SOLUTION OF THE ILL-CONDITIONED LOAD FLOW PROBLEM BY THE TENSOR METHOD

by

Li Fan

B.Eng. (Tianjin University, China)

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering

Department of Electrical Engineering
McGill University
Montréal, Canada

June 1989

ABSTRACT

A new load flow calculation method, called the tensor method, is introduced in this thesis. The tensor method is a general purpose method, intended especially for solving the ill-conditioned power systems load flow problems. The method bases each iteration on a quadratic model of the nonlinear algebraic load flow functions, the standard linear model augmented by a simple second order term. The second order term is selected so that the model interpolates function values from several previous iterates, the current function value and the Jacobian. A distinguishing feature of this tensor model with respect to the previous second order load flow models is that it is actually only quadratic in a small system (p equations) and linear in a large system (n-p equations), where n is the dimension of the problem and p is chosen to be equal or less than the square root of n A solution algorithm is proposed first solving the small quadratic p equations using an iterative nonlinear equation solver, then solving the large linear n-p equations using a forward elimination and back substitution technique. In extensive simulation tests, the tensor method demonstrated superior convergence characteristics and numerical stability over the standard load flow methods, both on well- and ill-conditioned test systems.

RESUME

Cette thèse propose une nouvelle technique de calcul pour les équations d'écoulement de puissance dans les grands réseaux électriques. Cette méthode itérative, appelée la méthode tensorielle, est tout à fait générale, mais elle est bien adaptée au systèmes d'équations mal conditionées. Pour le calcul itératif, cette méthode rajoute un terme quadratique au modèle des fonctions à évaluer, en plus du terme lineaire habituel. Ce nouveau terme est trouvé à coût minime par interpolation, à partir de la valeur présente et de valeurs antérieures de la fonction et la valeur présente du Jacobian. Parmi les méthodes quadratiques proposées pour l'écoulement de puissance, celle-ci est la première à traiter les termes quadratiques comme des variables. De par la structure du terme quadratique, il est possible de résoudre le problème quadratique à chaque itération en le décomposant en un petit systeme quadratique (à p équations) et un grand système linéaire (à n-p équations), où n est la dimension du problème et p est plus petit ou egal à sa racine carrée. Nous avons préparé un algorithme pouvant resoudre ces deux systemes, le premier à l'aide d'un programme iteratif et le deuxième par elimination gaussienne Nos essais avec cet algorithme demontrent la stabilite et l'excellente convergence de la methode tensorielle, en comparaison avec les methodes standards, pour toute une gamme de problèmes, dont quelques uns considéres tres difficiles.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Prof. M. Huneault for his excellent guidance, constant encouragement, and warm friendship during the course of this research.

The counsel and advice of Prof. B.T. Ooi is also greatly appreciated.

Thanks are also duo to my many friends and colleagues in the power group for their help, their good-natured company and their enthusiasm. Especially, Fulin Zhuang and Jinan Huang for readily sharing their knowledge, and for generously granting numerous favours Special thanks are also extended to Xiao Wang, Jian Liu, Yan Guo, Bakari Mwinyiwiwa, Nelson Lujara, Lester Loud, Abdelhafid Hellal and Louis Martin for their friendship.

The assistance and support from my dear aunt Florence Fan Chu and her family are also gratefully appreciated.

Finally, I would like to thank my parents and brothers for their continuous encouragement and support during the course of my graduate studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter I

Introduction

1.1 What is the Load Flow Problem ?

Load flow, or power flow, is the solution for the normal balanced three-phase steady state operating conditions of an electric power system. In general, load flow calculations are performed for power system planning and operational planning, and in connection with system operation and control. The data obtained from load flow studies are used for the studies of normal operating mode, contingency analysis, outage security assessment, as well as optimal dispatching and stability.

The difficulties and the importance of the load flow problem have fascinated mathematicans and engineers throughout the world for a number of years. Many people have devoted a large portion of their professional life to the solution of the problem. It has received more attention than any other power system problem. The amount of effort devoted to the problem has resulted in an enormous amount of technical publications. The nature of the problem probably precludes the development of a perfect procedure. Therefore, it is likely that progress will continue to be made on improved solutions for a long time.

The load flow problem can be defined as the calculation of the real and reactive powers flowing in each line and the magnitude and phase angle of the voltage at each bus of a given transmission system for specified generation and load conditions. The information obtained from the load flow studies can be used to test the system's capability to transfer energy from generation to load without overloading lines and to determine the adequacy of voltage regulation by shunt capacitors, shunt reactors, tap-changing transformers, and the var-supplying capability of rotating machines.

In load flow studies the basic assumption is that the given power system is a balanced three-phase system operating in its steady state with a constant frequency (50 or 60Hz) Therefore, the system can be represented by its single-phase positive sequence network with lumped series and shunt branches. The load flow problem can be solved either by using the nodal admittance matrix Y or the bus impedance matrix Z representation of the given network, but it is customary to use the nodal analysis approach. Mathematically, the problem is to solve a set of nonlinear algebraic equations for the complex bus voltages and then determine the line flows

Each bus of a network has four variable quantities associated with it: the real and reactive power, the line to ground voltage magnitude, and voltage phase angle. Any two of the four may be specified, and hence become independent variables, whereas the other two remain to be determined.

In general, there are three types of buses in a load flow problem, each with its own specified variables. (1) slack bus, (2) generator buses, and (3) load buses. Since transmission losses in a given system are associated with the bus voltage profile, until a solution is obtained the total power generation requirement cannot be determined. Therefore, the generator at the slack bus is used to supply the additional real and reactive power owing to the transmission losses. Thus, at the slack bus, the magnitude and phase angle of the voltage are known, and the real and reactive power generated are the quantities to be determined. It is only after a solution has converged, that is, after all bus voltages are known, that the real and reactive power generation requirements at the slack bus can be determined. In other words, the losses are not known in advance, and consequently the power at the slack bus connot be specified.

In order to define the load flow problem to be solved, it is necessary to specify the real power and the voltage magnitude at each generator bus. This is because these quantities are controllable through the governor and excitation controls, respectively. The generator bus is also known as a PV bus. The load bus is also known as a PQ bus. This is due to the fact that the real and reactive powers are specified at a given load bus. It is possible that some load buses may have transformers capable of tap-changing and phase-shifting operations These types of load buses are known as the voltage-controlled load buses. At the voltage-controlled load buses, the known quantities are usually the voltage magnitude in addition to the real and reactive powers, and the unknown quantities are usually the voltage phase angle and the turns ratio.

1.2 History of Load Flow Calculation

Before 1929 all load flow calculations were made by hand. In 1929, network calculators (of Westinghouse) or network analyzers (of

General Electric) were employed to perform load flow calculations. The first paper describing a digital method to solve the load flow problem was published in 1954 [Dunstan 1954]. However, the first really successful digital method was developed by [Ward & Hale 1956] Most of the early iterative methods were based on the Y-matrix approach of the Gauss-Seidel method. It requires minimum computer storage and needs only a small number of iterations for small networks Unfortunately, the number of required iterations can increase dramatically for large systems. In some cases, the method does not provide a solution at all.

Therefore, the slowly converging behavior of the Gauss-Seidel method and its frequent failure to converge in ill-conditioned situations caused the development of the Z-matrix methods for Gauss-Seidel [Brameller & Denmead 1962]. Even though these methods have considerably better convergence characteristics, they also have the disadvantage of needing a significantly larger computer storage memory owing to the fact that the Z-matrix is full, contrary to the Y-matrix which is sparse.

These difficulties encountered in load flow studies led to the implementation of the Newton-Raphson method. The method was originally developed in load flow by [Van Ness 1959] [Van Ness & Griffin 1961] and later improved by [Tinney & Hart 1967] [Dommel et.al. 1970]. The method is based on the Newton-Raphson algorithm to solve the simultaneous quadratic equations of the power network. Contrary to the Gauss-Seidel algorithm, it needs a longer time per iteration, but usually converges in only a few iterations, and convergence is significantly independent of the network size Therefore, most of the load flow problems that could not be solved by the Gauss-Seidel method are solved with no difficulty by this method. The development of a very efficient sparsity-programmed ordered elimination technique by [Tinney & Walker 1967] [Sato & Tinney 1963] to solve the simultaneous equations has enhanced the efficiency of the Newton-Raphson method, in terms of speed and storage requiements, and has made it the most widely used load flow method. The method has been further improved by the addition of automatic controls and adjustments (e.g , program controlled in-phase tap changes, phase-angle regulators, shunt compensation and area interchange control).

In order to further speed up the Newton-Raphson load flow algorithm and to substantially reduce the storage requirement, the Fast Decoupled load flow method has been developed by [Stott & Alsac 1974]. The method is based on the fact that in any power transmission network operating in the steady state, the coupling between P-θ (active powers and bus voltage angles) and Q-V (reactive powers and bus voltage magnitudes) is relatively weak, contrary to the strong coupling between P and θ and between Q and V. Therefore, the method solves the load flow problem by "decoupling" the P-θ and Q-V problems, and also takes advantage of some approximations in forming the constant Jacobian submatrices B' and B''. Because of its simplicity, speed, and small storage requirements, it is being widely used to replace the Newton-Raphson method in large system load flow calculations, multiple-case load flow calculations, on-line applications and contingency security assessments.

Another interesting development in the load flow problem has been that of Second Order load flow in which the basic load flow equations are expressed exactly by a set of quadratic algebraic equations in rectangular coordinates [Iwamoto & Tamura 1978] [Nagendra Rao et.al. 1982]. By using this accurate load flow model, it is intended to achieve better convergence characteristics over the Fast Decoupled load flow method on problems where the system has lines with high R/X ratios, and at the sime time retains all the other advantages of the Fast Decoupled load flow method. This attempt has been successful. Actually, this method is now being implemented in the electric industry in Japan [Iwamoto 1989]; it has not caught on elsewhere, however.

The present trends are towards the development of methods to solve the ill-conditioned system load flow problem [Iwamoto & Tamura 1981] [Tripathy et.al. 1982] and the development of interactive load flow programs [Lynch 1979].

1.3 Ill-Conditioning

Mathematically, a matrix is said to be ill-conditioned if it is extremely sensitive to small changes. Let us consider an example:

$$A = \begin{bmatrix} 1. & 1. \\ 1. & 1.0001 \end{bmatrix}$$

Qualitatively, A is nearly singular . If we change the last entry of A to $a_{22} = 1$, it is singular and the two columns become the same. Consider two very close right-hand sides for A:

$$(1) \qquad \begin{aligned} u + v &= 2 \\ u + 1.0001\, v &= 2 \end{aligned}$$

$$(2) \qquad \begin{aligned} u + v &= 2 \\ u + 1.0001\, v &= 2.0001 \end{aligned}$$

The solution to the first problem is $u = 2$, $v = 0$; the solution to the second is $u = v = 1$. A change in the fifth digit of b in $Ax = b$ amplified into a first digit change in the solution. Therefore, we say that matrix A is ill-conditioned.

In the load flow calculations, we sometimes encounter the same kind of problem, when dealing with a linearized load flow model. This model has the following form:

$$J\, \Delta x = F \qquad (1.1)$$

where J is the load flow Jacobian matrix,

Δx is the bus voltage state correction vector,

F is the power mismatch function vector.

In some power systems, the load flow Jacobian matrix J in equation (1.1) is very ill-conditioned. This causes the instability and/or divergence of the load flow solutions. We define this kind of problem as an ill-conditioned load flow problem.

The features which cause the instability and/or divergence in the power systems load flow calculations are the following:

1. bad choice of the slack bus,
2. large number of radial lines,
3. heavily loaded network,
4. existance of negative line reactance,
5. lines with high R/X ratios,
6. atypical circuit parameters.

A power system which has one or more of the above characteristics is likely to be ill-conditioned. So far, there has been no practical method to solve the ill-conditioned power system load flow calculations problem.

1.4 The Present Study

1.4.1 Objective

The intended objective of the present study is to apply the tensor method proposed by [Schnabel & Frank 1984] to solve the ill-conditioned power system load flow calculation problem.

1.4.2 Outline of the thesis

The chapters of this thesis are organized as follows:

Chapter I — Introduction

This chapter presents a comprehensive understanding of the nature of the load flow problem being solved First, we describe the load flow calculation problem, the definition and the mathematical interpretation. Secondly, a general overview of the history of the load flow calculations is presented. Then, we introduce the problem of ill-conditioning, both its mathematical meaning and features causing ill-conditioning in a power system. There follows the objective of this research work and an outline of the thesis organization. At the end of this chapter, we display the original contributions of this study.

Chapter II — Review of Load Flow Calculation Methods

This chapter presents a review and analysis of the most commonly used load flow calculation methods. Four main groups of load flow calculation methods are discussed. The state of the art on solving the ill-conditioned systems load flow problem is also discussed. Attention is paid especially to the basic formulations, characteristics, and limitations of each different approach.

Chapter III — Solution of the Ill-Conditioned Load Flow Problem by the Tensor Method

In the first part of this chapter, the mathematical foundations of the tensor method by Schnabel & Frank are presented. They include an introduction on solving ill-conditioned nonlinear equations, a derivation of the formulation of a simple quadratic model (tensor model), and a solution algorithm to solve the tensor model efficiently and stably In the second part of this chapter, applications of the tensor method to the load flow problem are carried out. Again, we present a brief introduction to reinforce the presentation of the problem to be solved. Then, the tensor model formulation of the load flow problem is performed and explained. The calculation of second order terms in the tensor model is conceptually new, and unknown in the previous Second Order load flow methods. Following the formulation, a solution algorithm is proposed and detailed program implementation steps are also given.

Chapter IV — Numerical Simulations

This chapter documents and analyzes the numerical results obtained from our TLF program. Tests were carried out on four ill-conditioned systems and three well-conditioned systems, ranging in size from 5 to 43 buses. In the first four sections, the numerical results for four ill-conditioned systems are presented seperately In each section, the load flow solutions, the convergence characteristics, and the condition numbers of each test system are presented and analyzed. Then, the numerical results obtained for three well-conditioned systems are presented altoghther. All the simulation results are analyzed and special features of the program on the hard-to-solve problems are discussed. This chapter closes with a discussion on the general performance of the tensor load flow method and comparisons with standard load flow methods.

Chapter V — Conclusions and Recommendations

In this short chapter, first, general conclusions of this research work are drawn. Then, recommendations for future research are suggested.

1.5 Claim of Originality

To the best of the author's knowledge, the application of the tensor method to solve the ill-conditioned load flow problem is an original contribution. By applying the tensor method, a new quadratic load flow model is formulated using information from the previous past iterates. A new solution algorithm to solve the tensor model is also developed It solves the ill-conditioned load flow problem by first solving a small quadratic system of equations and then a large linear system of equations. In this, it is quite distinct from the previous Second Order load flow methods. A program which implements the idea of the tensor algorithm is developed. The simulation results show the superior numerical stability and convergence characteristics on both well- and ill-conditioned power systems.

Chapter II

Review of Load Flow Calculation Methods

2.1 Introduction

In this chapter we review the most commonly used load flow calculation methods. Through a literature review, we briefly outline various load flow formulations, summarize their characteristics, and point out the limitations of each solution technique.

Four main groups of load flow calculation methods are discussed in section 2.2 to section 2.5 respectively. They are the Gauss-Seidel method, the Newton-Raphson method, the Fast Decoupled Load Flow method, and the Second Order Load Flow method. Following these four main groups of methods, we present some new load flow methods recently proposed for solving the ill-conditioned power system load flow problem. This chapter closes with a summary of the general behavior and comparison of these methods

We presume the reader is already familiar with the basic load flow equations in this chapter. To concentrate on the main properties of these solution methods, all the equations shown in this chapter are given without derivation, but the corresponding reference sources are given. The reader not familiar with the basic load flow equations can refer to section 3.2.2.1.

2.2 Gauss-Seidel Method

2.2.1 Basic equations

The Gauss-Seidel load flow calculation method is based on the iterative solution of the nonlinear equation (2.1) for n bus voltages, using a relaxation algorithm [Glimn & Stagg 1957].

$$\bar{V}_i = \frac{1}{\bar{Y}_{ii}} \left\{ \frac{P_i^{sp} - jQ_i^{sp}}{\bar{V}_i^*} - \sum_{j \in i} \bar{Y}_{ij} \bar{V}_j \right\} \tag{2.1}$$

$$i = 1, \cdots, n.$$

where the notation is provided in appendix 2.1.

Successive displacements are used in the Gauss-Seidel load flow method, which improve the accuracy of the bus voltages successively starting from an initial guess. The right hand side of the equation (2.1) is evaluated using the most recent updates of the voltages at the rest of the buses. For PV type buses, Q_i^{sp} is substituted by its calculated value, and the magnitude of the updated voltage is corrected according to the specified bus voltage.

The above set of equations does not include the slack bus for the reason discussed in Chapter I. Before the solution is started, the buses have to be ordered and the bus admittance matrix has to be determined. Next, a set of initial values for the complex bus voltages has to be assigned. In practical systems, the voltage magnitudes do not vary widely. The initial value of $1\angle 0$ can be assigned. Then, one can start obtaining the solution of Eq.(2.1). If the ith bus is a load bus, its loadings P_i and Q_i are known and hence a new value of complex voltage can be determined which may be denoted by $V_i^{(1)}$, where superscript $^{(1)}$ indicates the first cycle of calculation, i.e. first iteration. Obviously, the values of $V_i^{(1)}$ have been improved compared to their initial values $V_i^{(0)}$. Feeding this improved set of values in Eq.(2.1), one can determine another set of more improved values. This iterative process continues until the prescribed mismatch tolerance are met.

2.2.2 Characteristics

The Gauss-Seidel method converges slowly, mostly because of the loose mathematical coupling between the buses. At each iteration cycle, an improvement in each bus voltage can only effect the voltage improvements of the buses directly connected to it. Acceleration techniques are invariably used in practice to speed up the convergence

Computationally, the salient feature of this method is that the number of elements in the summation term in equation (2 1) is small. The off-diagonal elements of the admittance matrix, for use in the summation, are therefore stored and addressed compactly, often taking advantage of Y-matrix symmetry. Both the storage requirements and the computation per iteration are then small, and roughly proportional to the number of buses n. Since the number of iterations for a large well-conditioned system is of order n, the total iterative computing time varies approximately with n^2. As the size of the problem to be solved increases, the Gauss-Seidel method becomes less and less competitive with newer methods. However, its storage requirements are very low. The computation time per iteration is small, and efficient coding is very easily written. Therefore, it is often used to initialize the starting values of the Newton-Raphson method and Second Order Load Flow method. As a final point, an advantage of the GS method which is only starting to be recognized is that it is in a form easily amenable to parallel processing.

2.2.3 Limitations

Although the Gauss-Seidel method can be easily programmed and does not require a large number of computer storage, it has several limitations, some of which have already been mentioned:

1. needs many iterations to converge,
2. total computation times are long,
3. convergence characteristics are poor compared to newer methods,
4. unable to solve ill-conditioned systems.

In solving well-conditioned systems, the method converges and the number of iterations required depends upon the size of the system: the larger the system, the greater the number of iterations [Stott 1974]. The choice of the slack bus also affects the number of iterations.

2.3 Newton-Raphson Method

2.3.1 Basic equations

The general Newton-Raphson method is an iterative algorithm for solving a set of simultaneous nonlinear equations in an equal number of unknowns $F(x)=0$. At a given iteration point, each function $f_i(x)$ is approximated by its tangent hyperplane. Equation (2 2) is its basic form in the load flow application [Tinney & Hart 1967].

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} H & N \\ J & L \end{bmatrix} \begin{bmatrix} \Delta\delta \\ \Delta V \end{bmatrix} \qquad (2.2)$$

where the notation is provided in appendix 2.2.

This is the most widely used of all formulations, in which the Jacobian matrix equation is written for convenience of presentation in the partitioned form. Slack bus mismatches and voltage corrections are not included in (2.2), and likewise ΔQ_i and ΔV_i for each PV bus are absent. The elements of the submatrices H, N, J, L are given in section 3.2.2.2.

Like the GS method, the iterative process is initiated by assuming a set of voltage magnitudes and phase angles Coefficients H, N, J and L are then evaluated using equations given in Section 3.2 2 2 These coefficients are arranged in the form of a matrix (Jacobian) which is then augmented by including a column consisting of the residuals ΔP_i and ΔQ_i. The elimination process triangularises the augmented Jacobian matrix. The corrections $\Delta\delta_i$ and ΔV_i are then successively evaluated through back substitution. These corrections when added to the previous values of the variables generate new values of these variables. One iteration is complete. The next iteration starts with the calculation of P_i and Q_i of buses using improved values of the state variables. Residuals ΔP_i and ΔQ_i are evaluated and if these are found less than the mismatch tolerance, the process is terminated. Otherwise, the procedure of recalculation of the Jacobian matrix elements, triangularisation of the augmented matrix and reevaluation of the correction vectors $\Delta\delta$ and ΔV is repeated.

This formulation can be improved by a minor modification, which very often reduces the number of iterations by one, and can avoid divergence in some extreme case [Stott 1974]. Noting that the performance of the Newton-Raphson method is closely associated with the degree of problem nonlinearity, the best left hand defining functions are the most linear ones. It is therefore preferable to use a problem defining function $\Delta Q/V$ on the left hand side of equation (2.2) in place of ΔQ. Dividing ΔP by V has a very small effect, since the active power component of the problem is not strongly coupled with voltage magnitudes.

Both polar and rectangular version formulations can be implemented, with the rectangular version slightly less reliable and rapid in convergence than the polar version [Stott 1974].

2.3.2 Enhancements

A number of schemes are available for attempting to improve the performance of Newton's method [Dommel et.al. 1970]. One of the simplest of these is to impose limits on the permissible size of the voltage corrections at each iteration, thereby helping to negotiate humps in the defining functions. A better approach is to backtrack as soon as divergence is seen to have started, and then apply small limits on future state corrections.

With its quadratic convergence, Newton's method is most advantageous when it is fed good initial voltage estimates. Some programs perform one or two GS iterations before the Newton process. This is beneficial provided that the relatively weak GS method does not diverge when faced with a difficult problem. A most rapid and reliable Newton program can be created by calculating good initial angular estimates using the DC load flow and also good voltage magnitude estimates by a similar technique [Stott 1971].

Iteration time can be saved by using the same triangulated Jacobian matrix for two or more iterations. However, in this way, it is also necessary to save the lower triangle for repetition of the forward solution [Tinney & Hart 1967]. The algorithm requires about 40% more storage and loses some reliability of convergence

2.3.3 Characteristics

The Newton-Raphson method's quadratic convergence makes it very fast, and the process homes in very rapidly onto the solution point when it is close Its performance is sensitive to the behavior of the load flow function $F(x)$, i.e. the more linear it is, the more rapidly and reliably the Newton method converges

Although efficiency in programming techniques is important in all load flow methods for fast excution and storage economy, it is the cornerstone of methods such as Newton's. It must implement ordered elimination and sparse programming techniques for solving the large sparse matrix equations [Tinney & Walker 1967] [Tinney & Hart 1967].

If these programming requirements are fully satisfied, then the computing time per iteration of Newton's method rises little more than linearly with the number of buses in the system, on average. Since the number of iterations is almost size invariant, the superiority of Newton's method speedwise over the GS method increases rapidly as the size of the system to be solved increases.

With large modern computers, the extra storage compared with GS method does not prevent very large systems from being solved in core.

2.3.4 Limitations

Although the Newton-Raphson method is far superior to the GS method regarding its convergence characteristics and total computation time required, it has several limitations.

1. a large quantity of data must be handled simultaneously due to the linear transformation,
2. needs optimally ordered factorization and sparse programming technique,
3 sensitive to the nonlinearity of the load flow functions,
4. unable to solve ill-conditioned systems, like the 11 and the 43 bus systems.

Any production program using the Newton-Raphson method should consider the above facts and include such features as compact storage and optimal ordering schemes for system buses. A computer program based on the Newton-Raphson method is, therefore, in general more complicated than that based on the GS method.

2.4 Fast Decoupled Load Flow Method

2.4.1 Basic equations

An inherent characteristic of any practical electric power transmission system operating in the steady state is the strong interdependence between active powers and bus voltage angles, and between reactive powers and voltage magnitudes. Applied numerical methods are generally at their most efficient when they are able to take advantage of the physical properties of the system being solved.

Equation (2.3a) and (2.3b) are derived from equation (2 2) by taking advantage of these physical properties [Stott & Alsac 1974].

$$\Delta P/V = B'\Delta\delta \tag{2.3a}$$

$$\Delta Q/V = B''\Delta V \tag{2.3b}$$

where

$$B'_{ii} = \sum_{j \in i} 1/X_{ij} \ ,$$

$$B'_{ij} = -1/X_{ij} \quad \text{and} \quad B''_{ij} = -B_{ij} \qquad (i \neq j).$$

Matrices B' and B'' represent constant approximations to the slopes of the tangent hyperplanes of the functions $\Delta P/V$ and $\Delta Q/V$ respectively. Network elements that primarily affect the Q–V problem (e.g., shunt susceptances and transformer off-nominal taps) are not presented in B'. Similarly, phase shifts are not presented in B''. Consequently, both B' and B'' are always symmetrical, and their constant sparse upper triangular factors are calculated and stored once only at the beginning of the solution. To solve (2.3a) and (2.3b), forward and backward substitutions are performed using these factors.

The algorithm is to conduct each iteration cycle by first solving (2.3a) for $\Delta\delta$, and use the updated δ in constructing and then solving (2.3b) for ΔV. Each of these construction/solution cycles are performed alternately in the same storage area.

2.4.2 Enhancements

Because of its simplicity, speed, and relatively low storage requirements, the Fast Decoupled load flow method is now being widely used in the electric utilities. Over the years of experience, a number of new schemes have been developed.

To speed up the Fast Decoupled load flow method, a rectangular version of the program was developed [Masiello & Wollenberg 1975] [Babie 1983]. Also, in modern system reliability studies, the use of the FDLF method requires the frequent updating of the two Jacobian-like matrices. [Behnam-Guilnani 1987] introduced the hybrid version of FDLF method. In this new model, the first equation in the FDLF method is unchanged For the second equation, the decoupled nodal iterative model derived from the GS method is used. The hybrid model has the positive attributes of the FDLF method (fast, simple, and reliable) with the added advantage that it resolves the matrix updating problem. The study results show that the hybrid method is as much as 50% faster than the FDLF method even if B'' is not updated, however Stott & Alsac report the hybrid version's practical usefulness seems to hinge mainly on whether each GS reactive iteration has a sufficiently competitive convergence rate [Stott & Alsac 1988]. i.e., in the absence of regulated buses, the performance of the hybrid method will not be so great [Alvarado 1988]

To reduce the storage requirements, [Keyhani 1985] proposed a new version of FDLF method. The matrix B'' is set equal to the matrix B', and in the computation of B' shunt reactances and series resistances are neglected. Since shunt reactances are effective in MVAR flows, ΔQ is computed with all shunt capacitances and line chargings considered as reactive power generations at each bus. The algorithm requires 50% less memory for matrix inversion than the original FDLF method, and has approximately the same convergence characteristics

Practical experience confirms that abnormally high R/X ratios, particularly on heavily loaded branches, can slow convergence down, sometimes considerably [Wu 1977]. The individual convergence rates of the algorithms are determined by how well B' and B'' approximate the slopes of functions $\Delta P/V$ and $\Delta Q/V$ respectively These approximations are excellent around the $\delta=0$, $V=1$ points on the functions At high system loading (large δ's or poor V's), the approximations' deterioration is progressive as the R/X ratio of any branch terminating on the bus increases. An effective method to solve this kind of problem is to place fictitious nodes in the middle of the lines having high R/X ratios [Ramarao 1977], which helps to reduce the line R/X ratio so that the convergence performance is better.

2.4.3 Characteristics

The Fast Decoupled load flow method is now the fastest load flow calculation method. Computation time per iteration is roughly 1/5 that of Newton-Raphson method, and 3/2 that of the Gauss-Seidel method. For very large systems (5000 buses and above), the total computation time is 3 or 4 times smaller than that of the NR method [Sasson et.al. 1975].

The storage requirements of the Fast Decoupled load flow method are about 40% less than those of Newton's method. This saving is reduced somewhat if the sine and cosine terms are stored.

The method converges very reliably, usually in 2 to 5 iterations for practical accuracy on large systems. Using a standard triangulation package, programming is easier than the Newton's method in which the compact Jacobian matrix has to be computed at each iteration.

2.4.4 Limitations

Here are the limitations associated with the Fast Decoupled load flow method:

1. difficult to solve systems having lines with high R/X ratios,
2. when a system is heavily loaded, convergence requires many

iterations,

3. unable to solve ill-conditioned systems.

2.5 Second Order Load Flow Method

2.5.1 Basic equations

Second Order load flow methods (SOLF) were first proposed by [Sachdev & Medicherla 1977], in a polar form They are based on the Taylor series expansion of the load flow functions up to their third term. Then, [Iwamoto & Tamura 1978] presented an excellent paper, formulating their second order load flow model in rectangular coordinates. This was followed by [Rao 1978] [El-Hawary & Wellon 1982] and [Nagendra Rao et.al. 1982].

The load flow equations are a set of quadratic algebraic equations when expressed in rectangular coordinates That is, the equations are expressed completely and exactly using the first three terms in the Taylor series expansion.

The basic Second Order load flow equation is as follows:

$$\begin{bmatrix} \Delta P - SP \\ \Delta Q - SQ \end{bmatrix} = \begin{bmatrix} J \end{bmatrix} \begin{bmatrix} \Delta x \end{bmatrix} \tag{2.4}$$

where SP and SQ are the second order term vectors.

In Iwamoto and Tamura's formulation, they discovered an efficient way to formulate the second order term in their model. The evaluation of the second order term uses the same load flow functions but with the variable Δx (state corrections), where Δx is obtained from the previous iteration. At each iteration, the value of power mismatch term is a function of Δx, but the Jacobian matrix J is always kept the same. This makes the algorithm very fast, and its speed is comparable to the FDLF method.

Later, in Nagendra Rao's formulation, the model is developed by first considering a system with only PQ buses and then extended to accommodate PV buses. The computation is simplified to some extent by introducing two features:

1. All shunt connections (line charging capacitance, shunt reactors, shunt capacitors, shunt branches of equivalant π representation of off-nominial ratio transformers etc.) are taken into account by treating them as loads of constant impedance and hence they are not included in the Y-matrix.
2. The initial guesses for the voltages at all buses are assumed to be equal to the slack bus voltage.

The symmetry of the Jacobian matrix is achieved when all the shunt elements are treated as loads. This is a well-known approach especially in Z-matrix based load flow methods. The Jacobian is to be determined only once at the starting point for the first iteration.

At PV buses in the system, the voltage magnitude will be specified instead of the reactive power. In the evaluation of the second order terms, the branch admittances of lines connected to PV buses (in order to calculate reactive power mismatch ΔQ) only will have to be retained in the core during the iteration process. This results in a reduction in the memory requirement of the method as compared with Iwamoto's method, where the full branch admittance list must be stored in order to calculate the second order terms.

2.5.2 Characteristics

Here are the main characteristics of the two second order methods described above:

Iwamoto & Tamura's version:

1. faster than NR method, similar to FDLF method.
2. iteration counts for convergence are not affected by the system size.
3. memory requirement slightly higher than NR method.
4. did not converge to solutions for the cases in which NR method could not give the solutions.
5. initial estimates are quite important.

Nagendra Rao's version:

1. claimed faster than previous SOLF method.
2. memory requirement is comparable to FDLF method
3. shows better reliabity on high R/X ratio problem.
4. fairly complicated in its formulation.

Although there are a few other Second Order load flow methods [Sachdev & Medicherla 1977], [Rao 1978], [El-Hawary & Wellon 1982], they are not competitive compared to the methods proposed above, taking into account the mathmetical simplicity and computational efficiency. Actually, the method developed by Iwamoto and Tamura is quite good and is now being used by the power industry in Japan [Iwamoto 1989].

2.5.3 Limitations

Because the Second Order load flow methods uses the first three terms of the Taylor series expansion, generally they are mathematiclly more complicated than the Newton's method.

By reviewing all these second order load flow methods, another important point is found: that all algorithms solve equation (2.4). The second order terms are estimated from previous iteration values and substracted from the power mismatches in the left hand side of the equation (2.4). The right hand side of the equation (2.4) remains unchanged from the standard Newton-Raphson method. If the Jacobian matrix is ill-conditioned, these methods all have difficulties to converge to a solution. This enlightened fact provided us with a good reason for seeking a new second order algorithm to solve the ill-conditioned systems.

2.6 Methods for Ill-Conditioned Systems

Over the years, efforts have concentrated on speeding up the arithmetic of the load flow calculation algorithms, taking advantage of sparsity, decoupling, etc. The problem of solving ill-conditioned systems and determining the existence of load flow solutions have been treated very superficially, on the assumption that since NR and FDLF methods have proved so successful for "real life" problems, no rigorous validation is necessary. The assumption relies on the wishful thinking premise, that systems such as the 11 and 43 bus considered in this thesis are in some sense artificial and will never occur in practice. Even if this premise is true, the art of load flow solving, in order to attain the status of science, must address these fundamental questions.

In 1981, Iwamoto and Tamura first proposed a method to tackle the ill-conditioned systems load flow calculation problem [Iwamoto & Tamura 1981]. The model is based on their second order model [Iwamoto & Tamura 1978], incoroporating the nonlinear programming technique suggested by [Sasson 1969] and the idea of obtaining the optimal multiplier [Wallach & Konrad 1979]. Although the authors claimed theoretical proof of the nondivergence of their solution, the simulation results do show divergence phenomena in some cases.

Following the above approach, Brown's method was applied to solve ill-conditioned load flow problems [Tripathy et al. 1982] Brown's method is particularly effective for solving ill-conditioned nonlinear algebraic equations. It is a variation of Newton's method incorporating Gaussian elimination in such a way that the most recent information is always used at each step of the algorithm, similar to what is done in the Gauss-Seidel process. This contrasts sharply with Newton's method in which all equations are treated simultaneously Perhaps due to this reason, the ill-conditioning in the Jacobian matrix is avoided in the solution process.

Simulation results provided in [Tripathy et.al. 1982] look good. The method converges in fewer iterations for both ill- and well-conditioned systems. Computing time per iteration is 15% more t' the Newton-Raphson method, so that the total computation time is said to be comparable to the Newton-Raphson method. The storage requirement is slightly more (10%) than that of Newton-Raphson method.

2.7 Summary

This chapter has presented the most important solution techniques for the power system load flow calculations through a review of the literature. Attention is paid especially to the basic equations of different methods, their salient characteristics, and their limitations. To introduce the subject of this thesis, the state of the art in solving the ill-conditioned load flow problem is also presented.

It is felt that among all these methods, because of its simplicity, speed, and reduced storage requirement, the Fast Decoupled load flow method is superior to any other method for well-behaved systems. This has been proved, in fact, by the popularity and the wide-ranging implementations of the Fast Decoupled load flow method in the electric power utilities in recent years.

Regarding the ill-conditioned system load flow problem, so far no practical method has emerged. Future research and development is needed to give the answer to this problem. This thesis proposes a step in that direction.

Chapter III

Solution of the Ill-Conditioned Load Flow Problem by the Tensor Method

A new method, namely the tensor method, is introduced in this chapter for solving load flow problems. Tensor methods are general purpose methods intended especially for ill-conditioned nonlinear systems, such as for power systems where the load flow Jacobian matrix is singular or ill-conditioned at the solution.

Over the years, tensor theory has expanded greatly. During that time, Kron's approach [Kron 1959], much heralded in electrical engineering some three decades ago, has been overtaken by a vaster, more general theory. Hence, readers should not expect to see Kron's approach in this thesis.

3.1 Tensor Method for Nonlinear Equations

3.1.1 Introduction

We present a new class of methods, tensor methods, for solving the nonlinear problem

$$\text{given } F{:}R^n \rightarrow R^n, \text{ find } x_* \in R^n \text{ such that } F(x_*)=0, \qquad (3.1)$$

where it assumed that $F(x)$ is at least once continuously differentiable. The novel feature of these methods is that they base each iteration on a quadratic model of $F(x)$ whose second order term has a special, restricted form. Tensor methods are especially intended to improve upon the performance of standard methods on problems where the Jacobian matrix of F at x_*, $F'(x_*) \in R^{n\times n}$, is singular or ill-conditioned.

Standard methods for solving (3.1) base each iteration on a linear model $M(x)$ of $F(x)$ around the current iterate $x_c \in R^n$,

$$M(x_c + d) = F(x_c) + J_c d \tag{3.2}$$

where $d \in R^n$, $J_c \in R^{n\times n}$.

When the analytic Jacobian is available, the linear model (3.2) becomes

$$M(x_c + d) = F(x_c) + F'(x_c)d \tag{3.3}$$

The standard method for nonlinear equations, Newton's method, is defined when $F'(x_c)$ is nonsingular, and consists of setting the next iterate x_+ to the root of (3.3),

$$x_+ = x_c - F'(x_c)^{-1}F(x_c) \tag{3.4}$$

The distinguishing feature of Newton's method is that if $F'(x_c)$ is continuous in a neighborhood containing the root x_* and $F'(x_*)$ is nonsingular, then the sequence of iterates produced by (3.4) converges locally and quadratically to x_c. In practice, local quadratic convergence means eventual fast convergence.

Newton's method is not usually quickly locally convergent, however, if $F'(x_*)$ is singular. In practice, Newton's method usually exhibits local linear convergence with constant equal to one half on singular problems, much slower covergence then one would like [Decker & Kelley 1980].

The other well-known disadvantage of Newton's method is that it may not converge to any root x_* if it is started too far from any root. The main remedies used in practice are augmenting (3.4) by line search or trust region algorithms [Dennis & Schnabel 1983]. For load flow studies, this was suggested as early as 1975 by [Gross & Luini 1975].

Tensor methods are based on expanding the linear model (3.3) of $F(x)$ around x_* to the quadratic model

$$M_t(x_c + d) = F(x_c) + F'(x_c)d + \frac{1}{2} T_c dd \tag{3.5}$$

where $T_c \in R^{n x n x n}$. The three-dimensional object T_c is referred to as a tensor, hence we call (3.5) a tensor model, and solution methods based upon (3.5) tensor methods. We define the notation $T_c dd$ used in (3.5) before proceeding.

<u>DEFINITION 3.1</u> Let $T \in R^{n x n x n}$. Then T is composed of n horizontal faces $H_i \in R^{n x n}$, $i=1,\cdots,n$, where $H_i[j,k]=T[i,j,k]$ For $v,w \in R^n$, $Tvw \in R^n$ with

$$Tvw[i] = v^t H_i w = \sum_{j=1}^{n} \sum_{k=1}^{n} T[i,j,k]v[j]w[k]$$

Note that $M_t(x_c+d)$ is simply the n-vector of n quadratic models of the component functions of $F(x)$,

$$(M_t(x_c+d))[i] = f_i + g_i^t\, d + \frac{1}{2}\, d^t H_i d$$
$$i = 1, \cdots, n$$

where

$f_i = F(x_c)[i]$,

$g_i^t =$ row i of $F'(x_c)$,

H_i is the Hessian matrix of the ith component function of $F(x)$.

The obvious choice of T_c in (3.5) is the matrix $F''(x_c)$ of second partial derivatives of F at x_c; this makes (3.5) the first three terms of the Taylor series expansion of F around x_c. Several disadvantages, however, make (3.5) with $T_c = F''(x_c)$ unacceptable for algorithmic use. They include.

(1) The n^3 second partial derivatives of F at x_c would have to be computed at each iteration.

(2) The model would take more than $n^3/2$ locations to store.

(3) To find a root of the model, at each iteration one would have to solve a system of n quadratic equations in n unknowns, which is often as difficult as solving the original problem $F(x)=0$.

To use a model of form (3.5) and avoid these disadvantages, the tensor method proposed by [Schnabel and Frank 1984] uses a very restricted form of T_c. In particular, the tensor model requires no additional derivative or function information; the additional costs of forming and solving the tensor model are small compared to the $O(n^3)$ arithmetic cost per iteration of standard methods; and the additional storage required for the tensor model is small compared to the n^2 storage required for the Jacobian.

The remainder of section 3.1 summarizes the work of Schnabel and Frank in applying the tensor method to solve nonlinear equations. This will be followed, in the next section, by our formulation of the tensor method applied to the load flow problem.

3.1.2 Forming the tensor model

We now show how to select the tensor term $T_c \in R^{n \times n \times n}$ in the model (3.5). The choice of T_c will cause the second order term $T_c dd$ in (3.5) to have a simple, useful form.

It has already been stated that T_c will not contain actual second derivative information. Another way to form the second order term in (3.5) is through interpolation. In the tensor method, some set of p not necessarily consecutive past iterates $x_{-1}, \cdots, x_{-p}$ will be selected, and the model (3.5) will be required to interpolate the function values $F(x_{-k})$ at these points. That is, the model should satisfy

$$F(x_{-k}) = F(x_c) + F'(x_c)s_k + \frac{1}{2} T_c s_k s_k, \quad k = 1, \cdots, p \qquad (3.6a)$$

where

$$s_k = x_{-k} - x_c, \qquad k = 1, \cdots, p \qquad (3.6b)$$

First we describe how the past points $x_{-1}, \cdots, x_{-p}$ are selected. Then we show how to choose T_c to satisfy (3.6).

For (3.6) to always be consistent, the set of directions $\{s_k\}$ from x_c to the selected past points x_{-k} must be linearly independent. In that sense, each direction s_k should make a angle of at least θ degrees with the linear subspace spanned by the other directions; values of θ between 20° and 45° have proven appropriate in practice. At each iteration, therefore, we choose the past points $\{x_{-k}\}$ that we included in (3.6) by the following procedure. We consider the past iterates in order, starting with the most recent. We always select the most recent iterate, and then test each preceding past iterate.

We also set a practical upper bound p on the number of past function values interpolated by the model at each iteration.

$$p \leq \sqrt{n} \tag{3.7}$$

The bound is crucial to the efficiency in storage and arithmetic operations of the tensor method.

Now we discuss how we choose T_c to satisfy (3.6). It is convenient to rewrite (3.6) as

$$T_c s_k s_k = z_k, \qquad k = 1, \cdots, p, \tag{3.8a}$$

where

$$z_k = 2\ (F(x_{-k}) - F(x_c) - F'(x_c)s_k) \qquad z_k \in R^n \tag{3.8b}$$

This is a set of $np \leq n^{1.5}$ linear equations in the n^3 unknowns $T_c[i,j,k]$, $1 \leq i,j,k \leq n$. Since (3.8) is underdetermined, we follow the standard and successful practice in secant methods for nonlinear equations and optimization, and choose T_c that satisfies

$$\underset{T_c \in R^{n\times n\times n}}{\text{minimize}} \; \| T_c \|_f \tag{3.9}$$

$$\text{subject to } T_c s_k s_k = z_k, \quad k = 1, \cdots, p,$$

where $\| T_c \|_f$, the Frobenius norm of T_c, is defined by

$$\| T_c \|_f^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} (T_c[i,j,k])^2$$

The solution to (3.9) is given by Theorem 3.1 of [Schnabel & Frank 1984]. First we define a rank one tensor.

DEFINITION 3.2 Let u, v, $w \in R^n$. The tensor $T \in R^{n\times n\times n}$ for which $T[i,j,k] = u[i]\; v[j]\; w[k]$, $1 \le i,j,k \le n$ is called a rank one tensor and denoted $T = uvw$.

THEOREM 3.1 Let $p \le n$, let $s_k \in R^n$, $k = 1, \cdots, p$ with $\{s_k\}$ linearly independent, and let $z_k \in R^n$, $k = 1, \cdots, p$. Define $M \in R^{p\times p}$ by $M[i,j] = (s_i^t s_j)^2$, $1 \le i,j \le p$, $Z \in R^{n\times p}$ by column k of $Z = z_k$, $k = 1, \cdots, p$. Then M is positive definite, and the solution to (3 9) is

$$T_c = \sum_{k=1}^{n} a_k s_k s_k \tag{3.10}$$

where a_k is the kth column of $A \in R^{n\times p}$. A is defined by $A = Z M^{-1}$.

Substituting (3.10) into the tensor model (3.5) gives,

$$M_t(x_c + d) = F(x_c) + F'(x_c)d + \frac{1}{2}\sum_{k=1}^{p} a_k(d^t s_k)^2 \qquad (3.11)$$

The simple form of the second order term in (3.11) is the key to being able to efficiently form, store, and solve the tensor model. Since $p \le \sqrt{n}$, the additional storage required by (3.11) is $4p$ n-vectors, for $\{a_k\}$, $\{s_k\}$, $\{x_{-k}\}$, and $\{F(x_{-k})\}$. In the next section, we will see that the extra cost to solve the tensor modal also is quite small.

3.1.3 Solving the tensor model

In this section we show an efficient algorithm for finding a root of the tensor model derived in 3.1.2, that is,

$$\text{find} \quad d \in R^n \quad \text{such that}$$

$$M_t(x_c + d) = F(x_c) + F'(x_c)d + \frac{1}{2}\sum_{k=1}^{p} a_k(d^t s_k)^2 = 0 \qquad (3.12)$$

The solution of (3.12) can be reduced to the solution of a system of p quadratic equations in p unknowns, plus the solution of a system of $n-p$ linear equations in $n-p$ unknowns.

The basic idea of the algorithm is that (3.12) is linear on an $(n-p)$-dimensional subspace. (3.12) really only should be quadratic in p variables and linear in the other $n-p$ This is accomplished in steps 1 and 2 of the upcoming Algorithm 3.1 by making a linear transformation of the variable space; for this an orthogonal transformation is used. Then a second linear transformation of the equations, steps 3 and 4 of Algorithm 3.1, is used to eliminate the $n-p$ transformed linear variables from p of the nonlinear equations. The result is a system of p quadratic equations in p unknowns (3.15b), that is solved in step 5 of Algorithm 3.1, and a system of $n-p$ equations (3.15a) that are linear in the remaining $n-p$ unknowns that can be solved once the system of quadratics is solved.

Two notations are introduced: Given $v \in R^m$, $(v)^2$ denotes the vector $w \in R^m$ for which $w[i] = v[i]^2$, $i = 1, \cdots, m$. Define $S \in R^{n \times p}$ by column k of $S = s_k$. This allows us to denote the second order term of our tensor model by $\frac{1}{2} A \, (S^t d)^2$.

<u>ALGORITHM 3.1</u> Let $p \leq n$, $F \in R^n$, $J \in R^{n \times n}$, A, $S \in R^{n \times p}$, S having full column rank.

First transformation: Steps 1-2 transform the system of n equations in n unknowns

$$F + Jd + \frac{1}{2} A \, (S^t d)^2 = 0 \qquad (3.13)$$

to the system of n equations in the n unknowns $\hat{d}_1 \in R^{n-p}$, and $\hat{d}_2 \in R^p$,

$$F + \hat{J}_1 \hat{d}_1 + \hat{J}_2 \hat{d}_2 + \frac{1}{2} A \, (\hat{S}_2^{\,t} \, \hat{d}_2)^2 = 0 \qquad (3.14)$$

This eliminates the nonlinear terms in $\hat{d}_1$.

Step 1. Find an orthogonal $Q \in R^{nxn}$ such that $Q^t S = \hat{S}$,

where

$$\hat{S} \in R^{nxp} = \begin{array}{rcl} & p & \\ & \begin{bmatrix} 0 \\ \hline \begin{matrix} 0 & \\ \ast & \ast \end{matrix} \end{bmatrix} & \begin{array}{l} n-p \\ \\ p \end{array} \\ \hat{S}_2 \left\{ \right. & & \end{array}$$

and $\hat{S}_2$ has the lower triangular shape shown.

Step 2. Calculate $\hat{J} = JQ$,

- Let $\hat{J}_1 \in R^{nx(n-p)}$ and $\hat{J}_2 \in R^{nxp}$ denote the first $n-p$ and last p columns of $\hat{J}$, respectively.
- Define $\hat{d} = Q^t d$,
- Let $\hat{d}_1 \in R^{n-p}$ and $\hat{d}_2 \in R^p$ denote the first $n-p$ and last p components of $\hat{d}$, respectively.

Second transformation: Steps 3-4 transform the system of equations (3.14) to

$$\begin{array}{c} n-p \\ p \end{array} \begin{bmatrix} \tilde{F}_1 \\ \tilde{F}_2 \end{bmatrix} + \begin{array}{c} \begin{array}{cc} n-p & p \end{array} \\ \begin{bmatrix} \tilde{J}_1 & \tilde{J}_2 \\ 0 & \tilde{J}_3 \end{bmatrix} \end{array} \begin{bmatrix} \tilde{d}_1 \\ \hat{d}_2 \end{bmatrix} + \frac{1}{2} \begin{array}{c} p \\ \begin{bmatrix} \tilde{A}_1 \\ \tilde{A}_2 \end{bmatrix} \end{array} (\hat{S}_2{}^t \, \hat{d}_2)^2 = 0 \qquad (3.15)$$

a "large" system of $n-p$ equations in n unknowns

$$\tilde{F}_1 + \tilde{J}_1\tilde{d}_1 + \tilde{J}_2\hat{d}_2 + \frac{1}{2}\tilde{A}_1(\hat{S}_2^{\,t}\,\hat{d}_2)^2 = 0 \tag{3.15a}$$

and a "small" system of p equations in p unknowns

$$\tilde{F}_2 + \tilde{J}_3\hat{d}_2 + \frac{1}{2}\tilde{A}_2(\hat{S}_2^{\,t}\,\hat{d}_2)^2 = 0 \tag{3.15b}$$

This is accomplished by premultiplying (3.14) by the appropriate orthogonal matrix.

Step 3. Find an orthogonal $\hat{Q} \in R^{n\times n}$ and a permutation matrix $P \in R^{(n-p)\times(n-p)}$ such that

$$\hat{Q}\,\hat{J}_1\,P = \begin{array}{rcl} & n-p & \\ \tilde{J}_1\left\{\right. & \begin{array}{|c|}\hline \diagdown\ 0 \\ \hline 0 \\ \hline \end{array} & \begin{array}{l} n-p \\ p \end{array} \end{array}$$

$\tilde{J}_1$ is upper triangular with a nonzero diagonal.

Define $\tilde{d} = P_1^{\,t}\,\hat{d}_1$, $\tilde{d}_1 \in R^{n-p}$.

Step 4. Calculate

$$\hat{Q}\hat{J}_2 = \begin{array}{c} p \\ \begin{bmatrix} \tilde{J}_2 \\ \tilde{J}_3 \end{bmatrix} \end{array} \begin{array}{l} \\ n-p \\ p \end{array}$$

Similarly calculate $\tilde{A} = \hat{Q}A$,

- Let $\tilde{A}_1 \in R^{(n-p)\times p}$ and $\tilde{A}_2 \in R^{p\times p}$ denote the first $n-p$ and the last p rows of $\tilde{A}$, respectively;
- Calculate $\tilde{F} = \hat{Q}F$,
- Let $\tilde{F}_1 \in R^{n-p}$ and $\tilde{F}_2 \in R^{p}$ denote the first $n-p$ and last p components of $\tilde{F}$, respectively.

Step 5. Solve (3.15b) in the least squares sense

$$\underset{d_2 \in \mathrm{R}^p}{\text{minimize}} \left\| \tilde{F}_2 + \tilde{J}_3 \hat{d}_2 + \frac{1}{2} \tilde{A}_2 (\hat{S}_2^{\,t} \hat{d}_2)^2 \right\|_2 \qquad \cdots\cdots(3.16)$$

Step 6. Backsolve (3.15a) for $\tilde{d}_1$

$$\tilde{J}_1\tilde{d}_1 = -\tilde{F}_1 - \tilde{J}_2\hat{d}_2 - \frac{1}{2}\tilde{A}_1(\hat{S}_2^t\hat{d}_2)^2 \qquad \cdots\cdots(3.17)$$

Step 7. Calculate $\hat{d}_1 = P\tilde{d}_1$, $d = Q\hat{d}$.

The first virtue of Algorithm 3.1 is its efficiency. The dominant cost in Algorithm 3.1 is the QR factorization of $\hat{J}_1$ which requires about $2n^3/3 - n^2p + O(n^2)$ multiplications. The next largest cost is the $2n^2p + O(n^2)$ multiplications for the matrix multiplication JQ in step 2. All other portions of steps 1-7 require at most $O(n^2)$ multiplications.

The other virtue of Algorithm 3.1 is its numerical stability, even when the Jacobian J is singular or ill-conditioned. The whole point of the tensor algorithm when J is singular is that the possibly singular submatrix $\tilde{J}_3$ is used in the system of quadratic equations (3.15b), which also contain a portion of the second order infomation in the tensor model. This system is not necessarily ill-conditioned even if $\tilde{J}_3$ is; for example, one quadratic equation in one unknown with no linear term is usually not ill-conditioned.

3.2 Application of the Tensor Method to Load Flow Calculations

3.2 1 Introduction

Standard load flow algorithms using digital computers have been developed and have worked well for well-conditioned power systems (most systems). These include the Gauss-Seidel method, the Newton-Raphson method and its principle off-shoot, the Fast Decoupled load flow method.

A problem as yet to be adequately solved, and which has been tackled recently by many researchers [Iwamoto & Tamura 1981], [Tripathy et.al 1982], is how to solve the ill-conditioned power system load flow problem and determine the existance of solutions. Features which cause instability and divergence in the load flow calculations were listed in Chapter I. Mathematically, the nonlinear equations which described the system are such that small changes in the parameters will cause large changes in the solutions. So far, few methods have successfully solved the ill-conditioned load flow problem, while many proposals do not seem to provide much improvement. In order to incorporate more load flow information into the computation, the tensor method is applied to solve the ill-conditioned load flow problem.

3.2.2 Formulation of the load flow tensor model

3.2.2.1. The load flow function $F(x)$

As stated in chapter I, the load flow study determines the complex voltages at the system buses for a particular loading condition of the system. To do this with the help of a digital computer, a set of equations expressing the active and reactive powers at the buses in terms of complex voltages is needed. For a network having n nodes excluding ground, the following compact form of n equations can be written for current:

$$\bar{I}_i = \sum_{j \in i} \bar{Y}_{ij} \bar{V}_j, \quad i = 1, \cdots, n, \tag{3.18}$$

where

$\bar{I}_i$ = complex current entering in the bus i,

$\bar{V}_j$ = complex voltage to ground of the bus j,

$\bar{Y}_{ij}$ = complex admittance between buses i and j,
when $i = j$, Y_{ij} is the self admittance,
when $i \neq j$, Y_{ij} is the transfer admittance.

$j \in i$ signifies that bus j is connected to bus i, including the case $j = i$.

In a power system, however the complex power is the most important quantity. From circuit theory the complex power injection into a bus can be expressed as

$$P_i + jQ_i = \bar{V}_i \bar{I}_i^* \tag{3.19}$$

Superscript * in the above equation indicates conjugation. Substituting equation (3.18) into equation (3.19),

$$P_i + jQ_i = \bar{V}_i \sum_{j \in i} \bar{Y}_{ij}^* \bar{V}_j^* \quad i = 1, \cdots, n. \tag{3.20}$$

Using polar notation for $\bar{V}$, $\bar{V}_i = V_i \angle \delta_i$, and rectangular notation for $\bar{Y}$, $\bar{Y}_{ij} = G_{ij} + jB_{ij}$, equation (3.20) becomes:

$$P_i = V_i \sum_{j \in i} V_j \,(G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij}) \qquad (3.21a)$$

$$Q_i = V_i \sum_{j \in i} V_j \,(G_{ij} sin\delta_{ij} - B_{ij} cos\delta_{ij}) \qquad (3.21b)$$

where

$$i = 1, \cdots, n.$$

and

$$\delta_{ij} = \delta_i - \delta_j$$

This is a standard formulation of bus power injections versus the complex bus voltages.

In the load flow calculations, a number of quantities are specified. Typically·

- P_i is specified for all buses except one "slack" bus, $(n\text{-}1)$.
- Q_i is specified for all the load buses, $(n\text{-}pv\text{-}1)$.
- V_i is specified for all generation buses, which are often known as PV buses, and for a slack bus, (pv).
- δ_i is specified only for one reference bus.

The dimension of equation (3.21) now becomes $(2n\text{-}2\text{-}pv)$ and there are $(2n\text{-}2\text{-}pv)$ unknowns.

Load flow algorithms check the difference or mismatch between specified bus power values and calculated bus power values. Therefore, the following mismatch equations are used:

$$\Delta P_i = P_i^{sp} - V_i \sum_{j \in i} V_j (G_{ij} \cos\delta_{ij} + B_{ij} \sin\delta_{ij}) \qquad (3.22a)$$

$$\Delta Q_i = Q_i^{sp} - V_i \sum_{j \in i} V_j (G_{ij} \sin\delta_{ij} - B_{ij} \cos\delta_{ij}) \qquad (3.22b)$$

$$i = 1, \cdots, n\text{-}1.$$

This is a set of $(2n\text{-}2\text{-}pv)$ nonlinear equations. There are $(n\text{-}1)$ equations for bus real power mismatch ΔP_i and $(n\text{-}1\text{-}pv)$ equations for bus reactive power mismatch ΔQ_i.

From equation (3.22) the load flow calculation problem can be restated as this: For the certain specified loadings of P_i^{sp}, Q_i^{sp}, $i = 1, \cdots, n\text{-}1$, find the complex bus voltages V_i, $\angle\delta_i$, $i = 1, \cdots, n\text{-}1$, so that all the bus power mismatches ΔP_i, ΔQ_i, $i = 1, \cdots, n\text{-}1$, are within the required tolerance The reader should note that the subscript i for ΔQ_i and V_i may not be consecutive here.

3.2.2.2. The load flow Jacobian $J(x)$

The load flow Jacobian is the derivative matrix of the real and reactive power mismatches with respect to the voltage states:

$$\begin{bmatrix} \dfrac{\partial \Delta P}{\partial x^t} \\[2ex] \dfrac{\partial \Delta Q}{\partial x^t} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \Delta P}{\partial \delta^t} & \dfrac{\partial \Delta P}{\partial V^t} \\[2ex] \dfrac{\partial \Delta Q}{\partial \delta^t} & \dfrac{\partial \Delta Q}{\partial V^t} \end{bmatrix} \qquad (3.23)$$

while the elements in each submatrices are shown as follows:

$$H_{ij} = \frac{\partial \Delta P_i}{\partial \delta_j} = - V_i V_j (G_{ij} sin\delta_{ij} - B_{ij} cos\delta_{ij}) \qquad j \neq i$$

$$H_{ii} = \frac{\partial \Delta P_i}{\partial \delta_i} = V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} sin\delta_{ij} - B_{ij} cos\delta_{ij})$$

$$N_{ij} = \frac{\partial \Delta P_i}{\partial V_j} = - V_i (G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij}) \qquad j \neq i$$

$$N_{ii} = \frac{\partial \Delta P_i}{\partial V_i} = - \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij}) - 2V_i G_{ii}$$

$$J_{ij} = \frac{\partial \Delta Q_i}{\partial \delta_j} = V_i V_j (G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij}) \qquad j \neq i$$

$$J_{ii} = \frac{\partial \Delta Q_i}{\partial \delta_i} = - V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij})$$

$$L_{ij} = \frac{\partial \Delta Q_i}{\partial V_j} = - V_i (G_{ij} sin\delta_{ij} - B_{ij} cos\delta_{ij}) \qquad j \neq i$$

$$L_{ii} = \frac{\partial \Delta Q_i}{\partial V_i} = - \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} sin\delta_{ij} - P_{ij} cos\delta_{ij}) + 2V_i B_{ii}$$

3.2.2.3. The second order term

The second order term of the tensor model was formulated in general in the section 3 1.2 In the following, the second order term in the load flow model is introduced. First, we explain how the past load flow iterates $\{x_{-k}\}$ are selected; then how to construct the Z matrix and the M matrix; and finally how to calculate the A matrix and the complete second order term.

In the load flow study, the state x is chosen as the components of the complex bus voltages, that is the bus voltage angle δ and the bus voltage magnitude V. Hence, each difference vector s_k has the following form:

$$s_k = \begin{bmatrix} \Delta\delta_k \\ \Delta V_k \end{bmatrix} = \begin{bmatrix} \delta_{-k} \\ V_{-k} \end{bmatrix} - \begin{bmatrix} \delta_c \\ V_c \end{bmatrix} \tag{3.24}$$

where the notation is understood.

We always choose the most recent iterate x_c, then select each preceding past iterate x_{-k} if the step from it to x_c makes a angle of at least 45°. Normally a flat start value ($\delta_0 = 0$, $V_0 = 1$) is used as an initial guess in the load flow program, so that we set the starting past iterates to have $\delta_{-k} \approx 0$ and $V_{-k} \approx 1$.

Now we describe how to formulate the Z and M matrices Corresponding to equation (3.8b) we formulate each vector component z_k in the Z matrix by substituting the load flow functions (3.22) and the load flow Jacobian (3.23) in, we have.

$$
\begin{bmatrix} z_k \end{bmatrix} = 2 \left\{ \begin{bmatrix} \Delta P_{-k} \\ \Delta Q_{-k} \end{bmatrix} - \begin{bmatrix} \Delta P_c \\ \Delta Q_c \end{bmatrix} - \begin{bmatrix} H_c & N_c \\ J_c & L_c \end{bmatrix} \begin{bmatrix} \Delta \delta_k \\ \Delta V_k \end{bmatrix} \right\} \tag{3.25}
$$

where this notation is also understood.

Next we formulate the M matrix. Each element $m(i,j)$ is equal to the square of the inner product between the ith direction vector s_i and the jth direction vector s_j.

$$
M(i,j) = \left\{ \begin{bmatrix} \Delta \delta_i & \Delta V_i \end{bmatrix} \begin{bmatrix} \Delta \delta_j \\ \Delta V_j \end{bmatrix} \right\}^2 \tag{3.26}
$$

where

$$
i = 1,\cdots,p; \qquad j = 1,\cdots,p; \qquad p \le \sqrt{2n-2-pv}\,.
$$

Therefore, M is a square matrix and has the dimension of $p\times p$.

The last step in the formulation of second order term is calculating the A matrix. As described in Theorem 3.1, $A = ZM^{-1}$, $A \in R^{(2n-2-pv)\text{x}p}$. Finally, the complete second order term can be formulated as $\frac{1}{2}\, A\, (S^t d)^2$, where $S \in R^{(2n-2-pv)\text{x}p}$ and it is composed of p past direction vectors s_k, $k = 1,\cdot\cdot,p$.

3.2.2.4 Complete tensor model equations

By combining the second order term to the Newton load flow model, we now have the complete tensor model equations for load flow study.

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} + \begin{bmatrix} H & N \\ J & L \end{bmatrix} \begin{bmatrix} \Delta \delta \\ \Delta V \end{bmatrix} + \frac{1}{2} \begin{bmatrix} A \end{bmatrix} \left\{ \begin{bmatrix} S \end{bmatrix}^{t} \begin{bmatrix} \Delta \delta \\ \Delta V \end{bmatrix} \right\}^{2} = 0 \qquad (3.27)$$

This is a set of $2n$-2-pv nonlinear equatins for a n bus power system load flow problem. Since we only use p past iterates to estimate the second order terms, actually, equation (3.27) should be quadratic in p variables and linear in the other $(2n\text{-}2\text{-}pv)-p$ variables

The simple form of this quadratic term makes it easy to efficiently store and solve load flow equations in the second order formulation. Compared to those previous second order methods [Sachdev 1977] [Iwamoto 1978] [El-Hawary 1982] [Nagendra Rao 1982], the simplicity of the proposed method can easily be realized A more important feature of this formulation is that with the small changes added to the existing Newton method, the ill-conditioned problem which sometimes occurs in the load flow Jacobian matrix can now be easily solved Numerical simulations carried out so far both on well-conditioned systems and ill-conditioned systems give very good convergence performance.

3.2.3 Solution of the load flow tensor model

3.2.3.1. Solution algorithm

A solution algorithm is now proposed to solve the load flow tensor model. The tensor method is the principal component of the solution strategy. Another important component, used as an alternative, is a modified Newton method [Dennis & Schnabel 1983]. As reported below, the latter is used when the tensor step fails to compute an acceptable state correction.

Algorithm 3.2 An iteration of the tensor model

Given x_c, $F(x_c)$;

Step 1: Select the past points to use in the tensor model from the (p + iteration number) past points.

Step 2: Calculate the second order term of the tensor model, so that the tensor model interpolates $F(x)$ at all the points selected in step 1.

Step 3: Find the root of the load flow tensor model.

Step 4: Select $x_+ = x_c + \lambda_c d_c$, where d_c either is the step calculated in step 3 or the modified Newton step, using a line search to choose λ_c.

Step 5· Calculate the 2-norm power mismatch of $F(x)$ and decide whether to stop; if not Set $x_c \leftarrow x_+$, $F(x_c) \leftarrow F(x_+)$, go to step 1.

The reason of using the 2-norm of the bus power mismatches in Step 5 is that we always have a solution and the program never diverges. The value of this quantity becomes eventully zero if there is a solution from the initial estimate, and stays at a positive value if no solution exists. Here the positive value is the least squares solution of the problem. From this we can know how close we are from the actual solution. This idea was first introduced by [Sasson 1969]

The strategy used in Step 4 is as follows: if in Step 3 we find a root of the tensor model, then we will use this step to update the current state variable $x_c = (\Delta\delta_c, \Delta V_c)$; and if there is no such root for the tensor model we will apply the modified Newton step

In the simulation tests performed so far, for well-conditioned power systems like the 5, 6, 10 bus systems, we always find roots of the tensor models. Even for some claimed ill-conditioned power systems like the 13 and 20 bus systems, [Tripathy etc. 1982] and [Behnam-Guilani 1987] respectively, we have found roots of the tensor models at every iteration. Only for the very badly ill-conditioned power systems like the 11 and 43 bus systems, [Iwamoto & Tamura 1981] and [Tripathy etc. 1982], the algorithm has chosen the modified Newton step, for 1 or 2 iterations.

The modified Newton method implements the Levenberg-Marquardt step [Dennis & Schnabel 1983]. It is shown below:

$$d_n = -\ (J_c^t J_c + \alpha I)^{-1}\ J_c^t\ F(x_c) \qquad (3.28)$$

where

$$\alpha = \sqrt{n*machineps}\ \ ||\ J_c^t J_c ||_1$$

The complete strategy for choosing between the tensor or the modified Newton search direction is given in algorithm 3.3 below:

ALGORITHM 3.3 Step selection

Let J_c = load flow Jacobian $F'(x_c)$,

d_T = root of the tensor model,

d_N = modified Newton step,

IF (no root of the tensor model was found) THEN

$x_+ \leftarrow x_c + \lambda_c d_N$, $\lambda_c \in (0,1)$ selected by line search

ELSE

$x_+ \leftarrow x_c + \lambda_c d_T$, $\lambda_c \in (0,1)$ selected by line search

ENDIF

3.2.3.2. Program implementation

The main flow chart of our computer implemetaion is shown in figure 3.1. Some of the more detailed extra steps are given in what follows:

Step 1. Initialize all the variables before iteration

(a) Calculate computer *machineps* for later use in the modified Newton step.

(b) Read system source data. (number of buses, lines, specified generations, loads, etc.).

(c) Formulate admittance matrix Y.

(d) Set iteration counter and solution accuracy, set starting estimates of state $x_c(\delta_0, V_0)$ equal to $\delta_i^{(0)}=0$ and $V_i^{(0)}=1$, also generate starting p past iterates $\{x_{-p}\}$, and generate S matrix.

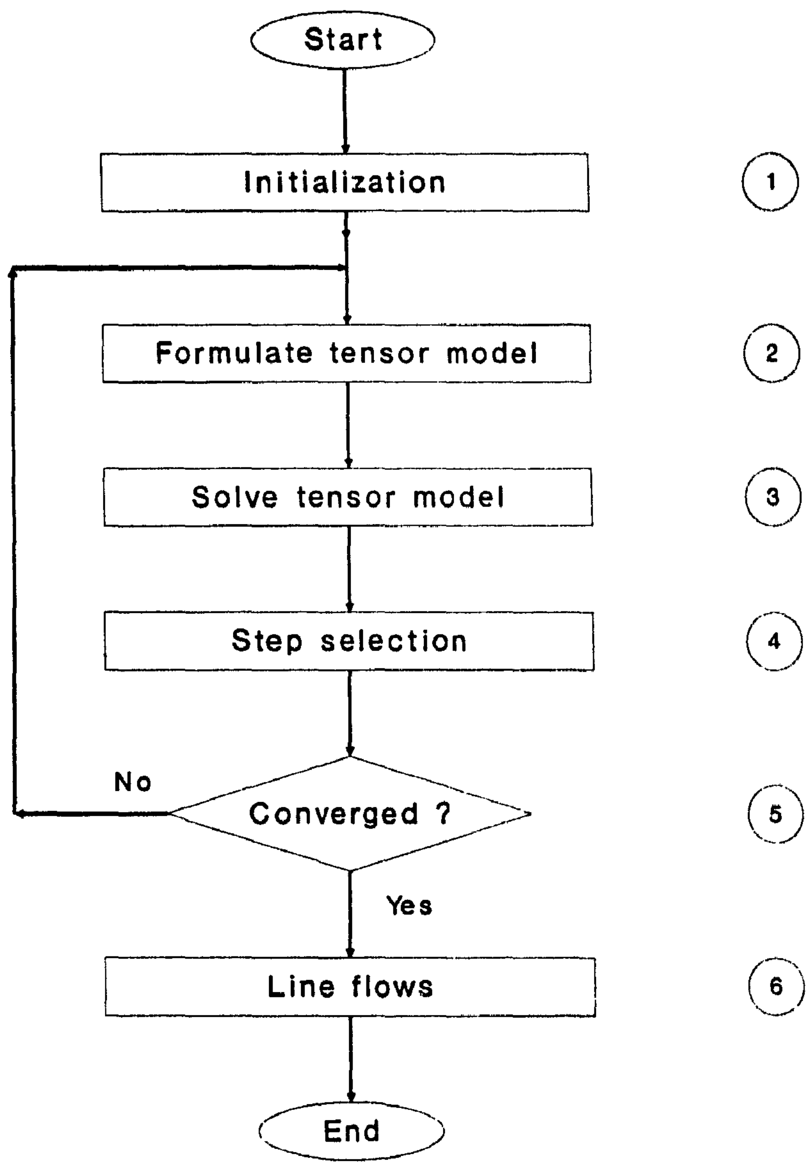

Figure 3.1 Main Flow Chart

Step 2. Formulate tensor model

(a) Select new p past iterates from the set of past iterates $\{x_{-k}\}$, formulate new S matrix. This step is bypassed in the first iteration. The Angle θ for choosing more orthogonal past iterates is set here to be 45°.

(b) Calculate $F(x_c)$, $J(x_c)$, $F(x_{-p})$.

(c) Formulate the Z, M, and A matrices.

Step 3. Solve the tensor model

(a) Estimate the condition number of the load flow Jacobian J matrix. Standard Linpack subroutine DGECO was used for this implementation. If ill-conditioning is detected, compute the modified Newton step for later use.

(b) Two QR decompositions are performed here, on the S matrix to get a lower triangular matrix S_2, and then on the J_1 matrix to split $\hat{d}_1$ from the p nonlinear equations. Both are implemented by using Linpack subroutines DQRDC, DQRSL [Dongarra et.al. 1979].

(c) Calculate the condition number of J_3 and set an initial estimate $\hat{d}_2^{(0)}$ for the p nonlinear equations. For well-conditioned J_3, we use the standard Newton step as the initial estimate. If the conditioning of the system is very bad (very large condition number), in order to avoid unreasonably large state corrections we choose the modified Newton step to start the iteration. Same as in the step 3 (a) Linpack subroutine DGECO is used here.

(d) A nonlinear equation solver NS01A in the Harwell Subroutine Library [Hopper 1977] was used to solve the p nonlinear equations in p unknowns. The call function used here is the small system corresponding to equation (3.15b).

(e) For the remaining n-p linear equations, we applied Linkpack linear equation solver DGEFA and DGESL. The corresponding equation solved here is the large system equation (3.15a).

Step 4. Step selection

(a) Determinate the step to use. This is accomplished by checking the error return messages given by the nonlinear equation solver. If there is a solution, which means the tensor model has a root, then we will choose the tensor step, otherwise if the error return message indicating that there is no solution close to the given initial estimate, then we will select the modified Newton step.

(b) Check if the full step is applicable If not, a simple backtrack line search strategy is used.

Step 5. Convergence test

(a) Calculate the 2-norm value of the bus power mismatches $|| F(x_+) ||_2$.

(b) Check if solution accuracy has been reached, if not go back to step 2; if yes, forward to step 6.

Step 6. Line flows and output results

(a) Calculate system line flows based on the solution state $x_s(\delta_s, V_s)$.

(b) Output the solution results: bus data, line data, and system data.

3.3 Summary

A new method of solving nonlinear equations is introduced in the beginning of this chapter, namely the tensor method, which is especially suitable of solving ill-conditioned nonlinear systems Application of the tensor method to solve the power system load flow problem is carried out in the latter part of this chapter Both the formulation and solution of the load flow tensor model are explained thoroughly, and details of our computer implementation are given in the final section. Although the proposed tensor method seems mathematically more complicated than the widely used stardard Newton method and the Fast Decoupled load flow method, its superior numerical stability on the ill-conditioned problems and fast convergence rate make it very attractive With skillful sparse programming techniques applied, it is very likely to perform the load flow calculations in real-time for electric power networks.

Chapter IV

Numerical Simulations

4.1 Introduction

A load flow program has been written implementing the ideas of chapter III for finding the ill-conditioned power system load flow solutions. Extensive numerical simulations have been carried out on the 11, 13, 20 and 43 bus ill-conditioned systems, as well as 5, 6, and 10 bus well-conditioned systems This chapter documents and analyzes all the simulation results.

The above chosen systems found in the literature impose a particular problem — difficulty in converging to a solution In our simulations, a measure of this ill-conditioning is computed and it was found that the condition numbers of the ill-conditioned systems are much higher than for the well-conditioned systems Despite ill-conditioning, we solved all the above systems by the tensor method Comparisons with the Fast Decoupled load flow method have also been made. In general, the tensor method shows very good convergence characteristics on both ill-conditioned and well-conditioned problems

The results are presented for each ill-conditioned system separately and for the three well-conditioned systems altogether Contents of the various tables and graphs in each section will be discussed in detail The format for presenting the results in this chapter is as follows.

In the simulation of ill-conditioned systems. First, the load flow solutions reached for each system is presented. Then, the convergence characteristics are shown, and the ill-conditioning of the system is analyzed. Finally, for the 11 and 43 bus systems, some special features of the program are discussed.

Following the sections for ill-conditioned systems, the results for the three well-conditioned systems are given and discussed.

This chapter closes with a discussion on the general performance of the proposed tensor method and comparisons with standard load flow methods

4.2 Simulation on an 11 Bus System

The 11 bus system is taken from the paper by [Tripathy et.al 1982]. The line and bus data are only available in the form of Y admittance matrix elements and net bus powers. These data and the schematic diagram for the 11 bus system can be found in Appendix 4.1.

The number of variables in this system is as follows:

Number of buses:	11
Number of transmission lines:	11
Number of generations:	1
Number of loads:	5
Number of state variables:	20

4.2.1 Load flow solutions

The program solved this system for the given data in seven iterations, to reach a prescribed tolerance of 0.001 MW(MVAR) for the 2-norm value of bus power mismatches. The final solutions for the 11 bus system are given in Table 4.1.

Table 4.1. Solutions of 11 Bus System

Bus	Voltage V(p.u.)	Angle δ(deg.)	Real Mismatch $\lvert\Delta P\rvert$ (MW)	Reactive Mismatch $\lvert\Delta Q\rvert$ (MVAR)
1	1.071	-24 605	0 234e-3	0 229e-4
2	1 057	-2 408	0 688e-5	0 114e-4
3	1 046	-4 102	0 182e-5	0 722e-5
4	1.031	-2.836	0 359e-4	0 484e-4
5	1 035	-4 851	0 488e-5	0 859e-5
6	1 051	-2.920	0 571e-5	0 207e-4
7	0 810	-12.504	0.274e-4	0 968e-4
8	0.910	-15 390	0 196e-3	0 392e-4
9	1 195	-16 319	0 163e-3	0 139e-3
10	0 816	-21 872	0 235e-3	0 163e-4
11	1.024	0.0	—	--

This is a reasonable and accurate solution for the system From the data and schematic diagram, we know this is a radial system with only one generation at one end (bus 11) of the network The other end (bus 1) has a large load, thus requiring the entire system to transfer power from the generation bus. This results in the largest bus voltage phase angle (-24 6°) being at load bus 1 This configuration also causes low bus voltage magnitudes at connection buses 7, 8, and 10, which read 0 81, 0 91, and 0 816 respectively

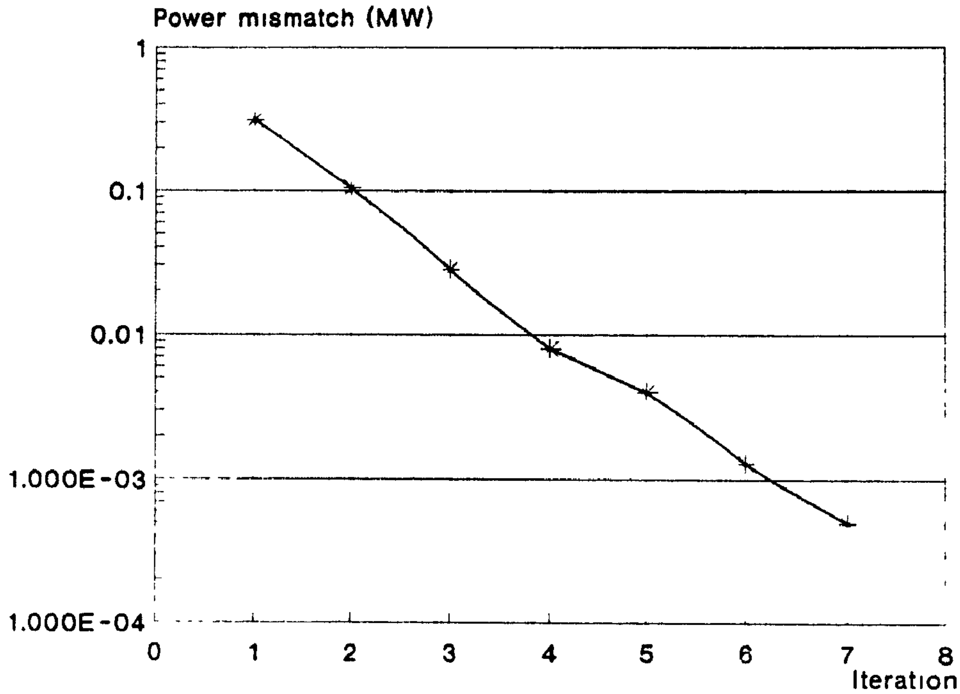

Figure 4.1. Power mismatch vs. iteration for 11 bus system

4.2.2 Convergence characteristics

Previously, load flow calculations for the 11 bus system were considered divergent by any method [Iwamoto & Tamura 1981] except for Brown's method [Tripathy et.al. 1982] Applying the proposed tensor method, the program has never diverged. In fact, it reduces the 2-norm bus power mismatch at every iteration In this simulation the program converges very quickly with only 7 iterations to reach a tolerance of 0 001 MW(MVAR) Figure 4 1 shows the convergence characteristics of the 11 bus test system

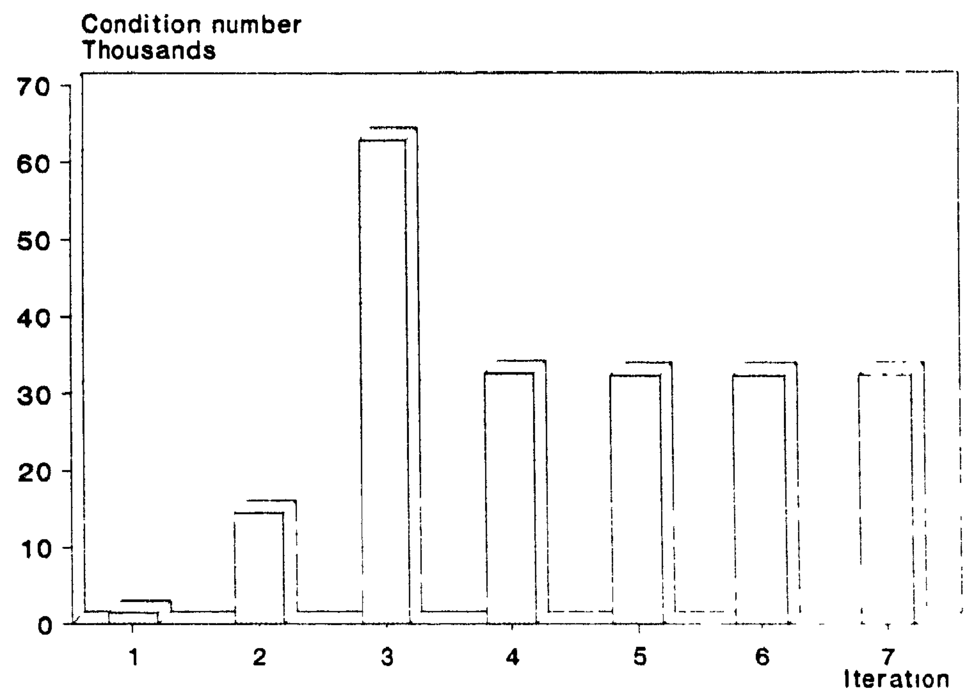

Figure 4.2. Condition number vs. iteration for 11 bus system

4.2.3 Ill-conditioning

Physically, this 11 bus system has two reasons to be considered an ill-conditioned power system First, it has a long radial network; secondly, there is only one generation bus and it lies at one end of the network To supply the large load at bus 1, power can only be transferred from the other end of the network, which makes for difficult operation of the system

In the simulation, this ill-conditioning phenomenon was detected using the condition number of the load flow Jacobian matrix Our simulations show that this system has a very high condition number at the solution For a general well-conditioned system of dimension 20, the condition number is typically around the hundreds For this particular system, however the condition number at the solution is around 30,000 Also we noticed in the third iteration an extremely high condition number (62,889) The tensor model did not have a root in this iteration, so to avoid large changes in the state variables, the program selected a modified Newton step for this iteration

Figure 4 2 shows the 11 bus system condition number versus the iteration number.

4.2.4 Behavior of the modified Newton step

In the simulations carried out on the 11 bus system, we found two solutions for the given data in different runs Both of them seem reasonable, but one is definitely a better operating condition than the other, being closer to the flat voltage profile.

We noticed that this behavior of the algorithm is related to the size of the modified Newton step. Actually it depends on the step size "α" value in equation (3.28) This defines how much change is made to the diagonal elements in the ill-conditioned load flow Jacobian matrix

In the first case we use the modified Newton step (3.28) without changing the α value. The program takes 17 iterations to reach a tolerance of 0 01 MW (MVAR) on the 2-norm bus power mismatch This is our first solution of the problem In this solution process we observed large corrections are made in the system bus voltages from one iteration to the next In the first two iterations the voltage magnitude at bus 10 had dropped down to a value of 0.155 p u and then slowly climbed back At the solution, the lowest bus voltage is 0 731 p u at bus 10

To avoid unreasonablly large corrections in the system bus voltage calculations, we modified the α value in the equation (3 28) to the following value,

$$\alpha = \sqrt{10^5 * n * machineps} \; \|J_c^t J_c\|_1$$

This results in the second solution, the better solution, which is given in Table 4.1. With this new α value, the program homed in on the load flow solutions more rapidly.

4.2.5 Single precision problem

In solving the 43 bus system, we extended the memory requirement beyond the 640k byte DOS memory limit when using a double precision version of the program. In the case of the 43 bus system, a single precision program was used and converged very slowly In order to gain more information on the performance of the single precision version program, a version of the 11 bus system was also run in single precision.

In our single precision test on the 11 bus system, after three iterations, the largest element of the initial guess sent to the nonlinear equation solver (Newton step in the small system) becomes less than 10^{-2} For the second order term $\frac{1}{2} T(dd)^2$, therefore, we need a machine accuracy of $(10^{-2} \times 10^{-2})^2 = 10^{-8}$ which is out of the single precision accuracy Hence the tensor model can not be correctly presented, and consequently the program could not find the root of the tensor model Then the program chose the modified Newton step instead, which resulted in very slow convergence. When the double precision program was excuted, this kind of problem did not happen.

4.3 Simulation on a 13 Bus System

The 13 bus system is taken from [Zollenkopf 1968] The system is claimed difficult to solve because it contains two series capacitors and also because of the position of the slack generator. The data and the schematic diagram for this system can be found in Appendix 4.2

The number of variables in this system is as follows:

Number of buses:	13
Number of transmission lines:	13
Number of generations:	6
Number of loads:	4
Number of transformer taps:	3
Number of state variables:	19

4.3 1 Load flow solutions

The program solved this system for the given data in six iterations to reach a prescribed tolerance of 0 001 MW(MVAR) for the 2-norm value of bus power mismatches. The final solutions for the 13 bus system are given in Table 4.2.

Table 4.2. Solutions of 13 Bus System

Bus	Voltage V(p.u)	Angle δ(deg)	Load P_1(p u)	Load Q_1(p u)	Generation P_g(p.u.)	Generation Q_g(p.u)
1	1.054	1 571	0.0	0 0	0.0	0 0
2	1.143	2.530	0 0	0 0	0.0	0 0
3	1.135	2.548	0.0	0 0	0 0	0.0
4	1.063	8.891	0 0	0 0	0 0	0 0
5	1.044	5.122	0 0	0.0	0.0	0.0
6	1.067	7.988	−0.050	−0 032	0.0	0 0
7	1.017	12.003	−0.050	−0 030	0 0	0 0
8	0.943	14.246	0.0	0 0	0 500	-1 006
9	1.100	8.232	0.0	0 0	0 0	−0 086
10	1.100	8 020	0 0	0 0	0 0	−0 784
11	1.000	2.623	0.0	0 0	0 0	−0 335
12	1.037	9.681	−0.050	−0.030	0 500	−0 421
13	1.000	0.0	−1.650	−0 560	0 824	0.146

Power Base = 1000 MVA

4.3.2 Convergence characteristics

The 13 bus system is claimed divergent by the Gauss-Seidel method [Zollenkopf 1968], [Keyhani et al 1989] Our own Fast Decoupled load flow program solves this system in six iterations to reach a tolerance of 0.001 MW(MVAR) for the maximum real or reactive power mismatch Applying the proposed tensor method, the program converges very fast, requiring only six iterations to reach a tolerance of 0 001 MW(MVAR) for the 2-norm value of bus power mismatches, which is a more accurate tolerance Figure 4 3 shows the convergence characteristics of the 13 bus test system

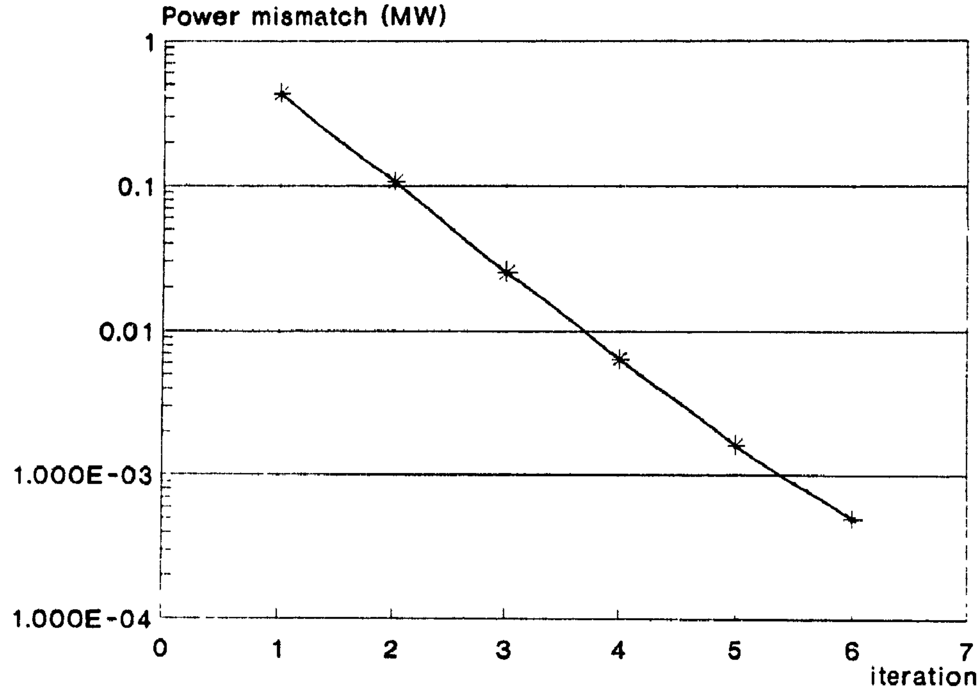

Figure 4.3. Power mismatch vs. iteration for 13 bus system

4.3.3 Ill-conditioning

This 13 bus system seems difficult to solve because it contains two series capacitor branches. Actually from our simulation results we found that the condition number of the system is always low. In fact, this system is fairly well-conditioned. This explains the success of our Fast Decoupled test, despite claims of difficulty from previous authors

Figure 4.4 shows the 13 bus system condition number versus the iteration number. From the figure we see that the condition number of this system at the solution is only 71. In the first iteration, it is 281 which is quite normal for a 19 dimensional system.

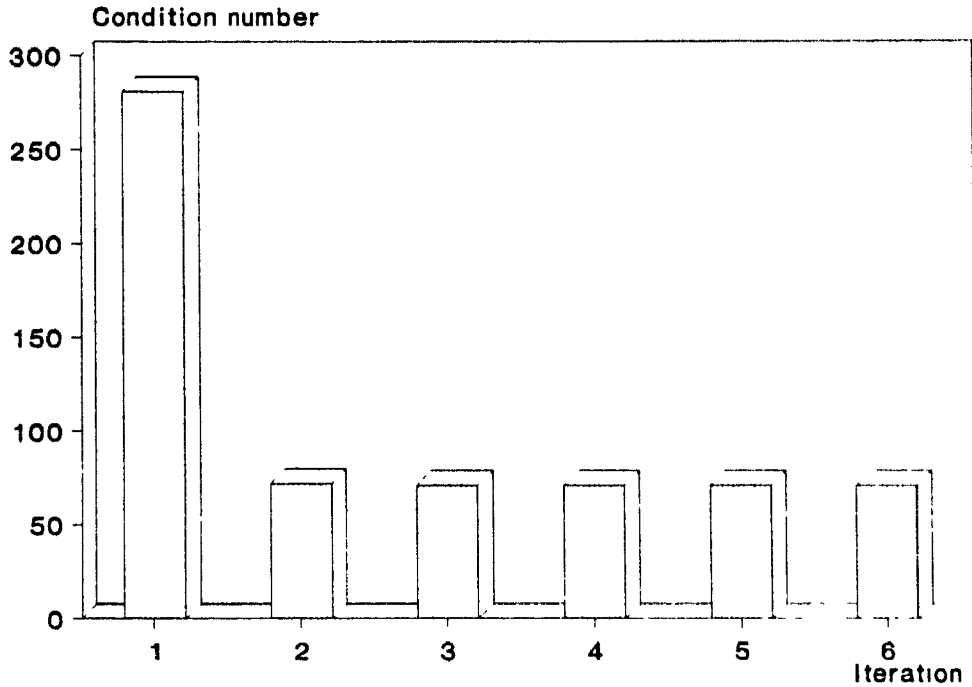

Figure 4.4. Condition number vs. iteration for 13 bus system

4.4 Simulation on a 20 Bus System

The 20 bus system is taken from [Behnam-Guilani 1987] It is difficult to solve because of the predominantly radial topology, atypical circuit parameters and clustered generations and loads The data and the schematic diagram for this system can be found in Appendix 4.3.

The number of variables in this system is as follows·

Number of buses:	20
Number of transmission lines:	22

Number of generations: 4

Number of loads: 8

Number of state variables: 35

4.4.1 Load flow solutions

The program solved this system for the given data in seven iterations to reach a prescribed tolerance of 0.001 MW(MVAR) for the 2-norm value of bus power mismatches. The final solutions for the 20 bus system are given in Table 4.3.

Table 4.3. Solutions of 20 Bus System

Bus	Voltage	Angle	Load		Generation	
	V(p u)	δ(deg.)	P_1(p.u.)	Q_1(p.u.)	P_g(p.u)	Q_g(p.u.)
1	0.975	−4.328	−1.500	−0.300	0.0	0.0
2	0.801	−14.932	−0.100	0 0	0.0	0.0
3	0.998	0.153	0.0	0.0	0.0	0.0
4	0.901	−15.446	−3.800	−0.600	0.0	0.0
5	1.001	3 923	0 0	0.0	0.0	0 0
6	0.968	−8.293	0.0	0.0	0.0	0.0
7	1 020	−3.341	−0 200	0.0	0.0	0 0
8	0.789	−15 305	−0 100	−0.200	0.0	0.0
9	0 999	9 333	0.0	0.0	0.0	0.0
10	0 991	7.024	−0.500	−0.100	0.0	0.0
11	1.002	1.779	0.0	0.0	0.0	0.0
12	1 020	−2 784	0 0	0.0	0.0	0.0
13	1 002	0 909	0.0	0.0	0.0	0.0
14	1 000	10 608	0.0	−0.100	0.0	0.0
15	1 001	−0.393	0.0	0.0	0.0	0.0
16	0 804	−14.726	−0.100	0.0	0.0	0.0
17	1.000	−13 111	0 0	0 0	1.000	1.778
18	1.000	0.566	0.0	0.0	1.000	−0.169
19	1 000	10.720	0 0	0.0	1.000	−0.050
20	1.000	0 0	0 0	0.0	3.925	0 337

Power Base = 100 MVA

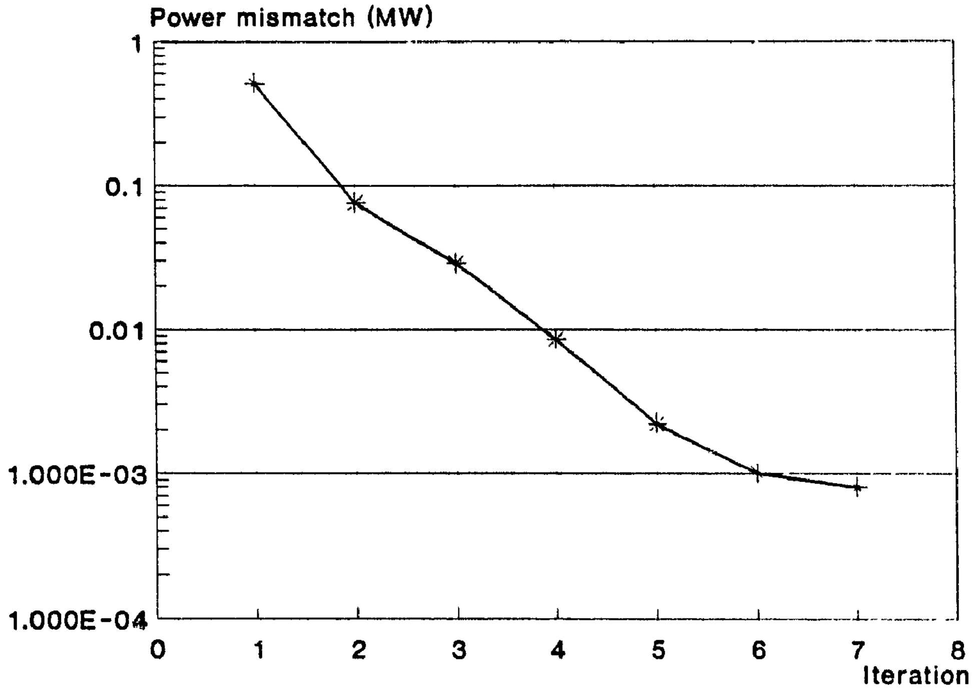

Figure 4.5. Power mismatch vs. iteration for 20 bus system

4.4.2 Convergence characteristics

Not many comparisons have been made previously on the convergence rate of the 20 bus system [Behnam-Guilani 1987]. Our Fast Decoupled load flow program solves the system in 22 iterations to reach a tolerance of 0.001 MW(MVAR) for the maximum bus power mismatch. Applying the proposed tensor method, the program converges in only seven iterations to a tolerance of 0.001 MW(MVAR) for the 2-norm value of bus power mismatches. Figure 4.5 shows the convergence characteristics of the 20 bus test system

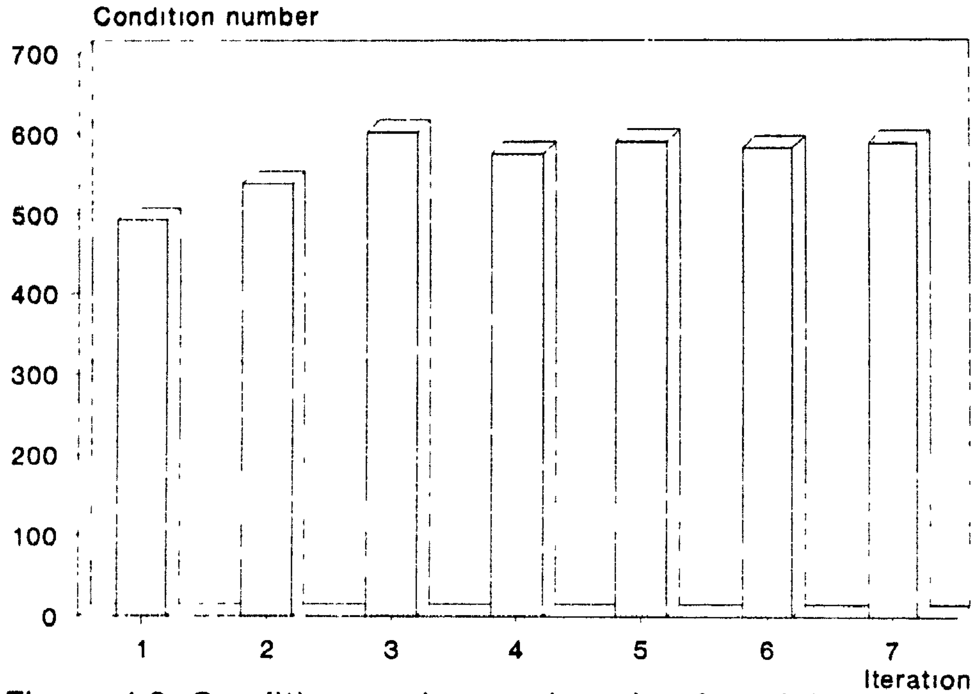

Figure 4.6. Condition number vs. iteration for 20 bus system

4.4.3 Ill-conditioning

The 20 bus system is claimed hard to solve in [Behnam-Guilani 1987] This difficulty was evidenced in our Fast Decoupled load flow program, which required 22 iterations to reach an accurate solution By contrast, the ill-conditioning has little or no effect on the tensor method, as witnessed in the convergence shown in figure 4.5 The condition number of the system was slightly high (around 600) for this 35 dimensional system.

Figure 4 6 shows the 20 bus system condition number versus the iteration number.

4.5 Simulation on a 43 Bus System

The 43 bus system is also taken from [Tripathy et al. 1982]. The line and bus data is only available in the form of Y admittance matrix elements and net bus powers These data and the schematic diagram for the 43 bus system can be found in Appendix 4.4

The number of variables in this system is as follows'

Number of buses	43
Number of transmission lines	42
Number of generations	5
Number of loads·	15
Number of state variables:	84

4.5.1 Load flow solutions

The program solved this system for the given data in seven iterations to reach a tolerance of 0 07 MW(MVAR) for the 2-norm value of bus power mismatches The corresponding maximum real and reactive power mismatches are 0 0077 MW and 0 0286 MVAR The final solutions for the 43 bus system are given in Table 4 4

The large dimension of this simulation resulted in a memory management problem The double precision version of the program exceeds the **640k bytes** memory space limit set by DOS Therefore, a single precision version program was actually tested As discussed earlier, the single precision contributes to the slow convergence, which improved only marginally after the third iteration

Table 4.4. Solutions of 43 Bus System

Bus	Voltage V(p u)	Angle δ(deg)	Real Mismatch $\lvert\Delta P\rvert$(MW)	Reactive Mismatch $\lvert\Delta Q\rvert$(MVAR)
1	1 032	-11 698	0 203e-3	0 325e-3
2	1 045	-11.137	0 171e-2	0 508e-2
3	1 031	-15 163	0 727e-3	0 281e-3
4	0 999	-13 690	0 613e-3	0 852e-3
5	1 042	-11 256	0 541e-4	0 136e-2
6	1.044	-11 506	0.555e-3	0.731e-3
7	1 049	-15 199	0 667e-3	0.990e-3
8	1 022	-13 358	0 165e-2	0 382e-3
9	1 020	-13 381	0 833e-4	0.751e-3
10	1 023	-13 510	0 101e-2	0 667e-5
11	1 023	-13 494	0 393e-3	0.863e-3
12	1 051	-15 430	0 279e-3	0 122e-2
13	1 001	-13 654	0 530e-3	0.854e-3
14	1 018	-14 532	0.656e-3	0.826e-3
15	1 041	-11 389	0 538e-3	0 261e-3
16	0 987	-17 519	0 641e-3	0.551e-3
17	1 023	-13 499	0 876e-3	0 328e-3
18	1 021	-15 883	0 651e-3	0.184e-3
19	1 043	-16 264	0 192e-2	0 388e-2
20	1 053	-14 300	0 653e-3	0 114e-2
21	1 012	-13 231	0 492e-4	0 187e-2
22	1 040	-16.319	0 672e-3	0 161e-2
23	1 020	-13 278	0 668e-3	0 558e-3
24	1 038	-16 999	0 316e-3	0.238e-3
25	1 009	-13 418	0 404e-3	0 122e-2
26	1 022	-21 094	0 196e-2	0 325e-2
27	1 035	-15 360	0 502e-3	0 886e-4
28	1 039	-11.553	0 106e-2	0.109e-2
29	1 017	-13 171	0 947e-3	0 193e-3
30	0 910	-12 533	0 391e-2	0.213e-1
31	1 039	0 0002	0 172e-3	0 567e-3
32	0 989	-6 505	0 768e-2	0 176e-1
33	1 006	-17 557	0 273e-2	0 215e-1
34	0 859	-22 928	0 197e-2	0.259e-1
35	1 008	-17 116	0 275e-2	0 215e-1
36	1 027	-24 078	0 237e-2	0.203e-1
37	1 004	-4 890	0 194e-3	0 758e-3
38	0 927	-22 906	0 221e-2	0 258e-1
39	1 034	-11 673	0 328e-3	0 704e-3
40	0 904	-12 759	0.295e-2	0 236e-1
41	1 013	-14 589	0 640e-3	0.701e-3
42	0 871	-20 329	0.308e-2	0.286e-1
43	1 136	0 0	—	—

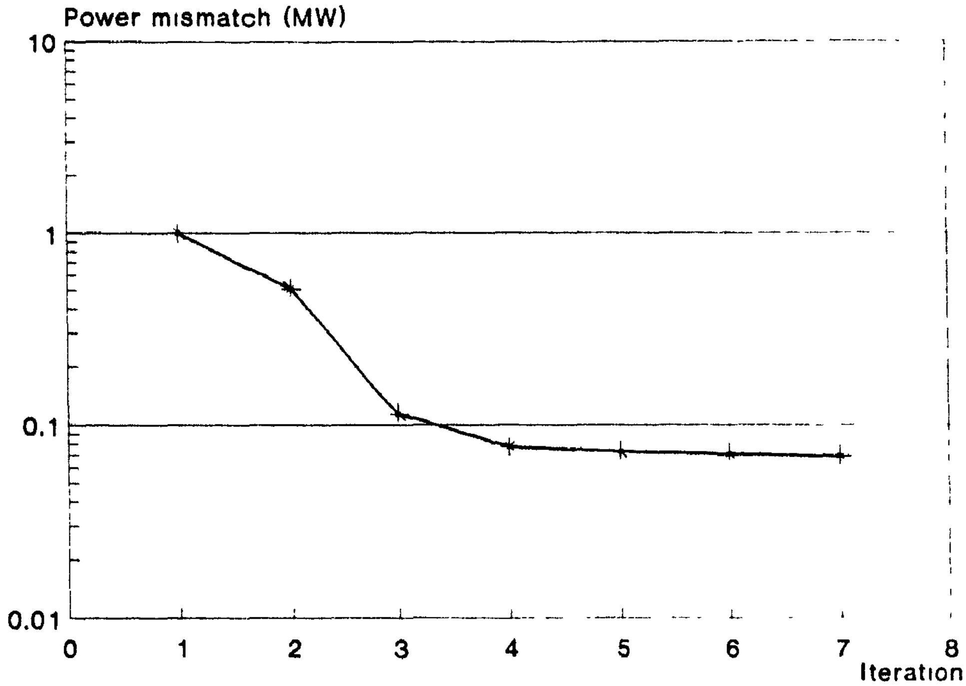

Figure 4.7. Power mismatch vs. iteration for 43 bus system

4.5.2 Convergence characteristics

Figure 4 7 shows the convergence characteristics of the 43 bus testing system, using the single precision version of the program. The reason for such slow convergence after three iterations is that the direction vector *d* in the tensor model drops to less than 10^{-2} Again, in the second order term of the tensor model we need at least a machine accuracy of 10^{-8} to correctly incorporate this information, but this cannot be satisfied by the single precision program That also explains why after three iterations the program could not find the root of the tensor model Instead, with a modified Newton step we get a very slow convergence rate Based on our similar simulations carried out on the 11 bus system, it is felt that convergence would be faster if double

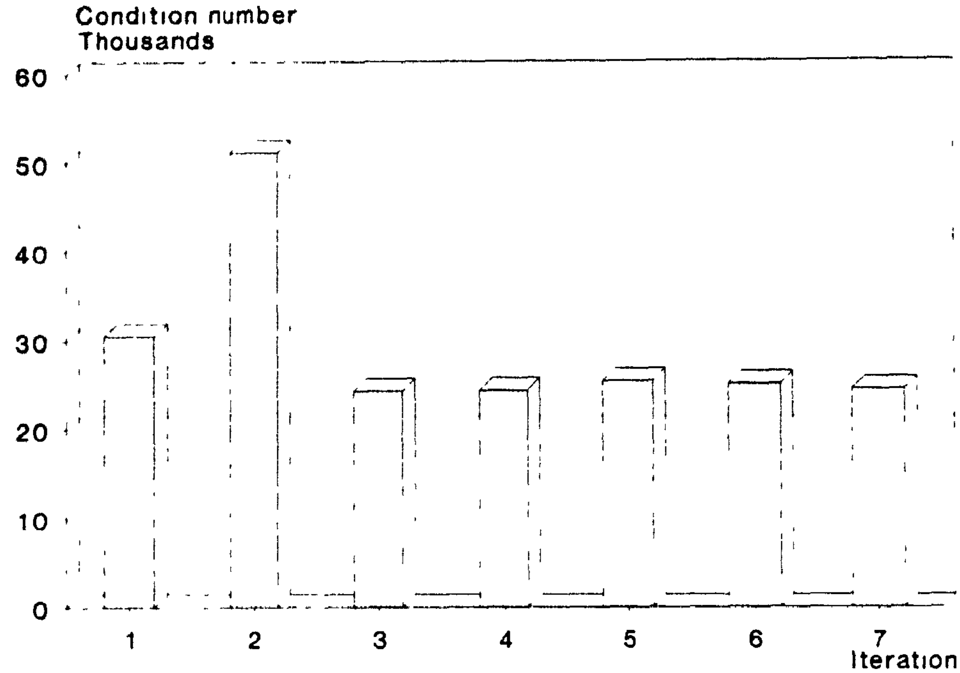

Figure 4.8. Condition number vs. iteration for 43 bus system

precision could be used.

4.5.3 Ill-conditioning

The 43 bus system is claimed difficult to solve by many researchers [Stott & Alsac 1974] [Iwamoto & Tamura 1981] [Tripathy et.al 1982] This system is difficult to solve because of high *R/X* ratios, some negative line reactances, and because of its radial topology

In the simulation this ill-conditioning phenomenon was observed. Figure 4.8 shows the 43 bus system condition number versus the iteration number This system has a very high condition number (around 25,000) at the solution, as for the 11 bus system. Even though the condition number at the second iteration is extremely high (51,125), our program still found a root for the tensor model

4.5.4 Behavior of the modified Newton step

In the simulation of this 43 bus system, as for the 11 bus system, the size of the modified Newton step has been an important factor in improving the convergence

At the beginning we applied equation (3 28) without changing the α value, but the convergence was very slow. The correction on the state variables is about 0 1% per iteration Then we tried various values for α. The fastest convergence was obtained with,

$$\alpha = \sqrt{10^{-8} * n * machineps} \; \| J_c^t J_c \|_1$$

For the time being, we only set this α value at the beginning of the first iteration. The optimal α values for each individual iteration would be very useful to gain an overall fast convergence, but the computational effort required is unknown.

4.6 Simulations on Well-Conditioned Systems

In order to test the versatility of the tensor method, simulations were also carried out on the well-conditioned systems A 5 bus system from [Xi'an Jiao-tong University 1978], a 6 bus system from [Dhar 1982] and a 10 bus system from [Huneault 1988] are chosen Comparisons between the Tensor method and the Fast Decoupled load flow method have been made

4 6.1 Convergence characteristics

These well-conditioned systems are solved without difficulty by the standard methods, but the convergence rates of these methods are quite different Applying the tensor method to the load flow calculations, fast convergence rates were obtained for the 5, 6, and 10 bus systems On solving the 6 and 10 bus systems, the tensor program only needed 4 iterations to reach a tolerance of 0 001 MW(MVAR) in the 2-norm value of bus power mismatches For the 5 bus system, which is quite heavily loaded, the program takes 5 iterations to reach the same accuracy In comparison, our Fast Decoupled load flow program required 10, 5 and 5 iterations for the 5, 6 and 10 bus systems respectively

Figure 4 9 shows the convergence characteristics of these well-conditioned systems by use of the tensor method.

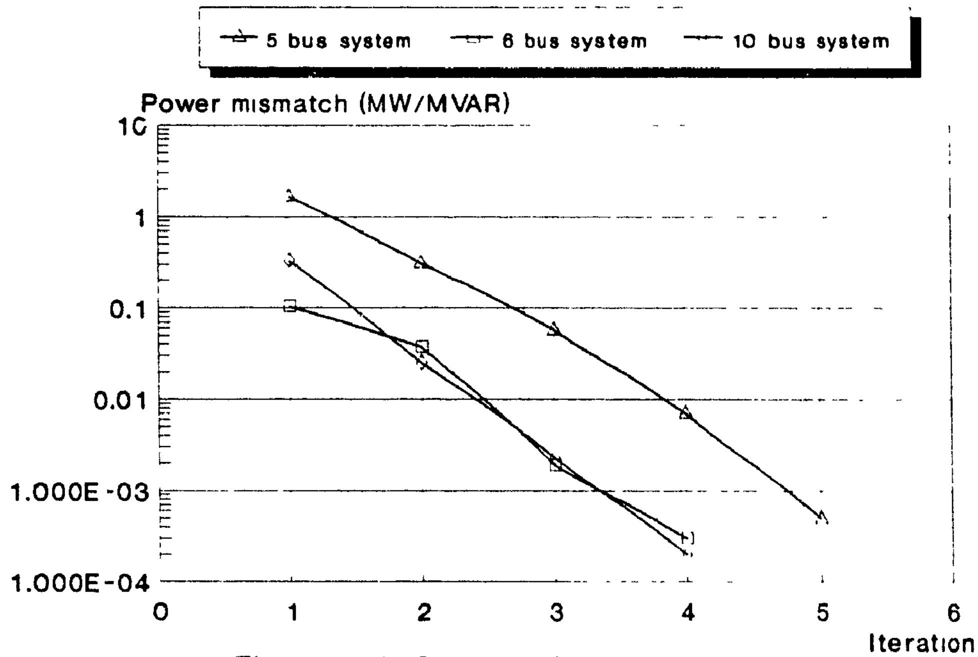

Figure 4.9. Power mismatch vs. iteration

4.6.2 Condition number

Conditioning of the 5, 6, and 10 bus systems were also tested. with condition numbers presented in Figure 4 10 These values are all quite small. The 5 bus system has a higher condition number than the 6 and 10 bus systems, possibly because it is more heavily loaded.

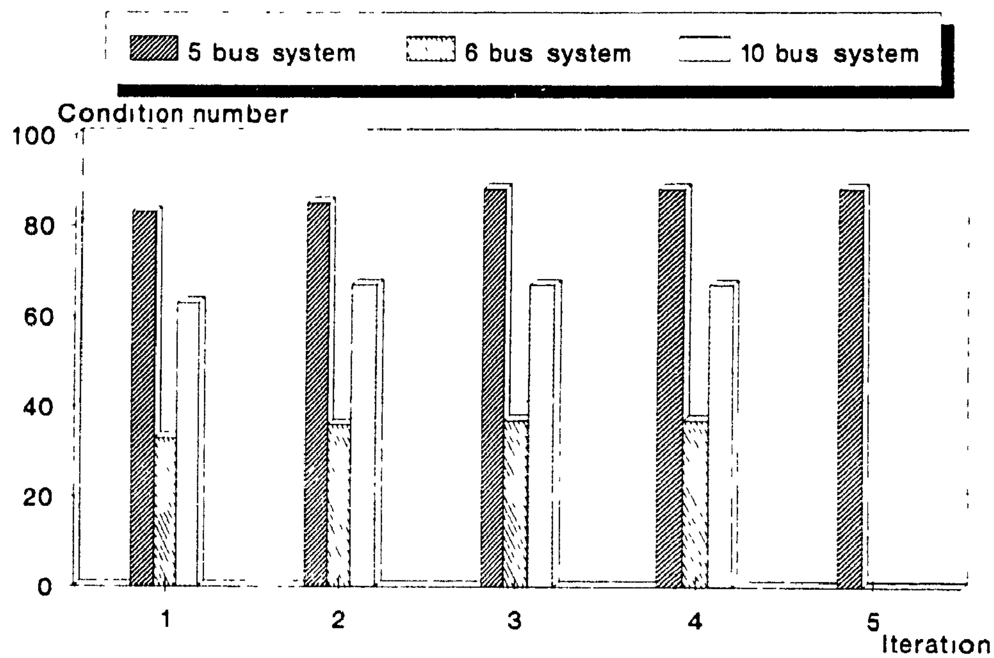

Figure 4.10. Condition number vs. iteration

4.7. Discussion

This section regroups the general results and comments made in the previous descriptive sections 4.2 to 4.6 It discusses the general behavior of our tensor algorithm as observed in the results, and proposes a comparison of the more important aspects between the tensor method and standard load flow methods.

In our tests the convergence observed for all the systems except the 43 bus (see the reason in Section 4 5) seems quadratic On most of the problems, the program only needs 3 or 4 iterations to get a acceptable solution and 6 or 7 iterations to reach an accurate solution

The other advantage of the proposed tensor method is its numerical stability In our tests, the program based on Algorithm 3 2 solves all the test problems In studying the results, we found that it reduces the bus power mismatches at every iteration

In practical planning, engineers often talk of "weak systems" Loadflow solutions may become difficult for such systems when the voltage operates on the verge of instability Typically, where system voltages drop to about 80% or below, loadflow solutions often fail to converge In our simulation on the 11 bus system, the tensor method shows great numerical stability in solving such kind of "weak systems". In this case, the voltages at certain buses dropped to about 81%, still the tensor method succeeded in finding a solution

In our simulations the step chosen by the algorithm in most cases is the tensor step, especially when approaching a load flow solution. This tensor step is observed to approach to a solution faster and is more stable than the standard Newton step In some cases, when the ill-conditioning of the system is extremely bad, like for the 11 and 43 bus systems, the modified Newton step is required. We find that the size of the modified Newton step is very important To obtain the fastest convergence characteristics, the α value in the modified Newton step needs to be determined At the current stage, we do not have an algorithm to select the optimum α value automatically Instead, we accomplished this by running many different trials

The preselected past iterates also showed some effect on the difficult problems We noticed in the simulations that the larger the angles between these past iterates, the better the stability of the algorithm. Normally, it is not difficult to find such past iterates

The tensor algorithm converges for the well-conditioned systems in fewer iterations than for the standard methods. This is not to say that the tensor method is faster than the standard load flow methods, because the latter may reqiure less time per iteration The two groups of programs (FDLF and TLF) are difficult to compare for times, because one was using compact storage taking advantage of sparse matrices, and the other was not. Nevertheless, the fewer iterations required by the tensor method are impressive.

Table 4.5. Convergence Comparison with FDLF Method*

Type of system	Number of buses	Iteraions to solution	
		FDLF method	Tensor method
Ill-conditioned systems	11 13 20 43	divergent 6 22 22	7 6 6 7**
Well-conditioned systems	5 6 10	10 5 5	5 4 4

* Solution accuracy set to be maximum power mismatch less than 0 001 MW and/or MVAR.

** In this case applied tensor method only reaches an accuracy of 0 02 MVAR See corresponding case study for the reason.

Comparisons between the tensor method and Fast Decoupled load flow method have been made regarding the convergence rate on all the test systems in Table 4.5. Comparisons between the tensor method and standard load flow methods on all aspects concerning the cost of computer implementations have also been made in Table 4 6

Table 4 5 shows the superior numerical stability and convergence characteristics of the tensor method over the Fast Decoupled load flow method, both on the ill-conditioned and well-conditioned test systems.

Table 4.6. Cost Comparison with Standard Method

Aspect of cost	Newton method	FDLF method	Tensor method
Speed	Standard	Fast	Major cost is on the QR transformation, which takes twice the time of a LU decomposition in NR method
Storage	Standard	Small	Only add $4p$ n-vectors to storage of NR method If sparse technique is used, only slightly higher than Newton method

Table 4.6 indicates the differences between the tensor method and standard load flow methods on the costs of computer implementation On the side of solution speed, using a QR decomposition, an iteration of the tensor method takes twice as much time as the standard Newton method, where a LU decomposition is often used. An alternative method using LU decomposition to solve the tensor model is also available [Schnabel & Frank 1984] Regarding the storage requirement of the

tensor method, we only added $4p$ n-vectors to the storage requirements of the standard Newton method. If sparse techniques and skillful programming are applied, the core requirments are only slightly higher than for the Newton method.

In summary the proposed tensor method has shown very good numerical stability and convergence characteristics on both ill-conditioned and well-conditioned test systems. Despite the fact that it requires a slightly higher storage requirement and that it takes a little more time per iteration than the standard load flow methods, its superior performance on ill-conditioned problems makes it very attractive.

Chapter V

Conclusions and Recommendations

This thesis studies the problem of ill-conditioned power systems load flow calculations. A new method, namely the tensor method, is applied to solve the ill-conditioned system load flow problem. Both the formulation and simulation are introduced and analyzed thoroughly. The results obtained from the simulations reveal both the strengths and weaknesses of the proposed tensor method.

5.1 Conclusions

1. The thesis has been very successful on achieving its goal — application of the tensor method for the ill-conditioned system load flow calculations.
2. The simulation results showed superior numerical stablity and convergence characteristics of the tensor method over the standard load flow methods on both ill-conditioned and well-conditioned test problems.
3 In particular, for solving ill-conditioned load flow problem, the tensor method has shown itself to be faster and more robust than the usual alternative, the modified Newton method.
4. Regarding the program complexity and computational effort, currently the proposed tensor method is recommended

especially for the ill-conditioned system load flow solutions, where the standard load flow methods usually fail.

5.2 Recommendations for Future Research

Here is a list of modifications which could improve the performance of the tensor algorithm:

1. To improve the performance of the modified Newton step on the difficult problems, we have mentioned the importance of the step size of the modified Newton step An algorithm for searching the optimium "α" value in the modified Newton step would be useful.
2. The method proposed in this thesis can be adapted easily to remain efficient on large, sparse power systems In particular, the main additional computational costs of the tensor method are QR transformations One suggested modification would be to use an efficient sparse factorization in the algorithm for solving the tensor model. Also, one could investigate the use of sparse QR factorization [George & Ng 1986].
3. Alternative approaches for solving the tensor model are available using a PLU factorization [Schnabel & Frank 1984].
4. As an experiment, one could use the Decoupled load flow Jacobian matrix approximation in the tensor model

formulations.

5 The addition of passive controls, i.e. taps, shunts and phase shifters control to the load flow formulation

REFERENCES

Alvarado F.L., "*Discussion of [Behnam-Guilani 1987]*", IEEE Trans on PS, vol.3, no.2, pp.741, May 1988.

Babié B.S., "*Decoupled Load Flow with Variables in Rectangular Form*", Proc.of IEE, vol.130, no.3, pp.98-102, 1983.

Behnam-Guilani K., "*Fast Decoupled Load Flow: The Hybrid Model*", IEEE PICA Conf., pp.354-359, Montreal, Quebec, Canada, May 18-22, 1987

Brameller A., Denmead J.K., "*Some Improved Methods of Digital Network Analysis*", Proc. of IEE, pt.A, vol.109, pp.109-116, 1962

Decker D.W., Kelley C.T., "*Newton's Method at Singular Points I*", SIAM Journal on Numerical Analysis, vol.17, pp 66-70, 1980

Decker D W., Kelley C.T., "*Newton's Method at Singular Points II*", SIAM Journal on Numerical Analysis, vol.17, pp.465-471, 1980

Dennis jr J.E., Schnabel R.B., "*Numerical Methods for Unconstrained Optimization and Nonlinear Equations*", Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

Dhar R.N., "*Computer Aided Power System Operation and Analysis*", McGraw-Hill, 1982.

Dommel H.W , Tinney W F., Powell W.L., "*Further Developments in Newton's Method for Power System Applications*", IEEE Winter Power Meeting, New york, Jan. 25-30, 1970.

Dongarra J J , Bunch J R., Moler C.B., Stewart W., "*LINPACK User's Guide*", Society for Industrial and Applied Mathematics, Philadephia, 1979

Dunstan J.B., "*Digital Load Flow Studies*", AIEE Trans., pt.3A, vol.73, pp.825-831, 1954

El-Hawary M E., "*Optimal Economic Operation of Electric Power Systems*", Academic Press, 1979.

El-Hawary M E., Wellon O.K., "*The Alpha-Modified Quasi-Second Order Newton Raphson Method for Load Flow Solutions in Rectangular Form*", IEEE Trans on PAS, vol 101, no.4, pp.854-866, 1982

George A., Ng E , "*Orthogonal Reduction of Sparse Matrices to Upper Triangular from Using Householder Transformations*", SIAM J Sci Stat. Comp , vol 7, no.2, pp 460-477, 1986

Glimn A F , Stagg G W , "*Automatic Calculation of Load-Flows*", AIEE Trans on PAS, vol 76, pp.818-828, Oct. 1957

Gonen T , "*Modern Power System Analysis*", John Wiley & Sons, 1988

Gross G., Luini J.F., "*Effective Control of Convergence of the Newton Loadflow*", Proc. of PICA Conf., New Orleans, pp.41-48, 1975

Hopper M.J., "*Harwell Subroutine Library, A Catalogue of Subroutines (1973) Supplement no.2*", AERE Harwell, August 1977.

Huneault M., "*An Investigation of the Solution to the Optimal Power Flow Problem Incorporating Continuation Methods*", PH D thesis, Department of Elect. Engg., McGill Univ., Montreal, Canada, Aug 1988

Iwamoto S., Tamura Y., "*A Load Flow Calculation Method for Ill-Conditioned Power Systems*", IEEE Trans. on PAS, vol 100, no 4, pp.1736-1743, April 1981.

Iwamoto S., Tamura Y., "*A Fast Load Flow Method Retaining Nonlinearity*", IEEE Trans. on PAS, vol.97, pp 1586-1599, Sep /Oct 1978.

Iwamoto S., "*Discussion of [Keyhani et.al. 1989]*", IEEE Trans on PS, vol.4, no 2, pp.825-826, May 1989.

Keyhani A., "*Study of Fast Decoupled Load Flow Algorithms with Substantially Reduced Memory Requirements*", Elec Power Syst Research, vol.9, no 1, pp.1-9, June 1985.

Keyhani A., Abur A., Hao S , "*Evaluation of Power Flow Techniques for Personal Computers*", IEEE Trans. on PS, vol 4, no 2, pp 817-824, May

1989

Kron G , "<u>*Tensors for Circuits*</u>", Formerly entitled A Short Course in Tensor Analysis for Electrical Engineers, 2nd Ed., Dover Publications, Inc , New York, 1959

Kusic G L., "<u>*Computer-Aided Power Systems Analysis*</u>", Prentice-Hall, 1986

Lynch C A , "*Network Graphics Based Interactive Power System Analysis*", IEEE Trans on PAS, vol.98, no 4, pp.1140, Jul/Aug 1979

Masiello R D , Wollenberg B F., "*Comments on [Stott 1974]*", Proc. IEEE, vol 63, pp 713-714, April 1975.

Nagendra Rao P S , Prakasa Rao K.S., Nanda J , "*A Exact Fast Load Flow Method Including Second Order Terms in Rectangular Coordinates*", IEEE Trans on PAS, vol 101, no 9, pp.3261-3268, 1982

Ramarao K A , "*Discussion of [Wu 1977]*", IEEE Trans on PAS, vol 96, pp 272-275, 1977

Rao L , "*Exact Second Order Load Flow*", Proc of the 6th PSCC Conf., Dermstadt, Germany, vol 2, pp 711-718, 1978

Sachdev M S., Medicherla K P T., "*A Second Order Load Flow Technique*", IEEE Trans on PAS, vol 95, no 1, pp 189-197, 1977.

Sasson A.M., Snyder W., Flam M., "*Comments on [Stott 1974]*", Proc IEEE, vol.63, pp.713-714, April 1975.

Sasson A.M., "*Nonlinear Programming Solutions for the Load-Flow, Minimum-Loss, and Economic Dispatching Problems*", IEEE Trans. on PAS, vol.88, pp.399-409, April 1969

Sato N., Tinney W.F., "*Techniques for Exploring the Sparsity of the Network Admittance Matrix*", AIEE Trans , pt 3, vol 82, pp 944-949, 1963.

Schnabel R.B., Frank P D., "*Tensor Methods for Nonlinear Equations*", SIAM Journal on Numerical Analysis, vol.21, no 5, pp 815-843, Oct 1984

Stott B , "*Effective Starting Process for Newton-Raphson Load Flows*", Proc. IEE, vol 118, pp 983-987, Aug 1971.

Stott B., Alsac O , "*Fast Decoupled Load Flow*", IEEE Trans on PAS, vol 93, no. , pp.859-867, May/June 1974

Stott B., '*Review of Load-Flow Calculation Methods*", Proc of IEEE, vol 62, no 7, pp 916-929, July 1974

Stott B , Alsac O , "*Discussion of [Behnam-Guilani 1987]*", IEEE Trans on PS, vol.3, no 2, pp 740, May 1988

Strang G , "*Linear Algebra and Its Applications*", 2nd Edition, Academic Press, New York, 1980.

Tinney W F , Hart C.E., "*Power Flow Solution by Newton's Method*", IEEE Trans. on PAS, vol.86, no.11, pp.1449-1456, Nov. 1967

Tinney W.F , Walker J.W , "*Direct Sulotions of Sparse Network Equations by Optimally Ordered Triangular Factorization*", Proc. IEE, vol.55, pp 1801-1809, Nov 1967

Tripathy S C , Durga Prasad G., Malik O P , Hope G.S , "*Load flow Solutions for Ill-Conditioned Power Systems by A Newton-Like Method*", IEEE Trans on PAS, vol.101, no.10, pp 3648-3657, Oct 1982.

Van Ness J E , "*Iteration Method for Digital Load Flow Studies*", AIEE Trans on PAS, vol 78, pp.583-588, Aug 1959

Van Ness J E , Griffin J H , "*Elimination Methods for Load Flow Studies*", AIEE Trans. on PAS, vol 80, pp.299-304, June 1961

Wallach Y , Konrad V , "*Faster Algorithms for State Estimation*", IEEE PES Winter Power Meeting, New York, 1979

Wallach Y , "*Calculations & Programs for Power System Networks*", Prentice-Hall, 1986

Ward J B , Hale H W , "*Digital Computer Solution of Power Flow*

Problems", AIEE Trans on PAS, vol.75, pp 398-404, June 1956

Wu F.F., "*Theoretical Study of the Fast Decoupled Load Flow*", IEEE Trans. on PAS, vol.96, pp 268-275, 1977.

Xi'an Jioa-tong Univ. etc., "*Power System Calculations*", Applications of digital computers, Hydroelectric Power Publisher, Beijing, China, 1978.

Zollenkopf K., "*Load-Flow Calculation Using Loss Minimization Techniques*", Proc. IEE, vol 115(1), pp 121-127, 1968

Appendix 2.1

Notation for Gauss-Seidel Load Flow Equation

For convenience, equation (2.1) is rewritten as follows:

$$\bar{V}_i = \frac{1}{\bar{Y}_{ii}} \left\{ \frac{P_i^{sp} - jQ_i^{sp}}{\bar{V}_i^*} - \sum_{j \in i} \bar{Y}_{ij} \bar{V}_j \right\}$$

$$i = 1, \cdot, n.$$

with $\bar{V}_i$ bus complex voltages,

$\bar{Y}_{ii}$ diagonal elements of complex admittance matrix,

P_i^{sp} specified bus real power injections,

Q_i^{sp} specified bus reactive power injections,

Y_{ij} off-diagonal elements of complex admittance matrix,

Superscript * signifies the complex conjugate,

$j \in i$ signifies that bus j is connected to bus i.

Appendix 2.2

Notation for Newton-Raphson Load Flow Equation

For convenience, equation (2.2) is rewritten as follows

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} H & N \\ J & L \end{bmatrix} \begin{bmatrix} \Delta \delta \\ \Delta V \end{bmatrix}$$

with ΔP bus real power mismatch vector,

ΔQ bus reactive power mismatch vector,

$\Delta \delta$ correction vector of bus voltage phase angles,

ΔV correction vector of bus voltage magnitudes,

H, N, J, L submatrice in the load flow Jacobian matrix the corresponding equations are given in Section 3 2 2 2

APPENDIX 4.1

DATA FOR THE 11 BUS ILL-CONDITIONED SYSTEM

Line Data (in the form of Y matrix elements)

Row i	Column j	G_{ij}	B_{ij}
1	1	0 283	-2 785
1	10	-0.374	3 742
2	2	12 051	-33 089
2	3	0 0	6 494
2	4	-12.051	13 197
2	11	0 0	14 148
3	3	2.581	-10 282
3	5	-2.581	3 789
4	4	12.642	-74.081
4	5	0 0	2 177
4	6	0.0	56 689
4	7	-0 592	0 786
5	5	2 581	-5 889
6	6	0.0	-55 556
7	7	3.226	-4.304
7	8	-2.213	2 959
8	8	2.893	-5 468
8	9	-0.138	1 379
8	10	-0.851	1 163
9	9	0.104	-1 042
10	10	1.346	-6 110
11	11	0.0	-14 939

Bus Data (in the form of net bus powers):

Bus No	V(p.u.)	θ(deg.)	P(p u.)	Q(p.u.)
1			-0.158	-0.057
2			0.0	0 0
3			-0.128	-0 062
4			0.0	0 0
5			-0 165	-0 080
6			-0 090	-0.068
7			0 0	0 0
8			0 0	0 0
9			-0 026	-0 009
10			0 0	0 0
11	1.024	0.0	—	—

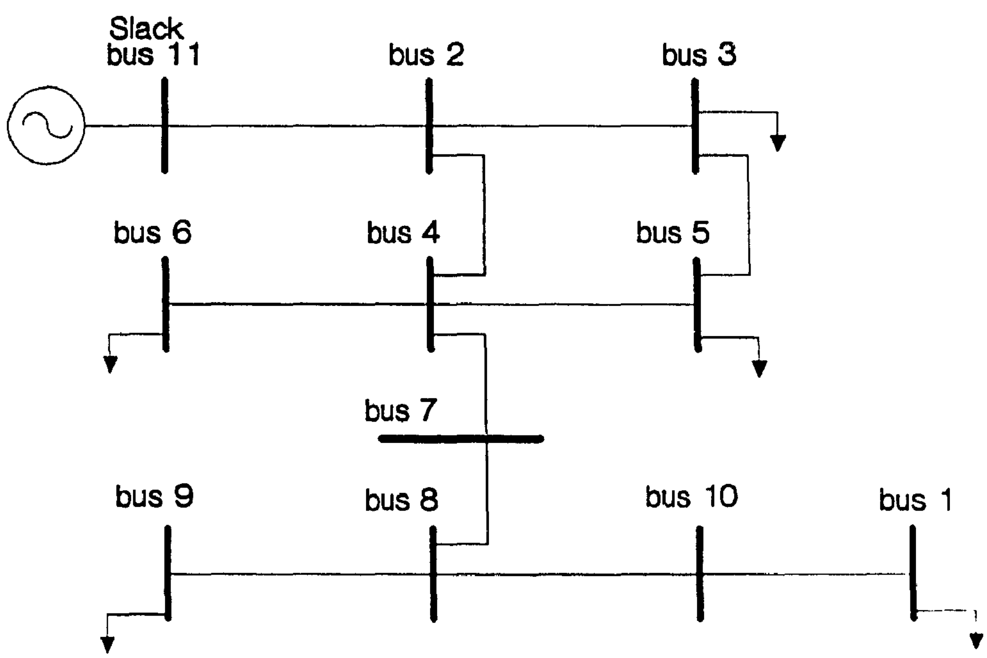

Fig. A4. 1. 1 Schematic Diagram of the 11 Bus Test System

APPENDIX 4.2

DATA FOR THE 13 BUS ILL-CONDITIONED SYSTEM

Line Data:

Line No	From	To	R(p.u.)	X(p u.)	B(p.u.)
1	1	12	0.0481	0.4590	0.246
2	1	13	0.0040	0 0850	0.0
3	2	3	0.0074	0 1430	0 436
4	2	10	0.0121	0.2330	0.712
5	2	13	0 0040	0.0947	0 0
6	3	11	0.0040	0 0947	0.0
7	4	10	0.0	0.1500	0 0
8	4	12	0.0090	0.1080	0.016
9	5	6	0.0075	0.1465	0.448
10	5	10	0.0	-0.1500	0.0
11	6	7	0.0086	0.1665	0.508
12	7	9	0.0	-0.1500	0.0
13	8	9	0.0105	0.2020	0.620

Power base = 1000 MVA

Transformer data:

Line No.	From	To	Tap setting
1	13	1	+ 5%
2	13	2	+10%
3	11	3	+10%

Bus Data:

Bus No.	V(p.u.)	θ(deg.)	P_G(MW)	Q_G(MVAR)	P_L(MW)	Q_L(MVAR)
1	1.000	0.0	0.0	0.0	0.0	0.0
2	1.000	0.0	0.0	0.0	0.0	0.0
3	1.000	0.0	0.0	0.0	0.0	0.0
4	1.063	0.0	0.0	0.0	0.0	0.0
5	1.000	0.0	0.0	0.0	0.0	0.0
6	1.000	0.0	0.0	0.0	50.00	32.00
7	1.000	0.0	0.0	0.0	50.00	30.00
8	0.943	0.0	500.00	–	0.0	0.0
9	1.100	0.0	0.0	–	0.0	0.0
10	1.100	0.0	0.0	–	0.0	0.0
11	1.000	0.0	0.0	–	0.0	0.0
12	1.037	0.0	500.00	–	50.00	30.00
13	1.000	0.0	–	–	1650.00	560.00

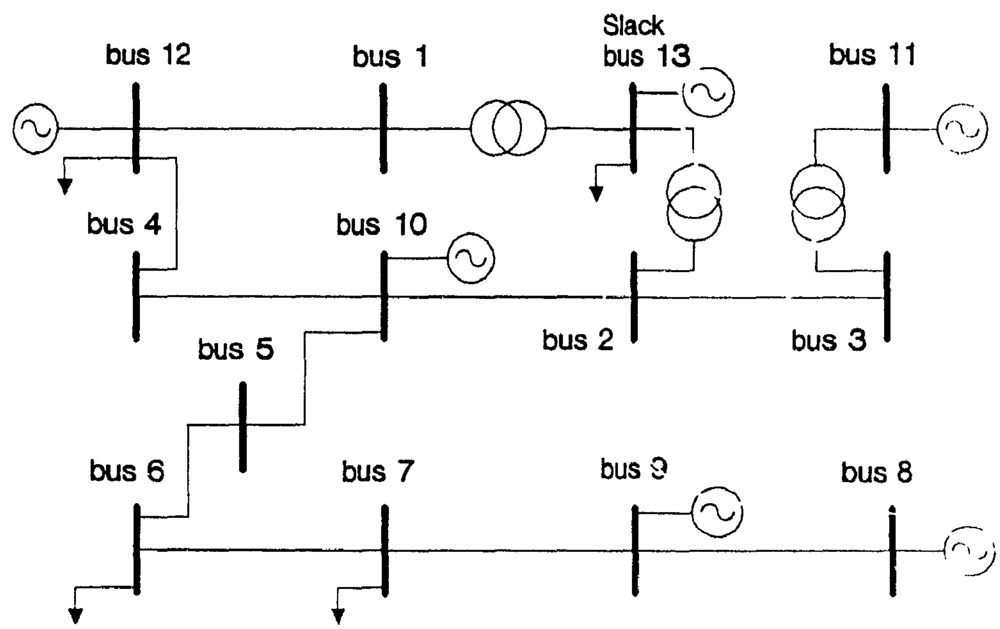

Fig. A4.2.1 Schematic Diagram of the 13 Bus Test System

APPENDIX 4.3

DATA FOR THE 20 BUS ILL-CONDITIONED SYSTEM

Line Data:

Line No	From	To	R(%)	X(%)	B(p u.)
1	1	20	0.50	5.00	0.024
2	2	8	0 50	5.00	0.067
3	2	16	0.0	5.00	0 0
4	2	17	60 00	60.00	0.0
5	3	5	20.00	20.00	0.0
6	3	20	0.11	1.52	0.0857
7	4	17	3.00	4.00	0.05
8	4	20	5.00	10.00	0.25
9	5	14	30.00	40.00	0.0
10	6	15	5.00	10.00	0.0
11	6	16	60.00	80.00	0.0
12	6	17	0.60	8.00	0.04
13	7	12	0.50	5.00	0.05
14	9	10	0.50	5.00	0.05
15	9	19	0.10	3.00	0.10
16	10	11	0.0	30 00	0.0
17	11	12	2.00	40.00	0.024
18	11	13	0.0	15.00	0 0
19	13	18	0.50	6.00	0.03
20	14	19	0.10	1.00	0.0
21	15	18	0.15	1.50	0 0
22	15	20	2.00	4.00	0 067

Bus Data:

Bus No.	P_G (MW)	P_L (MW)	Q_L (MVAR)
1	0	150	30
2	0	10	0
3	0	0	0
4	0	380	60
5	0	0	0
6	0	0	0
7	0	20	0
8	0	10	20
9	0	0	0
10	0	50	10
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	10
15	0	0	0
16	0	10	0
17	100	0	0
18	100	0	0
19	100	0	0
20	-	0	0

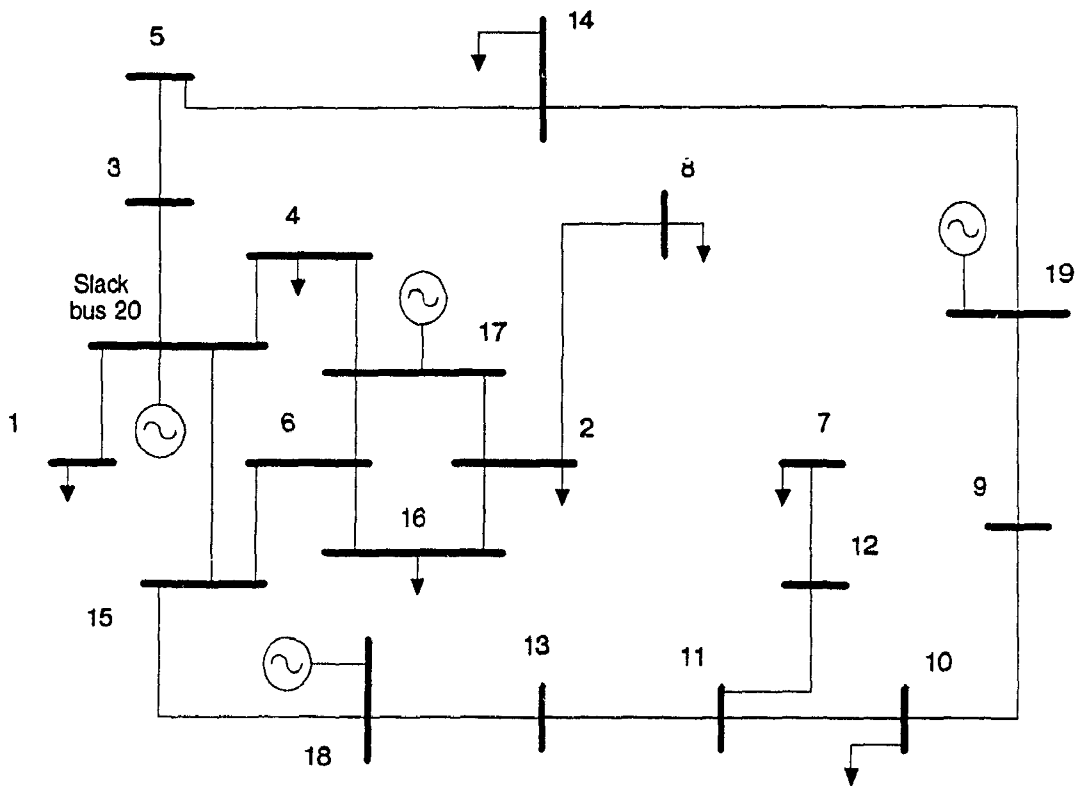

Fig. A4.3.1 Schematic Diagram of the 20 Bus Test System

APPENDIX 4.4

DATA FOR THE 43 BUS ILL-CONDITIONED SYSTEM

Line Data (in the form of Y matrix elements)

Row i	Column j	G_{ij}	B_{ij}
1	1	309 806	-408.029
1	14	0.0	15 400
1	39	-309.806	392.255
2	2	481.288	-1545.194
2	5	-277.195	873 583
2	6	-34 368	108.124
2	15	-169 726	534.322
2	43	0.0	30 609
3	3	0.0	-5.714
3	4	0 0	6.015
4	4	61.331	-69.160
4	13	-61.331	62.874
5	5	277 195	-916.892
5	7	0 0	21.277
5	8	0 0	20.513
6	6	34 368	-118 699
6	12	0.0	10 638
7	7	0.0	-20.000
8	8	452.840	-482.861
8	9	-288 938	295 777
8	23	-163.902	167.191
9	9	300.983	-317.044
9	10	-12.045	12 342
9	16	0.0	8 796
10	10	12.045	-20 855
10	11	0.0	2.857
10	17	0.0	5.714
11	11	0.0	-2.857
12	12	0 0	-10 000
13	13	92 381	-100 709
13	18	0 0	6 015
13	25	-31 050	31 640
14	14	0 0	-15 015
15	15	340 398	-916 783
15	19	0 0	8.649
15	20	0 0	15 791
15	28	-170 673	357 003
16	16	0 0	-8.576
17	17	0 0	-5 714
18	18	0 0	-5 714

Line Data (cont.):

Row i	Column j	G_{ij}	B_{ij}
19	19	164.292	-280.783
19	22	-164.292	272 805
20	20	0.0	-15.002
21	21	104.312	-143.609
21	24	0.0	9.267
21	29	-104.312	133.623
22	22	164.292	-282 281
22	26	0.0	9.023
23	23	321.579	-328 810
23	29	-157 677	161.760
24	24	0 0	-8.572
25	25	87 150	-106 814
25	27	0.0	9 023
25	29	-56.100	65.824
26	26	0.0	-8.572
27	27	0.0	-8.572
28	28	373.447	-612.837
28	39	-202.775	256.136
29	29	318.089	-372.311
29	30	0.0	3 766
29	37	0.0	7.895
30	30	125.789	-524 464
30	32	0 0	30 769
30	38	0.0	4.131
30	40	-125.789	485.547
31	31	0.0	-13.038
31	37	0.0	13.038
32	32	0 0	-30.769
33	33	0.0	-3.320
33	38	0 0	3.320
34	34	0.0	-7 365
34	38	0 0	6.852
35	35	0 0	-6 180
35	38	0 0	6.180
36	36	0.0	-2 703
36	38	0.0	2 703
37	37	0 0	-21 348
38	38	0.0	-22 398
39	39	512.581	-663 260
39	41	0.0	15 015
40	40	125 789	-508 837
40	42	0.0	21 622
41	41	0.0	-15 015
42	42	0 0	-20 000
43	43	0 0	-30 609

Bus Data (in the form of net bus powers):

Bus No.	V(p.u.)	θ(deg)	P(p u)	Q(p u.)
1			0.0	0 0
2			0.0	0.0
3			-0 16	-0 12
4			0.0	0 0
5			-0 53	-0 40
6			0 0	0 0
7			-1 60	-1 20
8			0 0	0 0
9			0.0	0 0
10			0.0	0 0
11			0.0	0 0
12			-0 80	-0 60
13			0 0	0 0
14			-0.80	-0 60
15			0 0	0 0
16			-0 64	-0 48
17			0 0	0 0
18			-0 24	-0 18
19			0 0	0 0
20			-0.88	-0 66
21			0 0	0 0
22			0 0	0 0
23			0 0	0 0
24			-0 64	-0 48
25			0 0	0 0
26			-0.80	-0 60
27			-0.32	-0 24
28			0 0	0 0
29			0 0	0 0
30			0 0	0 0
31			1 16	0 52
32			2 90	2 57
33			0 285	0 30
34			0 0	0 0
35			0 580	0 560
36			-0 050	0 030
37			0 0	0 0
38			-1 440	-1 02
39			0 0	0 0
40			0 0	0 0
41			-0 800	-0 300
42			-2 240	-1 680
43	1 136	0 0	—	—

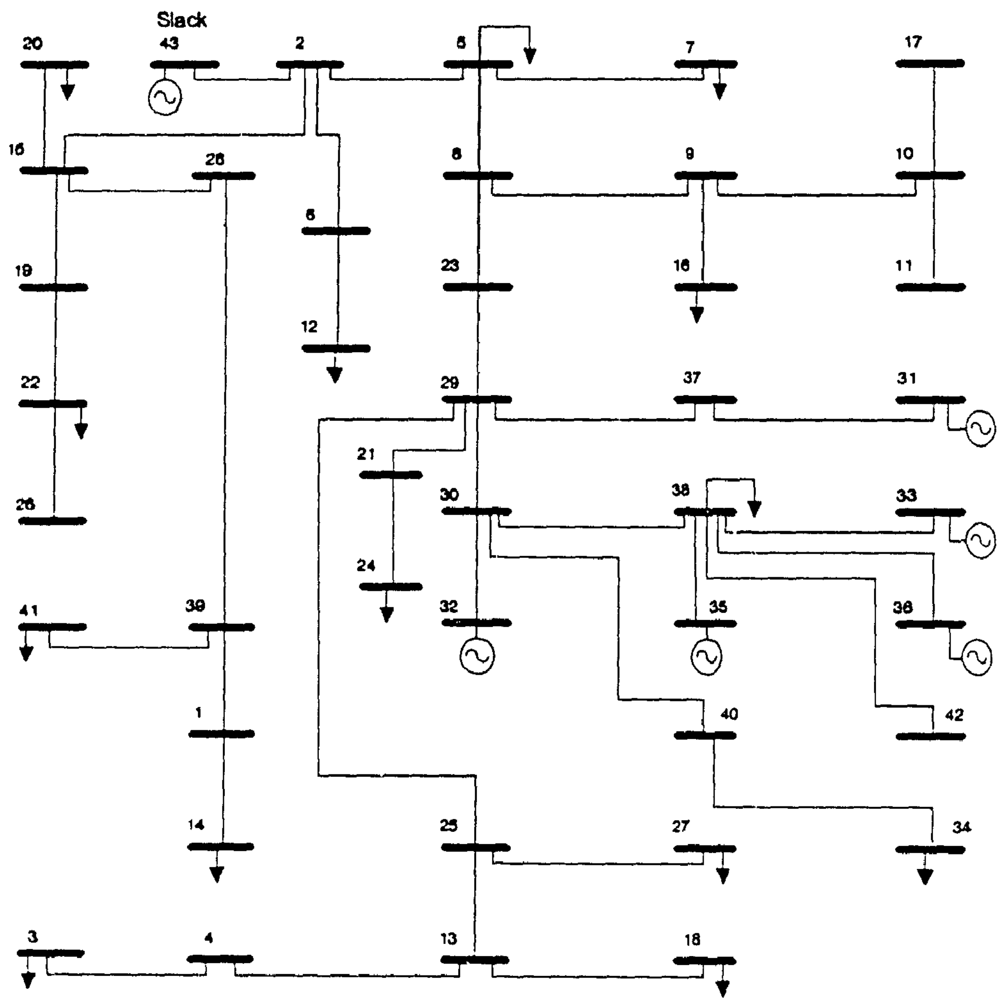

Fig A4 4 1 Schematic Diagram of the 43 Bus Test System