5.2.

In the test case, the Short-Circuit fault event near the power plant is assumed. The scenario is listed as below:

0s: A three-phase short-circuit fault at bus B1 occurs.

0.0667s (4 Cycles): The fault is cleared and a single line (B1-B7) is tripped

10s: Simulation stops.

For the purpose of algorithm validation, it is assumed that the dynamic parameters received from the ERCOT are correct. The dynamic simulation of this scenario is run based on these existing dynamic parameters. The simulation results (P_{gen} and Q_{gen}) are treated as the "measurement" data. After that, these parameters are assumed to be unknown and will be altered to some other values. The proposed method will use the "measurement" data to adjust these altered parameters back to the existing value which is defined as the "Target value" for the proposed method.

According to the simulation results, the reactive power of the generator is sensitive to the parameters of the exciter and the active power of the generator is sensitive to the parameters of the governor. Therefore, the task of parameter estimation is to adjust the key parameters of exciters and governors to make the simulation results fit the curves of reactive power and active power of the generators individually.

The algorithm is programmed in Python 2.5 with PSS/E simulation engine (version 31) and run on a computer with Intel Core 2 Q6600 Quad-Core CPU 2.40GHz and 4 GB RAM.

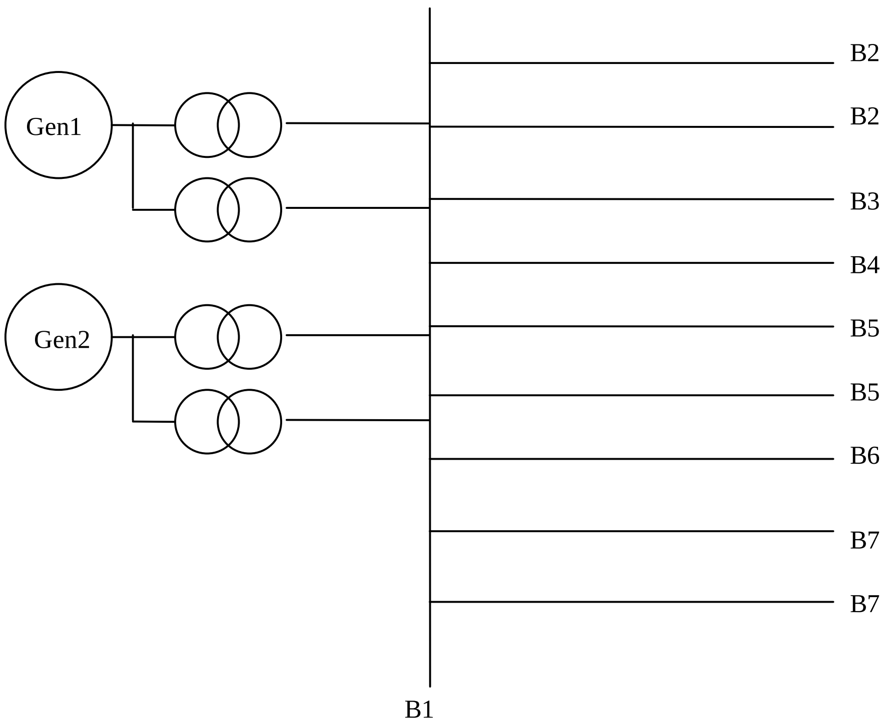


Figure 5.1 One Line Diagram of Local System of the Test Case

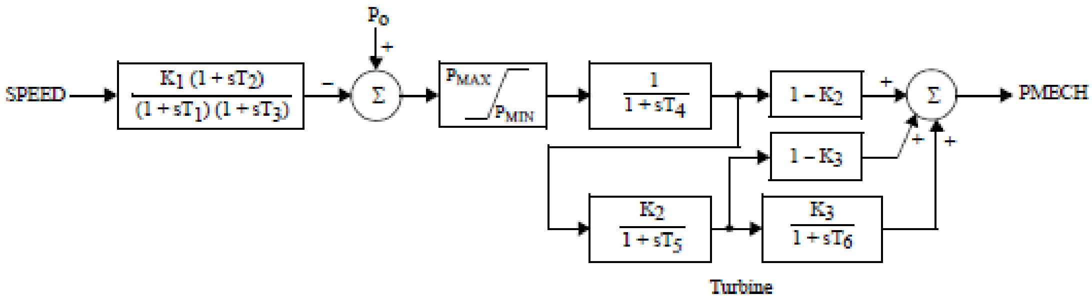


Figure 5.2 Model Diagram of IEESGO [28]

Table 5.1 EXAC1 Exciter Parameters

Name	Value	Name	Value
T_R	0.005	T_F	1
T_B	0	K_C	0
Tc	0	K_D	0
K_A	400	K_E	1
T_A	0.02	E_1	2.5725
V_{RMAX}	7.29	$S(E_1)$	1.05
V_{RMIN}	-6.56	E_2	3.43
T_E	1.1	$S(E_2)$	1.13
K_F	0.03	/	/

Table 5.2 IEESGO Governor Parameters

Name	Value	Name	Value
T_1	0	K_1	20
T_2	0	K_2	0.664
T_3	0.1	K_3	0
T_4	0.25	P_{MAX}	0.875
T_5	6	P_{MIN}	0
T_6	0.3	/	/

5.2 Key Parameter Identification

The key parameters are jointly identified from the results of PSS/E response test and the sensitivity analysis of the dynamic short-circuit event.

5.2.1. Exciter

According to the key parameter identification results using the PSS/E response test in Chapter 4, K_E, T_E, K_F, T_F, K_C and K_D are declared as the key parameters of the EXAC1 exciter model. However, K_C and K_D are not used in this model of the two particular generators since they are zero-valued.

The sensitivity analysis is also performed for the defined short-circuit event. For each parameter, J^* is the mismatch between original curve (generator reactive power output, denoted as ref) and the curve (denoted as chg) after a step change (10%) is added to that parameter. The mismatch index is defined by $J^* = (chg - ref)^T (chg - ref) / ref^T ref$. The computation results are shown in Table 5.3.

Several critical exciter parameters, V_{RMAX} , K_E , T_E , K_F , T_F , E_1, $S(E_1)$, E_2 and $S(E_2)$ are identified according to PSS/E simulation results of the dynamic event. V_{RMAX} is the set hard limit which is seldom tuned. The saturation characteristic ($E_1, S(E_1)$, E_2 and $S(E_2)$)are usually determined from the field test. So the key parameters of the exciter in this power plant are K_E, T_E, K_F and T_F from the sensitivity analysis results of the dynamic event. They are the same with the key parameters achieved from PSS/E response test. Therefore, the key parameters to be identified are: K_E, T_E, K_F and T_F.

5.2.2. Governor

According to the key parameter identification results using the PSS/E response test in Chapter 4, T_3, T_4, T_5, T_6, K_1, K_2 and K_3 are declared as the key parameters of the IEESGO governor model.

The sensitivity analysis is also performed for the defined short-circuit event. For each parameter, J^* is the mismatch between original curve (generator active power output, denoted as ref) and the curve (denoted as chg) after a step change (10%) is added to that parameter. The computation results are shown in Table 5.4. The mismatch index is defined by $J^* = (chg - ref)^T (chg - ref) / ref^T ref$.

Several critical governor parameters, T_3 , T_4 , K_1 , K_2 and P_{MAX} , are identified according to the PSS/E simulation results. P_{MAX} is the set hard limit which is seldom tuned. So the key parameters of the governors in this power plant are T_3, T_4, K_1 and K_2.

T_6 and K_3 are zero value in this particular governor model associated with this power plant which means the absence of the corresponding blocks. So they are excluded from the key parameters even though they are declared as key parameters in the PSS/E response test. T_5 is also excluded from the key parameters since the corresponding J^* is very small compared to other parameters.

Therefore, the key parameters to be identified are: T_3, T_4, K_1 and K_2.

Table 5.3 Identification Results of Exciter Key Parameters Using the Dynamic Event

Name	J^*	Name	J^*
T_R	0.000246	T_F	0.00623
T_B	0	K_C	0
Tc	0	K_D	0
K_A	0.001191	K_E	0.007053
T_A	0.00027	E_1	0.053793
V_{RMAX}	0.017049	$S(E_1)$	0.042164
V_{RMIN}	0	E_2	0.039508
T_E	0.005357	$S(E_2)$	0.043276
K_F	0.007358	/	/

Table 5.4 Identification Results of Governor Key Parameters Using the Dynamic Event

Name	J^*	Name	J^*
T_1	0	K_1	0.001061
T_2	0	K_2	0.002323
T_3	0.000691	K_3	0

Table 5.4-*Continued*

T_4	0.001043	P_{MAX}	0.018301
T_5	0.000062	P_{MIN}	0
T_6	0	/	/

5.3 Dynamic Parameter Estimation

5.3.1. Exciter parameters Estimation

To compare the proposed method with the conventional method, both of these two methods are used for dynamic parameter estimation in the test case.

5.3.1.1 Conventional Analysis

For illustration purpose, two cases with poor initial guesses are run by conventional sensitivity analysis. The target value of each of the four key parameters can be achieved from the initial guess in case 1. However, it takes 687.7 minutes. In case 2, the target values of four key parameters cannot be obtained since divergence occurs.

The results of these two cases are listed as Table 5.5 and 5.6. The mismatch between measurements and simulation results of the reactive power output is denoted by $J^*(x) = (z - h(x))^T (z - h(x)) / z^T z$.

5.3.1.2 Hybrid two-step method

A simulation result of the hybrid two-step method is listed as below:

Step1. Particle Swarm Optimization (PSO)

The PSO parameters are set as:

M (Population Size) = 40;

W (Inertia weight matrix) = I (Identity matrix);

C_1 (constant for swarm optima) = 1;

C_2 (constant for particle optima) = 1.

The range of the key exciter parameters is set in Table 5.7.

$J^*(x)$ decreases to 0.020 after 7 iterations that takes 94.13 minutes. The solution from the PSO is used as the initial guess for the second step.

Step2. Sensitivity Analysis (SA)

It takes another 83.21 minutes for $J^*(x)$ to decrease to 0.00009 after 20 iterations. The results of the second step are listed as Table 5.8.

As one can see, the final solution is nearly the same with the target value. The total time of two-step method is 177.34 minutes. The computational burden has been cut significantly compared to conventional sensitivity analysis.

Table 5.5 Conventional Sensitivity Analysis Case 1

	Target Value	Initial		Iter. Num.	Final		Time (min)
		Guess	J^*		Value	J^*	
K_E	1.0	2.9	0.187	166	1.001	0.00004	687.7
T_E	1.1	2.5			1.0990		
K_F	0.03	0.12			0.02997		
T_F	1.0	0.5			0.99995		

Table 5.6 Conventional Sensitivity Analysis Case 2

	Target Value	Initial Guess	J^* (Initial)	Result
K_E	1.0	4.5151	1.27	Diverge
T_E	1.1	3.2483		
K_F	0.03	0.1119		
T_F	1.0	3.9846		

Table 5.7 the Range of the Key Exciter Parameters

Parameters	Lower Limit	Upper Limit
K_E	0	5
T_E	0	5
K_F	0	0.15
T_F	0	5

Table 5.8 The results of the second step of the hybrid method

	Target Value	Initial		Iter. Num.	Final		Time (min)
		Guess	J^*		Value	J^*	
K_E	1.0	0.23	0.020	20	0.995	0.00009	83.21
T_E	1.1	1.6514			1.105		
K_F	0.03	0.0857			0.030		
T_F	1.0	2.304			1.002		

5.3.2. Governor parameters Estimation

5.3.2.1 Conventional Analysis:

One case with poor initial guesses is run by conventional analysis. The target value of the four key parameters cannot be achieved from the initial guess in this case since the local minimum is trapped. The detailed case is listed as Table 5.9.

5.3.2.2 Hybrid two-step method:

A typical result of the hybrid two-step method is listed as below:

Step1. Particle Swarm Optimization (PSO)

The PSO parameters are set as:

M (Population Size) = 24;

W (Inertia weight matrix) = I (Identity matrix);

C_1 (constant for swarm optima) = 1;

C_2 (constant for particle optima) = 1.

The range of the key exciter parameters is set in Table 5.10.

$J^*(x)$ decreases to 0.0145 after 4 iterations that takes 39.33 minutes. The solution of PSO will be used as the initial guess of the second step.

Step2. Sensitivity Analysis (SA)

It takes another 204.52 minutes for $J^*(x)$ to decrease to 0.000001 after 48 iterations. The results of the second step are listed as Table 5.11.

The final solution is nearly the same with the target value. The total time requirement of two-step method is 243.85 minutes.

Table 5.9 Conventional Sensitivity Analysis Case 1

	Target Value	Initial		Iter. Num.	Final	
		Guess	J^*		Value	J^*
T_3	0.1	1.4	1.65	1	1.398	1.65
T_4	0.25	0.28			0.281	
K_1	20	160			160.4	
K_2	0.664	10			10.01	

Table 5.10 the Range of the Key Governor Parameters

Parameters	Lower Limit	Upper Limit
T_3	0	2
T_4	0	5
K_1	0	400
K_2	0	10

Table 5.11 the Results of the Second Step of the Hybrid Method

	Target Value	Initial		Iter. Num	Final		Time (min)
		Guess	J^*		Value	J^*	
T_3	0.1	1.51	0.0145	48	0.100	0.000001	204.52
T_4	0.25	3.34			0.250		
K_1	20	351.99			19.99		
K_2	0.664	0.362			0.664		

5.3.3. Uncertainties Analysis

The parameter uncertainties of neighboring generators and measurement noise are analyzed in this dissertation. Gaussian noise (1% standard deviation) is added to measurements and Gaussian noise (10% standard deviation) is added to parameters of neighboring generators. The tests show the solution is slightly biased but still close to the target value.

5.3.3.1 Exciter parameters:

Step1. Particle Swarm Optimization (PSO)

The PSO parameter setting is the same with the previous case without noise.

After 65.92 minutes, $J^*(x)$ decreases to 0.016 after 5 iterations. The PSO solution will be used as the initial guess for the second step.

Step2. Sensitivity Analysis (SA)

It takes another 83.79 minutes for $J^*(x)$ to decrease to 0.010 after iterations. The results of the second step are listed as Table 5.12.

The final solution is still close to the target value. The total time of two-step method is 149.71 minutes.

5.3.3.2 Governor parameters:

Step1. Particle Swarm Optimization (PSO)

The PSO parameter setting is the same with the previous case without noise.

$J^*(x)$ decreases to 0.0150 after 5 iterations (36.10 minutes). The solution of PSO will be used as the initial guess for the second step.

Step2. Sensitivity Analysis (SA)

$J^*(x)$ decreases to 0.010 after 38 iterations (162.17 minutes). The results of the second step are listed as Table 5.13.

The final solution is still close to the target value. The total time of the two-step method is 198.27 minutes.

Table 5.12 the Results of the Second Step of the Hybrid Method for Exciter under Noise

	Target Value	Initial		Iter. Num.	Final		Time (min)
		Guess	J^*		Value	J^*	
K_E	1.0	0.248	0.016	20	0.922	0.010	83.79
T_E	1.1	1.792			1.142		
K_F	0.03	0.053			0.032		
T_F	1.0	0.993			1.004		

Table 5.13 the Results of the Second Step of the Hybrid Method for Governor under Noise

	Target Value	Initial		Iter. Num.	Final		Time (min)
		Guess	J^*		Value	J^*	
T_3	0.1	0.581	0.015	38	0.112	0.010	162.17
T_4	0.25	1.739			0.217		
K_1	20	79.30			19.84		
K_2	0.664	0.629			0.681		

5.3.4 Multi-core Computation

There are thousands of buses and hundreds of generators in the EROCT system. Dynamic simulation of the ERCOT system involves the computation of thousands of algebraic and differential equations. Usually one dynamic simulation of the ERCOT system requires about one minute using a typical PC. Dynamic simulation of the cases by PSS/E occupies most of the time of the proposed method. Fortunately, newer PC systems come with multiple core processors. Therefore, multiple dynamic simulation cases can be performed simultaneously on a multi-core PC with only one PSS/E license.

PSO significantly benefits from multi-core processors because of the population-based nature. Multi-core processors can also speed up the sensitivity analysis since gradients of some parameters can be evaluated simultaneously. An open source software package, Parallel Python [29], is integrated in the program to take full advantage of Quad-core CPU. The test cases show the benefits as below. The computation time of each iteration of the first step (PSO) can be cut down to 25 to 35 percent, and the computation time of each iteration of the second step (SA) can be cut down to 35 to 50 percent. It drastically relieves the computation burden.

The same level of Gaussian noise is added to measurements and the parameters of neighboring generators.

5.3.4.1 Exciter parameters

Step1. Particle Swarm Optimization (PSO)

The computation parameter setting of PSO is the same with the previous exciter parameter estimation case.

$J^*(x)$ decreases to 0.0244 after 10 iterations (38.85 minutes). The solution of PSO will be used as the initial guess of the second step.

Step2. Sensitivity Analysis (SA)

$J^*(x)$ decreases to 0.0094 after 24 iterations (42.34 minutes). The results of the second step are listed as Table 5.14.

The final solution is still close to the target value. The total time of two-step method is 81.19 minutes.

5.3.4.2 Governor parameters:

The solution of the proposed hybrid method is still reasonable even though the solutions are a little biased.

Step1. Particle Swarm Optimization (PSO)

The computation parameter setting of PSO is the same with the previous governor parameter estimation case.

$J^*(x)$ decreases to 0.0149 after 5 iterations (11.15 minutes). The solution of PSO will be used as the initial guess of the second step.

Step2. Sensitivity Analysis (SA)

$J^*(x)$ decreases to 0.010 after 44 iterations (77.33 minutes). The results of the second step are listed as Table 5.15.

The final solution is still close to the target value. The total time of two-step method is 88.48 minutes.

Table 5.14 the Results of the Second Step of the Hybrid Method for Exciter Using Multi-core Computation under Noise

	Target Value	Initial		Iter. Num	Final		Time (min)
		Guess	J^*		Value	J^*	
K_E	1.0	0.196	0.0244	24	0.883	0.0094	42.34
T_E	1.1	1.616			1.154		
K_F	0.03	0.102			0.0328		

Table 5.14-*Continued*

T_F	1.0	2.714			1.0139		

Table 5.15 the Results of the Second Step of the Hybrid Method for Governor Using Multi-core Computation under Noise

	Target Value	Initial		Iter. Num.	Final		Time (min)
		Guess	J^*		Value	J^*	
T_3	0.1	0.842	0.0149	44	0.111	0.010	77.33
T_4	0.25	1.531			0.220		
K_1	20	94.70			18.01		
K_2	0.664	0.385			0.652		

5.4 Summary

This chapter uses "EXAC1" exciter and "IEESGO" governor as test case to study the feasibility and effectiveness of the propose method. The simulation results show the proposed method can reach the target value of parameters. Measurement noise and parameter uncertainties have a slight effect on the solution. Multi-core processors can significantly speed up the program using parallel programming.

CHAPTER 6

APPLICATION OF THE PROPOSED METHOD WITH THE FIELD RECORDING INFORMATION

The proposed method exhibits promising results for the assumed test cases. ERCOT recently witnessed the occurrence of a dynamic event that resulted in the tripping of a generation facility. The proposed method is applied for the dynamic parameter estimation using the field recording information during the aforementioned event.

6.1 Introduction to the Power Plant and the Dynamic Event Scenario

The generation facility under study has three generator units (totaling to about 2400MW) and connected to the ERCOT transmission network through five 345kV lines as shown in Figure 6.1.

The Generation Unit #2 tripped due to an unidentified equipment failure in the plant during the course of the dynamic event. The dynamic event was not characterized by any fault on the power system network. The active and reactive power output of Generation Unit #3 was recorded during the event. The proposed method is used to validate/identify the dynamic parameters for Generation Unit #3 according to the field recording information.

The dynamic models of Generation Unit #3 in ERCOT dynamic data are listed in Table 6.1 ~ 6.5.

Table 6.1 The Generator Model of Generation Unit #3 (Type "GENROU")

T_{d0}'	5.0800	X_q	2.2110
T_{d0}''	0.0400	X_d'	0.4370
T_{q0}'	0.5600	X_q'	0.6500
T_{q0}''	0.0700	$X_d''=X_q''$	0.3500
H	2.6400	Xl	0.2700
D	0.0000	S(1.0)	0.1600
X_d	2.2340	S(1.2)	0.5370

Table 6.2 The Exciter Model of Generation Unit #3 (Type “EXAC1”)

T_R	0.0000	T_F	2.5000
T_B	0.0000	K_C	0.0500
T_C	0.0000	K_D	0.4500
K_A	600.0000	K_E	1.0000
T_A	0.0000	E_1	2.9200
V_{RMAX}	7.3100	$SE(E_1)$	0.6500
V_{RMIN}	-6.5800	E_2	3.8900
T_E	0.8000	$SE(E_2)$	0.8800
K_F	0.0350		

Table 6.3 The Governor Model of Generation Unit #3 (Type “IEESGO”)

T_1	0.1800	K_1	17.4000
T_2	0.0300	K_2	0.7100
T_3	0.1500	K_3	0.5700
T_4	0.2500	P_{MAX}	0.9900
T_5	10.0000	P_{MIN}	0.3000
T_6	0.5000		

Table 6.4 The PSS Model of Generation Unit #3 (Type “PSS2A”)

TW_1	10.0000	T_9	0.1000
TW_2	10.0000	KS_1	5.0000
T_6	0.0000	T_1	0.2000
TW_3	10.0000	T_2	0.0200
TW_4	0.0000	T_3	0.2000
T_7	10.0000	T_4	0.0200
KS_2	1.8900	VST_{MAX}	0.1000
KS_3	1.0000	VST_{MIN}	-0.0500
T_8	0.5000		

Table 6.5 The MAX Exciter Model of Generation Unit #3 (Type “MAXEX2”)

E_{FD}	68.0000	E_{FD3}	1.3000
E_{FD1}	1.0100	$TIME_3$	20.0000
$TIME_1$	45.0000	E_{FDDES}	1.0000
EFD_2	1.1500	K_{MX}	0.0020
$TIME_2$	35.0000	V_{LOW}	-0.2000

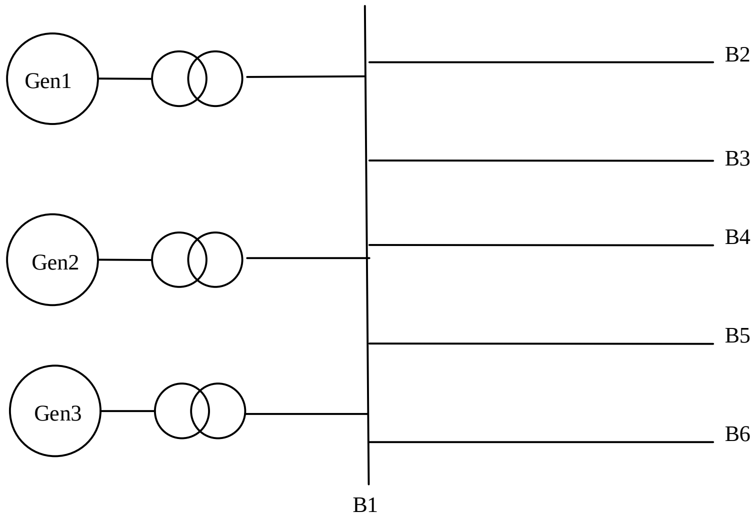


Figure 6.1 One Line Diagram of Local System of the Studied Case

6.2 Comparison between Simulation Results and Recording Information

The mismatch index between simulation results and recorded data is defined as:

$$J^* = \frac{(\mathrm{Re}\,c - Sim)' * (\mathrm{Re}\,c - Sim)}{\mathrm{Re}\,c' * \mathrm{Re}\,c}$$

where:

Rec is the vector corresponding to recording data.

Sim is the vector corresponding to simulation results.

The comparison results are listed in table 6.6 and figure 6.2 ~6.3.

Table 6.6 Mismatch Index for Generation Unit #3

	J^*
Active power output	2.28%
Reactive power output	41.34%

The active power output of Generation Unit #3 in the simulation results matches closely when compared to the recording data as evident from table 6.6 and figure 6.2. However, the reactive power output of Generation Unit #3 in the simulation deviates substantially from the recording data (table 6.6 and figure 6.3).

The active power output of the generator in question is mainly influenced by the parameters of governor while the reactive power output is mainly influenced by the parameters of exciter and PSS. Given the close proximity of model and field active power results, the governor parameters in the model seem to reasonably reflect the actual facility and need not to be tuned. However, the exciter and PSS parameters do not seem to accurately reflect the actual facility and need to be estimated and validated since there is no good agreement between simulation results and recording data for the generator reactive power output.

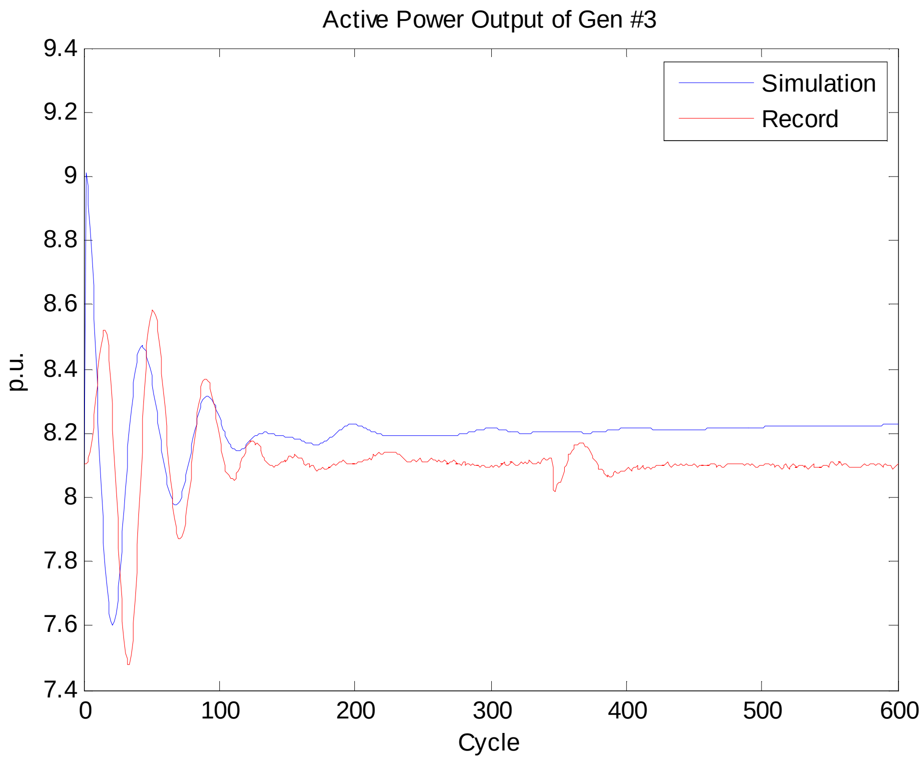


Figure 6.2 Comparison between Simulation Results and Recorded Data of Active Power Output

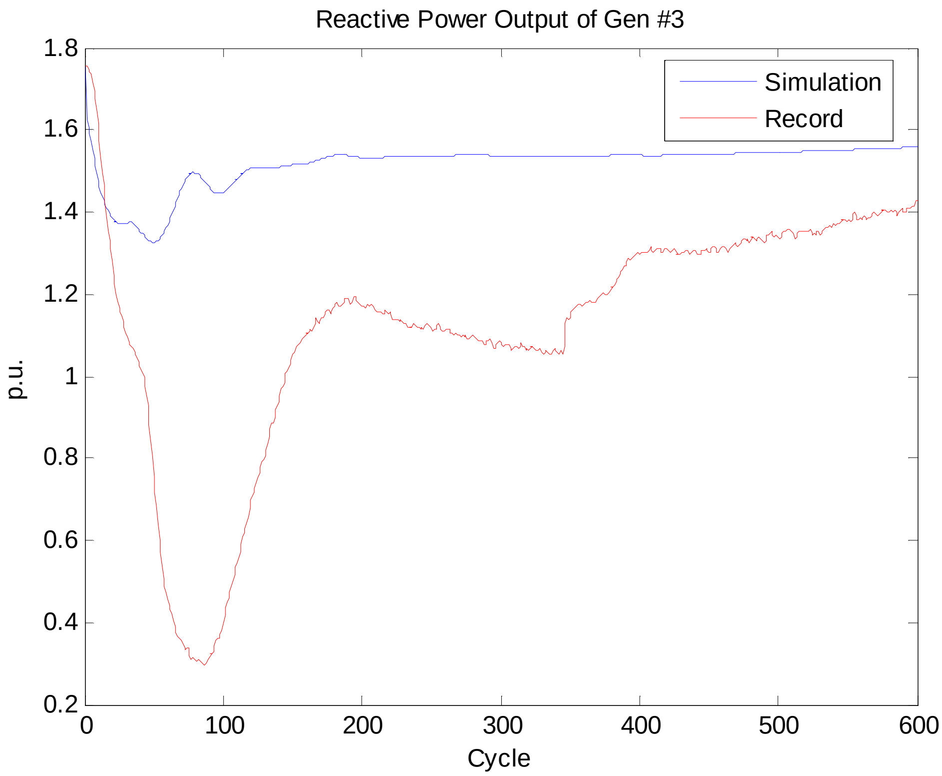


Figure 6.3 Comparison between Simulation Results and Recorded Data of Reactive Power Output

6.3 Key Parameter Identification

There are 17 parameters in exciter model “EXAC1” and 17 parameters in PSS model “PSS2A”. The model diagrams of “EXAC1” and “PSS2A” are shown in Figure 4.2 and Figure 6.4 respectively. According to Chapter 4, only the parameters in category 4 are defined as key parameters and need to be identified and estimated.

The key parameters are identified by means of sensitivity analysis. For each parameter, J^* is the mismatch between the original simulation curve of reactive power output and the curve after a step change (10%) is added to that parameter. The computation results are shown in Table 6.7~6.8. The threshold for key parameters is set as 0.1%. K_A, T_E, K_F, T_F, K_D and K_E.are identified as the key parameters of the exciter since their J^* are higher than the threshold. These results almost match the key parameter identification gained by PSS/E response test. TW_1, TW_2, T_7, KS_2, KS_3, T_8, T_9, KS_1, T_1 and T_3 are identified as the key parameter of PSS since their J^* are higher than the threshold (0.1%). So there are totally 16 parameters needed to be optimized as part of this exercise.

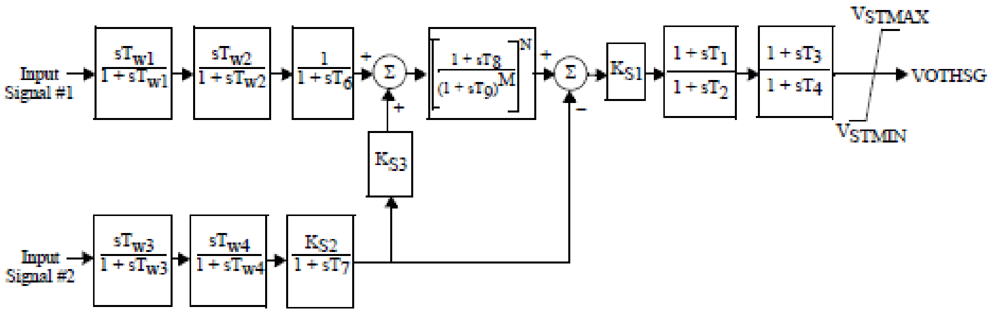


Figure 6.4 Model Diagram of "PSS2A"

Table 6.7 Key Parameter Identification for Exciter (EXAC1) of Generation Unit #3

Parameters	Original	J^*	Parameters	Original	J^*
T_R	0.0000	/	T_F	2.5000	0.32%
T_B	0.0000	/	K_C	0.0500	0.04%
T_C	0.0000	/	K_D	0.4500	0.40%
K_A	600.00	0.15%	K_E	1.0000	0.81%
T_A	0.0000	/	E_1	2.9200	/
V_{RMAX}	7.3100	/	$SE(E_1)$	0.6500	/
V_{RMIN}	-6.5800	/	E_2	3.8900	/
T_E	0.8000	0.23%	$SE(E_2)$	0.8800	/
K_F	0.0350	0.37%			

Table 6.8 Key Parameter Identification for PSS ('PSS2A') of Generation Unit #3

Parameters	Original	J^*	Parameters	Original	J^*
TW_1	10.0000	0.36%	T_9	0.1000	0.58%
TW_2	10.0000	0.36%	KS_1	5.0000	1.09%
T_6	0.0000	/	T_1	0.2000	0.36%
TW_3	10.0000	0.01%	T_2	0.0200	0.07%
TW_4	0.0000	/	T_3	0.2000	0.36%
T_7	10.0000	0.68%	T_4	0.0200	0.07%
KS_2	1.8900	0.85%	VST_{MAX}	0.1000	/
KS_3	1.0000	6.99%	VST_{MIN}	-0.0500	/
T_8	0.5000	0.57%			

6.4 Dynamic Parameter Estimation

The proposed method is applied to tune (optimize) the exciter and PSS parameters of Generation Unit #3 to drive the reactive power simulation results to fit the recorded data.

6.4.1 Optimization Objective

The primary focus of the exercise is to focus on the trend of the curve, not the value of each point of the curve since the trend is more important for dynamic simulation. It is also important to point out that in the absence of exact initial conditions characterizing the power flow conditions at the point of occurrence of the event, it would be impossible to match the values associated with the curve. However, the dynamic response of the model would be independent of the initial conditions and hence the focus on the trend of the curve.

The two curves will be normalized into [0, 1] frame by using two equations:

$$\mathrm{Re}c\_norm = \frac{\mathrm{Re}c - \min(\mathrm{Re}c)}{\max(\mathrm{Re}c) - \min(\mathrm{Re}c)}$$

$$Sim\_norm = \frac{Sim - \min(Sim)}{\max(Sim) - \min(Sim)}$$

The optimized objective is to minimize the normalized mismatch index $J^*(norm)$ which is defined as:

The normalized mismatch index is defined as:

$$J^*(norm) = \frac{(\mathrm{Re}c\_norm - Sim\_norm)' * (\mathrm{Re}c\_norm - Sim\_norm)}{\mathrm{Re}c\_norm' * \mathrm{Re}c\_norm}$$

The normalized mismatch index between original simulation results and recorded data is 34.66%. This index is very big since the trend of the two curves does not match.

6.4.2 Optimization Objective

The reactive power output of Generation Unit #3 during the event can be used to optimize the exciter and PSS parameters of Generation Unit #3. The proposed new algorithm is applied in order to optimize the parameters associated with the exciter and PSS respectively.

Step1. Particle Swarm Optimization (PSO)

The PSO parameters are set as:

M (Population Size) = 160

W (Inertia weight matrix) = I (Identity matrix);

C_1 (constant for swarm optima) = 1;

C_2 (constant for particle optima) = 1.

The range of the key exciter parameters is set in table 6.9

$J^*(x)$ decreases to 0.078 after 16 iterations (418 minutes). The PSO solution is listed in table 6.10. The values depicted in Table 6.10 correspond to the approximate solutions which would now be utilized as the initial guess of the second step.

Table 6.9 the Range of the Key Exciter and PSS Parameters

Model	Parameters	Lower Limit	Upper Limit
Exciter	K_A	300	1000
	T_E	0.3	2
	K_F	0.01	0.1
	T_F	1	6
	K_D	0.15	1.5
	K_E	0.3	3
PSS	TW_1	5	30
	TW_2	5	30
	T_7	5	30
	KS_2	0.6	5
	KS_3	0.3	3
	T_8	0.2	1.2
	T_9	0.05	0.2
	KS_1	2	15
	T_1	0.1	0.6
	T_3	0.1	0.6

Table 6.10 the Step 1 (PSO) Solution of the Key Exciter and PSS Parameters

Model	Parameters	Value
Exciter	K_A	571.13
	T_E	1.49
	K_F	0.015
	T_F	4.57
	K_D	0.45
	K_E	0.83

Table 6.10-*Continued*

PSS	TW_1	9.31
	TW_2	22.61
	T_7	22.70
	KS_2	3.51
	KS_3	0.98
	T_8	0.34
	T_9	0.08
	KS_1	8.91
	T_1	0.31
	T_3	0.29

Step2. Sensitivity Analysis (SA)

Sixteen parameters are still too many for sensitivity analysis since the convergence performance of SA is highly affected by the number of variables characterizing the optimization problem. Actually, some parameters have already been tuned to the optimization points and it is hard to tune them by the sensitivity analysis based on gradient-search. So the key parameter identification is run again to refine the key parameters at the second step. The computation results are shown in Table 6.11. All of the parameter corresponding to J^* lower than 1% are excluded from the key parameters at the second step. The remaining ten key parameters are: T_E, K_D and K_E of the exciter; T_7, KS_2, KS_3, T_9, KS_1, T_1 and T_3 of PSS.

The sensitivity analysis at the second step is run to tune the ten key parameters by three groups:

Group 1: T_E, K_D, K_E and T_7,

Group 2: KS_2, KS_3, T_9 and KS_1

Group3: T_1 and T_3.

The three groups are optimized one by one. The flow chart of the group sensitivity analysis is shown in Figure 6.5.

It requires 10 iterations for the optimization of group 1, 5 iterations for the optimization of group 2 and 6 iterations for the optimization of group 3. The total optimization time for the three groups is 25 minutes.

The final exciter and PSS parameter optimization results are shown in Table 6.12~6.13. The total computation time is 443 minutes. After two step optimization, the normalized mismatch index (J^*) decreases from the original 34.66 % to 5.84 %. The discrepancy still exists after the optimization as shown in Figure 6.6, however, the trend of the two curves matches as shown in Figure 6.7.

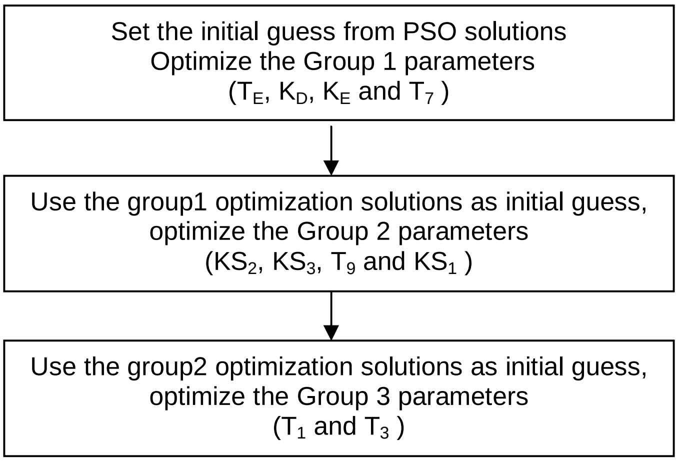


Figure 6.5 Group Sensitivity Analysis at the Second Step

Table 6.11 Key Parameter Identification for the Exciter and PSS of Generation Unit #3 at the Second Step

Model	Parameters	Solution of the First Step	J^*
Exciter	K_A	571.13	0.49%
	T_E	1.49	1.3%
	K_F	0.015	0.37%
	T_F	4.57	0.32%
	K_D	0.45	1.19%
	K_E	0.83	1.9%

Table 6.11-*Continued*

PSS	TW_1	9.31	0.98%
	TW_2	22.61	0.5%
	T_7	22.70	1.65%
	KS_2	3.51	2.64%
	KS_3	0.98	15.24%
	T_8	0.34	0.84%
	T_9	0.08	1.8%
	KS_1	8.91	3.22%
	T_1	0.31	1.43%
	T_3	0.29	1.4%

Table 6.12 Optimization Results for Exciter Parameters of Generation Unit #3

Parameters	Original	Optimization	Parameters	Original	Optimization
T_R	0.0000	/	T_F	2.5000	4.57
T_B	0.0000	/	K_C	0.0500	/
T_C	0.0000	/	K_D	0.4500	0.42
K_A	600.0000	571.13	K_E	1.0000	0.74
T_A	0.0000	/	E_1	2.9200	/
V_{RMAX}	7.3100	/	$SE(E_1)$	0.6500	/
V_{RMIN}	-6.5800	/	E_2	3.8900	/
T_E	0.8000	1.76	$SE(E_2)$	0.8800	/
K_F	0.0350	0.015			

Table 6.13 Optimization Results for PSS Parameters of Generation Unit #3

Parameters	Original	Optimization	Parameters	Original	Optimization
TW_1	10.0000	9.31	T_9	0.1000	0.08
TW_2	10.0000	22.61	KS_1	5.0000	9.25
T_6	0.0000	/	T_1	0.2000	0.24
TW_3	10.0000	/	T_2	0.0200	/
TW_4	0.0000	/	T_3	0.2000	0.36
T_7	10.0000	30.28	T_4	0.0200	/
KS_2	1.8900	3.62	VST_{MAX}	0.1000	/
KS_3	1.0000	0.99	VST_{MIN}	-0.0500	/
T_8	0.5000	0.34			

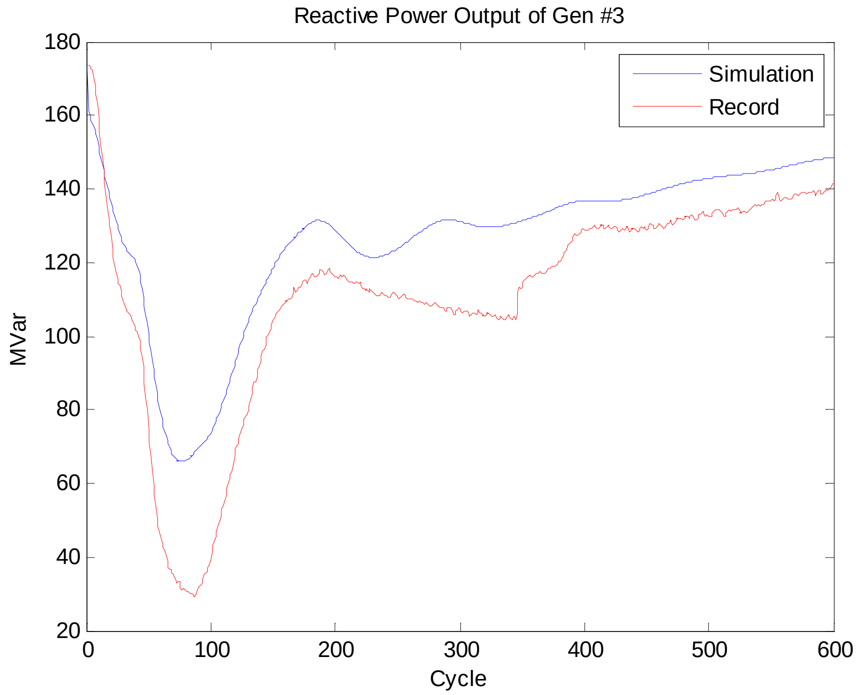


Figure 6.6 Comparison between Simulation Results after Optimization and Recorded data

Figure 6.7 Trend Comparison between Simulation Results after Optimization and Recorded data

CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH

7.1 Conclusions

The accuracy of the dynamic parameters affects the confidence in the power system dynamic simulation which in turn affects the economics and reliability of the power system. Inaccurate dynamic simulation results may lead to conservative estimation of the system transfer limit such as TTC and ATC and cause additional congestion in the power market. It will increase the market clearing price (MCP) or locational marginal price (LMP) and decrease the utilization rate of the transmission network. On the other hand, inaccurate dynamic simulation results may lead to over estimation of the system transfer limit. In the worst case, it may cause local or system level blackout. Therefore, it is very important to ensure the accuracy of the dynamic parameters in the system database for simulation. However, the discrepancy between simulation results and field recording data commonly exists as documented in numerous research papers and reports of fault-event investigation. The area of dynamic parameter estimation is being identified as a potential area of research by the researchers and engineers in the power system area.

The current methods of dynamic parameter estimation usually depend on the initial guess or suffer from tremendous computation time. The approach documented in this dissertation proposes a robust hybrid two-step method for accurate dynamic parameter estimation. The proposed method utilizes a new and intelligent method, PSO, to find an approximate solution of the parameters in the first step. Then the gradient-based sensitivity analysis is applied to find the accurate parameters starting with the approximate solution from the first step. The method is programmed with parallel multi-core programming to utilize the multi-core computer to speed up the method.

The dissertation proposes a key parameter identification approach to reduce the computation burden without compromising on the accuracy of the parameter estimation process. The key parameters of each exciter and governor model in ERCOT system are identified through the response tests in PSSE. The novelty associated with the proposed approach in conjunction with the application of the approach on tuning dynamic parameters for generation facilities in ERCOT will facilitate the wide-spread application of the approach in ERCOT.

The proposed method achieves the target value of the dynamic parameters such as exciters and governors in the assumed test case. In the real case, it successfully tunes the dynamic parameters of the exciter and PSS in a power plant to dramatically decrease the mismatch between simulation results and field recording data following the generator trip event in ERCOT.

7.2 Possible Future Research

7.2.1 Potential Research

The research documented in this dissertation opens doors to numerous similar opportunities, one of them being to develop an integral parameter validation/estimation system in an automated fashion.

Numerous technologies in the IT industry are expected to be adopted in power system through the concept of the "smart grid". One of the most important features of the smart grid technology is self-diagnosis [31]. The proposed approach enables rapid diagnosis and precise solutions to specific grid disruptions or outages. The approach would lend itself well to the estimation of inaccurate dynamic parameters in the system by the parameter estimation program using the data from smart meters including the SCADA/EMS system and IED/DFR devices following disturbance events.

As mentioned in the introductory chapter, the model/parameter validation is a long-term and on-going effort. The proposed approach should be automatically invoked following any fault event. The steady state (power flow) data from SCADA/EMS and filed recording data following the dynamic events could be sent to the computer where the proposed method is located. The proposed method would then utilize the captured data to validate the parameters in the model database. The optimized parameters will then be recommended by the program to decrease the discrepancy when the mismatch index is high. The basic system infrastructure required for the application of the proposed method is shown in Figure 7.1.

7.2.2 Potential Applications

The current installed wind generation capacity in Texas is more than 8,000MW which accounts for about 10% of the total installed generation capacity in ERCOT. According to the Competitive Renewable Energy Zones (CREZ) Transmission Optimization Study, more than 16,000MW of new wind generation will come in service in the near future in ERCOT [32]. At that time, the penetration of wind generation in ERCOT will be around one third of the total installed generation capacity. The potential stability problem caused by the wind farm is serious and typical to ERCOT and is bound to be exacerbated following the advent of the abovementioned wind generation due to the following reasons:

1) Most of farms are located in West Texas with weak transmission network connecting the West to the load centers in the North, South & Houston.
2) Relatively low demand in the West region in ERCOT
3) Most wind generators are induction generators and inherently do not possess voltage ride trough capability without special design.

So it is very important to conduct the dynamic study including frequency and voltage stability. However, the voltage stability results are highly affected by the dynamic load model in the system. As we know, there is no dynamic load model in current model database in ERCOT. It is very urgent for ERCOT to develop a dynamic load model, the lack of which prevents ERCOT from assessing phenomenon such as Fault Induced Voltage Recovery (FIDVR). The presence of FIDVR, as experienced in WECC, may cause voltage recovery following a dynamic event to delay to the extent that numerous wind farms trip leading to large frequency deviations and/or depletion of the responsive reserves. This dissertation focuses on the exciter, governor and PSS parameters. However, the proposed method can be extended to any other dynamic method such as load models and wind generator models since it is a model-independent method.

Recently, the Electrical Power Research Institute (EPRI) conducted a study of the CIM (Common Information Model) for power system dynamic models. The electric utility software vendors are encouraged to exchange XML (Extensible Markup Language) versions of the CIM to demonstrate interoperability of products. Including CIM in the loop will enhance the capability of the proposed parameter estimation process.

In other words, while demonstrated on the dynamic parameters estimation for generation facilities in ERCOT, the proposed approach has wide-spread areas of application which would go a long way in improving the accuracy of the models utilized for power system planning. Furthermore, the approach would bridge the gap that currently exists between system planning models and system operations. The approach would find great application in the dynamic load modeling initiatives currently being undertaken by WECC in order to better fine-tune their system planning models.

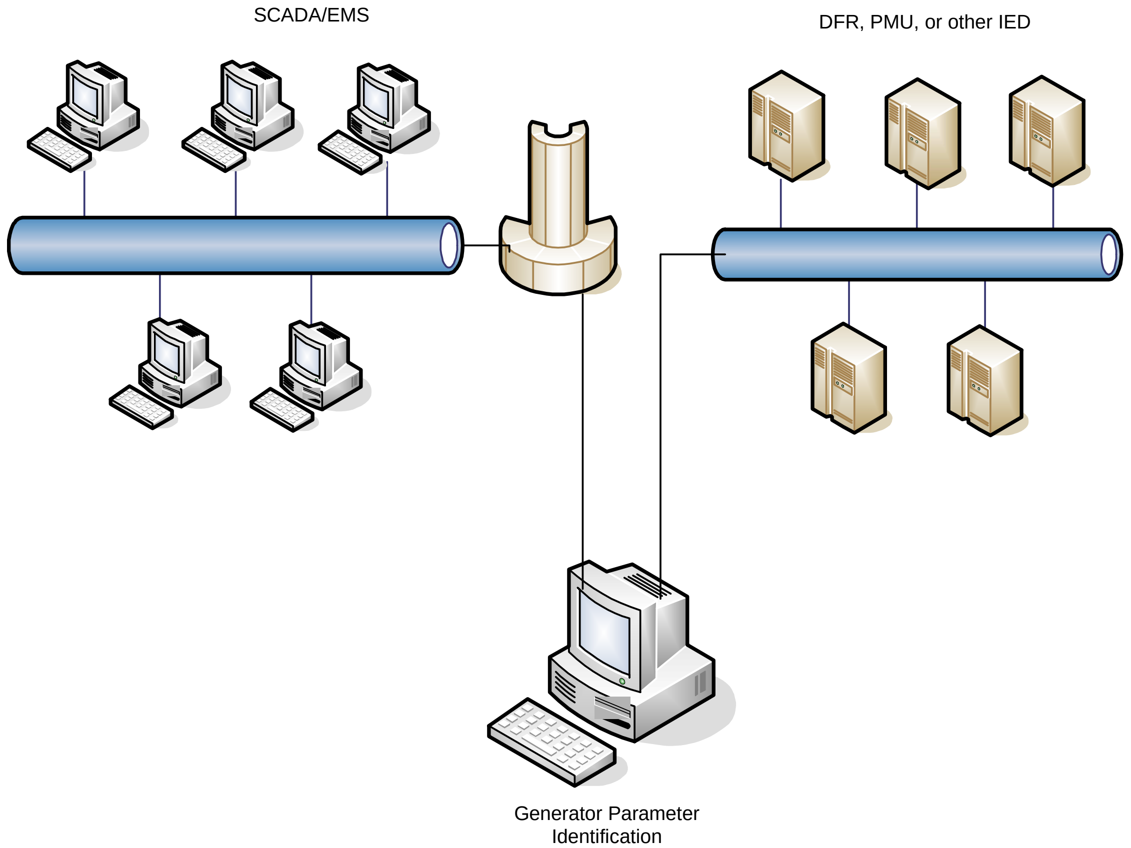


Figure 7.1 Infrastructure of the Application of the Proposed Method in Parameter Validation

APPENDIX A

SOURCE CODE

A.1
Source code of the Proposed Two-Step Method

Main.py

```
import multi_sc_test
import scipy
import scipy.linalg
import scipy.io
import random
import time
import math


####################################
######PSS/E set & Measurement######
####################################

##time issue in simulation
t_0=0
t_pause=0.0667
t_stop=10

Ex_para=1 ##exciter test
##Ex_para=0 ##govenor test

if Ex_para==1:
    exciter_type="EXAC1"
    Meas_ind=3;##Qelec

    ##base case calculation
    txt_file_out='sc_ex_base'
    ex_gov=['EXC',exciter_type]


    Upp_lim=[5.,5.,.15,5.]
    Dwn_lim=[0.0,0.0,0.0,0.0]

    para_ind=[13,8,9,10]

else:
    governor_type='IEESGO'
    Meas_ind=1##Pelec

    txt_file_out='sc_gov_base'
    ex_gov=['GOV',governor_type]

    M=40
    N=4
    Upp_lim=[10,10]
    Dwn_lim=[-10,-10]
```

```
##the studying generator
machine_id=['1','2']
t_=[t_0,t_pause,t_stop]
busnum=[5911,5912]
faultbus=5915
nb_bus=44200
para_ind=[1]
para_ratio=[1]

noise_para_flag=0 ##No noise to neighboring generators

##run the base case and get the measurement
txt_file_out='sc_gov_base_nonoisepara'
chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,[],t_,faul
tbus,nb_bus,txt_file_out,noise_para_flag)
result_data_base=scipy.array(chandata.values ())
result_data_base_nnp_nnm=scipy.copy(result_data_base[Meas_ind])##no noise on paras & no
noise on measures
num_pts_sample=len(result_data_base[0])
Result_sum_base=scipy.dot(result_data_base[Meas_ind],result_data_base[Meas_ind])
Result_sum_base=scipy.sqrt(Result_sum_base)
Result_sum_base_nnp_nnm=Result_sum_base
print "Base Sum no noise paras no noise measures=",Result_sum_base_nnp_nnm
mdict= {'NoNoise_Para_Meas': result_data_base}
scipy.io.savemat('NoNoise_Para_Meas.mat',mdict) ##save the result as mat file

noise_para_flag=0 ##add noise to neighboring generators

if noise_para_flag==1:
   ##run the base case and get the measurement under noise on paras
   txt_file_out='sc_gov_base_noisepara'

chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,[],t_,faul
tbus,nb_bus,txt_file_out,noise_para_flag)
   result_data_base=scipy.array(chandata.values ())
   result_data_base_np_nnm=scipy.copy(result_data_base[Meas_ind])##noise on paras & no
noise on measures
   num_pts_sample=len(result_data_base[0])
   Result_sum_base=scipy.dot(result_data_base[Meas_ind],result_data_base[Meas_ind])
   Result_sum_base=scipy.sqrt(Result_sum_base)
   Result_sum_base_np_nnm=Result_sum_base
   print "Base Sum noise paras no noise measures=",Result_sum_base_np_nnm
   mdict= {'Noise_Para_Nonoise_Meas': result_data_base}
   scipy.io.savemat('Noise_Para_Nonoise_Meas.mat',mdict)##save the result as mat file

noise_para_flag=0 ##No noise to neighboring generators while searching!!!!

Noise_meas_flag=0 ##add no noise to measurements

if Noise_meas_flag==1: ##if we have noise on measurement
   ##add white noise (sd=0.01)
   noise_signal=scipy.zeros([1,num_pts_sample])
```

```
    sd=0.01
    for n_i in range(0,num_pts_sample):
        noise_signal[0][n_i]=random.gauss (0,sd)*result_data_base[Meas_ind][n_i]
    result_data_base[Meas_ind]=result_data_base[Meas_ind]+noise_signal

    Result_sum_base=scipy.dot(result_data_base[Meas_ind],result_data_base[Meas_ind])
    Result_sum_base=scipy.sqrt(Result_sum_base)
    print "Base Sum with noise=",Result_sum_base

    temp=result_data_base[Meas_ind]-result_data_base_nnp_nnm
    noise_sum=scipy.dot(temp,temp)
    noise_sum=scipy.sqrt(noise_sum)
    print "Noise level=",noise_sum/Result_sum_base_nnp_nnm

    mdict= {'Noise_paras_Meas': result_data_base}
    scipy.io.savemat('Noise_paras_Meas.mat',mdict) ##save the result as mat file
####################################
######PSS/E set & Measurement######
####################################


#####################
#########PSO#########
#####################
tic = time.clock()

para_ind=[13,8,9,10]
para_ratio=[]
iter_num=0
txt_file_out=[]

w=1.00
c1=2.00
c2=2.00

N=4
N_level=3
M=int(math.pow(N_level-1,N))

Particle=scipy.zeros((M,3*N+1))

Result_details=scipy.zeros((M*30,N+1))
Result_iter=0

##set the interval for each para
Para_Level=scipy.zeros((N,N_level))
for i in range(0,N):
    for j in range(0,N_level):
        Para_Level[i][j]=Dwn_lim[i]+(Upp_lim[i]-Dwn_lim[i])*j/(N_level-1)

##set the initial group
M=2*M
```

```
Particle=scipy.zeros((M,N*3+1))
n_iter=0
for i_1 in range(0,N_level-1):
    for i_2 in range(0,N_level-1):
        for i_3 in range(0,N_level-1):
            for i_4 in range(0,N_level-1):
                Particle[n_iter,3]=random.uniform (Para_Level[3,i_4],Para_Level[3,i_4+1])
                Particle[n_iter,2]=random.uniform (Para_Level[2,i_3],Para_Level[2,i_3+1])
                Particle[n_iter,1]=random.uniform (Para_Level[1,i_2],Para_Level[1,i_2+1])
                Particle[n_iter,0]=random.uniform (Para_Level[0,i_1],Para_Level[0,i_1+1])
                n_iter=n_iter+1

for i_1 in range(0,N_level-1):
    for i_2 in range(0,N_level-1):
        for i_3 in range(0,N_level-1):
            for i_4 in range(0,N_level-1):
                Particle[n_iter,3]=random.uniform (Para_Level[3,i_4],Para_Level[3,i_4+1])
                Particle[n_iter,2]=random.uniform (Para_Level[2,i_3],Para_Level[2,i_3+1])
                Particle[n_iter,1]=random.uniform (Para_Level[1,i_2],Para_Level[1,i_2+1])
                Particle[n_iter,0]=random.uniform (Para_Level[0,i_1],Para_Level[0,i_1+1])
                n_iter=n_iter+1


print Particle[:,0:4]

for i in range(0,M):

    Particle[i,2*N:3*N]=Particle[i,0:N]


chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,[],Particle[i,0:N],t_,f
aultbus,nb_bus,txt_file_out,noise_para_flag)
    result_data=scipy.array(chandata.values ())
    dev_pt=result_data[Meas_ind]-result_data_base[Meas_ind]
    dev=scipy.dot(dev_pt,dev_pt)
    dev=scipy.sqrt(dev)
    Particle[i,-1]=dev/Result_sum_base

g_best=scipy.zeros(N+1)
g_best[0:N]=Particle[0,0:N]
g_best[-1]=Particle[0,-1]
for i in range(1,M):
    if Particle[i,-1]<g_best[-1]:
        g_best[0:N]=Particle[i,2*N:3*N]
        g_best[-1]=Particle[i,-1]


pso_tolerance=0.02
for iter in range(0,100):
    print iter+1,'th iteration'
    print g_best
```

```
    if g_best[-1]<pso_tolerance:
        break

    temp='Particle'+str(iter)+'.mat'
    mdict= {'Particle':Particle }
    scipy.io.savemat(temp,mdict) ##save the result as mat file

    for i in range(0,M):
        for j in range(N,2*N):
            Particle[i,j]=w*random.random()*Particle[i,j]+c1*random.random()*(Particle[i,j+N]-
Particle[i,j-N])+c2*random.random()*(g_best[j-N]-Particle[i,j-N])
            if abs(Particle[i,j])>(Upp_lim[j-N]-Dwn_lim[j-N])/2:
                Particle[i,j]=(Upp_lim[j-N]-Dwn_lim[j-N])/2*Particle[i,j]/abs(Particle[i,j])

        Particle[i,0:N]=Particle[i,0:N]+Particle[i,N:2*N]

        byd_lmt=0
        for j in range(0,N):
            if Particle[i,j]>Upp_lim[j]:
                Particle[i,j]=(Upp_lim[j]+Dwn_lim[j])*2/3
                byd_lmt=1
            if Particle[i,j]<Dwn_lim[j]:
                Particle[i,j]=(Upp_lim[j]+Dwn_lim[j])*1/3
                byd_lmt=1

        if byd_lmt==0:

chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,Particle[
i,0:N],t_,faultbus,nb_bus,txt_file_out,noise_para_flag)
            result_data=scipy.array(chandata.values ())
            dev_pt=result_data[Meas_ind]-result_data_base[Meas_ind]
            dev=scipy.dot(dev_pt,dev_pt)
            dev=scipy.sqrt(dev)
            y=dev/Result_sum_base

            if y<Particle[i,-1]:
                Particle[i,-1]=y;
                Particle[i,2*N:3*N]=Particle[i,0:N]

    for i in range(0,M):
        if Particle[i,-1]<g_best[-1]:
            g_best[0:N]=Particle[i,2*N:3*N]
            g_best[-1]=Particle[i,-1]


print 'optimized solution is:'
print g_best
toc = time.clock()
print 'The total time is', (toc - tic)/60.0,'min'
#####################
#########PSO#########
```

```
#####################



#####################
#########LS#########
#####################

tic = time.clock()

para_ini=g_best[0:4]

n_para=len(para_ind)
H=scipy.zeros((num_pts_sample,n_para))

para_ratio=[]
para_new=para_ini
iter_num=0
tolerance=0.01
txt_file_out=[]
##main iteration##
conv_flag=1

while conv_flag==1:
    iter_num=iter_num+1

    if iter_num>200:
        print 'Fail to converge'
        break

##    if iter_num==1:
##        txt_file_out='initial_solution'
##    else:
##        txt_file_out=[]


    print 'Iteration No.',iter_num

chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,para_ne
w,t_,faultbus,nb_bus,txt_file_out,noise_para_flag)

    result_data=scipy.array(chandata.values ())
    dev_pt=result_data[Meas_ind]-result_data_base[Meas_ind]
    dev=scipy.dot(dev_pt,dev_pt)
    dev=scipy.sqrt(dev)
    print 'The Dev is%',dev/Result_sum_base


    ##continue to iterate
##    print para_new
    for n_i in range(0,n_para):
          ##find the gradient
```

```
        para_ratio_temp=1.01
        para_new_temp=para_new[:]
        para_new_temp[n_i]=para_new_temp[n_i]*para_ratio_temp
        txt_file_out=[]

chandata=multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,[],para_new_temp,t
_,faultbus,nb_bus,txt_file_out,noise_para_flag)
        result_data=scipy.array(chandata.values ())
        temp_dev=0;
        ##set the sensitivity matrix
        for n_j in range(0,num_pts_sample):
            H[n_j][n_i]=(result_data[Meas_ind]-result_data_base[Meas_ind]-
dev_pt)[n_j]/(para_ratio_temp-1)/para_new[n_i]
    ##solve for delta
    HH=scipy.dot(scipy.transpose(H),H)
    bb=scipy.dot(scipy.transpose(H),-dev_pt)
    delta=scipy.linalg.solve(HH,bb)
    ##make sure each delta is smaller than 10%
    for n_i in range(0,n_para):
        if abs(delta[n_i])>.1*abs(para_new[n_i]):
            delta[n_i]=delta[n_i]/abs(delta[n_i])*abs(para_new[n_i])*.1
    ##set the new paras
    for n_i in range(0,n_para):
        para_new[n_i]=para_new[n_i]+delta[n_i]
    print 'Ke,Te,Kf,Tf=',para_new

    delta_pu=delta/para_new

    ##check convergence
    if max(abs(delta_pu))<tolerance:
        conv_flag=0 ##converge
##          txt_file_out='conv_solution'
##
multi_sc_test.multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,para_new,t_,faultb
us,nb_bus,txt_file_out,noise_para_flag)

toc = time.clock()
print 'The total time is', (toc - tic)/60.0,'min'
#####################
#########LS#########
#####################
```

Multi_sc_test.py

```
def
multi_sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,para_new,t_,faultbus,nb_bus,txt
_file_out,noise_para_flag):


    ##initiate PSS/E
```

```
import os,sys
sys.path.append('C:\\Program Files\\PTI\\PSSE31\\PSSBIN')
os.environ['PATH'] += ';C:\\Program Files\\PTI\\PSSE31\\PSSBIN'
import redirect
redirect.psse2py()
import psspy
psspy.psseinit(80000)
psspy.progress_output(islct=6)
psspy.prompt_output(islct=6)

os.chdir('D:\\ERCOT\\sc')

##open power flow data, convert and solve it
psspy.powerflowmode()
psspy.case(r"""2007FlatStart02282007.sav""")
psspy.conl(1,0,1,[0,0],[0.0,0.0,0.0,0.0])
psspy.bsys(1,0,[0.0,0.0],1,[1],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 44.0, 56.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[4],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[5],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 20.0, 80.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[6],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 59.0, 41.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[7],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[8],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[9],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[11],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 21.0, 79.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[13],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 21.0, 79.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],1,[17],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.bsys(1,0,[0.0,0.0],3,[12,20,21],0,[],0,[],0,[])
psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.conl(1,1,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.conl(1,1,3,[0,0],[ 50.0, 50.0,0.0, 50.0])
psspy.fdns([1,0,1,1,1,1,99,0])
psspy.cong(0)
psspy.ordr(0)
psspy.fact()
psspy.tysl(0)
psspy.tysl(0)

##load snap file without chan
psspy.dynamicsmode(1)
psspy.rstr(r"""2007FlatStart_nochan.snp""")

##    os.chdir('D:\\ERCOT\\sc\\Noise\\temp')
```

```
##    import random
##    temp=random.random();
##    temp=str(temp);
##    temp=temp[2:];
##    temp='ercot'+temp;
##    psspy.progress_output(2,temp,[0,0])
##    psspy.prompt_output(2,temp,[0,0])
##    os.chdir('D:\\ERCOT\\sc')

    import random
    ##add noise to neighboring generators
    if noise_para_flag==1:
        ##add noise to Gen 5903
        ierr, ival = psspy.mdlind(5903,'1','EXC','CON')
        for n_i in range(0,14):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(5903,'1','IEEET1',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(5903,'1','GOV','CON')
        for n_i in range(0,11):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(5903,'1','IEESGO',n_i+1, rval*(1+tmp_randn))
        ##add noise to Gen 48521
        ierr, ival = psspy.mdlind(48521,'1','EXC','CON')
        for n_i in range(0,17):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48521,'1','ESST4B',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(48521,'1','GOV','CON')
        for n_i in range(0,31):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48521,'1','GAST2A',n_i+1, rval*(1+tmp_randn))
        ##add noise to Gen 48522
        ierr, ival = psspy.mdlind(48522,'2','EXC','CON')
        for n_i in range(0,17):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48522,'2','ESST4B',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(48522,'2','GOV','CON')
        for n_i in range(0,31):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48522,'2','GAST2A',n_i+1, rval*(1+tmp_randn))
        ##add noise to Gen 48523(No GOV)
        ierr, ival = psspy.mdlind(48523,'3','EXC','CON')
        for n_i in range(0,17):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48523,'3','ESST4B',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(48523,'3','GOV','CON')
```

```
        ##add noise to Gen 48927
        ierr, ival = psspy.mdlind(48927,'7','EXC','CON')
        for n_i in range(0,16):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48927,'7','IEEEX1',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(48927,'7','GOV','CON')
        for n_i in range(0,11):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48927,'7','IEESGO',n_i+1, rval*(1+tmp_randn))
        ##add noise to Gen 48928
        ierr, ival = psspy.mdlind(48928,'8','EXC','CON')
        for n_i in range(0,16):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48928,'8','IEEEX1',n_i+1, rval*(1+tmp_randn))
        ierr, ival = psspy.mdlind(48928,'8','GOV','CON')
        for n_i in range(0,11):
            ierr, rval = psspy.dsrval('CON', ival+n_i)
            tmp_randn=random.gauss(0,0.1)
            psspy.change_plmod_con(48928,'8','IEESGO',n_i+1, rval*(1+tmp_randn))

    if ex_gov[0]=='EXC':
        for n in range(0,len(busnum)):
            ierr = psspy.machine_array_channel([-1,5,busnum[n]], machine_id[n], "")##Efd
            ierr = psspy.voltage_channel([-1,-1,-1,busnum[n]], "")##voltage magnitude
            ierr = psspy.machine_array_channel([-1,2,busnum[n]], machine_id[n], "")##Pelec
            ierr = psspy.machine_array_channel([-1,3,busnum[n]], machine_id[n], "")##Qelec
        ierr = psspy.voltage_channel([-1,-1,-1,faultbus], "")##voltage magnitude


        exciter_type=ex_gov[1]
        ##change the para_ind parameter of exciter

        if len(para_new)==0:
        ##print 'change with ratio'
            for i in range(0,len(busnum)):
                ierr, ival = psspy.mdlind(busnum[i], machine_id[i], ex_gov[0],'CON')
                for n in range(0,len(para_ind)):
                    ierr, rval = psspy.dsrval('CON', ival+para_ind[n]-1)
                    psspy.change_plmod_con(busnum[i],machine_id[i],exciter_type,para_ind[n],
rval*para_ratio[n])
        else:
        ##print 'change with new value'
            for i in range(0,len(busnum)):
                ierr, ival = psspy.mdlind(busnum[i], machine_id[i], ex_gov[0],'CON')
                for n in range(0,len(para_ind)):
                    ierr, rval = psspy.dsrval('CON', ival+para_ind[n]-1)
                    psspy.change_plmod_con(busnum[i],machine_id[i],exciter_type,para_ind[n],
para_new[n])
```

```
    elif ex_gov[0]=='GOV':

        for n in range(0,len(busnum)):
            ierr = psspy.machine_array_channel([-1,6,busnum[n]], machine_id[n], "")##Pmec
            ierr = psspy.machine_array_channel([-1,2,busnum[n]], machine_id[n], "")##Pelec
            ierr = psspy.machine_array_channel([-1,3,busnum[n]], machine_id[n], "")##Qelec
            ierr = psspy.bus_frequency_channel([-1,busnum[n]], "")##freq

        governor_type=ex_gov[1]

        if len(para_new)==0:
        ##print 'change with ratio'
            for i in range(0,len(busnum)):
                ierr, ival = psspy.mdlind(busnum[i], machine_id[i], ex_gov[0],'CON')
                for n in range(0,len(para_ind)):
                    ierr, rval = psspy.dsrval('CON', ival+para_ind[n]-1)
                    psspy.change_plmod_con(busnum[i],machine_id[i],governor_type,para_ind[n],
rval*para_ratio[n])
        else:
        ##print 'change with new value'
            for i in range(0,len(busnum)):
                ierr, ival = psspy.mdlind(busnum[i], machine_id[i], ex_gov[0],'CON')
                for n in range(0,len(para_ind)):
                    ierr, rval = psspy.dsrval('CON', ival+para_ind[n]-1)
                    psspy.change_plmod_con(busnum[i],machine_id[i],governor_type,para_ind[n],
para_new[n])



    ##set the calculation parameters of dynamical simulation
    idef,rdef =psspy.getbatdefaults()
    psspy.dynamics_solution_params([100,idef,idef,idef,idef,idef,idef,idef],[ 0.4,rdef, 0.004167,
0.016,rdef,rdef, 0.4,rdef],"")

    os.chdir('D:\\ERCOT\\sc\\noise')

    ##run short current simulation
    t_0=t_[0]
    t_pause=t_[1]
    t_stop=t_[2]
    ##Start & Run
    ierr = psspy.strt(0, 'test.out')
    ierr = psspy.run(0, t_0)
    ##add fault
    ierr = psspy.dist_bus_fault(faultbus, 1, 0, [0,-20000000000])
    ierr = psspy.run(0, t_pause)
    ##clear fault
    ierr = psspy.dist_clear_fault(1)
    ierr = psspy.dist_branch_trip(faultbus, nb_bus,'1')
    ierr = psspy.run(0, t_stop)
```

```
##get the result and return it
import dynchanextract
short_title, chanid, chandata = dynchanextract.get('test.out')

##save it as text file
if len(txt_file_out)>0:
    dynchanextract.txtout('test.out', txt_file_out)
return chandata
```

A.2

Source code of Key Parameter Identification Using PSS/E Response Test

Sc_Main.py

```
import sc_test
import math
import time
tic = time.clock()

t_0=0
t_pause=0.0667
t_stop=10

##Ex_para=1 ##exciter
Ex_para=0 ##govenor

if Ex_para==1:
    ##############################
    ######Exciter Para Change#####
    ##############################

    busnum=5911
    exciter_type="EXAC1"##12 paras
    n_para=17
    machine_id='1'

##    busnum=1045
##    exciter_type="EXST1"##12 paras
##    n_para=12
##    machine_id='1'
    ##busnum=48814
    ##exciter_type="URST5T" ##10 paras
    ##n_para=10
    ##machine_id='1'

    ##busnum=948
    ##exciter_type="IEEET1"##14paras
    ##n_para=14
    ##machine_id='1'



    ##base case calculation
    txt_file_out='sc_ex_base'
    para_ind=1
    para_ratio=1
    ex_gov=['EXC',exciter_type]
    t_=[t_0,t_pause,t_stop]
```

```
    faultbus=46020
    nb_bus=3391

chandata=sc_test.sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,t_,faultbus,nb_bus,t
xt_file_out)
    result_data_base=chandata.values ()
    num_pts_sample=len(result_data_base[0])
    temp_val1=0;
    temp_val2=0;
    temp_val3=0;
    for n in range(0,num_pts_sample):
        temp_val1=temp_val1+math.pow(result_data_base[0][n],2)
        temp_val2=temp_val2+math.pow(result_data_base[1][n],2)
        temp_val3=temp_val3+math.pow(result_data_base[2][n],2)
    Efd_sum_base=math.sqrt(temp_val1)  ##Vterm
    Vt_sum_base=math.sqrt(temp_val2)  ##Vterm
    Vfb_sum_base=math.sqrt(temp_val3)  ##Vterm
    print Efd_sum_base
    print Vt_sum_base
    print Vfb_sum_base

    ##deviation
    txt_file_out=[]
    dev_Efd=[]
    dev_Vt=[]##Vterm deviation
    dev_Vfb=[]
    for para_ind in range(1,n_para+1):
        if para_ind>100: pass
        else:
            para_ratio=2
            if para_ind==9:
                txt_file_out='sc_ex_1'
            else:
                txt_file_out=[]

chandata=sc_test.sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,t_,faultbus,nb_bus,t
xt_file_out)
            result_data=chandata.values ()
            temp_dev1=0;
            temp_dev2=0;
            temp_dev3=0;
            for n in range(0,num_pts_sample):
                temp_dev1=temp_dev1+math.pow(result_data[0][n]-result_data_base[0][n],2)
                temp_dev2=temp_dev2+math.pow(result_data[1][n]-result_data_base[1][n],2)
                temp_dev3=temp_dev3+math.pow(result_data[2][n]-result_data_base[2][n],2)
            dev_Efd.append(math.sqrt(temp_dev1)/Efd_sum_base)
            dev_Vt.append(math.sqrt(temp_dev2)/Vt_sum_base)
            dev_Vfb.append(math.sqrt(temp_dev3)/Vfb_sum_base)
    print 'The Dev_Efd is',dev_Efd
    print 'The Dev_Vt is',dev_Vt
    print 'The Dev_Vfb is',dev_Vfb
```

```
    ##############################
    ######Exciter Para Change#####
    ##############################

else:
    ################################
    ######Governor Para Change#####
    ###############################
    busnum=5911
    governor_type="IEESGO" ##11 paras
    n_para=11
    machine_id='1'



    ##base case calculation
    txt_file_out='sc_gov_base'
    para_ind=1
    para_ratio=1
    ex_gov=['GOV',governor_type]
    t_=[t_0,t_pause,t_stop]
    faultbus=5915
    nb_bus=44200

chandata=sc_test.sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,t_,faultbus,nb_bus,t
xt_file_out)
    result_data_base=chandata.values ()
    num_pts_sample=len(result_data_base[0])
    temp_val1=0;
    temp_val2=0;
    temp_val3=0;
    for n in range(0,num_pts_sample):
        temp_val1=temp_val1+math.pow(result_data_base[0][n],2)
        temp_val2=temp_val2+math.pow(result_data_base[1][n],2)
        temp_val3=temp_val3+math.pow(result_data_base[2][n],2)
    Pmech_sum_base=math.sqrt(temp_val1)  ##Vterm
    Spd_sum_base=math.sqrt(temp_val2)  ##Vterm
    Pline_sum_base=math.sqrt(temp_val3)
    print Pmech_sum_base
    print Spd_sum_base
##    print Pline_sum_base

    ##deviation
    txt_file_out=[]
    dev_Pmech=[]
    dev_Spd=[]##Vterm deviation
    dev_Pline=[];
    for para_ind in range(1,n_para+1):
    ##    if para_ind<>11: pass
    ##    else:
        para_ratio=1.1
        if para_ind==1:
```

```
            txt_file_out='sc_gov_1'
        else:
            txt_file_out=[]

chandata=sc_test.sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,t_,faultbus,nb_bus,t
xt_file_out)
        result_data=chandata.values ()
        temp_dev1=0;
        temp_dev2=0;
        temp_dev3=0;
        for n in range(0,num_pts_sample):
            temp_dev1=temp_dev1+math.pow(result_data[0][n]-result_data_base[0][n],2)
            temp_dev2=temp_dev2+math.pow(result_data[1][n]-result_data_base[1][n],2)
            temp_dev3=temp_dev3+math.pow(result_data[2][n]-result_data_base[2][n],2)
        dev_Pmech.append(math.sqrt(temp_dev1))
        dev_Spd.append(math.sqrt(temp_dev2))
        dev_Pline.append(math.sqrt(temp_dev3))
    print 'The Dev_Pmech is',dev_Pmech
    print 'The Dev_Spd is',dev_Spd
##    print 'The Dev_Pline is',dev_Pline


    ################################
    ######Governor Para Change#####
    ###############################

toc = time.clock()
print 'The total time is', (toc - tic)/60,'min'
```

sc_test.py

```
def sc_test(busnum,machine_id,ex_gov,para_ind,para_ratio,t_,faultbus,nb_bus,txt_file_out):


    ##initiate PSS/E
    import os,sys
    sys.path.append('C:\\Program Files\\PTI\\PSSE31\\PSSBIN')
    os.environ['PATH'] += ';C:\\Program Files\\PTI\\PSSE31\\PSSBIN'
    import redirect
    redirect.psse2py()
    import psspy
    psspy.psseinit(80000)
    psspy.progress_output(islct=6)
    psspy.prompt_output(islct=6)

    os.chdir('D:\\ERCOT')

    ##open power flow data, convert and solve it
```

```
    psspy.powerflowmode()
    psspy.case(r"""2007FlatStart02282007.sav""")
    psspy.conl(1,0,1,[0,0],[0.0,0.0,0.0,0.0])
    psspy.bsys(1,0,[0.0,0.0],1,[1],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 44.0, 56.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[4],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[5],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 20.0, 80.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[6],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 59.0, 41.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[7],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[8],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[9],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[11],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 21.0, 79.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[13],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 21.0, 79.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],1,[17],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.bsys(1,0,[0.0,0.0],3,[12,20,21],0,[],0,[],0,[])
    psspy.conl(1,0,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.conl(1,1,2,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.conl(1,1,3,[0,0],[ 50.0, 50.0,0.0, 50.0])
    psspy.fdns([1,0,1,1,1,1,99,0])
    psspy.cong(0)
    psspy.ordr(0)
    psspy.fact()
    psspy.tysl(0)
    psspy.tysl(0)



    ##load snap file without chan
    psspy.dynamicsmode(1)
    psspy.rstr(r"""2007FlatStart_nochan.snp""")

    import random
    temp=random.random();
    temp=str(temp);
    temp=temp[2:];
    temp='ercot'+temp;
    psspy.progress_output(2,temp,[0,0])
    psspy.prompt_output(2,temp,[0,0])

##    ierr =psspy.progress_output(2,r"""2007FlatStart.err""",[0,0])
##    import random
##    if ierr<>0:
##        temp=random.random();
```

```
##      temp=str(temp);
##      psspy.progress_output(2,temp,[0,0])
##
##   ierr =psspy.prompt_output(2,r"""2007FlatStart.err""",[0,0])
##   if ierr<>0:
##      temp=random.random();
##      temp=str(temp);
##      psspy.prompt_output(2,temp,[0,0])

    if ex_gov[0]=='EXC':

        ierr = psspy.machine_array_channel([-1,5,busnum], machine_id, "")##Efd
        ierr = psspy.voltage_channel([-1,-1,-1,busnum], "")##voltage magnitude
        ierr = psspy.voltage_channel([-1,-1,-1,faultbus], "")##voltage magnitude

        exciter_type=ex_gov[1]
        ##change the para_ind parameter of exciter
        ierr, ival = psspy.mdlind(busnum, machine_id, 'EXC','CON')
        ierr, rval = psspy.dsrval('CON', ival+para_ind-1)
        psspy.change_plmod_con(busnum,machine_id,exciter_type,para_ind, rval*para_ratio)
    elif ex_gov[0]=='GOV':

        ierr = psspy.machine_array_channel([-1,6,busnum], machine_id, "")##Pmech
        ierr = psspy.machine_array_channel([-1,7,busnum], machine_id, "")##Speed
##      ierr = psspy.branch_p_channel([-1,-1,-1,busnum,faultbus], '1', "")##P of branch

        governor_type=ex_gov[1]
        ierr, ival = psspy.mdlind(busnum, machine_id, 'GOV','CON')
        ierr, rval = psspy.dsrval('CON', ival+para_ind-1)
        psspy.change_plmod_con(busnum,machine_id,governor_type,para_ind, rval*para_ratio)

##      print rval
##      print rval*para_ratio

    ##set the calculation parameters of dynamical simulation
    idef,rdef =psspy.getbatdefaults()
    psspy.dynamics_solution_params([100,idef,idef,idef,idef,idef,idef,idef],[   0.4,rdef,   0.004167,
0.016,rdef,rdef, 0.4,rdef],"")

    ##run short current simulation
    t_0=t_[0]
    t_pause=t_[1]
    t_stop=t_[2]
    ##Start & Run
    ierr = psspy.strt(0, 'test.out')
    ierr = psspy.run(0, t_0)
    ##add fault
    ierr = psspy.dist_bus_fault(faultbus, 1, 0, [0,-20000000000])
    ierr = psspy.run(0, t_pause)
    ##clear fault
    ierr = psspy.dist_clear_fault(1)
    ierr = psspy.dist_branch_trip(faultbus, nb_bus,'1')
```

```
ierr = psspy.run(0, t_stop)


##get the result and return it
import dynchanextract
short_title, chanid, chandata = dynchanextract.get('test.out')

if len(txt_file_out)>0:
    dynchanextract.txtout('test.out', txt_file_out)
return chandata
```

APPENDIX B

THE DETAILS OF KEY PARAMETER PRE-SCAN OF ERCOT SYSTEM BY PSS/E RESPONSE TEST

x: exciter parameter

ref: original response test curve(E_{fd} or V_{term}) with original parameters

chg: new response test curve with modified parameters (10% modification)

J*(x): mismatch index defined by $J^*(x) = \frac{(ref - chg)^T * (ref - chg)}{ref^T * ref}$

B.1

Key Parameter Identification of Exciter by PSS/E Response Test

1. Identification results of "USRT5T" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.02	0.11%	0.00%	0.00%
T_{C1}	0.2	0.55%	0.02%	0.00%
T_{B1}	0.5	0.67%	0.03%	0.00%
T_{C2}	1	2.30%	0.09%	0.69%
T_{B2}	6.25	2.46%	0.11%	1.81%
K_R	500	0.54%	0.02%	0.00%
V_{RMAX}	4.66	3.05%	0.13%	5.97%
V_{RMIN}	-3.96	0.00%	0.00%	0.00%
T_1	0.003	0.00%	0.00%	0.00%
K_C	0	0.00%	0.00%	0.00%

2. Identification results of "ESST4B" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
KPR	3.5	2.14%	0.11%	0.00%
KIR	3.5	0.41%	0.05%	0.00%
V_{RMAX}	1	0.00%	0.00%	10.05%

V_{RMIN}	-0.87	0.00%	0.00%	0.00%
T_A	0.01	1.53%	0.01%	0.90%
KPM	1	2.17%	0.12%	10.05%
KIM	0	0.00%	0.00%	0.00%
V_{MMAX}	1	0.00%	0.00%	9.71%
V_{MMIN}	-0.87	0.00%	0.00%	0.00%
K_G	0	0.00%	0.00%	0.00%
K_P	5.71	2.20%	0.12%	20.02%
K_I	0	0.00%	0.00%	0.00%
V_{BMAX}	7.14	0.27%	0.04%	0.00%
K_C	0.07	0.02%	0.00%	0.72%
X_L	0	0.00%	0.00%	0.00%
$THETA_P$	0	0.00%	0.00%	0.00%

3. Identification results of “ESAC8B” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_P	50	0.18%	0.05%	0.94%
K_I	11	0.11%	0.02%	0.55%
K_D	20	0.04%	0.01%	0.19%
T_D	0.03	0.01%	0.00%	0.03%
K_A	0.005	0.29%	0.07%	1.40%
T_A	0	0.00%	0.00%	0.00%
V_{RMAX}	16.7	0.00%	0.00%	0.00%
V_{RMIN}	0	0.00%	0.00%	0.00%
T_E	0.45	0.05%	0.01%	0.24%
K_E	1	0.27%	0.06%	1.24%
E_1	3.64	0.43%	0.10%	2.58%
$S_E(E_1)$	2.28	0.37%	0.09%	2.59%
E_2	4.85	0.40%	0.10%	2.78%
$S_E(E_2)$	2.45	0.43%	0.10%	2.74%

4. Identification results of “ESST1A” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.02	0.34%	0.01%	0.00%
V_{IMAX}	0.12	0.00%	0.00%	1.53%
V_{IMIN}	-0.12	0.00%	0.00%	0.00%
T_C	0.02	1.38%	0.01%	0.41%
T_B	0.02	1.39%	0.01%	0.41%
T_{C1}	1	4.50%	0.11%	1.14%
T_{B1}	5	4.68%	0.12%	1.05%
K_A	200	4.58%	0.12%	1.53%
T_A	0.003	0.35%	0.00%	0.13%
V_{AMAX}	7.2	0.00%	0.00%	9.99%
V_{AMIN}	-5.76	0.00%	0.00%	0.00%
V_{RMAX}	7.2	0.00%	0.00%	9.99%
V_{RMIN}	5.76	0.00%	0.00%	0.00%
K_C	0	0.00%	0.00%	0.00%
K_F	0	0.00%	0.00%	0.00%
T_F	1	0.00%	0.00%	0.00%
K_{LR}	0	0.00%	0.00%	0.00%
I_{LR}	0	0.00%	0.00%	0.00%

5. Identification results of “ESAC5A” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.02	0.17%	0.01%	0.00%
K_A	300	0.71%	0.03%	0.00%
T_A	0.08	0.48%	0.01%	0.00%
V_{RMAX}	6.54	0.09%	0.00%	17.80%
V_{RMIN}	0	0.00%	0.00%	0.00%
K_E	1	1.46%	0.11%	13.14%
T_E	0.35	2.44%	0.10%	2.20%
K_F	0.06	2.71%	0.14%	0.00%
T_{F1}	2	2.61%	0.13%	0.00%

T_{F2}	0.198	1.90%	0.07%	0.00%
T_{F3}	0.026	0.49%	0.01%	0.00%
E_1	4.91	1.14%	0.15%	1.40%
$S_E(E_1)$	0.1	1.04%	0.14%	2.50%
E_2	6.54	1.16%	0.15%	0.34%
$S_E(E_2)$	0.12	0.98%	0.14%	0.80%

6. Identification results of "ESAC2A" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
T_B	0	0.00%	0.00%	0.00%
T_C	0	0.00%	0.00%	0.00%
K_A	400	2.34%	0.08%	0.02%
T_A	0.02	0.33%	0.01%	0.02%
V_{AMAX}	68	0.00%	0.00%	0.00%
V_{AMIN}	-68	0.00%	0.00%	0.00%
K_B	1	2.34%	0.08%	0.02%
V_{RMAX}	27.7	0.00%	0.00%	1.85%
V_{RMIN}	-25	0.00%	0.00%	0.00%
T_E	2	2.30%	0.08%	1.72%
V_{FEMAX}	7.1	0.00%	0.00%	14.26%
K_H	0	0.00%	0.00%	0.00%
K_F	0.03	2.88%	0.14%	0.00%
T_F	1	2.72%	0.13%	0.00%
K_C	0.17	0.05%	0.00%	1.22%
K_D	0.4	0.16%	0.01%	3.34%
K_E	1	3.05%	0.15%	10.99%
E_1	6.06	0.00%	0.00%	1.50%
$S_E(E_1)$	0.148	0.00%	0.00%	0.54%
E_2	4.55	0.00%	0.00%	3.87%
$S_E(E_2)$	0.022	0.00%	0.00%	0.22%

7. Identification results of “ESAC1A” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.01	0.40%	0.01%	0.00%
T_B	0	0.00%	0.00%	0.00%
T_C	0	0.00%	0.00%	0.00%
K_A	500	1.14%	0.02%	0.00%
T_A	0.01	0.25%	0.00%	0.00%
V_{AMAX}	6	4.27%	0.09%	8.21%
V_{AMIN}	-3	0.00%	0.00%	0.00%
T_E	0.3	4.03%	0.07%	0.16%
K_F	0.01	2.47%	0.04%	0.00%
T_F	1.5	2.36%	0.04%	0.00%
K_C	0.2	0.31%	0.01%	1.83%
K_D	0.38	0.50%	0.01%	2.75%
K_E	1	3.04%	0.07%	8.26%
E_1	3.01	4.24%	0.08%	5.15%
$S_E(E_1)$	0.523	2.60%	0.05%	3.68%
E_2	4.02	3.08%	0.05%	1.61%
$S_E(E2)$	0.724	2.09%	0.04%	1.00%
V_{RMAX}	6	4.24%	0.09%	8.21%
V_{RMIN}	-3	0.00%	0.00%	0.00%

8. Identification results of “EXPIC1” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	3.97	2.43%	0.13%	0.00%
T_{A1}	1	2.38%	0.12%	0.00%
V_{R1}	1	0.00%	0.00%	9.85%
V_{R2}	-0.87	0.00%	0.00%	0.00%
T_{A2}	0.01	1.57%	0.02%	0.94%
T_{A3}	0	0.00%	0.00%	0.00%
T_{A4}	0	0.00%	0.00%	0.00%

V_{RMAX}	1	0.00%	0.00%	9.50%
V_{RMIN}	-0.87	0.00%	0.00%	0.00%
K_F	0	0.00%	0.00%	0.00%
T_{F1}	1	0.00%	0.00%	0.00%
T_{F2}	1	0.00%	0.00%	0.00%
E_{FDMAX}	6.3	0.00%	0.00%	0.00%
E_{FDMIN}	0	0.00%	0.00%	0.00%
K_E	0	0.00%	0.00%	0.00%
T_E	0	0.00%	0.00%	0.00%
E_1	4.73	0.00%	0.00%	0.00%
$S_E(E_1)$	0	0.00%	0.00%	0.00%
E_2	6.3	0.00%	0.00%	0.00%
$S_E(E_2)$	0	0.00%	0.00%	0.00%
K_P	5.04	2.47%	0.13%	20.06%
K_I	0	0.00%	0.00%	0.00%
K_C	0.13	0.04%	0.00%	1.29%

9. Identification results of “EXST2A” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	120	0.23%	0.01%	0.00%
T_A	0.15	0.19%	0.01%	0.00%
V_{RMAX}	1.2	14.90%	4.19%	15.63%
V_{RMIN}	-1.2	0.00%	0.00%	0.00%
K_E	1.19	13.31%	3.59%	12.51%
T_E	0.5943	1.33%	0.14%	1.23%
K_F	0.02	0.40%	0.02%	0.00%
T_F	0.549	0.36%	0.02%	0.00%
K_P	1.19	14.86%	4.19%	3.06%
K_I	2.5	0.00%	0.00%	10.95%
K_C	0	0.00%	0.00%	0.00%
E_{FDMAX}	3.534	0.00%	0.00%	15.96%

10. Identification results of “EXST3” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
V_{IMAX}	99	0.00%	0.00%	0.00%
V_{IMIN}	-99	0.00%	0.00%	0.00%
K_J	200	2.18%	0.12%	0.00%
T_C	1	2.13%	0.10%	0.00%
T_B	10	2.23%	0.12%	0.00%
K_A	8	0.44%	0.01%	0.00%
T_A	0.4	0.43%	0.01%	0.00%
V_{RMAX}	1	0.00%	0.00%	5.02%
V_{RMIN}	0	0.00%	0.00%	0.00%
K_G	1	2.13%	0.12%	0.00%
K_P	4.4	0.48%	0.01%	5.83%
K_I	5.54	0.00%	0.00%	0.34%
E_{FDMAX}	5.25	0.00%	0.00%	19.29%
K_C	1.37	0.03%	0.00%	0.67%
X_L	0.44	0.00%	0.00%	1.17%
$V_{GMA}X$	5.74	0.00%	0.00%	0.00%
Theta P(degrees)	0	0.00%	0.00%	0.00%

11. Identification results of “EXST1” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.2	2.55%	0.14%	0.00%
V_{IMAX}	0.17	0.00%	0.00%	0.00%
V_{IMIN}	-0.17	0.00%	0.00%	0.00%
T_C	0.4	0.63%	0.02%	0.00%
T_B	0.5	0.65%	0.02%	0.00%
K_A	320	0.74%	0.03%	0.00%
T_A	0.02	0.19%	0.00%	0.00%
V_{RMAX}	7.99	0.00%	0.00%	20.08%

V_{RMIN}	-3	0.00%	0.00%	0.00%
K_C	0.214	0.00%	0.00%	3.14%
K_F	0.03	5.53%	0.22%	0.00%
$T_F(>0)$	1	5.26%	0.21%	0.00%

12. Identification results of "EXDC2" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	200	1.19%	0.03%	0.11%
T_A	0.3	1.15%	0.02%	0.10%
T_B	1	1.18%	0.03%	0.10%
T_C	1	1.19%	0.03%	0.11%
V_{RMAX}	5.71	0.00%	0.00%	11.08%
V_{RMIN}	0	0.00%	0.00%	0.00%
K_E	1	1.07%	0.03%	9.44%
T_E	0.02	0.50%	0.01%	0.26%
K_F	0.044	2.13%	0.10%	0.00%
T_{F1}	1	2.06%	0.09%	0.00%
Switch	0	0.00%	0.00%	0.00%
E_1	4.28	0.00%	0.00%	4.53%
$S_E(E_1)$	0.03	0.00%	0.00%	0.28%
E_2	5.71	0.00%	0.00%	8.81%
$S_E(E_2)$	0.3	0.00%	0.00%	1.06%

13. Identification results of "EXAC4" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.002	0.00%	0.00%	0.00%
V_{IMAX}	4	0.00%	0.00%	0.00%
V_{IMIN}	-4	0.00%	0.00%	0.00%
T_C	0.476	3.44%	0.13%	0.62%
T_B	0.111	1.97%	0.06%	0.56%

K_A	48.776	3.76%	0.17%	0.65%
T_A	0.793	3.68%	0.14%	0.62%
V_{RMAX}	7.95	0.00%	0.00%	19.94%
V_{RMIN}	-4.06	0.00%	0.00%	0.00%
K_C	0	0.00%	0.00%	0.00%

14. Identification results of "EXAC3" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.01	0.20%	0.02%	0.00%
T_B	1	1.01%	0.09%	0.00%
T_C	1	1.06%	0.09%	0.00%
K_A	26.16	0.93%	0.07%	0.00%
T_A	0.013	0.09%	0.01%	0.00%
V_A	1	2.13%	0.16%	9.33%
V_{AMIN}	-0.95	0.00%	0.00%	0.00%
T_E	2.3	1.99%	0.13%	3.43%
KLV	0.187	0.00%	0.00%	0.00%
K_R	5.17	2.75%	0.19%	9.33%
K_F	0.0715	1.64%	0.14%	0.00%
T_F	1.2	1.53%	0.13%	0.00%
K_N	0.025	0.00%	0.00%	0.00%
E_{FDN}	1.714	1.82%	0.14%	0.00%
K_C	0.123	0.03%	0.00%	1.63%
K_D	0.805	0.29%	0.02%	1.25%
K_E	1	0.42%	0.03%	1.89%
VLV	0.492	0.00%	0.00%	0.00%
E_1	5.17	0.00%	0.00%	4.65%
$S_E(E_1)$	0.26	0.00%	0.00%	0.35%
E_2	6.89	0.00%	0.00%	23.37%
$S_E(E_2)$	2.27	0.00%	0.00%	5.29%

15. Identification results of “EXAC2” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.1	0.71%	0.05%	0.00%
T_B	1	0.18%	0.02%	0.00%
T_C	1	0.18%	0.01%	0.00%
K_A	1000	0.07%	0.00%	0.00%
T_A	0.01	0.05%	0.00%	0.00%
$V_{A\ MAX}$	8.6	1.72%	0.05%	12.48%
$V_{A\ MIN}$	-8.6	0.00%	0.00%	0.00%
K_B	1	1.75%	0.05%	13.21%
$V_{R\ MAX}$	13.9	0.00%	0.00%	0.00%
$V_{R\ MIN}$	-13.9	0.00%	0.00%	0.00%
T_E	0.66	1.23%	0.03%	1.51%
K_L	4	0.00%	0.00%	2.47%
K_H	0	0.00%	0.00%	0.00%
K_F	0.05	2.70%	0.14%	0.00%
T_F	1	2.49%	0.12%	0.00%
K_C	0.1	0.02%	0.00%	0.62%
K_D	0.8	0.27%	0.03%	7.90%
K_E	1	2.62%	0.13%	11.14%
VLR	10.79	0.00%	0.00%	10.00%
E_1	4.73	0.05%	0.00%	1.87%
$S_E(E_1)$	0.02	0.05%	0.00%	0.26%
E_2	3.55	0.08%	0.00%	0.14%
$S_E(E_2)$	0.01	0.05%	0.00%	0.03%

16. Identification results of “EXAC1” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
T_B	0	0.00%	0.00%	0.00%
T_C	0.38	0.00%	0.00%	0.00%
K_A	400	0.32%	0.02%	0.00%

T_A	0.02	0.12%	0.00%	0.00%
$V_{R\ MAX}$	7.3	2.97%	0.12%	16.10%
$V_{R\ MIN}$	-6.6	0.00%	0.00%	0.00%
T_E	0.8	1.74%	0.06%	0.35%
K_F	0.03	1.54%	0.07%	0.00%
T_F	1	1.41%	0.07%	0.00%
K_C	0.5	0.14%	0.01%	1.93%
K_D	0.2	0.22%	0.01%	3.07%
K_E	1	1.41%	0.08%	8.68%
E_1	4.95	2.56%	0.12%	6.96%
$S_E(E_1)$	0.03	1.24%	0.07%	4.98%
E_2	6.6	2.44%	0.11%	1.04%
$S_E(E_2)$	0.1	3.08%	0.17%	0.62%

17. Identification results of “IEEEX1” model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	175	0.75%	0.04%	0.00%
T_A	0.05	0.45%	0.02%	0.00%
T_B	0	0.00%	0.00%	0.00%
T_C	0	0.00%	0.00%	0.00%
V_{RMAX}	3.5	0.95%	0.05%	2.92%
V_{RMIN}	-3.5	0.00%	0.00%	0.00%
K_E	-0.17	0.07%	0.01%	0.50%
T_E	0.95	1.58%	0.07%	0.73%
K_F	0.07	2.02%	0.14%	0.00%
T_{F1}	1	1.88%	0.12%	0.00%
SWITCH	0	0.00%	0.00%	0.00%
E_1	2.9	0.02%	0.00%	2.95%
$S_E(E_1)$	0.22	0.00%	0.00%	0.27%
E_2	3.5	0.00%	0.00%	13.35%
$S_E(E_2)$	0.95	0.00%	0.00%	3.66%

18. Identification results of “SEXS” model

Name	Value	J^* Open-circuit response Test E_{fd}	J^* Open-circuit response Test V_{term}	J^* Response Ratio Test E_{fd}
T_A	0.1	1.53%	0.14%	0.22%
T_B	10	0.47%	0.08%	0.00%
K	100	1.61%	0.17%	0.22%
T_E	0.1	0.68%	0.04%	0.20%
E_{MIN}	0	0.00%	0.00%	0.00%
E_{MAX}	3	0.00%	0.00%	19.96%

19. Identification results of “IEEET4” model

Name	Value	J^* Open-circuit response Test E_{fd}	J^* Open-circuit response Test V_{term}	J^* Response Ratio Test E_{fd}
K_R	0.03	1.20%	0.17%	0.00%
T_{RH}	20	3.16%	0.70%	0.00%
K_V	0.05	19.84%	4.95%	0.00%
V_{RMAX}	4.28	0.00%	0.00%	8.27%
V_{RMIN}	0	0.00%	0.00%	0.00%
T_E	0.76	2.56%	0.59%	2.35%
K_E	0.1	0.65%	0.10%	0.96%
E_1	2.543	1.74%	0.27%	15.24%
$S_E(E_1)$	0.105	0.24%	0.02%	3.79%
E_2	3.391	0.79%	0.11%	28.56%
$S_E(E_2)$	0.262	0.33%	0.03%	10.64%

20. Identification results of “IEEET3” model

Name	Value	J^* Open-circuit response Test E_{fd}	J^* Open-circuit response Test V_{term}	J^* Response Ratio Test E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	120	0.89%	0.04%	0.00%
T_A	0.15	0.84%	0.04%	0.00%

V_{RMAX}	1.2	3.78%	0.23%	2.42%
V_{RMIN}	-1.2	0.00%	0.00%	0.00%
T_E	0.5	2.77%	0.14%	1.11%
K_F	0.02	1.88%	0.10%	0.00%
T_F	0.43	1.57%	0.09%	0.00%
K_P	1.19	5.84%	0.40%	2.26%
K_I	2.426	0.00%	0.00%	9.86%
V_{BMAX}	3.71	0.00%	0.00%	15.34%
K_E	1	4.32%	0.28%	19.46%

21. Identification results of "IEEET2" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0.022	0.87%	0.05%	0.00%
K_A	500	0.78%	0.04%	0.00%
T_A	0.1	2.92%	0.19%	0.00%
V_{RMAX}	9	3.73%	0.26%	6.06%
V_{RMIN}	0	0.00%	0.00%	0.00%
K_E	1	13.76%	0.99%	2.94%
T_E	1.2	7.60%	0.37%	0.77%
K_F	0.017	9.49%	0.69%	0.00%
T_{F1}	0.6	6.62%	0.52%	0.00%
T_{F2}	1.2	7.84%	0.59%	0.00%
E_1	5.66	8.20%	0.62%	26.28%
$S_E(E_1)$	2.44	1.41%	0.10%	8.30%
E_2	7.57	4.15%	0.29%	13.49%
$S_E(E_2)$	5.24	2.30%	0.16%	5.82%

22. Identification results of "IEEET1" model

Name	Value	J^* Open-circuit response Test		J^* Response Ratio Test
		E_{fd}	V_{term}	E_{fd}
T_R	0	0.00%	0.00%	0.00%
K_A	50	0.35%	0.03%	0.00%
T_A	0.02	0.05%	0.00%	0.00%
V_{Rmax}	1	1.01%	0.08%	4.03%

V_{Rmin}	-1	0.00%	0.00%	0.00%
K_E	0	0.00%	0.00%	0.00%
T_E	0.614	1.21%	0.09%	1.42%
K_F	0.098	1.49%	0.15%	0.00%
T_F	1.137	1.31%	0.13%	0.00%
SWITCH	0	0.00%	0.00%	0.00%
E_1	2.749	0.07%	0.01%	4.48%
$S_E(E_1)$	0.084	0.00%	0.00%	0.29%
E_2	3.666	0.00%	0.00%	11.66%
$S_E(E_2)$	0.3274	0.00%	0.00%	3.49%

B.2

Key Parameter Identification of Governor by PSS/E Response Test

1. Identification results of “GGOV1” model

Name	Value	J^*	
		SPD	P_{MEC}
R	0.04	0.00%	0.00%
T_{PELEC}	1	0.00%	0.00%
M_{AXERR}	0.05	0.00%	0.00%
M_{INERR}	-0.05	0.00%	0.00%
K_{PGOV}	10	0.00%	0.00%
K_{IGOV}	2	0.00%	0.00%
K_{DGOV}	0	0.00%	0.00%
T_{DGOV}	1	0.00%	0.00%
V_{MAX}	1	40.46%	9.29%
V_{MIN}	0.15	0.00%	0.00%
T_{ACT}	0.5	0.24%	0.11%
K_{TURB}	1.5	20.81%	5.18%
W_{FNL}	0.2	0.92%	0.23%
T_B	0.5	0.78%	0.16%
T_C	0	0.00%	0.00%
T_{ENG}	0	0.00%	0.00%
T_{FLOAD}	3	1.99%	0.47%
K_{PLOAD}	1	0.85%	0.40%
K_{ILOAD}	0.2	0.66%	0.16%
L_{DREF}	1	91.28%	20.36%
D_M	0	0.00%	0.00%
R_{OPEN}	0.1	0.00%	0.00%
R_{CLOSE}	-0.1	0.00%	0.00%
K_{IMW}	0	0.00%	0.00%
A_{SET}	0.01	0.00%	0.00%
K_A	10	0.00%	0.00%
T_A	0.1	0.00%	0.00%
T_{RATE}	75	88.51%	19.71%
D_B	0	0.00%	0.00%

$T_{DLEADTC}$	4	0.00%	0.00%
T_{DLAGTC}	5	0.00%	0.00%
$M_{AXRATELOADLIMITINCR}$	99	0.00%	0.00%
$M_{AXREATELOADLIMITDEC}$	-99	0.00%	0.00%

2. Identification results of “GAST2A” model

Name	Value	J^*	
		SPD	P_{MEC}
W	20	21.13%	0.86%
X	0	0.00%	0.00%
Y	0.02	0.22%	0.03%
Z	1	20.51%	0.91%
E_{TD}	0.04	0.00%	0.00%
T_{CD}	0.2	2.04%	0.24%
T_{RATE}	80	321.00%	6.96%
T	0.0625	0.67%	0.08%
$M_{AXLIMIT}$	1	41.12%	0.95%
MINLIMIT	0.15	0.00%	0.00%
E_{CR}	0.01	0.23%	0.03%
K_3	0.77	55.78%	1.46%
A	1	203.95%	4.52%
B	0.2	2.04%	0.24%
C	1	95.54%	2.35%
T_F	0.2	2.04%	0.24%
K_F	0	0.00%	0.00%
K_5	0.2	0.00%	0.00%
K_4	0.8	0.00%	0.00%
T_3	15	0.00%	0.00%
T_4	2.5	0.00%	0.00%
T_t	1650	0.00%	0.00%
T_5	3.3	0.00%	0.00%
AF_1	1124	0.00%	0.00%
BF_1	575	0.00%	0.00%
AF_2	0.201	0.21%	0.01%
BF_2	1.3	203.96%	4.52%
CF_2	0.5	0.54%	0.02%

T_R	1100	54.17%	1.67%
K_6	0.23	0.31%	0.01%
T_C	1100	54.17%	1.67%

3. Identification results of "TGOV3" model

Name	Value	J^*	
		SPD	P_{MEC}
K	20	0.49%	0.05%
T_1	0.18	0.29%	0.03%
T_2	0.03	0.08%	0.01%
T_3	0.04	0.11%	0.01%
U_0	0.4	0.02%	0.00%
U_c	-0.4	0.00%	0.00%
P_{MAX}	0.9	129.48%	8.33%
P_{MIN}	0.27	0.00%	0.00%
T_4	0.25	0.79%	0.06%
K_1	0.3	54.81%	3.68%
T_5	9.5	6.19%	0.44%
K_2	0.266	42.59%	2.93%
T_6	0.45	0.47%	0.03%
K_3	0.434	68.44%	4.56%
T_A	0.25	0.00%	0.00%
T_B	5.25	0.00%	0.00%
T_C	28.25	0.00%	0.00%
P_{RMAX}	1	21.17%	1.61%

4. Identification results of "IEEEG2" model

Name	Value	J^*	
		SPD	P_{MEC}
K	0	0.00%	0.00%
T_1	0.063	0.00%	0.00%
T_2	0.45	0.00%	0.00%
T_3	0.2	0.00%	0.00%
P_{MAX}	0.62	43.38%	18.28%
P_{MIN}	0	0.00%	0.00%
T_4	1	3.38%	1.88%

5. Identification results of “IEEG1” model

Name	Value	J^*	
		SPD	P_{MEC}
K	20	0.00%	0.00%
T_1	0	0.00%	0.00%
T_2	0	0.00%	0.00%
T_3	0.15	0.00%	0.00%
U_0	1	0.00%	0.00%
Uc	-1	0.00%	0.00%
P_{MAX}	0.75	101.68%	19.53%
P_{MIN}	0.24	0.00%	0.00%
T_4	0.05	0.05%	0.05%
K_1	1	102.05%	19.60%
K_2	0	0.00%	0.00%
T_5	0	0.00%	0.00%
K_3	0	0.00%	0.00%
K_4	0	0.00%	0.00%
T_6	0	0.00%	0.00%
K_5	0	0.00%	0.00%
K_6	0	0.00%	0.00%
T_7	0	0.00%	0.00%
K_7	0	0.00%	0.00%
K_8	0	0.00%	0.00%

6. Identification results of “IEESGO” model

Name	Value	J^*	
		SPD	P_{MEC}
T_1	0.1	1.08%	0.06%
T_2	0	0.00%	0.00%
T_3	0.2	2.04%	0.11%
T_4	0.25	2.94%	0.15%
T_5	7	12.32%	0.55%
T_6	0.5	1.87%	0.10%
K_1	20.4	11.73%	0.34%
K_2	0.7	44.26%	2.01%
K_3	0.7	2.10%	0.11%

P_{MAX}	1	223.59%	5.91%
P_{MIN}	0	0.00%	0.00%

7. Identification results of "HYGOV" model

Name	Value	J^*	
		SPD	P_{MEC}
R	0.05	3.07%	0.23%
r	0.5	22.55%	1.75%
T_r	6	7.96%	0.76%
T_f	0.05	0.36%	0.05%
T_g	0.5	3.60%	0.49%
V_{ELM}	0.167	0.00%	0.00%
G_{MAX}	1	31.99%	2.72%
G_{MIN}	0	0.00%	0.00%
T_W	2	14.03%	1.75%
A_t	1.2	38.15%	3.03%
D_{turb}	0.2	1.41%	0.15%
a_{NL}	0.08	0.45%	0.06%

8. Identification results of "GAST" model

Name	Value	J^*	
		SPD	P_{MEC}
R	0.05	19.02%	0.62%
T_1	0.4	3.10%	0.30%
T_2	0.25	2.11%	0.21%
T_3	3	0.00%	0.00%
$A_{mTempLdLim}$	1.5	0.00%	0.00%
K_T	2	0.00%	0.00%
V_{MAX}	1.5	0.00%	0.00%
V_{MIN}	0.23	0.00%	0.00%
D_{turb}	0	0.00%	0.00%

9. Identification results of "TGOV1" model

Name	Value	J^*	
		SPD	P_{MEC}

R	0.05	9.37%	0.05%
T_1	0.869	2.23%	0.03%
V_{MAX}	1	44.39%	0.18%
V_{MIN}	0.02	0.00%	0.00%
T_2	0	0.00%	0.00%
T_3	0.5	1.58%	0.02%
D_t	20	11.42%	0.14%

REFERENCES

[1] Subir Ray, "Electrical Power System: Concept, Theory and Practice", 2007.

[2] Prabha Kundur, "Power System Stability and Control", 1994.

[3] Hans B. Puttgen, Donald R. Volzka, Melvin I. Olken, "Restructuring and Reregulation of the U.S. Electric Utility Industry", IEEE Power Engineering Review, pp. 8-10, vol. 21, no. 2, 2001.

[4] Dmitry N. Kosterev, Member Carson W. Taylor, William A. Mittelstadt, "Model Validation for the August 10,1996 WSCC System Outage", Vol. 14, No. 3, August 1999.

[5] Baj Agrawal, Dmitry Kosterev, "Model Validation Studies for a Disturbance Event That Occurred on June 14 2004 in the Western Interconnection", Power Engineering Society General Meeting. IEEE, June 2007

[6] M. R. AlRashidi, M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power systems," IEEE Trans. Evolutionary Computation, 2006.

[7] Tzong-yih Guo , Chuan-Shen Liu , Chiang-Tsung Huang, "Identification of Excitation System Model Parameters via Finalization Field Tests", IEE 2nd International Conference on Advances in Power System Control, Operation and Management, December 1993, Hong Kcng

[8] M. Rasouli and M. Karrari, "Nonlinear identification of a brushless excitation system via field tests," IEEE Trans. Energy Convers., vol. 19, no. 4, pp. 733–740, Dec. 2004.

[9] Meng Shen, Venkatasubramanian, V., Abi-Samra, N., Sobajic, D., "A new framework for estimation of generator dynamic parameters", IEEE Trans. Power Syst., vol. 15, no. 2, pp. 756 – 763, May 2000.

[10] J. J. Sanchez-Gasca, C. J. Bridenbaugh, C. E. J. Bowler, and J. S. Edmonds, "Trajectory sensitivity based identification of synchronous generator and excitation system parameters",

IEEE Trans. Power Syst., vol. 3, no. 4, pp. 1814–1822, Nov. 1988.

[11]J.-C. Wang, H.-D. Chiang, C.-T. Huang, Y.-T. Chen, C.-L. Chang, and C.-Y. Huang, "Identification of excitation system models based on on-line digital measurements," IEEE Trans. Power Syst., vol. 10, no. 3, pp. 1286–1293, Aug. 1995.

[12] Yuan-Yih Hsu, Chuan-Sheng Liu, Tain-Syh Luor, Chung-Liang Chang, Ah-Shing Liu, Yung-Tien Chen, Chiang-Tsung Huang, "Experience with the identification and tuning of excitation system parameters at the second nuclear power plant of Taiwan power company", IEEE Trans. Power Syst., vol. 11, no. 2, pp. 747 – 753, May 1996.

[13] Ali Abur, Antonio Gomez Exposito, "Power System State Estimation", ISBN: 0-8247-5570-7, Marcel Dekker, Inc. 2004.

[14] Q. F. Liao, D. C. Liu, L. M. Ying, X. Cui, Y. Li and W. T. He, "Parameter Identification of Excitation Systems Based on Hopfield Neural Network", 2006 International Conference on Power System Technology

[15] Meng shen, "New framework for automatic parameter estimation of power system models", Ph.D. dissertation, Washington State University, WA, USA, 2000.

[16] Puma, J.Q., Colome, D.G., "Parameters identification of excitation system models using genetic algorithms", Generation, Transmission & Distribution, IET, Vol. 2, no. 3, pp. 456-467, May 2008

[17] Dewi Jones, "Estimation of Power System Parameters", IEEE TRANSACTIONS ON POWER SYSTEMS", VOL. 19, NO. 4, NOVEMBER 2004.

[18] Zhenyu Huang, Bo Yang, Dmitry Kosterev, "Benchmarking of Planning Models Using Recorded Dynamics", IEEE Power and Energy Society Power Systems Conference and Exposition 2009, Seattle, WA.

[19] Greg Welch, and Gary Bishop, "An introduction to the Kalman Filter," TR95-041, Department of Computer Science, University of North Carolina at Chapel Hill, April 2004.

[20] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proc. IEEE Int. Conf. Neural Netw, vol. 4, pp. 1942–1948, 1995.

[21] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proc. 6th Int. Symp. Micro Machine Human Science, pp. 39–43, 1995.

[22] Picardi, C., Rogano, N., "Parameter identification of induction motor based on particle swarm optimization", Power Electronics, Electrical Drives, Automation and Motion, 2006. SPEEDAM 2006. International Symposium on, pp. 968 – 973, 2006

[23] Jingping Yang, Jing Zhang, Wulue Pan, "Dynamic Equivalents of Power System Based on Extended Two Particle Swarm Optimization", Natural Computation, 2007. ICNC 2007. Third International Conference on, vol. 5, pp. 609 – 613, 2007

[24] Yuhui Shi and Russell Eberhart, "A Modified Particle Swarm Optimizer", Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence.

[25] PSS/E 31.0 Program Operation Manual: Volume II, pp. 378, PTI, Dec. 2007

[26] PSS/E 31.0 Program Operation Manual: Volume II, pp. 364, PTI, Dec. 2007

[27] Chapter5, "Dynamic Simulation Activity Descriptions", PSS/E 31.0 Program Operation Manual: Volume I, PTI, Dec. 2007

[28] PSS/E 31.0 Program Operation Manual: Volume II, pp. 466, PTI, Dec. 2007

[29] Parallel Python Software Package, Vitalii Vanovschi, 2005-2008, http://www.parallelpython.com/

[30] PSS/E 31.0 Program Operation Manual: Volume II, pp. 299, PTI, Dec. 2007

[31] National Energy Technology Laboratory for the U.S. Department of Energy, "A Systems View of the Modern Grid", http://www.netl.doe.gov/moderngrid/opportunity/vision_technologies.html

[32] Competitive Renewable Energy Zones (CREZ) Transmission Optimization Study, System Planning of ERCOT, April 2 2008.

BIOGRAPHICAL INFORMATION

Yunzhi Cheng received Bachelor's and Master's degrees in Electrical Engineering department from Shanghai Jiaotong University, Shanghai, China, in 2000 and 2003. He studied power system restoration after blackout in his Master thesis. He worked in East China Electrical Power Design Institute (ECEPDI) in Shanghai during 2003-2006. He focused on power system planning for East China Power System, the second largest regional power grid in the world. His work experience also includes power plant (coal, gas, and nuclear) interconnection, 500kV/220kV short-circuit current limitation and Three Gorges (25.5GW) power distribution.

In 2006, Yunzhi Cheng came to Energy System Research Center (ESRC) at University of Texas at Arlington to pursue Ph D degree in Electrical Engineering. His areas of interests are power system dynamic parameter estimation, state estimation, and power market.