National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

ON THE APPLICATION OF PRE-CONDITIONED CONJUGATE GRADIENT ALGORITHMS TO POWER SYSTEM ANALYSIS PROBLEMS

by

Mohammad-Hossein Javidi-Dasht-Bayaz

B. Eng. (Tehran University, Tehran, Iran)
M.Sc. Eng. (Nagoya University, Nagoya, Japan)

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

Department of Electrical Engineering
McGill University
Montreal, Quebec, Canada
December, 1993

ISBN 0-315-94638-5

Canada

In the name of God,

the compassionate, the merciful

To my father in eternity,

one of his greatest wishes was to see his son graduate,

to my mother,

who is eagerly anticipating her son's return home

and

to my wonderful wife and children

Fatemeh, Ali and Reza,

without whose love and encouragement,
this work could not have been done.

ABSTRACT

Power system operation and planning relies heavily on computer simulation programs such as load flow, transient stability, contingency analysis, state estimation, short-circuit studies and optimal power flow. All of the above mentioned methodologies for planning and operation involve simulation programs which require the solution of numerous sets of simultaneous linear equations, Ax=b, whose coefficient matrices are in general very large and sparse. The main part of the computational effort involved in these algorithms is dedicated to solving such systems of linear equations.

This thesis investigates the properties of the coefficient matrix A that arises in power system analysis, as well as the application of more efficient alternative solution techniques for Ax=b which exploit these special properties. In particular, in this thesis, pre-conditioned conjugate gradient (PCG) methods have been applied and extensively tested for the first time to the solution of systems of linear equations arising in many power system operations and planning tasks.

In this vein, first, it is theoretically proven that some important power network coefficient matrices are positive definite and comply with the requirements for the convergence of the PCG method.

The PCG algorithm is then applied to the Fast Decoupled load flow and to the DC load flow. Its performance is numerically compared with a Frontal band-width direct solver (Frontal solver) as well as with a Sparspak solver (B5) with minimum degree ordering. The experimental results are based on a wide spectrum of power networks up to 5000 buses and about 10000 lines for two different types of networks: grid-type networks and star-type networks. These results demonstrate that the PCG method is clearly

superior to direct solvers for certain types of large power networks.

The performance of the PCG solver within other load flow algorithms is also numerically investigated.

A detailed investigation into the eigenvalue clustering effect of alternative pre-conditioners which utilize the intrinsic properties of power networks is also presented. In addition, the effect of their eigenvalue clusterings on the convergence of the PCG algorithm is analyzed and compared with that of the classical incomplete Cholesky pre-conditioner.

Furthermore, the usefulness of the PCG solvers is investigated for complex or indefinite power network matrices. A modified PCG method was applied to the IEEE test networks as well as to large synthetically generated networks (up to 6500 buses and 13000 lines) for the solution of systems of equations $Y x = b$, where Y is the complex admittance matrix. Comparison with direct solvers is provided.

Finally, a new technique is developed to synthetically generate realistic data sets which characterize power networks of arbitrary size and complexity. While these networks are randomly generated, the software allows the user to specify the system dimension, type of network, connectivity configurations and other network characteristics. This software was developed to overcome the difficulties associated with the collection of network data, especially for large scale systems.

RÉSUMÉ

L'exploitation et la planification des grands réseaux électriques fait appel à de nombreux outils de simulation, tels la répartition de puissance, la stabilité transitoire, l'analyse des contingences, l'estimation d'état, l'analyse de court circuits et la répartition optimale de puissance. Pour toutes ces méthodologies, on retrouve des algorithmes de résolution d'équations linéaires, $Ax = b$, au coeur des implantations numériques. En fait, la majeure partie du temps de calcul dans ces logiciels est consacré à la résolution des équations linéaires. Typiquement la matrice A est très grande et creuse.

Cette thèse cherche d'abord à établir quelques unes des propriétés fondamentales des matrices formées dans les problèmes de réseaux électriques. Elle propose ensuite des techniques de résolution, jusque-là inutilisées dans le domaine, pouvant mieux exploiter ces propriétés. En particulier, cette thèse fait l'essai de la méthode du gradient conjugué pré-conditionée (GCP) pour la résolution d'équations linéaires découlant de problèmes d'exploitation et de planification de réseau.

En un premier temps, on démontre que de nombreux types de matrices de réseaux rencontrent les exigences imposées par la méthode GCP, dont en particulier la condition d'être définie positive.

En un deuxième temps, l'algorithme GCP est implanté dans deux calculs de répartition de puissance, l'un complet utilisant la méthode découplée, l'autre simplifié utilisant la méthode dite à courant continu. Ses performances numériques sont comparées à celles de méthodes de résolution directes souvent utilisées dans le domaine, soit la méthode Frontale, et la méthode d'ordonnancement à degré minimal disponible dans le logiciel

Sparspak. Les matrices affectées aux essais proviennent de réseaux de differents types (grillage et étoile) et de differentes dimensions (allant jusqu'à 5000 barres et 10000 branches). Nos résultats démontrent la supériorité de la méthodologie GCP pour certains types de matrices de réseaux.

Nous avons également vérifié l'emploi de la méthodologie CGP dans d'autres algorithmes de répartition de puissance.

Cette thèse explore aussi les mérites de plusieurs modules pré-conditioneurs. Pour ce on évalue dans chaque cas, et pour plusieurs matrices de réseaux typiques, les regroupements des valeurs propres. Cela est intimement liée aux propriétés de convergence. On compare les performances de ces modules à celles du module préconditioneur classique de Cholesky.

De plus, un algorithme modifié CGP a servi à résoudre des systèmes d'équations indéfinies et des systèmes à coefficients complexes. Des matrices d'admittances complexes pour ces essais ont été tirées des réseaux tests de l'IEEE ainsi que de réseaux synthétiques de très grandes tailles (allant jusqu'à 6500 barres et 13000 branches).

Enfin, une nouvelle méthodologie est proposée pour synthétiser des réseaux fictifs à dimensions et à complexités arbitraires, comportant néanmoins des paramètres vraisemblables. Bien que la génération de ces réseaux soit aléatoire, l'usager contrôle le processus en spécifiant le type de réseau, ses dimensions, des éléments de sa connectivité, et d'autres paramètres. Cela permet d'alimenter librement les algorithmes de calcul tout en contourant les difficultés de collecter des données de réseaux.

ACKNOWLEDGEMENT

The author is indebted to many who have contributed to this research above all, I gratefully acknowledge the deep influence of my thesis supervisor, professor F.D. Galiana, for his expert guidance, continuous encouragement, his unflagging support, his friendship and his generosity which characterizes his special personality.

I also wish to express my sincere thanks to my co-supervisor, professor S. McFee, for the generous contribution of time, continuous encouragement, his invaluable comments and consultations and his friendship.

The counsel and advice of professor Silvester and the friendship of professor B.T. Ooi and Dr. M.H. Banakar are greatfully appreciated.

I wish to thank the departmental secretaries, specially Mrs. P. Hyland and R. Pinzarrone for their warm hospitality during my stay at McGill.

Special thanks are also extended to Dr. Maurice Huneault and my fellow Ph.D. students, in particular Lester Loud, Katia Campos de Almeida, Richard Marceau, Delly Oliveira and Mehrdad Kazerani for their warm companionship.

I wish to thank "Ferdowsi University of Mashhad, Iran" and "Ministry of Culture and Higher Education" for providing me the opportunity to pursue my study at McGill. I also acknowledge the financial support of NSERC.

Last, but certainly not least, I wish to express my sincere thanks to my wife, Fatemeh, and our two great children, Ali and Reza for their unwavering support, encouragement, patience and sacrifices during our stay in Montreal. We all missed many weekends to be together as a family because "I had work to do".

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER III

CHAPTER IV

CHAPTER V

LIST OF FIGURES

CHAPTER III

CHAPTER IV

CHAPTER V

CHAPTER VI

LIST OF ABBREVIATIONS

ABBREVIATION	MEANING
AC	Alternating Current
CG	Conjugate Gradient
CIC	Classical Incomplete Cholesky
CPU	Central Processing Unit
DC	Direct Current
DCLF	Direct Current Load Flow
DEIC	Dominant Element Incomplete Cholesky
DDIC	Diagonal Dominant Incomplete Cholesky
FDLF	Fast Decoupled Load Flow
IEEE	Institute of Electrical and Electronics Engineers
kW	Kilo Watts
Mb	Mega Byte
MHz	Mega Hertz
PCG	Pre-conditioned Conjugate Gradient
PQ	Power-Reactive Power (i.e. at a PQ bus, power and reactive power are specified in the power flow equations.)
PV	Power-Voltage (i.e. at a PV bus, power and voltages are specified in the power flow equations.)
RAM	Random Access Memory
SOR	Successive over-relaxation

LIST OF SYMBOLS

SYMBOL	MEANING
A_p	Pre-conditioned Coefficient Matrix
B	DC Load Flow Matrix
B′	Fast Decoupled Load Flow Matrix
B″	Fast Decoupled Load Flow Matrix
B_{ij}	B-Matrix Element Located in Row i and Column j of the Matrix
[cond(A)]	Condition Number of Matrix A
δ_{ij}	Phase Difference Between Voltages at Buses i and j
$d\delta$	Vector of Angle Mismatches
dP	Vector of Real Power Mismatches
dQ	Vector of Reactive Power Mismatches
dV	Vector of Voltage Mismatches
f	A Scaling Factor (Chapter IV)
Ji	Any of the Jacobian Sub-matrices (i = 1, 2, 3 or 4)
[Jac]	Jacobian Matrix
K	Pre-conditionting Matrix
L	Lower Triangular of the Cholesky Factor
l_{av}	The Average Value of the Non-zero Elements of the Matrix
m	Matrix Bandwidth
n	Network (Matrix) Dimension
nb	Number of Buses

nc	Number of Eigenvalues Located in the Largest Cluster
nd	Number of Diagonals Preserved in DDIC Pre-conditioning Matrix
nl	Number of Load Buses
nz	Number of Non-zero Elements of the Matrix
r	Vector of Right Hand Side Residuals
s	Average Number of Non-zero Elements of Matrix in Each Row
θ_{ij}	The Angle of the Line Impedance Phasor
w	A Weighting Factor (Chapter IV)
x_{ij}	Reactance of the Line Connecting Buses i and j

CHAPTER
ONE

INTRODUCTION

1.1 Background

The first power generation and transmission system was installed in 1882 at the Pearl Street Station in New York. It supplied a mere 30 kW to a number of electric incandescent street lamps. During the intervening decades, as a result of population growth and of the consumption-oriented culture of modern day societies, the demand for electric energy sharply increased. In the United States the annual electric energy production more than doubled every ten years from 1920 to 1960 [1].

In conjunction with load growth, there arose a need to increase the reliability of the supply and to reduce the high cost of spinning reserve, that is, the reserve generation which runs at almost no load ready to very rapidly respond to emergencies. One consequence of this need was the

interconnection of local power networks into very large pools. As a result of this trend, modern day power systems include thousands of interconnected transmission lines and hundreds of generating plants and substations.

The main responsibility of an electric power company is to produce, transmit and distribute energy to its customers at constant voltage and frequency, according to their needs, at a reasonable cost and high reliability. Typically, power requirements (amounts, duration and scheduling) are strongly influenced by the type of consumer applications, social and economic structure and locality and, therefore, demand can vary widely. In addition, power networks are exposed to frequent equipment failures which may significantly affect the ability of the network to meet its responsibilities.

Due to the high dependence of today's societies on electric energy, major interruptions such as the blackout in New York in 1977 can lead to significant economic losses and social disruption [2]. Society today does not accept such major power failures. Thus, power companies have been obliged to take measures to ensure a highly reliable continuity of supply during all possible disturbances of the networks.

So far, numerous investigations have been performed by the power industry to try to meet these difficult objectives. Electrical technology has progressed in a step-by-step fashion as a result of this research resulting in the development of sophisticated systematic methodologies for power system operation and planning [17, 24, 34]. These methods rely heavily on computer simulation programs such as load flow, transient stability, contingency analysis, state estimation, short-circuit studies, optimal power flow and others [23, 26, 33]. Because of the large dimension of modern power networks, their highly non-linear nature and the numerous possible disturbances and operating states, the computational burden associated with the planning and operation of power systems is huge. In fact, the high computational cost is a major obstacle in the development of more advanced operation and planning

methodologies. For this reason, an important research activity is being directed at reducing this computational effort.

Most of the above mentioned methodologies for planning and operation involve simulation programs which require the solution of numerous sets of simultaneous linear equations, $Ax=b$, whose coefficient matrices are in general very large and sparse. The size and complexity of the coefficient matrix is closely tied to the power network size and its interconnections. These systems of linear equations arise from the iterative solution of non-linear algebraic and differential equations [11-18]. The main part of the computational work involved in most of the above mentioned methodologies is dedicated to solving such linear systems of equations. Due to this fact, we have focused our investigation on the properties of the coefficient matrices that arise in power system analysis and the application of more efficient solution techniques. In particular, in this thesis, pre-conditioned conjugate gradient methods have been applied for the first time to large power system analysis problems.

1.2 The Present Thesis

The present thesis has investigated the application of pre-conditioned conjugate gradient algorithms to the solution of systems of linear equations arising in power system analysis problems. This required:

(a) detailed investigations of the properties of power network matrices,

(b) development of an algorithm to synthetically generate arbitrary representative power network data,

(c) numerous tests of the Incomplete Cholesky pre-conditioned conjugate gradient (PCG) algorithm on a wide spectrum of large scale power networks,

(d) an investigation of alternative pre-conditioning schemes exploiting power system network properties.

The following sections briefly overview the motivation and contents of the present thesis.

1.2.1 Motivation for thesis

Solving large, sparse systems of linear equations is at the heart of several power system problems. These systems of linear equations that arise in the planning, design and operation of power systems, are typically solved using direct methods that involve variations of Gaussian elimination.

It has been shown by Sato and Tinney [38], Carpentier [25] and Tinney *et al.* [26,50] that, through sparse programming and efficient ordering schemes, extremely fast direct method solutions can be computed with a minimum of memory requirements. Although these techniques are considered efficient for many applications, due to the increasing size and complexity of power networks, in cases involving repeated solutions such as security analysis, and in real time control [27], such direct solvers may still not be sufficiently fast.

The present thesis is concerned with the potential of iterative techniques in the solution of linear systems of equations that arise in many power system problems. More specifically, the main objective of this research has been to apply the semi-iterative pre-conditioned conjugate gradient (PCG) algorithm to these problems. The following enumerates the motivations underlying the present thesis:

i) The encouraging results of PCG in finite element analysis in electromagnetics where speed improvements of 100 to 1 compared with direct methods are not uncommon [28].

ii) The critical need to speed up large-scale power system analysis and simulation algorithms, such as security monitoring and control applications, specially in a real time environment.

iii) The fact that PCG methods had not been systematically evaluated in power system applications.

During this course of study a comparative investigation into the potential value of different pre-conditioning matrices in power applications was also conducted. This work was primarily motivated by:

iv) The impressive results of PCG with classical incomplete Cholesky pre-conditioning for the DC and Fast Decoupled load flows, for which a very high gain in speed is achieved compared with direct solvers [29].

v) The PCG performance gain is closely related to the eigenvalue clustering effect introduced by pre-conditioning [102]. Therefore, constructing more effective pre-conditioning matrices can reduce the number of iterations resulting in faster convergence.

vi) To our knowledge, the eigenvalue clustering of power network matrices using specially adapted pre-conditioning schemes had not been extensively investigated or exploited.

vii) The above studies were based on a comparative investigation of a wide spectrum of power networks up to 6500 buses and 13000 lines. Because of the difficulty of obtaining data from representative networks, we were motivated to develop an algorithm to synthetically create such networks.

An outline of this thesis is presented next.

1.2.2 Outline of thesis

1.2.2.1 Chapter I

Chapter I presents introductory background material, motivation for the thesis, a chapter by chapter outline of the thesis and a list of original contributions.

1.2.2.2 Chapter II

Commonly used power system studies where numerous large systems of linear equations must be solved are reviewed. These include load flow, transient stability, contingency analysis, state estimation and short-circuit analysis. In addition, recent advances for reducing the computation time of such problems are reviewed briefly.

Also, the principal general iterative techniques for the solution of systems of linear equations and their properties are reviewed. The conjugate gradient approach and its pre-conditioned form are described. A comparative investigation of the potential value of the pre-conditioned conjugate gradient algorithm versus direct solvers is performed.

1.2.2.3 Chapter III

In this chapter, first, it is theoretically proven that some important power network matrices are positive definite and comply with the requirements for the convergence of the PCG method.

Then, the Incomplete Cholesky PCG algorithm is applied to the Fast Decoupled load flow and to the DC load flow and numerically compared with a Frontal band-width based direct solver (Frontal solver) as well as with a

Sparspak solver (B5) with minimum degree ordering. The experimental results are based on a wide spectrum of power networks up to 5000 buses and about 10000 lines for two different types of networks: multi-block grid networks and multi-block star networks.

The performance of the PCG solver inside other load flow algorithms is also investigated.

1.2.2.4 Chapter IV

This chapter describes the results of a detailed investigation into the eigenvalue clustering effect of alternative pre-conditioners for power network matrices. In addition, the effect of their eigenvalue clusterings on the convergence of the PCG algorithm is analyzed and compared with that of the classical incomplete Cholesky pre-conditioner.

1.2.2.5 Chapter V

In this chapter, an investigation into the application of PCG to complex and indefinite power network matrices is presented. Modifications have been made to the PCG algorithm to apply it to complex admittance matrices. The modified PCG method has been applied to the IEEE test networks and synthetically generated networks of large sizes for the solution of systems of equations of the form $Y x = b$, where Y is the complex admittance matrix.

The computational cost of the new PCG algorithm is compared with that of a standard direct solver and the advantages of PCG in the solution of $Y V = I$ as arising in load flow and transient stability are discussed.

1.2.2.6 Chapter VI

This chapter presents a new technique that synthetically generates

realistic data for power networks of arbitrary size and complexity. While these networks are randomly generated, the software allows the user to specify the system dimension, type of network, connectivity configurations and other network characteristics.

This software was developed to overcome the difficulties associated with the collection of network data, especially for large scale systems.

1.2.2.7 CHAPTER VII

This chapter contains the concluding remarks, and recommendations for extending the scope of the present research and its future direction.

1.2.3 Original contributions

To the author's knowledge, the principal original contribution of this thesis has been the application and systematic evaluation of Pre-conditioned Conjugate Gradient (PCG) methods to power system problems.

This contribution includes the following parts:

i) The PCG method has been shown to be considerably more efficient than direct solvers in certain types of power system problems with positive definite real matrices arising from the DC and Fast Decoupled load flows.

ii) Extensive numerical tests were carried out to investigate the effect of network size and topology on the relative performance of direct and PCG methods.

iii) The eigenvalue clustering effect of different pre-conditioning matrices and their effect on the convergence of the PCG algorithm has been

investigated in power system problems. A more effective pre-conditioner (Dominant Element Incomplete Cholesky) which exploits intrinsic properties of power networks has been discovered to be more effective than the classical incomplete Cholesky.

iv) The PCG method has been shown, through extensive numerical tests, to have advantages over direct solvers not just for real positive definite matrices, but also for complex network matrices such as the admittance matrix.

v) A new technique has been presented that designs networks and synthetically generates realistic data for power networks of arbitrary size and complexity.

CHAPTER
TWO

COMMON POWER SYSTEM ANALYSIS PROBLEMS AND RECENT ADVANCES IN SOLUTION TECHNIQUES

2.1 Introduction

The objective of this chapter is to give the reader an overview of the basic power system analysis problems and to demonstrate that the fast solution of such problems depends very strongly on efficient techniques for solving systems of linear equations.

Thus, commonly encountered power system problems which require repeated solutions of large systems of linear equations are reviewed. In addition, recent advances in the reduction of computational and memory requirements relative to direct solvers such as sparse programming, ordering schemes, matrix partitioning, compensation methods and parallel processing are summarized.

Then, general properties of iterative techniques for the solution of large sparse systems of linear equations as well as the semi-iterative conjugate gradient (CG) algorithm are described and compared with those of direct solvers. Since the main objective of this thesis is the application of the pre-conditioned conjugate gradient (PCG) method to power system problems, special attention is directed toward the conjugate gradient (CG) and pre-conditioned conjugate gradient (PCG) algorithms as alternatives to direct solvers.

2.2 Load Flow Techniques

Load flow calculations are fundamental to most power system problems [3-12]. These calculations which characterize the sinusoidal steady-state behaviour of power networks are repeatedly carried out as part of system planning, operational planning, optimization, contingency analysis and transient stability studies. The load flow computation is the most time consuming part of these studies.

The trend today in programs using load flow algorithms is to analyze larger networks, to solve numerous repeated load flow cases, and do all this in real time. To speed up the solution algorithms for the power flow problem therefore becomes essential.

Basic to the understanding of the load flow problem is a mathematical model of the network, which can be written based on Kirchhoff's current law,

$$I = Y V \tag{2.1}$$

where, I is the vector of injected currents, V is the vector of bus voltages and Y represents the complex network admittance matrix.

However, in a power flow problem, the power injections at different buses are known instead of currents and, therefore, the steady state equations are often expressed in terms of powers and voltages. Thus, to establish the load flow equations, the complex power delivered to bus i is expressed as follows,

$$S_i = V_i I_i^* \tag{2.2}$$

or,

$$P_i + jQ_i = V_i \left[\sum_{j=1}^{n} Y_{ij} V_j\right]^* \tag{2.3}$$

where,

$$\begin{aligned} V_j &= V_j e^{j\delta_j} \\ Y_{ij} &= Y_{ij} e^{j\theta_{ij}} \end{aligned} \tag{2.4}$$

Now, equation 2.3 can readily be separated into real and imaginary parts as,

$$P_i = V_i \sum_{j=1}^{n} Y_{ij} V_j \cos(\delta_i - \delta_j - \theta_{ij}) \tag{2.5}$$

$$Q_i = V_i \sum_{j=1}^{n} Y_{ij} V_j \sin(\delta_i - \delta_j - \theta_{ij}) \tag{2.6}$$

Finally, we are faced with the problem of obtaining a numerical solution to the above systems of nonlinear algebraic equations for the unknown voltage components.

Several alternative methods have been proposed in the literature for the solution of the power flow equations. All of these different approaches

essentially start with an initial guess for the complex nodal voltages, which in turn is used in conjunction with the load flow equations to compute a new and better estimate of the solution. This process is repeated in an iterative fashion until the equations are satisfied.

Among the various solution algorithms, the methods based on the Newton-Raphson approach [8] are the dominant ones. These numerical methods are generally at their most efficient when they take advantage of the physical properties of the power network being solved. This resulted in several efficient and generally reliable load flow solution techniques such as the Decoupled and Fast Decoupled load flow (FDLF) algorithms [11, 12, 31]. Some of the properties of these Newton-like algorithms are described in the following sections and a brief comparison of the merits and demerits of these methods based on the available literature is presented.

2.2.1 Newton-Raphson load flow algorithm

The Newton-Raphson method for the solution of equations 2.5 and 2.6 requires repeated solutions of systems of linear equations of the form,

$$\begin{bmatrix} J1 & J2 \\ J3 & J4 \end{bmatrix}_i \begin{bmatrix} \Delta\delta \\ \Delta V \end{bmatrix}_i = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix}_i \tag{2.7}$$

where the sub-matrices making up the Jacobian matrix are given by,

$$\begin{aligned} J1 &= \frac{\partial P}{\partial \delta} \\ J2 &= \frac{\partial P}{\partial V} \\ J3 &= \frac{\partial Q}{\partial \delta} \\ J4 &= \frac{\partial Q}{\partial V} \end{aligned} \tag{2.8}$$

and where P, Q, δ, and V respectively represent vectors of real power

injections, reactive power injections, bus voltage phase angles and magnitudes. The index i represents the iteration number. After each iteration δ and V vectors are updated by adding the correction vectors $\Delta\delta$ and ΔV. The vectors of equation mismatches ΔP and ΔQ represent the difference between the specified and the calculated power injections.

The coefficient matrix in this equation is the Jacobian matrix which is a function of the latest voltage solution. The Newton-Raphson method is quite robust and converges relatively fast as measured by the number of repeated solutions. For power flow studies that do not call for special adjustments such as PV-PQ bus type switching, convergence to an acceptable accuracy will usually be obtained in less than 5 iterations for large as well as small systems [8, 32]. In their extensive studies Stagg and El-Abiad [33] have concluded that the required number of iterations for convergence of the Newton-Raphson method is independent of system size for well-behaved problems.

It is noteworthy to mention that the dimension of the system of equations 2.7 is equal to nb+nl-1, where nb and nl represent the number of buses and the number of load buses respectively. Equation 2.7 is partitioned into four blocks, J1, J2, J3 and J4. Based on equation 2.8, the matrix elements in each block of equation 2.7 can be written as below :

Non-diagonal elements for each block:

$$
\begin{aligned}
J1_{ij} &= \frac{\partial P_i}{\partial \delta_j} = V_i\, Y_{ij}\, V_j \sin(\delta_i-\delta_j-\theta_{ij}) \\
J2_{ij} &= \frac{\partial P_i}{\partial V_j} = V_i\, Y_{ij} \cos(\delta_i-\delta_j-\theta_{ij}) \\
J3_{ij} &= \frac{\partial Q_i}{\partial \delta_j} = -V_i\, Y_{ij}\, V_j \cos(\delta_i-\delta_j-\theta_{ij}) \\
J4_{ij} &= \frac{\partial Q_i}{\partial V_j} = V_i\, Y_{ij} \sin(\delta_i-\delta_j-\theta_{ij})
\end{aligned}
\qquad (2.9)
$$

Diagonal elements for each block:

$$
\begin{aligned}
J1_{ii} &= \frac{\partial P_i}{\partial \delta_i} = V_i \sum_{k=1}^{n} Y_{ik} V_j \sin(\delta_i - \delta_k - \theta_{ik}) \qquad k \neq i \\
J2_{ii} &= \frac{\partial P_i}{\partial V_i} = V_i Y_{ii} \cos(\theta_{ii}) + \sum_{k=1}^{n} Y_{ik} V_k \cos(\delta_i - \delta_k - \theta_{ik}) \\
J3_{ii} &= \frac{\partial Q_i}{\partial \delta_i} = V_i \sum_{k=1}^{n} Y_{ik} V_k \cos(\delta_i - \delta_k - \theta_{ik}) \\
J4_{ii} &= \frac{\partial Q_i}{\partial V_i} = V_i Y_{ii} \sin(\theta_{ii}) + \sum_{k=1}^{n} Y_{ik} V_k \sin(\delta_i - \delta_j - \theta_{ij})
\end{aligned}
\tag{2.10}
$$

It is observed from equations 2.9 and 2.10 that the Jacobian in its original form is not symmetric and has almost the same sparsity structure as the Y-matrix. In a real system, on the average, each bus may be connected to about three or four of the remaining buses. This means that, on the average, four or five of the elements in each row of the Y-matrix including the diagonal elements will be non-zero. This implies that for large systems the Y-matrix is very sparse. For example, for a typical 1000-bus power network the sparsity of the Y-matrix is about 99 percent.

The Newton-Raphson method has the advantages of being reliable and offering fast convergence, however, due to the need to update and factorize the Jacobian matrix at each iteration (when using direct methods to solve equation 2.7), the computational time of the solution increases rapidly with system dimension. Due to this fact, certain modifications have been applied to the Newton-Raphson method to reduce computational effort without sacrificing its strong convergence properties. One example of this is to maintain the same Jacobian matrix throughout all the iteration steps. Other, more elaborate modifications are discussed below.

2.2.2 Decoupled load flow algorithm

It is known that for a banded or Frontal Gaussian elimination linear equation solver, with only the active part of the matrix stored in memory, the total operation count and the storage requirements are respectively $O(nm^2)$ and $O(m^2)$, where n is the system dimension and m is the matrix bandwidth (the average width of non-zero elements about the diagonal) [46, 60]. This indicates that as the dimension of the system increases, the computational effort will grow as $O(n^{\alpha})$ where $1 < \alpha < 2$. Therefore the solution time of two decoupled systems of linear equations with dimensions n_1 and n_2 is much smaller than the solution time of one system of linear equations of size n, where n is equal to the sum of n_1 and n_2.

The decoupled load flow takes advantage of this property. It is based on a simplified Newton algorithm by exploiting the real power/angle and reactive power/voltage decoupling principle [11,32]. This principle states that, generally, for small changes in the magnitudes of the bus voltages, the real power bus injections do not change significantly. On the other hand, small changes in the phase angles do not affect the reactive power injections appreciably. This can also be concluded from equations 2.6 due to the fact that, in power networks, θ_{ij} is usually close to 90 degrees and the angle difference between the two end buses of a transmission line is generally small ($<30°$). The decoupling principle can be interpreted to mean that the elements of the sub-matrices J2 and J3 are numerically much smaller than those of J1 and J4. Then, approximating J2 and J3 with zero matrices of the same dimension, equation 2.7 reduces into two smaller systems of linear equations 2.11 and 2.12,

$$J_1 \cdot \Delta\delta = \Delta P \qquad (2.11)$$

$$J_4 \cdot \Delta V = \Delta Q \qquad (2.12)$$

This modification can significantly reduce the computational time involved in each iteration step, however in some cases due to this approximation, the number of iterations required for convergence increases.

In spite of the decoupling simplification, the sub-matrices J1 and J4 are non-symmetric matrices that need to be updated and refactored at each iteration step. A further simplification of the Jacobian matrix results in the Fast Decoupled load flow algorithm and is discussed next.

2.2.3 Fast Decoupled load flow algorithm

The Fast Decoupled load flow is the most commonly used algorithm for solving the non-linear AC load flow [12]. It is based on a simplified Decoupled load flow algorithm with a constant Jacobian matrix derived by exploiting the fact that in typical power networks, the resistive portions of the line impedances are small relative to reactive portions and thus can be neglected [12]. This fact also allows the programmer to substitute θ_{ij} by 90 degrees. In addition, if the power flows are small relative to the maximum power carrying line capacity, then $\sin(\delta_i-\delta_j)$ is approximated by $\delta_i-\delta_j$.

Thus, in each iteration of the Fast Decoupled load flow, two matrix equations need to be solved for $d\delta$ and dV,

$$B' \cdot \Delta\delta = \frac{\Delta P}{V} \qquad (2.13)$$

$$B'' \cdot \Delta V = \frac{\Delta Q}{V} \qquad (2.14)$$

In these equations B' and B" are both real, sparse and constant matrices. The equation mismatches, dP and dQ are updated at every iteration. The matrix B" is a principal sub-matrix of B'.

In this approach more iterations may be required for convergence compared to full Newton approach. However, due to the fact that B' and B" are constant, only one factorization will be required for repeated solutions of each of these two systems of linear equations. This is a very significant property which has made the Fast Decoupled method the dominant approach in power system studies.

2.2.4 DC load flow

The DC load flow is a widely implemented approximation of the general AC load flow. The DC load flow is used to analyze highly meshed power networks with short lines, low R to X ratios and angular differences between adjacent buses not exceeding 30 degrees. It is formulated by,

$$P = B\,\delta \tag{2.15}$$

where, P is the vector of net real power bus injections at all buses except for the reference bus, while B is the DC load flow network matrix [34].

2.2.5 Y V = I based load flow algorithm

An iterative technique using the complex admittance matrix is another approach to the solution of the power flow equations. The simplest way to implement this algorithm is to approximate the loads as constant impedances and the generator terminal currents $I_r + j\ I_i$ as given node currents and to adjust them iteratively until solution convergence is achieved. Experimental investigations on this approach suggest, however, that this method frequently fails to converge [13]. The convergence of the algorithm can be improved by adding some fictitious slack buses behind the generator impedances [13]. This method is frequently used for solving the steady state equations in transient stability programs [13, 14].

2.3 Transient Stability Analysis

Transient stability analysis deals with two simultaneous sets of equations that must be solved repeatedly as time evolves [13-15]. The first describes the steady state behaviour of the network, and includes steady state models of the loads and algebraic equations of the generator,

$$g\ (X,W)\ =\ 0 \tag{2.16}$$

while, the second set (differential equations) characterizes the dynamic behaviour of the machines and their control circuits,

$$\frac{dW}{dt}\ =\ f\ (X,W,t) \tag{2.17}$$

There are different solution methods for the transient stability problem, however, most approaches require iterative solutions of [13, 119],

$$[\ Y\ +\ Y_c\ +\ Y_g\ ]\ .\ V\ =\ I_r \tag{2.18}$$

where, Y is the network admittance matrix, while, Y_c and Y_g represent the loads and generators characteristics. The summation of these three matrices has a much larger dimension than the network size, being closely tied to the complexity of the associated models representing generators and loads [14] (for example 4800 by 4800 for the Hydro-Quebec 700-bus network).

2.4 Contingency Analysis and Contingency Ranking

Security is an important feature that should be considered in the design and operation of power systems. This implies that transmission networks must

be designed and operated such that in case of the outage of one of the network components, the remaining components should be able to continue operating without overloading. This is necessary since, although the operator can remotely control circuit breakers and other components, network changes can be so fast that he or she may not be able to respond to an outage in time.

In order to design and operate power systems securely, contingency analysis algorithms of different forms have been developed during the past two decades [16-20]. These programs are based on the fast but approximate simulation of the existing operating state and of the effects of possible outages. They allow operators to predict what should be done in case of potential emergencies to maintain system security.

Contingency analysis is mainly carried out by running fast load flow algorithms such as the Fast Decoupled and DC load flow. However, because of the large number of possible changes in a network, security analysis is still a very time-consuming task. To reduce this computational burden, the list of probable contingencies is typically ranked according to some simple-to-calculate performance indices [16,17], and only the most severe ones are analyzed in detail.

It is obvious that once again, in security analysis, we are confronted with the difficulties of solving numerous large systems of linear equations. Due to time constraints and to a wider spectrum of possible contingencies, these difficulties are even more pronounced when dealing with on-line applications.

2.5 State Estimation

Power companies have always been faced with the need to monitor and control the operating conditions of their networks in real time. Nowadays,

most of the control and dispatch centers are equipped with systems which include measurement and transmission of critical data to dispatch centers every few seconds.

The transmitted data has the following properties: (1) A direct measurement of every important quantity is not always possible, (2) there are more measurements than quantities to be monitored (redundancy), (3) due to measurement errors and telemetry problems, the data can be inaccurate or even wrong.

State estimation algorithms have therefore been developed to minimize the effect of measurement errors and telemetry problems and to produce a complete and reliable set of data for security analysis. These algorithms rely on the solution of repeated systems of linear equations [21, 22, 78].

2.6 Short-Circuit Analysis

Short-circuits can occur in power systems whenever there is a failure in the insulation of some part of the network, due to overvoltages, or as a result of mechanical failures. Thus, short-circuit calculations must be performed for selecting, setting and coordinating protective equipment such as circuit breakers, fuses, relays and instrument transformers. These computations rely on utilizing the short circuit algorithms in which, the bus admittance matrix is the key to calculating the fault currents [23] :

$$Y\,V = I \tag{2.19}$$

Thus, again, we must deal with the solution of large systems of linear equations (this time in the complex domain).

2.7 Solution Techniques and Recent Advances in Power System Analysis

It was demonstrated in previous sections that almost all power system analysis algorithms rely heavily on the repeated solutions of large sparse systems of linear equations. These solutions take up the bulk of the overall computational effort [26, 40]. Therefore, any attempt at making power system analysis programs more efficient should first be directed at improving the linear solver part.

Techniques to solve systems of linear equations may generally be classified as direct, involving a fixed number of arithmetic operations, and indirect or iterative methods, involving the repetition of certain steps. This repetition is continued until the required accuracy is achieved.

Iterative methods, such as the classical successive over-relaxation techniques [91, 92], are usually easy to program and need less storage but are, generally, less reliable and less efficient. Thus, for the most part, these techniques have been replaced by direct solvers in power system analysis [40].

2.7.1 Direct solvers

Direct solvers for the solution of Ax=b [46] are, typically, variations of Gaussian elimination. They yield the exact solution in a finite number of iterations and require $O(n^2)$ storage locations and $O(n^3)$ arithmetic operations without sparsity exploitation. These solvers essentially utilize some form of factorization of A into the product of a lower triangular (L) and an upper triangular matrix (U),

$$A = L\,U \tag{2.20}$$

Then, the solution, x, is obtained by a forward and a backward substitution,

$$L\,y = b$$
$$U\,x = y \qquad (2.21)$$

Direct solvers have been substantially improved over the years by the incorporation of some special advanced developments namely, sparse programming, ordering algorithms, matrix partitioning, compensation methods and parallel processing. Nevertheless, the computation time of direct methods can still be prohibitively high in cases involving very large systems, numerous repeated solutions, and in highly meshed networks that are difficult to order efficiently.

2.7.1.1 Sparse programming

It is well known that the solution of a system of n linear equations involves dimensioning an n by n coefficient matrix with n^2 storage requirements if the matrix is real and sparsity is not exploited. Obviously, even powerful machines will have difficulties in solving simple problems if the dimension is larger than a thousand. Thus, the symmetry and sparsity of the coefficient matrix must be considered.

Some techniques for saving sparse matrices indicate and store each matrix element with its column number and row number. Thus, the storage requirement for the data is about three times the number of entries. It is also clear that these data have to be unpacked for every operation which in turn requires some computer time. Therefore, it may not be advantageous to use sparse matrix techniques unless the density of non-zero elements is considerably low. However, for most large power network applications, sparsity is usually greater than 95% so that it pays substantially to exploit sparsity.

Due to the significant gain in computer storage for large sparse matrices, sparsity oriented programming is a standard feature in most industrial applications today, and much research has been devoted to this area during the past decades [35-51]. These investigations have resulted in the introduction of various kinds of storage schemes which differ in the way zeros are exploited [35-43]. Some of these schemes explicitly store some zeros to get a simpler storage scheme, while other schemes sacrifice simplicity in favour of fewer storage locations. The choice of storage scheme affects the solution strategies and has obvious impact on the memory requirements and execution time. Therefore, it can be considered as a tool for reducing computer storage, computer execution time or a combination of both objectives.

Based on the high degree of sparsity in typical power system matrices [47-49], sparsity programming has been extensively applied to power system algorithms as in other fields. Improvements in sparsity techniques in the power area are mainly due to Edelman, Sato, Walker, Tinney and Ogbuobiri [38, 39, 40, 41].

In their later work Tinney *et al.* have extended sparsity exploitation to vectors [50]. They have shown that sparse vector methods are very useful techniques for solving systems of linear equations if the right hand side vector is sparse or only a few elements in the unknown vector have to be determined. This aspect has been successfully exploited for on-line power network security analysis [51].

2.7.1.2 Ordering algorithms

The computation time for the direct solution of a sparse system of linear equations essentially depends on the total number of operations during the triangularization of the coefficient matrix. This includes the number of operations relating to non-zero elements of the coefficient matrix as well as those related to additional non-zero terms introduced in the reduction to

triangular form (fill-ins). It can be shown that, generally, this operation count varies with the sequence in which the rows of the coefficient matrix are processed. The order in which the Gaussian elimination is performed on sparse matrices affects the total number of these newly introduced non-zero elements. Thus, it is clear that the solution time will be essentially dependent on the ordering scheme used for the elimination. Depending on the coefficient matrix structure, some orderings can lead to a dramatic reduction in the amount of fill-ins and hence in the total computation time. However, the task of finding the best (or more realistically a good) ordering for a sparse solver is not that easy. This is because it is very difficult to define what is meant by best, and only heuristic algorithms are computationally feasible for large sparse systems [52]. Some orderings may be very efficient but very costly to compute, while another algorithm allowing more fill-ins but fewer computations in the ordering algorithm itself may be more economical overall. George and Liu, in their intensive study [37, 46], have shown that the execution times required to implement different orderings can vary dramatically. For these reasons, detailed investigations have been focused on the performance of different ordering schemes and many practical algorithms have been developed [53-70]. However, each ordering strategy has its own objectives and thus it is very difficult to choose a single best ordering. Only after extensive analysis and computational experiments is it possible to identify those suitable for a special class of matrices while rejecting others.

There are generally two basic objectives for ordering algorithms. One focuses on strategies to control numerical accuracy through pivoting schemes, while the other aims at conservation of the matrix sparsity. In power system problems, due to the nature of the network matrices and to the numerical accuracy of modern computers, Ogbuobiri, Tinney and Walker concluded that exploitation of sparsity is the main objective [40].

Among the ordering strategies which attempt to reduce the number of fill-ins, there are two approaches. One is to minimize the number of fill-ins

regardless of their position in the matrix [65]. The other confines the non-zero elements to a small region. Several different forms have been proposed for this group of ordering schemes, among which matrix banding schemes have received the most attention [53-62].

The objective of banding schemes is to find a permutation of the matrix such that its non-zero elements are clustered in a narrow band about the major diagonal [38] or about the minor diagonal [57]. If no row or column interchanges are performed, this banded form is retained in the corresponding Cholesky factor or during Gaussian elimination. This type of ordering has found several practical applications [57-61]. We usually say that a matrix has bandwidth $2m+1$ and semi-bandwidth m if m is the smallest integer such that $a_{ij} = 0$ for any $|i-j| > m$. In the use of band methods all zeros outside the defined band are ignored, while those located within the bandwidth are usually stored and treated as entries.

More sophisticated algorithms referred to as variable band (also called profile or envelope) schemes have shown to be more advantageous over simple band methods [53-62]. A widely used ordering algorithm of this type was proposed by Cuthill and McKee [53]. They designed their algorithm to reduce the bandwidth of a sparse symmetric matrix. Many other orderings proposed since then, have not offered significant advantages over this algorithm. However, George [62] in his study of the envelope methods found that reversing the Cuthill-McKee scheme often yields significant improvement in the total storage requirements within the envelope and in the number of arithmetic operations. Later Liu and Sherman [63] proved that the reverse algorithm is always superior to the original Cuthill-McKee as far as the total envelope storage and operation counts are concerned.

Apart from the matrix-banding schemes, Tinney and Walker [64] have introduced three other schemes which aim at the optimum conservation of matrix sparsity during Gaussian elimination. These are usually referred as

scheme 1, scheme 2 and scheme 3 ordering algorithms. These schemes are briefly explained below:

Scheme 1 : In this scheme, the rows of the coefficient matrix are numbered based on the number of its off-diagonal non-zero elements. The rows having the fewest non-zero elements are numbered first and if two rows have equal off-diagonal non-zero elements, either of them may proceed the other.

Scheme 2 : By far this scheme is the most popular fill-reducing algorithm proposed by Tinney and Walker in 1967 [64] and explained by Tinney in 1969 [65]. This scheme, sometimes referred to as the minimum degree algorithm [66], basically corresponds to the Markowitz scheme [67] introduced for unsymmetric matrices in 1957. In this scheme, the strategy is that at any elimination step the row which has the fewest number of off-diagonal non-zero elements is eliminated first.

The key to implement this algorithm efficiently, is to avoid the explicit storage of the fill-ins. This objective was popularized outside the power system area by Rose [68] and by George and Liu [37].

Scheme 3 : During the elimination of a node, new paths are added to the graph of the matrix as a result of the elimination process. In this scheme, the nodes generating the fewest number of newly generated paths are eliminated first.

Each of these three ordering schemes have been shown to be very helpful when applied to power network problems. However, it is also possible, in rare cases, that the schemes give a poor ordering.

In conclusion, ordering schemes are basic to the solution of large sparse systems of linear equations and have been extensively applied to sparse

matrices in electrical power problems [69, 70]. Of these, Tinney scheme 2 appears to be the most efficient one in terms of computation time and storage [40].

2.7.1.3 Matrix partitioning

Matrix partitioning is another method for reducing the computational effort for certain classes of problems [71-74, 77]. In this approach, a huge problem may be subdivided into smaller systems of equations and then sparse matrix techniques may be applied to these sub-problems. One strong feature of the partitioned approach appears when the subsystems associated with the coefficient matrix have some special characteristics that can be easily solved. George [71] describes some examples on partitioned factorization.

It is well known in the power system area that some control actions and small disturbances mainly affect the operating state of only a small portion of the power network electrically "close" to the disturbance area. Thus, it is possible to improve the performance of power system analysis problems by partitioning a power network (matrix) into strongly connected sub-networks (sub-matrices) and utilizing the partitioning aspect [74].

2.7.1.4 Compensation methods

In power system analysis, it is frequently required to modify the coefficient matrix of related linear systems of equations. These modifications normally do not involve changing a large number of elements, as in contingency analysis where usually only one line at a time is removed. In these cases, refactoring the coefficient matrix is rarely efficient. Instead, compensation methods [75] can be applied to available factors of the base case coefficient matrix to get the solution to the system more economically.

Compensations methods are essentially different interpretations of the Matrix Inversion lemma [75, 76, 112]. In this approach, the modified coefficient matrix is represented as,

$$A' = A + M\,[\Delta A]\,M^t \tag{2.22}$$

where A is the base case n by n coefficient matrix, ΔA is an m by m matrix containing modifications to A and M is an n by m connection matrix.

Compensation methods have been successfully applied to power system problems [75, 76] and have been shown to be very efficient for applications involving a series of network changes [76], specially for applications where the number of changes is small and the modifications are not permanent.

2.7.1.5 Parallel processing

The computer architecture is a fundamental factor affecting the efficiency of different algorithms. Unfortunately, so far, there are only a few investigations comparing different architectures for common problems. However, it is evident that with the rapidly changing computer technology toward vector processors and parallel architectures, possible gains are expected for some problems by parallelizing parts of the solution. For example, an advantage of parallel processors is their ability to perform calculations on full matrices efficiently. Another advantage appears when several different systems are to be solved simultaneously [117-119].

Thus, one new trend seems to be the application of parallelizing techniques to direct solvers [117-119] as well as indirect ones [79, 80]. In this vein, it has been shown that the Cholesky decomposition problem for a large dense system can be reduced to p independent smaller decomposition problems [81].

The potential of parallel processing architectures in power system studies has been investigated by some researchers during the past decade [86]. Its potential impact on load flow [82, 83] and contingency analysis [84, 85] seems to be very promising.

2.7.2 Indirect solvers

Indirect solvers iteratively approach the desired solution from an initial guess $x_{(0)}$ through a sequence of vectors,

$$x_{(0)} \rightarrow x_{(1)} \rightarrow x_{(2)} \rightarrow \ldots \tag{2.23}$$

The computation involved in each iteration step for these methods is essentially comparable to the multiplication of matrix A with a vector, a computational effort which is relatively modest if A is sparse. The convergence rate, however, is typically linear and thus, many iterations may be required to obtain an accurate solution. Furthermore, convergence is not guaranteed even with exact arithmetic. These general characteristics of iterative methods do not apply to conjugate gradient algorithms which are guaranteed to converge to the exact solution in a finite number of arithmetic operations for positive definite matrices. For this reason, some refer to this technique as semi-iterative.

2.7.2.1 Basic iterative techniques

In the following we describe some basic iterative and semi-iterative methods: Jacobi, Gauss-Seidel, successive overrelaxation (SOR), conjugate gradient (CG) and pre-conditioned conjugate gradient (PCG) methods.

The first three of these methods may be expressed in the form,

$$B\,x_{m+1} + (A-B)\,x_m = b \tag{2.24}$$

or,

$$x_{m+1} = (I-B^{-1}A)\,x_m + B^{-1}b \tag{2.25}$$

Assuming $(I-B^{-1}A)=G$ and $B^{-1}b=k$, we can write,

$$x_{m+1} = G\,x_m + k \tag{2.26}$$

where G is called the iteration matrix for the method.

From equation 2.26, it is clear that all of these methods are linear as both G and k are constant and do not depend on x_m. These methods are also of first degree, since x_{m+1} depends explicitly on x_m and not on $x_{m-1}, \ldots, x_0$. Such iteration methods, were first considered by Withmeyer in 1936 [87]. In this class of iterative techniques, each choice of a non-singular matrix B in equation 2.24 leads to a potential iterative method.

To choose the B matrix appropriately, it must satisfy the following conditions [88] :

i) The system of equations 2.25 can be solved easily for x^{m+1}.

ii) The eigenvalues of $I-B^{-1}A$ (the iteration matrix) have moduli which are as small as possible and smaller than one.

The better B agrees with A, the more likely the latter condition will be true and the better the algorithm will convergence, but at the expense of a heavier computational burden for B^{-1}.

The first three iterative methods mentioned above differ from each other according to the selection of the B matrix. These differences are described in Appendix A.

2.7.2.2 Conjugate gradient

The conjugate gradient scheme (CG) [99-101] is an important semi-iterative technique when the coefficient matrix is positive definite. This method was first presented by Hestenes and Stiefel in 1952 [99]. However, for various reasons, the CG method was not widely used for years after its appearance until the mid-1960s [100-104].

In the absence of rounding errors, in contrast to other iterative techniques, the CG method terminates with the exact solution in at most n steps [99-101]. Because of rounding errors, however, additional iteration steps may be required [88].

The basis of the method comes from the fact that the function,

$$F(x) = \frac{1}{2} x^t A x - x^t b \qquad (2.27)$$

is minimized by $x=A^{-1}b$. In other words, finding the minimum point of F will give us the solution to the linear system of equations $Ax=b$ if A is positive definite.

There are various ways of finding the minimum of F. Conjugate gradient algorithms use a sequence of linear search directions (p) starting from an initial guess x_0 and, at each iteration, a better estimate of the solution is obtained, such that,

$$x_{i+1} = x_i + \alpha_i p_i \tag{2.28}$$

where α_i is a scalar.

Thus, the final solution can be expressed as a linear function of the search directions,

$$x = \alpha_1 p_1 + \alpha_2 p_2 + \ldots + \alpha_m p_m \tag{2.29}$$

where, m is guaranteed to be no larger than the matrix size n and mainly depends on the distribution of the eigenvalues of the coefficient matrix [102]. It has been shown that the algorithm will converge in significantly fewer than n iterations if the eigenvalues of the A matrix are located in clusters [102]. The details of the CG algorithm are described in Appendix B.

One drawback of this method is that it is not guaranteed to converge for non-positive definite matrix problems [106]. On the other hand, it has the advantage that a sufficiently good approximation may be found after only a few steps in certain cases. Furthermore, when the eigenvalues of A form c clusters, then the solution will be obtained after O(c) iterations.

In contrast to other iterative techniques beginning with an initial guess, the convergence of the CG algorithm is not affected very much by the choice of the initial guess [97].

The amount of computational effort per iteration is about equal to that of multiplying the matrix A by a vector [88, page 572]. Thus, the method is not recommended for dense matrices or for very banded matrices where direct solvers are faster.

2.7.2.3 Pre-conditioned conjugate gradient algorithm

The CG method in its initial form proposed by Hestenes and Stiefel [99] is not competitive with modern fast direct solvers utilizing efficient sparsity and ordering techniques. However, as it was mentioned earlier, the algorithm will converge in significantly fewer than n iterations if the eigenvalues of the coefficient matrix are located in clusters [102].

The idea of pre-conditioning [100-101], in essence, is to apply a linear transformation to the system of linear equations Ax=b as follows,

$$A_p\, y = b_p \tag{2.27}$$

where,

$$A_p = [\, K^{-1} A\, (K^{-1})^t \,] \tag{2.31}$$

$$b_p = K^{-1}\, b \tag{2.32}$$

$$y = [\, K^t\, x \,] \tag{2.33}$$

such that the eigenvalues of A_p are grouped into a small number of clusters, thereby substantially improving the convergence and speed of the CG algorithm. The matrix K is termed the pre-conditioning matrix (also called the pre-conditioner). Obviously, the pre-conditioning step must be very efficient from the computational and storage points of view for the overall scheme to be superior to a straight CG approach.

A highly effective pre-conditioning matrix K can be obtained using approximations of the Cholesky factor L, where,

$$A = L\,L^t \tag{2.34}$$

A more detailed discussion on the choice of K is given in Chapter IV.

The pre-conditioned conjugate gradient (PCG) algorithm normally starts with a diagonal scaling and normalization of the original system of linear equations with respect to the largest absolute value of the right-hand-side entries to yield A x = b. Then, the PCG algorithm can be expressed as follows:

1) Initialization

a) Initialize ε

b) Guess x.

c) Form $r = b - A\,x$.

d) Form $p = H\,r$, *where* $H = (K\,K^t)^{-1}$.

While $\|r\| > \varepsilon$ *do*

2) Form $S = A\,p$.

3) Form $\sigma = p^t\,r$.

4) Form $\alpha = \sigma / (p^t\,S)$.

5) Update $x = x + \alpha\,p$.

6) Update $r = r - \alpha\,S$.

7) Form $h = K^{-1}\,r$.

8) Form $\beta = h^t\,h / \sigma$.

9) Update $p = K^{-t}\,h + \beta\,p$.

It should be noted that K^{-1} is not formed explicitly in the PCG solver.

Instead, whenever matrix-vector products involving $(K)^{-1}$ or $(K^t)^{-1}$ appear, it is more efficient to perform a forward elimination or back substitution (F/B) as required.

The number of iterations required for convergence of the pre-conditioned conjugate gradient (PCG) algorithm depends on the distribution spectrum of the eigenvalues of A_p as well as the rounding errors [107]. In practice, if the eigenvalues of A_p form c clusters (groups of eigenvalues close to one another), then the solution will be obtained in O(c) iterations [102, 107]. The better the eigenvalues are clustered, the lower the number of iterations required for convergence.

2.7.2.4 Theoretical comparison of PCG and direct solvers

Experience demonstrates that the PCG algorithm generally converges in $O(n^{0.5})$ iterations [116]. For power networks this convergence property is even more favourable as discussed in Chapter III. In addition, the PCG algorithm requires sn operations per iteration, assuming an average of s non-zeros per row. This then gives a total operation count of $O(sn^{1.5})$. The storage requirement is O(ns).

For a banded or Frontal Gaussian elimination solver, with only the active part of the matrix stored in memory, the corresponding figures are $O(nm^2)$ operations and $O(m^2)$ storage elements, where m is the matrix bandwidth (the average width of non-zero elements about the diagonal). This indicates that for large matrices (i.e. large n) with a bandwidth, m, which is not too small ($m^2 > n^{0.5}$), the PCG method will need fewer operations than a direct solver. If we consider the case that the bandwidth of the matrix, m, is of order $n^{0.5}$ then the ratio of arithmetic operations between the direct method and the PCG is $O(n^{0.5})$. If the bandwidth, m, is linear in n then this ratio is of order $n^{1.5}$.

To compare the performance of the PCG algorithm with direct solvers two different criteria should be considered.

Repeated solutions of the form,

$$A\ x_k = b_k\ (x_{k-1}) \qquad k = 1,2,... \tag{2.35}$$

such as those found in the fast decoupled load flow, represent a case where the direct solver has its greatest advantage over PCG methods. This is because the principal time consuming component of the direct solver, which is the matrix decomposition into its lower and upper factors, needs to be done only once for the entire sequence of solutions. Each solution is then found by the comparatively cheap step of a forward and backward substitution for each different right-hand-side b_k. The PCG method, on the other hand, applies the same algorithm for each k requiring the same computational effort for each new problem in the sequence. In the PCG method, no major improvement can be gained by taking advantage of the fact that A is constant. The reason being that finding the pre-conditioning matrix, K, is a relatively cheap part of the PCG solver for large systems. However, as will be shown in the next chapter, even in this type of sequence, where the direct solver has its greatest advantage, the PCG method can be much faster, if the number of load flow iterations is not too large or if the A matrix has a sufficiently wide bandwidth, m.

In cases where repeated solutions of the form,

$$A_k\ x_k = b_k \qquad k = 1,2,... \tag{2.36}$$

are required, the speed advantages of PCG over direct solvers are enhanced further, since for each new k, the coefficient matrix A_k must be refactored or its factors must be updated through compensation methods [75, 76] (if the matrix modification is of low rank). Such cases frequently occur in power system security analysis where the behaviour of the power network relative to numerous possible equipment outages has to be analyzed.

In security analysis where the outages being considered involve several pieces of equipment, the corresponding Jacobian matrix modification will not be of low rank and compensation methods lose their efficiency. In this case the direct solver approach requires a complete refactorization for each k, thus making the PCG method even more advantageous.

Another advantage of PCG is the ability to stop iterating after an acceptable solution error is obtained without necessarily reaching the final solution. This can have important applications in contingency ranking where a reasonable approximation of the effect of a contingency may be sufficient to analyze the relative effect of a set of contingencies.

A limitation of the PCG algorithm is that it has been extensively and successfully tested primarily when dealing with positive-definite matrices only. The practical application of PCG methods to indefinite or semi-definite matrices is yet to be as widely exploited.

Due to this comparison and the fact that PCG has been very successfully applied to problems in other fields such as finite element electromagnetics [28], we were motivated to investigate the algorithm in typical power system analysis problems.

2.7.2.5 Necessary conditions for fast convergence of the PCG algorithm

It was mentioned in section 2.7.2.2 that the only condition for the convergence of the CG algorithm with no rounding errors is that the coefficient matrix A should be positive definite. Therefore, in the PCG algorithm, the pre-conditioning of A as shown in equation 2.31, must ensure that the pre-conditioned coefficient matrix A_p remains positive definite.

To comply with the requirements of fast convergence, attention should be paid to two main properties :

i) A good pre-conditioner should be used to meet the requirements necessary for good clustering of the coefficient matrix eigenvalues. In this regard, Chapter IV discusses variations of the Incomplete Cholesky pre-conditioner.

ii) Precautions should include the influence of rounding errors. In practice, rounding errors may have substantial effect on the convergence properties of PCG. Van der Vorst in his studies has shown that rounding errors may severely deteriorate the convergence behaviour that would have been superior with exact arithmetic [107]. Double-precision is therefore recommended.

CHAPTER
THREE

EXPERIMENTAL RESULTS ON THE APPLICATION OF A PRE-CONDITIONED CONJUGATE GRADIENT ALGORITHM TO THE LOAD FLOW PROBLEM

3.1 Introduction

As mentioned in the previous chapter, the incomplete Cholesky pre-conditioned conjugate gradient (PCG) algorithm is a very powerful semi-iterative solver with proven significant speed advantages over direct methods in the area of finite element electromagnetic analysis. It was also stated that the convergence of the PCG algorithm requires that the coefficient matrix A should be positive definite.

This chapter begins by showing that the coefficient matrices arising in the Fast Decoupled and DC load flow algorithms are positive definite and, thus, comply with the requirements necessary for the convergence of PCG algorithm. The performance of PCG within the Fast Decoupled and DC load

flows is then experimentally analyzed by running extensive simulations on various types and sizes of power networks using both the PCG and direct methods. More specifically, the computation time of the PCG algorithm with classical incomplete Cholesky pre-conditioning is compared with that of two standard direct solvers, namely a bandwidth based Frontal solver [60] and the Sparspak-B5 solver with minimum degree ordering [110, 111] also known as the Tinney-2 method. This comparison is performed on a wide spectrum of power networks of up to 5000 buses and 10000 lines. The results of our experiments indicate that for certain classes of large sparse systems or for repeated solutions with matrix modifications, the PCG method is significantly more efficient than direct techniques and offers important savings in CPU time.

In this chapter, the effect of certain network parameters on the performance of PCG and direct solvers is also investigated. These parameters include the relative ordering of sub-networks (blocks) within the network, as well as the range of values of line reactances.

Finally, in section 3.5, some possible new load flow algorithms which make use of the PCG algorithm are numerically investigated.

3.2 Positive Definite Matrices in Power System Analysis

The DC load flow algorithm described in section 2.2.4 is formulated by a system of linear equations (2.15) whose coefficient matrix, B, is positive definite, a fact which can be shown as follows :

The B matrix is completely analogous to an equivalent conductance matrix, G, in a purely resistive grounded network if we make the following associations:

i) The reactance, x_{ij}, of the line between buses i and j becomes the analogous line resistance.

ii) The vector of power injections, P, becomes the analogous vector of current injections, I.

iii) The vector of phase angles, θ, becomes the equivalent voltage vector, V.

From energy conservation considerations, a connected and grounded network with positive resistances in all branches including the branch to ground has a positive definite G matrix. To see this consider a resistive network represented by its admittance matrix G. Then, if the network is excited by any set of nodal voltages expressed by the vector V, the input power to the network will be greater than or equal to zero,

$$P_{input} \geq 0 \tag{3.1}$$

Thus,

$$P_{input} = V^t I = V^t G V \geq 0 \tag{3.2}$$

Now, as the network is fully resistive and grounded, it is evident that for any non-zero vector V, at least one of the resistive elements in the circuit will have a non-zero current flow. This, then, implies that for any non-zero vector of voltages, the input power of the circuit will be greater than zero,

$$V^t G V > 0 \tag{3.3}$$

and, therefore, it can be concluded that the network admittance matrix G is positive definite.

This, then, implies that the B matrix is also positive definite and, therefore, that the DC load flow can be solved by a PCG method.

As examples of this result, the eigenvalues of the B matrices of the standard IEEE power networks with 14, 35, 57 and 118 buses were calculated.

Table 3.I: Range of the eigenvalues of IEEE standard power networks with 14, 30, 57 and 118 buses

IEEE sample networks	eigenvalue range	
	minimum	maximum
14-bus	0.6657	80.8529
30-bus	0.2392	113.5110
57-bus	0.2088	168.4240
118-bus	0.0559	582.5873

Table 3.I shows the range of these eigenvalues which are all real and positive.

Another instance of positive definite matrices occurring in power system analysis is the Fast Decoupled load flow (FDLF) matrices B′ and B″ (see section 2.2.3). The matrix B′ is identical to the B matrix of the DC load flow, while B″ is a principal sub-matrix of B′. Clearly, since B is positive definite, so are B′ and B″. Thus, in each iteration of the FDLF, we are dealing with two sets of linear equations whose coefficient matrices are positive definite and therefore, can be solved by using either a direct solver or the PCG.

Finally, matrices arising in state estimation [21, 22, 30] are also positive definite. These algorithms are, thus, suitable candidates to use the PCG solver.

3.3 Numerical Comparison of the Performance of the Pre-Conditioned Conjugate Gradient and Frontal Solvers

In this section, the performance of the pre-conditioned conjugate gradient (PCG) algorithm with classical incomplete Cholesky pre-conditioning is experimentally compared with that of the standard bandwidth based Frontal

direct solver [60]. This solver attains its highest efficiency when applied to systems of linear equations whose coefficient matrices are narrow banded. In section 3.4 the PCG solver is compared to another direct solver utilizing the Tinney-2 ordering method.

Both the PCG and Frontal algorithms are applied to solve systems of linear equations arising in the Fast Decoupled and DC load flow calculations. The computation time of the PCG algorithm is compared with that of the Frontal solver for a wide spectrum of networks of up to 5000 buses and 10000 lines. The impact of network size and topological connectivity on the relative performance of the PCG and Frontal solvers is also numerically investigated.

3.3.1 Criteria of comparison

To evaluate the time requirements for the solution of the FDLF, we have taken the conservative view that, for typical well-behaved power networks, starting from a flat voltage profile, 7 P and Q-iterations are required to converge on the average. Then, to evaluate the performance of the two solvers in the FDLF, the CPU time of one complete direct solution (one factorization and one F/B substitution) and the CPU time of seven complete PCG solutions (seven incomplete Cholesky factorizations and seven executions of the PCG algorithm) are compared. This conservative comparison places our results on the safe side.

For the DC load flow one complete solution of the PCG is compared with one complete direct solver solution.

The results of extensive comparisons of the above nature are shown in Figures 3.3 to 3.6 in the form of CPU time ratios versus network size.

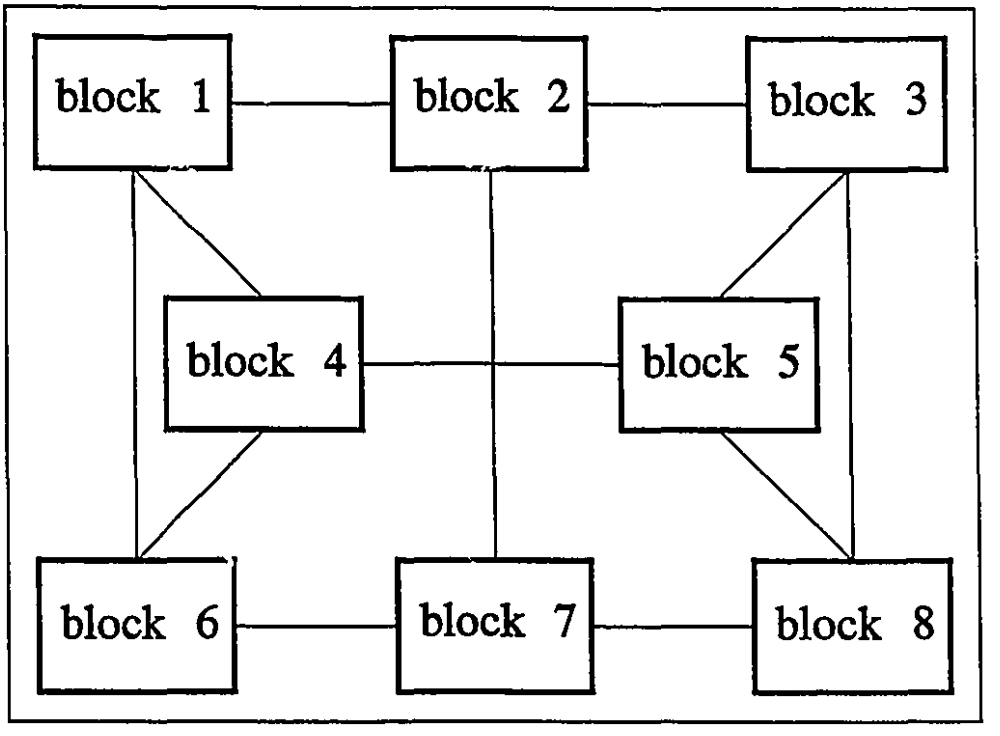

Figure 3.1: Block interconnections of the networks

3.3.2 Test networks

In all the examples tested, the networks were composed of sparse blocks of varying sizes which are, themselves, interconnected by a sparse network, an example of which appears in Figure 3.1. Figure 3.2 illustrates a typical sparsity structure of the corresponding network B matrix.The number of lines for all networks was about twice that of the number of buses.

The test networks were randomly generated using special software developed during the course of this research that designs realistic power networks of different sizes, topologies and line data (see Chapter VI for more details about this special network generation software). This software allows the user to specify the power network dimension and other characteristics, producing a series of random networks with the general specified properties.

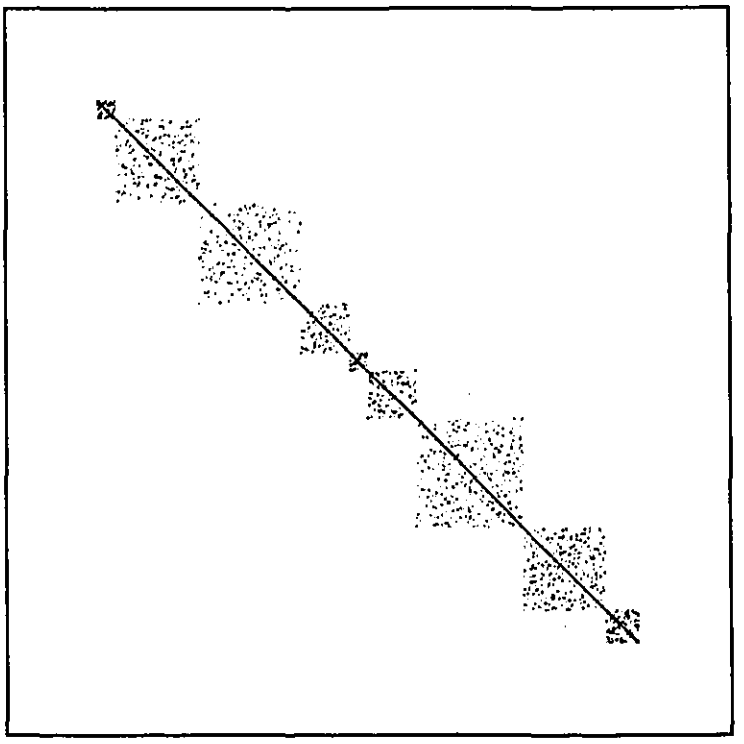

Figure 3.2: Sparsity structure of an example network

There is no limit on the size of the network that can be thus generated. The test cases produced here range up to 5000 buses and 10000 lines.

Although these networks are simulated, we have made them as realistic as possible by ensuring that the topology, line and bus data are similar to those of actual power networks.

As the Frontal solver is a bandwidth based solver and due to the fact that it gains its efficiency for narrow banded matrices, in these series of experiments, the input data for the solvers were generated in an ordered manner to have the minimum possible bandwidth around the major diagonal as indicated in Figure 3.2. This choice of data favours the performance of the Frontal solver.

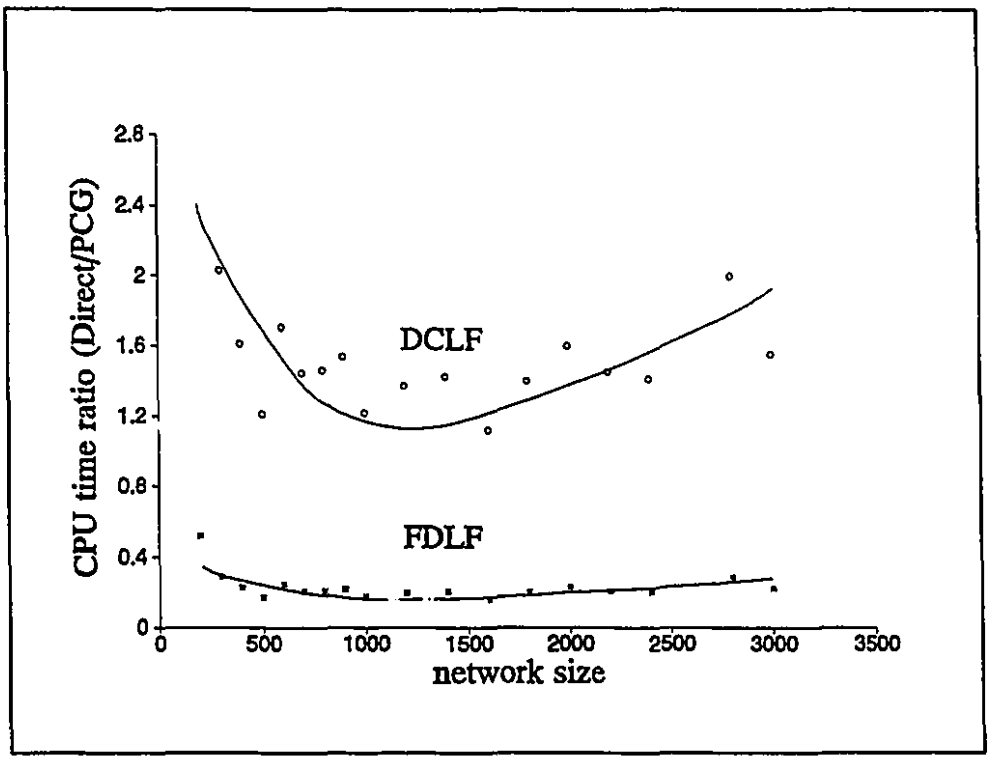

Figure 3.3: CPU time ratio of direct and PCG solvers
(Largest sparse block has 100 buses)

3.3.3 Discussion of results

The results are summarized according to four different network categories defined by the size of the largest sparse block in the network. In essence, we are comparing networks with different matrix bandwidths (the size of the diagonal band wherein most of the non-zero entries are located). For each category, we compare the ratio of CPU times of one direct solver (1 decomposition and 1 F/B substitution) versus the CPU time of seven PCG solutions (7 incomplete Cholesky factorizations and 7 applications of the PCG algorithm) for the FDLF. This assumes that the FDLF converges in seven iterations on the average. The results are shown in Figures 3.3 to 3.6 each of which shows two curves. The upper curves show the CPU time ratio for the DC load flow where one direct solution is compared to one PCG solution. The lower curves display the corresponding comparison for the FDLF.

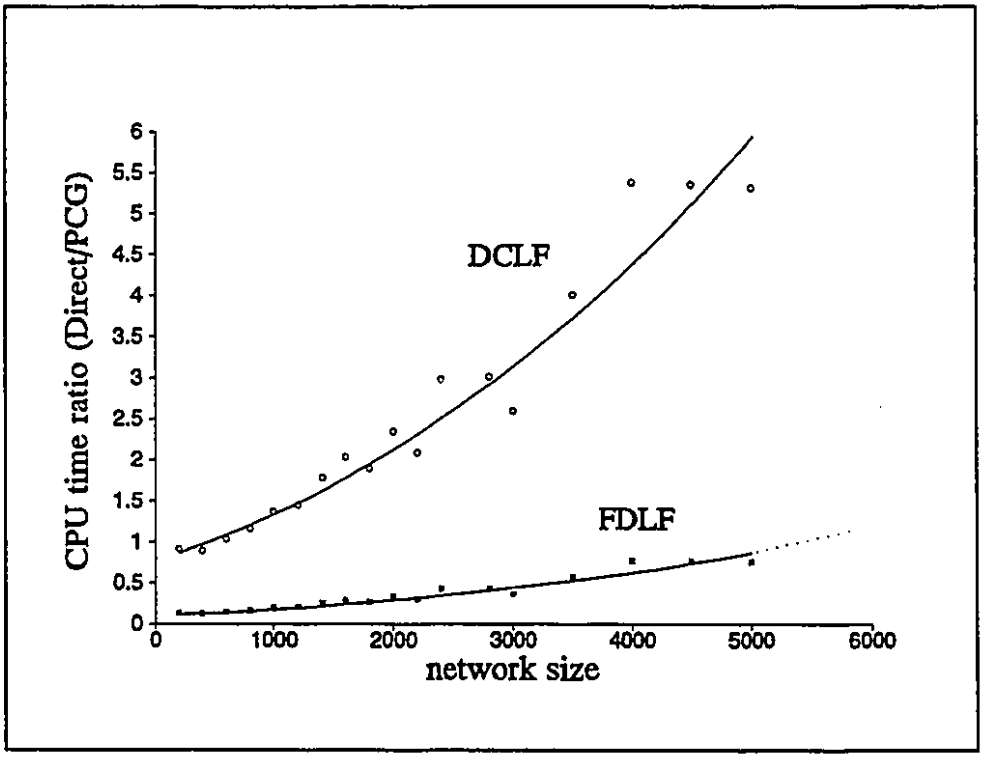

Figure 3.4: CPU time ratio of direct and PCG solvers
(Largest sparse block is one tenth the network size)

A) In the first category, experiments were carried out on networks of different sizes where the largest sparse block contains 100 buses. Other blocks in the network have random sizes ranging between 6 and 100 buses. The CPU times obtained from test results are compared in Figure 3.3.

Comparison of the two methods indicates that, for this category of narrow-banded networks, the direct solver is more efficient for the FDLF while, for the DC load flow, the PCG shows a slight advantage.

B) In the second category, networks with the largest sparse block equal to one tenth the size of the network were examined. The relative CPU times are plotted in Figure 3.4.

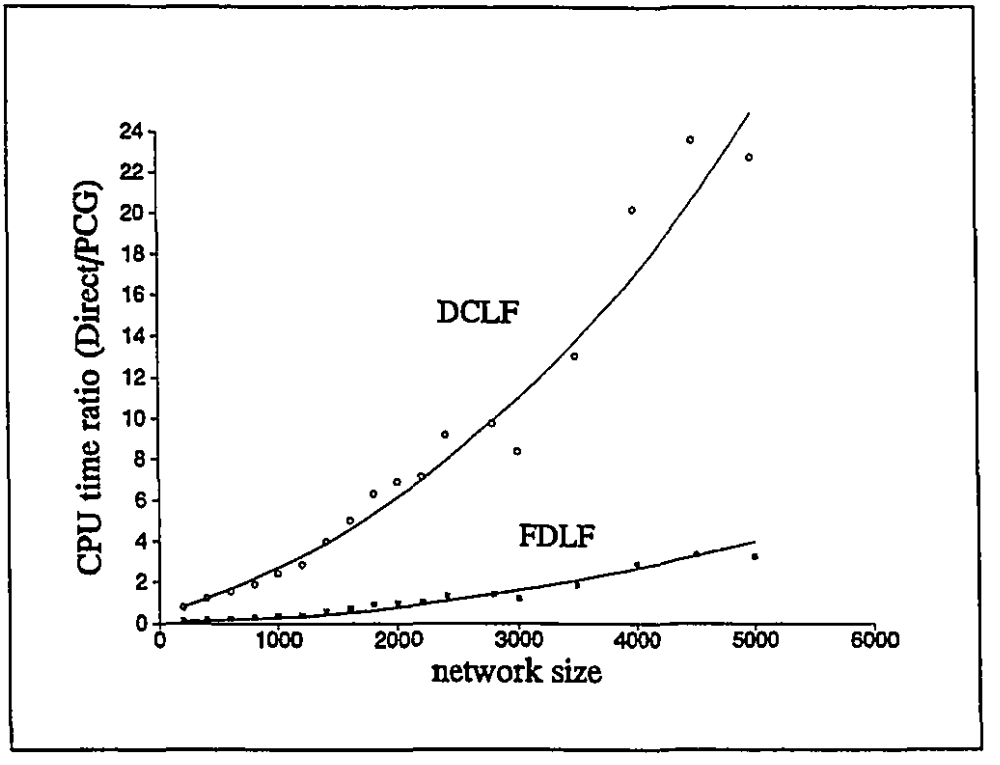

Figure 3.5: CPU time ratio of direct and PCG solvers
(Largest sparse block is one fifth the network size)

We see here that, for systems of more than 5750 buses (extrapolated) (with corresponding largest blocks of size 575 or more), the pre-conditioned conjugate gradient becomes more efficient for the FDLF. Still, no substantial advantage (more than 2) is gained by using the PCG for this category of network in the FDLF unless one goes to dimensions above 7000 buses. For the DC load flow, the PCG shows an advantage for networks above n=600.

C) In the third category of network, the biggest block has a size equal to one fifth the network size. The test results for these cases are plotted in Figure 3.5.

In this case, in the FDLF, the PCG method becomes more efficient than the direct method for systems larger than 2050 buses (i.e. a largest block of 410 or more buses). The advantage of the PCG method becomes even

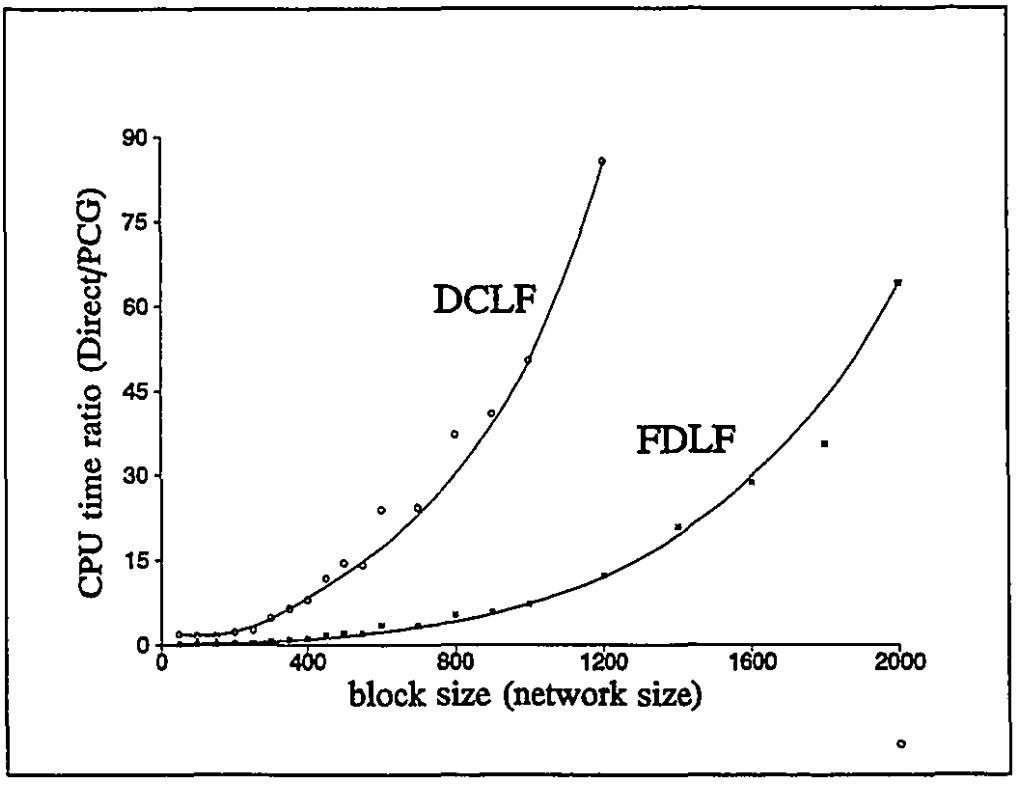

Figure 3.6: CPU time ratio of direct and PCG solvers
(Network is composed of only one block)

more pronounced for larger systems. For example, at n=5000, the ratio of CPU times is about 4. The advantage of PCG in the DC load flow is even more pronounced reaching the ratio of 24 at n=5000.

D) In the last group of experiments, the solvers were applied to networks composed of only one block, that is, a wide band sparse network (Figure 3.6) where the number of lines is as before about twice the number of buses.

For this type of network, the PCG method is considerably more efficient than the direct method in both the FDLF and the DCLF. The cross-over point where PCG becomes faster than direct methods starts near n=300 for the FDLF. At n=2000 buses, for example, the PCG solver is about 60 times faster for the FDLF and 420 times faster for the DC load flow.

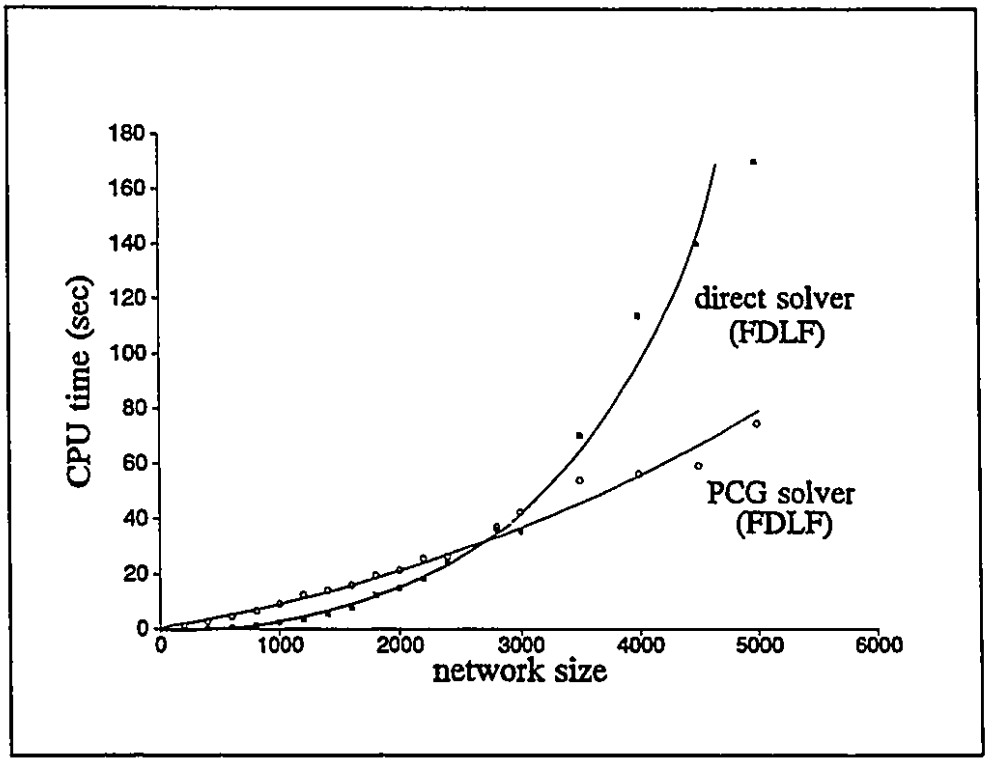

Figure 3.7: CPU times (Sun SPARC station 2, 32 Mb RAM) versus network size for two solvers
(largest sparse block is one fifth the network size)

As illustrated in Figure 3.7, and predicted by the theory presented in section 2.7.2.4, the CPU time for the PCG algorithm varies as $O(n^{1.5})$ whereas the time for the direct method varies as $O(nm^2)$. It was also observed, that the largest block size (matrix bandwidth) affects the PCG solver performance but not significantly. What is most important in the PCG performance is the clustering of the eigenvalues of the pre-conditioned matrix. In fact, in some cases, we have observed that the PCG solver actually took less CPU time for a larger system than for a smaller one due to the fact that the larger system was better clustered and converged in fewer iterations. For the direct solver, however, the solution time was observed to increase according to nm^2, where m is the dimension of the largest block. The size of the biggest block has a dominant effect on the CPU time of the direct solver (See Table 3.II). This is because a larger block generally results in a wider bandwidth and therefore more fill-in elements in the factorization.

Table 3.II: CPU times (sec) of the two solvers for n=2000 and variable largest block size, m (Sun SPARC station 2, 32 Mb RAM)

network size (n) = 2000				
Solver	matrix bandwidth (m)			
	100	200	400	2000
Frontal	3.83	6.09	14.81	438.39
PCG	2.40	2.60	2.15	0.98

Table 3.III: CPU time (sec) of the Frontal solver for different block arrangements (Sun SPARC station 10, 64 Mb RAM)

Size of the network	block arrangements		
	Random	small to big	big to small
200	0.090	0.096	0.085
600	0.719	0.859	0.660
1000	2.270	2.294	1.860
1400	5.680	5.690	5.400
1800	12.360	11.760	12.020
2200	18.560	20.010	18.320
2400	24.490	24.940	23.390
2800	36.220	36.600	34.020
3000	36.170	37.130	33.710
3500	70.780	69.630	65.560
4000	114.020	109.90	108.440
4500	139.160	136.160	132.01
5000	170.710	170.660	171.89

3.3.4 Effect of block arrangement on the solution time

To investigate the effect of block arrangement, in another series of experiments, network blocks were connected in two different manners. First, they were reordered such that smaller ones proceed larger ones along the major diagonal and, second, they were ordered from larger to smaller ones. The experimental results for networks of different sizes from 200 up to 5000 buses whose biggest block has a size equal to one fifth of the network size demonstrate that neither the performance of the PCG nor the performance of the Frontal solver is affected significantly by block arrangement. However, it was observed that for networks of smaller sizes, the Frontal solver shows a small speed advantage for those networks which have the blocks arranged from big to small (see Table 3.III).

3.3.5 Impact of variations of line parameters on eigenvalue clustering and convergence properties of the PCG algorithm

It is clear that the magnitudes of line reactances directly affect the magnitudes of the non-zero elements of the DC and Fast Decoupled load flow matrices. In the previously mentioned results, these values were chosen to be as realistic as possible. However, to investigate the effect of line parameter variations on the PCG, several cases were run where the network reactances were varied over several ranges of values.

The experiments were performed for networks of 2000 and 5000 buses composed of blocks ranging from 10 buses to one fifth of the network size.

First, it was observed that reactance variations do not affect the performance of the direct solver. This result is expected due to the fact that reactance variations do not change the matrix non-zero elements topology and thus do not have any impact on the number of fill-ins introduced during the factorization. On the other hand, it was observed that the bigger the variation

Table 3.IV: CPU time (sec) and iteration steps of the PCG solver for different line reactance variations (Sun SPARC station 10, 64 Mb RAM)

Size of the network	Line reactance variations					
	10%		100%		5000%	
	CPU	NI	CPU	NI	CPU	NI
2000 buses	1.08	40	1.21	43	1.55	56
5000 buses	3.47	49	4.21	60	4.72	69

of line reactances, the greater the number of PCG iterations (NI) and thus the more CPU time is required for PCG to converge (Table 3.IV). This can be explained by the fact that wide reactance values can broaden the eigenvalue spectrum of the coefficient matrix, thereby worsening the PCG convergence.

3.4 Numerical Comparison of the Performance of the Pre-Conditioned Conjugate Gradient and Sparspak Solvers

As discussed in chapter II, the minimum degree ordering scheme (Tinney-2) has been efficiently applied in power system problems. Thus, we were motivated to evaluate the performance of the PCG algorithm compared with efficient direct solvers other than the Frontal solver, particularly the one utilizing the Tinney-2 ordering algorithm. These efficient solvers were selected from the well known Sparspak package of sparse direct solvers [110, 111]. This package offers a collection of different approaches grouped as Sparspak-A solvers and an enhancement to it called Sparspak-B. For all of the variations in this package, the system of linear equations is solved through the following basic steps:

1) The non-zero structure of the coefficient matrix A is supplied to the package.

2) The original problem is reordered using a permutation P and then proper storage allocation is performed for the triangular factors as,

$$P A P^t = L U \tag{3.4}$$

3) The values of the non-zero elements of the coefficient matrix are supplied to the package.

4) The triangular factors L and U are computed.

5) The right-hand-side vector is supplied to the package.

6) The solution vector x is found using F/B substitutions.

In addition, the package disregards the numerical stability due to any permutation matrix P. This means that, the software assumes that for any permutation matrix P, an acceptably accurate factorization can be obtained. This is true for positive definite and diagonally dominant coefficient matrices [110] such as matrices arising in the Fast Decoupled and DC load flow algorithms. On the other hand, our PCG algorithm cares about diagonal scaling and stability problems. Therefore, this puts our experimental comparison with the PCG solver on the safe side as there is no CPU time dedicated by the Sparspak solvers for this consideration.

Sparspak solvers differ mainly from each other due to the various algorithms applied for choosing P, the corresponding storage methods and whether the coefficient matrix is considered symmetric or not. The effectiveness of these variations still is not well understood and mainly depends on the type of problem [111]. Due to these different characteristics, the performance of these solvers were numerically investigated for power system problems. We observed that Sparspak-B5, utilizing minimum degree ordering (Tinney scheme-2), is the most efficient for matrices arising in power system problems. Therefore, in our experimental investigation, the PCG

algorithm and the Sparspak-B5 solver including minimum degree ordering algorithm were used to solve systems of linear equations arising in DC load flow and Fast Decoupled load flow algorithms for networks of varying sizes from 100 buses to 4900 buses.

These experiments were carried out for two different types of networks, simple grid networks and more complicated star networks. In both cases, the number of lines is the same, that is, both grid and star networks have the same degree of sparsity.

a) Grid networks: This type represents networks which have the form of a matrix whose elements represent the nodal points of the network. In this category each bus is connected to adjacent buses only (Figure 6.2, chapter VI).

b) Star networks: In this type, first all of the nodes are laid on a ring of transmission lines to assure connectivity of all buses to the network. Then, extra random connections between the different nodes are created (Figure 6.3, chapter VI).

3.4.1 Discussion of the results

The performance of the PCG and Sparspak-B5 solvers has been experimentally investigated for the above mentioned types of networks. For each group, we compare the CPU time of one PCG solution with the CPU time required for one factorization and the Tinney-2 ordering scheme.

A) Grid networks: The experimental results for this group are presented in Table 3.V.

Comparison of the two methods indicate that, for grid networks, the Sparspak-B5 direct solver is comparable to that of the PCG. It is noteworthy

Table 3.V: CPU time comparison of PCG and Sparspak-B5 for grid-type networks (33 MHz, 486, 32 Mb RAM)

Size of the network	CPU time (Sec)			
	PCG	Sparspak-B5		
		Ordering	Factorization	Total
900	~1	< 1	1.5	< 2.5
1600	~2	< 1	3	< 4
2500	5	< 1	4	< 5
3600	8	1	6	7
4900	10	1	7	8

Table 3.VI: CPU time comparison of PCG and Sparspak-B5 for star-type networks (33 MHz, 486, 32 Mb RAM)

size of the network	CPU time (Sec)			
	PCG	Sparspak-B5		
		Ordering	Factorization	Total
400	< 1	< 1	< 1	~1
625	< 1	< 1	< 3	~3
900	~1	~1	5	~6
1225	~1	2	15	17
1600	< 2	3	33	36
2025	< 2	5	60	65
2500	< 2	7	125	132
3025	~2	10	230	240
3600	~3	14	345	359
4225	< 4	17	670	687
4900	~4	27	850	877

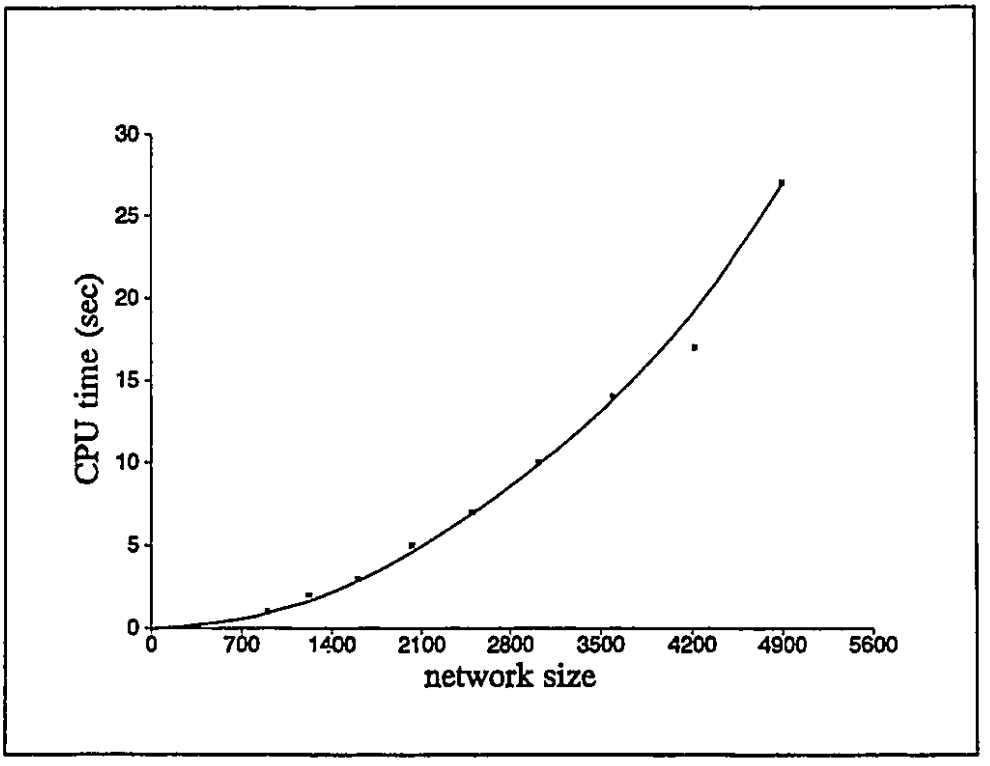

Figure 3.8: CPU time versus network size for ordering algorithm of the Sparspak-B5 solver (33 MHz, 486, 32 Mb RAM)

to mention that the stopping criterion for the PCG algorithm in all these experiments was a residual smaller than 10^{-6} where the right hand side of the equation has been normalized to one.

In all cases analyzed in this group of grid networks, as shown in Table 3.V, the CPU time devoted to ordering was equal to or smaller than one second.

B) Star networks: As in the grid networks, in this type, each bus is connected (on average) to four other buses. Test results for these cases are presented in Table 3.VI.

In this case, the PCG method becomes more efficient than the direct method for systems larger than 400 buses for the DC load flow. If we assume

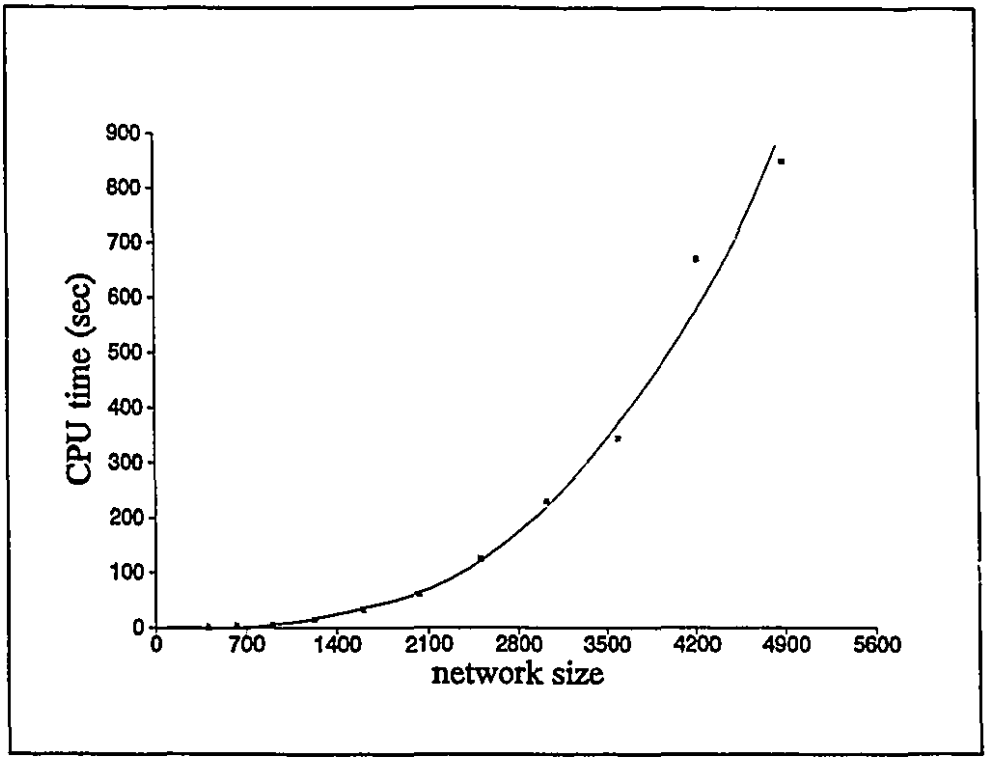

Figure 3.9: CPU time versus the network size for the factorization process in the Sparspak-B5 solver (33 MHz, 486, 32 Mb RAM)

the same criterion of comparison described in section 3.3.1 for the Fast Decoupled load flow, it can be observed that PCG shows marked advantages over Sparspak-B5 only for systems of 900 buses or larger.

It is also important to mention that in this table the F/B substitution time for the direct solver is not included. This is due to the fact the this time is very small compared with the factorization time. This puts our comparison on the safe side.

To get a better feeling about the performance of the direct solver, the CPU times required for ordering and for factorization are plotted versus the network size in Figures 3.8 and 3.9 respectively. As these Figures indicate, the CPU times in both cases increase very rapidly with network size.

We also examined the relative performance of the two solvers in one DC load flow solution for a real network of 688 buses and 920 lines. Both methods took less than one second to compute one solution. One noteworthy comment relative to the PCG method is that the number of PCG steps required to converge was less than one third of the theoretically expected number (7 instead of 26 iterations).

3.5 Alternative Load Flow Solution Algorithms Using PCG

This section describes an investigation into the convergence of other load flow algorithms using the PCG solver. These experiments can be classified into three different classes.

A) In the first class of experiments, the PCG algorithm was used to get the solution of systems of linear equations whose coefficient matrix is the polar (or rectangular) full load flow Jacobian matrix, [Jac]. This class of experiments was applied to the IEEE test networks of 5, 14, 30, 57 and 118 buses. None of these examples converged to the right solution. This was not unexpected since the Jacobian matrix is indefinite. The experiments simply serve to confirm our expectation.

B) In the second group of tests, the PCG algorithm was applied in the context of the load flow to solve systems of linear equations, Ax=b, where,

$$A=[Jac]\ [Jac]' \qquad (3.5)$$

and [Jac] represents the full Jacobian matrix in polar coordinates. These experiments were conducted for the IEEE example networks of 5, 14, 30 and 57-bus systems. The PCG algorithm converged to the right solution for all cases as expected since A is positive definite. However, due to the fact that the resultant coefficient matrix is not well-conditioned and that its eigenvalues

Table 3.VII: Convergence properties of PCG algorithm for a system of linear equations whose coefficient matrix is A=[Jac] × [Jac]′

IEEE system	n_j	cond(A)	Eigenvalues range	PCG steps
5-bus	7	400.57	$(0.0127\text{-}5.077)\times10^3$	1
14-bus	25	19413	$(0.005\text{-}5.334)\times10^3$	17
30-bus	53	247320	$(0.0000\text{-}1.2412)\times10^4$	33
57-bus	106	1.0215×10^8	$(0.0001\text{-}1.6678)\times10^5$	87

are distributed over a wide spectrum, the PCG algorithm did not show good convergence properties as indicated in Table 3.VII compared to the Fast Decoupled load flow. In this table, the dimension of the Jacobian matrix (n_j), the condition number of A [cond(A)], the range of eigenvalues, and the number of PCG steps required for convergence of the algorithm based on a residual smaller than 10^{-3} per unit are shown. The number of PCG steps also appears to be higher than the number expected from better conditioned matrices, that is, of the order of $n^{0.5}$ steps. It appears that the approach of converting the coefficient matrix to a positive definite one through premultiplication by its transpose is not competitive in the full Newton polar power flow.

C) In the final experimental approach, the Jacobian matrix in polar coordinates was replaced by the matrix A where,

$$A = \frac{[Jac] + [Jac]'}{2} \qquad (3.6)$$

Table 3.VIII shows the number of iteration steps required for the convergence of CG and PCG algorithms for a stop criterion based on residuals (r) smaller than 10^{-3} and 10^{-5} per unit.

Table 3.VIII: CG and PCG iteration steps for the solution of a system of linear equations whose coefficient matrix is A=([Jac] + [Jac′])/2

IEEE system	Size of A	$r < 10^{-3}$		$r < 10^{-5}$	
		CG	PCG	CG	PCG
5-bus	7	7	3	7	3
14-bus	25	21	10	26	12
30-bus	53	45	13	58	16
118-bus	192	115	28	138	33

Tests on the IEEE 5, 14, 30 and 118-bus systems show that the PCG approach converges in all cases. The number of PCG iterations is somewhat greater than the expected value (of the order of $n^{0.5}$), but it is still much less than the system dimension and considerably less than the CG method. It was also observed that the load flow algorithm with this approximate Jacobian does in fact converge, a result which is not obvious. However, we also noted an increase in the number of load flow iterations (each of which can be solved by the PCG method) of about 50% compared with the Newton-Raphson load flow algorithm using the conventional Jacobian. If this load flow convergence behaviour extends to large systems, then we expect the PCG method to be competitive in this approach.

3.6 Concluding Remarks

(1) The Incomplete Cholesky pre-conditioned conjugate gradient (PCG) algorithm can be applied to solve systems of linear equations arising in power system problems such as the DC load flow, the Fast Decoupled load flow, contingency analysis, state estimation, and a modified version of the full Newton polar load flow. It is shown here that these types of problems satisfy the requirement of our present PCG algorithm that the A matrix be positive definite.

(2) Tests comparing the PCG and a bandwidth based Frontal solver in power systems with up to 5000 buses and 10000 lines show that for large, sparse networks, with not too small bandwidth, substantial gains in computational speed (e.g. 65:1 for wide band networks of 2000 buses) can be achieved for the fast decoupled load flow. Even greater gains are obtained in the DC load flow.

(3) The relative efficiency of the PCG method compared with the bandwidth based direct solvers improves exponentially with the size of the network and even more rapidly with the size of the largest sparse matrix block.

(4) The tests on different orderings of network blocks show that the arrangement of the blocks does not have a significant affect on the performance of either the PCG or the Frontal solver. The size of the biggest block is the dominant factor on the efficiency of the Frontal solver.

(5) Experimental comparisons of the PCG and the Sparspak-B5 direct solver including minimum degree ordering algorithm (Tinney scheme-2) for power network matrices with up to 4900 buses and 9800 lines indicate that for grid-type networks, the PCG and direct solver are comparable. However, for large sparse star-type networks, substantial gains in computational speed can be achieved for the DC and the Fast Decoupled load flow.

(6) The performance of the PCG, unlike direct solvers, does not depend on the bandwidth of the network matrix appreciably. It depends mainly on the degree of sparsity and how tightly clustered the eigenvalues become after pre-conditioning.

(7) Our experiments on power networks have shown that the incomplete Cholesky pre-conditioning matrix is cheap to find, requires low storage, and clusters the eigenvalues tightly.

(8) The gains made by PCG are only in the solver component of the load flow and not in related parts such as input/output or mismatch evaluations. Since experience [26] indicates that the solver is the most time consuming part of load flow analysis, the gains made by the PCG should have an important impact in reducing the overall computation time of load flow algorithms.

(9) The PCG method should be seriously considered in power system analysis to revisit old algorithms or develop new ones where repeated modifications of the A matrix are needed. The use of direct methods has created a tendency to avoid such algorithms, even if they converge in fewer iterations and are more robust. This is because of the need to refactor the A matrix in these algorithms, which usually cancels out any speed gains. For example, in the load flow problem with control adjustments, in order to avoid refactoring the Jacobian, a large additional number of iterations are needed. This could possibly be avoided through the use of the PCG method whose performance is not affected by the modification of the Jacobian.

(10) Further investigation into the applications of PCG in power system computation are addressed in the following chapters including: more efficient pre-conditioning taking advantage of particular power network properties and applications to complex matrices.

CHAPTER
FOUR

INVESTIGATION OF EIGENVALUE CLUSTERING BY MODIFIED INCOMPLETE CHOLESKY DECOMPOSITION IN POWER NETWORK MATRICES

4.1 Introduction

It was mentioned in chapter II that a very fast method for computing the solutions of large sparse systems of linear equations when positive definite matrices are involved, is the pre-conditioned conjugate gradient (PCG) algorithm. In chapter III, PCG with classical incomplete Cholesky (CIC) pre-conditioning was shown to be remarkably efficient in the load flow problem. Specifically, it was demonstrated that PCG with CIC pre-conditioning can yield substantial gains in the computational performance of the Fast Decoupled load flow algorithm [29].

So far, the eigenvalue clustering of power network matrices using specialized pre-conditioning schemes has not been extensively investigated or

exploited. This chapter presents the results of a comparative investigation into the potential of different pre-conditioning matrices. The clustering efficiency of two competitive pre-conditioning schemes are compared with and ranked against that of the CIC approach. Experiments based on IEEE power networks of 14, 30, 57 and 118-bus systems show that superior pre-conditioning can be achieved if the intrinsic properties of the power networks are exploited. Thus, faster convergence for PCG can be obtained, which in turn, will reduce the CPU time required for the solution of power system problems.

4.2 Classical Incomplete Cholesky Pre-conditioning Matrix

A highly effective pre-conditioning matrix K (see 2.7.2.3) can be obtained using approximations of the Cholesky factor L, where,

$$A = L\,L^T \tag{4.1}$$

In the common approach, called the classical incomplete Cholesky (CIC) decomposition [101], only the entries of L which correspond to non-zero entries of A are computed, thus preserving the original matrix sparsity structure. In large sparse systems, this approximation sharply reduces the computational effort required to find K as well as any operations involving K^{-1}. In addition, the storage requirements for K are drastically decreased. Experience in many fields, such as finite element electromagnetic analysis [28, 108] and power system analysis [29] indicates that this type of pre-conditioning matrix K, although much cheaper to compute than L, is nevertheless a very good pre-conditioning matrix for A and clusters its eigenvalues efficiently. However, thus far, not much success has been reported on modifications of the pre-conditioning matrix, K, designed to improve its PCG accelerating effect.

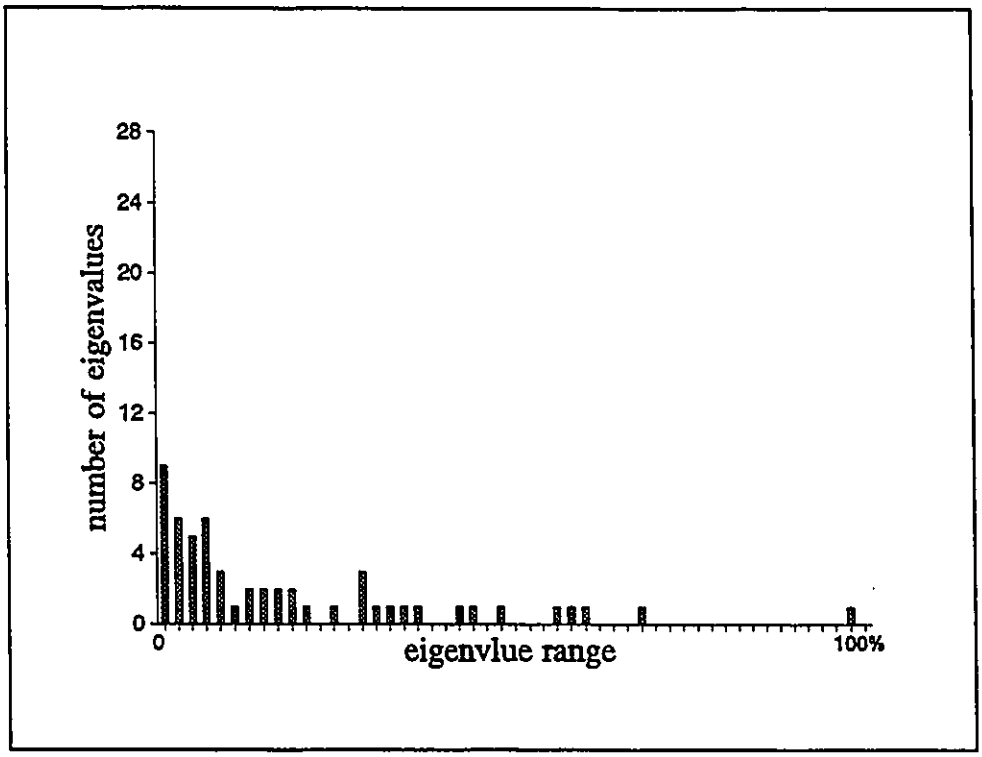

Figure 4.1: Eigenvalue distribution for IEEE-57 bus system without pre-conditioning

4.3 Clustering Effect of the CIC Pre-conditioning Matrix

Figures 4.1 and 4.2 illustrate the effect of the CIC pre-conditioning on the eigenvalue distribution of the B matrix for the IEEE 57-bus network. The eigenvalues are normalized with respect to their maximum and minimum levels. In this example, the effect of pre-conditioning is quite apparent, clustering the majority of the eigenvalues (24 out of 56) around the 30% level.

The number of iterations required for convergence of the conjugate gradient algorithm was measured for the solution of the system of linear equations $Ax=b$ in both the original form and the pre-conditioned case. In

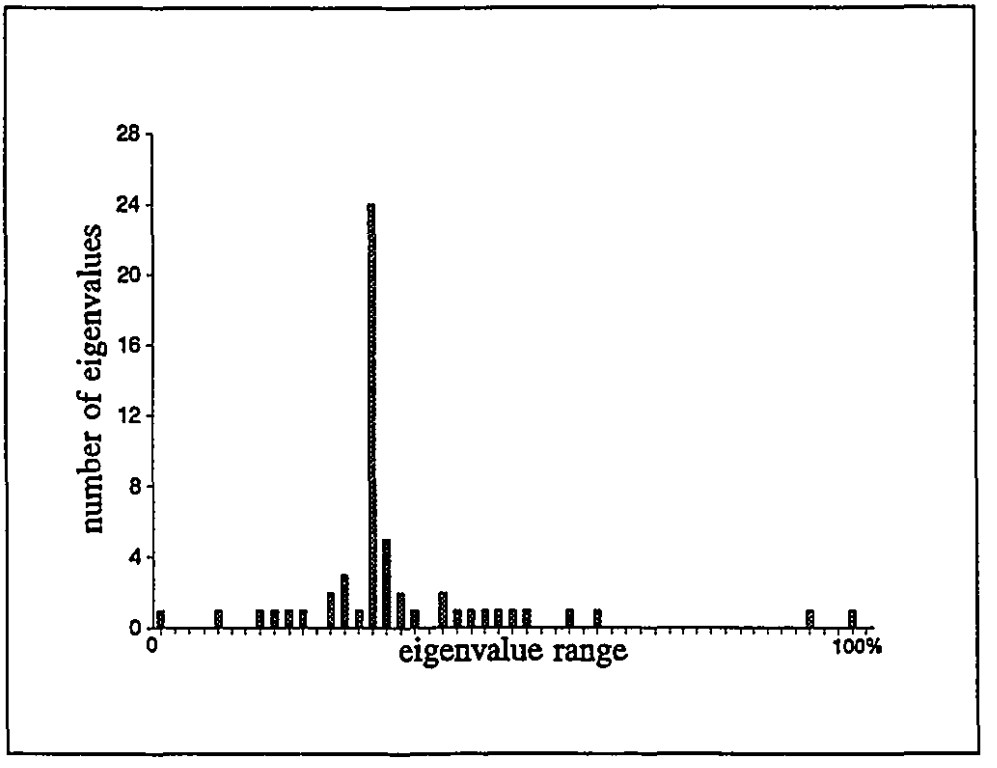

Figure 4.2: Eigenvalue distribution for IEEE-57 bus system with CIC pre-conditioning

Table 4.I: Number of iterations required for convergence of CG and PCG algorithms

IEEE example networks	Number of iterations	
	CG algorithm	PCG algorithm
5-bus system	4	1
14-bus system	11	6
30-bus system	23	9
57-bus system	41	11
118-bus system	68	22

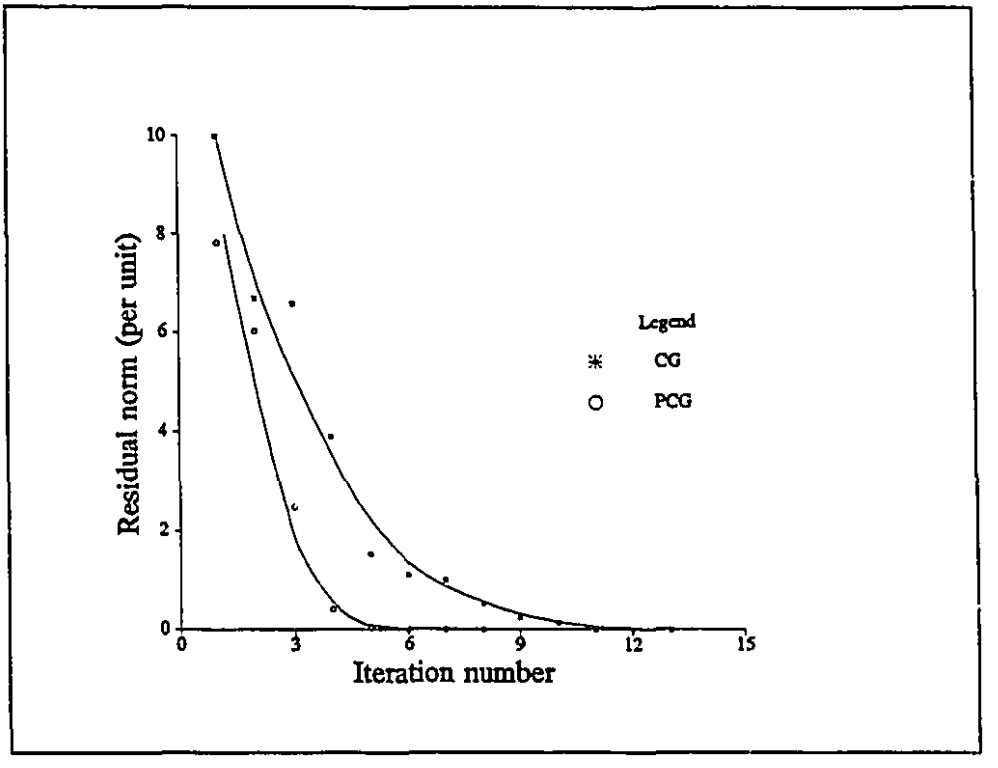

Figure 4.3: Convergence behaviour of CG and PCG algorithms for IEEE 14-bus network matrix

these examples, the A matrix corresponded to the B matrix (see chapter III) of the standard IEEE power networks [30]. Table 4.I shows the number of iterations required for these cases to converge to a solution resulting to a residual smaller than 10^{-3}. As the results indicate, the number of iterations needed for convergence in the pre-conditioned case compared with the general CG, tends to decrease as the system size increases. For the IEEE 5-bus network, the PCG algorithm with CIC pre-conditioning converges in one iteration to the exact solution. This is due to the fact that, in this case the CIC factor is equal to the complete Cholesky factor.

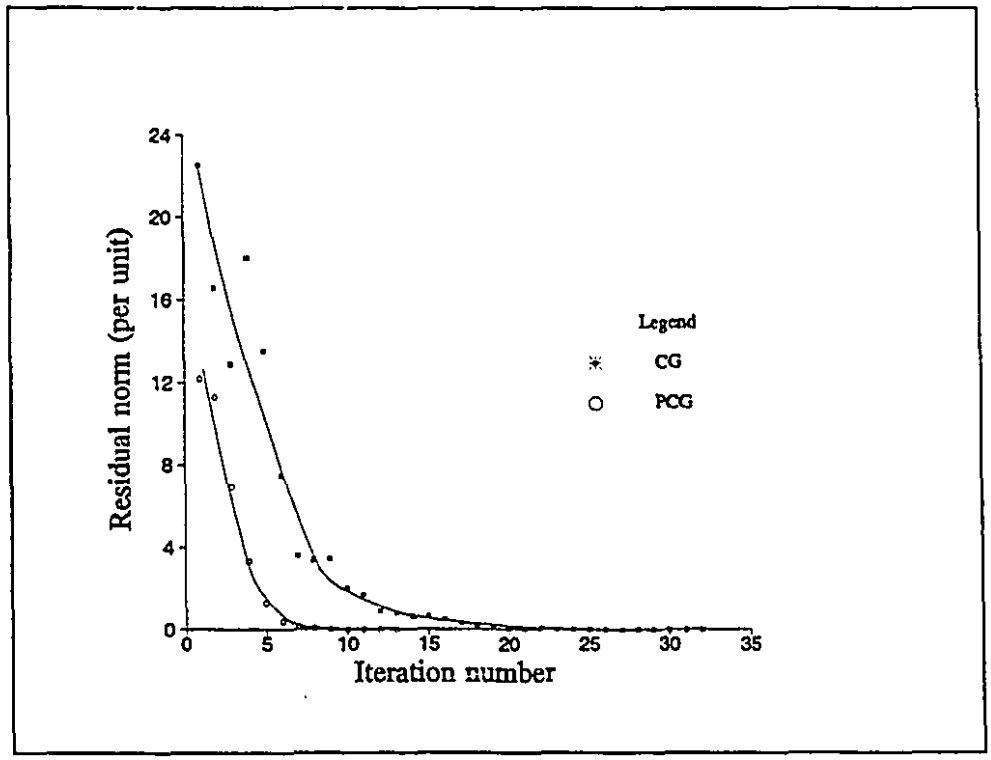

Figure 4.4: Convergence behaviour of CG and PCG algorithms for IEEE 30-bus network matrix

4.4 Investigation of the Convergence Properties of CG and PCG with CIC

To compare the convergence properties of CG and PCG algorithms with CIC pre-conditioning, the impact of the number of conjugate gradient steps on the norm of the residual vector for the solution of $Ax=b$, was numerically investigated for the IEEE example networks of 5, 14, 30, 57 and 118 buses. It was observed that all the experimental cases show a similar convergence pattern. In all test cases the residual corresponding to PCG with CIC pre-conditioning descends toward zero much faster than the CG algorithm. As examples, the norm of residual versus iteration number are plotted in figures 4.3 to 4.5 for IEEE 14, 30 and 57-bus systems.

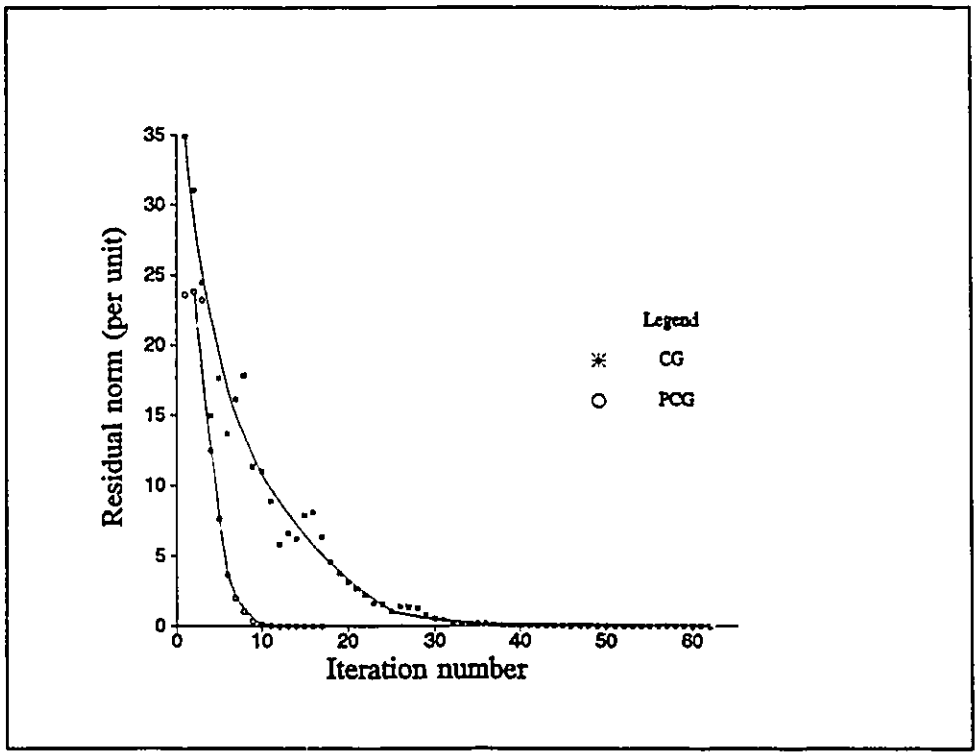

Figure 4.5: Convergence behaviour of CG and PCG algorithms for IEEE 57-bus network matrix

4.5 Experimental Comparison of CIC and Modified Incomplete Cholesky Pre-conditioners

4.5.1 Incomplete Cholesky modifications

In general, the true Cholesky factor is not as sparse as the original matrix, however, it has been observed that if only some important non-zero elements of the Cholesky factor are preserved, the resultant sparser matrix can still be a good pre-conditioning matrix.

Two classes of modifications are investigated. The first class of modifications puts more weight on those elements located near the diagonal

of the factor, while the second approach weighs more heavily the dominant elements of the factor regardless of their relative location. In both approaches, the clustering effect of the resultant pre-conditioning matrices are compared with that of CIC.

In all of the experiments, the matrices to be pre-conditioned, were the B matrices of the IEEE test networks.

All modifications were applied to IEEE standard network matrices in both scaled and non-scaled cases. The matrices are scaled such that they have unit diagonal elements. This type of scaling is sometimes called Evan's pre-conditioning [113]. To avoid confusion, it should be noted that Evan's pre-conditioning is intended to provide numerical stability to the PCG algorithm, and not to produce a clustering effect. The experimental results confirm that scaling does not affect the clustering condition, but only puts the eigenvalues in a tighter spectrum. However, to minimize stability problems in this investigation, these studies mainly focus on scaled matrices.

Both types of pre-conditioning modifications were compared with the CIC approach according to their clustering effects and convergence properties.

4.5.1.1 Diagonal dominant incomplete cholesky factor (DDIC):

In this class, the incomplete Cholesky factors are obtained by preserving only the non-zero elements of the complete Cholesky factors located within a specified diagonal band (nd).

Figures 4.6, 4.7 and 4.8 compare the eigenvalue spectrum of the pre-conditioned matrices obtained by preserving only the nd-diagonal non-zero elements. The number of non-zero elements preserved in each case is

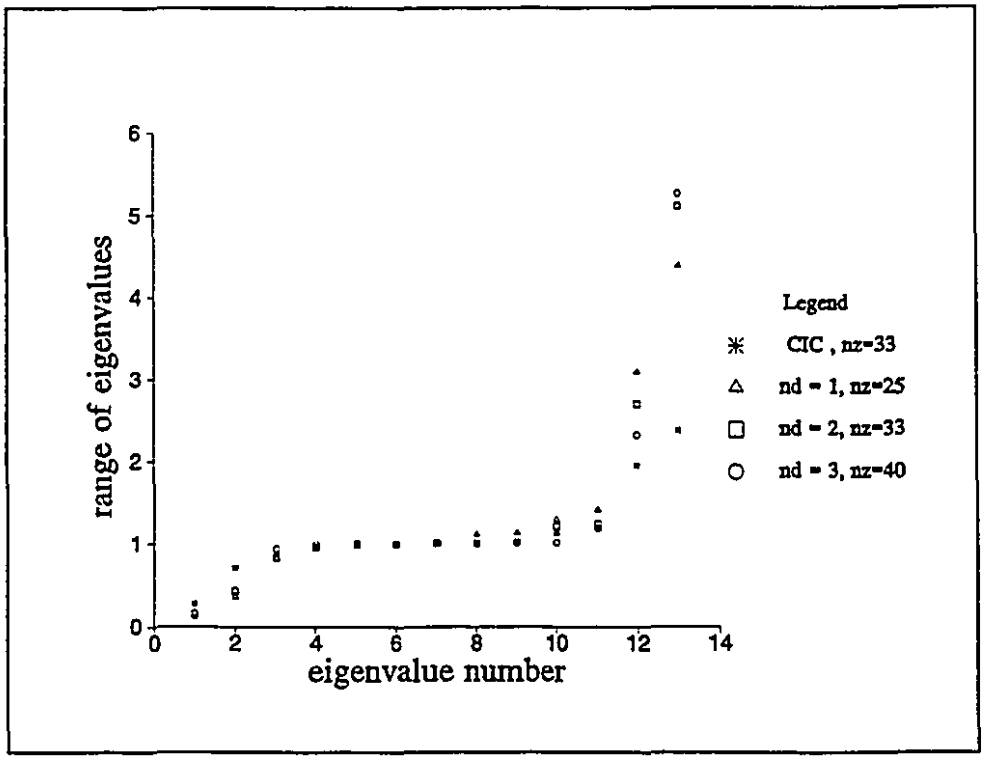

Figure 4.6: Comparison of eigenvalue range for IEEE-14 bus system (diagonal dominant pre-conditioning)

specified as nz. These test results show that as nd grows, the clustering effect improves and more eigenvalues group around the unity value. This is expected because, as the band of diagonals preserved increases, the approximated pre-conditioning matrix approaches the complete Cholesky factor L. However, this also implies that more computational effort is dedicated to the computation of the K factor. The results for all the systems considered also confirm that many of the eigenvalues of the pre-conditioned matrices tend to be clustered in one large group located around unity. This performance is similar to that of the CIC pre-conditioning. Table 4.II shows the total number of eigenvalues (and % in parenthesis) in the largest cluster (nc) located within a bandwidth of 1% of the eigenvalue range. The number of non-zero elements (nz) preserved in the pre-conditioning matrices is also shown for comparison. These numbers show that a modified Cholesky factor of this type, with almost

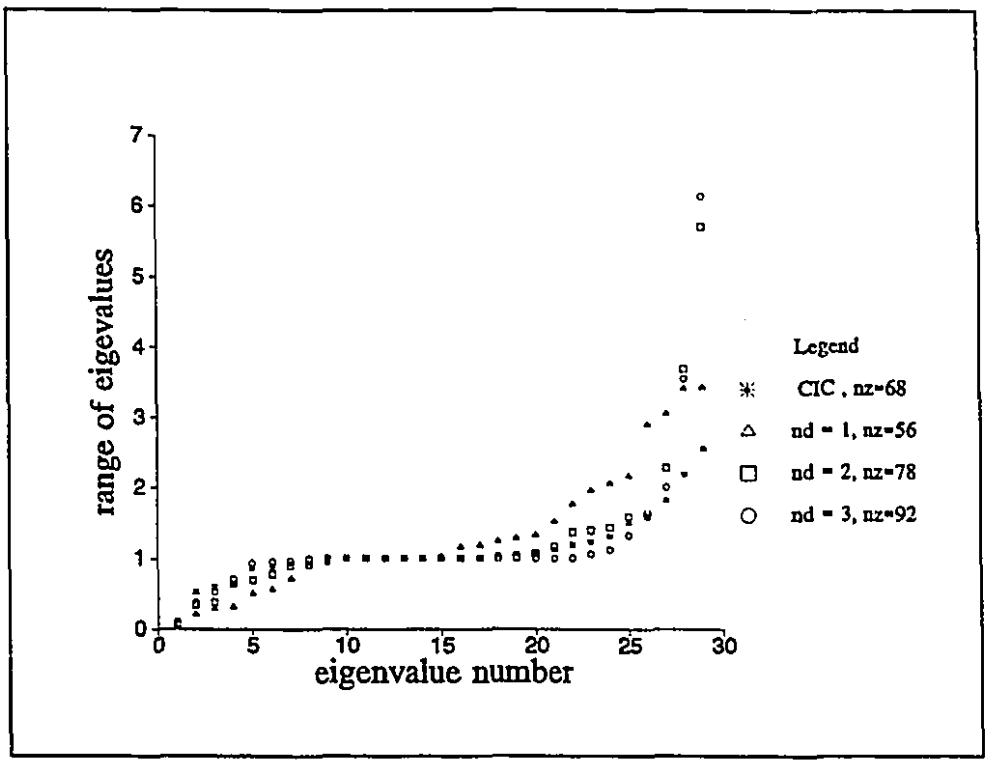

Figure 4.7: Comparison of eigenvalue range for IEEE-30 bus system (diagonal dominant pre-conditioning)

Table 4.II: Total number of eigenvalues clustered in the biggest group, for CIC and DDIC pre-conditioning matrices

Example networks		CIC	DDIC		
			nd = 1	nd = 2	nd =3
IEEE 14 buses	nc(%)	5 (38)	3 (23)	4 (31)	7 (54)
	nz	33	25	33	40
IEEE 30 buses	nc(%)	8 (28)	4 (14)	12 (41)	15 (52)
	nz	68	56	78	92
IEEE 57 buses	nc(%)	23 (41)	13 (23)	24 (42)	24 (42)
	nz	130	111	139	161

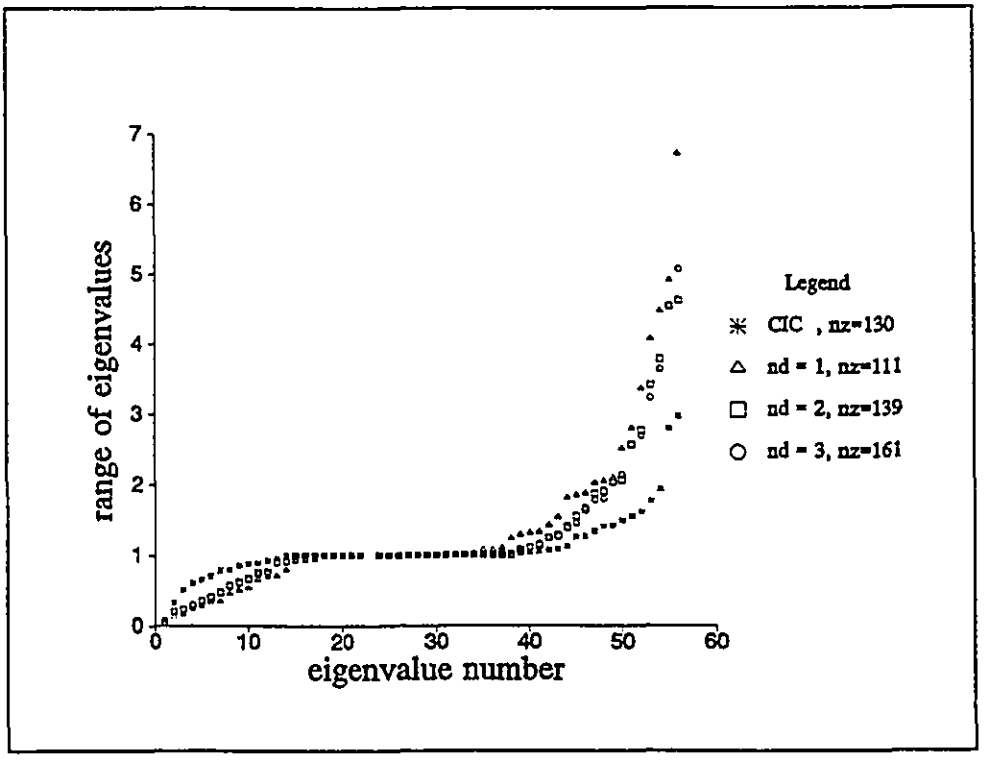

Figure 4.8: Comparison of eigenvalue range for IEEE-57 bus system (diagonal dominant pre-conditioning)

the same number of non-zero entries as a classical incomplete Cholesky factor, will provide a near identical clustering pattern. As observed in Table 4.II, for the IEEE 14-bus system with DDIC pre-conditioning, four eigenvalues are located in the largest cluster, while, for the CIC pre-conditioning with the same number of non-zero elements (33), this number (nc) will be five. This implies that this type of pre-conditioning may require more iterations for convergence when compared with the CIC pre-conditioning. Table 4.III shows the number of iterations required for the convergence of the PCG method with DDIC pre-conditioning compared with the number of iterations required with CIC pre-conditioning, for the IEEE 14, 30, 57 and 118-bus systems. In this table, nd and nz represent the number of diagonals preserved from the Cholesky factor and the total number of non-zero elements respectively. These data demonstrate that this type of pre-conditioning is not as efficient

Table 4.III: Number of iterations required for convergence of PCG

network	DDIC pre-conditioning			CIC pre-conditioning	
	nd	nz	number of iterations	nz	number of iterations
14-bus system	1	25	8	33	6
	2	33	7		
	3	40	7		
30-bus system	1	56	16	68	9
	2	78	12		
	3	92	10		
57-bus system	1	111	20	130	11
	2	139	17		
	3	168	17		
118-bus system	1	228	49	294	22
	2	303	38		
	3	356	33		

as the CIC pre-conditioning. As an indication, it can be easily seen that the PCG with DDIC pre-conditioning even with more non-zero elements than the CIC pre-conditioner, is not as fast as the PCG with CIC pre-conditioning.

4.5.1.2 Dominant element incomplete cholesky factor (DEIC):

This type of modified incomplete Cholesky factor involves measuring the relative importance of the dominant Cholesky factor elements. In this case, the modification is controlled by a weighting factor w. If the magnitude of an off-diagonal element in L is smaller than w, it is replaced by the 0 value.

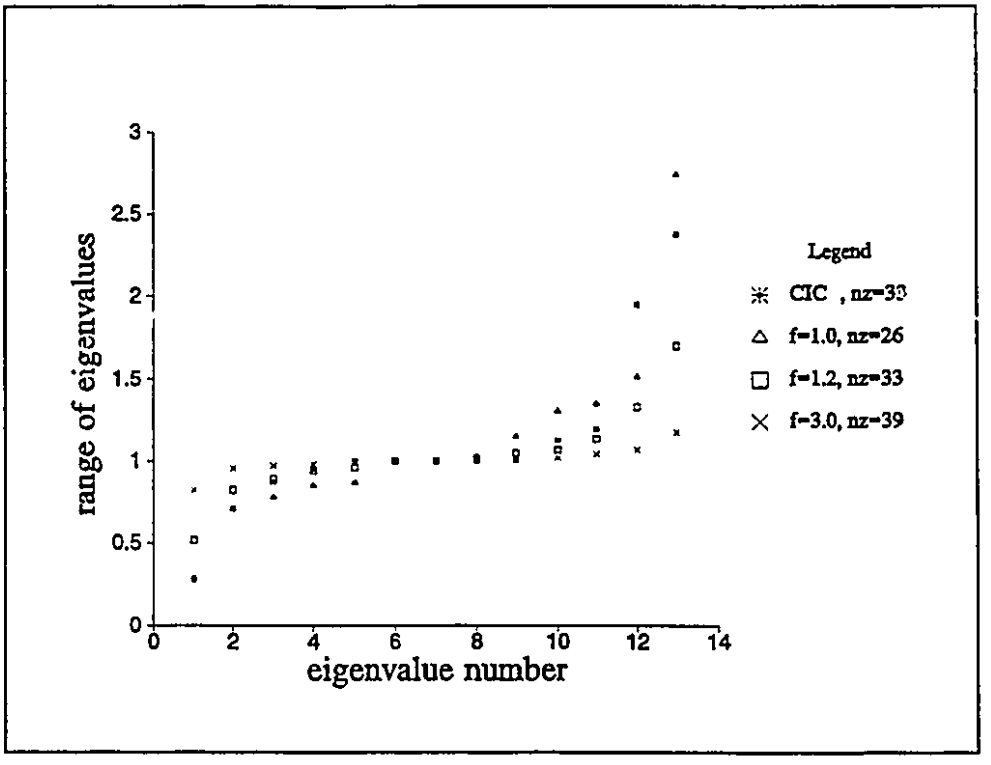

Figure 4.9: Comparison of eigenvalue range for IEEE 14 bus system (dominant element pre-conditioned case)

If l_{av} is the average value of the off-diagonal non-zero elements of L, then w is defined as:

$$w = \frac{l_{av}}{f} \tag{4.2}$$

where f is a scaling factor.

The pre-conditioning effect of incomplete factors of this type was examined for different scaling factors ranging from 0.4 to 3, for the IEEE 14, 30, 57 and 118-bus systems.

Table 4.IV illustrates the clustering effect for this class of Cholesky modifications. In this table the number of eigenvalues contained in the largest

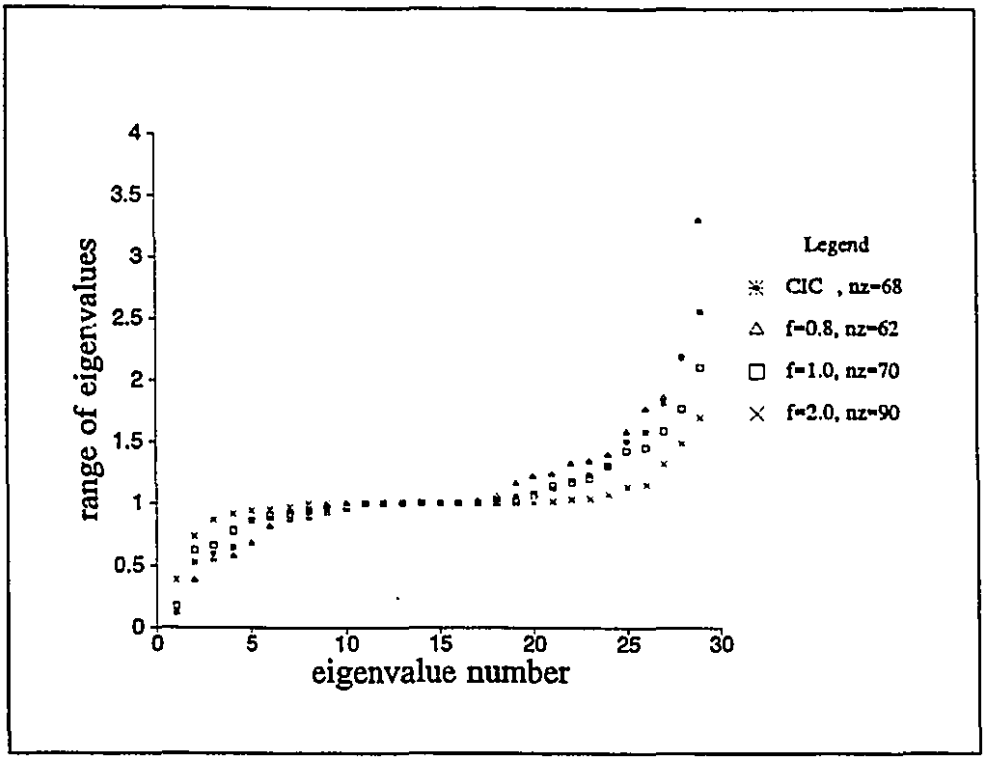

figure 4.10: Comparison of eigenvalue range for IEEE 30 bus system (dominant element pre-conditioned case)

cluster, located within a bandwidth of 1% of the eigenvalue range, are shown for scaling factors f varying from 0.6 to 1.8. These numbers imply that better clustering is achieved as f increases.

Table 4.IV: Total number of eigenvalues clustered in the biggest group, for DEIC pre-conditioning matrices

Scaling factor f	Number of eigenvalues in biggest cluster		
	IEEE 14-bus	IEEE 30-bus	IEEE 57-bus
0.6	1	5	22
1	2	8	23
1.8	4	12	25

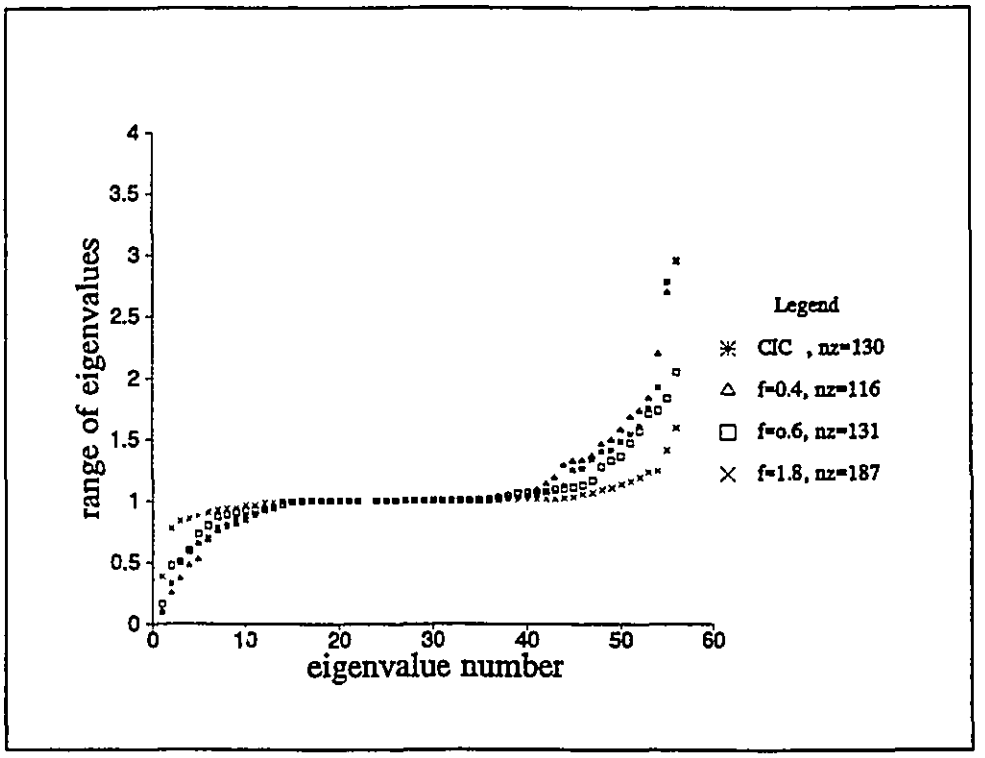

Figure 4.11: Comparison of eigenvalue range for IEEE 57 bus system (dominant element pre-conditioned case)

Figures 4.9, 4.10, and 4.11 illustrate some examples of the clustering effect provided by DEIC pre-conditioning for the IEEE 14, 30 and 57-bus network matrices. They show that a larger scaling factor f (more computed non-zero elements), yields a better clustering effect. As it is shown later in this section, this also results in faster convergence.

Tables 4.V to 4.VII show the number of iterations required for the convergence of the PCG algorithm with this type of pre-conditioning for the IEEE 30, 57 and 118-bus systems as an indication. In these tables the number of non-zero elements of the pre-conditioning matrices are shown for scaling factors f varying from 0.4 to 3. These data demonstrate that, as the clustering effect improves, the number of iterations required for convergence decreases.

Table 4.V: Number of iterations required for convergence of PCG (IEEE 30-bus network)

pre-conditioning matrix (K)		number of non-zeros	number of iterations
CIC		68	9
DEIC	f=0.5	53	11
	f=0.8	64	9
	f=0.9	65	8
	f=1.0	70	8
	f=2.0	93	5
	f=3.0	105	4

Figures 4.9, 4.10 and 4.11 also indicate that, for the same number of non-zero elements, DEIC pre-conditioning yields better clustering than the classical incomplete Cholesky method. To show this advantage more clearly, the number of iterations required for convergence of the PCG with classical incomplete Cholesky pre-conditioning is compared with those required with DEIC pre-conditioning including fewer or equal non-zero elements in Table

Table 4.VI: Number of iterations required for convergence of PCG (IEEE 57-bus network)

pre-conditioning matrix (K)		number of non-zeros	number of iterations
CIC		130	11
DEIC	f=0.4	114	11
	f=0.6	130	9
	f=1.8	187	5

Table 4.VII: Number of iterations required for convergence of PCG (IEEE 118-bus system)

pre-conditioning matrix (K)		number of non-zeros	number of iterations
CIC		294	22
DEIC	f=0.4	255	21
	f=0.7	326	13
	f=1.0	371	10
	f=1.4	417	9
	f=1.7	450	8
	f=2.0	477	7
	f=3.0	520	7

4.VIII. It is noteworthy to mention that the cost of computing the pre-conditioning matrices roughly corresponds to the number of non-zero elements computed. As indicated in this table for the IEEE 57-bus system, the PCG with DEIC pre-conditioning, with the same number of non-zero elements as the CIC pre-conditioner, converges faster than the PCG with CIC

Table 4.VIII: Number of iterations required for the convergence of the PCG with CIC and DEIC pre-conditionings

IEEE networks	number of non-zeros		number of iterations	
	CIC	DEIC	CIC	DEIC
30-bus	68	65	9	8
57-bus	130	130	11	9
118-bus	294	255	22	21

pre-conditioning. Furthermore, even in the case where the DEIC pre-conditioner has fewer non-zeros than the CIC pre-conditioner, the number of iterations required for convergence is still lower for the DEIC case (IEEE 30 and 118-bus systems).

This result can be interpreted by the fact that the classical incomplete factorization corresponds to a $L L^T$ factorization only in those locations where B has non-zero elements (b_{ij} is non-zero). These locations imply that buses i and j are connected to each other directly. However, this approach can ignore some dominant elements in the Cholesky factor and preserve some trivial ones. It can be seen that L_{ij} will be dominant if there are some indirect connections with low impedances between buses i and j through other buses, even though b_{ij} is zero. On the other hand, L_{ij} will be trivial if the line connecting i and j has a high impedance. This is because the $L L^T$ factorization process is essentially a sequence of linear transformations for the elimination of the lower triangular (upper triangular) elements of B to be able to perform backward (forward) substitution. In this process, to eliminate b_{ij} from the ith row, $-b_{ij}/b_{kk}$ times of kth row ($b_{kl} = 0$ for $l < k$) is added to the ith row of the B matrix. This process fills any zero position (i,m) of the ith row, with the non-zero element $-(b_{ij}\, b_{km})/b_{kk}$ if b_{km} is not equal to zero resulting in a new non-zero location. In general, as b_{ij} is small compared with b_{kk}, L_{mi} will be small. However, if b_{ij} and b_{km} are not small compared with b_{kk} or, if there are many lines connected to bus i or m, resulting in a high admittance between i and m, then L_{im} will not be trivial, and may be a dominant element in the Cholesky factor.

The results of these studies indicate that in general, CIC is an effective approximation to the Cholesky factor for accelerating PCG convergence. However, these results also show that by making use of the topological configuration of power networks, modified Cholesky factors with the same

sparsity degree can be constructed giving better clustering and faster convergence of the PCG algorithm.

4.6 Concluding Remarks

(1) Pre-conditioning matrices based on approximations to the Cholesky factor seem to be a promising approach to obtain a better clustering of the system eigenvalues.

(2) Comparison of the clustering effect of the classical incomplete Cholesky (CIC) factor, with those of diagonally dominant incomplete Cholesky (DDIC) factors indicate that the DDIC pre-conditioning is not as efficient as CIC pre-conditioning if an equal number of non-zero elements are preserved in each factor (equal computational effort is dedicated to both).

(3) Pre-conditioning with the dominant element incomplete Cholesky (DEIC) factor can yield an improved clustering of the system eigenvalues and, thus, faster convergence for PCG, when the physical properties of the network matrices are taken into account to construct the pre-conditioning matrix.

(4) If solution results of lower accuracy are acceptable, a small number of iterations is sufficient for convergence. This is an interesting property of iterative techniques which can be utilized in certain types of power system algorithms such as contingency ranking.

(5) The experimental results show that pre-conditioning of power network matrices with any of the incomplete Cholesky factors (CIC, DDIC or DEIC) will cluster many of the eigenvalues in a large group located around unity.

CHAPTER
FIVE

GENERALIZATION OF THE PRE-CONDITIONED CONJUGATE GRADIENT METHOD FOR COMPLEX MATRICES ARISING IN POWER SYSTEM ANALYSIS

5.1 Introduction

It was experimentally demonstrated in chapter III that the pre-conditioned conjugate gradient (PCG) algorithm can be significantly more efficient than direct techniques and can offer important savings in CPU time for the load flow problem. This conclusion was shown to hold for power system algorithms requiring repeated solutions of large sparse systems of linear equations with positive definite matrices. However, it is known that other important power system analysis problems such as transient stability and short circuit studies essentially depend on the repeated solution of systems of linear equations whose coefficient matrices are complex (see chapter II) and similar to the network admittance matrix.

In this chapter, an investigation of the potential of PCG solvers for complex admittance power network matrices is presented. In section 5.2, the

modified complex PCG method is applied to the IEEE test networks (14, 30, 57 and 118-bus systems) for the solution of systems of linear equations of the form Yx=b, where Y is the complex admittance network matrix.

In section 5.3, the complex PCG algorithm and the modified complex direct solvers are applied to synthetically generated networks of large sizes. The computation time of the complex PCG is compared with that of the standard direct solver. The comparison is performed for a wide spectrum of power networks of different topologies up to 6500 buses and 13000 lines. Furthermore, the effect of the size of the largest block in the network as well as the effect of the size of the network, on the performance of the two solvers are experimentally investigated. The results of our experiments show that the complex PCG algorithm converges to the correct solution for all experimental cases. This comparison also indicates that, for certain classes of large sparse systems, especially when repeated solutions with matrix modifications are required, the new PCG is significantly more efficient than the direct technique.

5.2 Application of the PCG Method to Complex Network Matrices

As explained in chapter II, many power system algorithms such as short circuit studies and transient stability analysis rely on the repeated solutions of systems of linear equations whose coefficient matrices are complex admittance matrices. In certain types of algorithms such as the transient stability problem, these types of linear systems of complex equations are much larger than the network dimension and need to be solved repeatedly. To limit the huge computational time of the factorization process in these algorithms, the coefficient matrix is updated and refactored only every few iterations [13]. This, in turn, may result in a higher number of iterations for the whole algorithm to converge. Therefore, we were motivated to examine the complex PCG algorithm as an alternative solver where a new updated admittance

matrix is applied at every integration step and every iteration of the stability algorithm.

The complex PCG algorithm is essentially identical to the real case used in Chapter III except that all numbers are complex. While experience in numerical electromagnetics suggests that CG algorithms for indefinite, non-symmetric and complex systems [122-127] are computationally too expensive to compete with direct methods, experimental tests with complex symmetric systems, have shown very encouraging results (Table 5.I).

As a first set of tests, numerical results applied to the IEEE standard networks showed convergence to the exact solutions for all cases. Table 5.I gives information on the number of iterations required for the solution of systems of linear equations $Yx' = b'$ and $Bx = b$, where Y and B are complex and real matrices respectively. These results suggest that:

1) Although the required number of iterations for convergence of the PCG in the complex case is a little higher than that of the real case, it still can be effectively applied to systems of complex linear equations arising in power system analysis.

Table 5.I: Number of iterations required for convergence of CG and PCG algorithms for the solution of systems of linear equations

IEEE test networks	Complex		real	
	CG	PCG	CG	PCG
14-bus	12	7	10	5
30-bus	20	11	18	7
57-bus	42	16	32	10
118-bus	55	23	57	19

2) Based on the comparison of the number of iterations required for PCG and CG to converge, it is expected that PCG would outperform direct solvers for complex matrix equations corresponding to large sparse power networks. This, then, may result in substantial computational performance improvement specially for the transient stability problem.

Due to these results, the performance of the PCG algorithm was next compared with a complex Frontal based direct solver for large sparse admittance matrices of synthetically generated networks of different sizes and varying topologies. Again, although the theory predicts PCG convergence only for positive definite matrices, it was nevertheless observed in all the experimental cases that the PCG converges to the right solution in a competitive number of iterations.

The admittance matrix Y, in essence, can be written as the summation of two real matrices G and B where,

$$Y = G - J\,B \tag{5.1}$$

where each of these two real matrices have similar topologies with dominant elements located on the major diagonal. Although, this property does not prove convergence, it may suggest an explanation for the observed convergence of the algorithm in all tested cases.

5.3 Numerical Comparison on the Performance of PCG and Frontal Solvers for Complex Network Matrix Equations

In this section, the performance of the PCG with classical incomplete Cholesky pre-conditioning is experimentally compared with that of the Frontal (Sparspak does not have a complex direct solver) direct solver [60]. In this

vein, first both algorithms were modified to complex solvers. Then, they were applied to solve systems of complex linear equations similar to those arising in power system problems such as short circuit and transient stability algorithms. The computation time of the two solvers were then compared with each other for a wide spectrum of network admittance matrices up to 6500 buses and 13000 lines. The impact of the network size and topological connectivity on the relative performance of the two solvers were also numerically investigated.

To compare the performance of the two solvers, the ratio of the corresponding CPU times is displayed as a function of the network size in Figures 5.1 to 5.4. The results indicate that the PCG can be advantageously applied to certain types of complex linear systems of equations arising in power system studies.

5.3.1 Test networks

In these series of experiments, the networks were composed of sparse blocks of varying sizes. These blocks, themselves, are interconnected by a sparse network. The buses in each block are connected to each other in a random fashion and have the star-type connection explained in chapter III. Each bus in the network is (on average) connected to four transmission lines and thus, the total number of the lines in the network add up to about two times of the total number of the buses in the network (Further information on the test networks can be found in chapter VI). The test network generated for these series of experiments range up to 6500 buses and 13000 lines.

It was described in chapter II that the Frontal solver is a bandwidth based solver. Thus, to ensure that the network data are not biased against the direct solver, the input data (admittance matrices) were generated such that the matrix has its minimum possible bandwidth around the major diagonal.

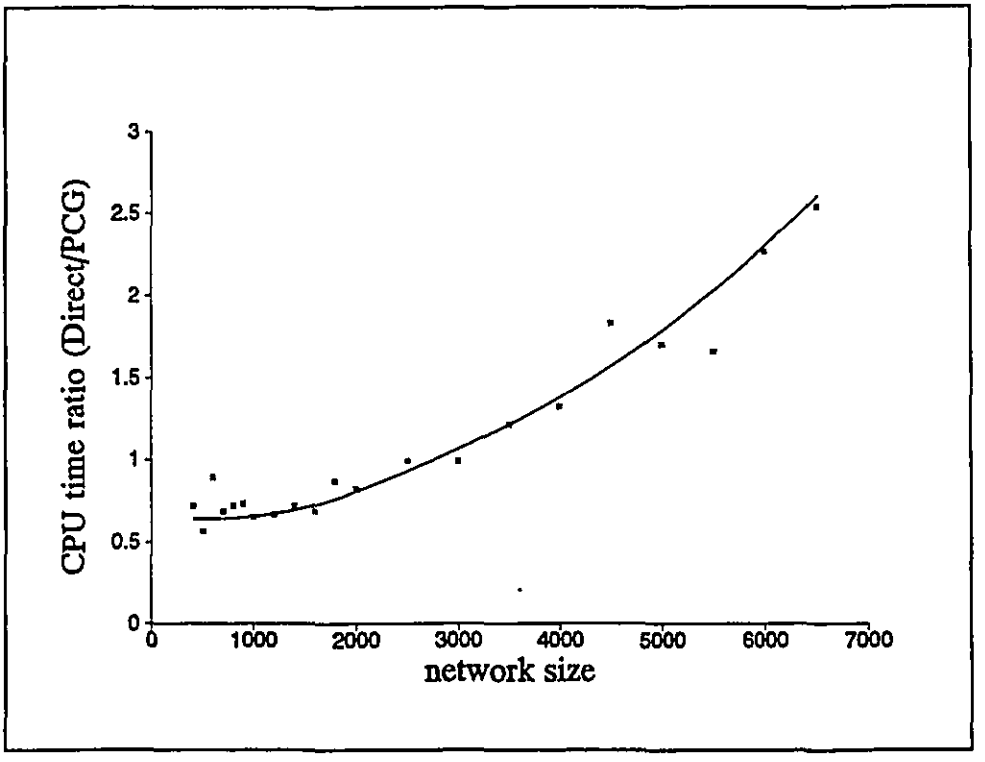

Figure 5.1: CPU time ratio of direct and PCG complex solvers versus the network size
(Largest sparse block has 100 buses)

5.3.2 Discussion of the results

The results of this section are grouped into four different network categories defined by the size of the largest sparse block in the network. In fact, the effect of the size of the network as well as the effect of the largest block in the network (matrix bandwidth) on the performance of the two solvers are evaluated. For each category, the ratio of the CPU time of one complete solution by direct solver and the CPU time of one complete solution by the PCG solver is plotted versus the network size in Figures 5.1 to 5.4.

A) In this group, the effect of the size of the network on the performance of the two solvers is investigated. This investigation is performed based on the evaluation of the performance of the two solvers on networks

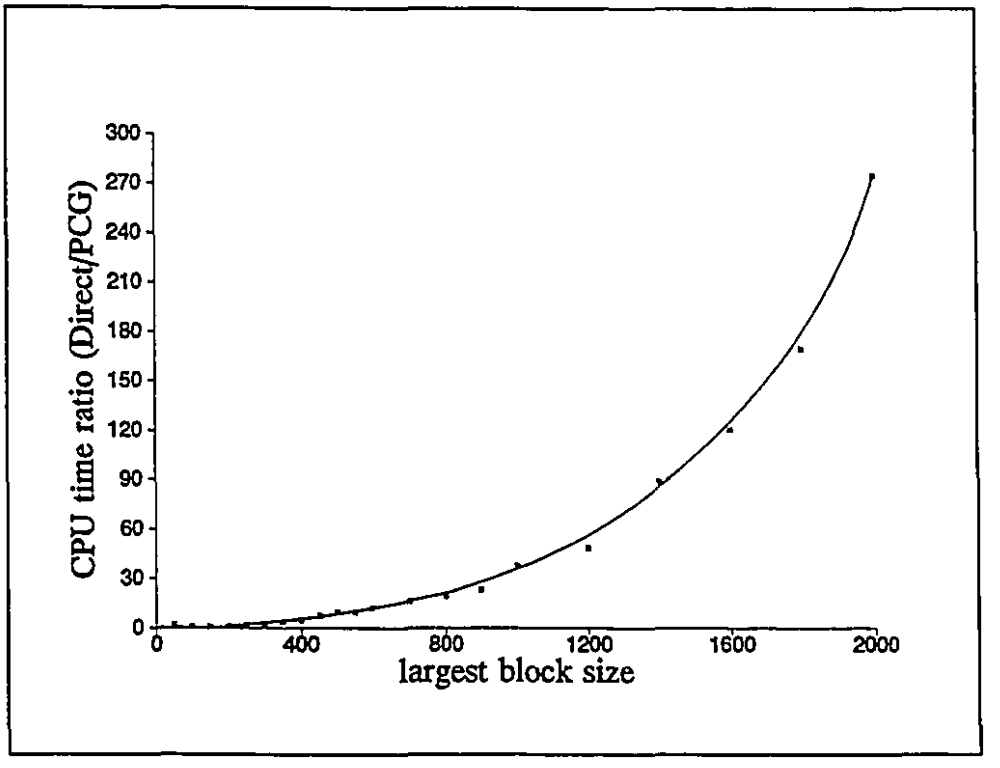

Figure 5.2: CPU time ratio of direct and PCG complex solvers versus the size of the largest sparse block in the network
(Each network contains 2000 buses and 4000 lines for all cases)

of varying sizes whose largest sparse blocks contain only one hundred buses of the network. Other blocks in the networks have random sizes varying from 6 to 100 buses. The CPU time ratio for the two solvers in this group of experiments are plotted in Figure 5.1.

Comparison of the two methods for the solution of the systems of complex linear equations indicate that for networks with sizes smaller than 3000 buses, the direct solver is advantageous, while, for the networks of larger sizes, the PCG shows a speed advantage over the direct solver. However, for cases involving many repeated solutions with a constant coefficient matrix, this advantage will be reduced. This is due to the fact that for these types of problems, only one factorization is necessary for all the repeated solutions.

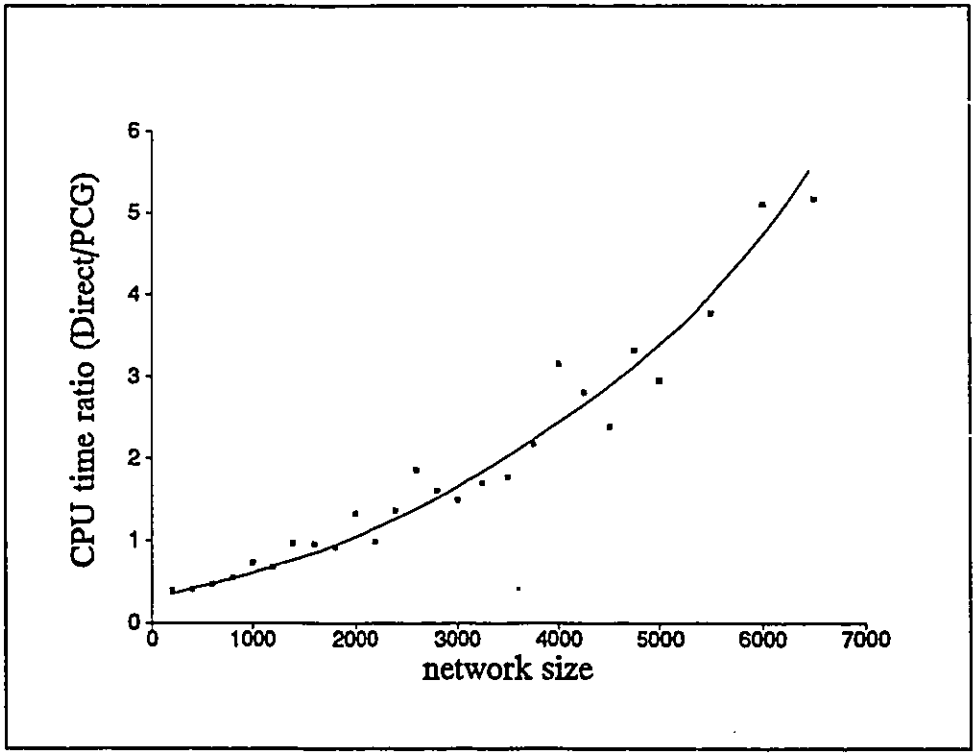

Figure 5.3: CPU time ratio of direct and PCG complex solvers versus the network size
(Largest sparse block is one tenth of the network size)

B) In the second category, the effect of the size of the largest block (matrix-bandwidth) in the network is investigated. Thus, the solvers were applied to networks of equal sizes containing 2000 buses, while the sparsity degree is the same for all these networks. Figure 5.2 shows the CPU time ratio for the two solvers versus the size of the largest block.

As indicated in Figure 5.2, for this type of network, the relative efficiency of the PCG over direct solver increases very rapidly with the size of the largest block. The cross-over point where the PCG becomes faster than direct methods starts near a block size of 200. As an example, for a 2000-bus network composed of only one block, the PCG solver is about 275 times faster than the direct solver.

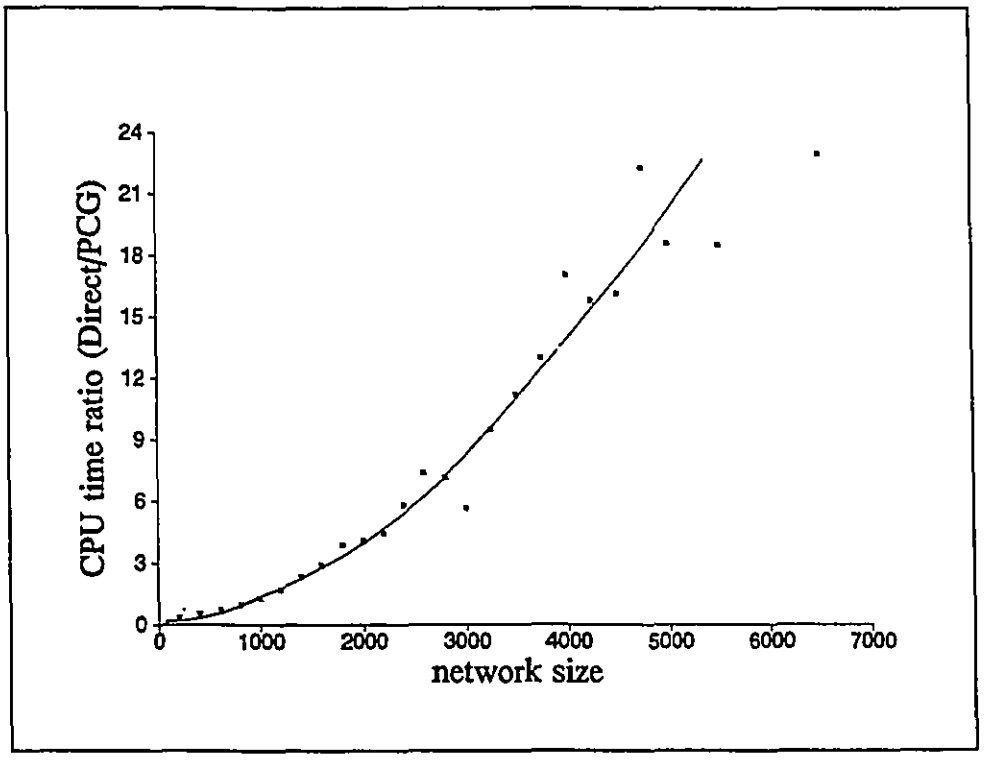

Figure 5.4: CPU time ratio of direct and PCG complex solvers versus the network size
(Largest sparse block is one fifth of the network size)

C) In the third category, networks with the largest sparse block equal to one tenth of the size of the network were investigated. Figure 5.3 shows the relative CPU times of the two solvers.

We see here that, for systems of more than about 2000 buses where, the size of the largest block is about 200, the PCG becomes more efficient. For a system of 6500 buses of this type of networks, the PCG shows a speed advantage about five times that of the direct solver.

D) In the last group of experiments, the PCG and Frontal solvers were applied to networks of varying sizes whose largest block has a size equal to one fifth of the network size. The test results for these cases are plotted in Figure 5.4. This group of networks is similar to the one described in the third category except that the largest block is twice the size.

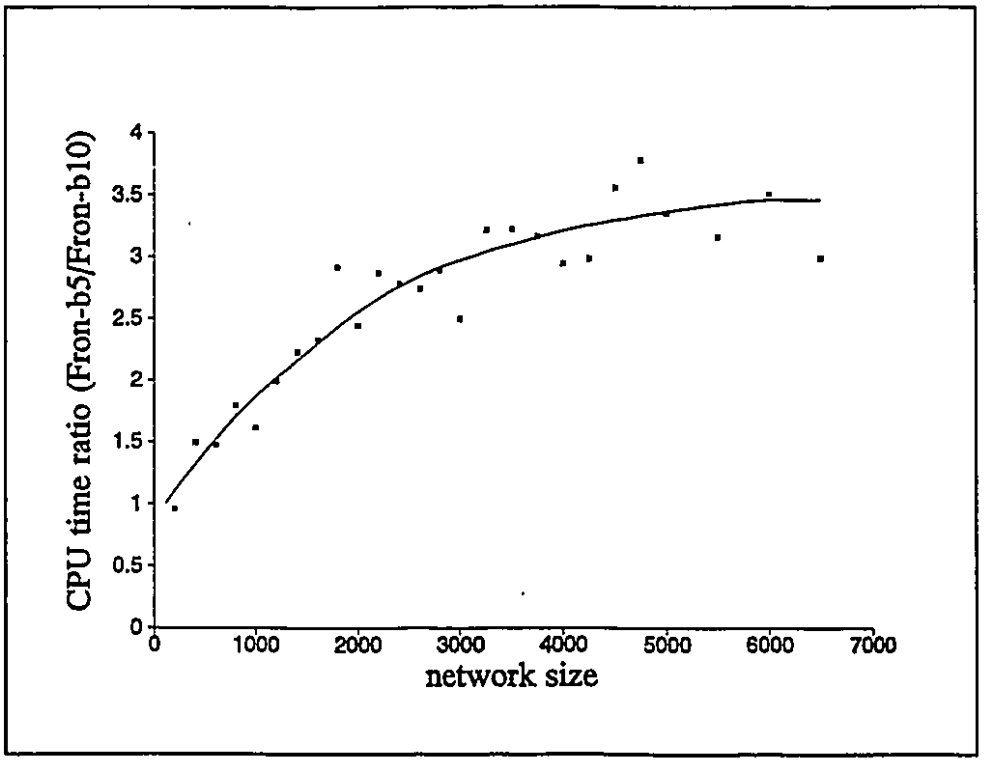

Figure 5.5: CPU time ratio of the Frontal solver for the solution of networks whose largest block is one fifth of the network size over those whose largest block is one tenth of the network size

In this case, the PCG method is considerably more efficient than the direct method compared with the previous case and becomes more efficient than the direct solver for systems larger than about 800 buses.

Figures 5.5, 5.6, 5.7 and 5.8 give a better understanding of the effect of the largest block in the network on the relative behaviour of the two solvers.

Figure 5.5 measures the effect of the largest block in the network on the performance of the direct solver for network matrices of the same size and sparsity degrees. This effect is shown in the figure by the CPU time ratio of the direct solver for networks whose largest block is one fifth the size of the network (Fron-b5) over those whose largest block is one tenth of the

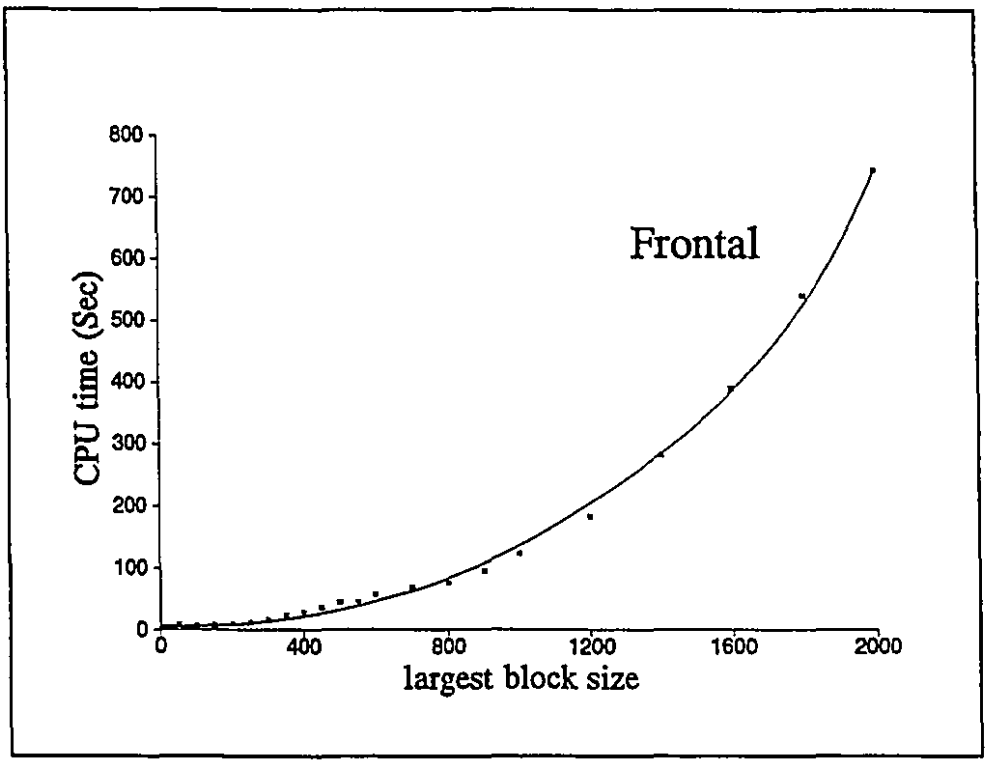

Figure 5.6: CPU time (Sun SPARC station 10, 64 Mb RAM) for the solution by direct solver versus the size of the largest sparse block in the network (Each network contains 2000 buses and 4000 lines for all cases)

network size (Fron-b10). It is illustrated that although this time ratio is about one for small networks, it approaches four for very large networks. The latter is a predicted value for the Frontal solver by the theory which establishes that the CPU time for the solver is $O(nm^2)$ where, m is the size of the largest block (matrix bandwidth). Figure 5.6 supports this result. This figure shows the CPU time (on a Sun station SPARC 10) required by the Frontal solver for the solution of network equations of the same size (2000 buses) and sparsity degree (4000 lines) whose largest blocks vary in size. As illustrated in this figure, for networks of constant sizes, the solution time increase is almost $O(m^2)$.

On the other hand, Figures 5.7 and 5.8 evaluate the effect of the largest block on the performance of the complex PCG solver for matrix

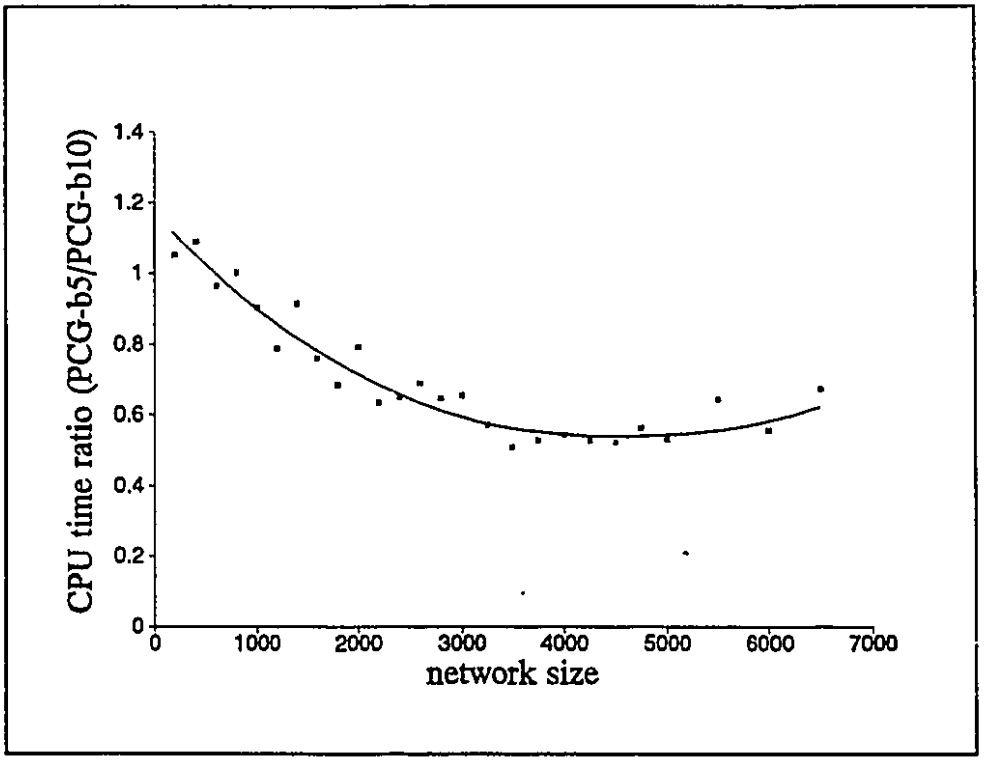

Figure 5.7: CPU time ratio of the PCG solver for the solution of networks whose largest block is one fifth of the network size over those whose largest block is one tenth of the network size

equations of power networks. Figure 5.7 shows the CPU time ratio of the complex PCG solver for those networks whose largest block is one fifth of the size of the network (PCG-b5) over those whose largest block is one tenth of the network size (PCG-b10). It is observed that, contrary to the direct solver, the ratio does not increase. In fact, as Figure 5.7 shows, this ratio decreases as network size increases. As an example, the PCG solution speed for a network of 6500 buses with a largest block of 1300 buses will be about 33% faster than for a network of the same size and sparsity degree whose largest block contains only 650 buses. The explanation for this surprising result may be seen in Table 5.II which shows the number of PCG steps for these two sets of networks. From this table, it can be observed that, in case of large networks the number of PCG iterations for a network whose largest block is one fifth of the network is generally smaller than that of a network with a

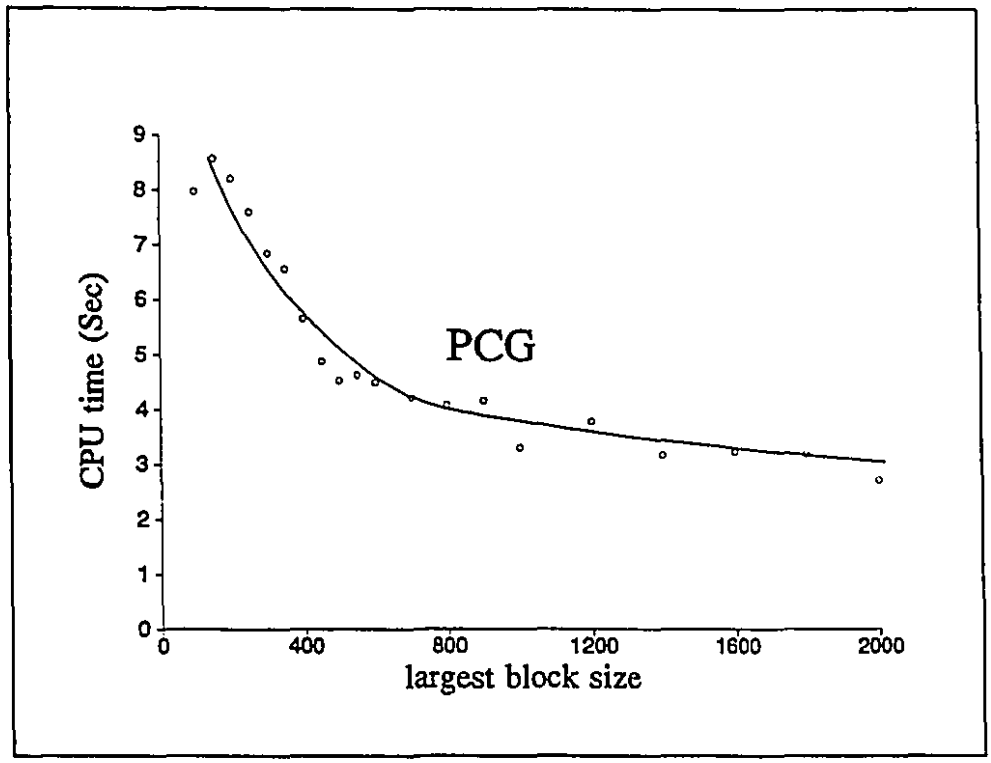

Figure 5.8: CPU time (Sun SPARC station 10, 64 Mb RAM) for the solution by PCG versus the size of the largest sparse block in the network (Each network contains 2000 buses and 4000 lines for all cases)

largest block of one tenth the network size. It is clear that this behaviour implies that the larger the block the better the clustering effect of the pre-conditioning. However, it is still unclear why this phenomenon occurs.

It was also observed, that for networks having a large bandwidth (largest block), the number of PCG steps tends to be much smaller than the anticipated value of $O(n^{0.5})$. This, in turn, indicates that for very large networks whose largest blocks are not very small, the PCG method will be faster than expected. To illustrate this conclusion, the number of actual iterations required for convergence of the PCG for networks of 2000 buses whose largest blocks vary in size are shown in Table 5.III. These range from 66 to 12 iterations, depending on the size of the block, compared to the anticipated value of 45.

Table 5.II: Number of iterations required for convergence of complex PCG solver for networks of different sizes

Size of the network	Size of the largest block in the network	
	one tenth the network size	one fifth the network size
200	22	25
400	31	33
600	39	35
800	37	40
1000	47	44
1200	55	42
1400	55	43
1600	61	42
1800	71	44
2000	61	45
2200	81	49
2400	74	43
2600	74	45
2800	76	45
3000	90	54
3500	102	44
4000	80	40
4500	102	50
5000	99	48
5500	91	55
6000	92	46
6500	95	58

Table 5.III: Number of iterations required for convergence of complex PCG solver for 2000-bus networks whose largest blocks vary in size

Size of the largest block	Number of iterations	Size of the largest block	Number of iterations
200	66	1200	21
400	41	1400	17
600	29	1600	16
800	25	1800	17
1000	20	2000	12

The CPU time for the solution of this type of network is plotted in Figure 5.8. It is illustrated in this figure that the CPU time required by the PCG solver for the solution of networks of the same size whose largest blocks vary in size, exponentially decreases as the size of the largest block increases. As an example, while the time for a network of 2000 buses whose largest block contains only 200 buses is 8.21 seconds, it decreases to 2.71 seconds for a 2000-bus network composed of only one block.

5.4 Concluding remarks

(1) The incomplete Cholesky pre-conditioned conjugate gradient (PCG) algorithm can be efficiently applied to solve systems of complex linear equations whose coefficient matrices have the same topology as the admittance matrix. These types of equations arise in power system analysis such as short circuit and transient stability studies.

(2) Our experimental evaluation of the performance of the modified complex PCG solver as compared with that of the Frontal direct solver for networks

of up to 6500 buses and 13000 lines indicates that, for large sparse networks, with not too small a bandwidth, substantial gains in computational speed can be achieved.

(3) The relative efficiency of the new PCG, compared with the Frontal direct solver, improves exponentially with the size of the network and even more rapidly with the size of the largest block.

(4) The performance of the PCG, unlike direct solvers, is not as significantly affected by the bandwidth of the network matrix. However, for power network matrix equations, in contrast to the direct solver, it is observed that the number of iterations required for convergence decreases with the size of the largest block for networks of equal size and sparsity. This phenomenon suggests that the larger the bandwidth the better the eigenvalues are clustered by the pre-conditioning process.

(5) The complex PCG method should be seriously considered in power system analysis where repeated solutions of systems of complex linear equations are required.

CHAPTER
SIX

ARTIFICIALLY SYNTHESIZING NETWORK DATA FOR POWER SYSTEM ANALYSIS

6.1 Introduction

To evaluate the performance and robustness of new power system analysis algorithms, extensive numerical tests are required. Thus, it is essential to have realistic data for a large set of power networks of various types and sizes. Testing such algorithms on a few individual networks, as is often the case, may not clearly identify all its advantages or disadvantages.

It is evident that there are many difficulties associated with the collection of real network data, especially for very large scale systems. These difficulties are both technical and due to confidentiality reasons. Numerical testing is therefore often restricted to the relatively small IEEE test networks or to a limited set of special power networks whose data are not available to the general research community. In other fields, such as finite element

electromagnetic analysis, to alleviate the difficulty of collecting realistic data, synthetically generated data are commonly used for many investigations [114, 115].

This chapter presents an approach to synthetically generate realistic data [121] for power networks of arbitrary size and complexity.

We were motivated to develop this software by:

(i) The critical need for network data to investigate and test many power system analysis methods (previous chapters).

(ii) The difficulty of gathering sufficient network data representative of different topologies and sizes.

(iii) The fact that, to our knowledge, there in no widely available systematic method for producing arbitrary network data for power system analysis.

(iv) The ability to be able to control the various network parameters which may impact on the performance of the algorithm being tested (e.g. size, topology, line and bus data).

This technique was used in the numerical investigations described in the previous chapters. It was found to be very valuable in the present thesis but it should also find numerous applications in other power network analysis areas where testing of algorithms is necessary on large scale systems.

6.2 Data Specification in the Power Flow Problem

In this chapter, the emphasis is essentially on the generation of realistic data for the power flow problem. However, this data also serves to test

algorithms related to other types of power system analysis such as short circuit studies, transient stability and contingency analysis. Thus, some of the conclusions of this section also apply to other types of power system analysis problems.

The power flow problem is formulated by a set of nonlinear algebraic equations describing the behaviour of the complex power flows and voltages throughout the network under steady state conditions (see Chapter II). To formulate a load flow problem, three basic sets of data are required: the topological, the line parameter and the bus data.

Topological data is defined by two index vectors describing how the buses are tied to each other through transmission lines or transformers. This is a very important set of data having a major impact on the speed and storage requirements of a load flow algorithm. In essence, network topology defines the number of non-zero elements to be stored and numerically manipulated during the solution. This point is discussed more thoroughly later on in this chapter.

We define a local network (or block) as a sub-network composed of a sub-set of the total number of nodes. Normally these local networks are interconnected to each other through some tie lines resulting in bulk interconnected networks.

Normally, transmission lines and transformers are modelled by lumped π-models. In this type of model each transmission line or transformer located between two nodal points is characterized by its series impedance (composed of series resistance, R and series reactance, X) together with its shunt admittance elements, G (resistive) and B (capacitive).

Bus data describes generators and loads as power sources and sinks respectively connected to the nodal points of the power network. They are divided into three categories:

a) Generation Buses (Voltage Controlled Buses): At these buses, the real power injections P_is and the magnitudes of the voltages V_is are given as input data. The reactive power injections Q_is and the voltage angles δ_is are computed by the power flow program.

b) Load buses: P_is and Q_is are known as the input data, and the magnitudes and the angles of the voltages are to be computed.

c) Slack Bus: The slack bus is a reference bus for which the voltage amplitude and angle are given as input data. Typical values are one per unit and zero radians respectively. The load flow program computes the corresponding injected powers P and Q at this bus.

In summary, the formulation of the load flow problem needs input data consisting of the topological connectivity index vectors, the line and transformer admittance values, as well as the specified P, Q and V values for the various bus types.

6.3 The Synthetic Network Generator Algorithm

6.3.1 The general approach

The algorithm starts by assigning the total number of buses to the network as per the user input data. Then, realistic random voltage magnitudes and angles are ascribed to each of these nodes (section 6.3.4). Based on the selected network types and connectivity, these nodes are connected to each other following a constrained random process related to the specified topology of the network (section 6.3.2). For networks constructed from a set of local blocks, interconnections between the local networks are determined in a similar random manner (section 6.3.2). Then, the power injections in the complete network are computed to complete the input data for the load flow

problem. Finally, user constraints are iteratively imposed to force the injection data to lie within the prescribed ranges. Figure 6.1 shows a basic flow chart for the algorithm.

6.3.2 Network configuration

The networks created by the developed software are composed of a single block or multiple interconnected local blocks.

6.3.2.1 Single block networks

In this category, the power system consists of a local sparse network with no external interconnections. There is no limit to the size of this single block network and the number of buses can be any integer greater than two. These single block networks may have two different topologies:

a) Star Networks: In this type, first, all of the network buses are connected to each other through a ring of transmission lines to assure connectivity of all buses. Then, extra connections among different buses are created randomly (Figure 6.2). The number of the connections from any bus is based on a random integer, typically four connections per bus on average (this generates twice as many lines as buses on average). This average value is an input to the program and can be controlled by the user. The buses to which the randomly generated lines are connected are also chosen randomly. Double circuit lines can be excluded by the program if the user so desires. This type of star network has been shown to be more difficult to order efficiently for computations using direct solvers (see chapter III).

b) Grid Networks: For this type, the network has the form of a matrix, whose elements represent the nodal points of the network (Figure 6.3). The width of the network will be *nw* buses, where *nw* is the greatest integer less than $nb^{0.5}$, where *nb* is the total number of the buses in the block. The last column may consist of fewer nodes than *nw*.

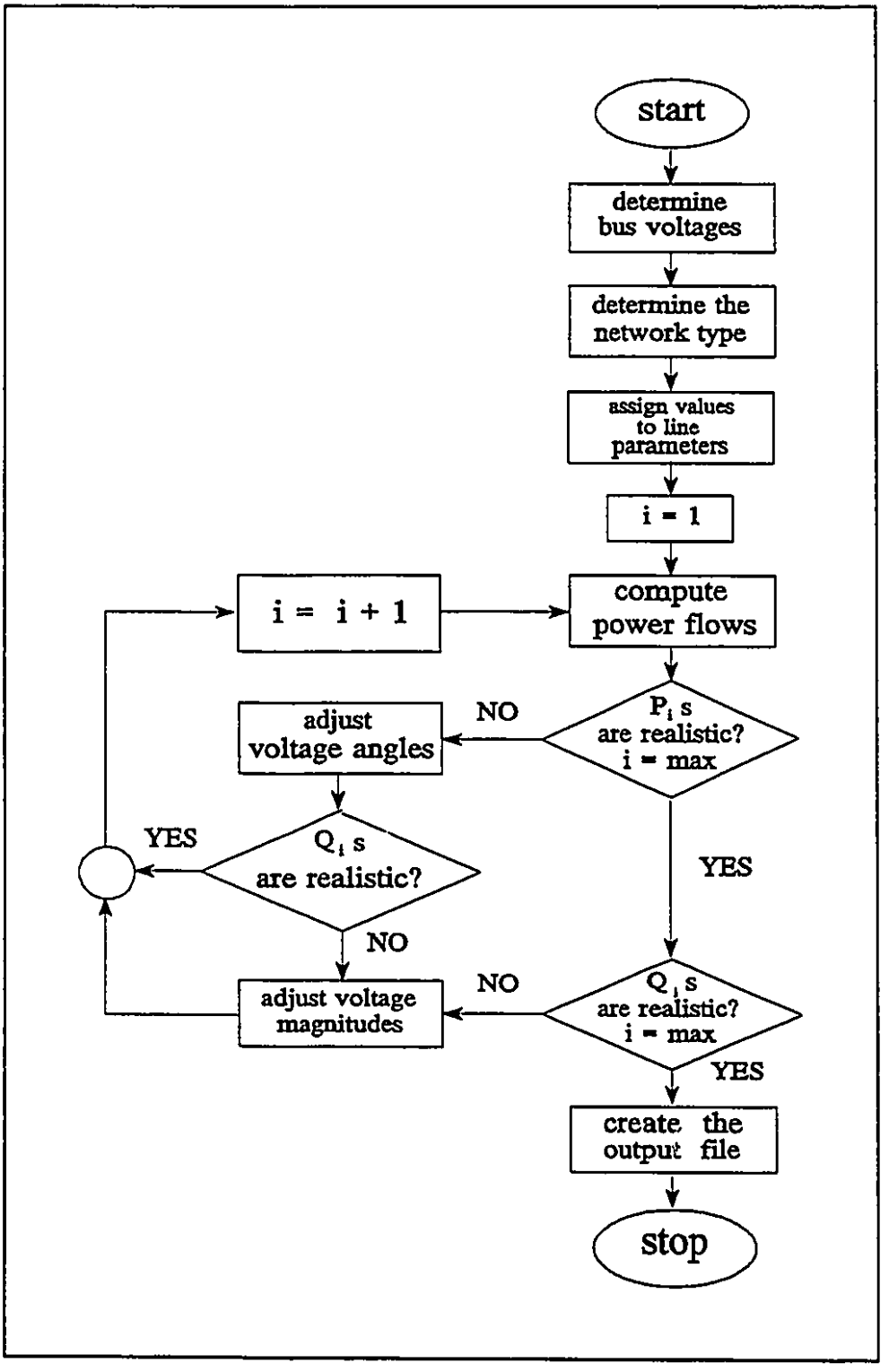

Figure 6.1: A basic flow chart for the network generator algorithm

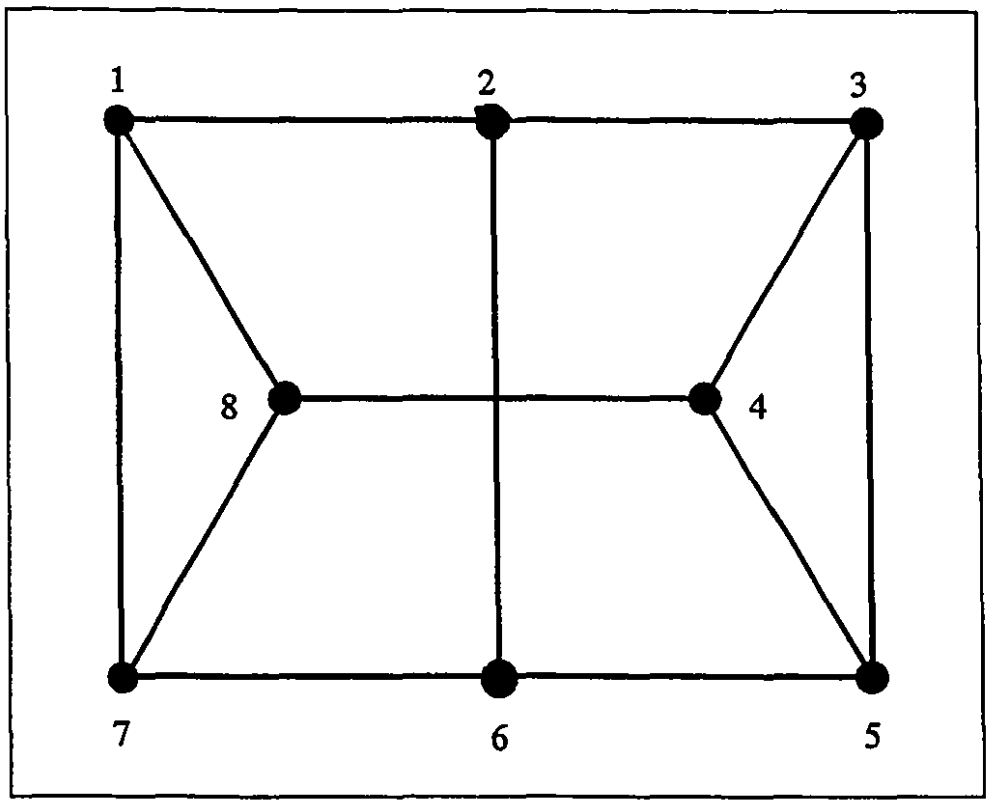

Figure 6.2: An example of an 8-bus star-type power network

In this category of networks, the peripheral points are first connected to each other, and then the adjacent nodes along each column are connected to each other to assure that no bus is isolated from the body of the network. Finally, additional random connections between n_{ij} and $n_{i'j'}$ are created, where i (i′) represents the row number and j (j′) denotes the column number of the grid. In our algorithm, generally $|i-i'| \leq 1$. However, $|j-j'|$ can be greater than one, if the generated network is expected to have non-planar lines. The total number of transmission lines connected to any bus is constrained to be within a realistic range. In the algorithm, the default ratio of the number of lines to the number of buses has been selected by inspection of the available real networks and IEEE example systems. However, the user has control over this ratio. Thus, networks with varying sparsity may be created.

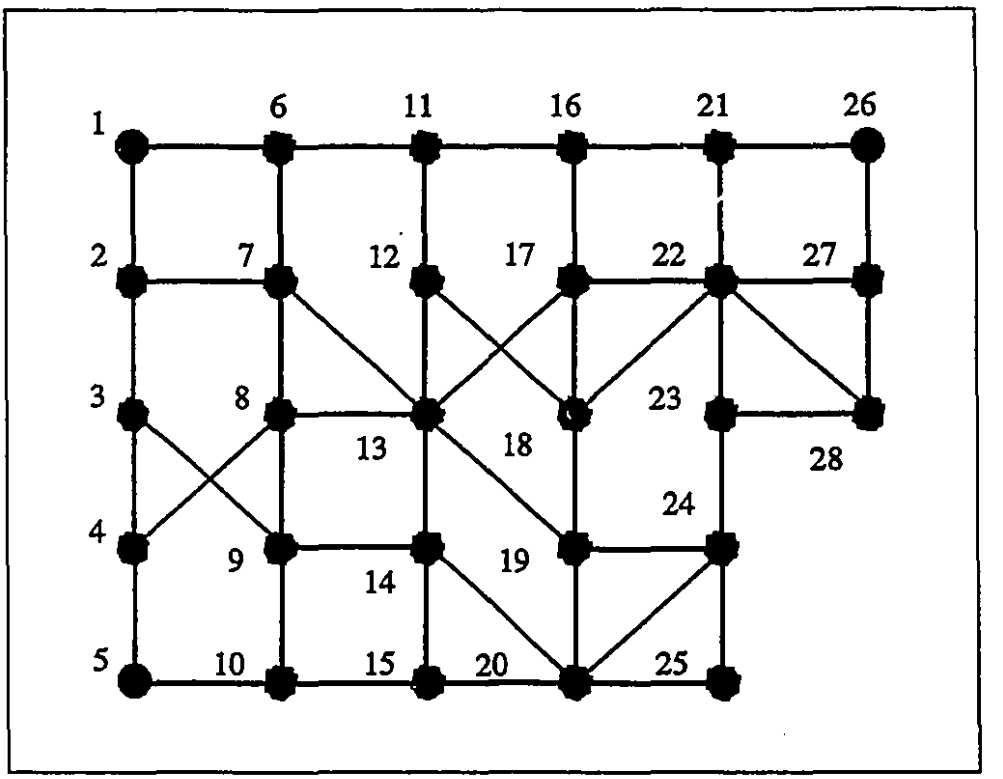

Figure 6.3: An example of a 28-bus grid-type power network

6.3.2.2 Multi-block networks

Networks in this category are composed of sparse blocks of varying sizes. The block sizes are random integers within a user specified range. The blocks themselves are interconnected into a sparse network. The inter-block connections are similar to the nodal point connections in each individual block, and therefore have the same topology. As an example, Figure 3.1 (chapter III) shows a star-type block interconnection for a multi-block network composed of eight blocks of varying sizes which is similar to the star-type network of Figure 6.2. Figure 3.2 (chapter III) shows the sparsity structure of the B matrix of a synthetically generated network of this type. The algorithm allows the user to number and connect blocks to each other either in a constrained random fashion, or according to a specific ordering.

6.3.3 The values assigned to line parameters

In the algorithm, line parameters are chosen to have random values within a certain range around their specified base values. These base values, which are in per unit, have been determined by inspection of real networks and IEEE example systems. They have been chosen to be not very far from the average values of these networks (typically within 20%).

6.3.4 Voltage magnitudes and angles

A random magnitude is assigned to the voltage at any node, which is within a user specified range provided as input. Typical values vary from 0.95 to 1.1 per unit.

The angles are chosen to lie within a user specified range, however the probable values within this angle range is not uniform. This is done to ensure that the algorithm produces a smaller number of buses with high angles compared to buses with low angles. Generally, buses with higher angles correspond to generation buses which are fewer in number than load buses which typically have lower angles.

6.3.5 Bus type determination

Power flows in all transmission lines (transformers) are computed based on the assigned values for the line (transformer) parameters and bus voltages and the load flow equations. Then, the injected real and reactive powers are calculated at all buses. The determination of generator and load bus types is based on these injected power quantities. The slack bus number and its voltage is user selected.

6.4 Control Actions Applied to the Algorithm

During the development of the algorithm, it was observed that some generated networks did not show good load flow convergence. It was found that this undesirable behaviour occurred when the real and imaginary power injections at adjacent buses varied very rapidly. This is an indication that the voltage profile (both in magnitude and angle) selected is not sufficiently smooth. Thus, some control operations were implemented in the algorithm to alleviate this problem.

6.4.1 Real power adjustment

The program checks the computed input power injected into any nodal point. If the absolute value is not within a specified range, the angle value of the voltage at this bus is shifted toward the average value of the angles for those buses which are directly connected to it. After each change in the network, the power flows and injections are re-computed. These adjustments may be repeated several times until the injections are within the specified range. The selected power injection range must be reasonable in the sense that adjacent buses do not have sharp differences in power injections relative to the capacity of the network to transmit this power.

6.4.2 Imaginary power adjustment

Imaginary power adjustment is similar to real power adjustment. However, in this process the magnitudes of the voltages rather than the angles are adjusted due to the high sensitivity of the imaginary power to voltage magnitudes. Here, however, the adjustments are made to try to keep the injected reactive power within 50% of the real power injection.

6.4.3 Output data file

The output data used for the load flow program can be written and saved directly as an ASCII file in any standard format, including the IEEE standard.

6.4.4 Applications

The data produced by the developed program are useful for analytical investigations in many power system studies. We have found this technique to be valuable in the numerical investigation of new solvers for the load flow problem [29], and in a numerical investigation of the eigenvalue properties of network matrices [120].

Since the user has control over base parameters such as, system size, number of transmission lines and their connectivity, and line coefficients, this program can be used for numerical comparison of the convergence rate, robustness and the validity of different algorithms as well as the relative impact of these parameters. This type of analysis is essential in planning and operations planning.

6.5 Concluding Remarks

A new algorithm has been developed that generates synthetic power networks with the following characteristics:

(1) The program can create networks with different types of dimensions, configurations and connectivities.

(2) The program allows the user to specify the base parameter values for lines and transformers.

(3) The program can constrain the randomly created data such as the power injections, voltage magnitudes and angles to lie within specified ranges.

(4) There is no limit on the size of the network that can be generated.

(5) The network configurations and the assigned values of the network elements can be controlled by the user to make the system as realistic as required.

(6) The network data are written directly into ASCII files in any standard format, including the IEEE standard.

Based on the above noted properties and on our own research, this program is an extremely useful tool for research and development of new or improved power system analysis algorithms.

CHAPTER
SEVEN

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

7.1 Conclusions

The majority of power system analysis studies essentially rely on the repeated solutions of large sparse systems of linear equations. The principal objective of this research has been to investigate the potential of the pre-conditioned conjugate gradient (PCG) algorithm to solve systems of linear equations arising in power system studies. Although PCG solvers have replaced the standard direct methods in various areas of engineering such as finite element analysis, such comparisons had never been attempted in power system problems up to now.

As part of this investigation, first it was proven (chapter III) that the B′ and B″ matrices arising in the DC and Fast Decoupled load flow algorithms are positive definite and, thus, comply with the convergence requirements of the PCG algorithm. Then, for the first time, the PCG

algorithm was applied to the solution part of the DC and Fast Decoupled load algorithms. The PCG approach was tested on the IEEE standard networks of 14, 30, 57 and 118 buses as well as on a large number of synthetically generated power networks up to 5000 buses and 10000 lines. The performance of the PCG for these series of experiments was evaluated and compared against two types of sparse matrix direct solvers: the bandwidth-based Frontal solver and the minimum degree ordering Sparspak-B5 solver (Tinney-2 algorithm).

More specifically, this thesis has numerically examined the effect on the performances of the PCG and direct methods of the following parameters: the size of the network, the size of the largest block in the network, the block arrangements in the network, the type of network configuration and its connectivity.

The results of this experimental evaluation reveal a significant speed advantage of the PCG solver over direct solvers in the analysis of very large sparse power networks. This speed advantage is even more pronounced in the analysis of networks which are not very narrow banded (i.e., networks which have at least one large sparse block) specially in comparison with the Frontal solver. This is due to the fact that the efficiency of the Frontal solver decreases for networks with large blocks. On the other hand, the performance of the PCG is not affected considerably by the matrix bandwidth. In fact, the PCG shows improved convergence behaviour for networks with larger bandwidths but with the same sparsity. This advantageous behaviour of the PCG can be attributed to a better clustering of the matrix eigenvalues for networks with larger bandwidth (although a clear fundamental reason for this general behaviour is not yet known).

It was also observed that, while the PCG performance is not significantly affected by the network type, direct solvers are very sensitive to the type of connectivity. For example, Sparspak-B5, which is one of the most

commonly used direct solvers and ordering schemes in power system analysis, is much slower for star-type as compared with grid-type networks. This behaviour is due to the fact that direct solvers are very sensitive to ordering algorithms.

The convergence of the PCG algorithm in the solution of power networks also proved to be faster than expected from previous experiences in the application of PCG methods. Convergence usually occurred in fewer iterations than the typical value of $n^{0.5}$ reported in the literature. This may be primarily due to the fact that power network matrices are essentially diagonally dominant and, when pre-conditioned, very good clustering of the eigenvalues is achieved.

Most of the above mentioned results relate to the Fast-Decoupled Load Flow, however, the possible use of the PCG in other load flow algorithms such as the Full Newton method were also investigated in this thesis. The results show that this type of load flow algorithm is difficult to adapt to the requirements of the PCG method and no advantage was observed in replacing direct solvers by the PCG.

In chapter IV, two new types of pre-conditioners were experimentally investigated. The performance of these two types of pre-conditioners was compared with that of the classical incomplete Cholesky (CIC) pre-conditioner. This comparison was based on the clustering effect of the eigenvalues of the resultant coefficient matrices as well as on the number of iterations required for the PCG to converge. This investigation showed that different pre-conditioning schemes based on approximations to the Cholesky factor seem to be the most promising approach to cluster the system eigenvalues efficiently. This study also indicates that superior pre-conditioning can be achieved if the intrinsic properties of power networks are exploited. This, then implies that more specialized and faster PCG algorithms could be developed which utilize the physical properties of power networks.

In chapter V, the PCG algorithm was modified such that it could be applied to systems of complex linear equations. It was demonstrated that the algorithm converges to the correct solution for systems of linear equations whose coefficient matrix is equal to the complex admittance matrix of a power network. The performance of the PCG for these types of problems was evaluated for power networks of different topologies and various sizes up to 6500 buses and 13000 lines. The evaluation was based on the comparison of the PCG with the standard bandwidth based Frontal direct solver. The encouraging results of this investigation suggest that the new PCG algorithm should also be seriously considered to replace direct solvers in transient stability and short-circuit studies involving the solution of complex systems of equations.

Finally, to be able to support our experimental conclusions by adequate test results, it was highly desirable to have access to a broad range of realistic data characterizing power networks of various types and sizes. Thus, a new technique was developed that generates realistic data from power networks of arbitrary size and complexity. While, these networks are randomly generated, control adjustments have been implemented in the software to make the generated networks as realistic as possible. The software allows the user to specify the system dimension, type of network, connectivity configurations and other network characteristics. This program has proven to be a very useful tool, not only for the application of this thesis, but it should also find numerous uses to thoroughly test any new power system planning and operation algorithms.

7.2 Recommendations for Future Research

Even though, this thesis has demonstrated the advantage of the application of the PCG method over direct solvers in power system studies, the potential of the PCG algorithm in all areas of power system operation and

planning is yet to be fully explored. Therefore, future work should be directed towards new applications of PCG in various power system studies. The main recommendations for future research are:

1) Further study should be devoted to explore the intrinsic properties of power network matrices. Thus, more specialized pre-conditionings may be possible for certain types of networks. This suggests that each utility may utilize a specific PCG algorithm for its own applications to enhance the efficiency of its power system algorithms.

2) Investigations should be also directed to modify the PCG algorithm such that it would converge to the solution for systems of linear equations whose coefficient matrix is not positive definite.

3) More detailed investigation should be directed toward the use of PCG solvers in the transient stability problem.

4) The power system state estimation problem should be considered as a good candidate for application of the PCG algorithm.

5) The potential application of the PCG algorithm to the optimal load flow problem, especially, for on-line computations should be investigated.

6) Parallel processing approaches appears to be promising in the PCG algorithm to enhance the efficiency of the overall computation.

APPENDIX
A

JACOBI, GAUSS-SEIDEL AND SUCCESSIVE OVERRELAXATION ITERATIVE METHODS

These methods were first considered by Withmeyer in 1936 [87]. All of these three methods are linear, first degree iterative techniques for the solution of systems of linear equations. In the following we describe these three iterative methods [88, 90, 95].

It was mentioned in chapter II that all these three methods may be expressed in the form,

$$B\,x^{m+1} + (A-B)\,x^{m} = b \tag{a.1}$$

or,

$$x^{m+1} = (I-B^{-1}A)\,x^{m} + B^{-1}b \tag{a.2}$$

Assuming $(I-B^{-1}A)=G$ and $B^{-1}b=k$, we can write,

$$x^{m+1} = G\,x^{m} + k \tag{a.3}$$

where G is called the iteration matrix for the method. In this class of iterative techniques, each choice of a non-singular matrix B (equation a.2) leads to a potential iterative method.

In the Jacobi iterative method [89] (sometimes called total step method), one chooses,

$$B = D \tag{a.4}$$

where,

$$D = diag \{ a_{11}, a_{22}, \ldots , a_{nn} \} \qquad (a.5)$$

Now, we decompose matrix A in the standard decomposition form,

$$A = D-E-F \qquad (a.6)$$

such that E and F are respectively lower and upper triangular n by n matrices. One thus obtains from equation a.2,

$$x^{m+1} = D^{-1} (E+F) x^m + k \qquad (a.7)$$

The matrix $D^{-1}(E+F)$ in equation a.7, is called the point Jacobi matrix associated with the matrix A. From this equation, the ith component of the x vector in its scalar form can be expressed as,

$$x_i^{m+1} = -\sum_{j=1}^{n} (\frac{a_{ij}}{aii}) x_j^m + \frac{K_i}{a_{ii}} \qquad (1 \le i \le n,\ j \ne i) \qquad (a.8)$$

It is clear from equation a.8 that in this iterative technique, all the components of the vector x^m must be saved during the computation of the new vector x^{m+1}. However, it seems advantageous to use the latest estimates x_j^{m+1} of the components x_j $(0<j\le i)$ to calculate x_{i+1}^{m+1} component. This idea will result in the computation of the solution vector components as below,

$$x_i^{m+1} = -\sum_{j=1}^{i-1} (\frac{a_{ij}}{aii}) x_j^{m+1} - \sum_{j=i+1}^{n} (\frac{a_{ij}}{a_{ii}}) x_j^m + \frac{K_i}{a_{ii}} \qquad (1 \le i \le n,\ j \ne i) \quad (a.9)$$

In this approach called point Gauss-Seidel or point single step iterative method [90], it is not necessary to save two approximations x_i^{m+1} and x_i^m

during the computational process. Equation a.9 can be expressed in matrix notation as,

$$x^{m+1} = (D-E)^{-1}F\,x^{m} + (D-E)^{-1}k \qquad \text{(a.10)}$$

where, (D-E) in this equation, is a non singular lower triangular matrix and $(D-E)^{-1}F$ is called the point Gauss-Seidel matrix associated with the matrix A.

Another iterative method for solving systems of linear equations closely related to the Gauss-Seidel method, is called the point successive overrelaxation (SOR) method [91]. In this approach, x_i^{m+1} is a weighted mean of x_i^m and x_i^{m+1} in the point Gauss-Seidel method. Thus, the components x_i^{m+1} of this methods are defined as,

$$x_i^{m+1} = x_{Gi}^{m} + \omega\,(x_{Gi}^{m+1} - x_{Gi}^{m}) = (1-\omega)\,x_{Gi}^{m} + \omega\,x_{Gi}^{m+1} \qquad \text{(a.11)}$$

where, x_{Gi}^m and x_{Gi}^{m+1} are the two successive estimations for the solution vector in the Gauss-Seidel iterative method and the quantity ω is called the relaxation factor. It is clear that the two weighting factors $(1-\omega)$ and ω depend only on ω and for $0 \le \omega \le 1$ both weights are non-negative. In this approach if $\omega > 1$, we will refer to it as overrelaxation, while $\omega < 1$ corresponds to underrelaxation.

One can combine two equations a.9 and a.11 to conclude,

$$x_i^{m+1} = x_i^{m} + \omega\,\{-\sum_{j=1}^{i-1}(\frac{a_{ij}}{a_{ii}})x_j^{m+1} - \sum_{j=i+1}^{n}(\frac{a_{ij}}{a_{ii}})x_j^{m} + \frac{k_i}{a_{ii}} - x_i^{m}\} \qquad \text{(a.12)}$$

This method like Gauss-Seidel method needs the storage of only one solution vector during the computation process. This algorithm in matrix form can be expressed as,

$$(D-\omega E)\, x^{m+1} = \{(1-\omega)I + \omega\, U\}\, x^{m} + \omega k \qquad \text{(a.13)}$$

Assuming $L \equiv D^{-1}E$ and $U \equiv D^{-1}F$, equation a.13 will be reduced to,

$$x^{m+1} = (I-\omega L)^{-1}\{(1-\omega)I + \omega U\}x^{m} + \omega(I-\omega L)^{-1}D^{-1}k \qquad \text{(a.14)}$$

The convergence of this method is dependent on the relaxation factor ω [92-94]. For $\omega = 1$, this method reduces to point Gauss-Seidel iterative method.

These iterative techniques have the advantage of less memory requirements when compared with direct solution methods. However, due to their frequent slow convergence, none of them are recommended for power system algorithms where reliable fast convergence is vital.

APPENDIX
B

CONJUGATE GRADIENT ALGORITHM

The conjugate gradient scheme (CG) [99-104] is an important semi-iterative technique when the coefficient matrix is positive definite. This method was first presented by Hestenes and Stiefel in 1952 [99].

In the absence of rounding errors, in contrast to other iterative techniques, the CG method terminates with the exact solution in at most n steps [99-101]. Because of rounding errors, however, additional iteration steps may be required [88].

The basis of the method comes from the fact that the function,

$$F(x) = \frac{1}{2} x'A\, x - x'b \qquad \text{(b.1)}$$

is minimized by $x=A^{-1}b$. Thus, finding the minimum point of F will give us the solution to the linear system of equations $Ax=b$ if A is positive definite. Conjugate gradient algorithms use a sequence of linear search directions (p) starting from an initial guess x_0 and, at each iteration, a better estimate of the solution is obtained, such that,

$$x_{i+1} = x_i + \alpha_i\, p_i \qquad \text{(b.2)}$$

Thus, the final solution can be expressed as a linear function of the search directions,

$$x = \alpha_1\, p_1 + \alpha_2\, p_2 + \ldots + \alpha_m\, p_m \qquad \text{(b.3)}$$

where, m is guaranteed to be no larger than the matrix size n and mainly depends on the starting direction p_1 and on the distribution of the eigenvalues of the coefficient matrix [102].

The main feature of this scheme is that any two search directions are conjugate. Two directions p_i and p_j are said to be conjugate with respect to matrix A if,

$$p_i^t A p_j = 0 \tag{b.4}$$

Thus, assuming that all p vectors are mutually conjugate and premultiplying equation b.3 by $p_i^t A$, we will have,

$$p_i^t A [\alpha_1 p_1 + \alpha_2 p_2 + \ldots + \alpha_m p_m] = p_i^t b \tag{b.5}$$

from which, we can conclude,

$$\alpha_i = \frac{p_i^t b}{p_i^t A p_i} \tag{b.6}$$

Hence, to find the solution vector x in equation b.3, one has to find the conjugate directions p_i. Having these search vectors along with the terms α_i then yields the solution.

Generally, in this algorithm,

$$p_1 = b - A x_1 \tag{b.7}$$

is a good value for the starting search direction. Now, all one has to do is to proceed updating the solution vector using equation b.2. Now, assume that we are at step i of the iterative process. Then, from equation b.3, the error vector at this step can be calculated as,

$$e_i = x - x^i = \alpha_i p_i + \alpha_{i+1} p_{i+1} + \ldots \tag{b.8}$$

The residual vector, r_i, can be written as,

$$Ae_i = A x - A x^i = b - A x^i = r_i \tag{b.9}$$

Then, from equations b.8 and b.9 one can write,

$$r_i = A(\alpha_i p_i + \alpha_{i+1} p_{i+1} + \ldots) \tag{b.10}$$

Premultiplying equation b.10 with p_i^t and applying the conjugacy property of equation b.4, we can conclude,

$$\alpha_i = \frac{p_i^t r_i}{p_i^t A p_i} \tag{b.11}$$

After calculating α_i from equation b.11, we may update the solution vector to x^{i+1} (equation b.2).

In this algorithm the search direction in subsequent iterative steps are calculated as,

$$p_{i+1} = r_{i+1} + \beta_i p_i \tag{b.12}$$

The first term of this expression is the residual at the last step which corresponds to the gradient of the function to be minimized. The second term is added to adjust p_{i+1} so that it can be conjugate to p_i. Thus to ensure conjugacy of two successive search directions, using equations b.4 and b.12 one will obtain,

$$\beta_i = -\frac{p_i^t A r_{i+1}}{p_i^t A p_i} \tag{b.13}$$

Choosing the search directions based on the above mentioned

procedure, it may be shown that p_{i+1} is mutually conjugate to any of the preceding search directions.

The above mentioned equations together will define the general conjugate gradient algorithm as follows :

1) Initialization:

a) Initialize ε.

b) Guess x.

c) Form $r = b - Ax$.

While $\|r\| > \theta$ *do :*

2) Form $\alpha = p^t p \,/\, p^t Ap$.

3) Update $x = x + \alpha p$.

4) Update $r = b - A x$.

5) Form $\beta = -r^t Ar \,/\, p^t Ap$.

6) Update $p = r - \beta p$.

REFERENCES

[1] Olle I. Elgerd, **ELECTRIC ENERGY SYSTEMS THEORY, AN INTRODUCTION**, Second Edition, McGraw Hill, 1983.

[2] Corwin, J.L., Miles, W. Tinney, "IMPACT ASSESSMENT OF THE 1977 NEW YORK CITY BLACK OUT", Systems Engineering for Power, Division of Electric Energy Systems, U.S., Department of Energy, July 1978.

[3] A.F. Glimn, G.W. Stagg, "AUTOMATIC CALCULATION OF LOAD FLOWS, "AIEE Trans. on PAS, Vol. 76, October 1957, PP. 817-825.

[4] A. Brameller, J.K. Denmead, "SOME IMPROVED METHODS FOR DIGITAL NETWORK ANALYSIS", Proc. IEE (London), Vol. 109, 1962, Pt. A, PP. 109-116.

[5] M.A. Laughton, M.W. Humphrey Davies, "NUMERICAL TECHNIQUES IN SOLUTION OF POWER SYSTEM LOAD FLOW PROBLEMS," Proc. IEE (London), Vol. 111, 1964, PP. 1575-1588.

[6] L.L. Freris, A.M. Sasson, "INVESTIGATION ON LOAD FLOW PROBLEM", Proc. IEE (London), Vol. 114, 1967, P. 1960.

[7] Y. Wallach, R.Evan, "APPLICATION OF NEWTON'S METHOD TO LOAD FLOW CALCULATIONS", Proc. IEE, Vol. 114, 1967, PP. 372-374.

[8] W.F. Tinney, C.E. Hart, "POWER FLOW SOLUTION BY NEWTON'S METHOD", IEEE Trans. on PAS, Vol. 86, November 1967, PP. 1449-1456.

[9] A.M. Sasson, F.J. Jaimes, "DIGITAL METHODS APPLIED TO POWER FLOW STUDIES", IEEE Trans. on PAS, Vol. 86, July 1967, PP. 860-867.

[10] A.M. Sasson, "NONLINEAR PROGRAMMING SOLUTIONS FOR LOAD FLOW, MINIMUM LOSS, AND ECONOMIC DISPATCHING PROBLEMS", IEEE Trans. on PAS, Vol. 88, No.4, April 1969, PP. 399-409.

[11] B. Sttot, "DECOUPLED NEWTON LOAD FLOWS", IEEE Trans. on PAS, Vol.91, September/October 1972, PP. 1955-1957.

[12] B. Sttot, O. Alsac, "FAST DECOUPLED LOAD FLOW", IEEE Trans. on PAS, Vol.93, May/June 1974, PP. 859-869.

[13] H.W. Dommel, N. Sato, "FAST TRANSIENT STABILITY SOLUTIONS", IEEE Trans. on PAS, Vol. 91, No. 4, July/August 1972, PP. 1643-1650.

[14] Zing Zhu, Anjan Bose, "A DYNAMIC PARTITIONING SCHEME FOR PARALLEL TRANSIENT STABILITY ANALYSIS", IEEE Trans. on PAS, Vol. 7, May 1992, PP. 940-946.

[15] G.T. Vuong, A. Valette, "A COMPLEX Y-MATRIX ALGORITHM FOR TRANSIENT STABILITY STUDY", IEEE Trans. on PAS, Vol. 104, No. 12, December 1985, PP. 3388-3394.

[16] G.C. Ejebe, B.F. Wollenberg, "AUTOMATIC CONTINGENCY SELECTION", IEEE Trans. on PAS, Vol.98, No. 1, January/February 1979, PP. 97-108.

[17] T.A. Mikolinnas, B.F. Wollenberg, "AN ADVANCED CONTINGENCY SELECTION ALGORITHM", IEEE Trans. on PAS, No. 2, February 1981, PP. 608-617.

[18] G.D. Irisarri, A.M. Sasson, "AN AUTOMATIC CONTINGENCY SELECTION METHOD FOR ON-LINE SECURITY ANALYSIS", IEEE Trans. on PAS, Vol. 100, No. 4, April 1981, PP. 1834-1844.

[19] F.D. Galiana, "BOUND ESTIMATES OF THE SEVERITY OF LINE OUTAGES IN POWER SYSTEM ANALYSIS AND RANKING", IEEE Trans. on PAS, Vol. 103, September 1984, PP. 2612-2622.

[20] V. Brandwajn, M.G. Lauby, "COMPLETE BOUNDING METHOD FOR AC CONTINGENCY SCREENING", IEEE Trans. on PAS, Vol. 4, No. 2, May 1989, PP. 724-729.

[21] Y. Wallach, E. Handschin, C. Bongers, "AN EFFICIENT PARALLEL PROCESSING METHOD FOR POWER SYSTEM STATE ESTIMATION", IEEE Trans. on PAS, Vol. 100, November 1981, PP. 4402-4406.

[22] Y. Wallach, E. Handschin, "STATE ESTIMATION BY ORTHOGONALIZATION AND SPLITTING", Electrical Power & Energy Systems, Vol. 4, No. 3, July 1982, PP. 177-184.

[23] J.D. Glover, M. Sarma, **POWER SYSTEM ANALYSIS AND DESIGN**, PWS Publishers, 1987.

[24] Robert L. Sullivan, **POWER SYSTEM PLANNING**, McGraw-Hill, New York, 1977.

[25] J. Carpentier, "ORDERED ELIMINATION", Proceedings of Power Systems Computation Conferences, London, 1963.

[26] Sherman M. Chan, Vladimir Brandwajn, "PARTIAL MATRIX REFACTORIZATION", IEEE Trans. on power systems, Vol. PWRS-1, No. 1, February 1986, PP. 193-200.

[27] Torsten Cegrell, **POWER SYSTEM CONTROL TECHNOLOGY**, Printice-Hall international, (UK) Ltd., London, 1986.

[28] O. Axelsson, "A SURVEY OF VECTORIZABLE PRE-CONDITIONING METHODS FOR LARGE SCALE FINITE ELEMENT MATRIX PROBLEMS", Technical Report CNA-190, Center for Numerical Analysis, The University of Texas at Austin, 1984.

[29] F.D. Galiana, H. Javidi, S. McFee, "ON THE APPLICATION OF A PRE-CONDITIONED CONJUGATE GRADIENT ALGORITHM TO POWER NETWORK ANALYSIS", Proceedings of PICA Conference, May 1993, Phoenix, PP. 404-410.

[30] Y. Wallach, **CALCULATIONS & PROGRAMS FOR POWER SYSTEM NETWORKS**, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1986.

[31] R. Bacher, W.F. Tinney, "FASTER LOCAL POWER FLOW SOLUTIONS : THE ZERO MISMATCH APPROACH", IEEE Trans. on PAS, Vol. 4, No. 4, October 1989, PP. 1345-1354.

[32] J. Carpentier, "APPLICATION OF NEWTON'S METHOD TO THE LOAD FLOW PROBLEM", Proc. Power Syst. Computations Conf., Queen Mary College, September 1963.

[33] G.W. Stagg, A.H. El-Abiad, **COMPUTER METHODS IN POWER SYSTEM ANALYSIS**, McGraw-Hill Book Company, 8th printing, 1981.

[34] A.J. Wood, B.F. Wollenberg, **POWER GENERATION, OPERATION AND CONTROL**, John Wiley, 1984.

[35] I.S. Duff, J.K. Reid, "A COMPARISON OF SPARSITY ORDERING FOR OBTAINING A PIVOTAL SEQUENCE IN GAUSSIAN ELIMINATION", SIAM J., Numer. Anal. 13, 1976, PP. 686-695.

[36] I.S. Duff, "A SURVEY OF SPARSE MATRIX RESEARCH", Proc. IEEE 65, 1977, PP. 500-535.

[37] Alan George, Joseph W.H Liu, "A MINIMAL STORAGE IMPLEMENTATION OF THE MINIMUM DEGREE ALGORITHM", SIAM J., Numer. Anal. 17, 1980, PP. 282-299.

[38] N.Sato, W.F. Tinney, "TECHNIQUES FOR EXPLOITING THE SPARSITY OF THE NETWORK ADMITTANCE MATRIX", IEEE Trans., Vol. 82, Pt. III, December 1963, PP. 944-949.

[39] H. Edelmann, "ORDERED TRIANGULAR FACTORIZATION OF MATRICES : A MULTIPURPOSE APPROACH FOR TREATING PROBLEMS IN LARGE ELECTRIC NETWORKS", Proc. Power Syst. Computation Conf., rept. 4.12, Stockholm, 1966.

[40] E.C. Ogbuobiri, W.F. Tinney, J.W. Walker, "SPARSITY-DIRECTED DECOMPOSITION FOR GAUSSIAN ELIMINATION ON

MATRICES", IEEE Trans. on Power Apparatus and systems, Vol. PAS-89, No. 1, January 1970, PP. 141-150.

[41] E.C. Ogbuobiri, "DYNAMIC STORAGE AND RETRIEVAL IN SPARSITY PROGRAMMING", IEEE Trans. On Power Apparatus and Systems, Vol. PAS-89, No. 1, January 1970, PP. 150-155.

[42] A. Jennings, "A COMPACT STORAGE SCHEME FOR THE SOLUTION OF SYMMETRIC LINEAR SIMULTANEOUS EQUATIONS", Comput. J.. No. 9, 1966, PP. 281-285.

[43] U.W. Pooch, A. Neider, "A SURVEY OF INDEXING TECHNIQUES FOR SPARSE MATRICES", ACM Computing Surveys, No. 5, 1973, PP. 109-133.

[44] C. Berge, "**THE THEORY OF GRAPHS AND ITS APPLICATIONS**", John Wiley & Sons Inc., New York, 1962.

[45] S.V. Parter, "THE USE OF LINEAR GRAPHS IN GAUSS ELIMINATION", SIAM Rev. No. 3, 1961, PP. 119-130.

[46] Alan George, Joseph W. Liu, "**COMPUTER SOLUTIONS OF LARGE SPARSE POSITIVE DEFINITE SYSTEMS**", Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1981.

[47] R. Baumann, "SPARSENESS POWER SYSTEM EQUATIONS", Proc. of the Oxford conference of Institute of Mathematics and Its Applications on Large Sparse Sets of Linear Equations, Oxford, 5-8 April 1970, PP. 105-125.

[48] M.E. Churchill, "A SPARSE MATRIX PROCEDURE FOR POWER SYSTEMS ANALYSIS PROGRAMS", Proc. of the Oxford conference

of Institute of Mathematics and Its Applications on Large Sparse Sets of Linear Equations, Oxford, 5-8 April 1970, PP. 127-138.

[49] E.C. Ogbuobiri, "SPARSITY TECHNIQUES IN POWER SYSTEM GRID-EXPANSION PLANNING", Proc. of the Oxford conference of Institute of Mathematics and Its Applications on Large Sparse Sets of Linear Equations, Oxford, 5-8 April 1970, PP. 219-230.

[50] W.F. Tinney, V. Brandwajn, S.M. Chan, "SPARSE VECTOR METHODS", IEEE Trans. on Power Apparatus and Systems, Vol. PAS-104, No. 2, February 1985, PP. 295-301.

[51] P. Ristanovic', M. Bjelogrlic', B.S. Babic', "IMPROVEMENTS IN SPARSE MATRIX/VECTOR TECHNIQUE APPLICATIONS FOR ON-LINE LOAD FLOW CALCULATION", IEEE Trans. on Power Systems, Vol. 4, No. 1, February 1989, PP. 190-196.

[52] I.S. Duff, A.M. Erisman, J.K. Reid, "**DIRECT METHODS FOR SPARSE MATRICES**", Oxford Science Publications, 1990.

[53] E. Cuthill, J. McKee, "REDUCING THE BANDWIDTH OF SPARSE SYMMETRIC MATRICES", Proceedings 24th National Conference of the Association for Computing Machinery, Brandon Press, New Jersy, 1969, PP. 157-172.

[54] N.E. Gibbs, W.G. Jr. Poole, P.K. Stockmeyer, "A COMPARISON OF SEVERAL BANDWIDTH AND PROFILE REDUCTION ALGORITHMS", ACM Trans. on Math. Software 2, 1976, PP. 322-330.

[55] N.E. Gibbs, W.G. Jr. Poole, P.K. Stockmeyer, "AN ALGORITHM FOR REDUCING THE BANDWIDTH AND PROFILE OF A SPARSE MATRIX", SIAM J. Numer. Anal., No. 13, 1976, PP. 236-250.

[56] R.J. Collins, "BANDWIDTH REDUCTION BY AUTOMATIC RENUMBERING", Int. J. Numer. Meth. Engng., Vol. 6, 1973, PP. 345-356.

[57] B.F. Wollenberg, "MINOR DIAGONAL ORDERING SCHEME", Private Communication, Research and Development Center, Leeds and Northrup Co., North Wales, Pa., September, 1968.

[58] E. Cuthill, "SEVERAL STRATEGIES FOR REDUCING THE BANDWIDTH OF MATRICES", in **Sparse Matrices and their Applications**, edited by D.J. Rose and R.A. Willoughby, Plenum Press, New York, 1972, PP. 157-166.

[59] C.A. Felippa, "SOLUTION OF LINEAR EQUATIONS WITH SKYLINE-STORED SYMMETRIC MATRIX", Computers and Structures 5, 1975, PP. 13-29.

[60] B. M. Iron, "A FRONTAL SOLUTION PROGRAM FOR FINITE ELEMENT ANALYSIS", Int. J. Numer. Meth. Engng., Vol. 2, 1970, PP. 5-32.

[61] R.J. Melosh, R.M. Bamford, "EFFICIENT SOLUTION OF LOAD DEFLECTION EQUATIONS", J. Struct. Div. ASCE, Proc. Paper No. 9861, 1973, PP. 1507-1526.

[62] Alan George, "COMPUTER IMPLEMENTATION OF THE FINITE-ELEMENT METHOD", Tech. Rept. STAN-CS-71-208, Stanford University, 1971.

[63] Joseph W. Liu, Andrew H. Sherman, "COMPARATIVE ANALYSIS OF THE CUTHILL-MCKEE AND REVERSE CUTHILL-MCKEE ORDERING ALGORITHMS FOR SPARSE MATRICES", SIAM J. Numer. Anal., No. 13, 1975, PP. 198-213.

[64] W.F. Tinney, J.W. Walker, "DIRECT SOLUTIONS OF SPARSE NETWORK EQUATIONS BY OPTIMALLY ORDERED TRIANGULAR FACTORIZATION", Proc. IEEE, Vol. 55, PP. 1801-1809, November 1967.

[65] W.F. Tinney, "COMMENTS ON USING SPARSITY TECHNIQUES FOR POWER SYSTEM PROBLEMS", in **Sparse Matrix Proceedings,** IBM Research Rept., RAI 3-12-69, 1969,

[66] Alan George, David R. McIntyre, "ON THE APPLICATION OF THE MINIMUM DEGREE ALGORITHM TO FINITE ELEMENT SYSTEMS", SIAM J. Numer. Anal., No. 15, 1978, PP. 90-112.

[67] H.M. Markowitz, "THE ELIMINATION FORM OF THE INVERSE AND ITS APPLICATION TO LINEAR PROGRAMMING", Management Science,No. 3, 1957, PP. 255-269.

[68] D.J. Rose, "A GRAPH-THEORETIC STUDY OF THE NUMERICAL SOLUTION OF SPARSE POSITIVE-DEFINITE SYSTEMS OF LINEAR EQUATIONS", in **Graph Theory and Computing**, edited by R.C. Read, Academic Press, New York, 1972, PP. 183-217.

[69] I.P. King, "AN AUTOMATIC REORDERING SCHEME FOR SIMULTANEOUS EQUATIONS DERIVED FROM NETWORK PROBLEMS", Int. J. Numer. Meth. Engng., Vol. 2, 1970, PP. 523-533.

[70] J.G. Lewis, W.G. Poole, "ORDERING ALGORITHMS APPLIED TO SPARSE MATRICES IN ELECTRICAL POWER PROBLEMS", in **Electric Power Problems : The Mathematical Challenge**, edited by A.M. Erisman, K.W. Nevesand M.H. Dwarakanath, SIAM, Philadelphia, 1980, PP. 115-124.

[71] A. George, "ON BLOCK ELIMINATION FOR SPARSE LINEAR SYSTEMS", SIAM J. Numer. Anal., No. 11, 1974, PP, 585-603.

[72] Alan George, Joseph, W. Liu, "ALGORITHMS FOR MATRIX PARTITIONING AND THE NUMERICAL SOLUTION OF FINITE ELEMENT SYSTEMS", SIAM J. Numer. Anal., No. 15, 1978, PP, 297-327.

[73] D.J. Rose, J.R. Banch, "THE ROLE OF PARTITIONING IN THE NUMERICAL SOLUTION OF SPARSE SYSTEMS", in **Sparse Matrices and their Applications**, edited by D.J. Rose and R.A. Willoughby, Plenum Press, New York, 1972. PP. 177-187.

[74] N. Müller, V.H. Quintana, "A SPARSE EIGENVALUE-BASED APPROACH FOR PARTITIONING POWER NETWORKS", IEEE Trans. on Power Systems, Vol. 7, No.
2, May 1992, PP. 520-527

[75] W.F. Tinney, "COMPENSATION METHODS FOR NETWORK SOLUTIONS BY OPTIMALLY ORDERED TRIANGULAR FACTORIZATION", IEEE Trans. Power App. Syst., Vol. PAS-91, January/ February 1972, PP. 123-127.

[76] O. Alsac, B. Stott, W.F. Tinney, "SPARSITY-ORIENTED COMPENSATION METHODS FOR MODIFIED NETWORK SOLUTIONS", IEEE Trans. on PAS, Vol PAS-102, No. 5, May 1983, PP. 1050-1060.

[77] V.H. Quintana, N. Müller, "PARTITIONING OF POWER NETWORKS AND APPLICATION TO SECURITY CONTROL", IEE Proceedings, Part C: Generation, Transmission and Distribution, Vol. 138, No. 6. November 1991, PP. 535-545.

[78] Iwamoto, Shinichi, Kusano, Masanori, Quintana, "HIERARCHICAL STATE ESTIMATION USING A FAST RECTANGULAR-COORDINATE METHOD", IEEE Trans. on PAS, Vol 4, No. 3, August 1989, PP. 870-880.

[79] Rod Cook, "A REFORMULATION OF PRECONDITIONED CONJUGATE GRADIENTS SUITABLE FOR A LOCAL MEMORY MULTI-PROCESSOR", in **Iterative Methods in Linear Algebra**, edited by R. Beauwen and P. de Groen, Elsevier Science Publishers B.V.(North-Holand), 1992, PP. 313-322.

[80] Malathi Ramdas, David R. Kincaid, "PARALLELIZING ITPACKV 2D FOR THE CRAY Y-MP", in **Iterative Methods in Linear Algebra**, edited by R. Beauwen and P. de Groen, Elsevier Science Publishers B.V.(North-Holand), 1992, PP. 323-337.

[81] Ilan Bar-on, " NEW DIVIDE AND CONQUER PARALLEL ALGORITHMS FOR THE CHOLESKY DECOMPOSITION AND GRAM-SCHMIT PROCESS", in **Iterative Methods in Linear Algebra**, edited by R. Beauwen and P. de Groen, Elsevier Science Publishers B.V.(North-Holand), 1992, PP. 251-262.

[82] R. Podmore, M. Liveright, S. Virmani, N.M. Peterson and J. Britton, "APPLICATION OF AN ARRAY PROCESSOR FOR POWER SYSTEM NETWORK COMPUTATIONS", Proc. 11th PICA Conference, PP. 325-334, Cleveland, May 1979.

[83] B. Dembart, A.C. Mong, K.W. Neves, D.S. Dodson and Y.M. Chang, "POWER FLOW CALCULATIONS UTILIZING ARRAY PROCESSORS", EPRI Report EL-4642, June 1986.

[84] G.P. Granelli, M. Montagna, G.L. Pasini, P. Marannino, "VECTOR

COMPUTER IMPLEMENTATION OF POWER FLOW OUTAGE STUDIES", IEEE Trans. on Power Systems, Vol. 7, No. 2, May 1992, PP. 798-804.

[85] J.P. Hulskamp, S.M. Chan, J.F. Fazio, "POWER FLOW OUTAGE STUDIES USING AN ARRAY PROCESSOR", IEEE Trans. on Power Apparatus & Systems, Vol. PAS-101, No. 1, January 1982, PP. 254-261.

[86] A. Viegas de Vasconcelos, "EXPLORING APPLICATIONS OF PARALLEL PROCESSING TO POWER SYSTEM PROBLEMS-A STATUS REPORT", IASTED Int. Conf. "Power High Tech. 89", july 1989.

[87] H. Wittmeyer, "ÜBER DIE LÖSUNG VON LINEAREN GLEICHUNGS SYSTEMEN DURCH ITERATION", Z. Angew. Math. Mech., No. 16, 1936, PP. 301-310.

[88] J. Stoer, R. Bulirsch, **INTRODUCTION TO NUMERICAL ANALYSIS**, Springer-Verlag, New York, 1983.

[89] C.G.J. Jacobi, "ÜBER EINE AUFLÖSUNGSART DER BEI DER METHODE DER KLEINSTEN QUADRATE VORKOMMENDEN LINEÄREN GLEICHUNGEN", Astr. Nachr. 22, No. 523, 57, 1845, PP. 297-306.

[90] R.S. Varga, **MATRIX ITERATIVE ANALYSIS**, Pribtice-Hall, Inc., Englewoodcliffs, New Jersy, 1962.

[91] O. Axelsson, "A GENERALIZED SSOR METHOD", BIT 13, 1972, PP. 443-467.

[92] G. Kjellberg, "ON THE CONVERGENCE OF SUCCESSIVE OVER-RELAXATION APPLIED TO A CLASS OF LINEAR SYSTEMS OF EQUATIONS WITH COMPLEX EIGENVALUES", Ericsson Tech. Stockholm 2, 1958, PP. 245-258.

[93] H.E. Kulsrud, "A PRACTICAL TECHNIQUE FOR THE DETERMINATION OF THE OPTIMUM RELAXATION FACTOR OF THE SUCCESSIVE OVERRELAXATION METHOD", Comm. ACM 4, 1961, PP. 184-187.

[94] J.K. Reid, "A METHOD FOR FINDING THE OPTIMUM SUCCESSIVE OVERRELAXATION FACTOR", Comput. J., No. 9, 1966, PP. 200-204.

[95] D.M. Young, **ITERATIVE SOLUTION OF LARGE LINEAR SYSTEMS**, Academic Press, New York, 1971.

[96] D.M. Young, "A HISTORICAL CVERVIEW OF ITERATIVE METHODS", Proc. of the Minnesota Supercomputer Institute Workshop on **PRACTICAL ITERATIVE METHODS FOR LARGE SCALE COMPUTATIONS**, edited by D.L. Boley, D.G. Truhlar, Y. Saad, R.E. Wyatt and L.E. Collins, Minneapolis, USA, 23-25 October, 1988, PP. 1-17.

[97] L.A. Hageman, D.M. Young, **APPLIED ITERATIVE METHODS**, Academic Press, Inc., London, 1981.

[98] H.A. Van der Vorst, "THE CONVERGENCE BEHAVIOUR OF SOME ITERATIVE SOLUTION METHODS", Proceed. of the fifth Int. Symposium on Numer. Methods in Engng., Vol. 1, edited by R. Gruber, J. Periaux and R.P. Shaw, Springer Verlag, Berlin, 1989, PP. 61-72.

[99] M.R. Hestenes, E. Stiefel, "METHODS OF CONJUGATE GRADIENTS FOR SOLVING LINEAR SYSTEMS", Nat. Bur. Standards J. Res. 49, 1952, PP. 409-436.

[100] J.A. Meijerink, H.A. Van der Vorst, "AN ITERATIVE SOLUTION METHOD FOR LINEAR SYSTEMS OF WHICH THE COEFFICIENT MATRIX IS A SYMMETRIC M-MATRIX", Math. Comput., No. 31, 1977, PP. 148-162.

[101] D.S. Kershaw, "THE INCOMPLETE CHOLESKEY-CONJUGATE GRADIENT METHOD FOR THE ITERATIVE SOLUTION OF SYSTEMS OF LINEAR EQUATIONS", J. Com. Phys., No. 26, 1978, PP.43-65.

[102] A. Jennings, "INFLUENCE OF THE EIGENVALUE SPECTRUM ON THE CONVERGENCE RATE OF THE CONJUGATE GRADIENT METHOD", J. Inst. Maths. Applics., Vol. 20, 1977, PP. 61-72.

[103] J.W. Daniel, "THE CONJUGATE GRADIENT METHOD FOR LINEAR AND NONLINEAR OPERATOR EQUATIONS", SIAM J. Numer. Anal., No. 4, 1967, PP. 10-26.

[104] J.K. Reid, "ON THE METHOD OF CONJUGATE GRADIENTS FOR THE SOLUTION OF LARGE SPARSE SYSTEMS OF LINEAR EQUATIONS", Proc. Conf. Large Sparse Sets of Linear Equations, Academic Press, New York, 1971, PP. 231-254.

[105] J.K. Reid, "THE USE OF CONJUGATE GRADIENTS FOR SYSTEMS OF LINEAR EQUATIONS POSSESSING PROPERTY A", SIAM J. Numer. Anal., No. 9, 1972, PP. 325-332.

[106] C.P. Jackson, P.C. Robinson, "A NUMERICAL STUDY OF VARIOUS ALGORITHMS RELATED TO THE PRECONDITIONED CONJUGATE GRADIENT METHOD", Int. J. for Numer. Methods in Engng., Vol. 21, 1985, PP. 1315-1338.

[107] H.A. Van der Vorst, "THE CONVERGENCE BEHAVIOUR OF PRECONDITIONED CG AND CG-S IN THE PRESENCE OF ROUNDING ERRORS", in Lecture Notes in Mathematic, Proc. Preconditioned Conjugate Gradient Method, edited by O. Axelsson, L.Yu. Kolotilina, June 19-21, 1989, PP. 126-136.

[108] O. Axelsson, N. Munksgaard, "A CLASS OF PRE-CONDITIONED CONJUGATE GRADIENT METHODS FOR THE SOLUTION OF A MIXED FINITE ELEMENT DISCRETIZATION OF THE BIHARMONIC OPERATOR", Int. J. Numer. Methods Engng., Vol. 14, 1979, PP. 1001-1019.

[109] A. Van der Sluis, H.A. Van der Vorst, "THE RATE OF CONVERGENCE OF CONJUGATE GRADIENTS", Numer. Math., No. 48, 1986, PP. 543-560.

[110] Eleanor Chu, Alan George, Joseph Liu, Esmond Ng, **SPARSPAK: Waterloo Sparse Matrix Package User's Guide for SPARSPAK-A**, Department of Computer Science, University of Waterloo, Waterloo, Ontario, CANADA, Research Report CS-84-36, November 1984.

[111] Alan George, Esmond Ng, **SPARSPAK: Waterloo Sparse Matrix Package User's Guide for SPARSPAK-B**, Department of Computer Science, University of Waterloo, Waterloo, Ontario, CANADA, Research Report CS-84-37, November 1984.

[112] A.S. Householder, **PRICIPLES OF NUMERICAL ANALYSIS**, McGraw-Hill, New York, 1953.

[113] S. Ratnajeevan H. Hoole, **COMPUTER-AIDED ANALYSIS AND DESIGN OF ELECTROMAGNETIC DEVICES**, Elsevier, New York, 1989.

[114] S. McFee, D.A. Lowther, "TOWARDS ACCURATE AND CONSISTENT FORCE CALCULATION IN FINITE ELEMENT BASED COMPUTATIONAL MAGNETOSTATICS", IEEE Trans. on Magnetics, Vol. Mag-23, No. 5, PP. 3771-3773, September 1987.

[115] J. Simkin,"RECENT DEVELOPMENTS IN FIELD AND FORCE COMPUTATION", J. Phys. Colloq. (France), Vol. 45(C-1), January 1984, PP. 851-860.

[116] J.P. Webb, "THE CONJUGATE GRAGIENT METHOD", Lecture notes, Electrical Engineering Department, McGill University, February 2, 1988.

[117] IEEE Committee Report, "PARALLEL PROCESSING IN POWER SYSTEM COMPUTATION", IEEE Trans. on PAS, Vol. 7, No. 2, May 1992, PP. 629-638.

[118] G.P. Granelli, M. Montagna, G.L. Pasini, "VECTOR COMPUTER IMPLIMENTATION OF POWER FLOW OUTAGE STUDIES", IEEE Trans. on PAS, Vol. 7, No. 2, May 1992, PP. 798-804.

[119] J.S. Chai, A. Bose, "BOTTLE NECKS IN PARALLEL ALGORITHMS FOR POWER SYSTEM STABILITY ANALYSIS", IEEE Trans. on PAS, Vol. 8, No. 1, February 1993, PP. 9-15.

[120] Hossein Javidi, S. McFee, F.D. Galiana, "INVESTIGATION OF EIGENVALUE CLUSTERING BY MODIFIED INCOMPLETE CHOLESKY DECOMPOSITION IN POWER NETWORK MATRICES", Proceedings of PSCC, August 30 to September 3, 1993, Avignon, France, PP. 189-195.

[121] Hossein Javidi, S. McFee, F.D. Galiana, "ARTIFICIALLY SYNTHESIZING NETWORK DATA FOR POWER SYSTEM ANALYSIS", Proceedings of 1993 Canadian Conference on Electrical and Computer Engineering, September 14-17 1993, Vancouver, Canada, PP. 566-569.

[122] R. Fletcher, "CONJUGATE GRADIENT METHODS FOR INDEFINITE SYSTEMS", Proceedings of Dundee Conference on Numerical Analysis, 1975, edited by G.A. Watson, Springer-Verlag, Berlin, 1976.

[123] C.C. Paige, M.A. Saunders, "SOLUTION OF SPARSE INDEFINITE SYSTEMS OF LINEAR EQUATIONS", SIAM J. Numer. Anal., Vol.12, 1975, PP. 617-629.

[124] H.P. Langtangen, "CONJUGATE GRADIENT METHODS AND ILU PRECONDITIONING OF NON-SYMMETRIC MATRIX SYSTEMS WITH ARBITRARY SPARSITY PATTERNS", Int. J. Numerical Methods in Fluids, Vol. 9, No. 2, February 1989, PP. 213-234.

[125] D.A.H. Jacobs, "A GENERALIZATION OF THE CONJUGATE-GRADIENT METHOD TO SOLVE COMPLEX SYSTEMS", IMA Journal of Numerical Analysis, Vol. 6, 1986, PP. 447-452.

[126] R.W. Freund, "CONJUGATE GRADIENT-TYPE METHODS FOR LINEAR SYSTEMS WITH COMPLEX SYMMETRIC COEFFICIENT MATRICES", SIAM J. Sci. Stat. Comput., Vol. 13, No. 1, January 1992, PP. 425-448.

[127] R.W. Freund, "A TRANSPOSE-FREE QUASI-MINIMAL RESIDUAL ALGORITHM FOR NON-HERMITIAN LINEAR SYSTEMS", SIAM J. Sci. Comput., Vol. 14, No. 2, March 1993, PP. 470-482.