A VARIABLE DIMENSION GAUSS-NEWTON METHOD FOR ILL-CONDITIONED PARAMETER ESTIMATION WITH APPLICATION TO A SYNCHRONOUS GENERATOR

by

Jorge L. Huaman-Qquellon

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
In
ELECTRICAL ENGINEERING

UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
2005

Approved by:

________________________________ ____________________
Agustín Irizarry-Rivera, PhD Date
Member, Graduate Committee

________________________________ ____________________
Efraín O'Neill-Carrillo, PhD Date
Member, Graduate Committee

________________________________ ____________________
Miguel Vélez-Reyes, PhD Date
President, Graduate Committee

________________________________ ____________________
José A. Colucci-Ríos, PhD Date
Representative of Graduate Studies

________________________________ ____________________
Isidoro Couvertier, PhD Date
Chairperson of the Department

ABSTRACT

This thesis presents a variable dimension Gauss-Newton (VDGN) parameter estimation algorithm that can be used for fault detection, and diagnosis of a synchronous generator. The algorithm is derived as an extension of the variable dimension Newton Raphson algorithm proposed to solve nonlinear systems of equations. We study the conditioning of the parameter estimation problem for a linearized small-signal model of the synchronous generator using local sensitivity analysis. The conditioning analysis is performed on simulated data and experimental data for the FC5HP synchronous generator located at the Four Corners Generating Station of the Arizona Public Service Company (APS), rated at 483 MVA. Results demonstrated that local sensitivity analysis is an effective tool to diagnose ill-conditioning. The developed VDGN algorithm is shown to be a robust method for ill-conditioned parameter estimation and its performance is compared with the subset selection method. Results using experimental and real data for the synchronous machine parameter estimation problem showed that the VDGN algorithm computes better parameter estimates than the subset selection method and requires less prior information to deal with the ill-conditioning.

RESUMEN

Esta tesis presenta el algoritmo de estimación de parámetros basado en la dimensión variable de Gauss Newton (VDGN) el cual puede ser usado para la detección y diagnóstico de fallas en un generador sincrónico. El algoritmo es derivado como una extensión del algoritmo de dimensión variable de Newton Raphson propuesto para resolver sistemas de ecuaciones no lineales. Estudiamos el problema de acondicionamiento en el problema de estimación de parámetros para el modelo linealizado de pequeña señal del generador sincrónico utilizando análisis de sensitividad local. El análisis de acondicionamiento es realizado con datos simulados y datos experimentales del generador sincrónico FC5HP de 483 MVA localizado en "Four Corners" estación de generación de la Compañía de Servicio Público de Arizona (APS). Los resultados demuestran que el análisis de sensitividad local es un instrumento efectivo para predecir el mal acondicionamiento. El desarrollo del algoritmo VDGN demuestra ser un método robusto para la estimación de parámetros mal acondicionados asimismo su comportamiento es comparado con el método de selección de subconjuntos. Los resultados del problema de estimación de parámetros de la maquina sincrónica usando datos simulados y experimentales, muestran que el algoritmo VDGN computa mucho mejor los parámetros estimados que el método de selección de subconjuntos y requiere poca o ninguna información a priori para solucionar el mal acondicionamiento

This thesis is dedicated to my wife

Lida Jauregui-Rivera

in recognition of her endless love and unflagging support.

ACKNOWLEDGEMENTS

During the development of my graduate studies in the University of Puerto Rico several persons and institutions collaborated directly and indirectly with my research. Without their support it would be impossible for me to finish my work. That is why I wish to dedicate this section to recognize their support.

I would like to thank Dr. Miguel Vélez-Reyes for giving me the chance to work with him in this research. His advice and assistance were very important for making this work possible. Acknowledgments also go to the professors in my committee: Dr. Agustín Irizarry Rivera and Efrain O'Neill Carrillo for their comments and suggestions in this thesis, thanks for their time to revise and evaluate the final document as well.

Acknowledgments as well go to the Dr. G.T. Heydt from Arizona State University and the Power Systems Engineering Research Center for the synchronous generator data provided for this research.

Special thanks go to my wife Lida Jauregui-Rivera for her support and comprehension during these years. I want also to thank my parents Crisólogo Huaman and Fabiana Qquellon, likewise to my brothers and sisters.

This work was supported primarily by the National Science Foundation under award number EEC-0328200. Other support came from NSF Award number ECS-9702860. This research made use of Center for Power Electronics Systems Shared Facilities supported by the National Science Foundation under Award Number EEC-9731677.

Table Of Contents

List of Tables

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1. JUSTIFICATION AND OBJECTIVES

The synchronous generator is considered the most important and expensive component of a plant in an electrical power system, thus it is of utmost importance that any anomaly in its operation is promptly corrected. Fault detection, isolation, and fault diagnosis for a synchronous generator is a desirable feature that could aid in better monitoring and automation of the machine behavior, and could have a significant impact in establishing an adequate maintenance schedule that ensures proper operation while taking into consideration cost and risk of having a large generator and its maintenance.

This research work looks at providing a robust parameter estimation algorithm that can be used in fault detection, isolation and diagnosis. In a broad sense, a fault is understood as any kind of anomaly or malfunction that leads to an undesired performance of the system under consideration [1].

The problem of fault detection and isolation in dynamical systems is the problem of generating diagnosis signals sensitive to the occurrence of faults. Regarding a fault as an input acting on the system, a diagnostic signal must be able to "detect" its occurrence, as well as to "isolate" this particular input from all other inputs (disturbances, other faults) affecting the system behavior.

A potential diagnosis signal is an on-line parameter estimate. Synchronous generator parameters have traditionally been obtained by performing a series of established tests. Several identification methods have been formulated when data are captured together with an initial set of parameters (usually provided by manufacturers). One of the most common methods used in parameter estimation is output error identification, which minimizes a measure of the error between the actual system output and the output of the system model. Output error identification generally involves nonlinear computations, where iterative methods are necessary to compute the estimates. Physical restrictions in actuation in power systems make it difficult the collection of optimal data for model parameter estimation as in large synchronous machine. On the other side, we could have freedom to excite the system but because of the high resolution of the model, we might not be able to identify all its parameters while a simplified model might not capture the dynamics of interest. In the other case the resulting parameter estimation problem is ill conditioned. Ill conditioned refer to the situation were parameters estimates are very sensitive to noise or other disturbances in the problem data.

For synchronous machine identification, it is usual to consider the parameters that involve the synchronous, transient and subtransient stages with the finality of reflecting the machine behavior under normal operation, before and after a perturbation. The importance of developing more complex machine models is evident because they permit to generate diagnosis signals sensitive to the occurrence of faults, to perform more accurate control, and to predict the machine behavior. For fault detection, we need capability to perform good parameter estimation. Therefore, the question of parameter conditioning is thus a decisive point in the application of fault detection and isolation. Condition analysis refers to the methodology used to study the sensitivity of the parameter estimate to perturbations in the problem data such as, noise and disturbances in the measured voltages and currents. An estimate is ill conditioned if it is very sensitive to such perturbations. Examples of dealing with ill-conditioning in electric machine parameter estimation are presented in [41-43] where an attempt of estimating all model parameters fails because of ill-conditioning, and in [26] using subset selection methodologies to handle ill conditioning in parameter estimation for a synchronous generator are presented. This research work studies the use of variable dimension optimization methods to handle ill conditioning in parameter estimation.

Additional objectives for this work are

- To compare the use of subset selection and Variable Dimension Gauss Newton algorithms for parameter estimation for a synchronous generator..
- To apply the proposed algorithm for the case of a synchronous machine parameter estimation using real data provided by Dr. G. Heydt from Arizona State University (ASU).

1.2. MOTIVATION

This work is motivated by the operating history of a 483 MW synchronous generator located at the Four Corners Generating Station of the Arizona Public Service Company (APS). This unit has undergone outages on a number of occasions due to a short circuit in its field winding. On each occasion this outage occurred, the rotor had to be rebuilt, thus leading to increased costs and possibly decreased reliability for the company. It is beneficial to develop methods to predict possible failures (for instance, a short circuit in the field winding). In this way, preventive maintenance, fault detection and fault diagnosis can be performed so as to avoid costly outages

1.3. CONTRIBUTION

In this thesis we have addressed the development of robust parameter estimation algorithms that can be used in fault detection and diagnosis. Our work has focused mainly on development, testing and implementation of a variable dimension Gauss-Newton (VDGN) algorithm to solve ill conditioned parameter estimation problems.

1.4. THESIS OUTLINE.

This thesis is organized as follows. Chapter 2 presents some fundamental concepts and approaches of fault detection and diagnosis. This is followed by a literature review of failure detection in synchronous generators. Chapter 3 introduces the modeling of the synchronous generator. Chapter 4 addresses the least square parameter estimation and its application to the nonlinear synchronous generator parameter estimation problem. The nonlinear parameter estimation problem is solved by using Gauss Newton method so that the parameter estimates are computed. In addition, sensitivity analysis and evaluation of conditioning of the synchronous generator is introduced. Chapter 5 develops parameter estimation using the subset selection technique to overcome ill conditioning. Chapter 6 presents parameter estimation using the variable dimension technique and its application to the synchronous machine parameter estimation. The conclusions and recommendations and future work of this thesis are presented in Chapter 7. The appendices present the data formats that are supported by the synchronous parameter estimation algorithms and the MATLAB® implementation of all algorithms.

CHAPTER 2 BACKGROUND AND LITERATURE REVIEW

2.1. INTRODUCTION

For the improvement of reliability, safety and efficiency advanced methods of supervision, fault detection and diagnosis become increasingly important for many technical processes. This holds especially for safety related equipment like aircraft, trains, automobiles, power plants and chemical plants.

In this chapter, some fundamentals concepts and approaches of fault detection and diagnostic will be introduced. The literature review is mainly focused in the following topics

- The scope of fault detection and diagnosis.
- Approaches to fault detection and diagnosis.
- Model-free methods
- Model-Based Methods.
- Failure detection in synchronous generator.

2.2. THE SCOPE OF FAULT DETECTION AND DIAGNOSIS

In this initial section, some fundamental concepts and approaches of fault detection and diagnosis will be introduced.

2.2.1. TYPES OF FAULTS

In general faults are deviations from the normal operation of the plant or its components. The faults of interest can be organized in the following categories [1-6].

Additive process faults. These are unknown inputs acting on the plant, which are normally zero and which, when present, cause a change in the plant outputs independent of the known inputs.

Multiplicative faults. These are changes (abrupt or gradual) in some plant parameters. They cause changes in the plant outputs which depend also on the magnitude of the known input. Such faults best describe the deterioration of plant equipment.

Sensor faults. These are discrepancies between the measured and actual values of individual plant variables. These faults are usually considered additive (independent of the measure magnitude), though some sensor faults (such as sticking or complete failure) may be better characterized as multiplicative.

Actuator faults. These are discrepancies between the input command of an actuator and its actual output. Actuator faults are usually handled as additive though, again kinds (sticking or complete failure) may be better described as multiplicative

From the point of view of diagnosis, it is of interest how a particular fault affects the plant outputs (additive or multiplicative faults).

2.2.2. FAULT DETECTION AND DIAGNOSIS

In this section, the fundamental concepts of fault detection and diagnosis using analytical redundancy are outlined. Fault detection and diagnosis in general include three functions [1]

- *Fault detection.* To indicate the presence of fault(s)
- *Fault Isolation.* To determine the location of the fault(s)
- *Fault identification.* To determine the size of the fault(s)

The isolation and identification tasks together are referred to as *fault diagnosis.* The detection function is indispensable and the isolation function is usually also required. However, the fault identification function can be omitted in most cases, except when the fault size is really important the combination of isolation and identification (or isolation alone if the latter is missing) is also referred to as fault diagnosis. In many cases “diagnosis” is used simply as a synonym to “isolation”.

Usually, the fault detection and diagnosis activity takes on-line, in real time. The detection and diagnosis may be performed in parallel or sequentially, that is by invoking the isolation function only once a fault has been detected, or in parallel, that is simultaneously.

Particularly in model-based fault detection and diagnosis the following conventions are usually adopted.

- It is assumed that *faults* are not present initially in the system but arrive at some later time. Faults are generally described by deterministic time-function which are unknown.
- One may speak of *additive disturbances* as well, which are also deterministic and unknown inputs to the system. The distinction between additive faults and disturbances is subjective: faults are those unknown inputs which we wish to detect and isolate while disturbances are nuisances which we wish to ignore.
- Any *noise* , that originates from the plant or from the sensors and actuators, is considered random with zero mean (any nonzero mean is handled as fault or disturbance)

- *Modeling errors* are discrepancies between the model (model parameters) and the true system. They may be present due to operating-point changes. Model errors are nuisances that we want to suppress.

In most practical situations, fault diagnosis needs to be performed in the presence of *disturbances, noise,* and *modeling errors.* These interfere with the diagnosis of fault and may lead to false alarms and miss-classification (miss isolation). Therefore the diagnosis algorithm needs to be so designed that it

- Is made *insensitive* to the disturbance
- Includes mechanisms to suppress the effects of noise ;
- Is r*obust* with *respect* to modeling error;
- Maintains sufficient *sensitivity* with respect to faults.

Fault sensitivity, and robustness arise from interplay between faults on the one hand, and noise, disturbance and model errors on the other hand, and are affected by the design of the detection algorithm.

2.3. APPROACHES TO FAULT DETECTION AND DIAGNOSIS

The methods of fault detection and diagnosis may be classified into two major groups: Those which do not utilize the mathematical model of the plant and those which do. This section is dedicated to the model based methods. Thus, we first, introduce, the model-free technique.

2.3.1. MODEL-FREE METHODS

The fault detection and isolation which do not use the mathematical model of the plant range from physical redundancy and especial sensors through limit-checking and spectrum analysis to logical reasoning [1-5].

Physical redundancy. In this approach, multiple sensors are installed to measure the same physical quantity. Any serious discrepancy between the measurements indicates a sensor fault. An example of this method is presented in [7].

Special sensors may be installed explicitly for detection and diagnosis. These are limit sensor (e.g., temperature or pressure) which perform limit checking (plant measurements are compared by computer to preset limits). Other special sensors may measure some fault indicating physical quantity. References [8-10] are examples of this technique

Spectrum analysis of plant measurements may also be used for detection and isolation. Most plant variables exhibit a typical frequency spectrum under normal operation condition; any deviation from this is an indication of abnormality, Reference [11] is a good example.

2.3.2. MODEL-BASED METHODS

Different approaches for fault detection using mathematical models have been used in the last 20 years (see [1-6, 9, 27, 31]). Model-based fault detection and diagnosis (MBFDD) methods employ an explicit mathematical model of the system under test. The task consists of the detection of fault in the processes, actuators and sensors by using the dependencies between different measurable signals. These dependencies are expressed by mathematical process models.

Most of the model-based fault detection and diagnosis methods rely on the concept of *analytical redundancy*, i.e., on static or dynamic relationships among measured variables. The entire fault detection, isolation and accommodation (FDIA) process consists then of the following three steps:

- The generation of so-called residual, i.e. of functions that carry information about faults
- The decision process that evaluates the residuals and monitoring if and where a fault has occurred
- The accommodation process by which the normal system operation is restored

Examining the literature on fault detection and isolation (FDI) based on analytical redundancy one can see that the wide variety of published methods can roughly be divided into two major groups:

- Parameter estimation methods [6, 9, 26]
- State (or output) estimation methods [4-6, 15, 16, 19, 27, 31]

The parameter estimation approach employs on-line identification of the mathematical model in order to determine the physical coefficients of the process. In the state approximation on-line reconstruction of sets or subsets of states or measured variables is done with the aid of parity equations, observers or Kalman filters whose estimates or innovations are then used for residual generation. In the literature one finds that the most commonly studied and applied fault diagnosis approach is the state estimation methods.

Figure 2.1 shows the basic structure of model-based fault detection. Based on measured input signal U and output signal Y the detection methods generate residuals r, parameter estimates $\hat{\alpha}$ or state estimates $\hat{\theta}$, which are called features. By comparison with the normal features, changes of features are detected, leading to analytical symptoms s.

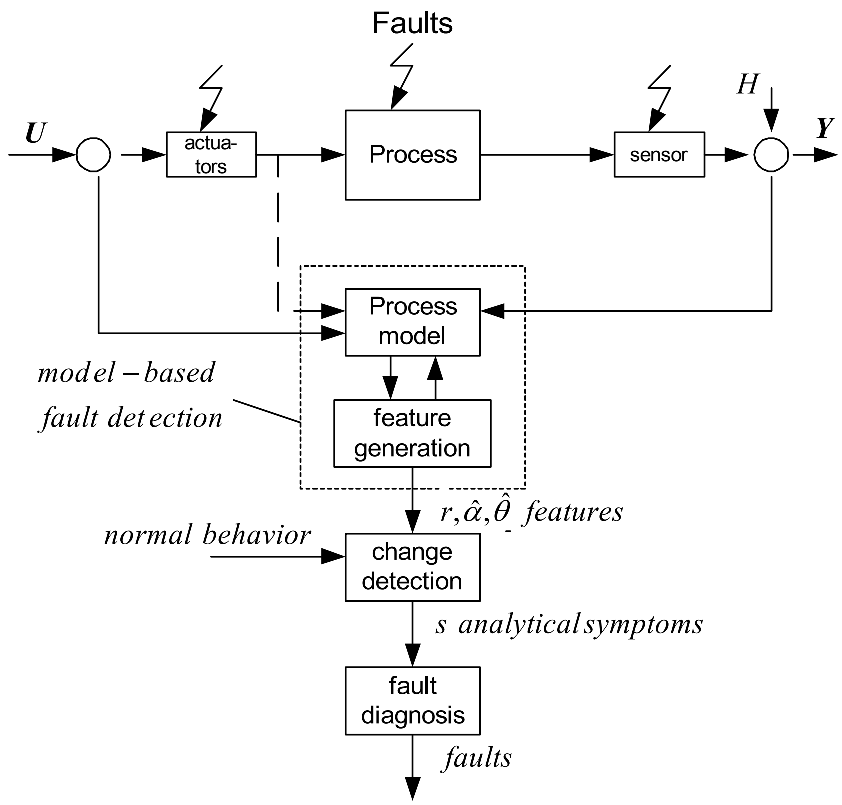

Figure 2.1: General scheme of process MBFDD [5]

2.3.2.1. PROCESS AND FAULTS MODELING

A fault is defined as a deviation of at least one characteristic property of a variable from an acceptable behavior [5]. Therefore, the fault is a state that may lead to a malfunction or failure of the system. The time dependency of faults can be distinguished such as an abrupt fault (stepwise), incipient fault (drift like), or intermittent fault. With regard to the process models, the fault can be further classified. According to Figure 2.2 *additive faults* influence a variable Y by an addition of the fault f, and *multiplicative faults* by the product of another variable U with f. As we saw previously, additive faults appear as offset sensors, whereas multiplicative faults are parameters changes within a process.

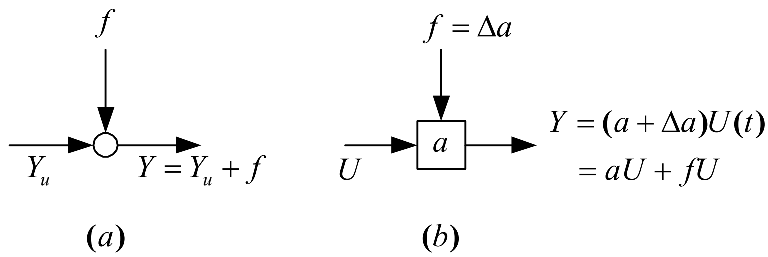

Figure 2.2 Basic models of faults: (a) additive faults; (b) multiplicative faults

2.3.2.2. STATIC PROCESS MODEL

The steady state behavior of a system can be frequently expressed by a *non-linear* characteristic as shown in Figure 2.3 .

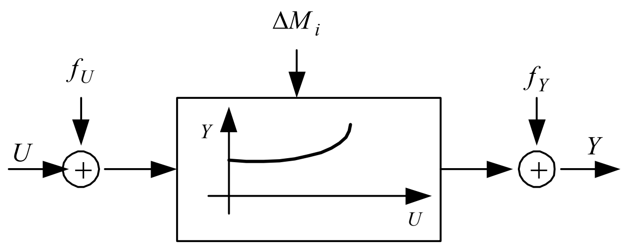

Figure 2.3: Fault detection of a nonlinear static process via parameter estimation for steady states.

where $U(t)$, $Y(t)$ are the measured signals, f_U, f_Y are the input and output additive faults; and ΔM_i are the multiplicative parameter faults. A polynomial model is given by:

$$Y = M_0 + M_1 U + M_2 U^2 + \ldots + M_q U^q$$
$$\rightarrow Y = \psi_s^T \alpha_s \tag{2.1}$$

where

$$\psi_s^T = [1, U, U^2, \cdots, U^q] \tag{2.2}$$

$$\alpha_s^T = [M_0, M_1, \ldots, M_q] \tag{2.3}$$

Estimates and changes of the parameter M_i can be obtained by parameter estimation methods such as least squares, using measurements of different input-output pairs $\lfloor Y_j, U_j \rfloor$.

2.3.2.3. DYNAMIC PROCESS MODEL

More information on the process can usually be obtained with *dynamic process model.* This process model can be presented in form of differential equation or a state space model as a vector differential equation. Figure 2.4 shows the basic input/output linear dynamical model and fault modeling in the form of a differential equation.

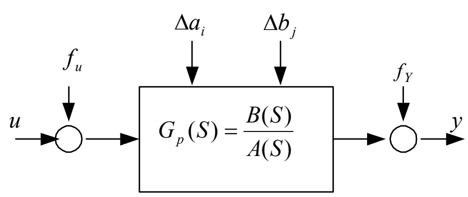

Figure 2.4: Linear dynamic process input/output model and fault modeling

In Figure 2.4 $y(t) = Y(t) - Y_\infty$ and $u(t) = U(t) - U_\infty$ are the measured signals, f_u and f_y are the input, output additive faults respectively, $\Delta a_i, \Delta b_j$ are the multiplicative parameter faults. The model equation

$$\begin{aligned} y(t) &= a_1 y^{(1)}(t) + \ldots + a_n y^{(n)}(t) \\ &= b_0 u(t) + b_1 u^{(1)}(t) + \ldots + b_m u^{(m)}(t) \\ &\rightarrow y(t) = \psi^T(t)\alpha \end{aligned} \tag{2.4}$$

where

$$\psi^T = \left[- y^{(1)}(t) \ldots - y^{(n)}(t) \quad u(t) \ \ldots \ u^{(m)}(t)\right] \tag{2.5}$$

$$\alpha^T = [a_1 \ldots a_n \; b_0 \ldots b_m] \tag{2.6}$$

Figure 2.5 shows the *state space linear dynamic model and fault modeling* in the form of vector differential equation.

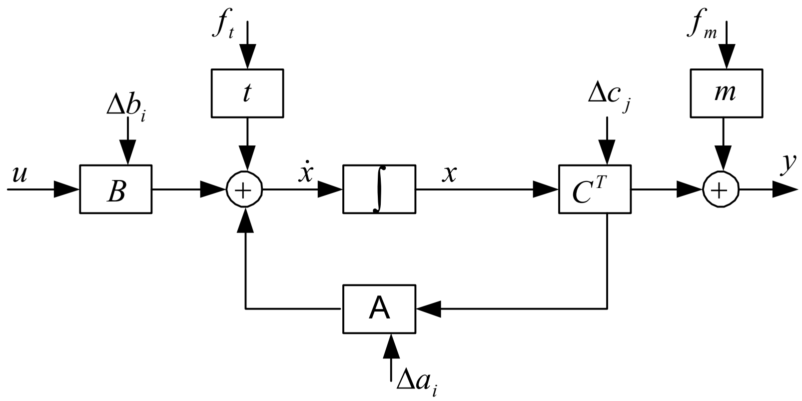

Figure 2.5 : State space model for linear dynamic process and fault modeling

In Figure 2.5 f_t are the input additive state variable fault, f_m the output additive fault, $\Delta a_i, \Delta b_i, \Delta c_j$ are the multiplicative parameter faults; and the state space basic equations are

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= C^T x(t) \end{aligned} \tag{2.7}$$

where

$$A = \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 1 & -a_1 \\ 0 & \cdots & 1 & 0 & -a_2 \\ \vdots & \iddots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & \cdots & -a_n \end{bmatrix}, B^T = [b_0 \quad b_1 \quad \cdots], C^T = [0 \quad 0 \quad \cdots \quad 1] \tag{2.8}$$

Similar representations hold for non-linear processes see [4], also in discrete time.

2.3.3. FAULT DETECTION USING PARAMETER ESTIMATION

In most practical cases the process parameters are partially or totally unknown. They can be determined with parameter estimation methods by measuring input and output signals if the model structure is known [2-5]. In [5] two approaches for parameter estimation using *equation error* and the *output error* are shown. The equation error is linear in the parameters and allows therefore direct estimation of the parameters. The Output error approach shown in Figure 2.6 needs nonlinear optimization methods and therefore iterative procedures, but may be more precise under the influence of process disturbances; see an example in [26].

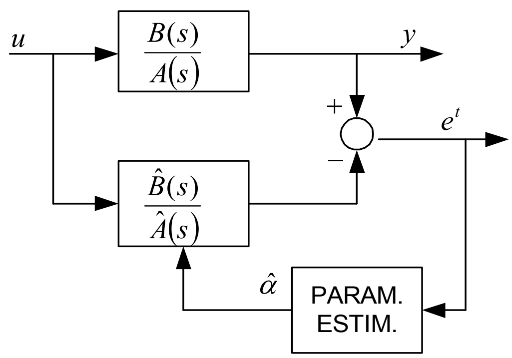

Figure 2.6: Minimization of output error

The fault symptoms appear in deviations of the process parameters $\Delta\alpha$. As the process parameters $\alpha = f(p)$ depend on physically defined process coefficients p (such as inductances). Determination of changes Δp allows usually a deeper insight and makes fault diagnosis easier. Parameter estimation methods usually need a process input excitation and are especially suitable for the detection of *multiplicative faults* [1, 5].

2.3.4. FAULT DETECTION USING OBSERVERS

If the process parameters are known, either *state observers* or *output observers* can be applied [1, 6]. The classical state observers can be applied if the faults are modeled as state variable changes Δx_i .In the case of multi-output processes, special arrangement of observers can be applied as shown in [2] and [8]. Another possibility is the use of output observers and\or unknown input observers if the reconstruction of the state variables $x(t)$ is not of interest, see Figure 2.7.

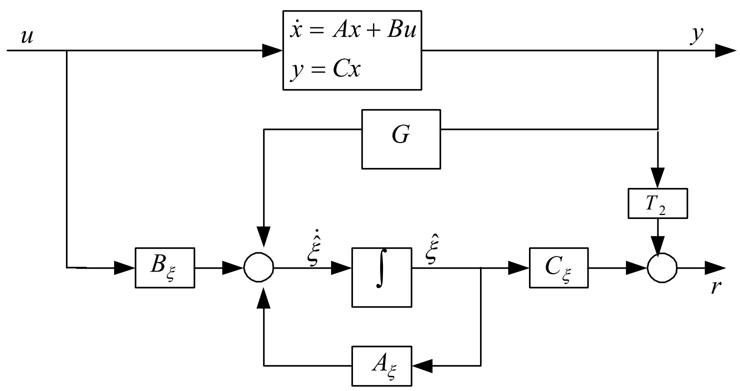

Figure 2.7: Output error fault detection for dynamic processes

In this case, a linear transformation then leads to new state variables $\xi(t)$. The residuals $r(t)$ can be designed such that they are independent on the unknown inputs $v(t)$ (disturbance signal), and of the state $x(t)$ and $u(t)$ by especial determination of the matrices C_ξ and T_2. The residuals then *depend only of the additive faults*. However, all process model matrices must be known precisely. A comparison with the parity equation approach developed in the next section shows similarities.

2.3.5. FAULT DETECTION USING PARITY EQUATIONS

A general approach of creating robustness in fault detection and isolation (FDI) has been pursued over the years by Willsky et al. (see [1-3, 5]). These investigators studied the problem of robust residual generation from the viewpoint of analytical redundancy relations, and introduced the concept of *general parity checks*. From this perspective, the innovation of an observer or a Kalman filter can be considered as the residual containing the complete set of redundancy relations. The alternative approach of increasing the robustness of observer schemes by using "robust" or "unknown inputs" observers has been tackled by [4].

A straightforward model-based method of fault detection is to take a fixed model G_M and run it parallel to the process, thus forming and *output error* [1].

$$r'(s) = \left[G_p(s) - G_M(s)\right]u(s) \tag{2.9}$$

If $G_p(s) = G_M(s)$, the output error for additive input and output faults becomes, (Figure 2.4)

$$r(s) = G_p(s) f_u(s) + f_y(s) \tag{2.10}$$

Another possibility is to generate a polynomial error or equation error. The residuals then depend only on the additive input faults $f_u(t)$ and output faults $f_y(t)$.

The same procedure can be applied for multivariable processes by using a state space model as shown in Figure 2.8.

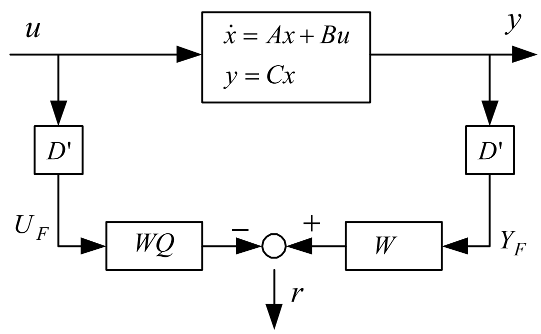

Figure 2.8 : State space model for fault detection with parity equations for dynamic process

In Figure 2.8 the parity equations are

$$
\begin{aligned}
&Y_F(t) = TX(t) + QU_F(t) \\
&W.Y_F(t) = W.T.x(t) + W.Q.U_F(t) \\
&W.T = 0 \\
&r(t) = W(Y_F(t) - QU_F(t))
\end{aligned}
\tag{2.11}
$$

where

$$
Q = \begin{bmatrix} 0 & 0 & 0 & \cdots \\ CB & 0 & 0 & \\ CAB & CB & 0 & \\ \vdots & & & \end{bmatrix}
\begin{matrix} U_F = D'u = \left[uu^{(1)} \ldots u^{(m)}\right]^T \\ ; Y_F = D'y = \left[yy^{(1)} \ldots y^{(n)}\right]^T \\ T = \left[C \quad CA \quad CA^2 \quad \cdots\right]^T \end{matrix}
\tag{2.12}
$$

The derivative of the signals can be obtained by state variable filters [6]. Corresponding equations exist for discrete time. The components of matrix W are selected such that one measured variable has no impact on a specific residual. This allows generating structured residuals in order to obtain good isolating patterns for the residuals. Hence *parity equations are suitable for the detection of additive faults*. They are simpler to design and to implement than output observer-based approaches and lead approximately to the same results

2.3.6. FAULT DIAGNOSIS METHODS

The task of fault diagnosis consists of the determination of the type of fault with as many details as possible such as the fault size, location and times of detection. The diagnostic procedure is based on the observed analytical and heuristic symptoms and the heuristic knowledge of the process [9, 21-24]. The inputs to a knowledge-based fault diagnosis system are all available symptoms as fact and the fault–relevant knowledge about the process, mostly in heuristic form. The symptoms may be presented just a binary values (0,1) or for example fuzzy sets to take gradual sizes into account.

2.3.6.1. CLASSIFICATION METHODS OF FAULT DIAGNOSIS

If no further knowledge is available for the relationship between features and faults; classification or pattern recognitions methods can be used [21].

In Figure 2.9 the reference vectors R_n are determined for the normal behavior. Then the corresponding input vectors R of the features are determined experimentally for certain faults F_j. The reference between F and R is therefore learned (or trained) experimentally and stored, forming and *explicit knowledge base* [6]. By comparison of the observed R with the normal reference R_n, faults F can be concluded.

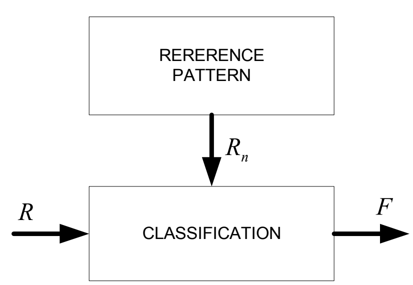

Figure 2.9 Classification methods

One distinguished between *statistical or geometrical classification methods.* With or without certain probability function [21]. A further possibility of fault diagnosis is the use of *neural networks* [12] or *neuro-fuzzy* technique approach [14] because of its ability to approximate non-linear relation and to determine flexible decision regions for faults in continuous or discrete form.

2.3.6.2. INFERENCE METHODS

For some technical processes, the basic relationships between faults and symptoms are at least partially known. Then this a-priori-knowledge can be represented in causal relations: fault-event-symptoms. For the establishment of this heuristic knowledge several approaches exist [6, 14].

2.4. FAILURE DETECTION IN SYNCRONOUS GENERATOR

Examining the literature on fault detection and diagnosis in synchronous generators one can see a wide variety of published methods. These methods use the concepts developed above to detect and to diagnose faults. Although some of these works use model-free methods a few works develop fault detection and diagnosis using model-based methods, especially fault detection using parameter estimation concepts.

Examples based on model-free methods can be seen in [11, 13, 18, 19, 25]. In [11] a method of detecting short circuits in both the stator and rotor windings of synchronous generator is proposed. The technique allows the identification of predictable harmonics in the rotor currents as well as predictable components in the stator current spectrum of the exciter. In [13], a combination of on-line monitoring and off-line diagnostic inspection, supported by data mining and expert systems is presented. In [25], thermal, acoustic, and rotor vibration sensors are monitored in real-time to provide early warning of impending failures in hydro-generator stator.

Examples using model-based concepts are developed in [9, 12, 16, 26, 31]. Development and implementation of a fault diagnostic scheme for generator winding protection using Artificial Neural Networks is introduced in [12]. This scheme has the ability to detect generator winding faults and classify the type of fault. In addition, the technique proposed in [12] has the ability to identify the faulty phases with higher sensitivity and stability boundaries as compared to differential relays. In [16] and [31] methods to detect faults using observers and extended Kalman filters are proposed. Process motor testing using parameter estimation fault detection is introduced in [9] and [26]. Several case studies obtained from field test at electric motor manufacturing plants are presented. Other special requirements for condition-base maintenance (CBM) in synchronous generator, maintenance experience and design, and system knowledge are discussed in [10, 17-20, 22-24].

2.5. SUMMARY

The literature review presented in this chapter, was mainly focused to approaches in fault detection and diagnosis. We described methods to monitor synchronous generator condition, process models and fault modeling, and the most practical cases of process parameters such as fault detection using parameter estimation, fault detection using parameter observers, fault detection using parity equations, and fault diagnosis methods. These methodologies were revised in order to provide an idea of how the parameter estimates, using different approaches, can be used to diagnose and detect possible synchronous machine faults.

CHAPTER 3 MODELING OF THE SYNCHRONOUS GENERATOR

3.1. INTRODUCTION

On line fault detection, isolation and fault diagnosis of generator is desirable feature that could aid in better monitoring and automation of the machine behavior. The problem of fault detection and isolation in dynamical systems is the problem of generating diagnosis signals sensitive to the occurrence of faults, in this sense the importance of obtaining accurate parameters of synchronous machine is a desirable feature. Thus developing more complex machine models is desirable since they permit to generate diagnosis signals sensitive to the occurrence of faults as well as more accurate control and prediction of the machine behavior.

Traditional methods have been developed under the guidance of standard methods to measure the parameters for example, the short-circuit test is a conventional off-line method to determine synchronous machine parameters on unload machines. The test procedures are specified in IEEE Standard 115-1983. These tests provide the d-axis parameters, but they do not give q-axis transient and subtransient constants. Furthermore, the IEEE standard tests do not include measurements of the field circuits during the short circuit test, and consequently the field circuit is not adequately defined. Several alternative testing and analytical methods have been proposed and used to obtain better models, for instance [28] presents an enhanced sudden short circuit test, stator decrement test, and frequency response tests. Indeed, all these methods are conducted under *off-line* conditions. Thus the parameters obtained by these methods cannot truly characterize the machine behavior under various *on-line* conditions. Many studies have been published with great contribution to the on-line methods for synchronous machines parameter estimation [28-31]. [30] used a nonlinear least squares parameter estimation approach for on-line identification. A full-scale experimental verification for large synchronous machines was presented. With the generator connected to a large power system and loaded normally, a sudden change of excitation (field voltage) is applied through some appropriate means. The transients in the line voltage, line currents and field voltage are recorded and used as the basis of parameter estimation. In [31] a Visual C++ engine and graphic user interface are implemented so as to allow parameter estimation for a synchronous machine from on-line measurements; the estimation is performed by using noise free data, resulting in accurate parameter estimates. The synchronous machine mathematical model used in [31] has three stator windings, one field winding and one damper winding all of them magnetically coupled.

The model and the data collected are of great importance in parameter estimation. For instance, models can suffer from over parameterization or the available data is not sufficiently rich for accurate parameter identification.

In order to formulate de on-line parameter estimation for a synchronous generator, it is necessary to employ a mathematical model which represents the synchronous generator in the conditions under study.

In this regard, the next section presents the mathematical model of a synchronous machine and introduces its steady-state and transient performance.

3.2. SYNCHRONOUS GENERATOR MODEL

It is necessary to make some assumptions in developing equations of a synchronous machine, as it is suggested in [26, 29]

- The stator winding are sinusoidally distributed along the air-gaps as far as the mutual effects with the rotor are concerned
- Magnetic hysteresis is negligible
- Magnetic saturation effects are negligible

Given those considerations, the machine consists of two essential components: the field and the armature. The field winding carries direct current and produces a magnetic field, which induces alternating voltages in the armature windings [29]. It is customary to have the armature on the stator. The three-phase winding of the armature are distributed 120^o apart in space so that, with uniform rotation of the magnetic field, voltages displaced 120^o in time will be produced in the windings. When carrying balanced three-phase currents, the armature will produce a magnetic field in the air-gap rotating at synchronous speed. The field produced by the direct current in the rotor winding, on the other hand, revolves with the rotor. For production of a steady state torque, the fields of stator and rotor must run at precisely the synchronous machine. It is common to have two or three damper windings in the rotor mounted in the rotor with the intention to damp out speed oscillations.

Figure 3.1 shows the circuit of a synchronous machine of salient poles. The machine consists of two circuits, the rotor and the stator [28]. The circuit of the rotor has a field winding and three damper windings, two of them in quadrature axis, and the other in the direct axis. Variables v_a, v_b, v_c are stator phase voltages; $v_{fd}, v_{1d}, v_{1q}, v_{2q}$ are field and damper winding voltages, and θ represents the rotor angle.

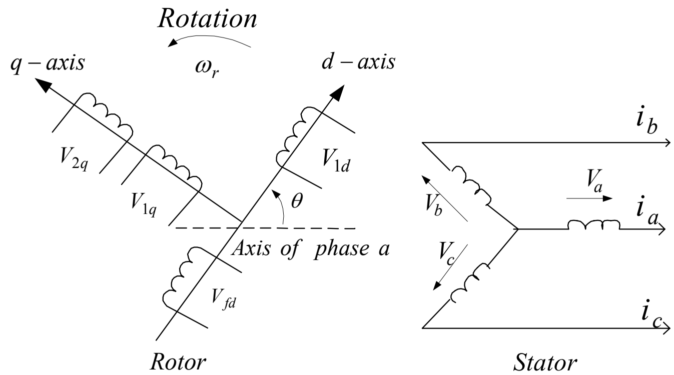

Figure 3.1 Rotor and stator circuits of the synchronous machine

Based on the above schematic, the electrical and mechanical subsystems are obtained as shown in [26, 28, 29].

3.2.1. ELECTRICAL AND MECHANICAL EQUATIONS OF THE SYNCHRONOUS MACHINE

The electrical subsystem of the synchronous machine is characterized by the stator a rotor equations. The quantities for the stator are

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \begin{bmatrix} -R_a & 0 & 0 \\ 0 & -R_a & 0 \\ 0 & 0 & -R_a \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \lambda_a \\ \lambda_b \\ \lambda_c \end{bmatrix} \tag{3.1}$$

where i_a, i_b, i_c, are the stator currents and $\lambda_a, \lambda_b, \lambda_c$ are the *a, b, c* flux linkages defined as follows

$$\begin{aligned} \lambda_a = &-i_a[L_{aa0} + L_{aa2}\cos 2\theta] + i_b[L_{ab0} + L_{aa2}\cos(2\theta + \frac{2\pi}{3})] + i_c[L_{ab0} \\ &+ L_{aa2}\cos(2\theta - \frac{2\pi}{3})] + i_{fd}L_{afd}\cos\theta + i_{kd}L_{akd}\cos\theta - i_{kq}L_{akq}\sin\theta \end{aligned} \tag{3.2}$$

$$\lambda_b = i_a[L_{ab0} + L_{aa2}\cos(2\theta + \frac{2\pi}{3})] - i_b[L_{aa2} + L_{aa2}\cos(2\theta - \frac{2\pi}{3})] + i_c[L_{ab0} + L_{aa2}\cos(2\theta - \pi)]$$
$$+ i_{fd} L_{afd}\cos(\theta - \frac{2\pi}{3}) + i_{kd} L_{akd}(\theta - \frac{2\pi}{3}) - i_{kq} L_{akq}\sin(\theta - \frac{2\pi}{3})$$

(3.3)

$$\lambda_c = i_a[L_{ab0} + L_{aa2}\cos(2\theta - \frac{2\pi}{3})] + i_b[L_{ab0} + L_{aa2}\cos(2\theta - \pi)] - i_c[L_{aa0} + L_{aa2}\cos 2(\theta + \frac{2\pi}{3})]$$
$$+ i_{fd} L_{afd}\cos(\theta + \frac{2\pi}{3}) + i_{kd} L_{akd}(\theta + \frac{2\pi}{3}) - i_{kq} L_{akq}\sin(\theta + \frac{2\pi}{3})$$

(3.4)

and the equations of the rotor are

$$\begin{bmatrix} v_{fd} \\ v_{1d} \\ v_{1q} \\ v_{2q} \end{bmatrix} = \begin{bmatrix} r_{fd} & 0 & 0 & 0 \\ 0 & r_{1d} & 0 & 0 \\ 0 & 0 & r_{1q} & 0 \\ 0 & 0 & 0 & r_{2q} \end{bmatrix} \begin{bmatrix} i_{fd} \\ i_{1d} \\ i_{1q} \\ i_{2q} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \lambda_{fd} \\ \lambda_{1dr} \\ \lambda_{1qr} \\ \lambda_{2qr} \end{bmatrix} \tag{3.5}$$

where $i_{fd}, i_{1d}, i_{1q}, i_{2q}$ are the rotor currents and $\lambda_{fd}, \lambda_{1dr}, \lambda_{1qr}, \lambda_{2qr}$ are the rotor flux linkage, defined by

$$\lambda_{fd} = L_{ffd} i_{fd} + L_{fkd} i_{kd} - L_{afd}[i_a\cos\theta + i_b\cos(\theta - \frac{2\pi}{3}) + i_c\cos(\theta + \frac{2\pi}{3}) \tag{3.6}$$

$$\lambda_{kd} = L_{fkd} i_{fd} + L_{kkd} i_{kd} - L_{akd}[i_a\cos\theta + i_b\cos(\theta - \frac{2\pi}{3}) + i_c\cos(\theta + \frac{2\pi}{3})] \tag{3.7}$$

$$\lambda_{k1q} = L_{kk1q} i_{k1q} + L_{akq}[i_a\cos\theta + i_b\cos(\theta - \frac{2\pi}{3}) + i_c\cos(\theta + \frac{2\pi}{3})] \tag{3.8}$$

$$\lambda_{k2q} = L_{kk2q} i_{k2q} + L_{akq}[i_a\cos\theta + i_b\cos(\theta - \frac{2\pi}{3}) + i_c\cos(\theta + \frac{2\pi}{3})] \tag{3.9}$$

The mechanical subsystem of the synchronous machine involves equations of motion and is given by:

$$\frac{d\theta}{dt} = \frac{2}{p}\omega \tag{3.10}$$

$$J\frac{2}{P}\frac{d\omega}{dt} = T_m - T_e - D\omega \tag{3.11}$$

$$T_e = \frac{3}{2}(\lambda_d i_q - \lambda_q i_d)\frac{p}{2} \tag{3.12}$$

where the number of magnetic poles per phase is p, $\omega = \frac{p}{2}\omega_{rm}$ is the rotor angular velocity expressed in electrical radians per second for a *p*-pole machine and ω_{rm} is the mechanical angular velocity of the rotor. J represents the inertia constant, T_m is the load torque, T_e is the electrical torque and D is a mechanical damping coefficient.

Equations (3.1)-(3.12) completely describe the electrical and mechanical behavior of a synchronous machine [28]. The electric equations contain inductance terms, which vary with angle θ, which in turn varies with time. This dependency is eliminated by means of the Park transformation defined in [28] so as to obtain the so-called *dq0* model. The general form of the transformation that accomplishes this is Park's transformation T_{dq0}. So that the following expressions are fulfilled

$$v_{dq0} = T_{dq0} v_{abc} \quad i_{dq0} = T_{dq0} i_{abc} \quad \psi_{dq0} = T_{dq0} \psi_{abc}$$

and

$$T_{dq0} = \begin{bmatrix} \cos\theta & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ -\sin\theta & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta + \frac{2\pi}{3}) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{3.13}$$

Using the expressions for $\lambda_a, \lambda_b, \lambda_c$ given in equations (3.2)-(3.41), transforming the flux linkage and currents into *dq0* components, and with suitable reduction of terms involving trigonometric terms, we obtain the following expressions of the Stator flux.

$$\lambda_q = -(L_{aa0} + L_{ab0} + \frac{3}{2}L_{aa2})i_d + L_{afd}i_{fd} + L_{akd}i_{kd} \tag{3.14}$$

$$\lambda_q = -(L_{aa0} + L_{ab0} - \frac{3}{2}L_{aa2})i_q + L_{akd}i_{kd} \tag{3.15}$$

$$\lambda_0 = -(L_{aa0} - 2L_{ab0})i_0 \tag{3.16}$$

setting $L_d = (L_{aa0} + L_{ab0} + \frac{3}{2}L_{aa2})$; $L_q = (L_{aa0} + L_{ab0} - \frac{3}{2}L_{aa2})$; $L_0 = (L_{aa0} - 2L_{ab0})$

Then the flux linkage equations become

$$\lambda_d = -L_d i_d + L_{afd} i_{fd} + L_{akd} i_{kd} \tag{3.17}$$

$$\lambda_q = -L_q i_q + L_{akd} i_{kd} \tag{3.18}$$

$$\lambda_0 = -L_0 i_0 \tag{3.19}$$

Using the same procedure to obtain the dq0 stator equations we can obtain the flux linkages for the rotor

$$\lambda_{fd} = L_{ffd} i_{fd} + L_{fkd} i_{kd} - \frac{3}{2} L_{afd} i_d \tag{3.20}$$

$$\lambda_{kd} = L_{fkd} i_{fd} + L_{kkd} i_{kd} - \frac{3}{2} L_{akd} i_d \tag{3.21}$$

$$\lambda_{k1q} = L_{k1q} i_{k1q} + L_{k2q} i_{k2q} - \frac{3}{2} L_{akq} i_q \tag{3.22}$$

$$\lambda_{k2q} = L_{k1q} i_{k1q} + L_{k2q} i_{k2q} - \frac{3}{2} L_{akq} i_q \tag{3.23}$$

Observe that the *dq0* components of stator flux linkages are seen to be related to the components of stator and rotor currents through constant inductances. For the *dq0* components of the rotor, all the inductances are also seen to be constant, i.e., they are independent of the rotor position. It should however be noted that the saturation effects are not considered here. The variations in the inductances due to the saturation are of different nature, which is not considered in this model. Using (3.13)-(3.19) Stator voltage equations in *dq0* components can be expressed as follows [28]

$$v_d = -R_s i_d - \omega \lambda_q + \frac{d\lambda_d}{dt} \tag{3.24}$$

$$v_q = -R_s i_q - \omega \lambda_d + \frac{d\lambda_q}{dt} \tag{3.25}$$

$$v_0 = -R_s i_o + \frac{d\lambda_o}{dt} \tag{3.26}$$

and using (3.20)-(3.23) the rotor voltage equations in *dq0* components can be expressed as follows

$$v_{fd} = R_{fd} i_{fd} + \frac{d\lambda_{fd}}{dt} \tag{3.27}$$

$$v_{kd} = R_{kd} i_{kd} + \frac{d\lambda_{kd}}{dt} \tag{3.28}$$

$$v_{k1q} = R_{k1q} i_{k1q} + \frac{d\lambda_{k1q}}{dt} \tag{3.29}$$

$$v_{k2q} = R_{k2q} i_{k2q} + \frac{d\lambda_{k2q}}{dt} \tag{3.30}$$

Reordering these last equations, we obtain

$$\frac{d\lambda_{ds}}{dt} = -R_s i_d + \omega \lambda_{qs} + v_d \tag{3.31}$$

$$\frac{d\lambda_{qs}}{dt} = -R_s i_q + \omega\lambda_{ds} + v_q \tag{3.32}$$

$$\frac{d\lambda_{0s}}{dt} = -R_s i_o + v_o \tag{3.33}$$

$$\frac{d\lambda_{fd}}{dt} = -R_{fd} i_{fd} + v_{fd} \tag{3.34}$$

$$\frac{d\lambda_{kd}}{dt} = -R_{kd} i_{kd} + v_{kd} \tag{3.35}$$

$$\frac{d\lambda_{k1q}}{dt} = -R_{kd} i_{k1d} + v_{k1q} \tag{3.36}$$

$$\frac{d\lambda_{k2q}}{dt} = -R_{kd} i_{k2d} + v_{k2q} \tag{3.37}$$

3.2.2. *dq0* Synchronous Machine Model

The dq0 equations shown above can be rewritten as follows

$$\begin{bmatrix} v_d \\ v_q \\ v_0 \end{bmatrix} = \begin{bmatrix} R_a & 0 & 0 \\ 0 & r_s & 0 \\ 0 & 0 & r_s \end{bmatrix} \begin{bmatrix} i_d \\ i_q \\ i_0 \end{bmatrix} - \omega \begin{bmatrix} \lambda_{qs} \\ -\lambda_{ds} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \lambda_{ds} \\ \lambda_{qs} \\ \lambda_{0s} \end{bmatrix} \tag{3.38}$$

and for the rotor

$$\begin{bmatrix} v_{fd} \\ v_{1d} \\ v_{1q} \\ v_{2q} \end{bmatrix} = \begin{bmatrix} r_{fd} & 0 & 0 & 0 \\ 0 & r_{1d} & 0 & 0 \\ 0 & 0 & r_{1q} & 0 \\ 0 & 0 & 0 & r_{2q} \end{bmatrix} \begin{bmatrix} i_{fd} \\ i_{1d} \\ i_{1q} \\ i_{2q} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \lambda_{fd} \\ \lambda_{1dr} \\ \lambda_{1qr} \\ \lambda_{2qr} \end{bmatrix} \tag{3.39}$$

where $r_s, r_{fd}, r_{1d}, r_{1q}, r_{2q}$ are the stator and rotor resistances; $i_d, i_q, i_{fd}, i_{1d}, i_{1q}, i_{2q}$, are the *dq* currents of the stator and rotor; $\lambda_{ds}, \lambda_{qs}, \lambda_{fd}, \lambda_{1dr}, \lambda_{1qr}, \lambda_{2qr}$ are the *dq* fluxes of the stator and rotor. For balanced systems it is usually assumed that the voltage and current i_o, v_o are equal to zero.

However we need to derive the parametrized model of the synchronous machine to use in the estimation process. In [29] the mathematical derivation for Small-Signal model of the synchronous generator is shown. An assumption mode is that the synchronous generator is connected to an infinite bus through a reactance x_e. In steady state conditions, the variations of the *dq-axis* fluxes $\frac{d\lambda_d}{dt}$ and $\frac{d\lambda_q}{dt}$ are usually neglected [28] , so the stator quantities will contain only fundamental frequency component, for representing the interconnecting transmission network as shown in the next Figure [26, 28]

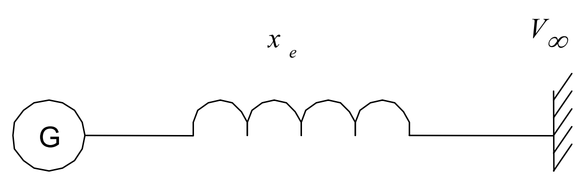

Figure 3.2 Synchronous Generator Connected to an Infinite Bus

Here x_e is the line reactance, and V_∞ is the voltage of the infinite bus. Another simplifying assumption normally made is that the per unit value of the rotor velocity ω_r is equal to 1.0 p.u. in the stator voltages equations. This is not the same as saying that speed is constant; it assumes that speed changes are small and do not have a significant effect on the voltage. Considering these assumptions described in [26, 28, 29] the incremental synchronous machine model for the electrical subsystem, linearized around an operating point is derived in [26] and given by

$$\frac{d\Delta E_q^{'}}{dt} = \frac{1}{T_{do}}\Delta E_q^{'} - \frac{xd - x'd}{T_{do}^{'} x_d^{"}}\Delta E_q^{"} + \frac{xd - x'd}{T_{do}^{'} x_d^{"}}\Delta V_q + \frac{k}{T_{do}^{'}}\Delta V_{fd} \tag{3.40}$$

$$\frac{d\Delta Eq''}{dt}=\left(\frac{1}{T''_{do}}-\frac{1}{T'_{do}}\right)\Delta E'q-\left(\frac{1}{T''_{do}}+\frac{xd-x'd}{T'_{do}x''d}+\frac{x'd-x''d}{T''_{do}x''d}\right)\Delta E''q$$
$$+\left(\frac{xd-x'd}{T'_{do}x''d}+\frac{x'd-x''d}{T''_{do}x''d}\right)\Delta Vq+\frac{k}{T'_{do}}\Delta V_{fd} \tag{3.41}$$

$$\frac{d\Delta E''_{d}}{dt}=-\frac{xq}{T''_{qo}x''_{q}}\Delta E''d-\frac{xq-x''d}{T''_{qo}x''_{q}}\Delta V_{d} \tag{3.42}$$

$$\Delta i_{d}=\frac{1}{x''_{d}}(\Delta E''_{q}-\Delta V_{q}) \tag{3.43}$$

$$\Delta i_{q}=\frac{1}{x''_{q}}\Delta E''_{d}+\frac{1}{x''_{q}}\Delta V_{d} \tag{3.44}$$

where $\Delta E'_{q}$ $\Delta E''_{q}$ $\Delta E''_{d}$, are the incremental *dq* transient and sub-transient voltages; $\Delta V_d, \Delta V_q, \Delta V_{fd}$ are the incremental *dq* and field input voltages; $\Delta i_d, \Delta i_q$ are the incremental *dq* output currents. $x_d, x_q, x'_d, x'_q, x''_d, x''_q$ are the transient and sub-transient *dq-axis* reactances; $T'_{do}, T''_{do}, T''_{qo}$, are the *dq-axis* transient and sub-transient time constants, k is the ratio of the mutual reactance of the direct axis and the field winding resistance given by $k = x_{md} / r_{fd}$. From the equations (3.40) to (3.44), we can identify the parameters of the synchronous generator electrical subsystem, which constitute the parameters to be estimated. These parameters are ordered in parameter vector α, as follows.

$$\boldsymbol{\alpha}=[x_d \quad x'_d \quad x''_d \quad T'_{d0} \quad T''_{do} \quad k \quad x_q \quad x''_q \quad T''_{qo}]^T \tag{3.45}$$

The electric subsystem has nine parameters. There is an additional parameter *k* in our model which relates the field winding with the direct axes. For stability analysis, these parameters involve the synchronous transient and subtransient conditions of the machine. It is important to know what these parameters represent when the synchronous machine is connected to a large system.

Suppose that a disturbance or fault happens when the machine is connected to an infinite bus like the one showed in Figure 3.2, so that, following the disturbance, currents are induced in the circuits of the machine rotor. Some of this induced rotor currents decay more rapidly than others. Machine parameters that influence rapidly decaying components are called the subtransient parameters ($x_d^{''}, T_{do}^{''}, x_q^{''}, T_{qo}^{''}$), while those influencing the slowly decaying components are called the transient parameters ($x_d^{'}, T_{d0}^{'}$) and those influencing sustained components are the *synchronous* parameters (x_d, k, x_q). The corresponding time constants determine the rate of decay of currents and voltages from the standard parameters used in specifying synchronous machine electrical characteristics.

During a rapid transient the limiting value of L_d is approached by the subtransient reactance $L_d^{''}$, which represents the effective inductance $\Delta\psi_d / \Delta i_d$ immediately following a sudden change. The quantity $L_d^{''}$ multiplied by the value of frequency leads to d-axis subtransient reactance $x_d^{''}$. In absence of damper winding, the limiting value of inductance is $L_d^{'}$, which leads to d-axis transient reactance. $x_d^{'}$.

The parameter vector shown in (3.45) will be used through this thesis and, it constitutes the parameter vector to be estimated.

3.3. SUMMARY

In this chapter, we developed the mathematical model of the synchronous generator. For synchronous machine identification, it is necessary to consider the parameters that involve the synchronous, transient and subtransient stages with the finality of reflecting the machine behavior under normal operation, before and after a perturbation. The importance of developing more complex machine models is evident because they permit to generate diagnosis signal sensitive to the occurrence of faults, as well as more accurate control and prediction of the machine behavior. In this regard, a synchronous machine model that introduces its steady-state and transient performance was developed. This led to the *dq* linearized small signal model of the synchronous generator that contains parameters corresponding to the steady state, transient and subtransient conditions.

In the next chapter we present a parameter estimation problem and parameter estimate condition analysis for the synchronous generator.

CHAPTER 4 PARAMETER ESTIMATION AND CONDITIONING ANALYSIS FOR THE SYNCHRONOUS GENERATOR MODEL

4.1. INTRODUCTION

One of the most common mathematical tools used to solve problems of optimization and parameters estimation is the least squares method. In general the goal of the parameter estimation is to compute estimates of the parameters of a system when data or measurements are available. The parameter estimates are chosen so as to minimize a measure of the error between the model predictions and the measurements. For linear models, the least squares parameter estimate can be found easily. When the model of a system is nonlinear the problem of parameter estimation becomes more difficult to solve. Here it is necessary to use iterative methods to compute the estimate; reference [38] presents a detailed explanation. This chapter looks at the nonlinear least squares parameter estimation for a synchronous generator. The chapter is organized as follows. First the estimation approach of the linearized state space of synchronous machine based on the output error is introduced. Second the parameter estimation problem is analyzed and finally the parameter estimation for the synchronous generator is presented.

4.2. OUTPUT ERROR FORMULATION

The linearized state space representation of synchronous machine model shown in equations (3.40)-(3.44) can be represented as follows [37]

$$\begin{aligned} \overrightarrow{\dot{x}(t)} &= A(\alpha)x(t) + B(\alpha)u(t) \\ y(t) &= C(\alpha)x(t) + D(\alpha)u(t) \end{aligned} \tag{4.1}$$

where $\alpha = [x_d \quad x_d^{'} \quad x_d^{''} \quad T_{d0}^{'} \quad T_{do}^{''} \quad k \quad x_q \quad x_q^{''} \quad T_{qo}^{''}]^T$ is the vector of parameters that we want to estimate, $u = [\Delta V_{fd} \quad \Delta V_d \quad \Delta V_q]^T$, is the input, $x(t) = [\Delta E_q^{'} \quad \Delta E_q^{''} \quad \Delta E''d]^T$ is the state, $y(t) = [\Delta i_d \quad \Delta i_q]^T$ is the output, and $A(\alpha), B(\alpha), C(\alpha), D(\alpha)$ are matrices given by

$$A(\alpha) = \begin{bmatrix} \dfrac{1}{T_{do}} & \dfrac{xd - x'd}{T_{do}^{'} x_d^{''}} & 0 \\ \left(\dfrac{1}{T_{do}^{''}} - \dfrac{1}{T_{do}^{'}}\right) & \left(\dfrac{1}{T_{do}^{''}} + \dfrac{xd - x'd}{T_{do}^{'} x''d} + \dfrac{x'd - x''d}{T_{do}^{''} x''d}\right) & 0 \\ 0 & 0 & \dfrac{xq}{T_{qo}^{''} x_q^{''}} \end{bmatrix} \quad B(\alpha) = \begin{bmatrix} 0 & \dfrac{xd - x'd}{T_{do}^{'} x_d^{''}} & \dfrac{k}{T_{do}^{'}} \\ 0 & \left(\dfrac{xd - x'd}{T_{do}^{'} x''d} + \dfrac{x'd - x''d}{T_{do}^{''} x''d}\right) & \dfrac{k}{T_{do}^{'}} \\ \dfrac{xq - x''d}{T_{qo}^{''} x_q^{''}} & 0 & 0 \end{bmatrix}$$

$$C(\alpha) = \begin{bmatrix} 0 & \dfrac{1}{x_d^{''}} & 0 \\ 0 & 0 & \dfrac{1}{x_q^{''}} \end{bmatrix} \qquad D(\alpha) = \begin{bmatrix} 0 & \dfrac{1}{x_d^{''}} & 0 \\ \dfrac{1}{x_q^{''}} & 0 & 0 \end{bmatrix}$$

(4.2)

The estimation approach is based on the output error method proposed in [26]. The output of the generator model is compared with the system and the parameters are estimated by minimizing a measure of the model prediction error

$$\mathbf{r}(\boldsymbol{\alpha}) = \mathbf{y} - \hat{\mathbf{y}}(\boldsymbol{\alpha}) \tag{4.3}$$

where $\hat{\mathbf{y}}(\boldsymbol{\alpha})$ is the N-vector of model predictions for the measurements and is given by

$$\hat{\mathbf{y}}(\alpha) = \begin{bmatrix} \hat{y}(t_1 / \alpha) \\ \vdots \\ \hat{y}(t_N / \alpha) \end{bmatrix} \quad (4.4)$$

and $\mathbf{y}$ is the N-vector of measurements. Notice that prediction error is a nonlinear function of the parameter vector $\boldsymbol{\alpha}$.

4.3. THE PARAMETER ESTIMATION PROBLEM

4.3.1. LINEAR LEAST SQUARES PROBLEM

Karl Friedrich Gauss formulated the principle of least squares at the end of the eighteenth century. He stated that the unknown parameters of a model can be chosen this way: *The sum of the squares of the difference between the actually observed and the computed values, multiplied by numbers that measure the degree of precision, is a minimum* [29].

Consider the problem of finding a vector $\mathbf{x} \in \Re^n$ such that

$$A\mathbf{x} = b \quad (4.5)$$

For a particular application A is $m \times n$ matrix, b represents the available measurements, and $\mathbf{x}$ is a vector of parameters to be estimated. For $m > n$, and Rank (A) = n, in general, there will be no solution to (4.5), so the linear least squares problem suggest to minimize the square norm of the error $e = A\mathbf{x} - b$ as follows

$$\hat{\mathbf{x}}_{\mathbf{LS}} = \arg\min_{x \in \Re^n} \| A\mathbf{x} - b \|_2 \quad (4.6)$$

where $\|.\|_2$ is the Euclidean norm or 2-norm and. The solution set of (4.6) is defined by

$$V = \left\{ \hat{\mathbf{x}} \in \Re^n \middle| A^T (A\hat{\mathbf{x}}_{\mathbf{LS}} - b) = 0 \right\}. \tag{4.7}$$

If Rank (A)= n, the columns of A are linearly independent and $\hat{\mathbf{x}}_{\mathbf{LS}}$ is given by

$$\hat{\mathbf{x}}_{\mathbf{LS}} = (A^T A)^{-1} A^T b\,. \tag{4.8}$$

4.3.2. NONLINEAR LEAST SQUARES PARAMETER ESTIMATION

Suppose that we have n observations (x_i ,y_i), i=1,2,...,m, from a fixed regresor nonlinear model with a known functional relationship f such that

$$y_i=f\,(x_i\,,\alpha^*)+\varepsilon_i \tag{4.9}$$

where E[ε_i]=0, $x_i \in \Re^n$ and the true value of $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$ is known to belong to a set $\Theta \subseteq \mathbb{R}^p$.

The least squares estimate of $\boldsymbol{\alpha}^*$ denoted by $\hat{\boldsymbol{\alpha}}$ minimizes the error sum of squares

$$V(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \left[y_i - f(x_i;\alpha) \right]^2 = \left\| \mathbf{y} - \mathbf{f}(\boldsymbol{\alpha}) \right\|^2 \tag{4.10}$$

where $\mathbf{y} = \left[y_1 \;\; y_2 \cdots y_n \right]^T$ and $\mathbf{f}(\boldsymbol{\alpha}) = \begin{bmatrix} f(x_1,\boldsymbol{\alpha}) \\ f(x_2,\boldsymbol{\alpha}) \\ \vdots \\ f(x_m,\boldsymbol{\alpha}) \end{bmatrix}$

$\mathbf{V}(\boldsymbol{\alpha})$ is a measure of model fit typically least squares

$$\mathbf{V}(\alpha) = \frac{1}{2} \left\| \mathbf{r}(\boldsymbol{\alpha}) \right\|^2 = \frac{1}{2} \sum_{i=1}^{N} \mathbf{r}_i^2 (\boldsymbol{\alpha}) \tag{4.11}$$

Notice that we can select different norms to measure the output error such as, grid norm, Euclidean norm, or Chebyshev norm [31].It should be distinguished that, unlike the linear least squares situation, $\mathbf{V}(\boldsymbol{\alpha})$ may have several local minima in addition to the global minimum $\hat{\boldsymbol{\alpha}}$. Like the linear problem, the parameter vector $\hat{\boldsymbol{\alpha}}$ estimates for the nonlinear case can be stated as the optimization problem,

$$\hat{\boldsymbol{\alpha}} = \underset{\alpha \in A}{\operatorname{argmin}} \mathbf{V}(\boldsymbol{\alpha}) \tag{4.12}$$

where each $f(x_i;\boldsymbol{\alpha})$ is differentiable with respect to $\boldsymbol{\alpha}$ and $\hat{\boldsymbol{\alpha}}$ will be a solution to

$$\mathbf{J}^T(\mathbf{y} - \mathbf{f}(\boldsymbol{\alpha})) = 0 \tag{4.13}$$

where $\mathbf{J} = \dfrac{\partial \mathbf{f}(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}}$ is the Jacobian matrix.

Solutions for nonlinear least squares problem are computed using iterative methods of which Gauss-Newton is among the most used [32]. Gauss Newton method start with an initial value of estimate, the next guess is then computed as.

$$\hat{\boldsymbol{\alpha}}_{i+1} = \hat{\boldsymbol{\alpha}}_i + \delta_i \mathbf{p}_i \tag{4.14}$$

where δ_i, is a scalar that fixes the step size in the Gauss Newton direction $\mathbf{p}_i$, which is calculate by solving the linear least squares problem

$$\mathbf{p}_i = \underset{\mathbf{p}}{\mathbf{arg\,min}} \left\| \mathbf{J}_i \mathbf{p} - r_i \right\| \tag{4.15}$$

where

$$\mathbf{J}_i = \mathbf{J}(\hat{\boldsymbol{\alpha}}_i) = \left. \frac{\partial r(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} \right|_{\alpha = \hat{\alpha}_i} \tag{4.16}$$

is the Jacobian of the residuals and $r_i = r(\hat{\boldsymbol{\alpha}}_i)$. Problems arise because poor practical identifiability causes this Jacobian to be ill conditioned and we need to deal with this in the estimation process. In this thesis, we study the use of Variable Dimension Gauss-Newton algorithm.

4.4. CONDITION ANALYSIS

Our approach to study parameter estimate sensitivity is by viewing the relation between data and parameter estimate as a mapping

$$\hat{\boldsymbol{\alpha}} = \mathbf{F}(\mathbf{y}) \tag{4.17}$$

This mapping is defined by the optimization problem. Depending on the optimization criterion and model structure this mapping can be defined explicitly or implicitly. The technique used here to analyze the conditioning of parameter estimates is based on local sensitivity analysis using relative condition numbers. The relative condition number measures the relative change in the parameter estimate caused changes in problem (see, e.g. [34]) and is given by

$$\frac{|\Delta\alpha_{\mathbf{i}}|}{|\alpha_{\mathbf{i}}|} = \mathbf{S}_{\mathbf{y}}^{\hat{\alpha}_{\mathbf{i}}} \frac{\|\Delta\mathbf{y}\|}{\|\mathbf{y}\|} \tag{4.18}$$

where

$$S_y^{\hat{\alpha}_i} = \frac{\|\mathbf{y}\|\|\Delta\mathbf{f}_i\|}{\|\hat{\boldsymbol{\alpha}}\|}\frac{\|\hat{\boldsymbol{\alpha}}\|}{|\hat{\alpha}_i|} \tag{4.19}$$

is the sensitivity function [26, 35, 42], $\nabla\mathbf{f}_i$ is the gradient of the i-th component of $\mathbf{F}$, and $\|\cdot\|$ is the 2-Norm or Euclidean norm. We will say that a parameter is *ill conditioned* if this number has a large magnitude and *well conditioned* otherwise [41]. We can see also that these numbers can give an idea of the quality of the estimates since good quality estimates will in general be associated with well-conditioned parameters. For nonlinear least square estimation, let $\hat{\boldsymbol{\alpha}} = \mathbf{F}_{\mathbf{NLS}}(\mathbf{y})$, the Jacobian for the nonlinear least square problem mapping is given by

$$\frac{\partial \mathbf{F}_{\mathrm{NLS}}}{\partial \mathbf{y}} = (\mathbf{J}'*\mathbf{J})^{-1} * \mathbf{J}' \tag{4.20}$$

where **J** is the Jacobian of the residuals defined previously.

Once ill-conditioning is identified, we can try to bring additional information into the problem to transform the ill-conditioned parameter estimation problem into a well-conditioned one. An approach to do this is the subset selection method proposed in [26, 41-43]. Subset selection is a methodology that is applied to parameter estimation in electric machines with the finality of overcoming ill-conditioning. A subset selection technique can be used as a way of incorporating prior information to the parameter estimation problem.

4.5. SYNCHRONOUS GENERATOR EXPERIMENTAL DATA EXAMPLE

We develop an example to study the synchronous generator parameter conditioning using the proposed sensitivity analysis. The synchronous generator under consideration is the FC5HP located at the Four Corners Generating Station of the Arizona Public Service Company (APS) [31]. The data used throughout this was provided by [31]. The measurements were taken under steady state conditions when the generator is serving a load. Voltages and currents of the stator and field windings were measured as well as the active and reactive power. The active power has a value of 142.2 MW and the reactive power is 3.7 MVA at a lagging power factor. Figure 4.1 and Figure 4.2 shows the voltage and current waveforms of the stator and Figure 4.3 and Figure 4.4 illustrates the current and voltage waveforms of the field winding

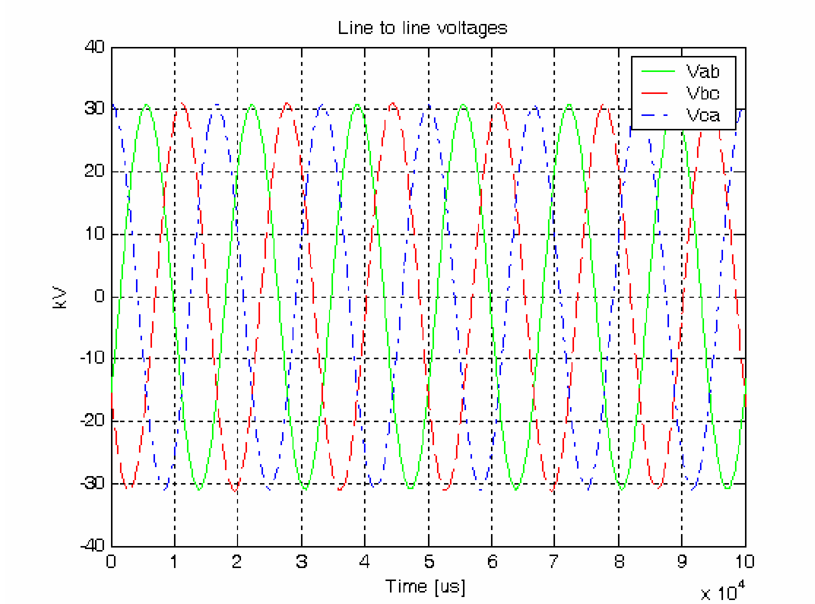

Figure 4.1 Steady estate operating condition, line to line stator voltages

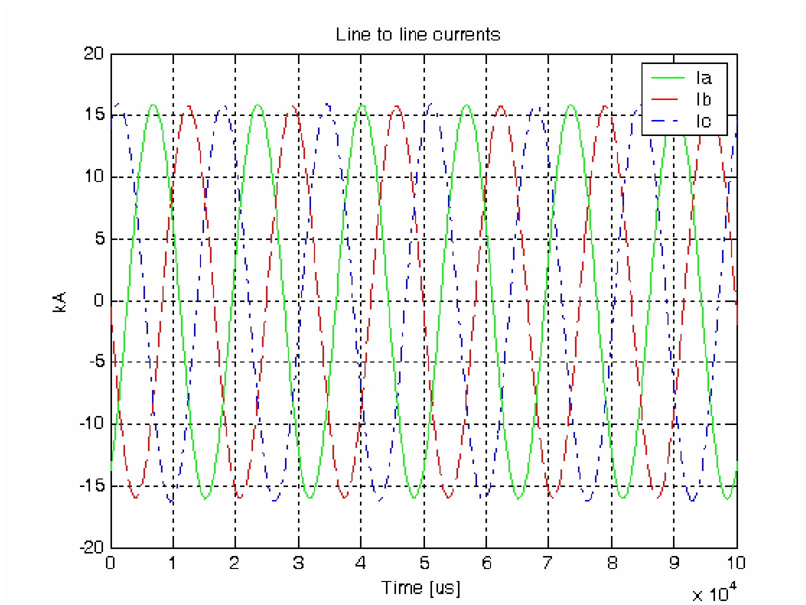

Figure 4.2 Steady estate operating condition, line to line stator currents

Notice that, the field voltage and currents are slowly varying DC signals, which are measured through a six pulse rectifier and thus their time plots have a rather unusual appearance as show in Figure 4.3.

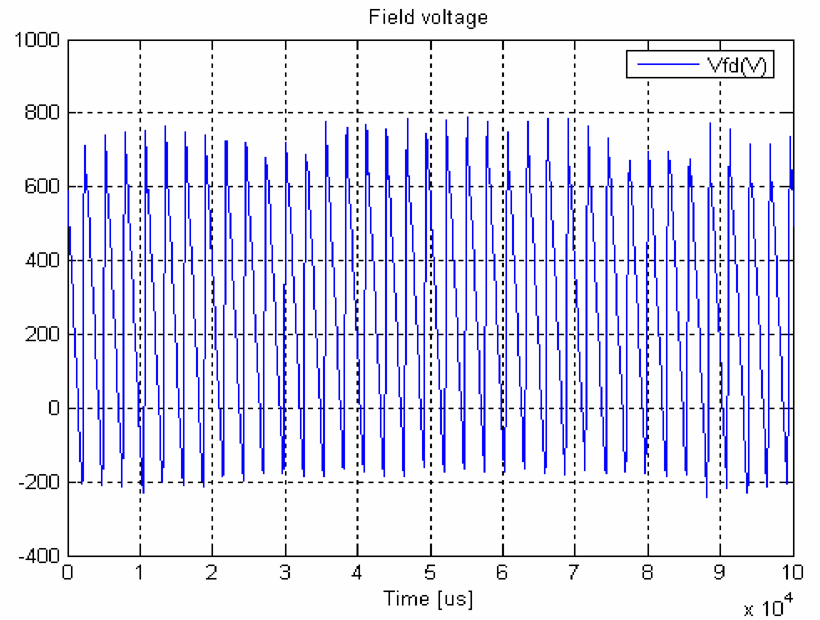

Figure 4.3 Steady state operating condition, field voltage

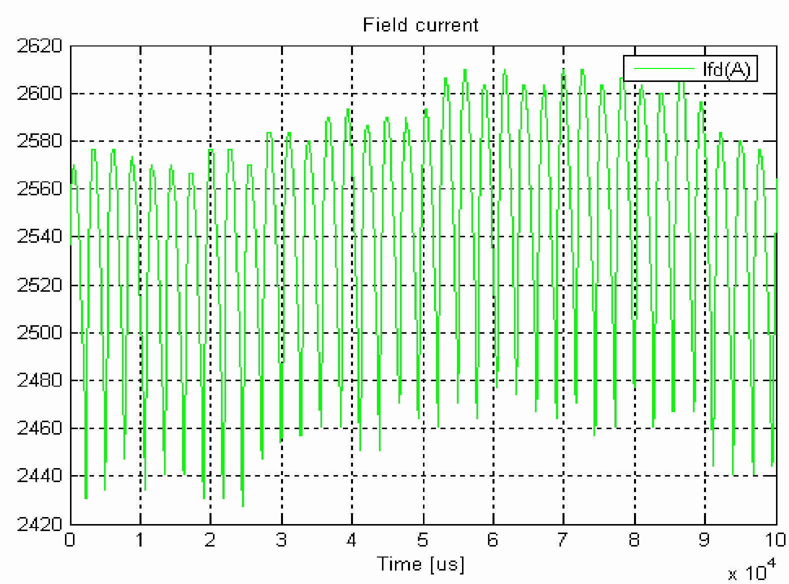

Figure 4.4 Steady state operating condition, field current

Appendix A shows the procedure of how the waveforms of the Figure 4.1-Figure 4.4 were obtained. The nameplate parameters of the FC5HP were extracted from [31] and used to calculate the nominal values of the parameters of the linearized model presented in Chapter 2. In this research we use the stator and voltage equations of the synchronous machine model connected to an infinite bus so as to compute the numerical values of nominal parameters [28, 30]. These values are shown in Table 4.1.

Table 4.1 Nominal parameter values for the 483MVA synchronous machine

Parameters	x_d	$x_d^{'}$	$x_d^{''}$	$T_{do}^{'}$	$T_{do}^{''}$	k	x_q	$x_q^{''}$	$T_{qo}^{''}$
Nominal Values (p.u.)	1.801	0.285	0.220	3.7	0.032	47.143	1.72	0.220	0.059

4.6. Computing the Jacobian

Recall that the Gauss-Newton algorithm needs the Jacobian matrix. To compute the Jacobian matrix, we start from (4.3), by deriving the output error $\mathbf{r}(\alpha)$ with respect to the parameter vector $\boldsymbol{\alpha}$, and is given by

$$\mathbf{J} = \left.\frac{\partial \hat{\mathbf{y}}_i(\alpha)}{\partial \alpha_r}\right|_{\alpha=\hat{\boldsymbol{\alpha}}} \tag{4.21}$$

where each row is given by

$$\mathbf{J}_i = \frac{\partial \mathrm{r}_i}{\partial \boldsymbol{\alpha}} = -\frac{\partial \hat{\mathrm{y}}_i}{\partial \boldsymbol{\alpha}} \tag{4.22}$$

and is computed by integrating the sensitivity equation

$$
\begin{aligned}
\frac{\partial \overrightarrow{\dot{\mathbf{x}}(\mathbf{t},\boldsymbol{\alpha})}}{\partial \hat{\alpha}_i} &= \mathbf{A}(\boldsymbol{\alpha})\frac{\partial \mathbf{x}(\mathbf{t})}{\partial \hat{\alpha}_i} + \frac{\partial \mathbf{A}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{x}(\mathbf{t}) + \frac{\partial \mathbf{B}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{u}(\mathbf{t}) \\
\frac{\partial \mathbf{y}(\mathbf{t},\boldsymbol{\alpha})}{\partial \hat{\alpha}_i} &= \mathbf{C}(\boldsymbol{\alpha})\frac{\partial \mathbf{x}(\mathbf{t})}{\partial \hat{\alpha}_i} + \frac{\partial \mathbf{C}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{x}(\mathbf{t}) + \frac{\partial \mathbf{D}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{u}(\mathbf{t})
\end{aligned}
\tag{4.23}
$$

Notice that the last equation can constitute another system of equations, if we assume that $\mathbf{S}_i = \frac{\partial \mathbf{x}}{\partial \hat{\alpha}_i}$ [37]. Then (4.23) can be rewritten as follows

$$
\begin{aligned}
\overrightarrow{\dot{\mathbf{s}}_{\mathbf{i}}^{\mathbf{x}}} &= \mathbf{A}(\boldsymbol{\alpha})\mathbf{S}_i + \frac{\partial \mathbf{A}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{x}(\mathbf{t}) + \frac{\partial \mathbf{B}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{u}(\mathbf{t}) \\
\overrightarrow{\dot{\mathbf{s}}_{\mathbf{i}}^{\mathbf{y}}} &= \frac{\partial \mathbf{y}(\mathbf{t})}{\partial \hat{\alpha}_i} = \mathbf{C}(\boldsymbol{\alpha})\mathbf{S}_i + \frac{\partial \mathbf{C}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{x}(\mathbf{t}) + \frac{\partial \mathbf{D}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i}\mathbf{u}(\mathbf{t})
\end{aligned}
\tag{4.24}
$$

or equivalently in state space form

$$
\begin{aligned}
\overrightarrow{\dot{\mathbf{s}}_{\mathbf{i}}^{\mathbf{x}}} &= \mathbf{A}(\boldsymbol{\alpha})\mathbf{x} + \begin{bmatrix} \frac{\partial \mathbf{A}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i} & \frac{\partial \mathbf{B}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i} \end{bmatrix} \begin{bmatrix} \mathbf{x}(\mathbf{t}) \\ \mathbf{u}(\mathbf{t}) \end{bmatrix} \\
\overrightarrow{\dot{\mathbf{s}}_{\mathbf{i}}^{\mathbf{y}}} &= \frac{\partial \mathbf{y}(\mathbf{t})}{\partial \hat{\alpha}} = \mathbf{C}(\boldsymbol{\alpha})\mathbf{x} + \begin{bmatrix} \frac{\partial \mathbf{C}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i} & \frac{\partial \mathbf{D}(\boldsymbol{\alpha})}{\partial \hat{\alpha}_i} \end{bmatrix} \begin{bmatrix} \mathbf{x}(\mathbf{t}) \\ \mathbf{u}(\mathbf{t}) \end{bmatrix}
\end{aligned}
\tag{4.25}
$$

and

$$
\mathbf{J} = -\begin{bmatrix} \frac{\partial \hat{\mathbf{y}}(t_i)}{\partial \boldsymbol{\alpha}} \\ \vdots \\ \frac{\partial \hat{\mathbf{y}}(t_N)}{\partial \boldsymbol{\alpha}} \end{bmatrix}
\tag{4.26}
$$

To get a numerical value for the Jacobian matrix all we need to do is to perform the simulation of equation (4.25), which is evaluated at the nominal values of the parameters shown in Table 4.1. Observe the new inputs of the system described in (4.25), they constitute the input $u = [\Delta V_{fd} \quad \Delta V_d \quad \Delta V_q]^T$ and the state variables $x(t) = [\Delta E_q^{'} \quad \Delta E_q^{"} \quad \Delta E"d]^T$ of the system described in (4.1), which have already been computed in the previous section.

4.7. Full Order Parameter Estimation

In this section, we evaluate the sensitivity of full order synchronous machine parameter estimation problem. The idea is to show how small amount of noise are greatly amplified into the parameter estimates due to the problem of ill-conditioning. For this purpose, the formulation presented in (4.1)-(4.16) and the nominal values shown in Table 4.1 were used so as to find the parameter estimates of the synchronous machines [26].

A Matlab code was implemented for this purpose and is shown in Appendix B. The obtained values shown in Table 4.2 were obtained by using the Gauss Newton algorithm after 6 iterations.

Furthermore, making a relative comparison of the estimated values, we can see that some parameters have error percentages relatively large in comparison with others. In particular x_d, $x_d^{"}$ and *k* are the most sensitive of all followed by $T_{d0}^{'}$. Observe that the error percentage corresponding to the *q* axis components are small in comparison with the others.

Table 4.2 Full order synchronous machine parameter estimation

Parameters	Nominal Values $\boldsymbol{\alpha}_{N}^{i}$	Parameters Estimates $\hat{\boldsymbol{\alpha}}_{E}^{i}$	Error (%) $\mathbf{E} = \left\lvert \frac{\boldsymbol{\alpha}_{N}^{i} - \hat{\boldsymbol{\alpha}}_{E}^{i}}{\boldsymbol{\alpha}_{N}^{i}} \right\rvert * 100\ \%$
x_d	**1.801**	**1.799**	**4.78**
$x_d^{'}$	0.285	0.285	0.01
$x_d^{''}$	**0.220**	**0.212**	**3.636**
$T_{d0}^{'}$	3.700	3.696	0.098
$T_{d0}^{''}$	0.032	0.032	0.000
k	**47.143**	**47.098**	**9.440**
x_q	1.720	1.720	0.000
$x_q^{''}$	0.220	0.220	0.000
$T_{q0}^{''}$	0.059	0.0590	0.000

4.8. SENSITIVITY ANALYSIS OF THE PARAMETERS TO BE ESTIMATED

Using (4.19) the *componentwise condition numbe*rs for full-order parameter estimation were calculated and shown in Table 4.3

Table 4.3 Componentwise condition numbers for full order parameter estimation problem

Parameter	$\mathbf{S_y^{\hat{\alpha}_i}}$
$\hat{x}_q$	$3.1597*10^1$
$\hat{T}_{qo}^{"}$	$2.1088*10^2$
$\hat{x}_d^{'}$	$3.0700*10^2$
$\hat{x}_d^{"}$	$6.5274*10^2$
$\hat{x}_q^{"}$	$1.0750*10^3$
$\hat{T}_{do}^{"}$	$1.8140*10^3$
$\hat{x}_d$	$8.9896*10^3$
$\hat{k}$	$1.8937*10^4$
$\hat{T}_{do}^{'}$	$1.9956*10^4$

From the results shown in Table 4.3, we can see that x_q has the smallest condition number. The remaining parameters have condition numbers that are in the order of 10^2 through 10^4. Thus from these results, we can classify the parameters in two groups: well conditioned and ill-conditioned. Making a relative comparison we can say that the parameters k and $T_{d0}^{'}$ are the ill-conditioned ones and the remaining parameter can be considered as well-conditioned. If we observe the results of the full order Gauss-Newton parameter estimation for the synchronous machine showed in Table 4.2, we can see that the parameters that have the largest error percentage are the parameters x_d, k and $T_{do}^{'}$, which are among the worst conditioned parameters in Table 4.3, These results show that the *componentwise condition numbers* [26, 35, 41, 42] are practical and easy way to determine the condition of the parameter estimate, prior to perform parameter estimation. In this way, we can determine which parameter can be reliably estimated from the available data.

Based in the results obtained in Table 4.3, we want to develop parameter estimation strategies that deal with the high sensitivity values and carry out reliable parameter estimation for the synchronous generator. The first strategy to be evaluated is an extended analysis of subset selection technique, previously presented by [26]; first it is necessary to find which parameter to fix. Once we know which parameter to fix, the subset selection strategy can be applied.

4.9. SUMMARY

In this Chapter, the synchronous generator parameter estimation is presented; we also introduced the conditioning analysis of the parameter estimation problem. The idea of output error and least squares methodology were used to solve the problem of nonlinear parameter estimation. The Gauss Newton method was used to compute the parameter estimates. A numerical experiment of the synchronous machine was presented, the input data correspond to the FC5HP synchronous generator located at the Four Corners Generating Station of the Arizona Public Service Company (APS) and the simulation model of [26]. The measurements were taken under steady state conditions when the generator was serving a load. The Gauss Newton method was applied and the estimates shown in Table 4.2. Making a relative comparison of the estimated values, we could see that some parameters have error percentages relatively large in comparison with others. In particular, x_d and k have the highest error followed by $T_{d0}^{'}$. Observe that the error percentage corresponding to the q axis components are small in comparison with the others. In this chapter, we also developed the study of parameter conditioning using the *componentwise condition numbers*. The numerical results of the synchronous machine example revealed relatively large condition numbers.

After a relative comparison we were able to identify that parameters k and $T_{d0}^{'}$ were the ill-conditioned ones and the remaining parameter were considered well-conditioned. If we observe the results of the full order Gauss-Newton parameter estimation for the synchronous machine showed in Table 4.2, we can see that the parameters that have the largest error percentage are the parameters x_d, k and $T_{do}^{'}$, which are among the worst conditioned parameters in Table 4.3

The next chapter presents the subset selection technique as a way to deal with ill-conditioning problem.

CHAPTER 5 SUBSET SELECTION ANALYSIS

As shown previously, local sensitivity analysis techniques to identify well-conditioned and ill-conditioned parameters in the synchronous generator model was used Table 4.3 showed these results. The question that arises from this analysis is: given such value of parameter sensitivity, how reliable the parameter estimates will result? How well the parameter estimates will converge to the desired values?

In Table 4.2, it was demonstrated that the parameter estimation from the available measurements is not totally unreliable. However the sensitivity values shown in Table 4.3 have shown that some parameters (x_d, k, and $T_{d0}^{'}$) present considerable high condition numbers. Therefore, it is desired to explore some methodologies to overcome such relatively high sensitivity values. In that sense, this chapter and the next will be focused on presenting some of these techniques and their application to synchronous machine parameter estimation when real data are used. In particular, we want to apply the subset selection technique developed in [26] and the Variable Dimension Newton Raphson method, which is discussed in detail in Chapter 6.

In the subset selection strategy [26] and [43], a subset of parameters is fixed to prior values while the remaining parameters are estimated from available data. Here we applied the procedure to determine how many parameters to fix by looking at the eigenstructure of the Hessian matrix. On the other hand, an extensive evaluation to determine which combination of parameters is the best subset to fix such that the condition numbers are reduced and therefore improve the parameter estimation problem.

5.1. ILL-CONDITIONING PROBLEM

Before treating the problem of ill conditioning, we want to introduce other kind of approach used in the Gauss-Newton algorithm to compute the parameter estimates. This approach consists on expressing the Hessian matrix in terms of the Jacobian matrix. Hessian matrix is approximated in [41] and [44] as follows

$$\mathbf{H}(\alpha) = \mathbf{J}^T(\boldsymbol{\alpha})\mathbf{J}(\boldsymbol{\alpha}) + \sum_{l=1}^{N} r_l(\alpha)\frac{\partial r_l(\alpha)}{\partial\alpha\partial\alpha'} \tag{5.1}$$

where $\mathbf{H}$ is the Hessian matrix, which involves the second derivatives of the fit model $\mathbf{V}(\alpha)$ with respect to the parameter vector $\boldsymbol{\alpha}$. $\mathbf{J} = \left.\frac{\partial \mathbf{r}_i(\hat{\alpha})}{\partial \alpha_r}\right|_{\alpha=\hat{\boldsymbol{\alpha}}}$ is the Jacobian matrix of the residuals $\mathbf{r}(\alpha)$. Notice that the Hessian matrix is a $n \times n$ matrix. For small residuals the Hessian can be approximated by

$$\mathbf{H}(\boldsymbol{\alpha}) \approx \mathbf{J}^T(\boldsymbol{\alpha})\mathbf{J}(\boldsymbol{\alpha}) \tag{5.2}$$

Observe the simplicity of the equation (5.2), once the Jacobian is computed; the calculation of the Hessian matrix is not difficult. Basically, this matrix contains the information for computing the parameter estimates. An important issue is to evaluate the eigenvalues of the Hessian matrix, its nature and properties of inversion

5.2. SUBSET SELECTION STRATEGY

Subset Selection is a methodology that has been applied to parameter estimation in electric machines with the finality of overcoming ill conditioning [26].

This strategy consists of adding prior information to the parameter estimation problem, selecting a subset of parameters and fixes them to prior values and still performs meaningful estimation of the remaining parameters [41-43]. In [41] and [43], the subset selection method was applied to induction motor and synchronous generator parameter estimation problems respectively. The number of parameters to fix was determined by eigenanalysis of the Hessian matrix and which parameters to fix were determined by the subset selection method for determining column independence in matrices presented in [43]. As in [26], in this research, we expand the study and perform the full combinatorial analysis to determine the best combination or combinations of parameters to be fixed.

Once the parameters to fix are determined, the following constrained optimization problem is solved.

$$\begin{gathered}\hat{\boldsymbol{\alpha}} = \underset{\alpha}{\operatorname{argmin}} \|\mathbf{r}(\boldsymbol{\alpha})\|_2^2 \\ \text{Subject to } \alpha_2 = \alpha_2^*\end{gathered} \tag{5.3}$$

where $\boldsymbol{\alpha} = (\alpha_1^T, \alpha_2^T)^T$ is a partitioning of the parameter vector where $\alpha_1 \in \Re^p, \alpha_2 \in \Re^q, p + q = n$ and α_1 is the subset of parameters to estimate and α_2 is the subset of parameters to fix. Notice that the Jacobian for the reduced-order problem (5.3) consists of a subset of the columns of the Jacobian for full-order problem (4.12).

5.2.1. QR Factorization and Subset Selection

The orthogonal-triangular QR factorization of an $m \times n$ matrix A where $m \geq n$, is given by

$$\boldsymbol{A}=\boldsymbol{QR} \tag{5.4}$$

where $Q \in \Re^{m \times m}$ is orthonormal and $R \in \Re^{m \times n}$ is upper triangular.

The singular value decomposition of a matrix $A \in \Re^{m*n}(m \geq n)$ is given by

$$U^T A V = diag(\sigma_1, \ldots, \sigma_n) \tag{5.5}$$

where $U = (\mathbf{u}_1, \cdots, \mathbf{u}_n) \in \Re^{m\times n}$ and $V = (\mathbf{v}_1, \cdots, \mathbf{v}_n) \in \Re^{n\times n}$ are orthonormal matrices $U^T U = V^T V = I_n$ .Next the subset selection strategy of [43]; is presented

5.3. SUBSET SELECTION ALGORITHM [41]

1. Given an initial parameter vector estimate $\hat{\boldsymbol{\alpha}}_0$, compute the Jacobian matrix of the residuals $\mathbf{J}$ and compute the Hessian Matrix by using (5.2)
2. Calculate the eigendescomposition of

$$\begin{aligned} \mathbf{H}(\hat{\alpha}_0) &= \mathbf{J}^{\mathbf{T}}\mathbf{J} = \mathrm{V}\Lambda\mathrm{V}^T \\ &= \mathrm{V}\Sigma^2\mathrm{V} \end{aligned} \tag{5.6}$$

   where $\Lambda = diag\{\lambda_1(\mathbf{H}), \lambda_2(\mathbf{H}), \ldots \lambda_n(\mathbf{H})\}$

3. Determine p, the number of parameter to estimate, such that the first p eigenvalues of $\mathbf{H}$ are larger than a threshold.
4. Make the partition $\mathrm{V} = [\mathrm{V}_p \quad \mathrm{V}_{n-p}]$ with V_p containing the first p columns of V.
5. Compute the QR decomposition with column-pivoting for $\mathrm{V}_p^{\mathrm{T}}$ [52]

$$\mathrm{V}_p^{\mathbf{T}}\mathbf{P} = \mathbf{QR} \tag{5.7}$$

6. Let $\overline{\boldsymbol{\alpha}} = \mathbf{P}^{\mathbf{T}}\boldsymbol{\alpha}$. The first p elements of this vector are the parameters to estimate, the *n-p* are the parameters to fix

As we mentioned previously, the Jacobian of the reduced order problem contains a subset of the columns of the full order problem and hence the Hessian is a sub-matrix of the full order Hessian. Steps 4-5 of the Algorithm above, guarantee that the reduced order Hessian is well conditioned. Here steps 4-5 are substituted by a combinatorial search and search for the best combination of *p*-column of the Jacobian that will result in the best conditioned Hessian. This combinatorial search is feasible in this low dimension problem, in high dimension problems the algorithm just presented is the way to go. Next the application of the modified subset selection to the synchronous machine parameter estimation is presented.

5.4. SYNCHRONOUS MACHINE REAL DATA EXAMPLE

The first step of the technique presented above suggests the computation of the eigendescomposition of the Hessian matrix, and is given by

$$
H = \begin{bmatrix}
2.46*10^{-3} & 4.79*10^{-2} & 3.48*10^{-3} & 6.57*10^{-3} & 1.27*10^{-3} & -5.8*10^{-4} & 0 & 0 & 0 \\
4.79*10^{-2} & 1.03*10^{+0} & 1.05*10^{-1} & 1.37*10^{-1} & -4.9*10^{-2} & -1.2*10^{-2} & 0 & 0 & 0 \\
3.48*10^{-3} & 1.06*10^{-1} & 3.02*10^{-2} & 1.35*10^{-2} & -3.7*10^{-2} & -1.1*10^{-3} & 0 & 0 & 0 \\
6.58*10^{-3} & 1.37*10^{-1} & 1.35*10^{-2} & 1.85*10^{-2} & -4.1*10^{-3} & -1.6*10^{-3} & 0 & 0 & 0 \\
1.27*10^{-3} & -4.9*10^{-2} & -3.7*10^{-2} & -4.1*10^{-3} & 7.00*10^{-2} & 3.01*10^{-4} & 0 & 0 & 0 \\
-5.8*10^{-4} & -1.2*10^{-2} & -1.2*10^{-2} & -1.6*10^{-3} & 3.01*10^{-4} & 1.44*10^{-4} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3.43*10^{-3} & 1.19*10^{-3} & -2.8*10^{-2} \\
0 & 0 & 0 & 0 & 0 & 0 & 1.19*10^{-3} & 5.29*10^{-4} & -1.1*10^{-2} \\
0 & 0 & 0 & 0 & 0 & 0 & -2.8*10^{-2} & -1.1*10^{-2} & 2.38*10^{-1}
\end{bmatrix}
$$

and its eigenvalues are shown in Table 5.1

Table 5.1 Eigenvalues of the Hessian

i	λ_i
1	1.068
2	$2.423*10^{-1}$
3	$8.349*10^{-2}$
4	$3.297*10^{-3}$
5	$2.384*10^{-4}$
6	$5.653*10^{-5}$
7	$7.326*10^{-6}$
8	$1.895*10^{-7}$
9	$2.101*10^{-13}$

The larger eigenvalue is 1.068 and the smaller is $2.101*10^{-13}$. Thus the condition number of the Hessian, would be $5.082*10^{12}$, which is relatively large, meaning that H might be close to singularity and nonlinear least square solution might become sensitive to perturbations in the input data. On the other hand this can lead to *severe* ill-conditioning and slow convergence as shown in [26]. However in our application this is not the case, the relative high sensitivity values shown in Chapter 3 (Table 3.3) do not dramatically affect the full order parameter estimation. Although this does not represent a severe problem, it is desired to evaluate the subset selection algorithm and perhaps obtain even more precise convergence and therefore smaller absolute errors of the estimates.

In this regard, looking at the results shown in Table 4.1, the last six eigenvalues are considerably smaller compared with the other two. If we set a threshold at 10^{-6} to neglect the eigenvalues, this will indicate that 7 parameters can be estimated more reliably from the available data. However we still do not know which parameter to fix and which to estimate.

The next step would correspond to investigate what parameters of the parameter vector to estimate and what parameters to fix. For this purpose we apply the subset selection strategy to our test system (synchronous generator). Remember that the parameter vector to be estimated with its respective ordering is

$$\boldsymbol{\alpha} = [\underset{\substack{\downarrow\\1}}{x_d} \quad \underset{\substack{\downarrow\\2}}{x_d^{'}} \quad \underset{\substack{\downarrow\\3}}{x_d^{''}} \quad \underset{\substack{\downarrow\\4}}{T_{d0}^{'}} \quad \underset{\substack{\downarrow\\5}}{T_{do}^{''}} \quad \underset{\substack{\downarrow\\6}}{k} \quad \underset{\substack{\downarrow\\7}}{x_q} \quad \underset{\substack{\downarrow\\8}}{x_q^{''}} \quad \underset{\substack{\downarrow\\9}}{T_{qo}^{''}}]^T \tag{5.8}$$

applying steps 5 and 6 of the subset selection algorithm described in section 5.3, the permutation vector **P** can be obtained and it gives

$$\mathbf{P} = \begin{bmatrix} 7 & 8 & 9 & 5 & 3 & 2 & 4 & 6 & 1 \end{bmatrix} \tag{5.9}$$

which results in the following parameter partitions

$$\overline{\boldsymbol{\alpha}} = \boldsymbol{\alpha}(\mathbf{P}) = \left[\underbrace{x_q \quad x_q^{''} \quad T_{qo}^{''} \quad T_{do}^{''} \quad x_d^{''} \quad x_d^{'} \quad T_{do}^{'}}_{\alpha_1} \quad \underbrace{k \quad x_d}_{\alpha_2} \right] \tag{5.10}$$

According to (4.10), k and x_d are the parameters to be fixed. This result is quite consistent with results shown in Table 4.3. As it was stated previously, this research presents a more extended analysis of subset selection, which consists on evaluating all parameter combinations of the permutation vector **P** when two (α_2) parameters are fixed. Then evaluate the Hessian matrix conditioning for such combinations. The results of the combinatorial analysis for this problem are shown in Table 5.2. The first and third column of the table give the parameter combination being considered and the second and fourth give the condition number of the reduced order Hessian for the particular combination.

Table 5.2 Conditioning for all possible combinations

COMBINATIONS							$v(\mathbf{H})$	COMBINATIONS							$v(\mathbf{H})$
9	5	3	2	4	6	1	5.0822E+012	7	8	5	3	2	6	1	4.5299E+007
8	5	3	2	4	6	1	5.0822E+012	7	8	5	3	2	4	1	5.6404E+006
8	9	3	2	4	6	1	7.1267E+008	7	8	5	3	2	4	6	2.3227E+008
8	9	5	2	4	6	1	1.0502E+009	7	8	9	2	4	6	1	1.9429E+007
8	9	5	3	4	6	1	7.6684E+008	7	8	9	3	4	6	1	6.5703E+006
8	9	5	3	2	6	1	4.5299E+007	7	8	9	3	2	6	1	2.2703E+006
8	9	5	3	2	4	1	5.6404E+006	7	8	9	3	2	4	1	3.2107E+005
8	9	5	3	2	4	6	2.3227E+008	7	8	9	3	2	4	6	2.0560E+007
7	5	3	2	4	6	1	5.0822E+012	7	8	9	5	4	6	1	1.1472E+006
7	9	3	2	4	6	1	7.1267E+008	7	8	9	5	2	6	1	3.5313E+005
7	9	5	2	4	6	1	1.0502E+009	7	8	9	5	2	4	1	1.4497E+005
7	9	5	3	4	6	1	7.7120E+008	7	8	9	5	2	4	6	2.6403E+006
7	9	5	3	2	6	1	4.5299E+007	**7**	**8**	**9**	**5**	**3**	**6**	**1**	**4.5769E+004**
7	9	5	3	2	4	1	5.6404E+006	**7**	**8**	**9**	**5**	**3**	**4**	**1**	**3.3078E+004**
7	9	5	3	2	4	6	2.3227E+008	7	8	9	5	3	4	6	5.3409E+005
7	8	3	2	4	6	1	7.1267E+008	7	8	9	5	3	2	1	1.4451E+005
7	8	5	2	4	6	1	1.0502E+009	7	8	9	5	3	2	6	2.4727E+005
7	8	5	3	4	6	1	6.3205E+008	7	8	9	5	3	2	4	1.4561E+005

In Table 5.2 we identify two combinations with the smallest condition numbers. These are the combinations [7 8 9 5 3 6 1] and [7 8 9 5 3 4 1]. From this, we can infer that fixing the combinations, $x_d^{'}$ and $T_{d0}^{'}$; $x_d^{'}$ and k respectively, the condition number of the Hessian matrix is reduced significantly in comparison with others combinations. Notice the importance of evaluating all parameter vector combinations. Here we can obtain potentially good combinations and select those where prior information is available or of that have better quality. Furthermore it allows us to observe patterns that can give more insight into the parameter conditioning issue. Now let us evaluate the conditioning of the estimates for the two best combinations given above and observe how the condition numbers change. As in Chapter 3, here we use equation (3.12). The results are shown in Table 5.3

Table 5.3 Parameter sensitivity for the better combinations obtained from subset selection. (NOTE: Hyphen denotes fixed parameters)

Parameter	Condition Numbers		
	Fixing x'_d, T'_{d0}	Fixing x'_d, k	Full Order GN
x_d	$7.8623*10^1$	$7.2516*10^1$	$8.9896*10^3$
x'_d	-	-	$3.0700*10^2$
x''_d	$3.3749*10^1$	$3.3676*10^1$	$1.0750*10^3$
T'_{do}	-	$1.2633*10^1$	$1.9956*10^4$
T''_{do}	$2.2476*10^2$	$2.2587*10^2$	$1.8140*10^3$
k	$1.2149*10^1$	-	$1.8937*10^4$
x_q	$3.1597*10^1$	$3.1597*10^1$	$3.1597*10^1$
x''_q	$6.5274*10^2$	$6.5274*10^2$	$6.5274*10^2$
T''_{qo}	$2.1088*10^2$	$2.1088*10^2$	$2.1088*10^2$

Observe that the parameter conditioning is improved (reduced) by fixing the combinations $x_d^{'}$, $T_{d0}^{'}$ and $x_d^{'}$, k respectively.

The next step would be to estimate the parameters for the reduced order case. Here we also use the Gauss-Newton algorithm and the initial conditions for parameter estimation as presented in Chapter 3. A Matlab code was also implemented for this case and is shown in Appendix B. Table 5.4 shows the parameter estimates for the reduced order case, and Table 5.5 illustrates the error percentages of the reduced and full order cases.

Table 5.4 Parameter estimates for the better combinations obtained from subset selection. (Note: Hyphen denotes fixed parameters)

Parameter	Parameter Estimates			
	Nominal Values	Fixing $x_d^{'}, k$	Fixing $x_d^{'}, T_{d0}^{'}$	Full Order GN
x_d	1.801	1.800	1.800	**1.799**
$x_d^{'}$	0.285	-	-	0.285
$x_d^{''}$	0.220	0.220	0.220	**0.212**
$T_{do}^{'}$	3.700	3.700	-	3.696
$T_{do}^{''}$	0.032	0.032	0.032	0.032
k	47.143	-	47.143	**47.098**
x_q	1.720	1.717	1.691	1.720
$x_q^{''}$	0.220	0.219	0.216	0.220
$T_{qo}^{''}$	0.059	0.059	0.059	0.0590

Table 5.5 Error of estimated parameters for the better combinations obtained from subset selection. (Note: Hyphen denotes fixed parameters)

Parameter	Error (%)of Estimated Parameters			
	Nominal Values	Fixing $x_d^{'}, k$	Fixing $x_d^{'}, T_{d0}^{'}$	Full Order GN
x_d	4.78	0.055	0.055	**4.78**
$x_d^{'}$	0.010	-	-	0.01
$x_d^{''}$	0.008	0.000	0.000	**3.636**
$T_{do}^{'}$	0.098	0.000	-	0.098
$T_{do}^{''}$	0.000	0.000	0.000	0.000
k	9.440	-	0.000	**9.440**
x_q	0.000	0.169	1.700	0.000
$x_q^{''}$	0.000	0.169	1.700	0.000
$T_{qo}^{''}$	0.000	0.000	0.000	0.000

The parameter estimates of Table 5.4 converged after eight and seven iterations respectively. In Table 5.5 note that fixing the parameters combination $x_d^{'}$, $T_{d0}^{'}$, and $x_d^{'}$, k results in considerable error percentages reduction that the full order Gauss Newton (GN) parameter estimation.

5.5. SUMMARY

In this Chapter we studied the Subset Selection strategy. This strategy consists of adding prior information to the parameter estimation problem, selecting a subset of parameters and fixing them to prior values and still performs meaningful estimation of the remaining parameters. Therefore the number of parameters to fix was determined by eigenanalysis of the Hessian matrix. Which parameters to fix were determined by the subset selection method for determining column independence in matrices. In this chapter, we also studied and implemented the combinatorial analysis to determine the best combination or combinations of parameters to be fixed. Once the parameters to fix were determined, the constrained optimization problem described in equation (5.3) was solved. The application of this methodology to the synchronous machine parameter estimation problem was then presented. The application of the combinatorial analysis produced two combinations with the smallest condition numbers corresponding to fixing the combinations, $x_d^{'}$, $T_{d0}^{'}$; and k respectively. The condition number of the Hessian matrix is reduced significantly in comparison with others combinations. Then the conditioning of the estimates for the two best combinations was evaluated. It was observed that the parameter conditioning was improved condition numbers (reduced) by fixing the combinations $x_d^{'}$, $T_{d0}^{'}$ and $x_d^{'}$, k respectively. Then the parameter estimates for the reduced order case were evaluated using the Gauss-Newton algorithm.

In general, we can conclude that the estimates and the performance of the Subset Selection algorithm are considerably improved in comparison to the ones obtained for the full order Gauss Newton algorithm (Table 4.2). Furthermore by fixing the parameters combination $x_d^{'}$, $T_{d0}^{'}$, k and $x_d^{'}$ results in high sensitivity and error percentages reduction as shown in Table 5.4, and Table 5.5 respectively .

The next Chapter is focused on an alternative algorithm that performs the nonlinear parameter estimation. The algorithm is based on the Variable Dimension Newton Raphson (VDNR) method. Our intention is to compare different computational strategies capable of overcoming ill conditioning and therefore provide different alternatives for making reliable parameter estimation and perform the fault detection analysis.

.

CHAPTER 6 PARAMETER ESTIMATION USING VARIABLE DIMENSION GAUSS-NEWTON METHOD

6.1. INTRODUCTION

Ill-conditioning in parameter estimation problem affects directly the parameter identification. It is evident that additional strategies are required to mitigate ill-conditioning and compute more reliable parameter estimates. In the previous chapter, it was shown how the subset selection strategy gives reliable estimates and reduces problem sensitivity.

In this chapter, a new algorithm to handle the ill-conditioned parameter estimation problem is presented. The new algorithm is based on variable dimension Newton-Raphson method [47-50] developed for the solution of circuit equation. The Generalized Newton Raphson method (GNR) is developed in [48] in order to solve the problem of the singular Jacobian of a nonlinear systems of equations and [50] used it as a core to further develop a Variable Dimension Newton Raphson (VDNR). According to [50], the VDNR method has a much better convergence property than the classical NR and is applicable to circuits with highly nonlinear behavior. In this section, the development of the variable dimension Gauss-Newton method (VDGN) will be presented. Finally the VDGN is applied to the synchronous machine parameter estimation problem. We compare its performance with that of the Gauss-Newton method applied to the full problem and to the reduced order problem from Subset Selection.

Since VDNR can handle problems with singular Jacobian, we expect that the proposed VDGN method will be able to handle ill conditioned parameter estimation problem with singular (or nearly singular) Hessian matrices.

6.2. FORMULATION OF NEWTON-RAPHSON METHOD

In this section the conventional Newton Raphson Method (NR) for non linear equation [50] is described. First we consider the problem of finding a solution or all the solutions for the set of nonlinear equation systems

$$\mathbf{F}(\boldsymbol{\alpha}) = \begin{bmatrix} f_1\left(\boldsymbol{\alpha}^1, \cdots, \boldsymbol{\alpha}^n\right) \\ \vdots \\ f_n\left(\boldsymbol{\alpha}^1, \cdots, \boldsymbol{\alpha}^n\right) \end{bmatrix} = 0 \tag{6.1}$$

where

$$\begin{aligned} \mathbf{S_i} &= \left\{\boldsymbol{\alpha} \notin \Re^{\mathbf{n}} \mid f_i(\boldsymbol{\alpha}) = \mathbf{0}\right\} \\ \mathbf{S} &= \left\{\boldsymbol{\alpha} \in \Re \mid \mathbf{F}(\boldsymbol{\alpha}) = \mathbf{0}\right\} = \bigcap_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \mathbf{S_i} \end{aligned} \tag{6.2}$$

The classical Newton-Raphson method is obtained by linearizing **F.** linearization is also a means of constructing iterative methods to solve (6.1). If we assume that $\alpha = \mathbf{z}$ is a zero for **F,** that α_0 is an approximation to z and that **F** is differentiable for $\alpha = \alpha_0$ then to first approximation of the Taylor series expansion of **F** about $\boldsymbol{\alpha}_0$

$$\mathbf{0} = \mathbf{F}(\mathbf{z}) = \mathbf{F}(\alpha_0) + \mathbf{DF}(\alpha_0)(\mathbf{z} - \alpha_0), \tag{6.3}$$

where

$$\mathbf{DF}(\alpha_0) = \left.\frac{\partial \mathbf{F}}{\partial \alpha}\right|_{\alpha=\alpha_0} = \begin{bmatrix} \frac{\partial f_1}{\partial \alpha_1} & \cdots & \frac{\partial f_1}{\partial \alpha_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial \alpha_1} & & \frac{\partial f_n}{\partial \alpha_n} \end{bmatrix}_{\alpha=\alpha_0} ,(z-\alpha_0) = \begin{bmatrix} z^1 - \alpha_0^1 \\ \vdots \\ z^n - \alpha_0^n \end{bmatrix} 0 \tag{6.4}$$

If the Jacobian $\mathbf{DF}(\alpha_0)$ is nonsingular, then the equation

$$\mathbf{F}(\alpha_0) + \mathbf{DF}(\alpha_0)(\alpha_1 - \alpha_0) = 0 \tag{6.5}$$

can be solved for α_1

$$\alpha_1 = \alpha_0 - \left(\mathbf{DF}(\alpha_\mathbf{0})\right)^{-1} \mathbf{F}(\alpha_\mathbf{0}) \tag{6.6}$$

and α_1 may taken as a closer approximation to the zero z. The generalized Newton method for solving system of equations (6.1) is given by

$$\hat{\boldsymbol{\alpha}}_{i+1} = \hat{\boldsymbol{\alpha}}_i - \gamma \left[\left.\frac{\partial \mathbf{F}}{\partial \boldsymbol{\alpha}}\right|_{\boldsymbol{\alpha}=\boldsymbol{\alpha}_i} \right]^{-1} \mathbf{F}(\boldsymbol{\alpha}_i) \tag{6.7}$$

where γ is the step length.

Geometrically this method projects a linear system onto $\mathbf{F}(\alpha)$ at the point $(\alpha_n, \mathbf{F}(\alpha_n))$. (See Figure 6.1), we obtained the linear system $\mathbf{F}(\alpha) = \mathbf{F}(\alpha_n) + \mathbf{DF}(\alpha_n)(\alpha - \alpha_n)$ that cut $\mathbf{F}(\alpha) = 0$ axis at the point $\alpha = \alpha_n - (\mathbf{DF}(\alpha_n))\mathbf{F}(\alpha_n)$ that is the $\hat{\alpha}_{i+1}$ of the Newton-Rhapson formula (6.7)

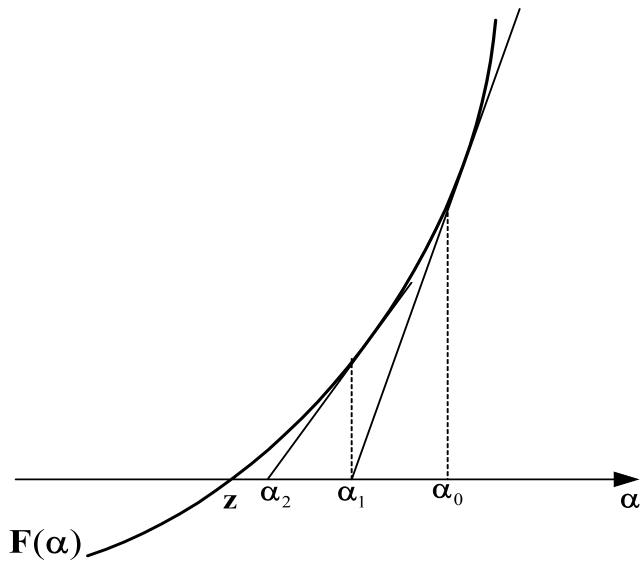

Figure 6.1 Newton-Raphson Method

Throughout this analysis we assume that the problem $\mathbf{F}(\boldsymbol{\alpha}) = 0$ has at least one solution.

The NR method is well known for its good convergence rate but its convergence is only local and may require a very good initial guess of the solution.

Various variants that can improve the convergence property have been proposed [48-49]. The homotopy method is one of them [55]. The basic principle of the homotopy method is to transform the original hard problem into a simpler problem that it easy to solve. The transformation is defined by a homotopy map. The homotopy method is globally convergent, provided that a correct homotopy map is chosen. A poor choice leads to numerous problems in the tracing of the solution, these problems include the presence of bifurcations, infinite solutions, abbreviate path and closely spaced solution curves. While most of them are solvable by sophisticated path-following algorithms, the best strategy is to use another homotopy map. However homotopy maps may not work for highly ill-conditioned problems. Thus an alternative approach to solve this problem is to use variable dimension Newton –

Raphson method (VDNR) that was originally proposed in [50]. It is based in the Generalized Newton method of Ben-Israel in [48] and then independently discovered in [50]. The advantage of the Generalized Newton method over the classical method is its enlarged convergence region. This method and the application of solution tracking will be discussed in the next section.

6.3. FORMULATION OF THE GENERALIZED NEWTON-RAPHSON METHOD AND THE SOLUTION TRACKING

The conventional NR method described above can be generalized to find one of the solutions of a nonsquare nonlinear problem $\mathbf{F_m}(\alpha)$

$$\mathbf{F_m} = \begin{bmatrix} f_1(\alpha) \\ \vdots \\ f_m(\alpha) \end{bmatrix} = 0 \tag{6.8}$$

with $m \le n$., $f_1(\alpha)\cdots f_m(\alpha)\cdots f_n(\alpha)$ is the first *m*-component of **F.** Since $m < n$, the solution set forms a *n-m* dimensional hypersurface. The basis iteration step of the Generalized Newton Raphson method (GNR) is [50]

$$\hat{\boldsymbol{\alpha}}_{\mathrm{i+1}} = \hat{\boldsymbol{\alpha}}_{\mathrm{i}} - \mathbf{J}^{+}_{\mathrm{F,i}}\mathbf{F_m}(\boldsymbol{\alpha}_{\mathrm{i}}) \tag{6.9}$$

where the Jacobian

$$\mathbf{J}_{\mathrm{F,i}} = \left.\frac{\partial \mathbf{F_m}}{\partial \boldsymbol{\alpha}}\right|_{\alpha=\alpha_{\mathrm{i}}} \tag{6.10}$$

On the assumption that $\mathbf{J}_{\mathrm{F,i}}$ has rank *m* at $\boldsymbol{\alpha} = \boldsymbol{\alpha}_{\mathbf{i}}$, $\mathbf{J}^{+}_{\mathbf{F,i}}$ is the pseudoinverse of $\mathbf{J}_{\mathbf{F,i}}$ defined to be

$$\mathbf{J}^{+}_{\mathrm{F,i}} = \mathbf{J}^{\mathrm{T}}_{\mathrm{F,i}}\left(\mathbf{J}_{\mathrm{F,i}}\mathbf{J}^{\mathrm{T}}_{\mathrm{F,i}}\right)^{\mathbf{-1}} \tag{6.10}$$

The method above described is locally convergent. When m is small, $\mathbf{J}^{+}_{\mathrm{F,i}}$ can be calculated efficiently by equation (6.10). However, when m is large (6.10) become inefficient because $\mathbf{J}_{\mathrm{F,i}}\mathbf{J}^{\mathrm{T}}_{\mathrm{F,i}}$ is dense. The authors in [50] demonstrates that the equation (6.9) can be transformed into (6.11) and (6.12) below and (6.11) can be solved efficiently by modified Doolittle LU decomposition

$$\Delta\boldsymbol{\alpha}_{\mathrm{i}} = \mathbf{M}^{-1}y \tag{6.11}$$

where
$$y = \left[\mathbf{F_m}(\boldsymbol{\alpha}_{\mathrm{i}})^{\mathrm{T}}, 0\right]^{\mathrm{T}}$$

$$\boldsymbol{\alpha}_{\mathrm{i+1}} = \boldsymbol{\alpha}_{\mathrm{i}} \quad \Delta\boldsymbol{\alpha}_{\mathrm{i}} \tag{6.12}$$

The application of the GNR gives only one of the solution of $\mathbf{F_m}(\alpha) = 0$. To find the solution of the complete problem $\mathbf{F_n}(\alpha) = \mathbf{F}(\alpha) = 0$, we must track the solution of $\mathbf{F_m}(\alpha) = 0$ in the correct direction until we meet a solution of $\mathbf{F_{m+1}}(\alpha) = 0$. The desired solution may, however, be another point on the *n-m* dimensional hyper surface. Assume that

- There is a continuous path between the desired solution and the starting point;
- The path is on the hypersurface;
- $(\mathbf{J}_{\mathrm{F}})$ has full row rank along the path

The desired solution can be found by repeated application of the following prediction and correction step. At the $\mathbf{k}+1$th **tracking** step we have

Prediction step: $$\boldsymbol{\alpha}_{\mathrm{p}} = \boldsymbol{\alpha}_{\mathrm{k}} + \left(\mathbf{I} + \mathbf{J}^{+}_{\mathrm{F,k}}\mathbf{J}_{\mathrm{F,k}}\right)\xi \tag{6.13}$$

Correction steps: $$\boldsymbol{\alpha}_{\mathrm{c.j+1}} = \boldsymbol{\alpha}_{\mathrm{c,j}} - \mathbf{J}^{+}_{\mathrm{F,c,j}}F_m\left(\alpha_{\mathrm{c,j}}\right) \tag{6.14}$$

where $\boldsymbol{\alpha}_{c,0} = \boldsymbol{\alpha}_p$ and j is the iteration count of the number of the correction steps applied . The vector $\boldsymbol{\xi}$ specifies the direction of the tracking and the size of the prediction step. The prediction step projects $\boldsymbol{\xi}$ onto the tangent plane of $\mathbf{F_m}(\boldsymbol{\alpha}_k) = \mathbf{0}$ at point $\boldsymbol{\alpha}_k$ by $\left(\mathbf{I} - \mathbf{J}^+_{\mathbf{F,k}} \mathbf{J}_{\mathbf{F,k}}\right)$ [53], in effect, the prediction step perturbs $\boldsymbol{\alpha}_k$ in the direction of the desired solution. The correction step brings the perturbed solution back onto the $\mathbf{F_m}(\alpha) = 0$ hypersurface. Thus, $\boldsymbol{\alpha}_{k+1}$ will be closer to desired solution than $\boldsymbol{\alpha}_k$. The correction step, which is actually repeated application of equation (6.9), should be continued until

$$\max_{i=1\ldots m} \left| f_i(\boldsymbol{\alpha}_{c,j}) \right| < \varepsilon_f \tag{6.15}$$

where $\varepsilon_{\mathbf{f}}$ is a user defined tolerance .

It is obvious that the prediction and correction step mentioned above are locally convergent if $\|\xi\|_\infty$ is not small enough and the error criterion (6.14) is not satisfied even in the number of iterations has exceeded an upper bound. We may restart the tracking step using a smaller ξ. If ξ become too small and the path is too difficult to track, the method may choose to stop.

In the case $\left(\mathbf{I} + \mathbf{J}^+_{F,k} \mathbf{J}_{F,k}\right)\xi = \mathbf{0}$, the tracking fails completely. At this point the $\mathbf{F_m}(\alpha) = 0$ hypersurface is turning and becomes orthogonal to ξ. By the proof in [50], we know that there is a linear dependency between ξ and one or more rows of $\mathbf{J}_{F,k}$. We must then use (6.16) and (6.17) to find out which row of $\mathbf{J}_{F,k}$ has a linear dependence with ξ. In effect, we know which function is the source of the trouble.

When the algorithm is applied to a large scale system, the tracking step $\left(\mathbf{I} - \mathbf{J}_{\mathrm{F,k}}^{+}\mathbf{J}_{\mathrm{F,k}}\right)\xi$ cannot be computed efficiently. Since the prediction step is started with $\mathbf{F_m}\left(\alpha_{\mathbf{k}}\right) \approx \mathbf{0}$, In[50] it is proved that the prediction step can be simplified to $\boldsymbol{\alpha}_{\mathrm{p}} = \boldsymbol{\alpha}_{\mathrm{k}} + \xi$. Then the failure condition $\left(\mathbf{I} - \mathbf{J}_{\mathrm{F,k}}^{+}\mathbf{J}_{\mathrm{F,k}}\right)\xi = \mathbf{0}$ should be replaced by

$$\min_{\mathrm{i=m+1...n}} \frac{\left(\alpha_{k+1}^{i} - \alpha_{k}^{i}\right)}{\xi_i} < \varepsilon_t \tag{6.16}$$

This criterion ensures that the tracking is advancing in the direction of ξ, the parameter $\varepsilon_{\mathbf{t}}$ is a user defined relative tolerance on the size of the corrected tracking step. When the algorithm fails to track the solution along the path and result in the removal of a function from tracking, the new tracking step may detect a zero crossover on the function just removed as shown in Figure 6.2 therefore, this crossover should be ignored.

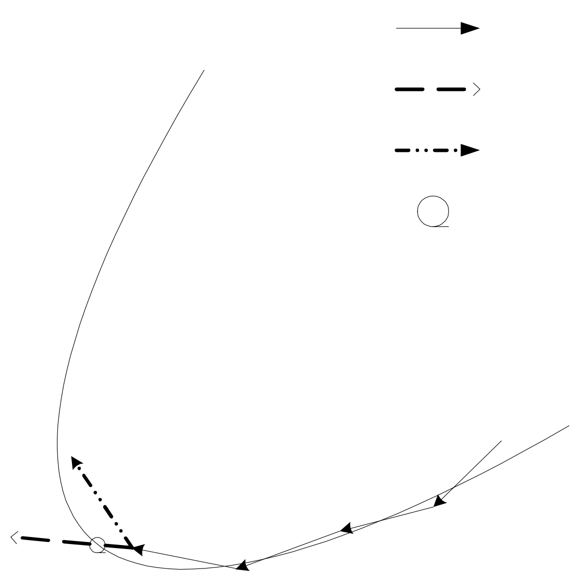

Figure 6.2 Example of false zero crossover of f_j

If the corrected tracking step is very small, it can be concluded that the hypersurface $\mathbf{F_m}(\alpha)=0$ at the point $\boldsymbol{\alpha}_k$ is nearly normal to ξ. Therefore, it is inefficient or impossible to track in the direction specified by ξ.

When the tracking step fails, it is necessary to know which of the functions under tracking should be removed. By making use of the knowledge that at least one of the functions to be removed can be computed by

$$\xi^T \mathbf{M}^{-1} = \left[\gamma^T, 0\right] \tag{6.17}$$

Then the corresponding function of each $|\gamma_{\mathbf{k}}| > \mathbf{0.9}\|\gamma\|_\infty$ should be removed, that is

$$\beta = \left\{ \mathrm{k} : |\gamma_{\mathrm{k}}| > \mu \|\gamma\|_\infty \right\} \text{ where } 0 < \mu < 1 \tag{6.18}$$

It is important to notice that the tracking method described above is only an outline of how the GNR method can be used to move around the $\mathbf{F_m}(\alpha)=0$ hypersurface. In practical implementations, the adaptive adjustment of the size of ξ may involve a complex step control method [50]. The GNR method and the associated tracking method described above form the core of the VDNR method and will be discussed in the next section.

6.4. THE VARIABLE DIMENSION NEWTON-RAPHSON (VDNR) METHOD

Variable dimension methods belong to the class of homotopy methods, but it is the dimension of the problem which is deformed rather than the equations [47]. This transformation is applicable to any problem and is computationally attractive because of its simplicity. From the point of view of parameter estimation problem, a fixed number of variables correspond to the number of degrees of freedom of the system modeled by the objective function (i.e. the model is fixed with respect to the structure of the problem). As soon as structural optimization is required, the number of parameters being estimated becomes a variable itself.

Thus we can say that the basic approach of the variable dimension Newton Raphson method is to solve the succession of systems $\mathbf{F}_1(\boldsymbol{\alpha}) = \mathbf{0}, \ldots, \mathbf{F}_\mathbf{m} = \mathbf{0}, \ldots$ until $\mathbf{F}_\mathbf{n}(\boldsymbol{\alpha}) = \mathbf{F}(\boldsymbol{\alpha}) = \mathbf{0}$, where $\mathbf{F}_\mathbf{m}(\boldsymbol{\alpha}) = [f_1(\alpha) \ldots f_m(\alpha)]^T$. The number of equations being solved will be increased or decreased accordingly to track the solution. When we have $\mathbf{m} \le \mathbf{n}$, $\mathbf{F}_\mathbf{m}(\boldsymbol{\alpha})$ has one or more than one solutions. The solution set forms a $\mathbf{n - m}$ dimensional hypersurface.

Although the VDNR method does not inherit the global convergence of the homotopy methods, it will not diverge to infinity. The success of the VDNR method relies on the following assumptions [50].

(i) $\mathbf{J}_\mathbf{m}$ has full row rank along the search path for all $\mathbf{m}$

(ii) $\mathbf{F}(\alpha)$ is twice continuously differentiable

Assumption (i) may easily be fulfilled by adding a small diagonal matrix to $\mathbf{J}_\mathbf{m}$. This is a common practice to avoid matrix singularity. Alternatively one may use a nonsingular approximation to $\mathbf{J_m}$, since the solution sets of $\mathbf{F_m}(\alpha)=0$ is a hypersurface, the use of an inexact $\mathbf{J}_{\mathbf{m}}^{+}$ has no effect in the VDNR method. However this approximated $\mathbf{J_m}$ should not produce a correction step which is opposite to the search direction. Assumption (ii) is necessary to ensure the convergence of the VDNR method. This condition is identical to the convergence criterion of the classical Newton Raphson method [44].

Similar to other homotopy methods, variable dimension methods have the advantage of being globally convergent on eventually passive systems [47].An additional advantage is that there is no need to choose a proper homotopy map.

6.4.1. The Variable Dimension Gauss –Newton Method

In the nonlinear least squares problems analyzed in section (4.3.2) before, the parameter estimate is chosen so that it minimizes the nonlinear cost function

$$\mathbf{V}(\boldsymbol{\alpha})=\frac{1}{2}\left\|\mathbf{y}-\hat{\mathbf{y}}(\boldsymbol{\alpha})\right\|_2^2 \tag{6.19}$$

$$\hat{\boldsymbol{\alpha}}=\underset{\alpha}{argmin}\,\mathbf{V}(\boldsymbol{\alpha}) \tag{6.20}$$

where $\hat{\boldsymbol{\alpha}}$ satisfy

$$\mathbf{J}_{\mathrm{r}}^{\mathrm{T}}(\mathbf{y}-\hat{\mathbf{y}}(\boldsymbol{\alpha}))=\mathbf{F}(\boldsymbol{\alpha})=0 \tag{6.21}$$

And $\mathbf{J}_{\mathrm{r}}=\dfrac{\partial\hat{\mathbf{y}}}{\partial\boldsymbol{\alpha}}$ is the Jacobian matrix of the residuals. For if **V** is differentiable, we can say that the each minimum point $\hat{\alpha}$ of $\mathbf{V}(\alpha)$ is a zero of the gradient $\mathbf{g}(\hat{\alpha})=\dfrac{\partial\mathbf{V}(\hat{\alpha})}{\partial\alpha}=0$.

Conversely each zero of equation (5.1) is also the minimum point of $\mathbf{V}$ [52], this is

$$\mathbf{F}(\hat{\boldsymbol{\alpha}}) = \mathbf{g}(\hat{\alpha}) = \frac{\partial \mathbf{V}(\hat{\alpha})}{\partial \alpha} = \mathbf{J}_{\mathbf{r}}^{\mathbf{T}} \mathbf{r}_{\mathbf{i}}(\alpha) = 0 \tag{6.22}$$

and the Jacobian $\mathbf{J}_{\mathbf{F}}$ [c.f. (6.10)]

$$\mathbf{J}_{\mathbf{F}} = \frac{\partial \mathbf{F}(\alpha)}{\partial \alpha} = \frac{\partial^2 \mathbf{V}(\alpha)}{\partial \alpha^2} = \mathbf{H}(\alpha) = \mathbf{J}_{\mathbf{r}}^{\mathbf{T}} \mathbf{J}_{\mathbf{r}} + \mathbf{A} \tag{6.23}$$

where (c.f. (5.1))

$$\mathbf{A} = \mathbf{A}(\alpha) = \sum_{\mathbf{i}=1}^{\mathbf{n}} \mathbf{r}_{\mathbf{i}}(\alpha) \frac{\partial^2 \mathbf{r}_{\mathbf{i}}(\alpha)}{\partial \alpha \partial \alpha^{\mathbf{T}}} \tag{6.24}$$

The Newton-Raphson step that solve (6.22) is [44]

$$\alpha_{\mathbf{i}+1} = \alpha - [\mathbf{H}^{-1} \mathbf{g}] \tag{6.25}$$

where the step

$$\mathbf{p} = -\mathbf{H}^{-1} \mathbf{g} \tag{6.26}$$

Thus the Newton step is

$$\mathbf{p} = -(\mathbf{J}_{\mathbf{r}}^{\mathbf{T}} \mathbf{J}_{\mathbf{r}} + \mathbf{A})^{-1} \mathbf{J}_{\mathbf{r}} \mathbf{r} \tag{6.27}$$

The Gauss Newton algorithm is obtained from Newton algorithm by ignoring part of the Hessian, namely $\mathbf{A}(\alpha)$ of (6.24) and becomes

$$\hat{\alpha}_{\mathbf{i}+1} = \hat{\alpha} - (\mathbf{J}_{\mathbf{r}}^{\mathbf{T}} \mathbf{J}_{\mathbf{r}})^{-1} \mathbf{J}_{\mathbf{r}} \mathbf{r}. \tag{6.28}$$

Although we have written the Gauss-Newton step in the form (6.28), in good implementation $\mathbf{J}_{\mathbf{r}}^{\mathbf{T}} \mathbf{J}_{\mathbf{r}}$ is never formed. Instead $\mathbf{p}$ is obtained by solving the linear least-squares problem (4.15)

$$\mathrm{p}_i = \arg\min_{\mathrm{p}} \left\| \mathrm{J}_{\mathbf{r}} \mathrm{p} - \mathbf{r}_{\mathbf{i}} \right\|.$$

Thus the VDNR method that solve problems with singular Jacobian is derived to Variable dimension Gauss-Newton algorithm (VDGN) to handle ill conditioned parameters estimation problem with singular or nearly singular Hessian matrices

6.4.2. FLOWCHART FOR VDGN ALGORITHM

A flowchart showing the basic implementation of the VDGN algorithm is shown in Figure 6.3 and Figure 6.4.

There are three important variables in this algorithm.

i) ρ is the set of indexes of functions which are solved, i.e., $f_i(\boldsymbol{\alpha}) = 0 \ \forall \ i \in \rho$

ii) β is the set of indexes of functions which have been removed from ρ by Step 12.

iii) h is the size of the search step.

The use of β in this algorithm is to avoid false inclusion of the same function after a function is removed from ρ in Step 12. This false inclusion happens when the algorithm restarts tracking from the wrong side of the function as shown in Figure 6.2

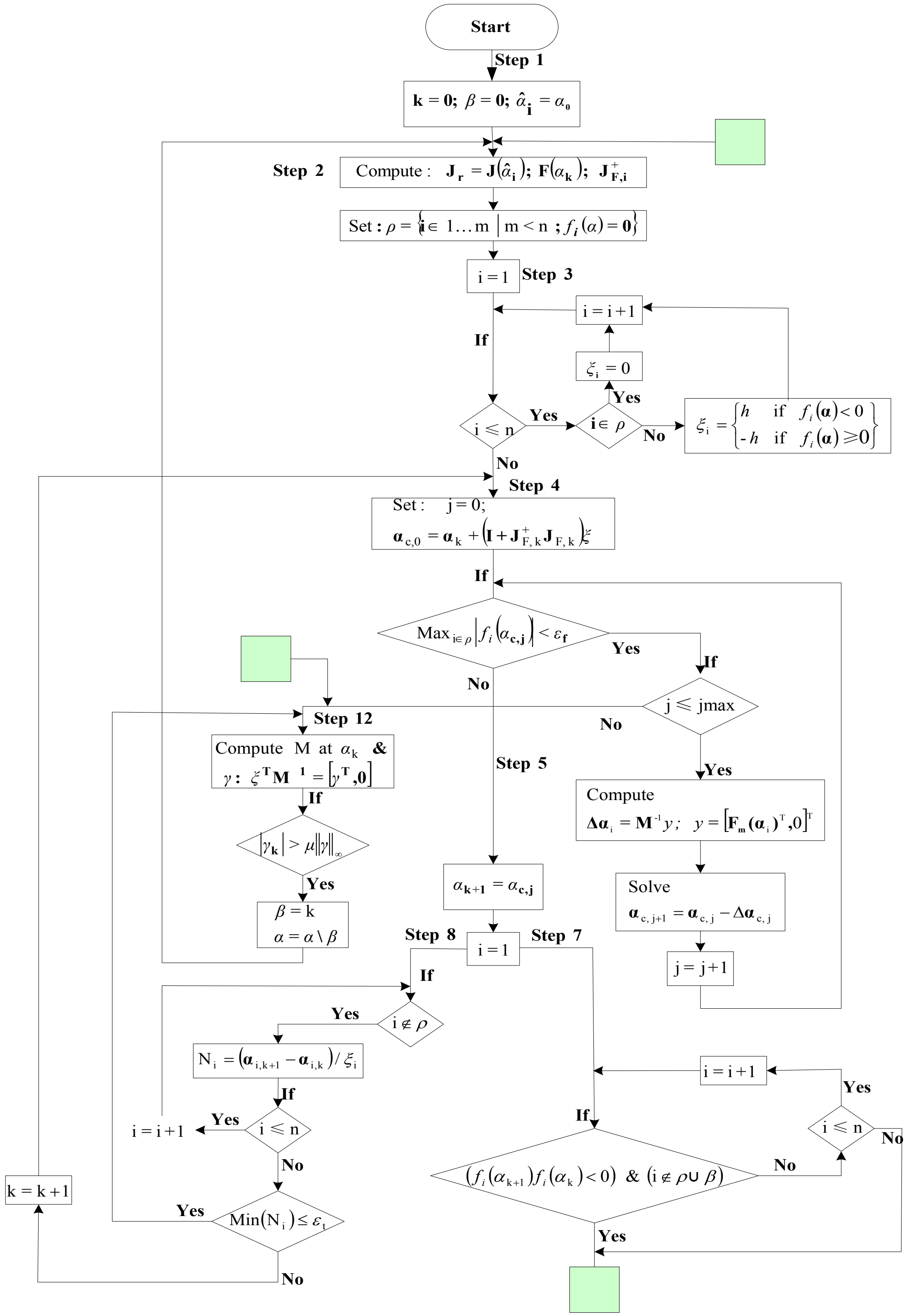

Figure 6.3 Flow Chart part 1

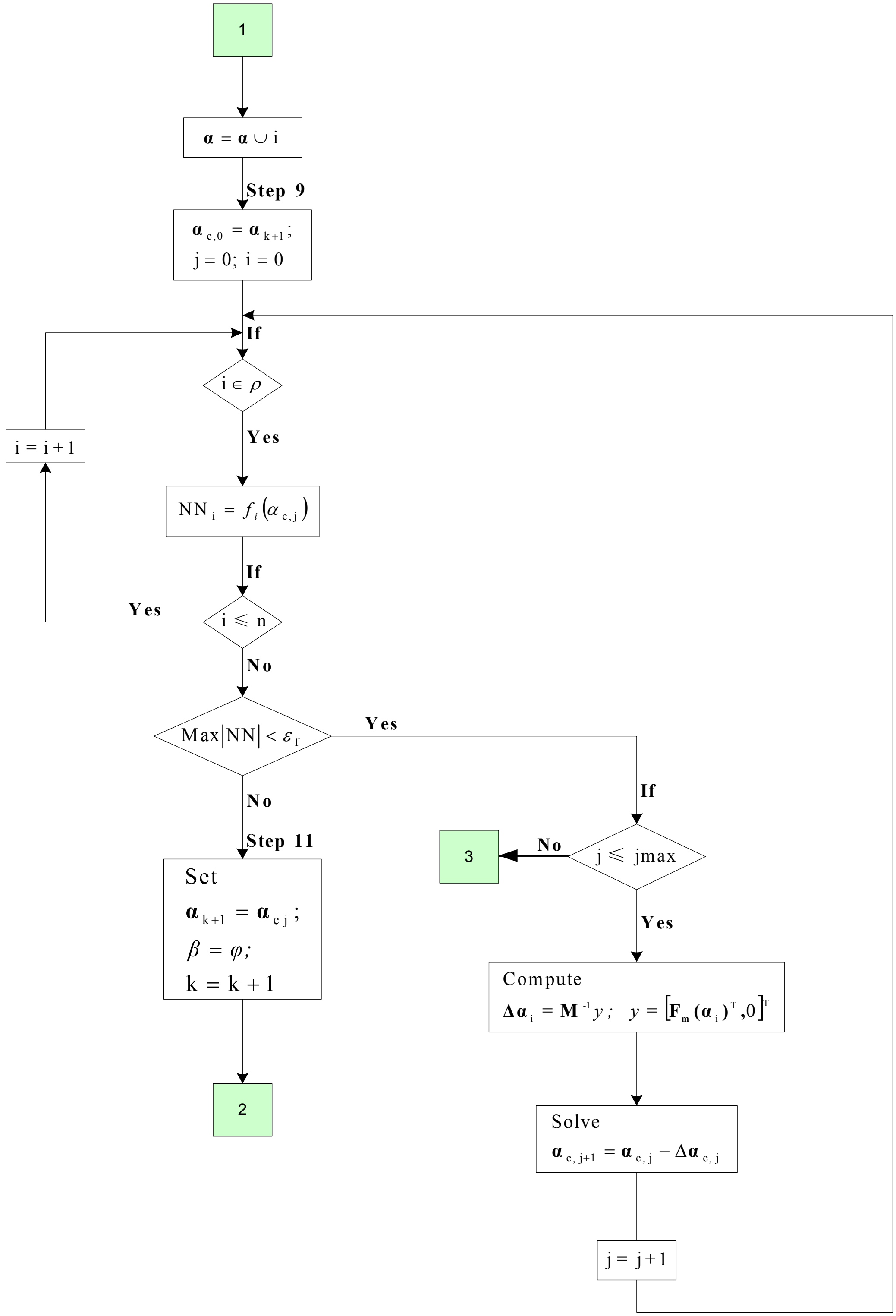

Figure 6.4 Flow Chart part 2

Step 4 in the flowchart of Figure 6.4 is the tracking step. Step 7 detects the zero crossover in any function. Step 9 ensures that the new set of $\mathbf{F}_{\rho}=\{f_{\mathrm{k}}:\mathrm{k}\in\rho\}$ has $f_{\mathrm{k}}=0$. Step 12 implements equation (6.16) and (6.17) respectively. In step 4 above, a failure in convergence will result in a reduction of dimension. This is a preliminary version of the final coding of this algorithm. The final coding allows the algorithm to restart from step 3, with a smaller prediction step size h until h is smaller than a user defined tolerance. A future study should involve the discussion of optimum step size control strategy for this algorithm.

It should be noted that even if the equation which causes the failure is not to removed by step 12 in the first instance; it will be removed later as long as the tracking problem persists. In the worst case, we have $\rho=\phi$ and we should have no tracking problem. This kind of over reduction of dimension will certainly reduce the efficiency of the method. However, by assumption (iii), the method will never diverge to infinity. Therefore, even if the efficiency is sometimes poor, the final solution will be found unless the method goes into a loop.

To improve the efficiency, it is necessary that the function which causes a tracking problem be identified and removed immediately.

6.5. SYNCHRONOUS MACHINE EXPERIMENTAL AND SIMULATED DATA EXAMPLE

In this section, the VDGN algorithm introduced previously is applied to the synchronous machine parameter estimation problem. In this case we also use the same initial conditions used previously. The algorithm was implemented in Matlab and shown in Appendix B. Table 6.1 illustrates the parameter estimates using experimental data of VDGN algorithm.

Table 6.1 VDGN synchronous machine parameter estimation (Experimental data)

Parameters	Nominal Values $\boldsymbol{\alpha}_{\mathrm{N}}^{\mathrm{i}}$	Parameters Estimates $\hat{\boldsymbol{\alpha}}_{\mathrm{E}}^{\mathrm{i}}$	Error (%) $\mathbf{E} = \left\lvert \frac{\boldsymbol{\alpha}_{\mathrm{N}}^{\mathrm{i}} - \hat{\boldsymbol{\alpha}}_{\mathrm{E}}^{\mathrm{i}}}{\boldsymbol{\alpha}_{\mathrm{N}}^{\mathrm{i}}} \right\rvert * 100\ \%$
x_d	1.801	1.800	0.055
$x_d^{'}$	0.285	0. 285	0.000
$x_d^{''}$	0.220	0.220	0.000
$T_{d0}^{'}$	3.700	3.700	0.000
$T_{d0}^{''}$	0.032	0.032	0.000
k	47.143	47.143	0.000
x_q	1.720	1.720	0.0004
$x_q^{''}$	0.220	0.220	0.008
$T_{q0}^{''}$	0.059	0.05899	0.003

In this example, the step size is set $h = 0.05$ and the starting point of the parameters, α_0, is the 60% of the nominal values. The initial set of indexes of functions to be solved is $\rho = \{\phi\}$ therefore, it is observed that after step 3 the search vector is $\xi = [0.05 \quad 0.05 \quad 0.05 \quad 0.05 \quad 0.05 \quad -0.05 \quad 0.05 \quad 0.05 \quad -0.05]^T$, point $f_1 = 0$ is the finish point of the step 7 ρ becomes $\rho = \{1\}$. The algorithm tries to track along f_1 in the same direction until the algorithm finds a zero crossover on f_2, ρ becomes $\rho = \{1 \quad 2\}$, now we have $\xi = [0 \quad 0 \quad 0.05 \quad 0.05 \quad 0.05 \quad -0.05 \quad 0.05 \quad 0.05 \quad -0.05]^T$. At this point, we can see that the curve $f_2 = 0$ is perpendicular to the search vector ξ. This invokes a failure condition explained in Section 5.2. The algorithm tries to track along $f_2 = 0$, since f_2 is the function being solved, we can set $\rho = \{1\}$ and $\beta = \{2\}$ in Step 12 above. The algorithm continues with the search vector $\xi = [0 \quad -0.05 \quad 0.05 \quad 0.05 \quad 0.05 \quad -0.05 \quad 0.05 \quad 0.05 \quad -0.05]^T$. The algorithm encounters a zero crossover in f_2 again. However, since β contains the index of f_2 this zero crossover will be ignored. The algorithm continues its search in the same direction until it finds a zero crossover in f_3. Then we set $\rho = \{1 \quad 3\}$ and $\beta = \{\phi\}$, Now we have $\xi = [0 \quad 0.05 \quad 0 \quad 0.05 \quad 0.05 \quad -0.05 \quad 0.05 \quad 0.05 \quad -0.05]^T$ and the algorithm tracks along $f_3 = 0$ until it reaches f_4, at this point, the algorithm encounters a false zero crossover and repeat the sequence same as f_2, finding f_4 again, the algorithm continues to track along f_4 until there is a zero crossover in f_5, Then we set $\rho = \{1 \quad 3 \quad 5\}$ and $\beta = \{\phi\}$, now we have $\xi = [0 \quad 0.05 \quad 0 \quad 0.05 \quad 0 \quad -0.05 \quad 0.05 \quad 0.05 \quad -0.05]^T$, and the algorithm track along f_5 until it reaches f_6 and the process is repeated for f_7 and f_8 until the algorithm finds the last solution at f_9.

By looking at Table 6.1 it can be seen that the values of the estimated parameters have smaller errors in comparison to the full order Gauss Newton estimation and the Subset Selection algorithm. This shows that the VDGN method also performs more efficiently than the Subset Selection algorithm shown in Chapter 4. Table 6.2 shows the results of applying VDNR to the problem of parameter estimation of the synchronous generator when the simulated data is used. The nominal values corresponds to the ones employed in [26], the initial conditions of estimation are the same as in previous case. Observe that the error percentages are considerably small and reliable parameter estimates are obtained.

Table 6.2 VDGN synchronous machine parameter estimation using data extracted from [26] (Simulated data)

Parameters	Nominal Values $\boldsymbol{\alpha}_N^i$	Full order Gauss Newton	VDGN Estimates $\hat{\boldsymbol{\alpha}}_E^i$	Error (%) $\mathbf{E} = \left\lvert \frac{\boldsymbol{\alpha}_N^i - \hat{\boldsymbol{\alpha}}_E^i}{\boldsymbol{\alpha}_N^i} \right\rvert * 100\ \%$
x_d	1.414	1.101	1.462	3.408
$x_d^{'}$	0.333	**0.483**	0.343	3.207
$x_d^{''}$	0.208	0.267	0.207	0.640
$T_{d0}^{'}$	5.85	**-5.092**	5.852	0.040
$T_{d0}^{''}$	0.194	0.108	0.198	2.153
k	1552	**-703.757**	1552.000	0.000
x_q	1.302	1.457	1.302	0.000
$x_q^{''}$	0.396	0.477	0.396	0.000
$T_{q0}^{''}$	0.955	1.0904	0.955	0.000

As a further analysis the VDGN results are compared to the values obtained by subset selection and full order Gauss Newton algorithm and shown in Table 6.3. Notice that the superiority of the VDGN method in comparison to the other methods (since it is possible to estimate efficiently all the parameters).

Table 6.3 Parameter estimates using VDGN , Subset Selection and full order Gauss Newton algorithms for experimental data

Parameters	Nominal Values	Parameter Estimates			
		VDGN	Subset Selection		Full Order
			I	II	
x_d	1.801	1.800	1.800	1.800	1.799
$x_d^{'}$	0.285	0. 285	-	-	0.285
$x_d^{''}$	0.220	0.220	0.220	0.220	0.212
$T_{do}^{'}$	3.700	3.700	3.700	-	3.696
$T_{do}^{''}$	0.032	0.032	0.032	0.032	0.032
k	47.143	47.143	-	47.143	47.098
x_q	1.720	1.720	1.717	1.691	1.720
$x_q^{''}$	0.220	0.220	0.219	0.216	0.220
$T_{qo}^{''}$	0.059	0.0589	0.059	0.059	0.059

Table 6.4 shows the error percentages obtained from the VDGN method, for purposes of comparison the Subset Selection and the full order Gauss Newton results are also presented.

Table 6.4 Error in estimated parameters for experimental data

Parameters	Error(%) of Parameter Estimates			
	VDGN	Subset Selection		Full Order
		I	II	
x_d	0.055	0.055	0.055	4.78
$x_d^{'}$	0.000	-	-	0.01
$x_d^{''}$	0.000	0.000	0.000	3.636
$T_{do}^{'}$	0.000	0.000	-	0.098
$T_{do}^{''}$	0.000	0.000	0.000	0.000
k	0.000	-	0.000	9.440
x_q	0.0004	0.169	1.700	0.000
$x_q^{''}$	0.008	0.169	1.700	0.000
$T_{qo}^{''}$	0.003	0.000	0.000	0.000

In Table 6.4, it can be seen that the VDGN shows smaller error percentages for the parameters of q-axis, x_q, $x_q^{''}$, $T_{qo}^{''}$. It is evident that VDGN performs better in comparison to the full order Gauss Newton method. Note that the error percentage obtained for x_d and k using the full order Gauss Newton is considerably larger than the value obtained from VDGN method. In general, we can say that the presence of relative large error percentages on the values obtained by using subset selection and VDGN methods might be due to the noise associated to the input signal (i.e., the field voltage and field current) of the system.

Furthermore, notice the advantage of VDGN methodology in estimating all parameters of the model. This capacity of VDGN provides complete information in terms of parameter estimates of the synchronous generator and represents a useful feature that will efficiently help to diagnose and detect possible failures in the operation of the generator.

In addition, Table 6.5 presents a comparison of the parameter estimates obtained in [26] (full-order Gauss Newton, subset selection and Tikhonov regularization) and the ones obtained by applying the VDGN method.

For this purpose the noisy simulated data and the nominal parameter presented in [26] was used for the application of the VDGN method.

Table 6.5 Comparison of parameter estimates using noisy simulated data

Parameter	Nominal values	Gauss- Full Order	Subset k and $T_{d0}^{'}$	Tikhonov	VDNR
x_d	1.414	1.101	1.414	1.259	1.462
$x_d^{'}$	**0.333**	**0.483**	**0.334**	**0.315**	**0.343**
$x_d^{''}$	0.208	0.267	0.196	0.201	0.207
$T_{d0}^{'}$	**5.85**	**-5.092**	**-**	**4.385**	**5.852**
$T_{d0}^{''}$	0.194	0.108	0.812	0.177	0.198
k	**1552**	**-703.757**	**-**	**1163**	**1552.000**
x_q	1.302	1.457	1.302	1.299	1.302
$x_q^{''}$	0.396	0.477	0.393	0.395	0.396
$T_{q0}^{''}$	0.955	1.0904	0.955	0.958	0.955

From Table 6.5 it can be noticed that VDGN method is acceptably competitive with the other methods of estimation. Observe that by using the VDGN algorithm the parameter estimates are more reliable than the ones obtained using the Tikhonov regularization. By comparing the VDGN and subset selection results it can be seen that the subset selection strategy performs better than VDGN for the x_d and $x_d^{'}$; however for the remaining parameters VDGN is superior. This can be seen more clearly in Table 6.6, in which we present the error percentages of the parameter estimates for the different methodologies. The subset selection results corresponds to the case when k and $T_{d0}^{'}$ are fixed.

Table 6.6 Comparison of error percentages using noisy simulated data

Parameter	Gauss Full Order	Subset Selection	Tikhonov	VDNR
x_d	22.07	0.0384	10.89	3.408
$x_d^{'}$	**45.109**	**0.118**	**3.55**	**3.207**
$x_d^{''}$	28.393	5.664	3.42	0.640
$T_{d0}^{'}$	**187.048**	**-**	**25.04**	**0.040**
$T_{d0}^{''}$	**44.148**	**6.594**	**11.16**	**2.153**
k	**145.345**	**-**	**25**	**0.000**
x_q	11.94	0.007	0.309	0.000
$x_q^{''}$	**20.474**	**0.04**	**0.241**	**0.000**
$T_{q0}^{''}$	1.0904	0.955	0.958	0.000

6.6. SUMMARY

In this Chapter we studied the Variable Dimension Gauss-Newton (VDGN) method. The VNGN algorithm as show above is based on the variable dimension Newton Raphson (VDNR) method. The VDGN algorithm was applied to the synchronous machine experimental example so as to obtain the parameter estimates. The VDGN results were compared to the values obtained by subset selection and full order Gauss Newton algorithm and shown in Table 6.3. Notice that the superiority of the VDGN method in comparison to the other methods (since it is possible to estimate efficiently all the parameters). Table 6.4 shows the error percentages obtained from the VDGN method, for purposes of comparison the Subset Selection and the full order Gauss Newton results are also presented. In Table 6.4, the VDGN shows smaller error percentages for the parameters of q-axis, x_q, $x_q^{"}$, $T_{qo}^{"}$. It was evident that VDGN performs better in comparison to the full order Gauss Newton method. Note that the error percentage obtained for x_d and k using the full order Gauss Newton is considerably larger than the value obtained from VDGN method. Notice the advantage of VDNR methodology of estimating all parameters of the model.

Table 6.5 presents a comparison of the parameter estimates obtained in [26] (full-order Gauss Newton, subset selection and Tikhonov regularization) and the ones obtained by applying the VDGN method. For this purpose the noisy simulated data and the nominal parameter presented in [26] were used for the application of the VDGN method. From Table 6.5 it can be noticed that VDGN method is acceptably competitive with the other methods of estimation. Observe that by using the VDGN algorithm the parameter estimates are more reliable than the ones obtained using the Tikhonov regularization. By comparing the VDGN and subset selection results it can be seen that the subset selection strategy performs better than VDGN for the x_d and $x_d^{'}$; however for the remaining parameters VDGN is superior. This can be seen more clearly in Table 6.6, in which we present the error percentages of the parameter estimates for the different methodologies. The subset selection results corresponds to the case when k and $T_{d0}^{'}$ are fixed.

.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

In this final chapter, the main results and the contributions of this thesis will be summarized. Then we outlined some potential future studies.

7.1. SUMMARY AND CONTRIBUTIONS

In this thesis we have addressed the use of robust parameter estimation algorithms that can be used in fault detection and diagnosis. Our work has focused mainly on studying the variable dimension Gauss-Newton algorithm to solve ill conditioned parameter estimation problem.

In Chapter 1, the justification and objectives of this research were introduced; Chapter 2 presents some fundamental concepts and approaches of failure detection and diagnosis in synchronous generators.

Chapter 3 is dedicated to the description of the mathematical model of the synchronous generator. The importance of developing more complex machine models is evident because they permit to generate diagnosis signals sensitive to the occurrence of faults as well as more accurate control and prediction of the machine behavior. In this regard, a synchronous machine model that introduces its steady-state, and transient was developed. This led to the *dq* linearized small signal model of the synchronous generator that contains parameters corresponding to the steady state, transient and subtransient conditions.

Chapter 4 presents the synchronous generator parameter estimation problem. In this chapter, we also introduced the conditioning analysis of the parameter estimation problem. The ideas of output error and least squares methodology were used to solve the problem of nonlinear parameter estimation. In this chapter, the Gauss Newton method was introduced and used to compute the parameter estimates. A numerical experiment of the synchronous machine was presented, the input data corresponds to the FC5HP synchronous generator of the Arizona Public Service Company (APS) and the simulation model of [26]. The measurements were taken under steady state conditions when the generator was serving a load. The Gauss Newton method was applied and the estimates shown in Table 4.2. Making a relative comparison of the estimated values, we could see that some parameters have error percentages relatively large in comparison with others. In particular, x_d and k have the highest error followed by $T_{d0}^{'}$. Observe that the error percentage corresponding to the q axis components are small in comparison with the others. In this chapter, we also developed the study of parameter conditioning using the componentwise condition numbers. The numerical results of the synchronous machine example revealed relatively large condition numbers. After a relative comparison we were able to identify that parameters k and $T_{d0}^{'}$ were the ill-conditioned ones and the remaining parameter were considered well-conditioned. If we observe the results of the full order Gauss-Newton parameter estimation for the synchronous machine showed in Table 4.2, we can see that the parameters that have the largest error percentage are the parameters x_d, k and $T_{do}^{'}$, which are among the worst conditioned parameters in Table 4.3.

Chapter 5 is mainly focused on Subset Selection analysis. This strategy consists of adding prior information to the parameter estimation problem, selecting a subset of parameters and fixing them to prior values and still performs meaningful estimation of the remaining parameters. Therefore the number of parameters to fix was determined by eigenanalysis of the Hessian matrix. Which parameters to fix were determined by the subset selection method for determining column independence in matrices. In this chapter, we also studied and implemented the combinatorial analysis to determine the best combination or combinations of parameters to be fixed. Once the parameters to fix were determined, the constrained optimization problem described in equation (5.3) was solved.

The application of this methodology to the synchronous machine parameter estimation problem was then presented. The application of the combinatorial analysis produced two combinations with the smallest condition numbers corresponding to fixing the combinations, $x_d^{'}$ and $T_{d0}^{'}$; and $x_d^{'}$ and k respectively. The condition number of the Hessian matrix is reduced significantly in comparison with others combinations. Then the conditioning of the estimates for the two best combinations was evaluated. It was observed that the parameter conditioning was improved condition numbers (reduced) by fixing the combinations $x_d^{'}$, $T_{d0}^{'}$ and $x_d^{'}$, k respectively. Then the parameter estimates for the reduced order case were evaluated using the Gauss-Newton algorithm. In general, we concluded that the estimates and the performance of the Subset Selection algorithm were considerably improved in comparison to the ones obtained for the full order Gauss Newton algorithm (Table 4.2). Furthermore fixing the parameters combination $x_d^{'}$, $T_{d0}^{'}$, k and $x_d^{'}$ resulted in high sensitivity reduction as shown in Table 5.4. However the approach requires good prior information of the parameters being fixed.

In Chapter 6 an alternative method based on the Variable Dimension Gauss-Newton (VDGN) algorithm was presented. The VNGN algorithm is based on the variable dimension Newton Raphson (VDNR) method. The VDGN algorithm was applied firstly to the synchronous machine experimental example so as to obtain the parameter estimates. It was observed that the values of the estimated parameters in Table 6.4 presented smaller errors in comparison to the full order Gauss Newton estimation and the Subset Selection algorithm. This showed that the VDGN method performed better than the Subset Selection case shown in Chapter 5. Furthermore, the VDNR does not need accurate prior information of the parameters. In Table 6.4, the VDGN shows smaller error percentages for the parameters of q-axis, x_q, $x_q^{''}$, $T_{qo}^{''}$. It was evident that VDGN performs better in comparison to the full order Gauss Newton method. Note that the error percentage obtained for x_d and k using the full order Gauss Newton is considerably larger than the value obtained from VDGN method. In general, we can say that the presence of relative large error percentages on the values obtained by using subset selection and VDGN methods might be due to the noise associated to the input signal (i.e. noise and harmonics in the field voltage and field current) of the system. Furthermore, notice the advantage of VDNR methodology of estimating all parameters of the model. This capacity of VDNR provides complete information in terms of parameter estimates of the synchronous generator and represents a useful feature that will efficiently help to diagnose and detect possible failures in the operation of the generator.

In addition, Table 6.5 presents a comparison of the parameter estimates obtained in [26] (full-order Gauss Newton, subset selection and Tikhonov regularization) and the ones obtained by applying the VDGN method. For this purpose the noisy simulated data and the nominal parameter presented in [26] were used for the application of the VDGN method. From Table 6.5 it can be noticed that VDGN method is acceptably competitive with the other methods of estimation. Observe that by using the VDGN algorithm the parameter estimates are more reliable than the ones obtained using the Tikhonov regularization. By comparing the VDGN and subset selection results it can be seen that the subset selection strategy performs better than VDGN for the x_d and $x_d^{'}$; however for the remaining parameters VDGN is superior. This can be seen more clearly in Table 6.6, in which we present the error percentages of the parameter estimates for the different methodologies. The subset selection results corresponds to the case when k and $T_{d0}^{'}$ are fixed.

From Table 6.5 and Table 6.6 we can conclude that the VDNR algorithm developed in this research performs as reliably as the other methods proposed in [26]. On the other hand, it has been observed that the simulation cost is higher in comparison to the other methods developed in this work. Since our problem is considerably small in terms of dimension (i.e. the model has nine parameters to be estimated) the VDNR is not costly as it would be for larger dimension problems.

In summary we can conclude:

For Experimental Data

- Subset selection is excellent but requires good prior estimate of the parameter to be fixed
- VDGN method perform better in comparison to the full order GN and Subset Selection algorithm and furthermore we need no accurate prior information of a parameter subset.

For Simulated Data

- VDGN performs better than Tikhonov and full-order GN and is relatively superior to the subset selection strategy.

In addition

- VDGN algorithm developed in this research performs as reliably as the other methods proposed in [26].
- The excellent VDGN capacity of providing reliable and complete synchronous generator parameters estimates, represent a useful feature that will efficiently help to detect and diagnose possible failures in the operation of the generator.

7.2. FUTURE WORK

Based on the overall results we want to discuss some potential future studies that follow the strategies analyzed in this research. As discussed in Chapter 1 the objective of having accurate parameter estimates is to use them in applications of fault detection and diagnosis of synchronous generators. In that sense, one possible future study could be focused in the next alternatives as future work

- Implementation of recursive VGND methods for on-line parameter estimation and use them in synchronous machine fault detection and diagnosis.
- Use of the approach in a diagnosis loop.
- Although VDGN has demonstrated great effectiveness for both the experimental and simulated data, it is necessary to prove mathematically the convergence of the algorithm.

REFERENCES

[1] J. Gertler, *Fault Detection and Diagnosis in Engineering Systems,* Marcel Dekker, Inc., New York, 1998.

[2] X.C. Lou, A.S.Willsky and G.L. Verguese "Optimal Robust Redundancy Relations for Failure Detection in Uncertain Systems". Automatica , Volume 22 pp. 333-344, 1986.

[3] A.S. Wilsky, "A survey of design methods for failure detection systems." Automatica. Vol.12, pp.601-611, 1976

[4] R.J. Patton and S.M. Kangethe, "Robust fault Diagnosis in Dynamic Systems". Proc. UNICOM Seminars on Failsafe control Systems, Kogan page, June 1988.

[5] R. Iserman, "Proces fault detection based on modeling and estimation methods", Automatica, Vol,20, pp.387-404 , 1984

[6] R. Iserman, "Fault diagnosis of machines via parameter estimation and knowledge processing' Automatica,Vol.29, pp 815-835, 1993.

[7] C. Bunks, D. McCarthy "Condition-Based Maintenance of Machines using Hidden Markov Models", BBN Corporation, Cambridge, MA. U.S.A., March 2000.

[8] C. Byinton, M. Roemer, T. Galie "Prognostic Enhancements to Diagnostic Systems for Improved Condition-Based Maintenance" IEEE, MT, March 2002

[9] E. Albas, T. Arikan, C. Kuzkaya " In-process motor testing results using model based fault detection approach" .Artesis, Turkey, 1994

[10] G. J. Kacprzynski, M. J. Roemer "Health Management Strategies for 21st Century Condition-Based Maintenance Systems" International COMANDEM Congress, Houston, TX, Dec. 2000

[11] J. Penman and H. Jiang, "The detection of stator and rotor winding short circuits in synchronous generators by analyzing excitation current harmonics", International Conference on Opportunities and Advances in International Electric Power Generation, 18-20 Mar, pp. 137-142, 1996

[12] H.A.Darwish, A.M.I. Taalab and T.A. Kawady, "Development and implementation of an ANN-based fault diagnosis scheme for generator winding protection", IEEE Transactions on Power Delivery, Volume 16, Issue 2, pp. 208-214, Apr 2001

[13] W.T. Thomson, M. Fenger "Current signature analysis to detect induction motor faults" Industry Applications Magazine, IEEE, Volume 7, Issue 4, pp. 26-34, July-Aug. 2001.

[14] R. Shoureshi, T.Norick, R.Swartzendruber "Intelligent Transformer Monitoring System Utilizing Neuro-Fuzzy Technique Approach", PSERC Publications 04-26, July 2004.

[15] C. De-Persis, A. Isidori, "A geometric approach to nonlinear fault detection and isolation" Automatic Control, IEEE Transactions, Volume 46, Issue 4, pp 853-865, Jun 2001.

[16] C. De-Persis, R. De-Santis, and A. Isidori, "Nonlinear actuator fault detection and isolation for a VTOL aircraft", American Control Conference 2001,IEEE Proceedings, Volume 6, pp. 4449-4454, Jun 2001.

[17] J. Cadick, P.E. "Condition Based Maintenance. How to get started" Cadick Corporation, 1999.

[18] J.D. McCalley, T. V. Voorhis and A.P. Meliopoulos, "Automated Integration of Condition Monitoring with an Optimized Maintenance Scheduler for Circuit Breakers and Power Transformers", A proposal to NSF Power Systems Engineering Research Center (PSERC), 2001.

[19] J. Sottile, F.C. Trutt and A. Leedy "Condition monitoring of brushless three-phase synchronous generators with stator winding or rotor circuit deterioration" Industry Application Conference, IEEE 2001 pp. 1587-1594, Oct 2001

[20] J. Endrenyi, S. Aboresheid, R.N. Allan, G.J. Anders, S. Asgarpoor, R. Billinton, N. Chowdhury, E.N. Dialynas, M. Fipper, R.H. Fletcher, C. Grigg, J.McCalley, S.Meliopoulos, T.C. Mielnik, P. Nitu, N. Rau, N.D. Reppen, L. Salvaderi, Schn "The present status of maintenance strategies and the impact of maintenance on reliability", IEEE Transactions on Power Systems, Volume 16, Issue 4, pp. 638-646, Nov 2001.

[21] J.T. Tou, and R.C. Gonzales *Pattern recognition principles,* Addison-Wesley Publishing. Mass 1974.

[22] M. Hoof, T. Laird, "A strategy for advanced condition based maintenance of large generators" Electrical Insulation Conference and Electrical Manufacturing & Coil Winding Technology Conference, 2003. Proceedings, 23-25, pp. 463-468, Sept. 2003

[23] J. Guo, Z. Li, Y. Chen, Y. Wang, and S. Cheng, "Virtual environment conception for CBM of hydro-electric generating units", Power System Technology, 2002. Proceedings. Power-Con 2002. International Conference on , Volume 3, 13-17, pp. 1957-1961,Oct.2002.

[24] X.Z. Yu, S. Birlasekaran, S.S. Choi, Y.C. Yong, T.F. Lim, "Condition monitoring of a power station" Power System Technology, 2000. Proceedings, Power-Con 2000. International Conference on, Volume 2, 4-7, pp.1029-1033, Dec.2000.

[25] L. Churchill and J. Edmonds "Rotor-mounted scanning of stator in hydro and turbine-driven generators" Electrical Electronics Isolations Conference , IEEE Proceedings, pp.38-46, 1991

[26] L. Jáuregui, *Ill-conditioning In parameter Estimation with Application to Synchronous Generator,* M. S. Thesis, University of Puerto Rico at Mayagüez,. Mayagüez, Puerto Rico, 2001

[27] J.W. Feltes, S. Orero, B. Fardanesh, E. Uzunovic, S.Zelingher "Deriving model parameters from test measurements" Power Technology, Inc, New York, USA, 2002.

[28] P. Kundur, *Power System Stability and Control.* , Power System Engineering Series, New York, Mc Graw Hill, 1994

[29] P.W. Sohuer and M.A. Pai, *Power Systems Dynamics and Stability*, Prentice-Hall, 1998.

[30] Z.M. Zhao, F.S. Zheng, J.D. Gao, And L.Y. Xu, "A Dynamic On-line Parameter Identification and Full-scale System Experimental Verification for Large Synchronous Machines", *IEEE Trans. Energy Conversion*, Vol. EC-10, No. 3, pp.392-398, September 1995**.**

[31] E.Kyriakides, *Innovative Concepts for on-line Synchronous Generator Parameter Estimation,* PhD. Thesis, Arizona State University, Phoenix, Arizona, 2003.

[32] G.A.F. Seber, C.J. Wild, *Nonlinear Regression*, Wiley Interscience,1989.

[33] H.K. Khalil, *Nonlinear Systems*, 2nd ed., Prentice-Hall, 1996.

[34] Saltelli, E. M. Scott And K. Chan, *Mathematical and Statistical Methods for Sensitivity Analysis*, John Wiley, 2000.

[35] M. Vélez-Reyes. *Decomposed Algorithm for Parameter Estimation*, Ph. D. Thesis pp. 97-122, MIT, Massachusetts, Set. 1992.

[36] P.W. Sohuer, *Power system Dynamics and Control*. First Edition pp. 23 - 42,1998.

[37] H. Khall *Nonlinear System,* Second edition" pp. 81- 84, 1996.

[38] L. Ljung, *System Identification,* Second Edition pp. 204-205,520-521,1999.

[39] É. Walter,and L. Pronzato. *Identification of Parametric Models*, pp. 163-299, First edition 1997.

[40] P.C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems*, p.p. 53 – 56,First Edition 1996.

[41] M. Burth, G.C. Verghese, M. Vélez-Reyes, "Subset Selection for Improved Parameter Estimation in on-line Identification of a Synchronous Machine", *IEEE Trans. on Power System*. Volume 141, pp. 218-225, March 1999.

[42] L. Jáuregui and M. Vélez-Reyes, "Conditioning Analysis of Parameter Estimation in a Synchronous Generator". In Proceedings of the 34th Annual North American Power Symposium, October 14-15, 2002.

[43] M. Vélez-Reyes, G.C. Verghese, "Subset Selection in Identification and Application to Speed and Parameter Estimation for Induction Machines", 4th IEEE Conference on Control Applications. Albany, N.Y. pp. 991-997 September 28-29, 1995.

[44] Seber, G. A. F. and C. J. Wild, *Nonlinear regression.* John Wiley & Sons, New York, pp 768-780. 1989.

[45] W. Jing-Cheng, C. Hsiao-Dong, H. Chiang-Tsung, C. Yung-Tien, C. Chung-Lian And C. Chiew-Yann, "On-line Measurement-Based Model Parameter Estimation for Synchronous Generators: Solution Algorithm and Numerical Studies", IEEE Transactions on Energy Conversion, Vol. 9, pp. 337-343, June 1994.

[46] J. Sprave and S. Rolf “Variable Dimensional Optimization With evolutionary Algorithms Using Fixed-Length Representations” 3rd IEEE/IAFE CIFERr Conference on Computational Intelligence for Financial Engineering New York March 1997

[47] L. Vandenbergue and J. Vandewalle “Variable dimension algorithms for solving resistive circuits” Int. J. Circuit Theory Appl. Vol. 18, pp.443-474,1990

[48] A. Ben-Israel, “Generalized inverse and applications”, edited by M. Zuhair Nashed, pp. 245-302, Academic Press. 1976.

[49] L.V. Kolev, “Finding all solutions of nonlinear resistive circuits equations via interval analysis” Int. J. of Circuit Theory and Applications, Vol. 18, pp. 257-267, 1990.

[50] S. W. Ng and Y.S. Lee “A Variable Dimension Newton-Raphson Method” IEEE Int. Symp. on Circuits and Systems, London, Vol. 6, pp. 129-132, May 1994.

[51] D.G. Luenberguer, *Introduction to Linear and Nonlinear Programing.* Reading Mass, Addison-Wesley 1973.

[52] J.Stoer and R. Bulirch, *Introduction to Numerical Analisis.* New York, Springer-Verlag 1980.

[53] R.A. Usmani, *Applied linear Algebra*: New York Marcel Decker, 1988

[54] *IEEE Standard Common Format for Transient Data Exchange (COMTRADE) for Power Systems,* IEEE Standard C37-111.1999, Mar.1999

[55] L.T. Watson. *Globally convergent homotopy methods: a tutorial* Applied Mathematics and Computation. pp. 369-396, 1989.

APPENDIX A

DATA FORMATS SUPPORTED BY THE ALGORITHMS

There are two data formats that are supported by the synchronous parameter estimation algorithms: an *abc* data format that can be read as a text file (.txt extension), and a combination of two data files that follow the COMTRADE format of the IEEE Srt. C37-111.1999 (.cfg and .dat extension) [54]. The purpose of having two data formats is to accommodate all possible configurations of data files currently used by utilities. The COMTRADE format data file is the output of the of digital fault recorders (DFRs) that are typically used for recording measurements at the terminal of ha synchronous generator, while .txt files can be generated using variety of word processors. And example of each data format is shown in section A.1 and A.2 respectively

A .1 TEX FILE (ABC) DATA FORMAT

Input data in the form of a text file (.txt extension) is a convenient data format for data that have been preprocessed after the required measurements have been obtained from the data recording device. The data are arranged in columns in the order shows in Table A.1. There are nine columns that are required. The first column contains the time measurements, while the remaining eight columns contain the stator and field voltages (V_{ab}, V_{bc}, V_{ca} and V_F) and the stator and field currents. (I_a, I_b, I_c and I_F).

Table A.1 arrangement of data of 5in the text data file

Time (s)	V_{ab} (kV)	V_{bc} (kV)	V_{ca} (kV)	V_F (V)	I_a (kA)	I_b (kA)	I_c (kA)	I_F (kA)
0	-29.4	20.9	8.5	274.0	-2.37	1.24	1.05	1.19
4.7×10^{-4}	-30.3	16.6	13.6	114.0	-2.46	0.98	1.40	1.16
9.40×10^{-4}	-30.2	12.0	18.2	-60.7	-2.37	.53	1.75	1.12
1.41×10^{-3}	-29.1	6.8	22.2	-231.0	-2.19	0.09	2.02	1.08
1.88×10^{-3}	-27.0	1.3	25.4	311.0	-1.93	-0.44	2.28	1.19
⋮	⋮		⋮	⋮	⋮	⋮	⋮	⋮

A 2 COMTRADE DATA FORMAT

The second option that is supported by the synchronous parameter estimation algorithms is the format that is based on COMTRADE data format which is an IEEE standard. A DFR outputs two data files, the first being a configuration data file containing general information for the signals, while the second file is .dat file and contains the measurements at the generator terminal. Sample configurations file of FC5HP synchronous generator of Arizona Public Service (APS) can be seen in Figure A.1. A complete explanation of each item in the configuration file is offered in [54]. In general, the first two lines in the file contain the heading and the number of channels that were used to record data. The next section of the file that is arranged in numerical order contains information for each measurement. This information contains the channel, the unit of measurement (e.g. V, kV or kA), the multiplication factor to be used for each signal and the offset (if any) because of the DFR settings.

```
Ben664_FC5&5,664,1999
8,8A,0D
1,U5 HP Ia      ,A,0,kA  ,0.0026987161,0.0000000000,0,-32767,+32767,1,1,P
2,U5 HP Ib      ,B,0,kA  ,0.0026987161,0.0000000000,0,-32767,+32767,1,1,P
3,U5 HP Ic      ,C,0,kA  ,0.0026987161,0.0000000000,0,-32767,+32767,1,1,P
4,U5 HP Vfd      ,,,V  ,0.0518153496,0.0000000000,0,-32767,+32767,1,1,P
5,U5 Vab Gen     ,A,0,kV  ,0.0018998961,0.0000000000,0,-32767,+32767,1,1,P
6,U5 Vbc Gen     ,B,0,kV  ,0.0018998961,0.0000000000,0,-32767,+32767,1,1,P
7,U5 Vca Gen     ,C,0,kV  ,0.0018998961,0.0000000000,0,-32767,+32767,1,1,P
8,U5 HP Ifd     ,,,A  ,0.2072613984,0.0000000000,0,-32767,+32767,1,1,P
60
1
10000.000,3219
25/07/2002,13:00:01.238600
25/07/2002,13:00:01.358600
ASCII
```

Figure A.1 FC5HP synchronous generator (.cfg) data file following the COMTRADE format

Finally, the last section of the configuration file contains the frequency of the generator, the sample rate and total number of samples, the start and end times and dates of the measurement, and the data file type (e.g. ASCII or binary). The configuration file is associated with a data file with a .dat extension. These are produced simultaneously by the recording device. The .dat file contains integer measurements of the signals. The column numbers in the .dat file correspond to the channel number in the .cfg file starting from column 3. Column 1 contains the sample number, while column 2 contains the timestamp. From the timestamp the time of each measurements can be obtained by,

$$t = (timestamp) \times (timemult)\mu s \tag{A.1}$$

where *timemult* is the timestamp multiplication factor obtained from the last line of the .cfg file. A sample .dat file can be seen in Figure. A.2

1,	0,	-5104,	-192,	5184,	11328,	-8192,	-8128,	16256,	12240
2,	100,	-4976,	-432,	5296,	10432,	-7664,	-8640,	16240,	12304
3,	200,	-4864,	-640,	5392,	9840,	-7104,	-9168,	16208,	12352
4,	300,	-4736,	-864,	5472,	9136,	-6528,	-9696,	16144,	12368
5,	400,	-4592,	-1088,	5552,	8336,	-5952,	-10176,	16048,	12384
6,	500,	-4448,	-1312,	5632,	7680,	-5376,	-10656,	15952,	12400
7,	600,	-4304,	-1504,	5696,	6912,	-4784,	-11104,	15824,	12400
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
3214,	321300,	3920,	-5744,	1696,	-3360,	15360,	-12304,	-3120,	12016
3215,	321400,	4096,	-5680,	1472,	-3888,	15536,	-11888,	-3712,	11984
3216,	321500,	4240,	-5616,	1248,	112,	15680,	-11440,	-4304,	11888
3217,	321600,	4400,	-5552,	1024,	7120,	15776,	-10960,	-4896,	11808
3218,	321700,	4560,	-5472,	800,	11440,	15920,	-10512,	-5488,	11856
3219,	321800,	4688,	-5376,	576,	9856,	16048,	-10048,	-6080,	11936

Figure A.2 FC5HP data (.dat) file following the COMTRADE format

APPENDIX B

MATLAB CODES AND FILES

This appendix shows the MATLAB codes and files for computing the parameter conditioning and parameter estimation of the FC5HP synchronous generator. Table B.1 correspond to the MATLAB® files for calculation of the Jacobian, Hessian and the respective generator waveforms. Table B.2 contains the MATLAB files for the VDNR method

B.1 FILES FOR SYNCHRONOUS MACHINE MODEL

Table B. 1 Main files and functions for the Synchronous Machine Model and Parameter Estimation

File	Description
PSERC-Data	This file calculates the voltages and currents of FC5HP synchronous generator
Elmode	This file calculates the electrical subsystem matrices Ae, Be, Ce and De using the parameter set of FC5HP synchronous generator
Gen_sim	This file calculates the system matrices A,B,C and D using the parameter set of FC5HP synchronous generator
Grade0	Calculates the Jacobian for electrical model, For no parameters fixed
Grade_1	Calculates the Jacobian for electrical model, 2 fixed parameters
Grade_2	Calculates the Jacobian for electrical model, 2 fixed parameters
Error0	This file computes the error function between the measurements and the prediction model of the electrical system with no parameters fixed
Error_1	Computes the error function between the measurements and the prediction model of the electrical system with two parameters fixed
Error_2	Computes the error function between the measurements and the prediction model of the electrical system with two parameters fixed
Error_3	Computes the error function between the measurements and the prediction model of the electrical system with two parameters fixed
Gauss_e	This files compute the parameter estimates by implementing Gauss Newton algorithm

PSERC-Data

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PSERC SYNCHRONOUS GENERATOR DATA
% VOLTAGES AND CURRENTS
% Loading FC5HP data from PSERC-ASU
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%JHQ%%%%%

clear all
clc
load('FC5HP_1.txt');
t=FC5HP_1(:,2);
%line to line voltages
Vab=0.0018998961*FC5HP_1(:,7);
Vbc=0.0018998961*FC5HP_1(:,8);
Vca=0.0018998961*FC5HP_1(:,9);
%line to line currents
Ia=0.0026987161*FC5HP_1(:,3);
Ib=0.0026987161*FC5HP_1(:,4);
Ic=0.0026987161*FC5HP_1(:,5);

Vfd=0.0518153496*-1*FC5HP_1(:,6);
Ifd=0.2072613984*FC5HP_1(:,10);

%subplot(311)
figure(1)
plot(t,Vab,'g-', t, Vbc,'r--', t,Vca,'b-.');
title('Line to line voltages ')
legend('Vab','Vbc','Vca')
xlabel('Time [us]');ylabel('kV');grid
axis([0 1*10^5 -40 40])

%subplot(312)
figure(2)
plot(t,Ia,'g-', t, Ib,'r--', t,Ic,'b-.');
title('Line to line currents ')
legend('Ia','Ib','Ic')
xlabel('Time [us]');ylabel('kA');grid
axis([0 1*10^5 -20 20])

%subplot(313)
figure(3)
plot(t,Vfd)
title('Field voltage');grid
legend('Vfd(V)')
xlabel('Time [us]')
axis([0 1*10^5 -1000 400])

figure(4)
plot(t,Ifd,'g');
title('Field current');grid
legend('Ifd(A)')
xlabel('Time [us]')
axis([0 1*10^5 2420 2620])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
```

```
MVAbase=483000000;
Wsyn=2*pi*60;
Vbase=17962.92;
Ibase=2416.05353;
VLLrms=22000;
```

Elmode

```
% Script-file for the simulation of the
% electrical subsystem of a synchronous generator
%
% This file calculates the electrical subsystem matrices
% Ae,Be,Ce and De using the parameter set of FC5HP
% synchronous generator "PSERC-ASU"
%
    % Input:  none

% Output: generator's electrical subsystem matrices Ae,Be,Ce,De
%
% After having calculated the system matrices, the
% output and state vectors
%
%ye = [id iq]'
%xe = [Eq' Eq'' Ed'']'
%
% of the electrical subsystem can be calculated invoking
%
%    [ye,xe] = lsim(Ae,Be,Ce,De,u,T);
%
% with
%
%u = [vd vq vfd]'
%
% the subsystem's input vector and T a time vector.
%
% Prior to calculating [ye,xe] the simulation file 'gen_sim' must
% be used to get the voltages vd and vq!

%*********** the calculations start here ************************

%%%%% Fixed Parameters (from 'china paper')%%%%%

xd=1.801;   % 1. parameter to be estimated
xd_=0.285;  % 2. parameter to be estimated
xd__=0.220; % 3. parameter to be estimated
Td0_=3.7;   % 4. parameter to be estimated
Td0__=0.032;   % 5. parameter to be estimated
k=47.143;      % 6. parameter to be estimated
xq=1.72; % 7. parameter to be estimated
xq__=0.220; % 8. parameter to be estimated
Tq0__=0.059;   % 9. parameter to be estimated
```

```
H=1.3114;      % parameter known due to prior estimations
Df=1.89; % parameter known due to prior estimations
X=0.016; % parameter known due to prior estimations
Vb=0.99; % parameter known due to prior estimations




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                                            %%
%%   Calculation of the system matrices       %%
%%                                            %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

d12=-1/xd__;
d21=1/xq__;

Ae=[a11 a12 0;a21 a22 0;0 0 a33];
Be=[0 b12 b13;0 b22 b23;b31 0 0];
Ce=[0 c12 0;0 0 c23];
De=[0 d12 0;d21 0 0];
```

Gen_sim

```
% Script-file for generator simulation
%
% This file calculates the system matrices A,B,C and D
% using the parameter set of FC5HP
% synchronous generator "PSERC-ASU"

% Input:  none
% Output: generator's system matrices A,B,C,D
%
% After having calculated the system matrices, the
% generator's output and state vectors
%
%    y = [id iq delta_t vd vq]'
%    x = [Eq' Eq'' Ed'' w phi]'
%
```

```
% can be calculated invoking
%
%    [y,x] = lsim(A,B,C,D,vfd,T);
%
% with vfd the incremental field voltage as the only system
% input and T a time vector.



%*********** the calculations start here ***********************

%%%%% specify operating point %%%%%%%%%%%%%%

P=1;     % active power p.u.
Q=0.1;      % reactive power p.u.
alpha_bus=70;  % angle between d-axis and Vbus,
      % see phasor relationships in Fig. 1

%%%%% Design Parameters (fixed) %%%%%%%%%%%%

xd=1.80; % 1. parameter to be estimated(pu)
xd_=0.285;  % 2. parameter to be estimated(pu)
xd__=0.220; % 3. parameter to be estimated(pu)
Td0_=3.7;   % 4. parameter to be estimated(sec)
Td0__=0.032;   % 5. parameter to be estimated(sec)
k=47.143;      % 6. parameter to be estimated
xq=1.72; % 7. parameter to be estimated(pu)
xq__=0.220; % 8. parameter to be estimated(pu)
Tq0__=0.059;   % 9. parameter to be estimated(sec)
H=1.314;    % 10.(Mw.sec/MVA)inertia constant
Df=1.89; % 11.(mechanical damping coefficient)
            %or Damping Torque.
X=0.016; % 12.(Xe)
Vb=0.99; % 13.(Vbus)infinite bus
w0=2*pi*60;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                                        %%
%%           Operating Point              %%
%%                                        %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

alpha=alpha_bus/180*pi; % alpha in rad
Vd=Vb*cos(alpha); % Vbus on the d-axis
Vq=Vb*sin(alpha); % Vbus on the q-axis

%%%%% Solution for id0 %%%%%%%%%%%%%%%%%%%%%%%

a=X*(Vq^2+Vd^2);
b=Vq^3+Vq*(Vd^2)-2*X*P*Vd;
c=X*(P^2)-Vq*Vd*P-Q*(Vq^2);

p=b/a/2;
q=c/a;

x1=-p+sqrt(p^2-q);
```

```
x2=-p-sqrt(p^2-q);

if x1>=0
   id0=x1;
else
   id0=x2;
end


%%%%% Solution for iq0 %%%%%%%%%%%%%%%%%%%%%%%%

iq0=(P-id0*Vd)/Vq;

%%%%% Solution for vd0 %%%%%%%%%%%%%%%%%%%%%%%%

vd0=Vd-X*iq0;

%%%%% Solution for vq0 %%%%%%%%%%%%%%%%%%%%%%%%

vq0=Vq+X*id0;

%%%%% Solution for delta_t0 %%%%%%%%%%%%%%%%%%%

delta_t0=atan(vd0/vq0);

%%%%% Solution for I0 %%%%%%%%%%%%%%%%%%%%%%%%%

I0=sqrt(id0^2+iq0^2);

%%%%% Solution for Vt0 %%%%%%%%%%%%%%%%%%%%%%%%

Vt0=sqrt(vd0^2+vq0^2);

%%%%% Solution for phi0 %%%%%%%%%%%%%%%%%%%%%

phi0=atan(vq0/vd0)-atan(iq0/id0);

%%%%% Solution for Theta0 %%%%%%%%%%%%%%%%%%%%%

Theta0=atan(vq0/vd0)-atan(Vq/Vd);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear a b c p q x1 x2


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                                            %%
%%   Calculation of the system matrices       %%
%%                                            %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%% d_phi %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
a1=1/(1+(iq0/id0)^2)*iq0/id0/id0;
a2=1/(1+(iq0/id0)^2)/id0;

%%%%%%%%%%%% d_I %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

c1=id0/I0;
c2=iq0/I0;

%%%%%%%%%%%% d_Vt %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

b2=(Vt0*X*cos(pi/2-phi0)-I0*(X^2))/(Vt0-I0*X*cos(pi/2-phi0));
b1=(Vt0*I0*X*sin(pi/2-phi0))/(Vt0-I0*X*cos(pi/2-phi0));
d1=b1*a1+b2*c1;
d2=b2*c2-b1*a2;

%%%%%%%%%%%% d_vd, d_vq %%%%%%%%%%%%%%%%%%%%%%%%%

e1=sin(delta_t0);
e2=Vt0*cos(delta_t0);
e3=cos(delta_t0);
e4=Vt0*sin(delta_t0);

f1=e1*d1;
f2=e1*d2;
f3=e2-e1*b1;
f4=e3*d1;
f5=e3*d2;
f6=e3*b1+e4;

%%%%%%%%%%%% d_id, d_iq %%%%%%%%%%%%%%%%%%%%%%%%%

g1=1+f4/xd__;
g2=1-f2/xq__;

h1=1/g1/xd__;
h2=f5/g1/xd__;
h3=f6/g1/xd__;
h4=1/g2/xq__;
h5=f1/g2/xq__;
h6=f3/g2/xq__;

k1=h1/(1+h2*h5);
k2=h2*h4/(1+h2*h5);
k3=(h3-h2*h6)/(1+h2*h5);
k4=h4-h5*k2;
k5=h5*k1;
k6=h6+h5*k3;

%%%%%%%%%%%% d_vd, d_vq %%%%%%%%%%%%%%%%%%%%%%%%%

l1=f1*k1+f2*k5;
l2=f2*k4-f1*k2;
l3=f1*k3+f3+f2*k6;
l4=f4*k1+f5*k5;
l5=f5*k4-f4*k2;
```

```
l6=f4*k3+f5*k6-f6;

%%%%%%%%%%%% d_P %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

m1=id0*l1+iq0*l4+vd0*k1+vq0*k5;
m2=id0*l2+iq0*l5-vd0*k2+vq0*k4;
m3=id0*l3+iq0*l6+vd0*k3+vq0*k6;

%%%%%%%%%%%% d_w %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n1=w0*m1/2/H;
n2=w0*m2/2/H;
n3=w0*m3/2/H;

%%%%%%%%%%%% d_Theta %%%%%%%%%%%%%%%%%%%%%%%%%%%%

r1=(I0*(X^2))/(Vb*Vt0*sin(Theta0));
r2=(Vb*cos(Theta0)-Vt0)/(Vb*Vt0*sin(Theta0));
r3=r1*c1+r2*d1;
r4=r1*c2+r2*d2;
r5=r2*b1;

s1=r3*k1+r4*k5;
s2=r4*k4-r3*k2;
s3=r3*k3-r5+r4*k6;

%%%%%%%%%%%% d_delta_t %%%%%%%%%%%%%%%%%%%%%%%%%%

t1=1/(1+s3);
t2=s1/(1+s3);
t3=s2/(1+s3);

%%%%%%%%%%%% d_Eq_ %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

alp1=1/Td0_;
alp2=(xd-xd_)/Td0_/xd__;
alp3=alp2;
alp4=k/Td0_;

bet1=alp3*l4-alp2-alp3*l6*t2;
bet2=-alp3*l6*t3+alp3*l5;
bet3=alp3*l6*t1;

%%%%%%%%%%%% d_Eq__ %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

gam1=(1/Td0__-1/Td0_);
gam2=(1/Td0__+(xd-xd_)/Td0_/xd__+(xd_-xd__)/Td0__/xd__);
gam3=k/Td0_;
gam4=(xd-xd_)/Td0_/xd__+(xd_-xd__)/Td0__/xd__;

nu1=gam4*l4-gam2-gam4*l6*t2;
nu2=-gam4*l6*t3+gam4*l5;
nu3=gam4*l6*t1;

%%%%%%%%%%%% d_Ed__ %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
zet1=xq/Tq0__/xq__;
zet2=(xq-xq__)/Tq0__/xq__;

mu1=zet1+zet2*l2-zet2*l3*t3;
mu2=zet2*l3*t2-zet2*l1;
mu3=zet2*l3*t1;

%%%%%%%%%%%% d_w %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
z1=n3*t2-n1;
z2=n2-n3*t3;
z3=n3*t1;

%%%%%%%%%%%% d_id %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A1=k1-k3*t2;
A2=-k3*t3-k2;
A3=k3*t1;

%%%%%%%%%%%% d_iq %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
B1=k5-k6*t2;
B2=k4-k6*t3;
B3=k6*t1;

%%%%%%%%%%%% d_vd, d_vq %%%%%%%%%%%%%%%%%%%%%%%%%
C1=l1-l3*t2;
C2=l2-l3*t3;
C3=l3*t1;
C4=l4-l6*t2;
C5=l5-l6*t3;
C6=l6*t1;

%%%%%%%%%%%% System Matrices %%%%%%%%%%%%%%%%%%%%
A=[alp1 bet1 bet2 0 bet3
   gam1 nu1 nu2 0 nu3
   0 mu2 -mu1 0 -mu3
   0 z1 -z2 -Df/H -z3
   0 0 0 1 0];

B=[alp4 gam3 0 0 0]';

C=[0 A1 A2 0 A3
   0 B1 B2 0 B3
   0 -t2 -t3 0 t1
   0 C1 C2 0 C3
   0 C4 C5 0 C6];

D=[0 0 0 0 0]';

clear a1 a2 c1 c2 b1 b2 d1 d2 e1 e2 e3 e4 f1 f2 f3 f4 f5 f6
clear g1 g2 h1 h2 h3 h4 h5 h6 k1 k2 k3 k4 k5 k6 l1 l2 l3 l4 l5 l6
clear m1 m2 m3 n1 n2 n3 r1 r2 r3 r4 r5 s1 s2 s3 t1 t2 t3
clear alp1 alp2 alp3 alp4 bet1 bet2 bet3 gam1 gam2 gam3 gam4
clear nu1 nu2 nu3 zet1 zet2 mu1 mu2 mu3 z1 z2 z3 A1 A2 A3 B1 B2 B3
clear C1 C2 C3 C4 C5 C6
```

Grade0

```
% grade0(p,u,t,ye)   gradient function for electrical model
%        no parameters fixed

function gf=grade0(p,u,T,y)

xd=p(1);
xd_=p(2);
xd__=p(3);
Td0_=p(4);
Td0__=p(5);
k=p(6);
xq=p(7);
xq__=p(8);
Tq0__=p(9);

a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

A=[a11 a12 0;a21 a22 0;0 0 a33];
B=[0 b12 b13;0 b22 b23;b31 0 0];
C=[0 c12 0;0 0 c23];

x=lsim(A,B,eye(3),zeros(3),u,T);

A11=A(1,1)*eye(9);
A12=A(1,2)*eye(9);
A13=A(1,3)*eye(9);
A21=A(2,1)*eye(9);
A22=A(2,2)*eye(9);
A23=A(2,3)*eye(9);
A31=A(3,1)*eye(9);
A32=A(3,2)*eye(9);
A33=A(3,3)*eye(9);

Ax=[A11 A12 A13;A21 A22 A23;A31 A32 A33];

B11=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__ 0 0 1/Td0_/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2)) 0 0 (xd-xd_)/(Td0_*(xd__^2)) 0
0 -(xd-xd_)/(xd__*(Td0_^2)) -k/(Td0_^2) -1/(Td0_^2) ...
(xd-xd_)/(xd__*(Td0_^2)) 0
```

```
0 0 0 0 0 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B21=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__+1/Td0__/xd__ 0 0 1/Td0_/xd__-1/Td0__/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2))-xd_/(Td0__*(xd__^2)) 0 0 ...
(xd-xd_)/(Td0_*(xd__^2))+xd_/(Td0__*(xd__^2)) 0
0 -(xd-xd_)/((Td0_^2)*xd__) -k/(Td0_^2) 1/(Td0_^2) ...
(xd-xd_)/((Td0_^2)*xd__) 0
0 -(xd_-xd__)/((Td0__^2)*xd__) 0 -1/(Td0__^2) ...
(xd_-xd__)/((Td0__^2)*xd__)+1/(Td0__^2) 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B31=[zeros(6)
-1/Tq0__/xq__ 0 0 0 0 -1/Tq0__/xq__
xq/(Tq0__*(xq__^2)) 0 0 0 0 xq/(Tq0__*(xq__^2))
(xq-xq__)/((Tq0__^2)*xq__) 0 0 0 0 xq/((Tq0__^2)*xq__)];

Bx=[B11;B21;B31];

C11=C(1,1)*eye(9);
C12=C(1,2)*eye(9);
C13=C(1,3)*eye(9);
C21=C(2,1)*eye(9);
C22=C(2,2)*eye(9);
C23=C(2,3)*eye(9);

Cx=[C11 C12 C13;C21 C22 C23];

Dx=zeros(18,6);

Dx(3,2)=1/(xd__^2);
Dx(3,5)=-1/(xd__^2);
Dx(17,1)=-1/(xq__^2);
Dx(17,6)=-1/(xq__^2);

yx=lsim(Ax,Bx,Cx,Dx,[u; x'],T);

J1=yx(:,1:9);
J2=yx(:,10:18);

[m,n]=size(J1);
J=zeros(2*m,n);

for n=1:m
   J(2*n-1,:)=J1(n,:);
   J(2*n,:)=J2(n,:);
end
gf=-J';
```

Grade_1

```
% grade2(p,u,t,ye)   gradient function for electrical model
%        2 fixed parameters

function gf=grade_1(p,u,t,y)

%  Vector of reorderings: p=[8 7 9 5 3 2 1]
%     1     2      3     4     5    6    7     8     9  ]
%  [  xd    xd'    xd"  Tdo' Tdo"  k    xq    xq"  Tqo"
%p=[7 8 9 5 3 4 1]   % Vector of reordering
a=[7 8 9 5 3 4 1];;
n=length(a);      % Number of unfixed parameters

xd=p(7);
xd_=0.285;%fixed
xd__=p(5);
Td0_=p(6);
Td0__=p(4);
k = 47.143;% fixed
xq=p(1);
xq__=p(2);
Tq0__=p(3);


a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

A=[a11 a12 0;a21 a22 0;0 0 a33];
B=[0 b12 b13;0 b22 b23;b31 0 0];
C=[0 c12 0;0 0 c23];

x=lsim(A,B,eye(3),zeros(3),u,t);

A11=A(1,1)*eye(n);
A12=A(1,2)*eye(n);
A13=A(1,3)*eye(n);
A21=A(2,1)*eye(n);
A22=A(2,2)*eye(n);
A23=A(2,3)*eye(n);
A31=A(3,1)*eye(n);
A32=A(3,2)*eye(n);
```

```
A33=A(3,3)*eye(n);

Ax=[A11 A12 A13;A21 A22 A23;A31 A32 A33];

B11=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__ 0 0 1/Td0_/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2)) 0 0 (xd-xd_)/(Td0_*(xd__^2)) 0
0 -(xd-xd_)/(xd__*(Td0_^2)) -k/(Td0_^2) -1/(Td0_^2) ...
(xd-xd_)/(xd__*(Td0_^2)) 0
0 0 0 0 0 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B21=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__+1/Td0__/xd__ 0 0 1/Td0_/xd__-1/Td0__/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2))-xd_/(Td0__*(xd__^2)) 0 0 ...
(xd-xd_)/(Td0_*(xd__^2))+xd_/(Td0__*(xd__^2)) 0
0 -(xd-xd_)/((Td0_^2)*xd__) -k/(Td0_^2) 1/(Td0_^2) ...
(xd-xd_)/((Td0_^2)*xd__) 0
0 -(xd_-xd__)/((Td0__^2)*xd__) 0 -1/(Td0__^2) ...
(xd_-xd__)/((Td0__^2)*xd__)+1/(Td0__^2) 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B31=[zeros(6)
-1/Tq0__/xq__ 0 0 0 0 -1/Tq0__/xq__
xq/(Tq0__*(xq__^2)) 0 0 0 0 xq/(Tq0__*(xq__^2))
(xq-xq__)/((Tq0__^2)*xq__) 0 0 0 0 xq/((Tq0__^2)*xq__)];

B11=B11(a,:);
B21=B21(a,:);
B31=B31(a,:);

Bx=[B11;B21;B31];

C11=C(1,1)*eye(n);
C12=C(1,2)*eye(n);
C13=C(1,3)*eye(n);
C21=C(2,1)*eye(n);
C22=C(2,2)*eye(n);
C23=C(2,3)*eye(n);

Cx=[C11 C12 C13;C21 C22 C23];

Dx=zeros(2*n,6);

Dx(5,2)=1/(xd__^2);
Dx(5,5)=-1/(xd__^2);
Dx(8,1)=-1/(xq__^2);
Dx(8,6)=-1/(xq__^2);
```

```
yx=lsim(Ax,Bx,Cx,Dx,[u' x],t);

J1=yx(:,1:n);
J2=yx(:,n+1:2*n);

[m,n]=size(J1);
J=zeros(2*m,n);
for n=1:m
   J(2*n-1,:)=J1(n,:);
   J(2*n,:)=J2(n,:);
end

gf=-J';


%J1=yx(:,1:7);
%J2=yx(:,8:14);

%[m,n]=size(J1);
%J=zeros(2*m,n);

%for n=1:m
%  J(2*n-1,:)=J1(n,:);
%  J(2*n,:)=J2(n,:);
%end

%gf=-J';
```

Grade_2

```
% grade2(p,u,t,ye)   gradient function for electrical model
%        2 fixed parameters

function gf=grade_2(p,u,t,y)

%     1     2      3     4     5   6    7     8     9  ]
%  [  xd    xd'    xd"  Tdo' Tdo"  k    xq    xq"  Tqo"

%p=[7 8 9 5 3 6 1];  % Vector of reordering
a=[7 8 9 5 3 6 1];
n=length(a);      % Number of unfixed parameters

xd=p(7);
xd_=0.285; %fixed
xd__=p(5);
Td0_=3.7;   % fixed
Td0__=p(4);
k =p(6);
xq=p(1);
xq__=p(2);
Tq0__=p(3);
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
```

```
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

A=[a11 a12 0;a21 a22 0;0 0 a33];
B=[0 b12 b13;0 b22 b23;b31 0 0];
C=[0 c12 0;0 0 c23];

x=lsim(A,B,eye(3),zeros(3),u,t);

A11=A(1,1)*eye(n);
A12=A(1,2)*eye(n);
A13=A(1,3)*eye(n);
A21=A(2,1)*eye(n);
A22=A(2,2)*eye(n);
A23=A(2,3)*eye(n);
A31=A(3,1)*eye(n);
A32=A(3,2)*eye(n);
A33=A(3,3)*eye(n);

Ax=[A11 A12 A13;A21 A22 A23;A31 A32 A33];

B11=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__ 0 0 1/Td0_/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2)) 0 0 (xd-xd_)/(Td0_*(xd__^2)) 0
0 -(xd-xd_)/(xd__*(Td0_^2)) -k/(Td0_^2) -1/(Td0_^2) ...
(xd-xd_)/(xd__*(Td0_^2)) 0
0 0 0 0 0 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B21=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__+1/Td0__/xd__ 0 0 1/Td0_/xd__-1/Td0__/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2))-xd_/(Td0__*(xd__^2)) 0 0 ...
(xd-xd_)/(Td0_*(xd__^2))+xd_/(Td0__*(xd__^2)) 0
0 -(xd-xd_)/((Td0_^2)*xd__) -k/(Td0_^2) 1/(Td0_^2) ...
(xd-xd_)/((Td0_^2)*xd__) 0
0 -(xd_-xd__)/((Td0__^2)*xd__) 0 -1/(Td0__^2) ...
(xd_-xd__)/((Td0__^2)*xd__)+1/(Td0__^2) 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B31=[zeros(6)
-1/Tq0__/xq__ 0 0 0 0 -1/Tq0__/xq__
```

```
xq/(Tq0__*(xq__^2)) 0 0 0 0 xq/(Tq0__*(xq__^2))
(xq-xq__)/((Tq0__^2)*xq__) 0 0 0 0 xq/((Tq0__^2)*xq__)];

B11=B11(a,:);
B21=B21(a,:);
B31=B31(a,:);

Bx=[B11;B21;B31];

C11=C(1,1)*eye(n);
C12=C(1,2)*eye(n);
C13=C(1,3)*eye(n);
C21=C(2,1)*eye(n);
C22=C(2,2)*eye(n);
C23=C(2,3)*eye(n);

Cx=[C11 C12 C13;C21 C22 C23];

Dx=zeros(2*n,6);

Dx(6,2)=1/(xd__^2);
Dx(6,5)=-1/(xd__^2);
Dx(8,1)=-1/(xq__^2);
Dx(8,6)=-1/(xq__^2);


yx=lsim(Ax,Bx,Cx,Dx,[u' x],t);

J1=yx(:,1:n);
J2=yx(:,n+1:2*n);

[m,n]=size(J1);
J=zeros(2*m,n);
for n=1:m
   J(2*n-1,:)=J1(n,:);
   J(2*n,:)=J2(n,:);
end

gf=-J';


%J1=yx(:,1:7);
%J2=yx(:,8:14);

%[m,n]=size(J1);
%J=zeros(2*m,n);

%for n=1:m
%  J(2*n-1,:)=J1(n,:);
%  J(2*n,:)=J2(n,:);
%end

%gf=-J';
```

Error_1

```
% erre2(p,u,t,ye) error function for electrical model
%        2 fixed parameters
%
%  Vector of reorderings: p=[8 7 9 5 3 2 1]
%     1     2      3     4     5   6    7     8     9  ]
%  [  xd    xd'    xd"  Tdo' Tdo"  k    xq    xq"  Tqo"
%p=[7 8 9 5 3 4 1]

function e = erre2(p,u,t,y)

xd=p(7);
xd_=0.285;%fixed
xd__=p(5);
Td0_=p(6);
Td0__=p(4);
k = 47.143;% fixed
xq=p(1);
xq__=p(2);
Tq0__=p(3);

a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

d12=-1/xd__;
d21=1/xq__;

Ax=[a11 a12 0;a21 a22 0;0 0 a33];
Bx=[0 b12 b13;0 b22 b23;b31 0 0];
Cx=[0 c12 0;0 0 c23];
Dx=[0 d12 0;d21 0 0];

yx=lsim(Ax,Bx,Cx,Dx,u,t);

er=y-yx;
[m,n]=size(er);
e=zeros(m*n,1);
for i=1:m
e([2*i-1:2*i],1)=er(i,:)';
end
```

Error_2

```
% erre2(p,u,t,ye) error function for electrical model
%        2 fixed parameters
%

%     1     2      3     4     5   6    7     8     9  ]
%  [  xd    xd'    xd"  Tdo' Tdo"  k    xq    xq"  Tqo"

%p=[7 8 9 5 3 6 1];  % Vector of reordering

function e = erre22(p,u,t,y)

xd=p(7);
xd_=0.285; %fixed
xd__=p(5);
Td0_=3.7;   % fixed
Td0__=p(4);
k =p(6);
xq=p(1);
xq__=p(2);
Tq0__=p(3);


a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

d12=-1/xd__;
d21=1/xq__;

Ax=[a11 a12 0;a21 a22 0;0 0 a33];
Bx=[0 b12 b13;0 b22 b23;b31 0 0];
Cx=[0 c12 0;0 0 c23];
Dx=[0 d12 0;d21 0 0];

yx=lsim(Ax,Bx,Cx,Dx,u,t);

er=y-yx;
[m,n]=size(er);
e=zeros(m*n,1);
for i=1:m
e([2*i-1:2*i],1)=er(i,:)';
end
```

Gauss_e

```
% This file implement the Gauss-Newton algorithm
% for parameter estimation
% Problem of the FC5HP synchronous generator PSERC-ASU
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
load('FC5HP_1.txt');%PSERC-ASU data
T=FC5HP_1(:,2);%time (us)
T=T*1E-6; %time(sec)ojo..
%vfd=(0.0518153496/199912.79)*FC5HP_1(:,6);%d_axis field voltage(pu)
vfd=(0.0518153496/153510)*FC5HP_1(:,6);%2nd assumption

% n= rand(1,length(vfd));
% m=0.05*n;
% vfd=vfd.*m';% noise
gen_sim;
[y,x]=lsim(A,B,C,D,vfd,T);
vd=y(:,4)';
vq=y(:,5)';
u=[vd;vq;vfd'];
elmod;
[ye,xe]=lsim(Ae,Be,Ce,De,u,T);
figure(1)
plot(T,ye);
figure(2);
plot(T,vfd);
%plot(T,u);
%Nominal values of the parameters
p1=[1.801 0.285 0.220 3.7 0.032 47.143 1.72 0.220 0.059]';%GAUSS-PSERC
p=.65*p1;%initial guess of the parameters to be estimated
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Gauss Newton Algorithm
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
alph=1; alp=0*p0;
i=0;p_prev=p0; change=100000;
%Gauss Newton implementation
while ((i<=500)&(change>=0.0001))
    e=erre0(p0,u,T,ye);
    J=(grade0(p0,u,T,ye))';%Jacobian matrix
    h=inv(J'*J)*J'*e;%search direction
    %p0=p0+(alph)^i*h'
    p0=p0-alph*h% parameter to be estimated
    i=i+1
    alp=p0

    change=norm(p_prev-alp)/norm(p_prev)
    norm(erre0(p0,u,T,ye))
    p_prev=alp;
end


%%%%Relative error%%%%%
[f,c]=size(p);
```

```
for n=1:f
   er(n)=(p(n,c)-alp(n,c))*100/p(n,c);
end
p
p0
E=((p-p0)*100)./p
%%%condition analysis%%%%%
H=J'*J;
conditionH=cond(H)
eigenvaluesH = svd(H)
%%%%%Fast Fourier Transf.%%%%%%%%
Y = fft(vfd,900);
Pyy = Y.* conj(Y) / 900;
f = 1000*(0:256)/900;
figure(3)
plot(f,Pyy(1:257))
title('Frequency content of vfd')
xlabel('frequency (Hz)')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

B.2 VDGN FILES

Table B. 2 Main Functions for VDNR method

Function	Description
Elmode	This file calculates the electrical subsystem matrices Ae, Be, Ce and De using the parameter set of FC5HP synchronous generator
Gen_sim	This file calculates the system matrices A,B,C and D using the parameter set of FC5HP synchronous generator
Erre_0	.Calculates the Jacobian for electrical for full order model,
Grade_0	Calculates the Jacobian for electrical for full order model,
Grade_VDGN	Calculates the Jacobian for electrical model, For VDGN. algorithm
Step_3	Calculates the size h of the search step *.
Step_4	Calculates the tracking step of the VDNR algorithm.
Step_12*	Calculates the new set of indexes (β)of functions which have been removed from ρ
VDNR Algorithm	Computes the VDNR algorithm .

Erre_0

```
% erre0(p,u,t,ye) error function for electrical model

% global ct
% ct=ct+1
%  p=[1 2 3 4 5 6 7 8 9]

function e = erre0(p,u,T,ya)

xd=p(1);     % 1. parameter to be estimated
xd_=p(2);    % 2. parameter to be estimated
xd__=p(3);   % 3. parameter to be estimated
Td0_=p(4);   % 4. parameter to be estimated
Td0__=p(5);  % 5. parameter to be estimated
k=p(6);            % 6. parameter to be estimated
xq=p(7);     % 7. parameter to be estimated
xq__=p(8);   % 8. parameter to be estimated
Tq0__=p(9);  % 9. parameter to be estimated

a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
```

```
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

d12=-1/xd__;
d21=1/xq__;

Ax=[a11 a12 0;a21 a22 0;0 0 a33];
Bx=[0 b12 b13;0 b22 b23;b31 0 0];
Cx=[0 c12 0;0 0 c23];
Dx=[0 d12 0;d21 0 0];

yx=lsim(Ax,Bx,Cx,Dx,u,T);

er=ya-yx;
[m,n]=size(er);
e=zeros(m*n,1);
for i=1:m
e([2*i-1:2*i],1)=er(i,:)';
end
```

Grade_0

```
% grade0(p,u,t,ye)      gradient function for electrical model

function gf=grade0(p,u,T,y)

xd=p(1);
xd_=p(2);
xd__=p(3);
Td0_=p(4);
Td0__=p(5);
k=p(6);
xq=p(7);
xq__=p(8);
Tq0__=p(9);

a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
```

```
c23=1/xq__;

A=[a11 a12 0;a21 a22 0;0 0 a33];
B=[0 b12 b13;0 b22 b23;b31 0 0];
C=[0 c12 0;0 0 c23];

x=lsim(A,B,eye(3),zeros(3),u,T);


A11=A(1,1)*eye(9);
A12=A(1,2)*eye(9);
A13=A(1,3)*eye(9);
A21=A(2,1)*eye(9);
A22=A(2,2)*eye(9);
A23=A(2,3)*eye(9);
A31=A(3,1)*eye(9);
A32=A(3,2)*eye(9);
A33=A(3,3)*eye(9);

Ax=[A11 A12 A13;A21 A22 A23;A31 A32 A33];

B11=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__ 0 0 1/Td0_/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2)) 0 0 (xd-xd_)/(Td0_*(xd__^2)) 0
0 -(xd-xd_)/(xd__*(Td0_^2)) -k/(Td0_^2) -1/(Td0_^2) ...
(xd-xd_)/(xd__*(Td0_^2)) 0
0 0 0 0 0 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B21=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__+1/Td0__/xd__ 0 0 1/Td0_/xd__-1/Td0__/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2))-xd_/(Td0__*(xd__^2)) 0 0 ...
(xd-xd_)/(Td0_*(xd__^2))+xd_/(Td0__*(xd__^2)) 0
0 -(xd-xd_)/((Td0_^2)*xd__) -k/(Td0_^2) 1/(Td0_^2) ...
(xd-xd_)/((Td0_^2)*xd__) 0
0 -(xd_-xd__)/((Td0__^2)*xd__) 0 -1/(Td0__^2) ...
(xd_-xd__)/((Td0__^2)*xd__)+1/(Td0__^2) 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B31=[zeros(6)
-1/Tq0__/xq__ 0 0 0 0 -1/Tq0__/xq__
xq/(Tq0__*(xq__^2)) 0 0 0 0 xq/(Tq0__*(xq__^2))
(xq-xq__)/((Tq0__^2)*xq__) 0 0 0 0 xq/((Tq0__^2)*xq__)];

Bx=[B11;B21;B31];

C11=C(1,1)*eye(9);
C12=C(1,2)*eye(9);
C13=C(1,3)*eye(9);
```

```
C21=C(2,1)*eye(9);
C22=C(2,2)*eye(9);
C23=C(2,3)*eye(9);

Cx=[C11 C12 C13;C21 C22 C23];

Dx=zeros(18,6);

Dx(3,2)=1/(xd__^2);
Dx(3,5)=-1/(xd__^2);
Dx(17,1)=-1/(xq__^2);
Dx(17,6)=-1/(xq__^2);

yx=lsim(Ax,Bx,Cx,Dx,[u; x'],T);

J1=yx(:,1:9);
J2=yx(:,10:18);

[m,n]=size(J1);
J=zeros(2*m,n);

for n=1:m
      J(2*n-1,:)=J1(n,:);
      J(2*n,:)=J2(n,:);
end

gf=-J';
```

Grade-VDGN

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     gradient function for electrical model
%     variable dimension
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function gf=gradeVDNR(alpha,p,u,T,y)
%****
n=length(alpha);
%****
xd=p(1);
xd_=p(2);
xd__=p(3);
Td0_=p(4);
Td0__=p(5);
k=p(6);
xq=p(7);
xq__=p(8);
Tq0__=p(9);

%******
%p=p(alpha,:)
%******
a11=1/Td0_;
a12=-(xd-xd_)/(Td0_*xd__);
a21=1/Td0__-1/Td0_;
a22=-(1/Td0__+(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__));
```

```
a33=-xq/(Tq0__*xq__);

b12=(xd-xd_)/(Td0_*xd__);
b13=k/Td0_;
b22=(xd-xd_)/(Td0_*xd__)+(xd_-xd__)/(Td0__*xd__);
b23=k/Td0_;
b31=-(xq-xq__)/(Tq0__*xq__);

c12=1/xd__;
c23=1/xq__;

A=[a11 a12 0;a21 a22 0;0 0 a33]
B=[0 b12 b13;0 b22 b23;b31 0 0]
C=[0 c12 0;0 0 c23]
T=T
u=u
x=lsim(A,B,eye(3),zeros(3),u,T)


A11=A(1,1)*eye(n);
A12=A(1,2)*eye(n);
A13=A(1,3)*eye(n);
A21=A(2,1)*eye(n);
A22=A(2,2)*eye(n);
A23=A(2,3)*eye(n);
A31=A(3,1)*eye(n);
A32=A(3,2)*eye(n);
A33=A(3,3)*eye(n);

Ax=[A11 A12 A13;A21 A22 A23;A31 A32 A33];

B11=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__ 0 0 1/Td0_/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2)) 0 0 (xd-xd_)/(Td0_*(xd__^2)) 0
0 -(xd-xd_)/(xd__*(Td0_^2)) -k/(Td0_^2) -1/(Td0_^2) ...
(xd-xd_)/(xd__*(Td0_^2)) 0
0 0 0 0 0 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];

B21=[0 1/Td0_/xd__ 0 0 -1/Td0_/xd__ 0
0 -1/Td0_/xd__+1/Td0__/xd__ 0 0 1/Td0_/xd__-1/Td0__/xd__ 0
0 -(xd-xd_)/(Td0_*(xd__^2))-xd_/(Td0__*(xd__^2)) 0 0 ...
(xd-xd_)/(Td0_*(xd__^2))+xd_/(Td0__*(xd__^2)) 0
0 -(xd-xd_)/((Td0_^2)*xd__) -k/(Td0_^2) 1/(Td0_^2) ...
(xd-xd_)/((Td0_^2)*xd__) 0
0 -(xd_-xd__)/((Td0__^2)*xd__) 0 -1/(Td0__^2) ...
(xd_-xd__)/((Td0__^2)*xd__)+1/(Td0__^2) 0
0 0 1/Td0_ 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0];
```

```
B31=[zeros(6)
-1/Tq0__/xq__ 0 0 0 0 -1/Tq0__/xq__
xq/(Tq0__*(xq__^2)) 0 0 0 0 xq/(Tq0__*(xq__^2))
(xq-xq__)/((Tq0__^2)*xq__) 0 0 0 0 xq/((Tq0__^2)*xq__)];
%*****
B11=B11(alpha,:);
B21=B21(alpha,:);
B31=B31(alpha,:);
%*****
Bx=[B11;B21;B31];

C11=C(1,1)*eye(n);
C12=C(1,2)*eye(n);
C13=C(1,3)*eye(n);
C21=C(2,1)*eye(n);
C22=C(2,2)*eye(n);
C23=C(2,3)*eye(n);

Cx=[C11 C12 C13;C21 C22 C23];

Dx1=zeros(18,6);
Dx1(3,2)=1/(xd__^2);
Dx1(3,5)=-1/(xd__^2);
Dx1(17,1)=-1/(xq__^2);
Dx1(17,6)=-1/(xq__^2);
%***********
%D11=Dx1(1:n,:);
%D21=Dx1((n+1):(2*n),:);
D11=Dx1(1:9,:);
D21=Dx1((9+1):(2*9),:);
D11=D11(alpha,:);
D21=D21(alpha,:);

Dx=[D11;D21];
%***********

yx=lsim(Ax,Bx,Cx,Dx,[u; x'],T);

J1=yx(:,1:n);
J2=yx(:,n+1:2*n);

[m,n]=size(J1);
J=zeros(2*m,n);

for n=1:m
      J(2*n-1,:)=J1(n,:);
      J(2*n,:)=J2(n,:);
end

gf=-J';
```

Step_3

```
function ei=step3(alpha,F,h,n)
for i=1:n
    inters=intersect(i,alpha)
    if (inters);
        ei(i,1)=0;
    else
        if F(i,1) >= 0
            ei(i,1)= -h;
        else ei(i,1)= h;
        end
    end
end
```

Step_4

```
function [pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p,maxiter);

%j=0;
ef=0.0001;
iter=0
%pc=pc0%p+ei;%correction step
n1=length(pc);
p_ind=(1:n1); % vector of indices of "pc"
set=setdiff(p_ind,alpha);% indices of "pc" that are not in "alpha"
Fk=F
 while ((iter <=maxiter)& (max(abs(Fk))) > ef ) %(change>=0.01))
     pc=pc
     e=erre0(pc,u,T,ye);%2048*1
            %J=(grade0(pc,u,T,ye))'%2048x9
            J=(gradeVDNR(alpha,pc,u,T,ye))';%2048xm
            Fk=J'*e; %mx1
            H_=inv(J'*J); %mxm
            pc(alpha) = pc(alpha,:) - H_*Fk %(mxm)(mx1)=nx1
            pc(set)=p(set,:) %(m-n)x1...9x1
            %j=j+1;

            iter =iter+1
 end
```

Step_12

```
function [alpha,beta,n3]=step12(ei,JF,alpha)

        gamma=ei'*JF;%(1x9)(9x9)=(1x9)
        [n1,m1]=size(gamma);
        u1=0.9;%user defined variable
        in=u1*norm(gamma,inf);
        for k=1:m1
            if abs(gamma(1,k)) > in
                 beta = k %set of ind func wich have been removed
```

```
                alpha=setdiff(alpha,beta)
                n3=alpha
                %alpha(1,k)= 0;
           end
        end
```

VDGN Algorithm

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file implement the VDNR algorithm              %
% for parameter estimation                             %
% Problem of the FC5HP synchronous generator PSERC-ASU  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
load('FC5HP_1.txt');%PSERC-ASU data
T=FC5HP_1(:,2);%time (us)
T=T*1E-6; %time(sec)..
%vfd=(0.0518153496/199912.79)*FC5HP_1(:,6);%d_axis field voltage(pu)
vfd=(0.0518153496/153510)*FC5HP_1(:,6);%2nd assumption

% n= rand(1,length(vfd));
% m=0.05*n;
gen_sim;
[y,x]=lsim(A,B,C,D,vfd,T);
vd=y(:,4)';
vq=y(:,5)';
u=[vd;vq;vfd'];
elmod;
[ye,xe]=lsim(Ae,Be,Ce,De,u,T);
figure(1)
plot(T,ye);
figure(2);
plot(T,vfd);
%plot(T,u);
%Nominal values of the parameters
p1=[1.801 0.285 0.220 3.7 0.032 47.143 1.72 0.220 0.059]';%VDNR(PSERC
p=.65*p1;%initial guess of the parameters to be estimated

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% VDNR algorithm
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
beta=[];
k=0;
h=0.1;% search step size
ef=10e-6%0.001%user relative error tolerance
et=.01%user def. tol. det when the tracking should be stopped
iter1=1;
iter=0;
maxiter=9;
alpha=[]%
%Step 2
```

```
%residual computing
n0=length(p1);
n2=alpha;
n3=[1 4];
kmax=5;
while (iter < iter1)% | (length(alpha)==0)
       %if lenght(alpha)>0
        e=erre0(p,u,T,ye);%2048x1
        J1=(grade0(p,u,T,ye))';%2048x9..Ji
        %J=(gradeVDNR(alpha,p,u,T,ye))';
        JF=J1'*J1;%9x9
        JFm=JF'*inv(JF*JF');%9x9
        F=J1'*e;%9x1 %F(k)
        [n,m]=size(F);

        n2=n3;
            %step 3
        ei=step3(alpha,F,h,n);
         %%%%%%%%%%%%%%%%%%%%%%%
         %step4
         %pc0=p+ei;
        pc=p+(eye(length(p))-JFm*JF)*ei;
        if length(alpha)~= 0
         %[pc,iter,Fk,J]=step4_m21(pc0,F,alpha,u,T,ye);%pcj1
        [pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p,maxiter);
        end
        %step 5
        if (iter < maxiter)| length(alpha)== 0

            iter=iter
            pk1 = pc; % step6...p(k+1)=p(c,j)
            %for step8
            %step7
            e=erre0(pk1,u,T,ye);
            J=(grade0(pk1,u,T,ye))';
            Fk1=J'*e;
            for i=1:n
                i=i
                un=union(alpha,beta);
                inter2 = intersect(i,un);
                [mm,nn]=size(inter2);
                cond2=(Fk1(i,1)*F(i,1));
                if (cond2<0) & (nn==0)%or isempty(iter2)
                    alpha =union(alpha,i) %union(i,alpha);

                    %and goto step 9
                     pc=pk1; %p(c,0)=p(k+1)
                    [pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p1,maxiter);

                  if iter > maxiter%goto step12
                        [alpha,beta,n3]=step12(ei,JF,alpha);
                        %step 3
                        ei=step3(alpha,F,h,n);
                        %step4
                        pc=p+(eye(length(p))-JFm*JF)*ei;
```

```
                    pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p,maxiter);
                     break
                 elseif iter<=maxiter %step 11  %
                     p = pc; %   p(k+1)=p(c,j)
                     beta=[];
                     iter=maxiter;% goto step 2
                     break
                                       %k=k+1;
                end
             end
         end
         if iter <=maxiter
         %step8
             for i=1:n
             i=i
             un=union(alpha,beta);
             inter2 = intersect(i,un);
             [mm,nn]=size(inter2);
                 if (nn==0)&(((pk1(i,1)-p(i,1))./(ei(i,1)))<=et)

                     [alpha,beta,n3]=step12(ei,JF,alpha)%goto step 12
                     %step 3
                     ei=track(alpha,F,h,n);
                     pc=p+(eye(length(p))-JFm*JF)*ei;
                     [pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p,maxiter);
                     break
                 elseif nn==0% goto step4
                 pc=p+(eye(length(p))-JFm*JF)*ei;
                 [pc,iter]=step4_m21(pc,F,alpha,u,T,ye,p,maxiter);
                 break
                 end
                 k=k+1;
             end
         end

     end

     if iter > maxiter
      [alpha,beta,n3]=step12(ei,JF,alpha)%step12
     end
     iter1=iter1+1
     alpha
 end


%%%Relative Error%%%%
pc
p1
E=(p1-pc)*100./p1.
```