FAST LOAD-FLOW ALGORITHMS

by

Hossein Jalali-Kushki

B.Sc., Arya-Mehr University of Technology, Tehran, Iran, 1971
M.A.Sc., University of British Columbia, Vancouver, Canada, 1973

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
THE FACULTY OF GRADUATE STUDIES
in the Department
of
Electrical Engineering

We accept this thesis as conforming to
the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
May, 1977

© Hossein Jalali-Kushki, 1977
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the Head of my Department or by his representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of  Electrical Engineering

The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

Date  September 1, 1977
ABSTRACT

New, fast and reliable algorithms for solving load-flow problems are presented in this thesis. Each of these algorithms iteratively solves a set of linear equations in terms of voltage magnitude squared and phase angles, and converges onto the final solution in a few iterations. Although the line losses of the system are used in deriving the equations of the basic line-loss load-flow algorithm, knowledge of their (approximate) values is not a prerequisite to using the algorithm. The basic line-loss load-flow algorithm is slightly modified to give an incremental-change line-loss algorithm which proves to be always preferable to the basic algorithm. By exploiting the weak interdependence between active power and voltage magnitude, and between reactive power and phase angle, two decoupled versions of the incremental-change line-loss algorithm were also developed.

All these algorithms have constant gradient characteristics, and their storage requirements are, at most, the same as those of the standard Newton-Raphson algorithm. If need be, the storage requirements can be reduced to those of the triangularized Y-matrix iterative algorithms.

Tests on various systems indicate fast and reliable convergence characteristics better than those of the Newton-Raphson algorithm and comparable to those obtained by Stott and Alsac with their decoupled Newton-Raphson load-flow algorithm.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF ILLUSTRATIONS</td>
<td>vi</td>
</tr>
<tr>
<td>NOMENCLATURE</td>
<td>vii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>ix</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 The Load-Flow Problem</td>
<td>1</td>
</tr>
<tr>
<td>1.2 The Review of Available Load-Flow Algorithms</td>
<td>5</td>
</tr>
<tr>
<td>1.3 The Newton-Raphson Algorithm</td>
<td>9</td>
</tr>
<tr>
<td>1.4 The Fast Decoupled (Newton-Raphson) Load Flow</td>
<td>12</td>
</tr>
<tr>
<td>1.5 The Line-Loss Algorithm</td>
<td>15</td>
</tr>
<tr>
<td>2. THE LINE-LOSS ALGORITHM</td>
<td>17</td>
</tr>
<tr>
<td>2.1 Derivation of the Algorithm</td>
<td>19</td>
</tr>
<tr>
<td>2.2 The Iterative Process</td>
<td>25</td>
</tr>
<tr>
<td>2.3 Programming and Storage Requirements</td>
<td>28</td>
</tr>
<tr>
<td>2.4 The Numerical Results</td>
<td>30</td>
</tr>
<tr>
<td>3. THE INCREMENTAL-CHANGE LINE-LOSS ALGORITHM</td>
<td>38</td>
</tr>
<tr>
<td>3.1 Derivation of the Algorithm</td>
<td>38</td>
</tr>
<tr>
<td>3.2 The Iterative Process</td>
<td>42</td>
</tr>
<tr>
<td>3.3 Programming and Storage Requirements</td>
<td>44</td>
</tr>
<tr>
<td>3.4 The Numerical Results</td>
<td>48</td>
</tr>
<tr>
<td>4. FAST DECOUPLED LINE-LOSS ALGORITHMS</td>
<td>60</td>
</tr>
<tr>
<td>4.1 Derivation of the Algorithm</td>
<td>60</td>
</tr>
<tr>
<td>4.2 The Iterative Process</td>
<td>63</td>
</tr>
<tr>
<td>4.3 Programming and Storage Requirements</td>
<td>65</td>
</tr>
<tr>
<td>4.4 The Numerical Results</td>
<td>66</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5. CONCLUSIONS</td>
<td>73</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>75</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>83</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>160</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2.1</td>
<td>Dimensions of &quot;Element&quot; (i,k) of the Matrix of Coefficients</td>
</tr>
<tr>
<td>2.2</td>
<td>Test Systems Used</td>
</tr>
<tr>
<td>2.3</td>
<td>Comparison of Number of Iterations</td>
</tr>
<tr>
<td>2.4</td>
<td>Comparison of Computation Times</td>
</tr>
<tr>
<td>3.1</td>
<td>Comparison of Number of Iterations</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparison of Computation Times</td>
</tr>
<tr>
<td>4.1</td>
<td>Comparison of Number of Iterations</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of Computation Times</td>
</tr>
<tr>
<td>Figure</td>
<td>Illustration</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2.1</td>
<td>A Simple Branch Representation</td>
</tr>
<tr>
<td>2.2</td>
<td>The Branch Corresponding to that of Fig. 2.1</td>
</tr>
<tr>
<td>2.3</td>
<td>$\pi$-Representation of Branch $(i,k)$</td>
</tr>
<tr>
<td>2.4</td>
<td>Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 9</td>
</tr>
<tr>
<td>2.5</td>
<td>Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 3</td>
</tr>
<tr>
<td>2.6</td>
<td>Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 6</td>
</tr>
<tr>
<td>3.1</td>
<td>Typical Voltage Magnitude and Phase Angle Variations for Test System 9</td>
</tr>
<tr>
<td>3.2</td>
<td>Typical Voltage Magnitude and Phase Angle Variations For Test System 10</td>
</tr>
<tr>
<td>3.3</td>
<td>Comparison of the Algebraic Sum of all the Power Mismatches (Test System 8)</td>
</tr>
<tr>
<td>3.4</td>
<td>Comparison of Maximum Power Mismatches (Test System 8)</td>
</tr>
<tr>
<td>3.5</td>
<td>Comparison of Sum of all the Absolute Values of Power Mismatches (Test System 8)</td>
</tr>
<tr>
<td>4.1</td>
<td>Flow-Chart Indicating the Logic of Decoupled Algorithms</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of the Algebraic Sum of all the Power Mismatches (Test System 8)</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of the Maximum Power Mismatches</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of Sum of all the Absolute Values of Power Mismatches</td>
</tr>
</tbody>
</table>
NOMENCLATURE

Voltage
\[ V_i \] complex voltage of bus \( i \)
\[ U_i \] magnitude of \( V_i \)
\[ \delta_i \] phase angle of \( V_i \)
\[ e_i \] complex voltage of bus \( i \) in the \((S-U^2)\) system

Power
\[ S_{ik} \] complex power flowing from bus \( i \) towards bus \( k \)
\[ S_i \] total complex power injection at bus \( i \)
\[ S_{ik}^2 \] complex power loss in line \((i,k)\)
\[ P_{ik}, P_i, P_{ik}^2 \] real parts of \( S_{ik}, S_i, S_{ik}^2 \), respectively
\[ Q_{ik}, Q_i, Q_{ik}^2 \] imaginary parts of \( S_{ik}, S_i, S_{ik}^2 \), respectively

Admittance
\[ y_{ik} \] value of admittance connecting buses \( i \) and \( k \)
\[ y_i \] total shunt admittance at bus \( i \)
\[ Y \] nodal admittance matrix
\[ Y_{ij} \] elements \((i,j)\) of \( Y \)
\[ G \] real part of \( Y \)
\[ B \] imaginary part of \( Y \)
\[ C_{ij}, B_{ij} \] elements \((i,j)\) of \( G \) and \( B \), respectively
\[ g_i \] total shunt conductance at bus \( i \)
\[ b_i \] total shunt susceptance at bus \( i \)
\[ x_{ik} \] reactance of branch \((i,k)\)
General

(i,k) branch connecting buses i and k
*
complex conjugate

j complex operator, \( \sqrt{-1} \)

Im imaginary part of a complex number

Re real part of a complex number

\( \varepsilon_{ik} \) correction term corresponding to branch (i,k)

n number of buses

\( n_{PV} \) number of P-V buses

\( n_{PQ} \) number of P-Q buses

i ∈ P-Q i is a P-Q (or P-V) bus
or P-V

Δ differential operator

superscript (v) indicates iteration number
ACKNOWLEDGMENT

I would like to express my deepest gratitude to my supervisor, Dr. M.D. Wvong, for his continued encouragement, help and guidance throughout the research and writing of this thesis.

I would also like to thank Dr. Y.N. Yu, for his constant encouragement during the research work and for proof-reading the thesis and offering valuable suggestions.

I am also grateful to Dr. H.W. Dommel for proof-reading the thesis and also for providing me the excellent opportunity to visit the Bonneville Power Administration and have valuable discussions with the people there.

Thanks are also due to Miss Sannifer Louie for typing the thesis.

The financial support from the National Research Council of Canada is gratefully acknowledged.
1. INTRODUCTION

1.1 The Load-Flow Problem

"A power system continuously experiences changes in its operating state. These changes can be due to load demand variations, rescheduling of power generation, redistribution of reactive power generation, changing tap settings of transformers or phase shifters and disconnecting lines, transformers and generators for maintenance or as a consequence of system faults. The effect of these variations is investigated both during system planning and operation. One of the objectives is to ensure that the quality of service provided to the consumers is acceptable. Another is to confirm that these operations do not cause overloading of system elements. Yet another objective during system planning is to determine the range of tap settings of transformers and the capacities of the excitors required for proper steady-state and transient operation. The real and reactive power flow patterns are also studied as an important part of system security monitoring process.... A large number of load flow solutions will have to be obtained for a present day power system." [G16]

"Power flow studies are the backbone of the design of a power system. They are the means by which the future operation of the system is known ahead of time .... The continual expansion of the demand for electrical energy due to the growth of industries, commercial centers, and residential sections, require never-ending additions to existing power systems. The system engineer must decide what components must be added to the system many years before they are put into operation and he does this by means of power flow studies." [R5]
"The power flow program is indispensible for the analysis of ac power systems. Although improved solution methods and faster computers have extended its capabilities, this gain has been largely offset by the increased number and size of studies that must be made." [D7]

"An increasing use is being made of Power Flow algorithms in the electric utility industry, and recently the interest has been in applying a form of the algorithm to real time situations. The new uses of Power Flow imply that faster and more efficient algorithms are desired." [N7]

"The load-flow program is certainly the most frequently used computer program for power system applications. It consumes the largest computer time per year. This time has been growing as power system planners see the need for solving ever larger cases in succession of base case and many contingent solutions .... The great demand that load-flow applications place on computing resources accounts for the continued interest in the development of techniques that are faster, exhibit quicker convergence for a variety of system conditions, and make efficient use of core." [D8]

These statements of the "authorities" in the field - from American Electric Power Services Corporation, to IBM Scientific Center, to Bonneville Power Administration - should suffice to indicate the great importance of load-flow studies and the necessity of developing even faster and more efficient load-flow algorithms in order to deal with the huge interconnected power systems of today.

The problem is to specify a set of complex voltages for the system nodes (or buses) that satisfy certain requirements. Depending upon these requirements, the buses of the system can be divided into three main categories:
(a) The load or P-Q buses; whose active and reactive power injections are given and whose voltage magnitudes and phase angles have to be determined. A system may have any number of these buses.

(b) The voltage regulated, generator or P-V buses; whose voltage magnitudes and active power injections are given and whose phase angles have to be specified. A system may have any number of these buses.

(c) The reference, swing or slack buses; whose voltage magnitudes and phase angles are pre-determined and have no constraints on their power injections. A system can have any number of these buses but must at least have one.

The problem can be formulated in terms of node or mesh variables. The most commonly used formulation is the one using the nodal equations:

\[ I = Y \cdot V \]  \hspace{1cm} (1-1)

where \( I \) is the vector of injected nodal currents, \( Y \) is the nodal admittance matrix and \( V \) is the vector of nodal voltages. However, since neither the voltages nor the currents are known, the nodal equations alone will not be adequate. Additional equations are required to relate the known quantities, the active and reactive power injections, to the nodal voltages and currents. These are:

\[ S_i = V_i^* I_i \quad i = 1, \ldots, n \]  \hspace{1cm} (1-2)
where \( S_i \) is the total complex power injection at node \( i \) and * is used to indicate the conjugate of a complex number. The nonlinearity of the load-flow problem arises from the fact that:

1. equation (1-2) is nonlinear, and
2. for P-V buses, the problem is further complicated because, in equation (1-2), only the real part of \( S_i \) and the magnitude of \( V_i \) are known.

Equations (1-1) and (1-2) represent the basic formulation of the load-flow problem. All the load-flow algorithms that use nodal analysis, use some combination of these equations to correct the voltages - in steps - from a set of initial estimates onto the final solution. The differences in the convergence characteristics and the computing speed of various algorithms are due to the fact that each algorithm uses (1-1) and (1-2) differently.

Normally the per unit quantities of the variables are used and certain practical assumptions are usually made in developing the load-flow algorithms. Among these are the facts that for any practical power system:

1. The voltage magnitudes are approximately 1 P.U. and all the phase differences across various branches are small;
2. The transmission lines are highly inductive;
3. The interdependence between active powers and voltage magnitudes and between reactive powers and phase angles are weak and may be neglected; and
4. The losses of the system constitute a small percentage of the total generated power.
The first assumption is made not only in developing load-flow algorithms, but also in choosing a reliable starting point. This is normally referred to as a "flat-start", and is obtained by setting all P-Q bus voltage magnitudes to 1 P.U. and all the phase angles to zero*.

Other practical considerations, such as the required speed and accuracy of the solution, play a role both in developing load-flow algorithms and in choosing a particular algorithm for a specific study.

1.2 Review of Available Load-Flow Algorithms

Before digital computers became available, the load-flow problem was solved on network analysers. The first totally automatic algorithm for solving the load-flow problem on a digital computer was presented by Ward and Hale in 1956 [Y4]. Since then, considerable attention has been paid to the development of faster, more efficient and more reliable load-flow algorithms and enormous progress has been made in this direction**. This section is intended to give a general account of the development of load-flow algorithms. It is by no means a complete and exhaustive survey of the literature.

The Ward-Hale algorithm [Y4], as it is normally referred to, is one of a group of load-flow algorithms classified as the "Y-matrix Iterative Methods" [Y1-Y4]. The reason for this classification is that all these algorithms are based on the iterative solution of the set of linear equations in (1-1). Their storage requirements are minimal and, for that reason, they were very suitable to the early generation of computers. Furthermore, the Y-matrix iterative methods are easy to

* Other starting points are also used. For example, in assessing the system outages, the base case solution may be preferable to the flat-start values.

** See "Bibliography" for a partial listing.
program and perform satisfactorily on many problems. The changes in the system configuration can be easily accounted for and the automatic controls can be easily included in these algorithms.

On the other hand, the Y-matrix iterative methods converge very slowly and sometimes do not converge at all. In practice, acceleration techniques are invariably used to speed up the convergence of these algorithms. However, even with the use of the "best" acceleration factors, for large systems, the total computation time of these algorithms is far greater than that of the newer methods. Nowadays, with the constant growth of the power systems and the increase in the number of load-flow problems that have to be solved for each system, and also due to the appearance of the modern and very powerful digital computers with storage capabilities far beyond those of the earlier computers, the Y-matrix iterative methods are becoming less and less attractive.

In the early 1960's another group of load-flow algorithms, classified as the "Z-matrix Methods" [Z1-Z4], were introduced. These algorithms, which are based on the direct solution of the linear set of equations in (1-1), have more reliable convergence characteristics than the Y-matrix iterative methods. On the other hand, their storage requirements and computation times are considerably more and grow enormously with system size. They are rarely competitive with the Y-matrix iterative methods and still less competitive with the newer load-flow algorithms. The Z-matrix methods did not become very popular.

In 1959, the Newton-Raphson method of solving the load-flow problem was shown to have powerful convergence characteristics [N13, N14]. Due to its large storage and computer time requirements, however, it was not competitive with the Y-matrix iterative methods. It was not until
1967, when, by using sparsity programming and ordered Gaussian elimination these problems were overcome \([G23,N11]\), that the algorithm became practical and widely used.

Because of its quadratic convergence, the Newton-Raphson algorithm converges very rapidly to very accurate solutions. For practical accuracies, it always converges in a few iterations, irrespective of the system size. It is far superior to any of the Y-matrix iterative methods but it is also far more complicated to program. Its storage requirements, even with sparsity programming and ordered elimination, are much more than those of the Y-matrix iterative methods, but impose no serious problem on most present-day computers.

Soon after the appearance of the paper by Tinney and Hart in 1967 \([N11]\), the Newton-Raphson algorithm became very popular and replaced the "classical" Ward-Hale algorithm. Subsequently, a great deal of work was done in the direction of developing the Newton-Raphson load-flow algorithm and improving upon certain aspects of it. The polar form of this algorithm became to be widely regarded as the standard method of solving load-flow problems. Even today, the Newton-Raphson method of solving load-flow problems is the algorithm which is most widely used by the industry. Also, this algorithm is almost invariably used as the basis of comparison for the newly developed load-flow algorithms.

In 1967, Bonaparte and Maslin \([D1]\) presented a new load-flow algorithm which was based on exploiting the weak interdependence between real power flows and voltage magnitudes. In this algorithm - the DC load-flow as it is normally called - a set of linear equations relating the real power injections to voltage phase angles is directly solved at each iteration. The matrix of coefficients remains constant throughout
the process and, therefore, the computation time per iteration is greatly reduced. The algorithm was proposed as a means of obtaining fast and approximate real power flow solutions and as such, it became very popular and widely used.

The idea of exploiting the weak interdependence between the real powers and voltage magnitudes and between the reactive powers and voltage phase angles of any practical power system was used in several subsequent papers. In the proposed decoupled algorithms, the P-U and Q-δ couplings are altogether neglected and the load-flow equations are simplified. The idea of decoupling the load-flow equations did not become very popular until, in a recent paper by Stott and Alsac [D11], it was shown that a decoupled version of the Newton-Raphson algorithm has better characteristics than the original undecoupled algorithm. The Fast Decoupled (Newton-Raphson) Load Flow, as the authors called it, is becoming increasingly popular [D5,D8] and is expected to replace the Newton-Raphson algorithm.

Many other load-flow algorithms have been developed during the past two decades which do not belong to any of the above categories, In fact, with the incredible growth of power systems, the scope of the problem has become so wide and the number of publications related to the subject so numerous that it would be practically impossible for this survey to give a complete coverage. A more complete account of the development in this field can be found in several review papers [R1-R6] that have appeared on the subject.
1.3 The Newton-Raphson Algorithm \[^{(6)}\] [N11]

Combining equations (1-1) and (1-2) we obtain:

\[
S_i = V_i \sum_{k=1}^{n} Y_{ik} V_k^* \tag{1-4}
\]

where \( Y_{ik} \) is the \((i,k)\)th element of the nodal admittance matrix, \( Y \).

Using polar coordinates:

\[
V_i \Delta U_i e^{j\delta_i} \\
Y_{ik} \Delta \left| Y_{ik} \right| e^{j\theta_{ik}} \tag{1-5}
\]

where \( j \) is the complex operator, \( \sqrt{-1} \), equation (1-4) becomes:

\[
S_i = P_i + jQ_i = \sum_{k=1}^{n} U_i U_k \left| Y_{ik} \right| e^{j\left(\delta_i - \delta_k - \theta_{ik}\right)} \tag{1-6}
\]

The Newton-Raphson technique is then used to solve the set of nonlinear equations in (1-6).

The elements of the Jacobian matrix are the partial derivatives of \( P_i \) and \( Q_i \) with respect to voltage magnitudes and phase angles:

\[
\Delta P_i = \sum_{k=1}^{n} \left( H_{ik} \Delta \delta_k + N_{ik} \frac{\Delta U_k}{U_k} \right) \tag{1-7a}
\]

and

\[
\Delta Q_i = \sum_{k=1}^{n} \left( J_{ik} \Delta \delta_k + L_{ik} \frac{\Delta U_k}{U_k} \right) \tag{1-7b}
\]

where:

\[
H_{ik} \Delta \frac{\partial P_i}{\partial \delta_k} ; \quad N_{ik} \Delta \frac{\partial P_i}{\partial U_k} U_k \tag{1-8}
\]

\[
J_{ik} \Delta \frac{\partial Q_i}{\partial \delta_k} ; \quad L_{ik} \Delta \frac{\partial Q_i}{\partial U_k} U_k
\]

\[^{6}\] Throughout this thesis, the phrase "the Newton-Raphson algorithm" refers to the application of this well-known algorithm for solving a set of nonlinear equations to the polar formulation of the load-flow problem.
and

\[ \Delta P_i \triangleq P_i \text{(scheduled)} - P_i \text{(calculated)} \]

\[ \Delta Q_i \triangleq Q_i \text{(scheduled)} - Q_i \text{(calculated)} \]

The partial derivatives defined in (1-8) are real functions of the admittance matrix and node voltages. Even though the problem is formulated in polar coordinates, these partial derivatives should be calculated by rectangular complex arithmetic. Assuming the following rectangular expressions for the admittances and voltages:

\[
V_i = e_i + jf_i
\]

\[
Y_{ik} = G_{ik} + jB_{ik}
\]

and defining:

\[
a_k + jb_k = (e_k + jf_k)(G_{ik} + jB_{ik})
\]

we have for \( i \neq k \):

\[
H_{ik} = L_{ik} = a_k f_i - b_k e_i
\]

\[
N_{ik} = -J_{ik} = a_k e_i + b_k f_i
\]

and for \( i = k \):

\[
H_{ii} = -Q_i - B_{ii} U_i^2
\]

\[
L_{ii} = Q_i - B_{ii} U_i^2
\]

\[
N_{ii} = P_i + G_{ii} U_i^2
\]

\[
J_{ii} = P_i + G_{ii} U_i^2
\]
For a system of \( n \) nodes (not including the slack buses) of which \( n_{\text{PQ}} \) are P-Q type, the number of unknowns is \( n + n_{\text{PQ}} \). The number of equations is likewise \( n + n_{\text{PQ}} \) because for every P-V or P-Q bus we can write (1-7a) and for every P-Q bus we can also write (1-7b). In this way, for every P-Q bus, there is a "double row" and a "double column" in the Jacobian matrix; for every P-V bus there is a single row and a single column; and there are no rows or columns corresponding to the slack buses. The element \((i,k)\) of the Jacobian matrix is therefore a submatrix whose dimensions are 2-by-2, 2-by-1, 1-by-2, or 1-by-1, depending upon the types of buses \( i \) and \( k \). The structure of the Jacobian matrix would then be the same as that of the nodal admittance matrix, but the former will not be symmetrical even if the latter is.

At every step of the iterative process, the Jacobian matrix is evaluated and the following set of linear equations is directly solved:

\[
\begin{bmatrix}
\Delta P \\
\Delta Q
\end{bmatrix} = 
\begin{bmatrix}
H & N \\
J & L
\end{bmatrix}
\begin{bmatrix}
\Delta \delta \\
\Delta U
\end{bmatrix}
\]

(1-12)

using Gaussian elimination; i.e., through the process of triangularization and back substitution. Although the original Jacobian matrix is highly sparse, because of the new elements that are created in the course of the triangularization process, its triangularized form may not be sparse any more. In order to prevent this, Tinney and Hart [N11] suggested several ordering schemes for the nodes of the system that tend to reduce the number of newly accumulated elements to a minimum. One of these schemes, which the authors used in their program and which is also used
in LFP*, calls for numbering the nodes "so that at each step of the elimination the next node to be eliminated is the one having the fewest connected branches. This method requires simulation of the elimination process to take into account the changes in the node-branch connections effected at each step".

Using the above scheme, the rows of the Jacobian matrix are ordered only once in the beginning of the process. The same order is then used at every iteration step. The augmented Jacobian matrix is formed and triangularized, row by row in that order, and the result is stored in compact form. The sign of integer row and column pointers indicate types of rows and columns and, therefore, number and order of the elements of the Jacobian submatrices. This storage scheme, which is explained in detail in [N11] was also used in programming the Newton-Raphson algorithm in LFP.

1.4 The Fast Decoupled (Newton-Raphson) Load-Flow [D11]

This algorithm was first presented in 1973 in a paper by Stott and Alsac [D11]. The polar form of the Newton-Raphson algorithm is taken as the starting point. This means starting with equation (1-12), repeated here for convenience:

\[
\begin{bmatrix}
\Delta P \\
\Delta Q \\
\end{bmatrix} = \begin{bmatrix} H & N \\
J & L \\
\end{bmatrix} \begin{bmatrix} \Delta \delta \\
\Delta U \\
\end{bmatrix} \quad (1-12)
\]

The algorithm takes advantage of the weak interdependence between real powers and voltage magnitudes and between the reactive powers and phase

---

* LFP is a collection of programs for solving the load-flow problem using various algorithms. The programs were written for the purpose of comparing the performance of various algorithms. See Appendix A.
angles (for practical power systems operating in steady state) to de-
couple P-δ and Q-U problems. "The first step" in this direction is to
neglect the coupling submatrices [N] and [J] in (1-12), obtaining two
separated sets of equations:

\[
[\Delta P] = [H][\Delta \delta]
\]
\[
[\Delta Q] = [L][\Delta \frac{U}{U}]
\]

where the elements of [H] and [L], for \( i \neq k \), are

\[
H_{ik} = L_{ik} = U_i U_k [G_{ik} \sin(\delta_i - \delta_k) - B_{ik} \cos(\delta_i - \delta_k)]
\]

and:

\[
H_{ii} = -B_{ii} U_i^2 - Q_i
\]
\[
L_{ii} = -B_{ii} U_i^2 + Q_i
\]

However, it was found that the decoupled equations of (1-13) represent
an unstable algorithm unless other practical assumptions are made and
an alternative equation is used instead of (1-13b) [D9]. The "best"
stable algorithm was derived in [D11] using further physically justi-
fiable simplifications as follows:

"In practical power systems the following assumptions are
almost always valid:

\[
\cos(\delta_i - \delta_k) \approx 1;
\]
\[
G_{ik} \sin(\delta_i - \delta_k) \ll B_{ik};
\]
\[
Q_i \ll B_{ii} U_i^2
\]

so that good approximations to (1-13a) and (1-13b) are:
\[ \Delta P = [U \cdot B' \cdot U] \Delta \delta \] (1-15a)

\[ \Delta Q = [U \cdot B'' \cdot U] \frac{\Delta U}{U} \] (1-15b)

where \([B']\) and \([B'']\) are matrices whose elements are strictly elements of \([-B]\) and \([U \cdot B' \cdot U]\) represents a matrix whose element \((i,k)\) has the expression \(U_i \cdot B'_{ik} \cdot U_k\). The decoupling process and the final algorithmic forms are now completed by:

(a) omitting from \([B']\) the representation of those network elements that predominantly affect MVAR flows; i.e., shunt reactances and off-nominal in-phase transformer taps
(b) omitting from \([B'']\) the angle shifting effects of phase shifters
(c) taking the left-hand \(U\) terms in (1-15a) and (1-15b) onto the left-hand sides of the equations, and then in (1-15a) removing the influence of MVAR flows on the calculation of \([\Delta \delta]\) by setting all the right-hand \(U\) terms to 1 P.U. Note that the \(U\) terms on the left-hand sides of (1-15a) and (1-15b) affect the behaviour of the defining functions and not the coupling
(d) neglecting series resistances in calculating the elements of \([B']\), which then becomes the DC approximation load-flow matrix. This is of minor importance, but is found experimentally to give slightly improved results."

With the above modifications the final fast decoupled load-flow equations become:

\[ \frac{\Delta P}{U} = [B'][\Delta \delta] \] (1-16a)

\[ \frac{\Delta Q}{U} = [B''][\Delta U] \] (1-16b)
In the above algorithm, both \([B']\) and \([B'']\) are constant, real and sparse matrices that need to be triangularized only once at the beginning of the study. Equations (1-16a) and (1-16b) are solved separately and repeatedly until all the power flows are within the desired tolerance of their scheduled values. Starting with (1-16a), equations (1-16a) and (1-16b) are solved alternatively unless all the active powers or all the reactive powers are within the desired tolerance of their final values in which case the solution of the corresponding set of equations is skipped. The solution of either (1-16a) or (1-16b) is considered to be one half iteration. Experiments have shown that the convergence of this algorithm is very fast and reliable [D5,D8,D11] and that the algorithm is much preferred to the undecoupled Newton-Raphson algorithm.

In LFP, (1-16a) and (1-16b) are solved using ordered Gaussian elimination. The buses are numbered such that at each stage of the elimination process of \([B']\), the next row to be eliminated is the one having the fewest number of off-diagonal elements. The ordering is assumed to be the same for \([B']\) and \([B'']\). Both these matrices are formed and triangularized row by row and the results are separately stored in compact form. Negative pointers are used to indicate P-V type buses.

1.5 The Line-Loss Algorithms [09,014,D4]

Applied numerical methods are most efficient when they take advantage of the physical properties of the system being solved. For example, exploiting the weak (P-U) and (Q-δ) interdependencies and other physical properties of the practical power systems, resulted in the development of the Fast Decoupled (Newton-Raphson) Load-Flow, which is
shown to have much better characteristics than the undecoupled Newton-Raphson algorithm.

In any practical electrical power transmission network, the line losses of the system form a very small percentage of the system's total power generation. In the following chapters, this physical property of the system is exploited and a number of very fast and reliable load-flow algorithms are developed. The idea of using the line losses of the system in a load-flow algorithm is new and results in a set of linear equations in terms of voltage magnitudes squared and voltage phase angles. The solution of the load-flow problem is obtained by repeatedly solving this set of equations.

Although knowledge of the approximate values of line losses can be effectively used with the line-loss algorithms, such knowledge is not a pre-requisite to using the algorithms. Since losses are usually small, their values may be initially set to zero. Alternatively, a flat start may be used to start the algorithms.

Two slightly different formulations of the line-loss algorithm, as well as two decoupled versions of this algorithm, are presented in this thesis. In Chapter 2, the basic line-loss algorithm is derived. Chapter 3 contains a modified version of this algorithm which is faster and has better convergence characteristics. The decoupled versions of the algorithm are introduced in chapter 4. The performance of these algorithms are compared with those of the Newton-Raphson algorithm and the Fast Decoupled (Newton-Raphson) Load-Flow in respective chapters.
2. THE LINE-LOSS ALGORITHM

During the last two decades, remarkable progress has been made in developing better algorithms for the digital solution of the load-flow problem. On the other hand, the same period has witnessed a tremendous growth in the size of power systems and the emergence of huge interconnected electrical power transmission networks involving many utilities, which has resulted in a sizeable increase in the number and size of load-flow problems to be solved. This growth has been so enormous that it has largely offset the gains of the aforementioned development. Consequently, due to the ever-increasing system size and the growing complexity of the problems associated with it, even today, there is still a need for faster, more efficient and more reliable load-flow algorithms. This is why the constant-gradient fast-converging algorithms, such as the Fast Decoupled (Newton-Raphson) Load-Flow [Dll] are becoming very popular. The line-loss load-flow algorithm presented here, is a constant-gradient algorithm with very fast and reliable convergence.

This algorithm was developed as the result of an attempt to transform the nonlinear load-flow problem into an ordinary linear circuit problem. It was noticed that, in a lossless* system, the power flow through a line remains constant and that, in any system, the algebraic sum of powers entering any node is zero. Both of these are characteristics of current. Therefore, in a lossless system, the nodal power injections may be assumed nodal currents. The solution to the load-flow

* Throughout this thesis the word "loss" always refers to "complex power loss". A lossless system, likewise, refers to an ideal system with zero impedances.
problem for the former case would then be the same as the solution of the current flow problem for the latter.

At the same time we know that, in any practical power system, the losses represent only a small percentage of the generated power. Therefore, we expect the above power/current analogy to be approximately valid for a practical power system. Furthermore, we expect this approximation to be even better if we know the (approximate) values of the line losses for our system. As a matter of fact it can be easily shown that, for a DC system, knowing the exact values of line losses means that we can transform the nonlinear load-flow problem into an exact linear equivalent. The variables in the linear equivalent model, are the square of the voltages in the original DC system.

Of course, for an ordinary AC system, the problem is far more complicated. On the one hand, for P-V buses, only the real power injections are specified and, on the other hand, the magnitudes of voltages for these buses are constrained. The problem is further complicated by the fact that there is not a one to one relationship between the voltages of our original AC network and the voltages which will be obtained by assuming nodal power injections to be currents. Hence, for an AC system, even if the exact values of line losses are known, the problem cannot be solved in one step. Using certain approximations, however, would result in an iterative algorithm for the digital solution of the load-flow problem. This algorithm, which may be derived using the expression for the line losses of the system, is called the line-loss algorithm.

From a mathematical point of view, the line-loss algorithm transforms the load-flow equations into a set of "almost linear" equations in terms of voltage magnitudes squared and voltage phase angles.
The nonlinear terms in the resulting formulation are very small. These terms are grouped together and taken to the known side of the equations. Their values are set to some initial estimates (usually zero) in the beginning of the process and are calculated using the updated nodal voltages at each subsequent iteration.

2.1 Derivation of the Algorithm

The algorithm can be derived in two different ways. The first is to use circuit analysis concepts and power/current analogy, while the second is to derive the algorithm by manipulating the power-flow equations from a mathematical point of view. The former approach led to the development of the algorithm while the latter approach was later used to indicate the relationship between the equations used in the line-loss algorithm and the exact load-flow equations. The following presentation will be made in the same order. The strictly mathematical derivation is given in section 3.1.

The idea, as mentioned earlier, is to extend the analogy that exists between the power flows and the currents of a lossless system, to practical systems for which the losses are usually very small. Assume a branch (i,k), represented - as shown in Fig. 2.1 - by an admittance $y_{ik}$ connecting nodes i and k.

![Fig. 2.1. A Simple Branch Representation.](image)
We know that, in a lossy system, the power "sent" from node i through the branch \((i,k)\) is different from the power "received" by node k. The difference between the two (or more precisely, the algebraic sum of the power flow from node i to node k and the power flow from node k to node i) is equal to the complex power losses in the branch \((i,k)\):

\[
S_{ik}^L = [U_i^2 + U_k^2 - 2U_iU_k \cos(\delta_i - \delta_k)]y_{ik}^\ast
\]

(2-1)

where \(U_i\) and \(\delta_i\) (\(U_k\) and \(\delta_k\)) represent the magnitude and the phase angle of voltage at node i (node k) and \(S_{ik}^L\) is the complex power loss in the branch \((i,k)\).

Now, if we subtract the losses from the actual power flow in branch \((i,k)\), the branch can be considered lossless. In other words, after subtracting the line losses, the power flow in the branch remains constant and can be treated as current. In order to subtract the line losses, which are in reality distributed along the branch, we can either introduce a new node at the center of the branch \((i,k)\) whose total injected power equals negative of \(S_{ik}^L\) in (2-1), or we can subtract \(S_{ik}^L/2\) from the powers flowing into both nodes i and k. The first approach resembles the way the T-equivalent of a line is obtained while the second approach would be similar to finding the \(\pi\)-equivalent of the line. The two methods are identical in every way except that the former requires introduction of new nodes into the system while the latter does not. For this reason, the second approach was chosen.

Subtracting half of the line losses, \(S_{ik}^L\) as given in (2-1), from the powers flowing into nodes i and k, would leave two identical values. The fact that these values, which still have the dimension of power, are equal indicate that we can use them as the power flowing
through a lossless line and, subsequently, use them as currents\(^*\). We represent the resulting power flow from node \(i\) to node \(k\), after subtracting \(\frac{S_{ik}^L}{2}\), by the symbol \(S'_{ik}\) to show that: 1) it is power, and, 2) it is different from the actual power flow (before subtracting the losses) which we call \(S_{ik}\). Using equations (1-1), (1-2) and (2-1), the expression for \(S'_{ik}\) can be easily found:

\[
S'_{ik} = S_{ik} \frac{S_{ik}^L}{2}
\]

\[
= \frac{1}{2} \left[ U_i^2 - U_k^2 - 2j U_i U_k \sin(\delta_i - \delta_k) \right] j y_{ik}^* \tag{2-2}
\]

where \(j\) is the complex operator, \(\sqrt{-1}\).

As mentioned earlier, we can now treat \(S'_{ik}\) as a current flowing through the branch. Equation (2-2) indicates that the best results would be obtained when the admittance of the branch is conjugated, as shown in Fig. 2.2.

\[
e_i = \frac{1}{2} U_i^2 - j \delta_i \quad y_{ik}^* \quad e_k = \frac{1}{2} U_k^2 - j \delta_k
\]

Fig. 2.2. The Branch Corresponding to that of Fig. 2.1.

If current \(S'_{ik}\) flows through the branch \((i,k)\) of Fig. 2.2, the voltage across the branch would be

\[
e_i - e_k = \frac{U_i^2}{2} - \frac{U_k^2}{2} - j U_i U_k \sin(\delta_i - \delta_k) \tag{2-3}
\]

\(^*\) The value of this "power-flow" through the lossless line is exactly equal to the power that was flowing through the middle point of the original branch.
where \( e_i \) and \( e_k \) represent the nodal voltages for the branch of Fig. 2.2.

The real part of each voltage, \( e_i \), can be readily equated to \( u_i^2 \frac{1}{2} \). This would be consistent with the results obtained for a DC network. The imaginary parts of the voltages cannot be so easily related to the voltages of the original branch, as the imaginary part of (2-3) represents a coupling between the voltages of nodes \( i \) and \( k \). However, we can write:

\[
U_i \sin(\delta_i - \delta_k) = \delta_i - \delta_k - \epsilon_{ik}
\]

(2-4)

where \( \epsilon_{ik} \) is a correction term which is usually very small since in practical power systems all the voltage magnitudes are approximately 1 P.U. and all the phase differences across the branches are small.

Using (2-3) and (2-4), we can assume the imaginary part of \( e_i \) to be \(-\delta_i\) and modify the current \( S_{ik} \) according to the value of the correction term \( \epsilon_{ik} \). In other words, using

\[
e_i = \frac{1}{2} u_i^2 - j \delta_i \quad i = 1, \ldots, n
\]

(2-5)

would result in a current equal to \( S_{ik} - j \epsilon_{ik} y^*_{ik} \) flowing through the branch \((i,k)\) of Fig. 2.2. For convenience, from now on, we refer to the original system as the (I-V) system and to the system whose admittances and voltages are as shown in Fig. 2.2, as the \( (S-U^2) \) system.

Writing the voltage/current relationship - equation (1-1) - for the \( (S-U^2) \) system, and considering the fact that, \( Y_{ik} \), the element \((i,k)\) of the nodal admittance matrix has the value of \(-y^*_{ik}\), we obtain:

\[
S_i - \frac{1}{2} \sum_{k=1}^{n} S_{ik}^2 + j \sum_{k=1}^{n} Y_{ik}^* \epsilon_{ik} = \sum_{k=1}^{n} Y_{ik}^* \left( \frac{U_{ik}^2}{2} - j \delta_k \right) \quad i = 1, \ldots, n
\]

(2-6)
The above equation has to be slightly modified when the branches of the (I-V) system are represented, as shown in Fig. 2.3, by their π-equivalents.

Fig. 2.3. π-Representation of Branch (i,k).

The problem can be handled by treating each "leg" of the π-equivalent as the simple branch in Fig. 2.1, but there is an easier approach: first we subtract the power consumed by each π-equivalent "leg" from the net power injection at the corresponding node. The remaining values would represent power flows through simple branches, like that in Fig. 2.1. Therefore, equation (2-6) becomes:

\[
S_1 - U_1^* y_1 - \sum_{k=1}^{n} \frac{S_{ik}^2}{2} + j \sum_{k=1}^{n} \gamma_{ik}^* \xi_{ik} = \sum_{k=1}^{n} \gamma_{ik}^* \left( \frac{U_k^2}{2} - j\delta_k \right) \tag{2-7}
\]

where \( y_1 \) represents the total shunt admittance at node \( i \):

\[
y_1 = \sum_{k=1}^{n} y_{ik} \tag{2-8}
\]

Using (2-8), equation (2-7) can be rewritten as:
The above equation represents what we call the line-loss algorithm. It is the current flow equation for the (S-U) system. At the same time, it has to (and does) correspond to the power-flow equation for the (I-V) system. Note that irrespective of the fact that the branches of the (I-V) system may have been represented by their \( \pi \)-equivalents, the corresponding branches in the (S-U) system are always represented as in Fig. 2.2.

By separating the real and imaginary equations in (2-9) we obtain the final form of the line-loss algorithm:

\[
S_i = \sum_{k=1}^{n} \frac{S_{ik}^2}{2} + j \sum_{k=1}^{n} Y_{ik}^* \varepsilon_{ik} = \sum_{k=1}^{n} Y_{ik}^* \left( \frac{U_k^2}{2} - j \delta_k \right) + \frac{U_i^2}{2} (y_i^* + y_i^*) - j \delta_i (y_i^* - y_i^*) \quad (2-9)
\]

\( i = 1, \ldots, n \)

\( \text{\textsuperscript{*}} \) See section 3.1.
where:

\[ S_i \triangleq P_i + jQ_i \]

\[ S_{ik} \triangleq P_{ik} + jQ_{ik} \]

\[ Y_{ik} \triangleq G_{ik} + jB_{ik} \]

\[ y_i \triangleq g_i + jb_i \]

For each P-Q bus, there are two unknowns, \( U_i \) and \( \delta_i \), and both equations (2-10a) and (2-10b) are written. For each P-V bus, there is only one unknown, \( \delta_i \), and only equation (2-10a) can be written. There are no equations written for slack buses whose voltages are known ahead of time. In the resulting set of equations, all the \( U \) terms corresponding to P-V buses and all the \( U \) and \( \delta \) terms corresponding to the slack buses are calculated and moved to the known side of the equations.

2.2 The Iterative Process

Once the set of equations in (2-10) is formed, it will be used in the following iterative process:

1. Initially, all the \( \varepsilon_{ik} \)'s are set to zero. The line losses are set to their initial estimates when such estimates are available. Otherwise, they are set to zero.

2. The resulting set of linear equations is directly solved using ordered Gaussian elimination and back substitution. The results will provide an initial set of voltages.

3. If the present set of voltages satisfy the power constraints the problem is solved. Otherwise, using the present values of voltages and equations (2-1) and (2-4), the new values of \( \varepsilon_{ik} \) and \( S_{ik} \) are calculated and the known side of the equations (2-10a) and (2-10b) are updated.
(4) The linear set of equations is solved again, using the updated known vector. Note that since the matrix of coefficients remains constant, the triangularization/back substitution process need to be performed on the known vector only.

(5) The new values of voltages obtained as the result of step 4 replace the previous values of voltages. The process then continues with step 3.

The algorithm is very fast and has reliable convergence. It is very fast because of the fact that the matrix of coefficients remains constant throughout the process; it is triangularized only once. The statement with respect to reliability of the convergence is not easy to prove. Indeed, the usual practice for confirming the reliability of a particular algorithm is to test it by using several numerical examples. With respect to the line-loss algorithm, the test results confirm the reliability of convergence. The following explanation is added to indicate why consistently good convergence characteristics can be expected from this algorithm.

The basic equations of the algorithm, equations (2-10a) and (2-10b), represent the following recursion formula:

\[ S + F(X^{(v)}) = A \cdot X^{(v+1)} \]
\[ F(X^{(0)}) = 0 \]  \hspace{1cm} v = 0, 1, \ldots, n \hspace{1cm} (2-11)

where \( S \) is the vector of injected active and reactive powers, \( F(X) \) is the nonlinear vector comprising all the line loss terms and \( \varepsilon \) terms in (2-10), \( A \) is the matrix of coefficients in (2-10) and \( X \) is the vector of unknown voltage magnitudes squared and voltage phase angles. Superscript
(v) indicates the iteration number. As mentioned earlier, for any practical power system, the loss terms and the ε terms are very small. In other words, for practical power systems we have

\[
\frac{||F(X)||}{||S||} \ll 1
\]  

(2-12)

where \( ||\cdot|| \) indicates the norm of a vector.

From (2-11) and (2-12) we can conclude that the condition for (2-11) to converge onto the final solution is that matrix A should be "well-conditioned". Although there is no definition for what is a "well-conditioned" system and what is not, the matrices with dominant diagonal elements are normally considered to be well conditioned. Hence, since the matrix of coefficients, A in (2-11), which is directly related to the nodal admittance matrix, has dominant diagonal terms, it is well conditioned and, therefore, (2-11) converges.

We can also use the condition number, \( \mu \), of A to show that it is a well conditioned matrix. Here again, there is not a set of values of \( \mu \) corresponding to the "well conditioned" matrices and another set corresponding to the "ill conditioned" ones. However, the matrices whose condition numbers are of about the same order as the matrix itself are considered to be well conditioned. The condition number of A was calculated for a few of the test systems and it was found that the above condition was satisfied. The condition numbers for Test Systems 3 (IEEE 14-bus system) and 5 (IEEE 30-bus system) were 10.89 and 24.46, respectively.

Although the condition numbers may be used to calculate an upper bound for the voltage mismatches at the end of each iteration, such bounds would be far greater than the actual mismatches and, therefore, would have no practical significance.
2.3 Programming and Storage Requirements

In matrix form, we have a double-row and a double-column for each P-Q bus; a single-row and a single-column for each P-V bus; and no row and column for slack buses. The "element" \((i,k)\) of the matrix of coefficients would be a submatrix whose dimensions are given in Table 2.1.

<table>
<thead>
<tr>
<th>(k) bus type</th>
<th>P-Q</th>
<th>P-V</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i) bus type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-Q</td>
<td>2 x 2</td>
<td>2 x 1</td>
</tr>
<tr>
<td>P-V</td>
<td>1 x 2</td>
<td>1 x 1</td>
</tr>
</tbody>
</table>

Table 2.1. Dimensions of "Element" \((i,k)\) of the Matrix of Coefficients.

The elements of these submatrices are the elements of the (modified) nodal admittance matrix. Therefore, if there is no branch connecting buses \(i\) and \(k\), the corresponding \((i,k)\) element would be null. It follows that:

1. The matrix of coefficients is highly sparse;
2. The matrix of coefficients has exactly the same structure as the Jacobian matrix of the Newton-Raphson algorithm in polar coordinates;
3. The matrix of coefficients remains constant throughout the algorithm; and
4. Using \(-\delta_i\) and \(\frac{U_i^2}{2}\) as our variables and assuming symmetry for the nodal admittance matrix, the matrix of coefficients would also be symmetrical except in some of its diagonal "elements".
The last proposition can be verified by examining equations (2-10a) and (2-10b): The "element" \((i,k)\) of the matrix of coefficients, when both \(i\) and \(k\) are P-Q buses and \(i \neq k\), would be:

\[
E_{ik} = \begin{bmatrix}
B_{ik} & G_{ik} \\
G_{ik} & -B_{ik}
\end{bmatrix}
\]

(2-13)

which is equal to the transpose of "element" \((k,i)\). It can be easily seen that this is still valid when one or both of the buses are of P-V type. Therefore, we have symmetry as far as the off-diagonal "elements" are concerned.

The diagonal "elements" of P-V buses are scalar \((B_{ii} - b_i)\) and, as such, do not alter the symmetry of the matrix. For P-Q type buses, however, the diagonal "elements" are:

\[
D_{ii} = \begin{bmatrix}
B_{ii} - b_i & G_{ii} + g_i \\
G_{ii} - g_i & -(B_{ii} + b_i)
\end{bmatrix}
\]

(2-14)

which would not be symmetric if \(g_i \neq 0\). It must be mentioned, however, that \(g_i\) is usually equal to zero. Even in cases where \(g_i \neq 0\), it can be accounted for in other ways such that the matrix of coefficients becomes completely symmetrical.

The above features of the matrix of coefficients play an important role in programming the algorithm: Since this matrix is highly sparse, the ordered Gaussian elimination can be effectively utilized to minimize the storage requirements; and since it remains unchanged
throughout the process, the triangularization process need be carried out only once. Furthermore, we can exploit the (almost) symmetry of this matrix and further reduce the storage requirements: It can be easily shown that the upper half of a triangularized symmetrical matrix contains all the information necessary for carrying out forward and back-substitution processes on any R.H.S. vector. In this case, however, due to slight asymmetry, we have to store an additional $n^2$ elements.

Taking advantage of the fact that the matrix of coefficients contains all the information about the nodal admittance matrix, the storage requirements of the algorithm, which are equal to those of the standard Newton-Raphson algorithm, can be further reduced. This results in a considerable increase in computation time but may be desirable for smaller computers.

2.4 The Numerical Results

A load-flow program (LFP(*) was written - mostly in FORTRAN IV language - to compare the performance of the various algorithms. The program, exploiting the sparsity of the matrices, can handle very large systems. It contains the line-loss algorithm (including the variations of this algorithm to be presented in the following chapters), as well as the well known Newton-Raphson and Fast Decoupled (Newton-Raphson) Load-Flow algorithms.

All the algorithms share the same input/output as well as some other routines which are common to all of them. Furthermore, since all the programs were written by the author and all the tests were performed on the same digital computer - UBC's IBM 370/168 - the results are not biased by programming or computer efficiency.

(*) See Appendix A for details.
Several test systems of various sizes and configurations were used to compare the algorithms. These are listed in Table 2.2.

<table>
<thead>
<tr>
<th>System No.</th>
<th>No. of Buses</th>
<th>No. of Lines</th>
<th>No. of Transf.</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>0</td>
<td>[G21]</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>[Y4]</td>
</tr>
<tr>
<td>3*</td>
<td>14</td>
<td>17</td>
<td>3</td>
<td>[R2]</td>
</tr>
<tr>
<td>4</td>
<td>21</td>
<td>29</td>
<td>3</td>
<td>[D1]</td>
</tr>
<tr>
<td>5*</td>
<td>30</td>
<td>37</td>
<td>4</td>
<td>[R2]</td>
</tr>
<tr>
<td>6</td>
<td>33</td>
<td>23</td>
<td>11</td>
<td>[G14]</td>
</tr>
<tr>
<td>7</td>
<td>38</td>
<td>39</td>
<td>9</td>
<td>[G16]</td>
</tr>
<tr>
<td>8*</td>
<td>57</td>
<td>63</td>
<td>17</td>
<td>[R2]</td>
</tr>
<tr>
<td>9</td>
<td>93</td>
<td>99</td>
<td>57</td>
<td>[G13]</td>
</tr>
<tr>
<td>10</td>
<td>138</td>
<td>219</td>
<td>75</td>
<td>[G6]</td>
</tr>
</tbody>
</table>

Table 2.2. Test Systems Used.

* IEEE Test Systems

All the recorded computation times are CPU time spent in the main part of the algorithm; in other words, the time taken for input/output operations, formation of the nodal admittance matrix, dynamic storage allocations, etc., are not included. This ensures that the recorded times can be taken as very good measures of comparing the speed of various algorithms.

Tables 2.3 and 2.4 compare the line-loss algorithm with the well-known Newton-Raphson algorithm and the Fast Decoupled Load-Flow.
<table>
<thead>
<tr>
<th>System</th>
<th>N-R</th>
<th>FDL</th>
<th>line loss alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>3</td>
</tr>
<tr>
<td>3*</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>2</td>
</tr>
<tr>
<td>5*</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>NC</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>2\frac{1}{2}</td>
<td>6</td>
</tr>
<tr>
<td>8*</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>4\frac{1}{2}</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 2.3. Comparison of Number of Iterations.  
(NC: No Convergence)

<table>
<thead>
<tr>
<th>System</th>
<th>N-R</th>
<th>FDL</th>
<th>line loss alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>3*</td>
<td>36</td>
<td>31</td>
<td>32</td>
</tr>
<tr>
<td>4</td>
<td>79</td>
<td>54</td>
<td>49</td>
</tr>
<tr>
<td>5*</td>
<td>112</td>
<td>75</td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td>119</td>
<td>101</td>
<td>NC</td>
</tr>
<tr>
<td>7</td>
<td>162</td>
<td>98</td>
<td>153</td>
</tr>
<tr>
<td>8*</td>
<td>336</td>
<td>224</td>
<td>218</td>
</tr>
<tr>
<td>9</td>
<td>808</td>
<td>563</td>
<td>836</td>
</tr>
<tr>
<td>10</td>
<td>2337</td>
<td>1091</td>
<td>1751</td>
</tr>
</tbody>
</table>

Table 2.4. Comparison of Computation Times.  
(Times in milliseconds; NC: No Convergence)
All the results were obtained by specifying a tolerance of .01 P.U. (1 MW/1 MVAR) on maximum power mismatch. In the case of the Newton-Raphson algorithm and the Fast Decoupled Load-Flow, a flat-start was used. For the line-loss algorithm two different alternatives were tested: (1) setting the initial estimates of line losses to zero and; (2) using a flat-start to calculate the initial values of line losses and ε’s. The two approaches proved to be identical.

It can be seen from Tables 2.3 and 2.4 that, even when the (approximate) values of line losses are not specified initially, the line-loss algorithm performs very satisfactorily. For most systems, the algorithm converges in a few iterations and its overall computation time is less than that of the Newton-Raphson algorithm. Of course, the algorithm performs even better when some initial estimates of line losses are available.

Figures 2.4, 2.5 and 2.6 show the typical voltage magnitude and phase angle variations for various test systems. It is interesting to note the similarity of these patterns to that of an underdamped, critically damped or overdamped second-order system.

Nevertheless, the line-loss algorithm, in this present form, may have a considerable computational round-off error which, in some cases, can delay or even prevent convergence. This can be seen by examining equations (2-10a) and (2-10b). At a point fairly close to the
final solution, the corrected R.H.S. values of equations differ by a very small percentage from their previous values. The resulting corrections to voltage magnitudes and phase angles would also be a small percentage of their previous values. Since the equations are in terms of the voltage magnitudes and phase angles and not in terms of corrections to these values, the roundoff error may completely offset the effect of these corrections. In other words, at some point in the process, the corrections are lost due to numerical errors and it would be impossible to get any closer to the solution point. This is precisely what happens in the case of Test System 5.

The line-loss algorithm, however, can be reformulated to overcome this problem. In the next chapter, first the strictly mathematical derivation of the line-loss algorithm is presented. Then the equations are slightly modified and reformulated in terms of incremental changes to variables. The new formulation does not have the roundoff error problem and as well has several other advantages over the basic line-loss algorithm.
Fig. 2.4. Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 9.
Fig. 2.5. Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 3.
Fig. 2.6. Typical Voltage Magnitude (a) and Phase Angle (b) Variations for Test System 6.
3. THE INCREMENTAL-CHANGE LINE-LOSS ALGORITHM

In previous chapter the line-loss algorithm was derived using circuit theory concepts and current/power analogy. Since the algorithm represents the exact expression of the power flows in the system, we must be able to derive it directly from the power flow equations. It must be mentioned, however, that we are able to do so precisely because we know what we are looking for and why. Without such knowledge, there would have been no reason to "regroup" the terms in this particular way while there are almost endless other possibilities.

In this chapter, first we present the new way of deriving the basic equation of the line-loss algorithm. This derivation enables us to proceed to reformulate the equations in terms of the incremental changes to variables. The modified version, which we call the incremental-change line-loss algorithm, has all the advantages of the basic line-loss algorithm while it does not have the serious drawback of being vulnerable to roundoff errors. It also takes less computation time per iteration and less storage as compared with the basic line-loss algorithm.

3.1 Derivation of the Algorithm

For an n-bus system the total complex power, \( S_i \), injected at any bus, \( i \), is given by:

\[
S_i = V_i \sum_{k=1}^{n} Y_{ik} V_k^* = \sum_{k=1}^{n} Y_{ik}(V_i V_k^*) + Y_{ii} U_i^2
\]  

(3-1)

where \( V_i \) represents the complex voltage at bus \( i \), \( Y_{ik} \) represents the \((i,k)\)th element of the nodal admittance matrix, \( U_i \) is the magnitude of \( V_i \) and \(^*\) indicates the conjugate of a complex number. At the same time,
if \( y_{ik} \) is the value of admittance connecting buses \( i \) and \( k \) in the \( \pi \)-equivalent representation of the branch \((i,k)\), we have the following expression for the complex power losses in that branch:

\[
S^l_{ik} = (V_i - V_k)y^*_{ik} (V_i - V_k)^*
\]

\[
= y^*_{ik}[U_i^2 + U_k^2 - 2 \text{Re}(V_i V_k^*)]
\]

(3-2)

where \( \text{Re}(\ ) \) represents the real part of a complex number. Now, from the fact that the element \((i,k)\) of the nodal admittance matrix is equal to the negative of \( y_{ik} \); i.e.:

\[
Y_{ik} = -y_{ik}
\]

(3-3)

it follows that:

\[
\text{Re}(V_i V_k^*) = \frac{1}{2}[U_i^2 + U_k^2 + \frac{S^l_{ik}}{Y_{ik}}]
\]

(3-4)

We also know that the imaginary part of \( V_i V_k^* \) is:

\[
\text{Im}(V_i V_k^*) = U_i U_k \sin(\delta_i - \delta_k)
\]

(3-5)

which we assume to be

\[
U_i U_k \sin(\delta_i - \delta_k) = \delta_i - \delta_k - \varepsilon_{ik}
\]

(3-6)

where \( \varepsilon_{ik} \) is the correction, or error term. Note that equation (3-6) presents no approximation.

From equations (3-4), (3-5) and (3-6) it follows that:

\[
V_i V_k^* = \frac{1}{2}[U_i^2 + U_k^2 + \frac{S^l_{ik}}{Y_{ik}}] + j(\delta_i - \delta_k - \varepsilon_{ik})
\]

(3-7)

Replacing for \( V_i V_k^* \) in equation (3-1) and rearranging the terms we obtain:
\[
S_i - \sum_{k=1}^{n} \frac{S_{ik}}{2} + j \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} \epsilon_{ik} = \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} \left( \frac{U_k}{2} - j \delta_k \right)
\]

\[
+ \left( \frac{U_i}{2} + j \delta_i \right) \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} + Y_{ii}^{*} \frac{U_i^2}{1}
\]

(3-8)

Considering the fact that:

\[
\sum_{k=1, k \neq i}^{n} Y_{1k}^{*} + Y_{ii}^{*} = y_i
\]

(3-9)

equation (3-8) becomes

\[
S_i - \sum_{k=1}^{n} \frac{S_{ik}}{2} + j \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} \epsilon_{ik} = \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} \left( \frac{U_k}{2} - j \delta_k \right)
\]

\[
+ \left( \frac{U_i}{2} \right) (Y_{ii}^{*} + y_i^{*}) - j \delta_i (Y_{ii}^{*} - y_i^{*})
\]

(3-10)

which is exactly the same as equation (2-9), the basic equation of the line-loss algorithm. The recursion formula, (2-11) in the line-loss algorithm, uses the above equation in the following way:

\[
S_i^{(v)} - \sum_{k=1}^{n} \frac{S_{ik}^{(v)}}{2} + j \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} e_{ik}^{(v)} = \sum_{k=1, k \neq i}^{n} Y_{1k}^{*} \left( \frac{U_k}{2} - j \delta_k^{(v+1)} \right)
\]

\[
+ \left( \frac{U_i^{(v+1)}}{2} \right) (Y_{ii}^{*} + y_i^{*}) - j \delta_i^{(v+1)} (Y_{ii}^{*} - y_i^{*})
\]

(3-11)

where superscripts \((v)\) and \((v+1)\) denote the iteration cycle. At the same time, we know that equation (3-10) was obtained without any approximation and, therefore, it expresses the exact relationship between the
nodal power injections and nodal voltages. It can be written for any set of voltages and their corresponding powers. In other words, if we use $S_i^{(v)}$ to represent the calculated value of injected power at bus $i$ at the end of iteration cycle $v$, we have:

\[
S_i^{(v)} = \sum_{k=1, k \neq i}^{n} \frac{S_{ik}^{(v)}}{2} + j \sum_{k=1, k \neq i}^{n} Y_{ik}^{*} e_{ik}^{(v)} = \sum_{k=1, k \neq i}^{n} Y_{ik}^{*} \left(\frac{U_k^{2(v)}}{2} - j \delta_k^{(v)}\right)
\]

\[
+ \frac{U_i^{2(v)}}{2} (Y_{ii}^{*} + y_i^{*}) - j \delta_1^{(v)} (Y_{ii}^{*} - y_i^{*})
\]

Equation (3-12) does not represent any recursion formula. On the contrary it expresses the exact power-flow equations at the end of iteration cycle $v$. Now, subtracting equations (3-11) and (3-12) we obtain:

\[
\Delta S_i^{(v)} = \sum_{k=1, k \neq i}^{n} Y_{ik}^{*} [\Delta \frac{U_k^{2(v)}}{2} - j \Delta \delta_k^{(v)}] + \Delta \frac{U_i^{2(v)}}{2} + \Delta \frac{U_i^{2(v+1)}}{2} - \Delta \frac{U_i^{2(v)}}{2}
\]

\[
- j \Delta \delta_1^{(v)} (Y_{ii}^{*} - y_i^{*})
\]

where:

\[
\Delta S_i^{(v)} = S_i - S_i^{(v)}
\]

= power mismatch at the end of iteration cycle $v$;

\[
\Delta \frac{U_i^{2(v)}}{2} = \Delta \frac{U_i^{2(v+1)}}{2} - \Delta \frac{U_i^{2(v)}}{2}
\]

and

\[
\Delta \delta_1^{(v)} = \delta_1^{(v+1)} - \delta_1^{(v)}
\]
Note that in the R.H.S. of equations (2-10) and (3-10) we included all the buses connected to bus $i$. However, the constant terms, corresponding to the voltage magnitudes of P-V buses and to the voltage magnitudes and phase angles of slack buses, cancel out in the subtraction and do not appear in the R.H.S. of equation (3-13).

Since equation (3-13) applies to any $(v)$, we drop the superscript $(v)$ from this equation. Then, separating the real and imaginary equations in (3-13), we obtain the final equations used in the incremental-change line-loss algorithm:

$$\Delta P_i = \sum_{k=1}^{n} \frac{U_k^2}{2} (G_{ik} \Delta \delta_k - B_{ik} \Delta \delta_i) + (G_{ii} + g_i) \Delta \frac{U_i^2}{2} - (B_{ii} - b_i) \Delta \delta_i \quad (3-14a)$$

$$\Delta Q_i = \sum_{k=1}^{n} \frac{U_k^2}{2} (-B_{ik} \Delta \delta_k - G_{ik} \Delta \delta_i) - (B_{ii} + b_i) \Delta \frac{U_i^2}{2} - (G_{ii} - g_i) \Delta \delta_i \quad (3-14b)$$

For any P-Q bus, both equations (3-14a) and (3-14b) can be written and two correction terms have to be calculated. For a P-V type bus, only equation (3-14a) can be written and only one correction term has to be calculated. Slack buses have no unknowns and none of the above equations can be written for them.

3.2 The Iterative Process

The set of equations in (3-14), in matrix form, have exactly the same matrix of coefficients as the set of equations in (2-10). The reason for this is that although in the R.H.S. of equation (2-10) the
terms corresponding to any bus connected to i are included, nevertheless, all the U terms corresponding to P-V buses and all the U and δ terms corresponding to slack buses are known. The matrix of coefficients is not affected by these terms and, hence, there is the complete similarity between the matrices of coefficients.

Another point to be mentioned here is that, contrary to the basic line-loss algorithm of equation (2-10), the incremental-change line-loss algorithm cannot be started without having an initial set of estimates for the voltages. Once a set of estimates are available, they can be corrected using equation (3-14). If the approximate values of line losses are available, we can obtain these initial estimates by performing one cycle of the basic line-loss algorithm. Note that, since the matrices of coefficients are exactly identical, this step can be very easily implemented. If no approximate values of losses are initially specified, we can either use a flat-start or perform one cycle of the line-loss algorithm with losses set to zero. Both approaches were found to be equally good.

The iterative process can be summarized as follows:

(1) If some initial estimates for the line losses of the system are available, a cycle of the basic line-loss algorithm is performed to obtain the initial values of voltages. Otherwise, a flat-start is used (*).

(2) Using the present estimates of voltages, the power flows of the system are calculated. If they are within the desired tolerance of their specified values, the problem is solved. Otherwise,

(*) Alternatively, one cycle of the basic line-loss algorithm with initial estimates of line losses set to zero may be performed. LFP permits both options. See Appendix A.
(3) the power mismatches calculated in step 2 are used in equation (3-14) to calculate the corrections to the voltage magnitudes and phase angles. The set of equations is solved by Gaussian elimination and back substitution. Note that since the matrix of coefficients remains constant it need be triangularized only once. At each step of the process, the triangularization/back substitution process is only performed on the vector of power mismatches.

(4) Using the answers obtained as the result of step 3, the present estimates of the voltages are corrected. The new values of voltages, then, would be the new estimates and the process continues with step 2.

The algorithm has a very fast convergence since the matrix of coefficients remains constant and does not have to be formed and triangularized at every step. Furthermore, the algorithm is very reliable. The test results confirm this fact. Furthermore, all the comments made about the reliability of the basic line-loss algorithm apply here as well. In fact, this version of the line-loss algorithm is more reliable since it is formulated in terms of incremental changes to variables and, therefore, is not vulnerable to roundoff errors.

3.3 Programming and Storage Requirements

As mentioned earlier, the matrix of coefficients in this algorithm is exactly the same as that in the basic line-loss algorithm of Chapter 2. Therefore, all the comments about programming and storage requirements of the latter apply here as well; viz, if we use \((-\Delta \delta_1)\) and \(\frac{U_1^2}{2}\) as our variables:
(a) The matrix of coefficients is a highly sparse, constant and almost symmetrical matrix which can be very easily constructed; therefore,

(b) sparsity programming techniques and ordered Gaussian elimination can be effectively used to minimize the storage requirements. The matrix of coefficients is triangularized only once. Only the upper half of the triangularized matrix of coefficients plus an additional \( n_{PQ} \) elements need be stored; and,

(c) if need be, the storage requirements of the algorithm can be reduced to those of the Y-matrix algorithms, at the expense of increased computation time.

The storage requirements of the algorithm are, at most, equal to those of the standard Newton-Raphson algorithm. This statement can be verified by comparing the structure of the matrix of coefficients in this algorithm and the Jacobian matrix in the Newton-Raphson algorithm. The "element" \((i,k)\) in both of them is a submatrix whose dimensions depend on the type of buses \(i\) and \(k\). For the same bus types, these submatrices will have the same dimensions in both cases. Also, in both cases, the slack buses are not included in these matrices. This means that the structure of the matrix of coefficients in (3-14) (and also in (2-10)) is exactly the same as that of the Jacobian matrix in the Newton-Raphson algorithm with polar coordinates.

However, the Newton-Raphson algorithm does not store the entire Jacobian matrix, but only its triangularized half. In other words, the Jacobian matrix is formed and triangularized, row by row, and the triangularized Jacobian is the only thing that is stored. Exactly the-
same procedure can be followed in case of the incremental-change line-loss algorithm. In the Newton-Raphson algorithm this is possible because the Jacobian has to be formed at every step of the iterative process and, hence, no information other than the upper half of the triangularized matrix is required. In the incremental-change line-loss algorithm this is possible because the matrix of coefficients is symmetrical. Thanks to this symmetry, only the upper half of the triangularized matrix of coefficients would contain sufficient information to carry out triangularization and back substitution processes on any right hand side vector. However, we need to store all the diagonal elements, as well as \( n_{PQ} \) additional elements if the matrix is not completely symmetrical. These elements can be stored in place of the active and reactive power mismatches in the Newton-Raphson algorithm.

The storage scheme used (in LFP) for calculating and storing the elements of the matrix of coefficients can be summarized as follows:

All the triangularized elements of the matrix of coefficients, along with proper pointers, are stored in a vector, from now on referred to as the vector of the triangularized elements. In another vector the pointers to the starting location of each row are stored. These pointers are positive for P-Q rows (double rows) and negative for P-V rows (single rows). The elements of the triangularized row are stored in locations starting with the one specified by the pointer for that row and ending with the one specified by the next pointer. The first one or three locations, depending upon whether the row is P-V or P-Q, are used for storing the diagonal elements. This is followed by an integer pointer indicating the column number of an "element" in the row. This pointer, too, is positive for P-Q columns (double columns) and negative for P-V
columns (single columns). The number and the order of the elements that follow this pointer are dependent upon, and determined by the sign of both row and column pointers. In the location following the last item for that particular column (which may be one, two or four locations away) the column pointer for the next element in the row is stored, followed by the value of its elements. And so it continues.

Each row is formed and triangularized in a vector, from now on referred to as the working-row, prior to being stored in the vector of the triangularized elements. Each element of the working-row consists of four locations in which the four possible elements of the submatrix (i,k) are stored. It also has a column pointer which follows the same rule as any other pointer with respect to P-Q and P-V elements. The length of the working-row is sufficient to store a full row of the matrix of coefficients. Once the row is formed, a linear combination of the previously stored rows is added to it such that it will not have any elements in the column range that has been processed before. During this process, new elements may be created and some of the elements may be deleted and modified. New elements are added to the end of the working-row and deleted elements are indicated by a zero column pointer. At the end of the process, only the elements with non-zero column pointers will be stored in the compact vector of the triangularized elements.

It should be mentioned that the storage requirements of the basic line-loss algorithm is roughly the same as those of the incremental-change line-loss algorithm. The former requires slightly more space for storing additional terms corresponding to P-V and slack buses and other information necessary for updating the known vector.
3.4 The Numerical Results

The incremental-change line-loss algorithm was programmed using sparse matrix techniques and tested upon the test systems listed in Table 2.2. Tables 3.1 and 3.2 compare the required number of iterations and the CPU time taken by this algorithm with those of the Newton-Raphson algorithm and the Fast Decoupled (Newton-Raphson) Load-Flow. The number of iterations and the CPU times taken by the basic line-loss algorithm are also included for comparison. All the results were obtained by specifying a tolerance of .01 P.U. (1 MW/ MVAR) on maximum power mismatch. Furthermore, for the Newton-Raphson algorithm and the Fast Decoupled (Newton-Raphson) Load-Flow, a flat-start was used while, the incremental-change line-loss algorithm was started by using equations (2-10a) and (2-10b) during the first iteration and switched to equations (3-14a) and (3-14b) thereafter. The correction terms and the line-losses were assumed to be initially zero.

It is evident from Tables 3.1 and 3.2 that the incremental-change line-loss algorithm has all the advantages of the basic line-loss algorithm while, at the same time, it is not vulnerable to computational roundoff error. In case of Test System 6, where the original line-loss algorithm failed to converge, the new version converges in 5 iterations. Table 3.2 also shows that the incremental-change line-loss algorithm is faster than the basic line-loss algorithm, because the line losses and correction terms need not be computed at each iteration. Even without any approximate values of line losses specified, the incremental-change line-loss algorithm converges considerably faster than Newton-Raphson algorithm and, in most cases, even faster than the Fast Decoupled (Newton-Raphson) Load-Flow.
<table>
<thead>
<tr>
<th>System</th>
<th>N-R</th>
<th>FDL</th>
<th>line loss alg.</th>
<th>Inc. change line loss alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3*</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2\frac{1}{2}</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>5*</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>NC</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>2\frac{1}{2}</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>8*</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>6</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>4\frac{1}{2}</td>
<td>13</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 3.1. Comparison of Number of Iterations.  
(NC: No Convergence)

<table>
<thead>
<tr>
<th>System</th>
<th>N-R</th>
<th>FDL</th>
<th>line loss alg.</th>
<th>Inc. change line loss alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>12</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>3*</td>
<td>36</td>
<td>31</td>
<td>32</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>79</td>
<td>54</td>
<td>49</td>
<td>42</td>
</tr>
<tr>
<td>5*</td>
<td>112</td>
<td>75</td>
<td>70</td>
<td>61</td>
</tr>
<tr>
<td>6</td>
<td>119</td>
<td>101</td>
<td>NC</td>
<td>80</td>
</tr>
<tr>
<td>7</td>
<td>162</td>
<td>98</td>
<td>153</td>
<td>113</td>
</tr>
<tr>
<td>8*</td>
<td>336</td>
<td>224</td>
<td>218</td>
<td>188</td>
</tr>
<tr>
<td>9</td>
<td>808</td>
<td>563</td>
<td>836</td>
<td>616</td>
</tr>
<tr>
<td>10</td>
<td>2337</td>
<td>1091</td>
<td>1751</td>
<td>1247</td>
</tr>
</tbody>
</table>

Table 3.2. Comparison of Computation Times.  
(Times in milliseconds; NC: No Convergence)
The algorithm always converges in a few iterations, except for Test Systems 9 and 10, where the number of iterations required are 12 and 13, respectively. This was found to be caused by very large phase differences (up to about 70°) across some of the branches of these systems. Such large phase differences do not occur in practice. However, even in these cases, the computation time is significantly reduced as compared with the Newton-Raphson algorithm. It is interesting to note that the pattern of voltage magnitude and phase angle variations for these systems - as shown in Figures 3.1 and 3.2(*) - resemble the pattern of an under-damped second-order system.

Several other modifications were tried to see if the performance of this algorithm, especially with respect to the number of iterations taken for Test Systems 9 and 10, can be further improved. Some of these actually did reduce the number of iterations considerably. Unfortunately, however, these modifications were those that required changing the matrix of coefficients at every step and, consequently, also increased the total computation time considerably. Nevertheless, their improved performance with respect to the number of iterations makes them attractive alternatives. They are also of special interest to those who may want to pursue this research work further. A few of these alternatives which showed the greatest improvement are briefly mentioned below:

One such alternative was to use another equation instead of (3-6), which would provide us with better values at the end of first iteration. Since we know that:

(*) For all the plots a flat-start was used. If one cycle of the line-loss algorithm with zero initial estimates was performed, the plots would start at the points corresponding to iteration 1.
Fig. 3.1. Typical Voltage Magnitude and Phase-Angle Variations for Test System 9.
(Incremental-Change Line-Loss Algorithm with a Flat-Start)
Fig. 3.2. Typical Voltage Magnitude (a) and Phase-Angle Variations for Test System 10.
(Incremental-Change Line-Loss Algorithm with a Flat-Start)
we can use the following equation instead of (3-6):

\[ U_k U_k \sin(\delta_i - \delta_k) = (\delta_i - \delta_k) \left[ \frac{2}{3} + \frac{1}{3} \cos(\delta_i - \delta_k) \right] - \text{correction term} \]  

Furthermore, we can replace for \( \cos(\delta_i - \delta_k) \) an expression of the line losses. Assuming that the magnitudes of voltages are almost 1 P.U., we obtain:

\[ \frac{S_{ik}}{2Y_{ik}} \sim \cos(\delta_i - \delta_k) - 1 \]  

Therefore, (3-16) becomes

\[ U_k U_k \sin(\delta_i - \delta_k) = (\delta_i - \delta_k) \left[ 1 + \frac{S_{ik}^l}{6Y_{ik}^*} \right] - \varepsilon_{ik} \]  

where \( \varepsilon_{ik} \) represents the new correction term which is different from (and much smaller than) the correction term used in (3-6). Using (3-18) instead of (3-6), and following the same procedure as explained in section 3.1, we get:

\[ S_i - \sum_{k=1}^{n} \frac{S_{ik}^l}{2} + j \sum_{k=1}^{n} \frac{Y_{ik}^* \varepsilon_{ik}}{k \neq i} = \sum_{k=1}^{n} \frac{U_k^2}{2} Y_{ik}^* - j \sum_{k=1}^{n} \delta_k (Y_{ik}^* + \frac{S_{ik}^l}{6}) \]

\[ + \frac{1}{2} \sum_{k=1}^{n} \frac{S_{ik}^l}{Y_{1i}^* + Y_i^*} - j \sum_{k=1}^{n} \delta_k (Y_{1i}^* - Y_i^* - \frac{S_{ik}^l}{6}) \]  

which should replace the basic equation (3-10). It can be seen that the matrix of coefficients is no longer constant. Using the above equation
reduced the number of iterations taken for Test System 9 by one third. On the other hand, the total computation time was much more than that of the incremental-change line-loss algorithm. When the variable terms in the matrix of coefficients were transferred to the other side of the equations and evaluated on the basis of the present values of voltages the performance of the algorithm became worse than that of the incremental-change line-loss algorithm.

Another alternative which showed a great reduction in the number of iterations was to leave the sine terms in (3-6) intact. The derivation of the incremental-change line-loss algorithm, would then include the difference between two sine terms. In other words, \( \Delta(\sin(\delta)) \) will appear in equation (3-13) instead of \( \Delta \delta \). However, assuming that the change in \( \delta \) from one step to the next is small\(^(*)\), we can write:

\[
\sin[(\delta_i - \delta_k) + \Delta(\delta_i - \delta_k)] \approx \sin(\delta_i - \delta_k) + \Delta(\delta_i - \delta_k)\cos(\delta_i - \delta_k)
\]

(3-20)

Here again, due to the cosine term in the right-hand side of (3-20), we will get a matrix of coefficients which changes from one iteration step to the next. Using this approach, the number of iterations taken for Test System 9, was reduced to 6. The computation time, on the other hand, increased threefold. Once again, taking the variable terms in the matrix of coefficients to the other side of the equations, and evaluating them on the basis of the present values of the voltages resulted in a worse performance.

\(^(*)\) This assumption is indeed valid. Test results showed that after one cycle of the basic line-loss algorithm, even when the initial estimates were set to zero, the calculated voltages were within less than 10% of their final values.
Yet another approach was to use the following expression, instead of equation (2-5), for the voltages of the (S-U^2) system:

\[ e_i = \frac{1}{2} U_i^2 - j \Delta_i \]  

(3-21)

such that:

\[ \Delta_i - \Delta_k = U_i U_k \sin(\delta_i - \delta_k) \]  

(3-22)

for all \( i \) and \( k \). The problem with this approach is that the phase angle for each bus can be calculated from several different routes; and the values do not necessarily agree with each other. The approach would be unsuccessful if the first route that becomes available is chosen and the phase angle is calculated from that route. On the other hand, when all the possible values for each bus are calculated and a least squares hyper-plane is fitted to these values the algorithm improves tremendously (with respect to the number of iterations). This latter approach, however, involves finding the parameters of the least squares hyper-plane, which requires the direct solution of another set of equations. Although the matrix of coefficients for the new set of equations is the same as the system's incidence matrix, nevertheless, the increased storage and computation time requirements offset the gains achieved in reducing the number of iterations from 12 to 6 (in case of Test System 9).

Many other alternatives were tried but proved to be unsuccessful. In particular, the use of various acceleration factors, including the ones that are related to power mismatches and vary from one iteration to the next, did not improve the performance of the algorithm. Except for decoupling, which improved the overall results, the best algorithm remained the one derived in section 3.1 - the incremental-change line-loss algorithm. Figures 3.3, 3.4 and 3.5 compare the performance of this
Fig. 3.3. Comparison of the Algebraic Sum of all the Power Mismatches.

(Test System 8)
Fig. 3.4. Comparison of the Maximum Power Mismatches.

(Test System 8)
Fig. 3.5. Comparison of Sum of all the Absolute Values of
Power Mismatches. (Test System 8)
algorithm, and that of the basic line loss algorithm, with the performance of the Newton-Raphson algorithm. Note that, for the sake of clarity, the three curves are displaced by one iteration step from one another. The figures show that the line-loss algorithms have a much greater rate of convergence during the first iterations than during the later iterations. This is due to the constant-gradient characteristic of these algorithms.

The overall performance of the line-loss algorithms was greatly improved when the weak (P-U) and (Q-δ) interdependencies were exploited and the incremental-change line-loss algorithm was decoupled. The decoupled version, as we will see in the next chapter, has a remarkable resemblance to the Fast Decoupled (Newton-Raphson) Load-Flow.
4. FAST DECOUPLED LINE-LOSS ALGORITHMS

In a practical power system, the interdependence between active powers and voltage magnitudes, and between reactive powers and phase angles are weak and may be neglected. This is usually referred to as "the decoupling principle". In this chapter, we exploit this feature of practical power systems and decouple the incremental-change line-loss algorithm. It will be seen that, contrary to the Newton-Raphson algorithm, applying the decoupling principle to the incremental-change line-loss algorithm will result in a stable decoupled algorithm.

Two slightly different decoupled algorithms will be presented and compared with the Fast Decoupled (Newton-Raphson) Load-Flow. Both these algorithms are very fast and reliable. Test results indicate that the performance of these decoupled versions is even better than that of the undecoupled incremental-change line-loss algorithm. Indeed, the decoupled formulations are nothing but approximations to the original formulation. The reason for their improved performance seems to be that, in the decoupled versions, the corrections are calculated and applied at the end of every "half-iteration" rather than every iteration.

4.1 Derivation of the Algorithm

The incremental-change line-loss algorithm uses equations (3-14a) and (3-14b), rewritten below,

\[ \Delta P_i = \sum_{k \neq 1}^{n} (G_{ik} \Delta \frac{U_k^2}{2} - B_{ik} \Delta \delta_k) + (G_{ii} + g_i) \Delta \frac{U_i^2}{2} - (B_{ii} - b_i) \Delta \delta_i \]  

(3-14a)
\[ \Delta Q_i = \sum_{k=1}^{n} (-B_{ik} \Delta \frac{U_k^2}{2} - G_{ik} \Delta \delta_k) - (B_{ii} + b_i) \Delta \frac{U_i^2}{2} - (G_{ii} - g_i) \Delta \delta_i \quad (3-14b) \]

which, in matrix form, can be written as:

\[
\begin{bmatrix}
\Delta P \\
\Delta Q
\end{bmatrix} =
\begin{bmatrix}
-B' & G' \\
-G'' & -B''
\end{bmatrix}
\begin{bmatrix}
\Delta \delta
\end{bmatrix}
\]

\[ (4-1) \]

(a) Algorithm A

Assuming that submatrices \([G']\) and \([G'']\) in the above equation are null, we get:

\[
\begin{bmatrix}
\Delta P \\
\Delta Q
\end{bmatrix} =
\begin{bmatrix}
-B'
\end{bmatrix}
\begin{bmatrix}
\Delta \delta
\end{bmatrix}
\]

\[ (4-2a) \]

\[
\begin{bmatrix}
\Delta P \\
\Delta Q
\end{bmatrix} =
\begin{bmatrix}
-B''
\end{bmatrix}
\begin{bmatrix}
\Delta \frac{U_i^2}{2}
\end{bmatrix}
\]

\[ (4-2b) \]

This follows the decoupling technique commonly used [D11]; namely, that of neglecting P-U and Q-δ interdependencies altogether.

From equations (3-14a) and (3-14b), it is seen that the submatrices \([B']\) and \([B'']\) are identical with the nodal susceptance matrix entries, except for the main diagonal elements.

(b) Algorithm B

In this algorithm we also neglect the effect of the series resistances in calculating the elements of \([B']\). The equations used are still (4-2a) and (4-2b) but the elements of \([B']\) are different:
In other words, the DC load-flow algorithm [D1] is used instead of equation (4-2a) for calculating the voltage phase angles. Equation (4-2b) remains unchanged. This was found, experimentally, to slightly improve the results. Stott and Alsac experienced the same results when they applied the above approximation to the decoupled Newton-Raphson algorithm [D11].

It is very interesting to compare the equations (4-2a) and (4-2b), which are the basic equations for the decoupled line-loss algorithms, with equations (1-16a) and (1-16b), which represent the Fast Decoupled (Newton-Raphson) Load-Flow. The two sets of equations are remarkably similar. Indeed, the matrices of coefficients in equations (4-2a) and (1-16a) are exactly identical. Equation (4-2b) is only slightly different from equation (1-16b); but if the approximation

\[ \Delta U^2 \frac{\Delta U}{2} \]

is made, then the two will become almost identical. It must be mentioned, however, that equations (1-16a) and (1-16b) were obtained by making a number of practical assumptions after applying the decoupling principle while, equations (4-2a) and (4-2b) are obtained simply by neglecting P-U and Q-\( \delta \) couplings. Between the two decoupled line-loss algorithms, algorithm B is more similar to the Fast Decoupled (Newton-Raphson) Load-Flow.
4.2 The Iterative Process

Like the incremental-change line-loss algorithm, the decoupled algorithms A and B need a set of initial estimates for the voltages of the system. However, unlike the former algorithm, the matrices of coefficients in the latter are not the same as that of the basic line-loss algorithm. Therefore, although it is possible to start these algorithms by first performing one cycle of the basic line-loss algorithm, this would require forming and solving a different set of equations during the first iteration step. This is a very time consuming process. For this reason, the decoupled line-loss algorithms are always started using a flat-start.

The solution of either (4-2a) or (4-2b) is considered to be one "half iteration". Prior to solving (4-2a) or (4-2b), the active or reactive power mismatches are checked against the desired tolerance. If all these mismatches are within the desired limits, the solution of the corresponding set of equations is skipped. Hence, depending upon the convergence characteristics of different systems, one set of equations may have to be solved more often than the other. The algorithm always starts by trying to solve (4-2a), because this was found (experimentally) to be always preferable. The flow-chart of Figure 4.1 summarizes the iterative process for decoupled algorithm A and B. Note that since the matrices of coefficients, [B'] and [B''], are symmetrical and constant, they need to be triangularized only once. As before, the solution of (3-2a) or (3-2b) is always obtained by using the information in the upper triangularized half of these matrices.
Fig. 4.1. Flow-Chart Indicating the Logic of Decoupled Algorithms.
4.3 Programming and Storage Requirements

Both $[B']$ and $[B'' \]$, in equations (4-2a) and (4-2b), are constant, symmetrical and highly sparse matrices. Due to the sparsity of these matrices, the decoupled algorithms A and B can benefit a great deal from ordered Gaussian elimination and sparsity programming techniques. The storage requirements of these algorithms, then, would be much less than those of the Newton-Raphson algorithm and exactly the same as those of the Fast Decoupled (Newton-Raphson) Load-Flow. Furthermore, if storage is a very serious limitation, the storage requirements of these algorithms can be substantially reduced, of course, at the expense of increased computation time per iteration.

Since both matrices are symmetrical and constant, they need to be triangularized only once and only the upper half of their triangularized results need be stored. The ordering of the rows is exactly the same for $[B']$ and for the Jacobian matrix in the Newton-Raphson algorithm and the matrix of coefficients in the incremental-change line-loss algorithm. The same ordering is assumed for the rows of $[B'\]$. With such ordering of the rows, however, the decoupled algorithms require just slightly over half the computer storage needed by the other algorithms mentioned above.

The storage scheme used for these algorithms, which is the same as the one used\(^{(*)}\) for programming the Fast Decoupled (Newton-Raphson) Load-Flow, is very similar to that of the incremental-change line-loss algorithm, as explained in section 3.3. Both $[B']$ and $[B'\]$

\(^{(*)}\) This is the storage scheme used in LFP. Stott and Alsac did not mention what storage scheme they used in [D11].
are formed and triangularized, row by row, and stored in compact form in two separate vectors. There are two vectors of pointers, indicating the starting locations of each row in \([B']\) and \([B'']\). The compact forms of \([B']\) and \([B'']\) consist of integer column pointers followed by respective entries of the triangularized matrices. The scheme is considerably simpler than that used for the incremental-change line-loss algorithm (and also for the Newton-Raphson algorithm) since there is no necessity of distinguishing between P-Q and P-V pointers. As was the case with the incremental-change line-loss algorithm, however, each row is formed in a working-row, combined with the previously processed rows and triangularized, and then stored in compact form. Here again, only the entries in the upper half of the triangularized matrices (plus the diagonal elements) are stored. This information is sufficient for carrying out triangularization/back substitution process on any right-hand side vector.

4.4 The Numerical Results

The results of comparison between decoupled algorithms A and B, and other algorithms are shown in Tables 4.1 and 4.2. A flat-start and a tolerance of .01 P.U. (1 MW/1 MVAR) on maximum power mismatch was specified for all the runs. The test systems are those listed in Table 2.2. The rate of convergence of these algorithms are compared with that of the Fast Decoupled (Newton-Raphson) algorithm in Figures 4.1, 4.2 and 4.3.

Since \([B']\) and \([B'']\) are of smaller dimensions than the matrix of coefficients of the incremental-change line-loss load-flow algorithm, a reduction in computation time per iteration is expected as a result of
decoupling. However, Table 4.2 shows that this is not the case. The
reason for this is that in the decoupled algorithms almost all the
search operations are duplicated. This cannot be avoided if the flexibili-
ity of performing each half iteration separately is to be preserved.
The only way to eliminate this problem would be to solve (4-2a) and
(4-2b) simultaneously which, however, greatly degrades the performance
of the decoupled algorithms.

Tables 4.1 and 4.2 indicate that the performance of the de-
coupled line-loss algorithms is comparable to that of the Fast Decoupled
(Newton-Raphson) Load-Flow. This was to be expected since there is a
remarkable similarity between the equations used by the two algorithms.
However, in order to better compare the reliability of the decoupled line-
loss algorithms A and B with that of the Fast Decoupled (Newton-Raphson)
Load-Flow, it was decided to check their behaviour with respect to
heavily overloaded systems. Test System 5 (IEEE 30-bus system) was
overloaded, step by step, and the load-flow problem was solved for each
loading level using various algorithms. The Newton-Raphson algorithm,
the basic line-loss algorithm and the incremental-change line-loss
algorithm all diverged for the same loading level. This breaking point
was when all the power injections (both loads and generations) were 1.8
times greater than their original values.

On the other hand, all the decoupled algorithms converged for
this loading level: the Fast Decoupled (Newton-Raphson) Load Flow
converged in 27 half-iterations (216 milliseconds); algorithm A converged
in 16 half iterations (164 milliseconds); and algorithm B converged in
20 half iterations (187 milliseconds). The differences in computation
times and the number of iterations are noticeably in favour of the
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1.5</td>
<td>1.5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2.5</td>
<td>3</td>
<td>3</td>
<td>2.5</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td>3*</td>
<td>2</td>
<td>2.5</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2.5</td>
<td>2</td>
<td>2</td>
<td>3.5</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td>5*</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>NC</td>
<td>5</td>
<td>3.5</td>
<td>4.5</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>2.5</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>8*</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3.5</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>6</td>
<td>12</td>
<td>12</td>
<td>4.5</td>
<td>4.5</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>4.5</td>
<td>13</td>
<td>13</td>
<td>6.5</td>
<td>4.5</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1. Comparison of Number of Iterations (NC: No Convergence).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>12</td>
<td>13</td>
<td>10</td>
<td>12</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>3*</td>
<td>36</td>
<td>31</td>
<td>32</td>
<td>25</td>
<td>57</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>79</td>
<td>54</td>
<td>49</td>
<td>42</td>
<td>63</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>5*</td>
<td>112</td>
<td>75</td>
<td>70</td>
<td>61</td>
<td>87</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>119</td>
<td>101</td>
<td>NC</td>
<td>80</td>
<td>94</td>
<td>107</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>162</td>
<td>98</td>
<td>153</td>
<td>113</td>
<td>108</td>
<td>108</td>
<td></td>
</tr>
<tr>
<td>8*</td>
<td>336</td>
<td>224</td>
<td>218</td>
<td>188</td>
<td>235</td>
<td>210</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>808</td>
<td>563</td>
<td>836</td>
<td>616</td>
<td>505</td>
<td>495</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2337</td>
<td>1091</td>
<td>1751</td>
<td>1247</td>
<td>1263</td>
<td>1063</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2. Comparison of Computation Times.  
(Times in milliseconds)  
(NC: No Convergence)
Fig. 4.2. Comparison of the Algebraic Sum of all the Power Mismatches.

(Test System 8)
Fig. 4.3. Comparison of the Maximum Power Mismatches.

(Test System 8)
Fig. 4.4. Comparison of Sum of all the Absolute Values of Power Mismatches. (Test System 8)
decoupled line-loss algorithms, especially algorithm A, which has less similarity with the Fast Decoupled (Newton-Raphson) Load-Flow.

When the system was slightly (one per cent) more overloaded, the Fast Decoupled (Newton-Raphson) Load-Flow diverged while both decoupled line-loss algorithms converged, each taking 20 half iterations. As the overloading continued, in small one per cent steps, both decoupled algorithms converged for three more steps until, finally, algorithm B diverged. For this loading level, algorithm A was the only algorithm that solved the problem. After this point, none of the load-flow algorithms converged.

The above test indicates that while the decoupled line-loss algorithms have the same performance as the Fast Decoupled (Newton-Raphson) Load-Flow, they are more reliable. The assumption made in deriving algorithm B, slightly improves the performance of this algorithm while, at the same time, makes it slightly less reliable.
5. CONCLUSIONS

In a power system, the line losses can be used to derive the constant gradient, fast converging line-loss load-flow algorithm. The algorithm iteratively solves a set of linear equations in terms of voltage magnitudes squared and phase angles, and converges onto the final solution in a few iterations. Although the performance of the algorithm improves if the (approximate) values of line-losses are initially known, such knowledge is not a pre-requisite to using the algorithm. For practical accuracies, even when the initial estimates of line-losses are assumed zero, the line-loss algorithm converges considerably faster than the standard Newton-Raphson algorithm.

A slightly modified version of the line-loss algorithm, written in terms of the incremental changes to the voltage magnitudes squared and phase angles, performs even better than the original version. For all the test systems used, it converges considerably faster than the Newton-Raphson algorithm - in some cases it is almost twice as fast.

The decoupled versions of the incremental change line-loss algorithm have convergence characteristics similar to those of the Fast Decoupled (Newton-Raphson) Load-Flow [D11]. As compared with the undecoupled version of this algorithm, they show definite improvement with respect to the required number of iterations; but not with respect to computation time. In the decoupled algorithms, the corrections are calculated and applied at the end of every half iteration; resulting in improved overall performance and the increased computation time per iteration.
All the line-loss algorithms have very reliable convergence. In particular, the results of using these algorithms for solving the load-flow problem on heavily overloaded systems showed that the convergence of the line-loss algorithms is at least as reliable as that of the Newton-Raphson algorithm. The results also showed that the decoupled line loss algorithms are more reliable than the Newton-Raphson algorithm and the Fast Decoupled (Newton-Raphson) Load-Flow.

The storage requirements of the line-loss algorithms are, at most, equal to those of the standard Newton-Raphson algorithm. The decoupled versions of the line-loss algorithm have the same storage requirements as the Fast Decoupled (Newton-Raphson) Load-Flow. However, if storage is a serious limitation, in both cases, the storage requirements can be reduced to those of the Y-matrix algorithms.

The matrix of coefficients in the line-loss algorithms is very easy to construct and is directly related to the system nodal admittance matrix. Hence, slight changes in the system configuration can be easily reflected in the algorithm. Furthermore, approximate load-flow solutions can be obtained after performing only one iteration of the line-loss algorithm. These characteristics make the line-loss algorithms particularly attractive for study of system outages, where numerous load-flow problems, only slightly different from one another, have to be solved and only approximate answers are required.
BIBLIOGRAPHY

General:


Load-Flow Review Papers:


Convergence & Uniqueness:


Y-matrix Algorithms:


Z-matrix Algorithms:


Newton-Raphson Algorithm:


Decoupled Load-Flow Algorithms:


Diakoptics (Tearing):


Other Load Flow Algorithms:


APPENDIX A

USER'S MANUAL FOR LFP

A.1 Introduction

LFP stands for the "Load Flow Package". It refers to a collection of computer programs written for the purpose of comparing the performance of various load-flow algorithms. At the present time, only the Newton-Raphson algorithm, the Fast Decoupled (Newton-Raphson) Load-Flow, and the different versions of the line-loss algorithm described in this thesis, are available. However, the structure of the package is such that any other algorithm can be added to it without difficulty. Also, LFP has fairly extensive monitoring features which make it particularly attractive for research work.

The package consists of three distinct modules, which operate in succession to one another in the following order:

1. The input module; this module reads the input data, rearranges this data according to an internal format while saving the external bus numbers and other information not required by the load-flow algorithm, provides a listing of input data if necessary, forms the compact nodal admittance matrix, initializes the proper vectors, sets proper pointers to the load-flow algorithm requested by the user, etc., etc. The output from this module is ready to be processed by the proper load-flow algorithm.

2. The process module; this module performs the proper iterative process on the data provided by the input module. It also prints the values of variables at the end of various iteration steps if such information is required. The output from this
module is the final values of voltages along with some other information such as the number of iterations and the computation time taken by that particular algorithm.

(3) The output module; this module prints and/or plots the output information.

The same input and output modules are shared by all the algorithms while each algorithm has its own process module.

LFP was written for use on UBC's IBM 370/168 digital computer. Except for two small routines in IBM 370 assembly language, the programs are written in FORTRAN IV. They make use of several MTS features as well as some of the programs provided by UBC Computing Centre. All the programs use single precision arithmetic. Dynamic storage allocation and sparsity programming techniques are employed to ensure efficient use of storage. With these features, LFP is capable of handling very large power systems - systems of the order of few thousand buses and transmission lines.

The main purpose of this Appendix is to explain the data preparation and use of the LFP. When necessary, comments are also made about the programming details and the structure of the package.

A.2 How to Use

The object module is in the file LFP. The prefix & is used to specify the package (especially useful in terminal mode). The program may be used by issuing the following command:

$$R \text{ LFP SCARDS = ... SPRINT = ... PAR = (Parameter list as) defined below}$$

SCARDS may be assigned to the file or device from which the load-flow data is to be read and SPRINT to the unit on which the output should
appear. In terminal mode, the communication with the user is done via GUSER and SERCOM.

The PAR field contains the information about the load-flow algorithm to be used, the starting point and the data file. This information is provided using the following Keyword parameters. (Only the first letter of each Keyword would suffice):

1. **ALGORITHM** = name of the load-flow algorithm to be used. Only one name must be specified and it must be one of those specified in section A.3, or names of other algorithms which may be added to the package at a later date. There is no default for this Keyword. Not specifying it either will terminate the job (batch mode) or will result in an error message followed by a request for that information (terminal mode).

2. **DATA** = name of the file or device that contains the load-flow data. The default for this Keyword is *SOURCE*. The format of the data must be as explained in section A.4.

3. **START** = F or N. The default value is F, which indicates a flat-start or zero initial estimates for the losses, depending upon the load-flow algorithm being used. **START** = N indicates a non-flat starting point. The initial estimates of losses or voltages must be provided at a later point in the program. See section A.5 for more details.

Except for the name of the algorithm, these Keywords are optional. Their order is not important and they may be separated by commas and/or blanks. If a Keyword is specified several times the last value will be used. Note that using **SCARDS = ...** or **DATA = ...** Keywords, gives identical results.
If no PAR field is specified, either the job would be terminated (batch mode) or the user is requested to provide a PAR field (terminal mode). The information in this field is printed as part of the heading on every page of the output (unless other heading information is provided; see section A.4). If an error is detected in the PAR field, either the job is terminated (batch) or the user is requested to respecify the PAR field (terminal).

A.3 The Algorithms

At the present time, only the algorithms listed in this section may be specified. Other load-flow algorithms may be programmed and added to LFP, in which case the name of their respective entry points (SUBROUTINE or ENTRY) are used to specify the algorithms. The only restriction (apart from those imposed by the programming language) is that these names should not start with the letter "L". Names which start with L are considered as belonging to the line-loss algorithms. For these algorithms, START = N means that the initial estimates of line losses (not voltages) are specified.

The algorithms that may be specified are one of the following:

NEWTON: The Newton-Raphson algorithm.
LOSS1: The basic line-loss algorithm. The algorithms starts by setting the line losses either to zero or to their initial estimates, depending upon whether START = F or N:
FLOSS1: The basic line loss algorithm starting from a flat-start or from another point specified by the user, depending upon whether START = F or N. This is another entry point in LOSS1.
LSS1M: The incremental-change line-loss algorithm. Starts with one cycle of the basic line-loss algorithm in which the losses are initially set to zero or to their estimated values, depending upon whether \( \text{START} = F \) or \( N \).

FLSS1M: The incremental-change line-loss algorithm, starting either from a flat-start or from another starting point specified by the user. This is another entry point in LSS1M.

DECPL1: The decoupled line-loss algorithm A.

DECPL2: The decoupled line-loss algorithm B. This is another entry point in DECPL1.

Apart from the above algorithms, the following names may also be specified:

TEST: Used only for testing the input module. Prints out the information processed by this module.

FLOWS: This is an entry point in LFP which may be specified if the power flows corresponding to a set of voltages are to be calculated. In that case, the \( \text{START} = N \) must be specified.

A.4 The Data Preparation

Each data file contains several different kinds of data records. Some records contain information about different elements in the system; some contain the general information controlling the number of iterations, the length and the type of the output, etc; and some contain comments or the heading information which is to be printed on the top of every output page. Different records are distinguished by means of the character in their first column. For example, a record starting with letter B contains the information about one of the system's buses while a record starting with letter H contains the heading information.
Because of this, the input records may be arranged in any arbitrary order. The following is a brief description and format of each data record.

A.4.1 Bus Information Records:

Letter in the first column is "B".

Each record contains the information about one particular bus in the system. This information (and its format, in parenthesis) is as follows:

BUS TYPE: A one digit integer number in column 2; (I). 1 is used for P-Q buses; 2 is used for P-V buses; 3 is used for the slack buses.

BUS NUMBER: A five digit integer number in columns 3 to 7; (I5). This can be any arbitrary number. Used in communications with the user.

BUS NAME: An eight character alphanumerical string in columns 9 to 16; (A8). This can be any arbitrary combination of eight characters. Used as the bus name in the output printout.

BUS GENERATION: A complex number in columns 19 to 32; (2F7.2). Specifies the total active and reactive power generation of the bus in MW. Positive signs are used for generation.

BUS LOAD: A complex number in columns 33 to 48; (2F8.2). Specifies the total active and reactive bus load in MW. Positive signs are used for load.

GENERATOR Q LIMITS: Two real numbers in columns 49 to 63. These are the lower and upper limits for the reactive power generation of P-V buses in MW. At present, LFP does not use this information.
BASE VOLTAGE: A real number in columns 64 to 70; (F7.2). Specifies the base voltage of the bus in KV.

VOLTAGE MAGNITUDE: A real number in columns 71 to 77; (F7.2). Specifies the desired magnitude of voltage for P-V and slack buses in KV.

A.4.2 Transmission Line Information Records

Letter in the first column is "L".

Each record contains the information about one particular transmission line in the system. This information (and its format, in parenthesis) is as follows:

LINE NUMBER: A six digit integer number in columns 2 to 7; (I6). This is an arbitrary number assigned by the user and used in the printout.

LINE NAME: An eight character alphanumeric string in columns 9 to 16; (A8).

SENDING END BUS NO.: A five digit integer number in columns 17 to 21 indicating the bus number of one end of the transmission line; (I5). This number must also be specified on a bus information record.

RECEIVING END BUS NO.: A five digit integer number in columns 22 to 26 indicating the bus number of the other end of the transmission line; (I5). This number must also be specified on a bus information record.

LINE IMPEDANCE: A complex number in columns 29, 44; (2F8.5). Specifies the per unit value of the line impedance.
SENDING END SUSCEPTANCE: A real number in columns 49 to 55; (F7.5).

   Specifies the per unit value of shunt admittance at the
   sending end of the line.

RECEIVING END SUSCEPTANCE: A real number in columns 56 to 62; (F7.5).

   Specifies the per unit value of shunt admittance at the
   receiving end of the line.

TRANSFORMER INDICATOR: A logical value in column 64; (LI). A "T" in that
   column indicates that the line has a transformer as well.
   In this case, the data file must also contain a trans­
   former record with the same number. Otherwise, an error
   occurs.

A.4.3 Transformer Records

   Letter in the first column is "T".

   Each record contains the information about one transformer in
   the system. Transformer taps must be fixed. If there is a transmission
   line which also has a transformer, the information about the transmission
   line must appear on a separate "L" record. (The order of the records is
   not important). The information on each record is as follows:

TRANSFORMER NUMBER: A six digit integer number in columns 2 to 7; (I6).

   This number is used to specify the transformer in the
   output printout. If a transformer is part of a trans­
   mission line, this number must be the same as the
   transmission line number on the "L" record.

TRANSFORMER NAME: An eight digit character string in columns 9 to 16;
   (A8).

SENDING END NUMBER: A five digit integer number in columns 17 to 21;
   (I5). Specifies the bus number of the sending end of
the transformer. This number must be specified on a bus information record. If the transformer is part of a transmission line which is specified on a "L" record, this number should be the same on the two records.

RECEIVING END NUMBER: A five digit integer number in columns 22 to 26; (I5). Specifies the bus number of the receiving end of the transformer. The comments made about the above item apply here as well.

TRANSFORMER IMPEDANCE: A complex number in columns 27 to 41; (2F8.5). Specifies the per unit value of transformer impedance.

SENDING END TAP: A real number in columns 43 to 50; (F8.4). Specifies the transformer tap (turn ratio) on the sending end side of the transformer.

RECEIVING END TAP: A real number in columns 51 to 58; (F8.4). Specifies the transformer tap (turn ratio) on the receiving end of the transformer.

A.4.4 The Heading Records

Letter in the first column is "H".

The information in the first 48 columns of this record, with the exception of column 1, are printed at the top of every page as part of the heading information. The remaining positions in the heading will contain date and time. If no such record is given the default heading will be used. If there are several "H" records in a data file, the last one will overwrite the previous ones.
A.4.5 The Comment Records

Letter in the first column is "C".

The information in columns 2 to 80 of these records are printed before the first output page is printed. Up to ten "C" records may be specified for each data file. This data is printed in the same order that it appears in the data file.

A.4.6 The General Control Records

Letter in the first column is "G".

This record may contain the information about the maximum number of iterations permitted, the length and type of output, the desired tolerance (on maximum power mismatches), the "ZERO" of the run (values with absolute values smaller than "ZERO" are assumed zero.), and the MVA base of the system. If no "G" record is specified, or if any of the above information is not specified on the "G" record, the default values would be used. If any of the values is assigned several times, the last assignment would be used.

The information on this record is specified by using several Keywords as follows. Only the first character of each Keyword would be sufficient:

ITERATIONS = maximum number of iterations permitted. This must be a positive integer number. The default value is 100.

LIST = an integer number indicating the length and the type of output to be produced. The following is an explanation of various possibilities:

LIST = 1 indicates that only the first page of the output should be printed. This page contains a summary of the results.
LIST = 2 is used if the full output is to be printed. The output for this case is the output of LIST = 1 plus the complete printout of power flows through various branches. This is the default value.

LIST = 6 or 7 is identical to LIST = 1 or 2 except that, in this case, the input data is printed as well.

LIST = n1, n2, n6 or n7, where n is any positive integer, is exactly identical to LIST = 1, 2, 6 or 7 except that, in this case, the values of desired voltage magnitudes and phase angles and also the values of power mismatches are printed at the end of every n iteration. (See section A.5 for details).

LIST = -1, -2, -6 or -7 is the same as LIST = 1, 2, 6 or 7, respectively except that, in this case, the values of power mismatches and the voltage magnitudes and phase angles of the desired buses (see section A.5 for details), will be plotted as well.

LIST = -n1, -n2, -n6 or -n7 where n is any positive integer, is the same as LIST = -1, -2, -6 or -7, respectively except that, here, the values of power mismatches and the voltage magnitudes and phase angles of the desired buses are also printed at the end of every n iteration. In other words, the effect is that of the two above categories combined.

TOLERANCE = any real-number indicating the desired tolerance for the load-flow study in P.U. The default value is .005 P.U.

ZERO = any real number; specifying the minimum non-zero absolute
value of a number. Numbers with an absolute value less than "ZERO" are assumed zero. The default value is .00005.

BASE MVA: any real number; specifying the MVA base of the system. This value is used in converting the P.U. quantities back into their real values. The default value is 100 MVA.

A.4.7 Data Set Definition Records

 Letter in the first column is "D".

 This record is used to indicate to LFP that the rest of data records are to be read from the unit specified on this record. The format of this record is a "D" in the first column and, a two digit integer number, indicating the number of a logical unit, in columns 10 and 11. This record is usually used as the last record in a data file. In this way, after the data in the file is read, the control is transferred to GUSER, enabling the user to enter any particular commands (data records) he may have for that run.

A.4.8 The End Record

 Letter in the first column "E".

 The first letter is the only information used from this record. As soon as an "E" record is encountered, the input module stops reading any other input records. The effect of "E" records is identical to, and can be achieved by, using $END OF FILE.

 If any record with an initial other than those mentioned above is encountered, either the job will be terminated (batch mode) or the user will be asked to intervene (terminal mode). A sample data file is listed below. This is the data for Test System 6:
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
<th>Value 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>B3</td>
<td>1 ING230</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>3 JHT 13</td>
<td>127.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>4 LDR 13</td>
<td>47.0</td>
<td>5.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>5 SCA 13</td>
<td>25.8</td>
<td>5.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>6 PUN 13</td>
<td>24.5</td>
<td></td>
<td>17.8</td>
<td>8.15</td>
</tr>
<tr>
<td>B2</td>
<td>7 ASH 13</td>
<td>26.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>8 GGA 13</td>
<td>72.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>10 JHTI32</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>11 DBY132</td>
<td></td>
<td>64.0</td>
<td>6.4</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>12 LDR132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>13 SCA122</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>14 GLOD132</td>
<td>0.0</td>
<td>0.0</td>
<td>5.0</td>
<td>2.5</td>
</tr>
<tr>
<td>B1</td>
<td>16 TAS132</td>
<td>0.0</td>
<td>0.0</td>
<td>18.0</td>
<td>5.4</td>
</tr>
<tr>
<td>B1</td>
<td>17 CTI132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>19 CBL132</td>
<td>0.0</td>
<td>0.0</td>
<td>10.6</td>
<td>5.3</td>
</tr>
<tr>
<td>B1</td>
<td>20 PUN132</td>
<td>0.0</td>
<td>0.0</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>B1</td>
<td>22 DMR132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>23 PAL132</td>
<td>0.0</td>
<td>0.0</td>
<td>107.3</td>
<td>36.3</td>
</tr>
<tr>
<td>B1</td>
<td>24 GClL132</td>
<td>0.0</td>
<td>0.0</td>
<td>6.9</td>
<td>2.6</td>
</tr>
<tr>
<td>B1</td>
<td>25 ASH132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>26 NDR132</td>
<td>0.0</td>
<td>0.0</td>
<td>9.6</td>
<td>4.8</td>
</tr>
<tr>
<td>B1</td>
<td>27 NMC132</td>
<td>0.0</td>
<td>0.0</td>
<td>24.0</td>
<td>7.2</td>
</tr>
<tr>
<td>B1</td>
<td>29 JPT132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>30 VIT230</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>31 VIT 12</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>33 JPT 60</td>
<td>0.0</td>
<td>0.86</td>
<td>27.8</td>
<td>15.9</td>
</tr>
<tr>
<td>B1</td>
<td>34 JPT 12</td>
<td>15.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>35 GGA132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>36 CFT132</td>
<td>0.0</td>
<td>0.0</td>
<td>65.4</td>
<td>20.3</td>
</tr>
<tr>
<td>B1</td>
<td>37 ARN230</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>38 ARH132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>39 ST132</td>
<td>25.0</td>
<td>93.0</td>
<td>197.5</td>
<td>140.5</td>
</tr>
<tr>
<td>B1</td>
<td>40 VIT132</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>L</td>
<td>15 TAS132</td>
<td>14</td>
<td>16</td>
<td>0.01575</td>
<td>0.01439</td>
</tr>
<tr>
<td>L</td>
<td>17 DBY132</td>
<td>10</td>
<td>11</td>
<td>0.0017</td>
<td>0.0052</td>
</tr>
<tr>
<td>L</td>
<td>19 GLOD132</td>
<td>13</td>
<td>14</td>
<td>0.0516</td>
<td>0.125</td>
</tr>
<tr>
<td>L</td>
<td>20 SCA122</td>
<td>12</td>
<td>13</td>
<td>0.0168</td>
<td>0.1448</td>
</tr>
<tr>
<td>L</td>
<td>21 LDR132</td>
<td>10</td>
<td>12</td>
<td>0.0017</td>
<td>0.0136</td>
</tr>
<tr>
<td>L</td>
<td>22 CBL132</td>
<td>17</td>
<td>19</td>
<td>0.00416</td>
<td>0.00128</td>
</tr>
<tr>
<td>L</td>
<td>23 CTI132</td>
<td>10</td>
<td>17</td>
<td>0.0208</td>
<td>0.0314</td>
</tr>
<tr>
<td>L</td>
<td>24 PUN132</td>
<td>17</td>
<td>20</td>
<td>0.04102</td>
<td>0.0296</td>
</tr>
<tr>
<td>L</td>
<td>25 DMR132</td>
<td>10</td>
<td>22</td>
<td>0.0495</td>
<td>0.1227</td>
</tr>
<tr>
<td>L</td>
<td>26 DMR132</td>
<td>20</td>
<td>22</td>
<td>0.0464</td>
<td>0.1194</td>
</tr>
<tr>
<td>L</td>
<td>27 ASH132</td>
<td>24</td>
<td>25</td>
<td>0.0161</td>
<td>0.0427</td>
</tr>
<tr>
<td>L</td>
<td>28 GCLL132</td>
<td>23</td>
<td>24</td>
<td>0.0176</td>
<td>0.0458</td>
</tr>
<tr>
<td>L</td>
<td>29 PAL132</td>
<td>12</td>
<td>23</td>
<td>0.0167</td>
<td>0.0326</td>
</tr>
<tr>
<td>L</td>
<td>30 JPT132</td>
<td>11</td>
<td>12</td>
<td>0.0354</td>
<td>0.075</td>
</tr>
<tr>
<td>L</td>
<td>31 JPT132</td>
<td>27</td>
<td>29</td>
<td>0.0165</td>
<td>0.0431</td>
</tr>
<tr>
<td>L</td>
<td>32 JPT132</td>
<td>27</td>
<td>29</td>
<td>0.0165</td>
<td>0.0431</td>
</tr>
<tr>
<td>L</td>
<td>33 ST0132</td>
<td>29</td>
<td>30</td>
<td>0.0440</td>
<td>0.1299</td>
</tr>
<tr>
<td>L</td>
<td>34 GGA132</td>
<td>29</td>
<td>35</td>
<td>0.0416</td>
<td>0.1231</td>
</tr>
<tr>
<td>L</td>
<td>35 CFT132</td>
<td>25</td>
<td>36</td>
<td>0.0128</td>
<td>0.038</td>
</tr>
<tr>
<td>L</td>
<td>36 ST0132</td>
<td>36</td>
<td>39</td>
<td>0.0094</td>
<td>0.0279</td>
</tr>
<tr>
<td>L</td>
<td>37 VIT132</td>
<td>39</td>
<td>40</td>
<td>0.00373</td>
<td>0.03133</td>
</tr>
<tr>
<td>L</td>
<td>39 VIT132</td>
<td>38</td>
<td>40</td>
<td>0.02842</td>
<td>0.06428</td>
</tr>
<tr>
<td>L</td>
<td>40 ARN230</td>
<td>1</td>
<td>37</td>
<td>0.0226</td>
<td>0.0186</td>
</tr>
<tr>
<td>T</td>
<td>1 SCA132</td>
<td>5</td>
<td>13</td>
<td>0.0632</td>
<td>0.272</td>
</tr>
<tr>
<td>T</td>
<td>2 LDR132</td>
<td>4</td>
<td>12</td>
<td>0.00949</td>
<td>0.1583</td>
</tr>
<tr>
<td>T</td>
<td>3 JHT132</td>
<td>3</td>
<td>10</td>
<td>0.00375</td>
<td>0.06245</td>
</tr>
<tr>
<td>T</td>
<td>4 PUN132</td>
<td>6</td>
<td>20</td>
<td>0.03534</td>
<td>0.05893</td>
</tr>
<tr>
<td>T</td>
<td>5 ASH132</td>
<td>7</td>
<td>25</td>
<td>0.02202</td>
<td>0.03671</td>
</tr>
<tr>
<td>T</td>
<td>6 JPT 60</td>
<td>29</td>
<td>33</td>
<td>0.0188</td>
<td>0.0296</td>
</tr>
<tr>
<td>T</td>
<td>7 JPT 12</td>
<td>33</td>
<td>34</td>
<td>0.00641</td>
<td>0.1602</td>
</tr>
<tr>
<td>T</td>
<td>8 GGA132</td>
<td>8</td>
<td>35</td>
<td>0.0393</td>
<td>0.0962</td>
</tr>
<tr>
<td>T</td>
<td>10 VIT132</td>
<td>30</td>
<td>40</td>
<td>0.003</td>
<td>0.096</td>
</tr>
<tr>
<td>T</td>
<td>11 VIT 12</td>
<td>30</td>
<td>30</td>
<td>0.00334</td>
<td>0.0667</td>
</tr>
<tr>
<td>T</td>
<td>13 ARN132</td>
<td>37</td>
<td>38</td>
<td>0.00114</td>
<td>0.0228</td>
</tr>
</tbody>
</table>
A.5 Comments and Restrictions

(1) If START = N is specified, LFP expects to read the ini-
tial estimates of line loss or voltages - depending upon the load-flow
type being used - from a file or device. The name of this file or
device is read from GUSER. In terminal mode, the user is asked to pro-
vide this information while, in batch mode, the program expects to find
this information immediately after the data records. (There is one
exception to this. See the next comment). The name of this file or
device must be found in the first 20 columns of the input record. This
file or device must contain the initial values of voltages - for
algorithms whose name does not start with "L" - or line losses - for
algorithms whose names start with "L". If no name is given, the data
is expected to come from GUSER.

The format of the data for entering the values of voltages is
different from that for entering the initial values of line losses. In the
former case, the initial values of bus voltages must be specified, in
the same order as that of the input bus records but skipping over the
slack buses, using 2F10.5 format. In other words, no bus numbers are
necessary since the order is already fixed. The voltage magnitudes
(in P.U.) and the voltage phase angles (in degrees) are specified using
the above format.

In the latter case, the sending and receiving end bus numbers
are specified, followed by the P.U. values of line losses. The order
of the records is not important since the sending and receiving end
numbers are specified. The format of each record is 2I5, 2F10.5. Note
that the Per Unit values of line-losses have to be specified.

In each case the terminal user is given any necessary informa-
tion with respect to the points mentioned here.
(2) When the LIST Keyword (on the "G" record) is specified such that printing and/or plotting of variables at the end of various iteration steps is desired, the program expects to read the bus numbers whose voltage magnitudes and phase angles are wanted. The terminal user is asked for this information at the proper place. The batch user is expected to enter this information (from *MSOURCE*) immediately after the "E" record. Note that, if START = N is specified at the same time, the information with respect to the initial estimates must follow the information mentioned here.

The format of the data is 2014. Unspecified bus numbers are ignored. However, if a bus number is entered, and later the negative value of the same number is encountered, the information with respect to that bus is not printed/plotted. In other words, the negative bus numbers may be used to cancel the positive numbers previously entered. If the information about all the voltages (except the slack buses) is required, the word ALL can be entered on the first three columns of the record. Otherwise, the bus numbers are read, using 2014 format, until a zero bus number is encountered at which point the information is considered to be complete. The power mismatches will always be printed, along with the information about the desired voltages, at the end of the respective iterations. Therefore, if only the power mismatches are wanted, a blank record must follow the "E" record, to indicate that there are no bus numbers specified.

(3) The following routines were used from UBC Computing Centre's library. They must be available at the time of the run:
Character Manipulation Routines: MOVEC, SETC, FINC, FINDST, IGC, DTB, EQUC.
Input/Output and Dynamic Storage Allocation Routines: SERCOM, GUSER, SETPFX, GINFO, CUINFO, FTNCMD, EMPTYF, CREPLY, GSPACE, FSPACE, CALLER.

Bit Manipulation Routines: SHFTL, SHFTR, LOR, LXOR.

Plotting (Metric): PLCTRL, SYMBOL, NUMBER, PLOT, AXCTRL, SCALE, AXPLOT.

Other Routines: CDATE, PAR, LDINFO, TIME.

Apart from the above routines, many of the MTS features are also used. The program has to be updated according to the changes that may occur to these features. A listing of the program follows.
SUBROUTINE PARR

LOGICAL*1 IST(255),BATCH,HDNGS(80),MTSU(30),NAME(16),ALG(8)
COMMON /OUTINF/ NPAGE,NLINE,HDNGS,ISTNO
COMMON /ADS/ IA,ID,IS,ALG,BATCH
INTEGER*2 LEN
INTEGER START,FINISH
LENUM=1
CALL MOVEC(14,*ASSIGN SCARS=*,MTSU)
CALL SETC(80,HDNGS,* *)
CALL MOVEC(5,*ALG=*,HDNGS)
CALL MOVEC(5,*DATA=*,HDNGS(14))
CALL MOVEC(6,*START=*,HDNGS(39))
CALL DATE(49))
CALL PAR(IST,NI,255,&100)
GO TO 300
10 ICIFIBATCH) GO TO 150
LEN=6
CALL SERCOM(* PAR=*LEN,0)
110 CALL SETPF(*?*,1)
CALL GUSER(IST,LEN,0,LNUM,&150)
21 NF=LEN
23 GO TO 300
24 LEN=15
25 CALL SERCOM(* NO PAR FIELD GIVEN*,LEN,0)
26 CALL EXIT
27 300 IA=1
28 ID=0
29 IS=0
30 CALL FINDC(IST,NI,*ADS*,3,START,FINISH,ICF,&400,&400)
31 IF(IA .NE. 0) GO TO 500
32 LEN=27
33 IF(PATCH) CALL EXIT
34 CALL SERCOM(* ALGORITHM DOES NOT DEFAULT*,LEN,0)
35 LEN=24
36 IF(PATCH) CALL EXIT
37 CALL FINDC(IST,NI,*ADS*,3,START,FINISH,ICF,600,600)
38 IF(START .EQ. FINISH) GO TO 350
39 LEN=19
40 IF(START .NE. FINISH) GO TO 350
41 LEN=19
42 CALL SERCOM(* ERROR IN PAR FIELD*,LEN,0)
43 CALL EXIT
44 CALL FINDC(IST,NI,*ADS*,3,START,FINISH,ICF,600,600)
45 IF(START .EQ. FINISH) GO TO 350
46 CALL EXIT
47 GC TC (1000,2000,3000),ICF
48 CALL IGC(IST,NI,* *,1,START,FINISH,ICF,600,600)
49 CALL EXIT
50 LEN=19
51 IF(START .EQ. FINISH) GO TO 350
52 GC TC (1000,2000,3000),ICF
53 CALL EXIT
54 IF(IA .GE. 6) IA=6
55 CALL SETC(8,ALG,* *)
56 CALL MOVEC(IA,IST(START),ALG)
57 CALL MOVEC(8,ALG,HDNGS(6))
58 START=FINISH+1
GO TO 350

100.

FIRST IGNORE ALL THE PRECEDING BLANKS:

CALL IGC(IST,NI,' ',1,START,FINISH,6400,6400)

CALL FINDC(IST,NI,' ',2,START,FINISH,ICF,62100,6400)

CHECK TO SEE IF "DATA=", " HAS HAPPENED:

IF(FINISH .EQ. START) GO TO 350

GO TO 2200

FINISH=NI+1

NUMB=FINISH-START

ID=NUMB

CALL MOVEC(NUMB,IST(START),HDNGS(19))

CALL MOVEC(NUMB,IST(START),MTSU(15))

NCMTS=NUMB+14

LL=19+NUMB

NUMB=20-NUMB

CALL SETC(NUMB,HDNGS(LL),' ')

CALL FTNCDU(PTSU,NOMTS)

START=FINISH+1

GO TO 350

FIRST IGNORE ALL THE PRECEDING BLANKS:

CALL IGC(IST,NI,' ',1,START,FINISH,63100,63100)

START=FINISH

CALL FINDC(IST,NI,' ',2,START,FINISH,IS,63100,63100)

IF(IS .EQ. 1) GO TO 3050

CALL FINDC(IST,NI,' ',2,START,FINISH,ICF,63020,63020)

START=FINISH+1

GO TO 3050

START=NI+1

FINISH=NI+1

IF(IS .LT. 3) GO TO 3200

CALL MOVEC(4, 'N ', HDNGS(45))

IS=2

GO TO 350

START=NI-1

IS=1

GO TO 350

IF(IS .NE. 0) GO TO 5500

CALL GTNAME('SCAROS',NAME,LI)

CALL MOVEC(L,NAME,HDNGS(19))

LL=19+L

L=20-L

CALL SETC(L,HCNC(L), ' ')

IF(IS .EQ. 1) CALL MOVEC(4,'F ',HDNGS(45))

RETURN

END

SUBROUTINE INPUT(COMNT,J)

INTEGER FINISH,DSET,START

COMMON/INF/1,12,13,ITMAX,TOLERN,ZERO,BMVA,L1NE,LIST,IEND

COMMON/ADS/1,10,IS,ALG,BATCH

INTEGER*2 TYPE,NO,FRMT,TI,TLEN,BUSES(20)

LOGICAL*1 CODE,CODES(8),/ 'B', 'L', 'T', 'H', 'C', 'E', 'G', 'D'/

LOGICAL*1 HCNC-SCS(8),HCNC-HDNGS(EC),COMNT(1),ERROR(80),TX,BATCH,ALG(8)

LOGICAL EUC

REAL*8 NAME

EQUIVALENCE (BUSES(1),ERROR(1))

REAL*4 BASE,SLS,SURE,PTAP,QTAP,U

COMPLEX GEN,LO,Z
BMVA=100
ICD=-10
LIST=2
ITMAX=100
TOLERN=0.005
ZERO=.00005

105
I1=0
I2=0
I3=0
LINE=0
J2=0
DSET=5

112
CALL FTNCMD('ASSIGN 1=-BUSINF*,16)
113
CALL EMPTYF(1)
114
CALL FTNCMD('ASSIGN 2=-LININF*,16)
115
CALL EMPTYF(2)
116
CALL FTNCMD('ASSIGN 3=-TXINF*,15)
117
CALL EMPTYF(3)
118
CALL FTNCMD('ASSIGN 4=-SCRATCH*,17)
119
CALL EMPTYF(4)

120
50 CONTINUE
121
IF( ICD .LT. 0) GO TO 60
122
DSET=IOD
123
ICD=-10
124
CALL SETPFX1,1
125
GO TO 50
126

130
100 I1=1
131
READ(DSET,101(TYPE,NO,NAME,GEN,LD,BASE,U
132
WRITE(I)TYPE,NO,NAME,GEN,LD,BASE,U
133
101 FCPTM11X,I1,I5.1X,A8,2X,2F7.2,2F8.2,15X,
134
GC TC 50
135
102 I2=12+1
136
READ(DSET,201)NAME,FROM,TO,Z,SUSE,SURE,TF
137
WRITE(2)NC,NAME,FROM,TO,Z,SUSE,SURE,TF
138
201 FORMAT(I6,1X,A8,2I5,2X,2F8.5,4X,2F7.5,1X,L1
139
IF(.NOT. TF) LINE=LINE+1
140
GC TC 50
141
103 C TRANSFORMER DATA CARD:
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
C GENERAL INFORMATION RECORD:

REAC(DSET,601)ERROR

FORMAT(80A1)
LEN2=80
START=2

CALL FINDC(ERROR,LEN2,"ITZLB",5,START,FINISH,ICF,650,650)
START=FINISH+1

CALL FINDC(ERROR,LEN2,"=",1,START,FINISH,ICE,605,605)
START=FINISH+1

LIMIT=LEN2-START+1
GO TO (610,620,610,620,ICF)

CALL DTB(ERROR(START),INT,LIMIT,NSD,*,*615)
IF(ICF .EQ. 1) GO TO 612

LIST=INT
GO TO 615

ITMAX=INT
START=START+LIMIT+1
GO TO 605

CALL DTB(ERRCPF(START),INT1,LIMIT,NSD,*,*622)
START=START+1
GO TO 605

LIMIT=LEN2-START+1
CALL DTB(ERROR(START),INT2,LIMIT,NSD,*,*630)
START=START+LIMIT
LIMIT=LIMIT-1
IF(INT1 .EQ. C AND. INT2 .EQ. 0) GO TO 605
A1=INT1
A2=INT2
A=A1+A2*10**(-LIMIT)
GO TO (627,627,628,628,629),ICF

START=START+LIMIT+1
IF(INT1 .EQ. 0) GO TO 605

A=INT1
GO TO 626

TOLERN=A
GO TO 605

ZERG=A
GO TO 605

BMVA=A
GO TO 605

START=START+LIMIT+1
IF(INT1 .EQ. 0) GO TO 605
A=INT1
GO TO 626

700 REAC(DSET,701)NEWD

7C1 FORMAT(9X,12)
IF(IO00 .GT. 0)IOO=NEWD
DSET=NEWD
IF(DSET = EQ. 15) CALL FTNCMD("ASSIGN SCARDS=*SCURCE*",22)
GO TO 50

800 REAC(DSET,61)CCDE

850 LINE=LINE+13
IF(LIST .GT. 10 OR. LIST .LT. 0) GO TO 900
RETURN

500 IF(.NOT. BATCH)WRITE(16,901)
901 FORMAT(* ENTER THE DESIRED BUS NUMBERS IN 2014 FORMAT OR "ALL"
*/* (A ZERO BUS NUMBER INDICATES END OF SERIES)"
READ(15,61)CODE
IF(ECUC(CODE,'A')) GC TO 950
BACKSPACE 15
910 REAC(15,911)ELSESES

511 FORMAT(2014)
240 WRITE(11) BUSES
241 IF(BUSES(20) .EQ. 0) RETURN
242 GO TO 910
243 550 BUSES(11) = -999
244 WRITE(11) BUSES
245 RETURN
246 1000 READ(DSET, 1001) ERROR
247 1001 FORMAT(80A1)
248 LEN=41
249 CALL SERCOM(' INVALID INITIAL ON THE FOLLOWING RECORD: ', LEN, 0)
250 LEN=50
251 CALL SERCOM(ERROR, LEN, 0)
252 IF(BATCH) GO TO 1500
253 LEN=54
254 CALL SERCOM(' ENTER Y TO REPLACE N TO TERMINATE OR RETURN TO I G N O R
255 'E', LEN, 0)
256 CALL SETPFX('?', 1)
257 READ(16, 61) CODE
258 CALL SETPFX('E ', 1)
259 IF(EQUC(CODE, 'Y')) GO TO 1100
260 IF(EQUC(CODE, 'Y')) GC TO 1200
261 WRITE(16, 1501)
262 CALL EXIT
263 1100 LEN=15
264 CALL SETPFX('E ', 1)
265 CALL SERCOM(' RECORD IGNORED', LEN, 0)
266 1200 DSET=15
267 GO TO 1500
268 1500 CALL SETPFX('E ', 1)
269 WRITE(6, 1501)
270 1501 FORMAT(' JOB TERMINATED BY ERROR')
271 STOP
272 END
273 SUBROUTINE HEAD
274 COMMON /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO
275 LOGICAL*1 HDNGS(80)
276 NPAGE = NPAGE + 1
277 WRITE(16, HDNGS, NPAGE = 1
278 WRITE(6, HDNGS, NPAGE = 1
279 WRITE(16, HDNGS, NPAGE = 1
280 1 FORMAT(1H1, 5X, 8GA1, 15X, 'PAGE' , 13/110('*')/1
281 NLINE=3
282 RETURN
283 END
284 SUBROUTINE BLSHDR
285 COMMON /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO
286 LOGICAL*1 HDNGS(80)
287 COMMON /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO
288 LOGICAL*1 HDNGS(80)
289 CALL HEAD
290 WRITE(6, 101)
291 101 FORMAT(' BUS INFORMATION :' ['X, '30(' ')/5X, 'BUS # ')
294 NLINE = NLINE + 6
295 RETURN
296 END
297 SUBROUTINE LFDR
298 COMMON /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO
299 LOGICAL*1 HDNGS(80)
300 CALL HEAD
301 WRITE(6, 101)
104.

1 FORMAT(* LINE INFORMATION : *)

200 WRITE(6,201)

2C1 FORMAT(* X, 'LINE #', 5X, 'LINE NAME', 9X, 'S.E.', 3X, 'R.E.'


2C1 * EP.* (X, 'TX?/*)

305 NLINE=NLINE+5

318 RETURN

319 END

320 SUBROUTINE TXHDR

331 COMMN /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO

311 CALL * READ

312 WRITE(6,11)

313 FORMAT(* TRANSFORMER INFORMATION : *)

315 WRITE(6,201)


317 NLINE=NLINE+5

318 RETURN

319 END

320 SUBROUTINE BUSINFO (SCHEC, V, U, ANG, NO, TYPE, INPRT)

321 LOGICAL*1 BATCH, HDNGS(80), INPRT, MTSU(30), ALG(8)

322 COMMN /ININF/ I1, 12, 13, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND

323 COMMN /OUTINF/ NPAGE, NLINE, HDNGS, ISTNO

324 COMMN /ADS/ 1A, ID, 15, ALG, BATCH

325 LOGICAL EQUC

326 REAL*8 NAME

327 INTEGER*2 NO(1), BUSES(20), TYPE(1)

328 REAL*8 U(I), ANG(I), BASE

329 COMPLEX SCHEC(I), V(I), GEN, LD

330 EQUIVALENCE (NBUS, U(I))

331 EQUIVALENCE (MTSU(I), NRTST)

332 DATA NRTST /4C404040/

333 READ(1) TYPE(I), NO(I), NAME, GEN, LD, BASE, U(I)

334 IF (.NOT. INPRT) GO TO 75

335 U(I)=U(I)/BASE

336 I=1

337 5C READ(1) TYPE(I), NO(I), NAME, GEN, LD, BASE, U(I)

338 IF (.NOT. INPRT) GO TO 75

339 IF (NLINE .GT. 50) CALL BUSHDR

340 WRITE(6,51) INC(I), NAME, TYPE(I), GEN, LD, BASE, U(I)

341 NLINE=NLINE+1


344 * EP.* (X, 'TX?/*)

345 75 IF (TYPE(I) .EQ. 3) GO TO 100

346 U(I)=U(I)/BASE

347 SCHED(I)=1GEN-LDI/BMVA

348 ANG(I)=0.

349 IF (1 .EQ. IENC) GO TO 200

350 I=I-1

351 GC TO 50

352 100 TYPE(IEND)=TYPE(I)

353 NO(IEND)=NO(I)

354 U(IEND)=U(I)/BASE

355 V(IEND)=U(IEND)

356 ANG(IEND)=0.

357 IEND=IEND-1

358 IF (IEND .LT. I) GO TO 200

359 GO TO 50

360 200 DO 220 I=1, IEND

361 IF (TYPE(I) .EQ. 1) U(I)=1.
V(I) = U(I)

READ(1, END=250) BUSES
IF(BUSES(I) .LT. 0) GO TO 243
DO 240 I = 1, NBUS
IF(BUSES(I) .EQ. 0) GO TO 250
DO 230 J = 1, NBUS
IF(IN(J) .EQ. BUSES(I)) GO TO 243
CONTINUE
GO TO 240
230 CONTINUE
GO TO 240
235 NO(J) = - NO(J)
240 CONTINUE
243 DO 246 I = 1, NBUS
NO(I) = - NO(I)
246 CONTINUE
250 IF(IS .EQ. 1 .OR. EQUC(ALG(1), 'L')) RETURN
IF(BATCH) GO TO 260
WRITE(16, 251)
251 FORMAT(' ENTER NAME OF THE UNIT CONTAINING STARTING ** VALUE DATA')
260 IF('RETURN' .EQ. 'EOF') RETURN
CALL SETPFX(I*, 1)
261 IF(INTRN .EQ. NRTST) GO TO 270
CALL MOVEC(12), ASSIGN 8 = MTSU
CALL FTNCMD(MTSU, 30)
270 CONTINUE
CALL FTNCMD('EQUATE 8=GUSER*, 14')
280 IF(BATCH) GO TO 280
WRITE(16, 271)
271 FORMAT('* ENTER MAG*, ANG* IN 2F10.5 FORMAT/*
* ORDER THE SAME AS BUS DATA, SKIP OVER SLACK BUSES.**
* VOLTAGES ARE READ IN P.U., ANGLES IN DEGREES:
* THE ORDER IS THE SAME AS BUS DATA RECORDS EXCEPT THAT NO DATA
* IS GIVEN FOR SLACK BUS(ES).
* DO 290 I = 1, IEND
READ(8, 281, END=300) U(I), ANG(I)
281 CONTINUE
290 CALL SETPFX(*6*, 1)
300 CONTINUE
RETURN
END
REWIND 2
REWIND 3
C TO SET POINTERS FOR DIAGONAL ELEMENTS:
K=1
DO 20 I=1,NBUS
Y(2*I) = 0.
CPONT(K) = 1
RPONT(K) = I
NXTR(K) = K
NXTC(K) = K
20 CONTINUE
C TO SET POINTERS FOR OFF-DIAGONAL ELEMENTS:
GP = K
LAST = GP + LINE*8 - 16
IF (GP .GT. LAST) GO TO 35
DO 30 I = GP, LAST, 8
NXTR(I) = I + 8
30 CONTINUE
DO 35 I = GP, LAST + 8 = 0
IF (13 .EQ. 0) GO TO 60
IF (INPRT) CALL TXHDR
DO 50 I = 1, 13
READ(3) TXN0(I), NAME, PSIDE(I), QSIDE(I), TXZ(I), PTAP(I), QTAP(I)
50 CONTINUE
CALL TXHDR
IF (INPRT) CALL LHDR
DO TCO J = 1, 12
READ(2) LTNO, NAME, FROM, TO, Z, SESU, RESU, TX
IF (.NOT. INPRT) GO TO 75
IF (NLINE .LE. 50) GO TO 70
CALL LHDR
WRITE(6, 71) ILTNC, NAME, FROM, TO, Z, SESU, RESU, TX
NLINE = NLINE + 1
75 CONTINUE
YP1 = CMPLX(0., SESU)
YP2 = CMPLX(0., RESU)
IF (.NOT. TX) GO TO 300
DO 150 I = 1, 13
IF (TXNO(I) .EQ. LTNO) GO TO 160
CONTINUE
150 CONTINUE
TXNO(I) = 0
C TRANSFORMERS ARE ALWAYS ASSUMED TO BE AT S.E.
FPX = PSIDE(I)
TC = QSIDE(I)
RATIO = QTAP(I)/PTAP(I)
IF (CABS(TXZ(I)) .gt. 0.) GO TO 200
Z = 1./Z
ZA = YP1 + Z
YP = RATIO*Z
YP2 = ZA + Z
YP1 = YB + Z
GO TO 350
200 ZA = Z
ZC = TXZ(I) / RATIO

YP1 = (1. - RATIO) / TXZ(I) + YP1

Z1 = ZA + ZC + ZA * 2C * YB

Y3 = YB * ZA / Z1

YP2 = YP2 * Y2

YP1 = RATIO * (RATIO - 1.) / TXZ(I) + Y3

Z = -1. / Z1

GO TO 350

Z = -1. / Z

CONTINUE

WRITE(4) LTN0, NAME, FROM, TO, YP1, YP2, Z

CALL SET(IFRCM, TO, Z, YP1, YP2, NO, Y, CPCNT, RPONT, NXTR, NXTC, GP)

CONTINUE

IF(I3 .EQ. 0) GO TO 1000

RETHEN 3

DO 800 I = 1, 13

READ(3) LTN0, NAME, FROM, TO, FILLER

IF(TXNO1I) .EQ. 0) GO TO 800

RATIO = QTAP(I) / PTAP(I)

IF(CABS(TXZ(I)) .EQ. 0.) TXZ(I) = CMPLX(0., ZERO)

YP2 = 1. / TXZ(I)

Z = -YP2 * RATIO

YP2 = YP2 * Z

YP1 = -RATIO * YP2

WRITE(4) LTN0, NAME, FROM, TO, YP1, YP2, Z

CALL SET(IFRCM, TO, Z, YP1, YP2, NO, Y, CPCNT, RPONT, NXTR, NXTC, GP)

CONTINUE

1000 RETURN

END

SUBROUTINE SET(FROM, TO, SY, YP1, YP2, NO, Y, CPCNT, RPONT, NXTR, NXTC, GP)

COMMON/INFIN/11, 12, 13, ITMAX, TOLERN, ZERO, BHYA, LINE, LIST, IEND

INTEGER GP, VP, PP

INTEGER*2 FROM, TO, NO(I), CPCNT(I), RFCNT(I), NXTR(I), NXTC(I)

COMPLEX SY, YP1, YP2, Y(I)

EQUIVALENCE(I1, NBUS)

SY IS THE NEGATIVE OF SELF ADMITTANCE

YP1 AND YP2 ARE THE LOCAL ADMITTANCES AT SENDING AND RECEIVING ENDS.

TO AND FROM ARE THE EXTERNAL BUS NUMBERS OF SENDING AND RECEIVING ENDS.

FIRST FIND THE INTERNAL NUMBERS CORRESPONDING TO "TO" AND "FROM".

GO 100 I = 1, NBUS

NOI = NO(I)

K = IABS(NO I)

IF(K .EQ. TO) TO = -1

IF(K .EQ. FROM) FROM = -1

CONTINUE

TC = -TC

FROM = -FROM

CHANCE THE SELF ADMITTANCES OF BOTH ENDS:

Y(KK) = Y(KK) - SY + YP1

JJ = TC * 2

YIJJ = YJJ - SY + YP2

SET THE MUTUAL ADMITTANCE ELEMENT:

IF THE ELEMENT EXISTS, MODIFY IT:

CALL SCAN(Y, CPCNT, NXTR, FROM, TO, VP, PP)

IF(VP .EQ. 0) GO TO 600

Y(VP) = Y(VP) + SY

GO TO 700

IF THE ELEMENT IS NEW, INSERT IT:
CALL INSRT(Y,CPCNT,RPONT,NXTR,NXTC,GP,FROM,TC,SY,&700)
RETURN
END
SUBROUTINE SCAN(Y,CPCNT,NXTR,FROM,TC,VP,PP)
INTEGER*2 CPCNT(1),NXTR(1),FROM,TC,TEMP
INTEGER FIRST,VP,PP
COMPLEX Y(1)
ONLY UPPER HALF OF THE ADMITTANCE MATRIX IS FORMED.
THEREFORE, ELEMENT (ROW, COL) EXISTS IF (ROW .LE. COL).
IF(FRCM-TO)1CC,2C0,300
VP=0
PP=0
FIRST=FROM*8-7
NEXT=NXTR(FIRST)
12C IF(NEXT.EQ.FIRST)GO TO 150
IF(CPCNT(NEXT) .EQ. TO) GO TO 130
NEXT=NXTR(NEXT)
GO TO 120
13C PP=NEXT
VP=(NEXT+7)/4
RETURN
200 VP=FROM*2
FP=VP*4-7
RETURN
300 TEMP=FROM
FRCM=TO
TO=TEMP
GO TO 100
END
SUBROUTINE INSRT(Y,CPCNT,RPONT,NXTR,NXTC,GP,FROM,TC,VALUE,*)
INTEGER VP,PP,GP
COMPLEX VALUE,Y(1)
INTEGER*2 CPCNT(1),RPONT(1),NXTR(1),NXTC(1),FRCM,TC,VALUE,*)
ERROR RETURN 1 = NO INSERTION REQUIRED
ERROR RETURN 2 = THERE IS NO SPACE IN GARBAGE C
GARBAGE POINTER(GP) IS THE (PP) CF THE SPECIFIED LOCATION.
IF(CABS(VALUE) .EQ. 0.)RETURN 1
IF(GP .EQ. 0.)RETURN 2
LL=NXTR(GP)
PP=FRCM*8-7
L=NXTR(PP)
NXTR(PP)=GP
NXTR(GP)=L
RPONT(GP)=FRCM
PP=TC*8-7
L=NXTC(PP)
NXTC(PP)=GP
NXTC(GP)=L
CPCNT(GP)=TC
VP=(GP+7)*4
Y(VP)=VALUE
GP=LL
RETURN
END
EXTERNAL BUSINF,LININF,PROSS,FLOWS,LNFS T
INTEGER ASCHEC,AV,AU,AANG,AND,ATYPE,AY,AJQ,FT
INTEGER ECHO1,ECHO2/I/,ATZ,OUT(20)
LOGICAL I.CCMN(790),HONGS(80),BATCH,INPRT,ALG(8)
COMMON /OUTINF/ NPAGE,NLINE,HONGS,ISTNO
COMMON /OUTINF/ NPAGE,NLINE,HONGS,ISTNO
COMMEN /OUTINF/ NPAGE,NLINE,HONGS,ISTNO
BEGIN
COMMON /ADS/ IA, ID, IS, ALG, BATCH

LOGICAL EQUC

COMMON /LOSSSES/ SL

C EQUVALENCE(NABUS,11)

C STORE ECHO STATUS OF USER

CALL QUINFO( 'ECHOFF ' , ECHO1)

C THEN SET ECH=E=OFF:

CALL QUINFO( 'ECHOFF ' , ECH2)

NPAGE = 0

ISTNC = 0

BATCH = .TRUE.

IF (J2 .EQ. 00 TO 100)

CALL HEAD

WRITE(6,1)(CCMNT(J),J=1,J2)

1 FORMAT ('COMMENTS : ' , 17(•*) // */5X, •*)

100 CALL GSPACE(ASCHED, NBUS*28)

AV = ASCHED + NBUS*8

AL = AV - NBUS*8

AANG = AV - NBUS*8

ANC = AANG + NBUS*4

4TYPE = ANC * NBLS * 2

INPRT = IABS(MCC(List,10)) .GT. 5

CALL CALLER (BL INF , ASCHED, AV, AU, AANG, ANC, ATYPE, IPTR(INPRT))

CALL GSPACE(ATZ)

GO TO 120

CALL CALLER (LINF I N, IPTR(INPRT), AV, AU, AANG, ANC, ATYPE, IPTR(INPRT))

CALL GSPACE (ATZ)

GO TO 120

CALL CALLER (LINF I N, IPTR(INPRT), AV, AU, AANG, ANC, ATYPE, IPTR(INPRT))

CALL GSPACE (ATZ)

LIST = IABS(MCC(List,10)) .GT. 5

CALL CALLER (PROSS, IPTR(OUT), AV, AU, AANG, ASEQ, &1111)

CALL GSPACE (ASEQ)

LIST = IABS(LIST)

CALL GSPACE (ASEQ)

C RESTORE USER'S ECHO STATUS

CALL QUINFO ( 'ECHOFF ' , ECHO1)

WRITE(6,201)

END
READ STOP
READ WRITE(6,112) ISTNO
READ FORMAT('
**ERROR RETURN **/5X,ISTNO=*,17)
READ GC TC 200
READ END
READ SUBROUTINE PROSS(ALG,Y,CPONT,NXTR,RPONT,NXTC,NO,TYPE,SCHED,V,U,
        *ANG,SEQ*)
READ EXTERNAL NEWTON
READ COMPLEX Y(I),SCHED(I),V(I)
READ INTEGER*2 CPCNT(I),NXTR(I),RPONT(I),NXTC(I),NC(I),TYPE(I)
READ INTEGER*2 SEC(I)
READ REAL U(I),ANG(I)
READ CALL ALG(V,U,ANG,NO,SCHED,TYPE,Y,CPONT,NXTR,RPONT,NXTC)
READ RETURN
READ RETURN
READ END
READ SUBROUTINE TEST(V,U,ANG,NO,SCHED,TYPE,SEO,Y,CPONT,NXTR,RPONT,NXTC)
READ INTEGER*2 SEC(I)
READ COMPLEX Y(I),SCHED(I),V(I)
READ INTEGER CPCNT(I),NXTR(I),RPONT(I),NXTC(I),NO(I),TYPE(I)
READ REAL U(I),ANG(I)
READ COMMON /OUTINF/I1,12,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
READ COMMON /ADS/I1,ND,IS,ALG,BATCH
READ LOGICAL I1 I11S(80),BATCH,ALG(8)
READ EQUIVALENCE NBUS(I)
READ N1=(NBUS+LINE*2)
READ WRITE(6,101)
READ FORMAT(*'/20X,'ADDITIONAL MATRIX:  '/*7X,'VALUE',
        *7X,'CPONT',5X,'NXTR',5X,'RPONT',5X,'NXTC'//)
READ K=1
READ I=2
READ WRITE(6,2)Y(I),CPCNT(K),NXTR(K),RPONT(K),NXTC(K)
READ FORMAT(2F10.4,4I10)
READ K=K+8
READ IF(I.GT.N1IGC TO 100
READ GO TO 10
READ WRITE(6,103)
READ FORMAT(*'/20X,'INTERNAL BUS ARRANGEMENT:',
        *'/5X,'NO',5X,'TYPE',10X,'SCHED',10X,'V',5X,'U',5X,'ANG'//)
READ WRITE(6,1022)NO(I),TYPE(I),SCHED(I),V(I),U(I),ANG(I),I=1,NBUS
READ FORMAT(3X,2I5,6F10.4)
READ WRITE(6,105)
READ WRITE(6,103)
READ FORMAT(*'/20X,'COMMON AREA S:',/5X,'/ININF://)
READ WRITE(6,104)I1,12,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
READ FORMAT(4I7,3F10.4,2I71
READ WRITE(6,106)NPAGE,NLNE,HONGS
READ FORMAT(//5X,'OUTINF:',//2I10//80A1)
READ WRITE(6,107)IA,IC,IS,ALG,BATCH
READ FORMAT(*'/5X,'/ADS://3110,5X,8A1,5X,LL)
READ CALL EXIT
READ RETURN
READ END
READ SUBROUTINE ORDER(CNN,N1,CPONT,NXTR,RPONT,NXTC,NO,TEMP,NPQ,NPV
        *+TYPE)
READ COMMON /ININF//11,12,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
READ INTEGER SHFTL,FIRST,PP,JC,SHFTL
READ INTEGER CNM(11),ONE/280C0C000/
INTEGER*2 TEMP(I),NO(I),TYPE(I)
INTEGER*2 CPONT(I),NXTR(I),RPONT(I),NXTC(I)

C AT RETURN THE "NO" IS CHANGED TO INDICATE THE OPTIMAL
C ORDERING OF THE NODES. ALSO NO(IEND+1)=0
NPQ=0
NPV=0
JJ=1

C TO FORM CONNECTION MATRIX:
C ROWS OF "V" ARE IN COLUMNS OF "CONN"
DO 100 I=1,IEND
DO 10 J=1,N1
CCNN(J,I)=0

C DIAGONAL ELEMENT:
TEMPl=I-1/32+1
K=I-J*32+31
L=SHFTR(ONE,K)
CCNN(J,I)=L
FIRST=I*8-7
PP=NXTR(FIRST)

20 IF(PP.EQ.FIRST)GO TO 50
CCL=CPONT(PP)
IF(CCL.EQ.0)GO TO 30
TEMP(I)=TEMP(I)+1
J=(CCL-1)/32+1
K=CCL-J*32+31
L=SHFTR(ONE,K)
CCNN(J,I)=LCCNN(J,I).L
PP=NXTR(PP)
GO TO 20

50 PP=NXTC(FIRST)
70 IF(PP.EQ.0)GO TO 100
COL=RPCNT(PP)
IF(CCL.EQ.0)GO TO 80
TEMP(I)=TEMP(I)+1
J=(CCL-1)/32+1
K=CCL-J*32+31
L=SHFTR(ONE,K)
CCNN(J,I)=LCCNN(J,I).L
PP=NXTC(PP)
GO TO 70

100 CONTINUE
C TO FIND MINIMUM COUNT OF ROWS NOT YET PROCESSED
C (AT PROCESSED ROWS TEMP=0):
MIN=IEND+1
INDEX=0
DO 200 I=1,IEND
IF(TEMP(I).EQ.0)GO TO 200
IF(TEMP(I).GE.MIN)GO TO 200
MIN=TEMP(I)
INDEX=I
200 CONTINUE
IF(INDEX.EQ.0)GO TO 400
NC(JJ)=INDEX
IF(TYPE(INDEX).EQ.1)GO TO 210
NFV=NPV+MIN
GO TO 220

210 NPQ=NPQ+MIN
220 IF(JJ.EQ.IEND)GO TO 400
122.

123.  

780  JJ=JJ+1
781  TEMP(INDEX)=O
782  C  TO CHANGE THE OTHER ROWS & FIND NEW COUNTS:
783  DO 300 1=1,IEND
784  IF(TEMP(I) .EQ. 0)  GO TO 300
785  J=1
786  K=INDEX=1
787  230  IF(1 .LE. 31)  GO TO 240
788  K=K-32
789  J=J+1
790  GC TO 230
791  C  CHECK TO SEE IF ROW "I" HAS AN ELEMENT IN POSITION COL=INDEX
792  240  L=SHFTL(CCNN(J,J),K)
793  IF(L .GE. O)  GO TO 300
794  GC 250  M=1,N1
795  250  CCNN(M,I)=LCR(CCCN(M,I),CONN(M,INDEX))
796  L=SHFTR(CNE,K)
797  CCNN(I,J)=LCR(CONN(J,  I),L)
798  TFMP(I)=NAMEER(CCNN(1,  I),IEND)
799  300  CONTINUE
800  GC TO 150
801  8CC  RETURN
802  END
803  SUBROUTINE NEWCN(V,U,ANG,NO,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,
804  *NXTC,*)
805  EXTERNAL ORDER,JCBN,BACK,CHANGE,DRAW
806  COMPLEX Y(1),U(1),SCHE L(1)
807  INTEGER N1,NXTR,RPONT(1,NXTR)
808  REAL U(1),ANG(1)
809  LOGICAL BATCH,ALG(8),HDNGS(80),PLOT,PRINT
810  COMMON/INFN/II,12,13,ITMAX,TOLERN,ZERO,BMVALINE,LIST,IEND
811  COMMON/OUTINF/NPAGE,NLINE,HDNGS,NSTO
812  COMMON/TYM/RES
813  EQUIVALENCE (ABUS,11)
814  CALL TIMEO)
815  CALL PLOT=LIST .LT. 0
816  IF(PLOT)  REWIND 3
817  IF(PLOT)  CALL EMPTYF(3)
818  PRINT=ABS(LIST) .GT. 10
819  NSTEP=ABS(LIST)/10
820  N1=(IEND-1)/32+i
821  NN=N1*IEND*4
822  CALL GSPACE(ACCNN,NN)
823  CALL GSPACE(A)TEMP,IE NDA
824  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
825  CALL FSPEC(ATEMP)
826  CALL FSPEC(ATEMP)
827  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
828  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
829  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
830  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
831  NWORDS=NPQ*3-IEND*2
832  CALL GSPACE(A)JO,NWORDS*4)
833  CALL GSPACE(A)WR,NBUS*16)
834  CALL GSPACE(A)WS,NBUS*2)
835  CALL GSPACE(A)PNTR,NBUS*2)
836  IT=0
837  LIMIT=0
838  2CO  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
839  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
840  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
841  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
842  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
843  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
844  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
845  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
846  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
847  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
848  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
849  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
850  CALL CALLER(JCBN,ORDER,ACCNN,IPTR(N1),IPTR(NPQ),IPTR(NPQ),IPTR(TYPE))
*PF, APTR, IPTR(ERR1), IPTR(ERR2), IPTR(IERR2), IPTR(ERR3), &4000

123.

IERR2=NO(IERR2)

IF(PLOT) CALL FLCTR(U, ANG, ERR1, ERR2, ERR3, NO)

IF(IERR2 .LT. 0) IERR2=-IERR2

IF(U .NE. LIMIT) GO TO 300

CALL CALLER(BACK, AJQ, AJQ, APTR, IPTR(SEQ), AWR, AWR+NBUS*4, &4000)

CALL CALLER CHANGE, AWR, AWR+NBUS*4, IPTR(U), IPTR(ANG), IPTR(V), IPTR(S)

*EQ, IPTR(TYPE))

IT=IT+1

GO TO 200

ISTNO=250

CALL FSCHANGE(AJQ)

CALL FSCHANGE(AWR)

CALL FSCHANGE(AWI)

CALL FSCHANGE(APNTR)

CALL TIME(3, C, RES)

IF(.NOT. PLOT) GO TO 350

IT=IT+1

NLINE=(IEND*2+3)*IT+4

CALL GSPACE(ARRAY, NLINE)

CALL CALLER(DRAW, IPTR(1), IPTR(NC), ARRAY)

CALL FSCHANGE(ARRAY)

IT=IT-1

CALL CUTKIT.ERR1, ERR2, IERR2, ERR3

IF(ISTNO .NE. 0) RETURN 1

RETURN 1

END

SUBROUTINE PRINTR(IT, V, U, ANG, ERR1, ERR2, IERR2, ERR3, &4000)

COMMCN /ININF/ I1, I2, I3, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND

COMMCN / CUTINF/ NPAGE, NLINE, HDNGS, ISTNO

COMMCN / ADS/ IA, ID, IS, ALG, BATCH

LOGICAL*1 ALG(8), BATCH

LOGICAL EQUC

REAL U(1), ANG(1)

LOGICAL*1 HDNGS(80)

CCOMPLEX V(1)

INTEGER*2 NC(1)

IF(IT .EQ. 0 .OR. NLINE .GE. 40) CALL HEAD

IF(IT .EQ. 1 .AND. EQUC(ALG(I1), 'L')) CALL HEAD

WRITE(6, 1) IT


**15X, 'ANG')

NLINE=NLINE+5

IF(NO(I) .GT. IEND) GO TO 100

K=-NO(I)

ANG1=ANG(I)*180./3.141592

WRITE(6, 2)*K, 'U', 'ANG1

WRITE(6, 3)*2 FORMAT(15X, 14, 5X, F8.5, 4X, F8.5, 3X, F9.4, 5X, F10.5)

NLINE=NLINE+1

CONTINUE

WRITE(6, 101) ERR1, ERR2, IERR2, ERR3

FORMAT(15X, 'ERR1=', F10.6, 5X, 'ERR2=', F10.6, 5X, 'IERR2=', I4, 5X, 'IEPR2=', 'EPR1', F10.6)

NLINE=NLINE+2
SUBROUTINE BACK(CR, CI, POINTR, SEO, DV, DANG, \*)
COMMON /ININF/ll, 12, 13, ITMAX, TCLERN, ZERO, AMVA, LINE, LIST, IEND
COMMON /OUTINF/NPAGE, NLIN, HDNGS, ISTNO
LOGICAL HDNGS(80)
EQUIVALENCE (NBUS, II)
REAL CR(1), CV(1), DANG(I)
INTEGER CI(1), FRCM, TG, COL
INTEGER*2 POINTR(1), SEO(1)
DC ICO I=1, IEND
RCW1 = 0.
RCW2 = 0.0
J = IEND - I + 1
JP1 = J + 1
FROM = POINTR(J)
TO = POINTR(JP1)
IF (TO .LT. 0) TO = -TO
IF (FRCM .LT. 0) FRCM = 0 TO 500
INDEX = FROM - 3
50 IF (INDEX .EQ. TO) GO TO 200
CCL = CI(INDEX)
IF (CCL .LT. 0) CCL = -CCL
DO 100 K = JP1, IEND
100 IF (SEO(K) .EQ. COL) GO TO 150
INDEX = INDEX + 1
ROW1 = ROW1 - CR(INDEX) * DANG(K)
INDEX = INDEX + 1
ROW2 = ROW2 + CR(INDEX) * DANG(K)
INDEX = INDEX + 1
IF (CCL .LT. 0) GO TO 50
RCW1 = ROW1 + CR(FRCM+2) * X2
GO TO 1000
500 FROM = FROM
INDEX = FROM + 1
150 COL = CI(INDEX)
INDEX = INDEX + 1
ROW1 = ROW1 + CR(INDEX) * DANG(K)
INDEX = INDEX + 1
ROW2 = ROW2 + CR(INDEX) * DANG(K)
INDEX = INDEX + 1
IF (CCL .LT. 0) GO TO 550
CCL = CI(INDEX)
INDEX = INDEX + 1
IF (CCL .LT. 0) CCL = -CCL
DO 500 K = JP1, IEND
INDEX = INDEX + 1
GO TO 500
550 IF (INDEX .EQ. TO) GO TO 700
CCL = CI(INDEX)
INDEX = INDEX + 1
IF (CCL .LT. 0) CCL = -CCL
DO 600 K = JP1, IEND
INDEX = INDEX + 1
GO TO 600
INDEX = INDEX + 1
GO TO 550
RETURN
SUBROUTINE CHANGE(DV, DANG, U, ANG, V, SEQ, TYPE)
COMM CN /ININF/ 11, 12, 13, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
EQUIVALENCE (NBUS, I)
REAL*4 DV(1), CAAC(1), U(1), ANG(1)
COMPLEX V(1)
INTEGER*2 SEC(1), TYPE(1)
DC 100 I = 1, IEND
J = SEC(I)
ANG(J) = ANG(J) + DANG(I)
IF(TYPE(J) .EQ. 1) U(J) = U(J) + DV(I) * U(J)
A1 = U(J) * COS(ANG(J))
A2 = U(J) * SIN(ANG(J))
V(J) = CMPLX(A1, A2)
CONTINUE
RETURN
END
SUBROUTINE WRKR(Y, V, TYPE, NXT, PONT, hRR, WRI)
COMM CN /PINF/ IK, FIRST, ROW
COMM CN /1NINF/ I1, I2, I3, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
INTEGER*2 TYPE(1), NXT(1), PCNT(1), WRI(1)
INTEGER FIRST, PP, VP, ROW
COMPLEX Y(1), V(1), IJK, IK
REAL WRR(4, 1), NKM
PP = NXT(FIRST)
IF(PP .EQ. FIRST) GO TO 300
J = PONT(PP)
WRI(K) = J
IF(TYPE(J) .EQ. 2) WRI(K) = -J
VF = (PP*7)/4
IJK = V(J) * Y(VP)
IK = IK + IJK
IF(J .GT. IEND) GO TO 200
IJK = V(ROW) * CCNJG(IJK)
HKM = A IMAG(IJK)
NKM = REAL(IJK)
WRR(1, K) = HKM
WRR(2, K) = -NKM
WRR(3, K) = NKM
WRR(4, K) = HKM
K = K + 1
PP = NXT(PP)
GO TO 50
RETURN
END
SUBROUTINE JCBN(WRR, WR1, QR, Q1, SEQ, Y, CPONT, NXTR, RPONT, NXTC, X)
COMM CN / ININF/ 11, 12, 13, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
INTEGER*2 SEC(1), PCINTP(1), TYPE(I)
INTEGER*2 SEQ(1), PINTP(1), TYPE(1)
COMPLEX Y(I), V(I), IK, SK, DS, TEMP, SCHED(I)
INTEGER RCW, FIRST, CIAG, OII(1), CCL
COMMON /ININF/ 11, 12, 13, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
COMMON /OUTINF/ NPAGE, NLINE, HONGS, ISTD
END
1020 LOGICAL*1 TYPE2, TYPE1, HDNGS(80)
1021 COMMON /PINF/IK,K,FIRST,ROW
1022 EQUIVALENCE (II,NBUS)
1023 C CAUTION! BE VERY CAREFUL WITH THIS PROGRAM.
1024 C IT WORKS! AND IT HAS TAKEN A LOT OF TIME TO DO SO.
1025 C SO, IF SOMETHING DOES NOT LOOK RIGHT IN THE FIRST GLANCE
1026 C DO NOT CHANGE IT QUICKLY. READ THE WHOLE PROGRAM AND THINK AGAIN.
1027 C
1028 C ROUTINE TC FOR TRIGONOMETRIC JACOBIAN.
1029 C WRR IS REAL-PART WORKING ROW; CONTAINS H,J,N,L.
1030 C WR1 IS INTEGER-PART WORKING ROW: CONTAINS COL.
1031 C Q1 & Q2 ARE EQUIVALENT (JACOBIAN MATRIX COMPACT FORM).
1032 C
1033 C DVI=C.
1034 DV2=C.
1035 ERR2=0.
1036 ERR3=0.
1037 PCINTR(1)=1
1038 DO 1000 I=1,IEND
1039 RWR=SEQ(I)
1040 IK=0.0
1041 C TC Form WORKING ROW:
1042 WRI(1)=ROW
1043 TYPE2=TYPE(PCW) .EO. 2
1044 TYPE1=.NOT. TYPE2
1045 IF(TYPE2)WRI(1)=-ROW
1046 K=2
1047 FIRST=R0W*8-7
1048 C NOTE THE INFORMATION SHARED THROUGH /PINF/
1049 CALL WRMKR(Y,V,TYPE,NXTC,RPONT,WRR,WRI)
1050 CALL WRMKR(Y,V,TYPE,NXTR,CPONT,WRR,WRI)
1051 TEMP=Y(ROW*2)
1052 IK=IK+V(ROW)*TEMP
1053 SK=V(ROW)*CCNJG(IK)
1054 DS=SCANED(ROW)-SK
1055 CP=REAL(CS)
1056 DVI=DVI+DP
1057 ADD=ABS(DP)
1058 IF(ERR2 .GE. ADD) GO TO 50
1059 ERR2=ADD
1060 ERR2=ROW
1061 IF(TYPE2) GO TO 70
1062 DO=AIMAG(DS)
1063 DV2=DV2-DD
1064 ADD=ABS(DO)
1065 IF(ERR2 .GE. ADD) GO TO 60
1066 ERR2=ADD
1067 IF(ERR2=ROW
1068 50 ERR3=ERR3=ADD
1069 IF(TYPE2) GO TO 70
1070 DO=AIMAG(DS)
1071 DV2=DV2-DD
1072 ADD=ABS(DO)
1073 IF(ERR2 .GE. ADD) GO TO 60
1074 ERR2=ADD
1075 IF(ERR2=ROW
1076 60 ERR3=ERR3=ADD
1077 TEMP=TEMP*U(ROW)**2
1078 WRR(1,1)=-AIMAG(SK*TEMP)
1079 WRR(2,1)=REAL(SK*TEMP)
1080 WRR(3,1)=REAL(SK*TEMP)
1081 WRR(4,1)=AIMAG(SK*TEMP)
1082 K=K-1
1083 IF(I .EQ. 1) GO TO 300
1084 J=1
1085 C TC PERFORM TRIGONOMETRIC PROCESS ON THE ROW BEFORE STORING IT:
1080 90 IDONE=SEQ(J)
1081 IF(TYPE1.IDONE) .EQ. 2) IDONE=-IDONE
1082 JJ=0
1083 100 JJ=JJ+1
1084 IF(IDONE .EQ. WRRI(JJ))GO TO 500
1085 150 IF(JJ .LT. K) GO TO 100
1086 J=J+1
1087 IF(J .LT. 1) GO TO 90
1088 C ROW TRIANGULARIZED: DIVIDE THE ROW BY ITS DIAGONAL ELEMENT
1089 C AND STORE THE RESULTS:
1090 300 RATIO=1./WRRI(1,1)
1091 DC 350 J=1,K
1092 IF(WRRI(J) .EQ. 0)GO TO 350
1093 WRRI(1,J)=WRRI(1,J)*RATIO
1094 IF(WRRI(J) .GT. 0)WRRI(J,J)=WRRI(J,J)*RATIO
1095 350 CONTINUE
1096 DF=DP*RATIO
1097 IF(TYPE2)GO TO 400
1098 RATIO=-WRRI(2,1)
1099 WRRI(4,1)=WRRI(4,1)+WRRI(3,1)*RATIO
1100 RATIO2=1./WRRI(4,1)
1101 DC 360 J=1,K
1102 IF(WRRI(J) .EQ. 0)GO TO 360
1103 WRRI(2,J)=WRRI(2,J)-WRRI(1,J)*RATIO*RATIO2
1104 IF(WRRI(J) .GT. 0)WRRI(4,J)=(WRRI(4,J)+WRRI(3,J)*RATIO)*RATIO2
1105 360 CONTINUE
1106 DC=(DC+DP*RATIO)*RATIO2
1107 C ROW IS READY TO BE STORED
1108 400 INDEX=PCINTR(I)
1109 IF(TYPE2)PCINTR(I)=-INDEX
1110 J=2
1111 OR(INDEX)=DP
1112 INDEX=INDEX+1
1113 IF(TYPE2)GO TO 420
1114 OR(INDEX)=DC
1115 INDEX=INDEX+1
1116 CR(INDEX)=WRRI(3,1)
1117 INDEX=INDEX+1
1118 420 IF(J .GT. K) GO TO 460
1119 JJ=WRRI(J)
1120 IF(JJ .EQ. 0)GO TO 450
1121 QI(INDEX)=JJ
1122 INDEX=INDEX+1
1123 OR(INDEX)=WRRI(1,J)
1124 INDEX=INDEX+1
1125 IF(TYPE2)GO TO 430
1126 OR(INDEX)=WRRI(2,J)
1127 INDEX=INDEX+1
1128 430 IF(JJ .LT. K)GO TO 450
1129 OR(INDEX)=WRRI(3,J)
1130 INDEX=INDEX+1
1131 IF(TYPE2)GO TO 450
1132 OR(INDEX)=WRRI(4,J)
1133 INDEX=INDEX+1
1134 450 J=J+1
1135 GO TO 420
1136 460 PCINTR(I+1)=INDEX
1137 GO TO 1000
1138 C TO CCMBINE A RCW WITH A PREVIOUSLY PROCESSED ROW:
1139 500 INDEX=PCINTR(JJ)
IF(INDEX .LT. 0) INDEX=-INDEX
LAST=POINTR(J)+1
IF(LAST .LT. 0) LAST=-LAST
RATIO=-WRR(1,JJ)

FIRST COMBINE THE PRESENT ROW WITH ONLY THE FIRST ROW OF THE
PREVIOUSLY PROCESSED ROW:
IF(TYPE1) RATIO2=-WRR(2,JJ)
DP=DP+QR(INDEX)*RATIO
IF(TYPE1) DC=CC+QR(INDEX)*RATIO2

INDEX=INDEX+1
WRR(3,JJ)=WRR(3,JJ)+QR(INDEX)*RATIO
IF(TYPE1) WRR(4,JJ)=WRR(4,JJ)+QR(INDEX)*RATIO2
INDEX=INDEX+1
CCL=QI(INDEX)
INDEX=INDEX+1
C
SEE IF THE CORRESPONDING OFF DIAGONAL ELEMENT EXISTS:
DO 550 NA=1,K
IF(CCL .EQ. WRI(NA)) GO TO 600
CONTINUE
55C
56C
WRR(NB,K)=0
NA=K
CHANGE THE VALUE OF THE OFF DIAGONAL ELEMENT (THAT CERTAINLY
EXISTS NOW):
WRR(1,NA)=WRR(1,NA)+QR(INDEX)*RATIO
IF(TYPE1) WRR(2,NA)=WRR(2,NA)+QR(INDEX)*RATIO2
INDEX=INDEX+1
IF(WRI(JJ) .GT. 0) INDEX=INDEX+1
IF(CCL .EQ. WRI(NA)) GO TO 700
WRR(3,NA)=WRR(3,NA)+QR(INDEX)*RATIO
IF(TYPE1) WRR(4,NA)=WRR(4,NA)+QR(INDEX)*RATIO2
INDEX=INDEX+1
IF(WRI(JJ) .GT. 0) INDEX=INDEX+1
IF(TYPE1) WRR(2,NA)=WRR(2,NA)+QR(INDEX)*RATIO
INDEX=INDEX+2
INDEX=INDEX+2
C
CHANGE THE SECOND ROW OF THE PREVIOUS PROC. ROW:
IF(WRI(JJ) .LT. 0) GO TO 800
RATIO=-WRR(3,JJ)
IF(TYPE1) RATIO2=-WRR(4,JJ)
INDEX=POINTR(J)
INDEX=IABS(INDEX)+1
IF(TYPE1) DC=CC+QR(INDEX)*RATIO2
INDEX=INDEX+2
CCL=CI(INDEX)
INDEX=INDEX+2
DO 740 NA=1,K
IF(CCL .EQ. WRI(NA)) GO TO 760
CONTINUE
74C
75C
76C
IF(TYPE1) WRR(1,NA)=WRR(1,NA)+QR(INDEX)*RATIO
IF(TYPE1) WRR(2,NA)=WRR(2,NA)+QR(INDEX)*RATIO2
INDEX=INDEX+1
IF(CCL .LT. 0) GO TO 780
INDEX=INDEX+1
1200 WRRI[3,NAL=WRR(3,NA)*CR(INDEX)*RATIO
1201 IF(TYPEJROW).EQ.1)WRRI(4,NA)=WRRI(4,NA)*OR(INDEX)*RATIOO
1202 INDEX=INDEX+1
1203 780 IF(INDEX.LT.LAST)GO TO 730
1204 800 WRRI(JJ)=0
1205 CONTINUE
1206 ERR1=ERROR(DV1+DV1+DV2+DV2)
1207 RETURN
1208 END
1209 SUBROUTINE OUT1(IT,ERR1,ERR2,1ERR2,ERR3)
1210 COMMON/ININF/11,12,13,ITMAX,T0LERN,ZERO,BMVA,LINE,LIST,IEND
1211 COMMON /OUTINF/ NPAGE,NLINE,HONGS,ISTNO
1212 COMMON /TYM/ RES
1213 INTEGER RES(2)
1214 LOGICAL HMsg(80)
1215 CALL READ
1216 WRITE(6,1111,12,13,LINE
1217 FORMAT(/" TABLE OF PARAMETERS : •/1X,38(•+'),4(/),5
1218 •NC. OF BUSES=', 14,1SX,\'N0. CF TRANSMISSION LINES=• , 15///5X, • NO.
1219 • CF TRANSFORMERS=•,15,1OX, • TOTAL NUMBER OF BRANCHES=•, 15///)
1220 WRITE(6,2)BMVA,LIST
1221 2 FORMAT(5X,'BASE MVA SPECIFIED=',F7.2,10X,'LIST=',14//)
1222 WRITE(6,3)IT,ITMAX,ERR1,T0LERN,ERR2,1ERR2,ERR3
1223 3 FORMAT(5X,'NC. OF ITERATIONS=',16///5X,'MAX NO. OF ITERATIONS=',16
1224 • OF POWER MISMATCHES=',F10.6///5X,'TOLERANCE SPECIFIED=',
1225 • AT BUS ',14///5X,'TOL
1226 • ABS POWER MISMATCHES=',F10.6)
1227 WRITE(6,4)RES
1228 4 FORMAT(//5X,'CPU TIME USED IN PR OCESS=',16///,5X, ' ELAPSED TIMF IN PR
1229 • OSS=',16///)
1230 IF(LIST.LT.0) CALL PLCTND
1231 RETURN
1232 END
1233 END
1234 SUBROUTINE FLOWS(V,U,ANG,NC)
1235 EXTERNAL OUTPUT
1236 COMMON /ININF/11,1Z,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
1237 COMMON /OUTINF/ NPAGE,NLINE,HONGS,ISTNO
1238 EQUIVALENCE (11,NBUS)
1239 LOGICAL HMsg(80)
1240 COMMON /V(1)
1241 REAL U(1),ANG(1)
1242 INTEGER NO(1)
1243 INTEGER FT
1244 CALL GSPACE(LTNO,LINE*2)
1245 CALL GSPACE(NAME,NBUS*18)
1246 CALL GSPACE(FT,LINE*4)
1247 CALL GSPACE(IMP,LINE*24)
1248 CALL GSPACE(EASE,NBUS*4)
1249 CALL CALLER(OUTPUT,IPTR(V),IPTR(U),IPTR(ANG),NAME,IPTR(NO),FT,FT+L
1250 • INE*2,IMP,M*LINE*8,IMP+LINE*16,BASE,LTNO)
1251 CALL FSPACE(NAME)
1252 CALL FSPACE(FT)
1253 CALL FSPACE(IMP)
1254 CALL FSPACE(EASE)
1255 CALL FSPACE(LTNO)
1256 RETURN
1257 END
1258 SUBROUTINE DLTPUT(V,U,ANG,NAME,NC,FPCH,TO,YP1,YP2,Z,BASE,LTNO)
1259 COMMON/ININF/11,12,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
1260      COMMON /CUTINF/ NPAGE,NLINE,NHNGS,INSTNO
1261      COMMON /ADS/IA,IG,IS,ALG,BATCH
1262      EQUIVALENCE (11,NBUS)
1263      LOGICAL*1 HDNGS(80),ALG(8),BATCH
1264      LOGICAL*1 TEST(8)/'F','L','O','W','S','T','L','T'/
1265      COMPLEX V(11),Z(1),YP1(1),YP2(1),S,ST,YPI
1266      REAL U(1),ANC(1),FILLER(4),BASE(1)
1267      INTEGER*2 NC(11),TYPE,FRCM(11),TO(1),LTNO(1)
1268      REAL*8 NAME(1),LTNAME
1269      INTEGER ITES(2),IALG(2)
1270      EQUIVALENCE (ALG(1),IALG(1)),(TEST(1),ITEST(1))
1271      READ 1
1272      REWIND 1
1273      IEND=NBUS
1274      PI=3.14159
1275      I=1
1276      100 READ(1)TYPE,NO(I),NAME(I),FILLER,BASE(I),VOLT
1277      IF(TYPE .NE. 3) GO TO 200
1278      NC(IEND)=NO(I)
1279      NAME(IEND)=NAME(I)
1280      BASE(IEND)=BASE(I)
1281      IEND=IEND-1
1282      I=I+1
1283      200 READ(4)LTNO(I),LTNAME,FROM(I),TO(I),YP1(I),YP2(I),Z(I)
1284      CALL HEAD
1285      DO 1000 I=1,NBUS
1286      K=NC(I)
1287      IC=NLINE+6
1288      DC 400 J=1,LINE
1289      IF(K .LE. FPCM(J)) GO TO 500
1290      IF(K .EQ. TC(J)) GO TO 700
1291      KK=FPCM(J)
1292      YPI=YP1(J)
1293      550 CONTINUE
1294      FORMAT(/10X,BUS NO*,15,2X,A8,4X,F5.3,* PU*,4X,F8.3,* KV*,4X,F8.3*
1295      *,' Deg*/30X,' TO:*')
1296      NLINE=NLINE+4
1297      DD 700 J=1,LINE
1298      IF(K .NE. TC(J)) GO TO 500
1299      IF(K .GE. FCM(J)) GO TO 700
1300      KK=FCM(J)
1301      YPI=YP2(J)
1302      500 GO TO 550
1303      KK=TC(J)
1304      550 IF(EC .GE. 60) CALL HEAD
1305      ST=0.
1306      VCLT=U(1)*BASE(I)
1307      DEG=ANG(11)*180./PI
1308      WRITE(16,401)*YPI,KK,NAME(I),U(1),VCLT,DEG
1309      401 FORMAT(/10X,BUS NO*,15,2X,A8,4X,F5.3,* PU*,4X,F8.3,* KV*,4X,F8.3*
1310      *,' Deg*/30X,' TO:*')
1311      500
1312      600 CONTINUE
1313      650 CONTINUE
1314      500 KK=TC(J)
1315      IF(KK .EQ. 650) GO TO 650
1316      IF(KK .EQ. NC(I1)) GO TO 650
1317      WRITE(6,651)*YPI,KK,NAME(I1),S,LTNO(I1)
1318      651 FORMAT(5X,15,2X,A8,210X,F10.3,4X,'(''#14,')')
1320 7CO CONTINUE
1321 WRITE(6,7011)ST
1322 701 FORMAT(38X,52('1'1/35X,'TOTALS =',3X,21G1X,F10.3))
1323 NLIN=NLIN+2
1324 1000 CONTINUE
1325 IF(IAGL(1) .EQ. ITEST(1) .AND. IAGL(2) .EQ. ITEST(2)) STOP
1326 RETURN
1327 END
1328 SUBROUTINE LOSS1(V,U,ANG,NC,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,NXTC)
1329 **
1330 EXTERNAL CRCEP,MCCEF,FCMMP,FCRKB,LCMGG,DRAW,ERSLC
1331 COMPLEX Y, NC(1),SCHED(1)
1332 INTEGER Y(1),V(1),SCHED(1)
1333 INTEGER*2 CPONT(1),NXTR(1),RPONT(1),NXTC(1),NO(1),TYPE(1)
1334 REAL U(1),ANG(1)
1335 LOGICAL=1 BATCH,ALG(8),HDONGS(80),PLCT,PRINT
1336 COMMON /ININF/ 11,12,13,ITMAX,TOLERN,ZERO,BMVA,LLINE,LIST,IEND
1337 COMMON /ADS/ IA,IC,IS,ALG,BATCH
1338 COMMON /OUTINF/ NPAGE,NLINE,HDONGS,IISTNO
1339 COMMON /TYM/ RES
1340 COMMON /LOSSES/ SL
1341 EQUIVALENCE (IA,NBUS)
1342 IA=1
1343 GC TC 10
1344 ENTRY FLOSS1(V,U,ANG,NO,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,NXTC,*)
1345 CALL GSPACE(SL,NBUS*8)
1346 IA=2
1347 1C CALL TIME(0)
1348 1C PLOT=LIST .LT. 0
1349 IF(PLCT) REWIND 3
1350 IF(IPLCT) CALL EMPTYF(3)
1351 PRINT=IARS(LIST) .GT. 10
1352 NSTEF=IABS(LIST)/10
1353 N1=(IEND-11)/32+1
1354 ALIGN=N1*1END+4
1355 CALL GSPACE(ACCNN,N1)
1356 CALL GSPACE(ATEMP,IEND+2)
1357 CALL CALLER(CRCEP,MCCEF,AWR,AI,AM,BV,IPTR(SEQ),IPTR(Y),IPTR(CPONT),1
1358 *IPTR(NXTR),IPTR(RPONT),IPTR(NXTC),IPTR(SCHED),IPTR(U),IPTR(ANG1,IPTR(SCHED,1
1359 ,SL,IPTR(Y),IPTR(CPONT),1
1360 APTRPRESSURE))
1361 CALL FSPACE(AWRI
1362 CALL FSPACE(AWI)
1363 CALL FSPACE(AWR)
1364 CALL FSPACE(AW)
1365 CALL FSPACE(NBUS*4)
1366 CALL FSPACE(NBUS*8)
1367 CALL FSPACE(NBUS*16)
1368 CALL FSPACE(NBUS*32)
1369 CALL FSPACE(NBUS*64)
1370 CALL FSPACE(NBUS*128)
1371 CALL FSPACE(NBUS*256)
1372 CALL FSPACE(NBUS*512)
1373 CALL FSPACE(NBUS*1024)
1374 CALL FSPACE(NBUS*2048)
1375 IT=0
1376 IF(IT .EQ. 1) GO TO 50
1377 CALL CALLER(ERSL, BP,NUMB,IPTR(SEQ),IPTR(TYPE),IPTR(V),IPTR(U),IP
1378 *TR(ANG),IPTR(SCHED),IPTR(SEQ),SL,IPTR(Y),IPTR(CPONT),IPTR(NXTR),IP
1379 *TR(RPONT),IPTR(NXTC),IPTR(ERR1),IPTR(ERR2),IPTR(ERR3))
1380 IERR2=NO(IERP2)
1381 IF(IERR2 .LT. 0) IERR2=-IERR2
1382 IF(PLCT) CALL PLOTRA(U,ANG,ERR1,ERR2,ERR3,NO)
1383 IF(.NOT. PRINT) GO TO 20
1384 LIMIT=NSTEP
1385 CALL PRINTR(IT,V,U,ANG,ERR1,ERR2,ERR3,NO)
1386 20 IF(IERR2 .LE. TOLERN) GO TO 300
1387 GO TO 100
1388 LIMIT=1
1389 100 IF(PLCT) CALL PLOTRA(U,ANG,ERR1,ERR2,ERR3,NO)
1390 IF(.NOT. PRINT .OR. IT .NE. LIMIT) GO TO 20C
1391 LIMIT=LIMIT + NSTEP
1392 CALL PRINTR(IT,V,U,ANG,ERR1,ERR2,ERR3,NO)
1393 20C IF(ERR2 .LE. TOLERN) GO TO 300
1394 IF( IT .GE. UMAX) GO TO 250
1395 GO TO 100
1396 25C ISTNO=250
1397 300 CALL FSPACE(AM)
1398 CALL FSPACE(IPNTR)
1399 CALL FSPACE(BV)
1400 CALL FSPACE(BP)
1401 CALL TIME(IT,RES)
1402 IF(.NOT. PLCT) GO TO 350
1403 IF(IA .EQ. 2) IT=IT+1
1404 NN=IEND*2+3*IT*4
1405 CALL GSPACE(ARRAY,NN)
1406 CALL CALLER(DRAW,IT,ARRAY)
1407 CALL FSPACE(ARRAY)
1408 IF(IA .EQ. 2) IT=IT-1
1409 350 CALL CUT1(IT,ERR1,ERR2,ERR3)
1410 IF(ISTNO .NE. 0) RETURN 1
1411 RETURN
1412 RETURN 1
1413 END
1414 SUBROUTINE LCHNGE(VP,NUMB,SEC,TYPE,V,U,ANG,SCHED,IT,SL,Y,CPONT,NXT)
1415 REAL VP(I),U(I),ANG(I)
1416 INTEGER*8 NUMB(I),CPONT(I),NXTR(I),RPOPT(I),NXTC(I),SEQ(I),TYPE(I)
1417 COMPLEX Y(I),V(I),SL(I),SCHED(I),IK,DV,SLK,DS
1418 INTEGER FIRST,PP,VP
1419 COMMON /ININF/ Y,I,J,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
1420 DO 1=1,IEND
1421 K=SEQ(I)
1422 IF(TYPE(K) .EQ. 1) U(1)=SORT(AMB+1)*Z,
1423 IF(TYPE(K) .EQ. 2) U(1)=SORT(AMB+1)*Z,
1424 RETURN
1425 END
1426 SUBROUTINE LCHNGE(VP,NUMB,SEC,TYPE,V,U,ANG,SCHED,IT,SL,Y,CPONT,NXT)
1427 REAL VP(I),U(I),ANG(I)
1428 INTEGER*2 NUMB(I),CPONT(I),NXTR(I),RPOPT(I),NXTC(I),SEQ(I),TYPE(I)
1429 COMPLEX Y(I),V(I),SL(I),SCHED(I),IK,DV,SLK,DS
1430 INTEGER FIRST,PP,VP
1431 COMMON /ININF/ Y,I,J,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
1432 DO 1=1,IEND
1433 K=SEQ(I)
1434 IF(TYPE(K) .EQ. 1) U(1)=SORT(AMB+1)*Z,
1435 IF(TYPE(K) .EQ. 2) U(1)=SORT(AMB+1)*Z,
V (K) = CMPLX( A1, A2)

CONTINUE
ENTRY ERSLCIB, NUMB, SEG, TYPE, V, U, ANG, SCHED, IT, SL, Y, CPONT, NXTR,
+ RPONT, NXTC, ERR1, ERR2, IERR2, ERR3)
DV1 = 0.
DV2 = 0.
ERR2 = 0.
ERR3 = 0.
DO 200 I = 1, IEND
IK = 0.
SLK = 0.
FIRST = I * 8 - 7
PP = NXTR (FIRST)

110 IF (PP * EQ. FIRST) GO TO 120
VP = (PP + 7) / 4
JP = PCNT (PP)
IK = IK + Y(VP) * V(J)
DV = V(I) - V(J)
SLK = SLK - DV * CONJG (DV * Y(VP))
IF (PP = NXTR (PP))
GO TO 110

120 PP = NXTC (FIRST)
IF (PP * EQ. FIRST) GO TO 150
VP = (PP - 7) / 4
JP = PCNT (PP)
IK = IK + Y(VP) * V(J)
DV = V(I) - V(J)
SLK = SLK - DV * CONJG (DV * Y(VP))
IF (PP = NXTC (PP))
GO TO 130

150 SL (I) = SLK / 2.
IK = IK + Y(I + 2) * V(I)
IK = V(I) * CONJG (IK)
DS = SCHED (I) - IK
DP = REAL (DS)
DV1 = DV1 + DP
ACD = ABS (DP)
IF (ERR2 .GE. ADD) GO TO 160
ERR2 = ADD
IERR2 = I

160 ERR3 = ERR3 + ACD
IF (TYPE(I) .EQ. 2) GO TO 200
DG = AIMAG (DS)
DV2 = DV2 + DG
ACD = ABS (DG)
IF (ERR2 .GE. ADD) GO TO 170
ERR2 = ADD
IERR2 = I
CONTINUE

200 ERR3 = ERR3 + ACD
CONTINUE
ERR1 = SQRT (DV1**2 + DV2**2)
RETURN
PP = NXT(FIRST)
1501 50 IF(PP .EQ. FIRST) GO TO 200
1502 J = PCNT(PP)
1503 WRI(K) = J
1504 IF(TYPE(J) .EQ. 2) WRI(K) = -J
1505 VP = ((PP+7)/4)
1506 SUM = SUM + (VP)
1507 GIK = REAL(Y(VP))
1508 BIK = IMAG(Y(VP))
1509 IF(J .GT. I) GO TO 110
1510 WRR(1, K) = BIK
1511 WRR(2, K) = GIK
1512 IF(TYPE(J) .EQ. 2) GO TO 100
1513 WRR(3, K) = GIK
1514 WFR(4, K) = BIK
1515 K = K + 1
1516 GO TO 150
1517 100 K = K + 1
1518 11C UI = U(J) * 2 / 2.
1519 BI(NB) = BI(NB) - GIK * UI
1520 I = (FIRST + 7) / 8
1521 IF(TYPE(I) .EQ. 2) GO TO 150
1522 BI(NB+1) = BI(NB+1) + BIK * UI
1523 15C PP = NXT(PP)
1524 GO TO 50
1525 200 RETURN
1526 END
1527 SUBROUTINE LNFS(ND, SL)
1528 INTEGER*2 NC(1)
1529 COMPLEX SLID, TEMP
1530 CCINC, /INC/ 11, 12, 13, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
1531 EQUIVALENCE (II, NBUS)
1532 CCINC, /ADS/ IA, ID, IS, ALG, BATCH
1533 LOGICAL*1 BATCH, ALG(8), MTSU(30)
1534 DATA NRTST /404040/
1535 EQUIVALENCE (MTSU, NRTST)
1536 IF(BATCH) GO TO 100
1537 WRITE(16,1)
1538 1 FORMAT(* ENTER NAME OF THE UNIT CONTAINING STARTING*/
1539 ** VALUES (RETURN OR EOF IF *MSOURCE=*))
1540 CALL SETPFX(*,*)
1541 100 READ(15, 101). END = 200 (MTSU(11) I = 13, 30)
1542 101 FORMAT(20A1)
1543 IF(NRTST .EC. NRTST) GO TO 200
1544 CALL MOVECIL, *ASSIGN 8 = MP, MTSU)
1545 CALL FTNCD(MTSU, 30)
1546 GO TO 210
1547 200 CALL FTNCD(8 = GUSR, 14)
1548 IF(BATCH) GO TO 210
1549 WRITE(16,201)
1550 201 FORMAT(* ENTER P,U. VALUES OF **NO1**, **NO2**, **SL**/
1551 ** IN (215, 2F10.5) FORMAT*)
1552 210 DC 220 I = 1, NELS
1553 220 SL(I) = 0
1554 250 DC 290 I = 1, LINE
1555 READ(8, 251). END = 3CO) NO1, NO2, TEMP
1556 251 FORMAT(215, 2F10.5)
1557 DC 280 J = 1, NBUS
1558 280 1 = 1, NBUS
1559 K = NO1
1560 IF(K .LT. 0) K = -K
1560 IF(K.EQ. NO1) NO1=-J
1561 IF(K.EQ. NO2) NO2=-J
1562 IF(NC1 .LT. 0 .AND. NO2 .LT. 0) GO TO 285
1563 CONTINUE
1564 285 NO1=-NO1
1565 NO2=-NO2
1566 SL(NO1)=SL(NC1)+TEMP/2.
1567 SL(NC2)=SL(NC2)+TEMP/2.
1568 CONTINUE
1569 300 CALL SETPFX(*&*,1)
1570 RETURN
1571 END
1572 SUBROUTINE FORBKW(AM, MA, BP, PCINTR, NUMB, SEQ)
1573 REAL AM(1), BP(1)
1574 INTEGER MA(1)
1575 COMMON /ININF/ I1, I2, I3, ITMAX, TOLERN, ZERO, BMVA, LINE, LIST, IEND
1576 EQUIVALENCE (I1,NBUS)
1577 INTEGER*2 PCINTR(1), NUMB(1), SEQ(1)
1578 DC 500 I=1, IEND
1579 K=SEQ(I)
1580 NB=NUMB(K)
1581 J=PCINTR(I)
1582 LAST=PCINTR(I-1)
1583 IF(LAST .LT. 0) LAST=-LAST
1584 IF(J .LT. 0) J=-J
1585 IF(PCINTR(I) .LT. 0) GO TO 200
1586 BP(NB+1)=BP(NB+1)-BP(NB)*AM(J+2)
1587 1CC J=J-3
1588 IF(J .EQ. LAST) GO TO 300
1589 K=IAES(MA(I))
1590 KB=NUMB(K)
1591 BP(KB)=BP(KB)-BP(NB)*AM(J+1)-BP(NB+1)*AM(J+2)
1592 IF(MA(J) .LT. 0) GO TO 100
1593 KB=KB+1
1594 J=J+2
1595 BP(KB)=BP(KB)-BP(NB)*AM(J+1)-BP(NB+1)*AM(J+2)
1596 GO TO 100
1597 200 J=J+1
1598 IF(J .EQ. LAST) GO TO 300
1599 K=IAES(MA(I))
1600 KB=NUMB(K)
1601 J=J-1
1602 BP(KB)=BP(KB)-BP(NB)*AM(J)
1603 IF(MA(J-1) .LT. 0) GO TO 200
1604 KB=KB+1
1605 J=J+1
1606 BP(KB)=BP(KB)-BP(NB)*AM(J)
1607 GO TO 200
1608 300 J=PCINTR(I)
1609 IF(J .LT. 0) J=-J
1610 BP(NB)=BP(NB)*AM(J)
1611 IF(PCINTR(I) .GT. 0) BP(NB+1)=BP(NB+1)*AM(J+1)
1612 CCCONTINUE
1613 C BACK SUBSTITUTION STARTS:
1614 DC 1000 I=1, IEND
1615 I=IEND-I+1
1616 J=PCINTR(I)
1617 IF(J .LT. 0) J=-J
1618 LAST=PCINTR(I+1)
1619 IF(LAST .LT. 0) LAST=-LAST
1620  SUM1=0.
1621  IF(PCINT1(I) .LT. 0) GO TO 700
1622  SUM2=0.
1623  600 J=J+3
1624  IF(J .EQ. LAST) GO TO 800
1625  K=IABS(MA(J))
1626  KB=NUMB(K)
1627  SUM1=SUM1+AM(J+1)*BP(KB)
1628  SUM2=SUM2+AM(J+2)*BP(KB)
1629  IF(MA(J) .LT. 0) GO TO 600
1630  J=J+2
1631  KB=KB+1
1632  SUM1=SUM1+AM(J+1)*BP(KB)
1633  SUM2=SUM2+AM(J+2)*BP(KB)
1634  GO TO 600
1635  700 J=J+1
1636  IF(J .EQ. LAST) GO TO 800
1637  K=IARS(MA(J))
1638  KB=NUMB(K)
1639  J=J+1
1640  SUM1=SUM1+AM(J)*BP(KB)
1641  IF(MA(J-1) .LT. 0) GO TO 700
1642  J=J+1
1643  KB=KB+1
1644  SUM1=SUM1+AM(J)*BP(KB)
1645  GO TO 700
1646  800 J=PCINT1(I)
1647  IF(J .LT. 0) J=-J
1648  K=SEQ(I)
1649  NB=NUMB(K)
1650  IF(PCINT1(I) .LT. 0) GO TO 900
1651  BP(NB+1)=BP(NB+1)-SUM2
1652  SUM1=SUM1+AM(J)*BP(KB)
1653  900 BP(NB)=BP(NB)-SUM1
1654  1000 CONTINUE
1655  RETURN
1656  END
1657  SUBROUTINE FCGBP(B,BP,SEQ,SL,IT,V,ANG,Y,CPCNT,NXTR,RPONT,NXTC
1658  * ,TYPE,U)
1659  INTEGER FIRST,PP,VP
1660  REAL B(1),BP(1),ANG(1),U(1)
1661  INTEGER SEQ(1),CPCNT(1),NXTR(1),RPONT(1),NXTC(1),TYPE(1)
1662  COMPLEX SL(1),V(1),Y(1),SUM
1663  COMMON /IN1F/ I1,I2,I3,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
1664  COMMON /ADS/ IA,IDO,IS,ALG,BATCH
1665  LOGICAL*1 ALG(1),BATCH
1666  EQUIVALENCE (I1,NBUS)
1667  IF(I1 .EQ. 2) GO TO 100
1668  IF(IS .EQ. 2 .OR. IT .NE. 0) GO TO 100
1669  DO 50 I=1,IEND
1670  5C SL(I)=0.
1671  100 NB=0
1672  DO 200 I=1,IEND
1673  K=SEQ(I)
1674  NB=NB+1
1675  BP(NB)=BP(NB)-REAL(SL(K))
1676  IF(TYPE(K) .EQ. 2) GO TO 200
1677  NB=NB+1
1678  BP(NB)=BP(NB)-AIMAG(SL(K))
1679  200 CONTINUE
IF (IT .EQ. 0 .AND. (IA+ISI .NE. 4)) GO TO 450
NOTE THAT IT WILL NOT BRANCH IF IA=IS=2.

NB=0
DO 400 1=1,IEND
NB=NB+1
K=SEC(I)
FIRST=K*B-7
NB=NB+1
DO 400 1=1,IEND
NB=NB+1

C NOTE THAT -EPS (ACCORDING TO THE DEFINITION) IS BEING FORMED

IF(PP .EQ. FIRST) GO TO 300
J=CFCNT(PP)
VP=(PP+7)/4
DANG=ANG(K)-ANG(J)
EPS=U(K)*U(J)*SIN(DANG)-DANG
SUM=SUM+EPS*CCNJG(Y(VP))
PP=NXTTR(PP)
GO TO 250
300
PP=NXTTR(FIRST)
320
IF(PP .EQ. FIRST) GO TO 350
J=RPCNT(PP)
VP=(PP+7)/4
DANG=ANG(K)-ANG(J)
EPS=U(K)*U(J)*SIN(DANG)-DANG
SUM=SUM+EPS*CCNJG(Y(VP))
PP=NXTTR(PP)
GO TO 320
350
BP(NB)=BP(NB)+AIMAG(SUM)
IF(TYPE(K) .EQ. 2) GO TO 400
NB=NB+1
BP(NB)=BP(NB)-REAL(SUM)
CONTINUE

400
CONTINUE
450
CONTINUE
RETURN
END
1740 NUMB(ROW)=KB
1741 WRI(1)=RCW
1742 PV=TYPE(ROW) .EQ. 2
1743 PC=' .ACT. PV
1744 IF(PV) WRI(1)=-RCW
1745 K=2
1746 FIRST=ROW+8-7
1747 SUM=0.
1748 CALL LWRMKR(WRR,WRI,Y,CPONT,NXTR,U,FIRST,K,TYPE,KB,B,SUM)
1749 CALL LWRMKRURR,WRI,Y,RPONT,NXTC,U,FIRST,K,TYPE,KB,B,SUM)
1750 TEMP=Y*(ROW+2)
1751 SUM=SUM+TEMP
1752 GII=REAL(TEMP)
1753 RII=AIMAG(TEMP)
1754 GI=REAL(SUM)
1755 RI=AIMAG(SUM)
1756 WRR(1,1)=RII-BI
1757 IF(PV) GY TC 20
1758 KB=KB+1
1759 WRR(2,1)=GII-GI
1760 WRR(3,1)=GII+GI
1761 WRR(4,1)=-BII-BI
1762 GO TO 30
1763 UI=U(ROW)**2/2.
1764 B(KB)=B(KB)-(GII+GI)*UL
1765 30 K=K-1
1766 IF(1 .EQ. 1) NC TO 300
1767 C TC PERFORM TRIANGULARIZATION PROCESS ON THE ROW BEFORE STORING IT:
1768 J=1
1769 9C IDONE=SEC(J).
1770 IF(TYPE(IDONE) .EQ. 2) IDONE=-IDONE
1771 JJ=0
1772 10C JJ=JJ+1
1773 IF(IDONE .EQ. WRI(JJ))GO TO 500
1774 150 IF(JJ .LT. K) GO TO 100
1775 J=J+1
1776 IF(J .LT. 1) GC TO 90
1777 C RCW TRIANGULARIZED. DIVIDE THE ROW BY ITS DIAGONAL ELEMENT
1778 C AND STORE THE RESULTS:
1779 300 RATIO=1./WRR(1,1)
1780 DIAG1=RATIC
1781 DO 350 J=1,K
1782 IF(WRR(JJ .EQ. 0))GO TO 350
1783 WRR(1,J)=WRR(1,J)*RATIG
1784 IF(WRR(J) .GT. 0)WRR(3,J) = WRR(3,J)*RATIG
1785 CONTINUE
1786 DO 350 J=1,K
1787 IF(WRR(J) .EQ. 0)GO TO 360
1788 WRR(4,J)=WRR(4,J)+WRR(3,J)*RATIG
1789 RATIG2=1./WRR(4,J)
1790 DIAG2=RATIG2
1791 DO 360 J=1,K
1792 IF(WRR(J) .EQ. 0)GO TO 360
1793 WRR(2,J)=WRR(2,J)*WRR(1,J)*RATIG2
1794 IF(WRR(J) .GT. 0)WRR(4,J) = (WRR(4,J)+WRR(3,J)*RATIG2
1795 CONTINUE
1796 C ROW IS READY TO BE STORED
1797 40C INDEX=POINTR(I)
1798 IF(PV)POINTR(I)=-INDEX
1799 J=2
1800 AM(INDEX)=DIAG1
1801 INDEX=INDEX+1
1802 IF(PVIGO TO 420
1803 AM(INDEX)=DIAG2
1804 INDEX=INDEX+1
1805 AM(INDEX)=WRRI(3,1)
1806 INDEX=INDEX+1
1807 420 IF(JJ .GT. K) GO TO 460
1808 JJ=WRI(J)
1809 IF(JJ = EQ. 0) GO TO 450
1810 MA(INDEX)=JJ
1811 INDEX=INDEX+1
1812 AM(INDEX)=WRRI(1, JJ)
1813 INDEX=INDEX+1
1814 IF(PVIGO TO 430
1815 AM(INDEX)=WRRI(2, JJ)
1816 INDEX=INDEX+1
1817 430 IF(JJ .LT. 0) GO TO 450
1818 AM(INDEX)=WRRI(3, JJ)
1819 INDEX=INDEX+1
1820 IF(PVIGO TO 450
1821 AM(INDEX)=WRRI(4, JJ)
1822 INDEX=INDEX+1
1823 450 J=J+1
1824 GO TO 420
1825 460 PCINTR(J+1)=INDEX
1826 GC TO 1000
1827 C TO CCMBINE A RCH WITH A PREVIOUSLY PROCESSED RCH:
1828 500 INDEX=PCINTR(J)
1829 IF(INDEX .LT. 0) INDEX=-INDEX
1830 LAST=PCINTR(J+1)
1831 IF(LAST .LT. 0) LAST=-LAST
1832 RATIO=WRRI(1, JJ)
1833 C FIRST CCMBINE THE PRESENT RCH WITH ONLY THE FIRST RCH OF THE
1834 C PREVIOUSLY PROCESSED RCH:
1835 IF(PC)RATI02=-WRRI(2, JJ)
1836 520 INDEX=INDEX+1
1837 IF(WRI(JJ) .LT. 0) GO TO 540
1838 INDEX=INDEX+1
1839 WRRI(3, JJ)=WRRI(3, JJ)+AM(INDEX)*RATIO
1840 IF(PC)WRRI(4, JJ)=WRRI(4, JJ)+AM(INDEX)*RATIO2
1841 INDEX=INDEX+1
1842 540 COL=MA(INDEX)
1843 INDEX=INDEX+1
1844 C SEE IF THE CCRSPCDING OFF DIAGONAL ELEMENT EXISTS:
1845 DO 550 NA=1, K
1846 C IF NOT, CREATE ONE WITH ZERO VALUES:
1847 550 CONTINUE
1848 C IF NCT, CREATE CNE WITH ZERO VALUES:
1849 K=K+1
1850 WRRI(K)=COL
1851 DC 560 NB=1, 4
1852 560 WRRI(NB, K)=0.
1853 NA=K
1854 C CHANGE THE VALUE OF THE OFF DIAGONAL ELEMENT (THAT CERTAINLY
1855 C EXISTS NOW):
1856 600 WRRI(1, NA)=WRRI(1, NA)+AM(INDEX)*RATIO
1857 IF(PC)WRRI(2, NA)=WRRI(2, NA)+AM(INDEX)*RATIO2
1858 INDEX=INDEX+1
1859 IF(WRI(JJ) .LT. 0) INDEX=INDEX+1
140.

1860 IF (COL .LT. 0) GO TO 700
1861 WRR(3,NA) = WRR(3,NA) * AM(INDEX) * RAT10
1862 IF (PC) WRR(4,NA) = WRR(4,NA) * AM(INDEX) * RAT102
1863 INDEX = INDEX + 1
1864 IF (WRR(JJ) .LT. 0) INDEX = INDEX + 1
1865 IF (INDEX .LT. LAST) GO TO 540
1866 C IF THE PREV PROC. ROW IS PC, COMBINE THE SECOND ROW OF THAT:
1867 IF (WRR(JJ) .LT. 0) GO TO 800
1868 RAT10 = WRR(3, JJ)
1869 IF (PC) RAT102 = WRR(4, JJ)
1870 INDEX = PCINT(R(JJ))
1871 INDEX = IABS(INDEX) + 1
1872 INDEX = INDEX + 2
1873 73C CCL = MA(INDEX)
1874 INDEX = INDEX + 2
1875 DO 740 NA = 1, K
1876 IF (COL .LT. 0) GO TO 760
1877 740 CONTINUE
1878 ISTNC = 740
1879 RETURN 1
1880 760 WRR(1, NA) = WRR(1, NA) * AM(INDEX) * RAT10
1881 IF (PC) WRR(2, NA) = WRR(2, NA) * AM(INDEX) * RAT102
1882 INDEX = INDEX + 1
1883 IF (CCL .LT. 0) GO TO 780
1884 INDEX = INDEX + 1
1885 WRR(3, NA) = WRR(3, NA) * AM(INDEX) * RAT10
1886 IF (TYPE(ROW) .EQ. 1) WRR(4, NA) = WRR(4, NA) * AM(INDEX) * RAT102
1887 INDEX = INDEX + 1
1888 78C IF (INDEX .LT. LAST) GO TO 730
1889 800 WRR(JJ) = 0
1890 GO TO 150
1891 CONTINUE
1892 RETURN
1893 END

SUBROUTINE LSSIM(V, U, ANG, NO, SCHEC, TYPE, SEQ, Y, CPONT, NXTR, RPONT, NXTC

1895 *) 
1896 EXTERNAL CREFF, MOCEF, FORMBP, FORBKW, MODCHG, DRAW, MODER
1897 COMPLEX Y(1), V(1)
1898 SCHED(I)
1899 INTEGER*2 PCNT(1), NXTR(I), RPONT(I), NXTC(I), NC(I), TYPE(I)
1899 INTEGER*2 SEC(1), NSTEP
1900 INTEGER RES(2), ACCNN, ATEMP
1901 REAL U(1), ANG(1)
1902 LOGICAL*1 BATCH, ALG(I), HDGNS(80), PLOT, PRINT
1903 COMMON /INIT/ II, T2, T3, ITMAx, TOLER, BMTN, LINE, LIST, IEND
1904 COMMON /ADS/ IA, ID, IS, ALG, BATCH
1905 COMMON /OUTINF/ NPAGE, NLINE, HDGNS, ISTNO
1906 COMMON /TYM/ RES
1907 CC MCK/LCSSES/ SL
1908 EQUIVALENCE (II, ABUS)
1909 IA = 1
1910 CC TC 10
1911 ENTRY FLSSIM(V, U, ANG, NO, SCHEC, TYPE, SEQ, Y, CPONT, NXTR, RPONT, NXTC, *)
1912 IA = 2
1913 1C CALL TIME(0)
1914 PLOT = LIST .LT. 0
1915 IF (PLOT) REWIND 3
1916 IF (PLOT) CALL EMPTYF(3)
1917 PRINT = IABS(LIST) .GT. 10
1918 NSTEF = IABS(LIST) / 10
1919 NI = (IEND - 1)/32 + 1
141.

1920 NN=N1*IEND*4
1921 CALL GSPACE(ACCNN,NN)
1922 CALL GSPACE(ATEMP,IEND*2)
1923 CALL CALLER(CORDER,ACCNN,IPTR(N1),IPTR(PCONT),IPTR(NXTR),IPTR(PONT
1924 *) ,IPTR(NXTC),IPTR(SEQ),ATEMP,IPTR(NPC),IPTR(NPV),IPTR(TYPE))
1925 CALL FSPACE(ATEMP)
1926 CALL FSPACE(ACCNN)
1927 NWORDS=NPV=5+NPV*3*IEND*2
1928 CALL GSPACE(AM,NWORDS*4)
1929 CALL GSPACE(AWR,NPUS*16)
1930 CALL GSPACE(AK,NBUS*2)
1931 CALL GSPACE(ANP,NBUS*2)
1932 CALL GSPACE(EV,IEND*8)
1933 CALL GSPACE(NUMB,IEND*2)
1934 CALL CALLER(MOCEF,AWR,ANP,AM,AM,BV,IPTR(SEQ),IPTR(Y),IPTR(PCONT),I
1935 *IPTR(NXTR),IPTR(HPONT),IPTR(NXTC),IPTR(SCHED),IPTR(U),IPTR(TYPE).
1936 *APTR,NUMB,6400)
1937 CALL FSPACE(AWR)
1938 CALL FSPACE(AM)
1939 CALL FSPACE(EV)
1940 IT=0
1941 IF(ITA .EQ. 1) GO TO 50
1942 CALL CALLER(MODER,AP,NUMR*IPTR(SEQ),IPTR(TYPE),IPTR(V),IPTR(U),IP
1943 *TR(ANG),IPTR(SCHED),IPTR(TL),SL,IPTR(Y),IPTR(PCONT),IPTR(NXTR),IP
1944 *R(HPONT),IPTR(NXTC),IPTR(SCHED),IPTR(U),IPTR(TYPE),IP
1945 IERR2=NO(IERR2)
1946 IF(IERR2 .LT. 0) IERR2=-IERR2
1947 IF(PLOT) CALL PLOTRIU(ANG,ERR1,ERR2,ERR3,NO)
1948 IF(.NCT. PRINT) GO TO 20
1949 LIMIT=NSTEP
1950 CALL PRINTIT(V,U,ANG,ERR1,ERR2,ERR3,NO)
1951 20 IF(IERR2 .LE. TOLERN) GO TO 50
1952 GO TO 100
1953 50 LIMIT=1
1954 CALL CALLER(MRMBP,AM,AM,AM,AP,APTR,NUMR,IPTR(SEQ))
1955 CALL CALLER(MODCHG,AM,AM,AP,APTR,NUMR,IPTR(SEQ),IPTR(SCHED),IPTR(V),IPTR(U),IP
1956 *TR(ANG),IPTR(SCHED),IPTR(TL),SL,IPTR(Y),IPTR(PCONT),IPTR(NXTR),IP
1957 R(HPONT),IPTR(NXTC),IPTR(SCHED),IPTR(U),IPTR(TYPE),IP
1958 IERR2=NO(IERR2)
1959 IF(IERR2 .LT. 0) IERR2=-IERR2
1960 IF(.NOT. PRINT OR IT .NE. LIMIT) GO TO 200
1961 LIMIT=LIMIT+NSTEP
1962 CALL PRINTIT(V,U,ANG,ERR1,ERR2,ERR3,NO)
1963 IF(IERR2 .LE. TOLERN) GO TO 50
1964 200 IF(IERR2 .GE. ITMAX) GO TO 250
1965 GO TO 100
1966 250 ISTD=250
1967 300 CALL FSPACE(AM)
1968 CALL FSPACE(AM)
1969 CALL FSPACE(EV)
1970 CALL FSPACE(BP)
1971 CALL FSPACE(NUMB)
1972 CALL TIME(3,C,RES)
1973 IF(I IN T .LE. PLCTR) GO TO 350
1974 IF(ITA .EQ. 2) IT=IT+1
SUBROUTINE MODER(NP,NUMB,SEQ,TYPE,V,U,ANG,SCHED,IT,SL,Y,CPONT,NXT,R,PONT,NXTC,ERR1,ERR2,ERR3)

REAL PP(l),L(ll,ANG(ll)
INTEGER*2 NUMB(l),CPONT(l),NXTR(l),RPONT(l),NXT(l),SEQ(l),TYPE(l)
COMPLEX Y(l),V(l),SL(l),SCHED(l),IK,DS

COMMON /ADS/ IA,10,15,ALG,BATCH
LOGICAL ALG(l,l,BATCH
EQUIVALENCE (l,NBUS)

100 CONTINUE
GO TO 200

150 DC 100 1=1,IEND
160 DC 170 I=1,1END
170 DC 400 I=1,1END
180 DC 90 I=1,1END
190 DC 300 I=1,1END
200 DC 400 1=1,1END
210 DC 90 1=1,1END
220 DC 400 I=1,1END
230 DC 90 I=1,1END
240 DC 400 I=1,1END
250 DC 90 I=1,1END
260 DC 400 I=1,1END
270 DC 90 I=1,1END
280 DC 400 I=1,1END
290 DC 90 I=1,1END
300 DC 400 I=1,1END
310 DC 90 I=1,1END
320 DC 400 I=1,1END
143.

2040 330 IF(PP .EQ. FIRST) GO TO 350
2041 VP=(PP-7)/6
2042 J=RPCNT(PP)
2043 IK=I+Y(VP)*V(J)
2044 PP=NXTC(PP)
2045 GO TO 330
2046 350 CONTINUE
2047 IK=I+Y(I)*V(K)
2048 IC=SCHED(KI-IK)
2049 DP=REAL(DS)
2050 BP(KB)=DP
2051 DV=(CV1+DP)
2052 ADD=ABS(DP)
2053 IF(ERR2 .GE. ADD) GO TO 360
2054 ERR2=ADD
2055 IERR2=K
2056 360 ERR=ERR+ADD
2057 400 CONTINUE
2058 ERR1=V1*2*V2**2
2059 RETURN
2060 END
2061 SUBROUTINE STCTT(V,U,ANG,NO,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,NXTC)
2062 *
2063 EXTERNAL ORCER,DPDC,DPDC,CRA
2064 COMPLEX Y(11,VI1) .SCHED(1)
2065 INTEGER*2 CPONT(1),NXTR(1),RPONT(1),NXTC(1),NC(1),TYPE(1)
2066 INTEGER SEC(1),NSTEP
2067 REAL U(N),ANG(1)
2068 EQUIVALENCE (I1,NBUS)
2069 CALL TIME0
2070 COMM /ININF/ I1,12,13,TMAX,TOLERN,ZERO,BMVA,LINELIST,IEND
2071 COMM /ADS/ IA,ID,SL,G,BATCH
2072 COMM /OUTINF/ NPAGE,NLINE,HONGS,ISTIC
2073 COMM /TYM/ RES
2074 CALL PLOT=LIST,LT,C
2075 IF(IPLCT)REWIND 3
2076 IF(IPLCT) CALL EMPTYFI3)
2077 PRINT=IABS(LIST),I10
2078 NSTEP=ABS(LIST)/10
2079 31 N=I1(IEN0-1)/32+1
2080 N=IN/IEND*4
2081 CALL GSPACE(ACCNN,NN)
2082 CALL GSPACE(ATEMP,IEND*2)
2083 CALL CALLER(CORD,ACCNN,IPTR(N1),IPTR(NXTR),IPTR(RPONT)
2084 *IPTR(NXTC),IPTR(SEQ),ATEMP,IPTR(NPV),IPTR(TYPE))
2085 CALL FSPACE(ATEMP)
2086 CALL FSPACE(ACCNN)
THE STORAGE IS SLIGHTLY OVER-ALLOCATED.

NW1 = (NPV + NPG) * 2
NW2 = NPG * 2
CALL GSSPACE(B1, NW1 * 4)
CALL GSSPACE(B2, NW2 * 4)
CALL GSSPACE(AWR, NBUS * 4)
CALL GSSPACE(AWI, NBUS * 2)
CALL GSSPACE(NUMB, NBUS * 2)
CALL GSSPACE(P1ST, NBUS * 2)
CALL GSSPACE(B2ST, NBUS * 2)
IND = 1
CALL CALLER(BPPPPP, B1, B1, AWR, AWI, B1ST, IPTR(Y), IPTR(CPONT), IPTR(NXTR)
*), IPTR(RPONT), IPTR(NXT), IPTR(SEQ), IPTR(TYPE), NUMB, IPTR(IND))
IND = 2
CALL CALLER(BPPPPP, B2, B2, AWR, AWI, B2ST, IPTR(Y), IPTR(CPONT), IPTR(NXTR)
*), IPTR(RPONT), IPTR(NXT), IPTR(SEQ), IPTR(TYPE), NUMB, IPTR(IND))
CALL FSPACE(AW1)
CALL GSSPACE(CEL1, IENC * 4)
CALL GSSPACE(CEL2, IEND * 4)
IT = 0
LIMIT = 0
TLAST = ITMAX * 2
KP = 1
KQ = 1
1CC IND = 1
CC 110 I = 1, IEND
A1 = U(I) * COS(ANG(I))
A2 = U(I) * SIN(ANG(I))
1CC V(I) = CMPLX(A1, A2)
CALL CALLER(DPDC, DEL1, IPTR(Y), IPTR(CPONT), IPTR(NXTR), IPTR(RPONT),
*), IPTR(NXT), IPTR(SEQ), IPTR(TYPE), IPTR(U), IPTR(V), IPTR(SCHED), IPTR(U), IPTR(V),
*ERR1, IPTR(ERR2), IPTR(ERR2), IPTR(ERR3), IPTR(ERR3), IPTR(KP), IPTR(IND))
IERR2 = NO(INERR2)
IF(IERR2 < 0) IERR2 = -IERR2
IF(PLOT) CALL PLOTR(U, ANG, ERR1, ERR2, ERR3, NO)
IF(.NOT. PRINT .OR. IT .NE. LIMIT) GO TO 120
CALL PRINTR(IT, V, U, ANG, ERR1, ERR2, ERR3, NO)
LIMIT = LIMIT * ASTEP
1CC IF(KP .EQ. C) GO TO 200
1CC IF(IT .GE. ITLAST) GO TO 350
CALL CALLER(DPDC, DEL2, IPTR(Y), IPTR(CPONT), IPTR(NXTR), IPTR(RPONT),
*), IPTR(NXT), IPTR(SEQ), IPTR(TYPE), IPTR(U), IPTR(V), IPTR(SCHED), IPTR(U), IPTR(V),
*ERR1, IPTR(ERR2), IPTR(ERR2), IPTR(ERR3), IPTR(ERR3), IPTR(KP), IPTR(IND))
IERR2 = NO(INERR2)
IF(IERR2 < 0) IERR2 = -IERR2
IF(PLOT) CALL PLOTR(U, ANG, ERR1, ERR2, ERR3, NO)
IF(.NOT. PRINT .OR. IT .NE. LIMIT) GO TO 150
CALL PRINTR(IT, V, U, ANG, ERR1, ERR2, ERR3, NO)
LIMIT = LIMIT * ASTEP
1CC IF(KP .EQ. C) GO TO 200
1CC IF(IT .GE. ITLAST) GO TO 350
CALL CALLER(DPDC, DEL2, IPTR(Y), IPTR(CPONT), IPTR(NXTR), IPTR(RPONT),
*), IPTR(NXT), IPTR(SEQ), IPTR(TYPE), IPTR(U), IPTR(V), IPTR(SCHED), IPTR(U), IPTR(V),
*ERR1, IPTR(ERR2), IPTR(ERR2), IPTR(ERR3), IPTR(ERR3), IPTR(KP), IPTR(IND))
IERR2 = NO(INERR2)
IF(IERR2 < 0) IERR2 = -IERR2
IF(PLOT) CALL PLOTR(U, ANG, ERR1, ERR2, ERR3, NO)
IF(.NOT. PRINT .OR. IT .NE. LIMIT) GO TO 150
CALL PRINTR(IT, V, U, ANG, ERR1, ERR2, ERR3, NO)
LIMIT = LIMIT * ASTEP
145.

2160 IF(IT .GE. ITLASTI) GO TO 350
2161 CALL CALLER(DVDD,B2,B2,B2ST,DEL2,IPTR(U),IPTR(TYPE),IPTR(SEQ),
2162 *NUMB,IPTR(IND))
2163 IT=IT+1
2164 GO TO 100
2165 200 IF(KQ .EQ. 0) GO TO 400
2166 GO TO 130
2167 30C IF(KP .EQ. 0) GO TO 400
2168 GO TO 100
2169 35C IST=350
2170 400 CALL FSSPACE(B1)
2171 CALL FSSPACE(B2)
2172 CALL FSSPACE(B1ST)
2173 CALL FSSPACE(B2ST)
2174 CALL FSSPACE(NUMB)
2175 CALL TIME(C,C,R)
2176 IF(.NOT. PLCT) GC TO 450
2177 IT=IT-1
2178 IMAX=-IMAX
2179 IX=(IEND*2+3)*IT*4
2180 CALL GSPACE(ARRAY,NN)
2181 CALL CALLER(CRAW,IPTR(I1),IPTR(NC),ARRAY)
2182 IMAX=-IMAX
2183 CALL FSSPACE(ARRAY)
2184 IT=IT-1
2185 45C CALL OUT1(IT,ERR1,ERR2,IERF2,ERR3)
2186 IF(ISTNO .NE. 0) RETURN 1
2187 RETURN
2188 END
2189 SUBROUTINE BPBPP(BR,BI,WRR,WRI,BST,Y,CPONT,NXTR,RPCNT,NXTC,SEQ,TYP
2190 *E,NUMB,IND)
2191 REAL RR1,WRR1
2192 INTEGER BI(1),FIRST,VP,PP
2193 COMPLEX Y1)
2194 INTEGER*2 CPCNT(1),NXTR(1),RPCNT(1),NXTC(1),SEC(1),TYPE(1)
2195 INTEGER*2 WR1(1),BST(1),NUMB(1)
2196 COMMON /ININF/ I1,I2,I3,IMAX,TOLERN,ZERO,BMVA,LINE,LIST,EEND
2197 EQUIVALENCE(I1,NBUS)
2198 LOGICAL=1 Dprime
2199 Dprime= IND .NE. 1
2200 IF(Dprime) GO TO 15
2201 DC 10 I=1,IENC
2202 K=SEC(I)
2203 NUMA(K)=I
2204 10 CONTINUE
2205 15 BST(I)=1
2206 DC 1000 II=1,IENC
2207 I=SEC(II)
2208 IF(Dprime .AND. TYPE(I) .NE. 1) GO TO 700
2209 K=2
2210 FIRST=I*8-7
2211 SUM=O.
2212 PP=NXT(FIRST)
2213 20 IF(PP .EQ. FIRST) GO TO 50
2214 VP=(PP+7)/4
2215 J=PCPTNT(PP)
2216 IF(Dprime .AND. TYPE(J) .NE. 1) GO TO 30
2217 BIJ=AIMAG(Y(VP))
2218 IF(Dprime) GO TO 25
2219 GIJ=REAL(Y(VP))
RATIC = GIJ / BIJ
BIJ = RIJ * (1. + RATIO**2)
SUM = SUM + BIJ
IF (J > IEND) GO TO 80

WR(K) = -BIJ
WRI(K) = J
K = K + 1

PP = NXTR(PP)
GO TO 70

PP = NXTC(FIRST)
IF (PP .EQ. FIRST) GO TO 100

VP = (PP + 7) / 4
J = RPCNT(PP)
IF (DFRIME .AND. TYPE(J) .NE. 1) GO TO 10

BIJ = IMAG(VP)
GO TO 75

GIJ = REAL(Y(VP))

BIJ = BIJ * (1. + RATIO**2)
SUM = SUM + BIJ
IF (J > IEND) GO TO 80

WRR(K) = -BIJ
WRI(K) = J
K = K - 1

WORKING RSF FORMATION IS COMPLETED

IF (II .EQ. 1) GO TO 500

IM1 = II + 1
DO 220 JJ = 1, IM1
J = SEQ(JJ)
IF (DFRIME .AND. TYPE(J) .NE. 1) GO TO 223
INDEX = BST(JJ) + 1
JJ1 = JJ
JJ1 = JJ1 + 1
LAST = BST(JJ1)
IF (LAST .EQ. 0) GO TO 105

INDEX = RST(JJ1) + 1
JJ1 = JJ

K = JJ + 1
DO 226 KK = 1, K
IF (WR(KK) .EQ. JK) GO TO 140
CONTINUE
K = K + 1
WRR(KK) = 0
JK = SI(INDEX)
DO 221 KK = 1, K
IF (WR(KK) .EQ. JK) GO TO 140
CONTINUE
K = K + 1
WRR(KK) = 0
WRI(KK) = JK
JKK = K
WRR(JKK) = WR(JKK) + BR(INDEX + 1) * RATIO
INDEX = INDEX + 2
IF(INDEX .LT. LAST) GO TO 120

CONTINUE

WORKING ROW COMBINATION PROCESS COMPLETED

NOW DIVIDE THE ROW BY ITS DIAGONAL AND STORE THE RESULTS

RATIO=1./WRRI

INDEX=BST(I)

INDEX=INDEX+2

CONTINUE

INDEX=INDEX+1

IF(K .EQ. 0) GO TO 650

DO 600 KK=2,K

INDEX=INDEX+1

CONTINUE

GO TO 650

DO 600 KK=2,K

INDEX=INDEX+1

CONTINUE

GO TO 1000

CONTINUE

END

SUBROUTINE CP (CEL.Y, CPON.T, NXTR, RPONT, NXC.T, SEC, TYPE, V, SCHD1., U, ERR

*1, ERR2, ERR3, KPQ, IND)

REAL CEL(1), U(1)

COMPLEX Y(1), V(1), SCHD(1), I, K, SK, DS

INTEGER*2 CPCNT(1), NXTR(1), RPONT(1), NXT.C(1), SEC(1), TYPE(1)

COMMON /ININF/ I1, I2, I3, ITMAX, TOLER, ZERO, VMVA, L.I.NE, LIST, IEND

EQUIVALENCE (I1, NSUS)

LOGICAL*1 DPRIME

INTEGER FIRST, VP, PP

KPQ=C

ERR2=0.

ERR3=0.

DV1=0.

DV2=0.

DPRIME=IND .EQ. 2

I=SEC(I)

FIRST=I*8-7

IF(PP .EQ. FIRST) GO TO 50

VP=(PP+7)/4

J=CPCTN(PP)

IK=K+Y(PP)*V(J)

PF=NXTR(PP)

GO TO 30

PP=NXT.C(FIRST)

IF(PP .EQ. FIRST) GO TO 100

VP=(PP+7)/4

J=RPCNT(PP)

IK=K+Y(PP)*V(J)

PP=NXTC(PP)

GO TO 80

SK=V(I)*CONJG(K)

DS=SCHD(I)-SK

DP=REAL(DS)
ADD=ABS(DP)
IF(DPRIME) GO TO 120
DEL(I)=DP/VI(I)
IF(ACD*GT*TCLERN) KPQ=1
DV1=DV1+DP
ERR3=ERR3+ADD
IF(ADD*GE*ADD) GO TO 130
ERR2=ADD
IERR2=1
IF(TYPE(I)*EQ.*2) GO TO 200
DO=AIMAG(DS)
ACD=ABS(DQ)
IF( .NOT. CPRIME) GO TO 140
DEL(I)=DO/U(I)
IF(ADD*GT*TCLERN) KPQ=1
DV2=DV2+DO
ERR3=ERR3+ADD
IF(ADD ADD) GO TO 130
ERR=ADD
IERR2=I
CONTINUE
IF(ERR1=SORT(DV1**2+DV2**2)
RETURN
END
SUBROUTINE DVCDI BR,BI,BST,CEL,UA N G,TYPE,SEQ,NUMB,IND
REAL BR(I),CEL(I),UA N G(I)
INTEGER BI(I)
INTEGER*2 BST(I),TYPE(I),SEQ(I),NUMB(I)
COMMON /ININF/ II, I2, I3, ITMAX, TCLERN, ZERO, BMVA, LINE, LIST, IEND
EQUIVALENCE (II,ABUS)
LOGICAL*1 DPRIME
DPRIME=IND+EQ.*2
DO 200 II=1, IEND
I=SEQ(I)
INDEX=BST(I)+1
II=II+1
LAST=BST(II)
IF(LAST*EQ.*0) GO TO 40
JJ=BI(I)
J=NUMB(J)
INDEX=INDEX+2
IF(JM*GT*LAST) GO TO 200
INDEX=BST(II)
DEL(I)=DEL(II)-BR(I)*INDEX
INDEX=INDEX+2
GO TO 50
DO 400 II=1, IEND
JJ=IEND+1-II
I=SEQ(JJ)
IF(DPRIME*AND*TYPE(I)*EQ.*2) GO TO 400
INDEX=BST(JJ)+1
DO 400 II=1, IEND
JJ=IEND+1-II
I=SEQ(JJ)
IF(DPRIME*AND*TYPE(I)*EQ.*2) GO TO 400
INDEX=BST(JJ)+1
2400 JJ1=JJ
2401 JJ1=JJ1+1
2402 LAST=BST(JJ1)
2403 IF(LAST .EQ. 0) GO TO 240
2404 IF(INDEX .GE. LAST) GO TO 300
2405 J=BI(INDEX)
2406 J=NUMB(J)
2407 SUM=SUM+ARR(INDEX+1)*DEL(J)
2408 INDEX=INDEX+2
2409 GO TO 250
2410 DEL(JJ1)=DEL(JJ1)-SUM
2411 400 CONTINUE
2412 C BACK SUBSTITUTION ENDED.
2414 C DO 500 I=1,IEND
2415 J=SEQ(I)
2416 IF(DP) GO TO 450
2417 UANG(J)=UANG(J)+I
2418 GO TO 500
2420 450 IF(ITYPE(J) .EQ. 2) GO TO 550
2421 UANG(J)=UANG(J)+OEL(I)*UANG(J)
2422 550 CONTINUE
2423 RETURN
2424 END
2425 SUBROUTINE CRAWIT, NC, ARRAY)
2426 REAL ALAY(IN,1)
2427 INTEGER NC(I)
2428 COMMON /ININF/ IT,12,13,ITMAX,TOLERN,ZERO,BMVAgLIE,LISEND
2429 COMMON /OUTINF/ NPG,NLINE,HONGS,ISTNO
2430 LOGICAL HDNGS(1,1)
2431 EQUIVALENCE (IT, NBU)
2432 IF(IT .LE. 1) RETURN
2433 CALL FTNCMDASS IGN9=PLCTFILE•,17
2434 CALL EMPTYFI9)
2435 REWIND 3
2436 J=1
2437 50 K=1
2438 DC 100 I=1,. IEND
2439 IF(NC(I) .GT. 0) GO TO 100
2440 READ(3)ARRAY(J,K),ARRAY(J,K+1)
2441 K=K+2
2442 100 CONTINUE
2443 READ(3)ARRAY(J,K),ARRAY(J,K+1),ARRAY(J,K+2)
2444 J=1+1
2445 IFJ .LE. IT) GO TO 50
2446 CALL PLCTRL(*METR*,1)
2447 K=1
2448 DC 20C I=1,.IEND
2449 IF(NC(I) .GT. 0) GO TO 200
2450 AND=--KO(I)
2451 CALL DRAW2(1, IT, ITMAX, ARRAY, K)
2452 CALL SYMBOL(10.5, 2...35, 'BUS #',90.,5)
2453 CALL NUMBER(10.5, 4...35, 'ANG',90.,-1)
2454 CALL SYMBOL(10.5, 5...35, 'U',90.,3)
2455 CALL SYMBOL(10.5, 14...35, 'BUS #',90.,5)
2456 CALL NUMBER(10.5, 16...35, 'AND',-1)
2457 CALL SYMBOL(10.5, 17...35, 'ANG',90.,5)
2458 CALL PLOT(15.,C.,-3)
2459 K=K+2
CONTINUE
CALL DRAW2(IT,ITMAX,ARRAY,K)
CALL SYMBOL(10.5,2..35,'ERR 1',90..5)
CALL COMPLEMENTARY(10.5,14.5,35,'ERR 2',90..5)
CALL PLOT(15,0,.3)
K=K+2
K=-K
CALL DRAW2(IT,ITMAX,ARRAY,K)
C PLOT AN INVISIBLE BOUNDARY:
CALL PLOT(10.5,24.5,3)
CALL SYMBOL(10.5,2..35,'ERR 3',90..5)
CALL SYMBOL(10.5,25,HONGS,0,48)
CALL SYMBOL(3,16,25,HONGS(49),0,32)
RETURN
END
SUBROUTINE PLOTRIU,ANG,ERR1,ERR2,ERR3,NO
REAL U(I),ANG(I)
INTEGER*2 NO
COMMCN ININF IT1,12,13,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
EQUIVALENCE (Il,NBUS)
00 İCC 1=1,IEND
IF IMOI I ) .GT. 0) GO TO 100
ANG 1 = ANGI 1)* 180./3.14159
WRITE 13 1 U(I,ANGI
CONTINUE
WRITE 13 1 ERR 1,ERR2,ERR3
RETURN
END
SUBROUTINE CRAW2(IT,ITMAX,ARRAY,KK)
REAL ARRAY(IT,1)
COMMON /ACS/ IA,IC,IS,ALG,BATCH
LOGICAL*1 ALG(8),BATCH
LOGICAL EQUC
INDEX=0
IF(KK .GT. 0) GC TO 20
KK=-KK
INDEX=1
Y = 1.
K = KK
FACTOR=1.
IF(ITMAX .LT. 0) FACTOR=.5
DX=ABS(ITMAX)/10.
XINC=FACTOR/DX
CALL AXCTRL( 'YORIG',Y)
CALL AXCTRL('CIGITS',-1)
CALL AXPLOT(ITERATION;10,.0,.0,.0,.0,.DX)
DIFF=ARRAY(I,K)-ARRAY(IT,K)
IF(DIFF) .GT. .0000001) GO TO 70
IF(MOD(K,2) .EQ. 0) GO TO 65
YMIN=-8.
DY=-.05
DC 6C I=1,IT
ARRAY(I,K)=ARRAY(I,K)-.8)*20.
GO TC 80
YMIN=-4.
DY=1.
DO 67 I=1,IT
ARRAY(I,K)=ARRAY(I,K)+4.
GO TC 80
CALL SCALE(ARRAY(I,K),IT,1C,YMIN,DY,1)
CALL AXCTRL('CIGITS',-2)
CALL AXPLOT(1:,SC..10.,YMIN,DY)
X=0.
IF(ECUC(ALG(1),L') X=XINC
YSCLD=ARRAY(I,K)+Y
CALL PLOT(X,YSCLD,3)
CALL PLOT(X,YSCLD,2)
DO 100 I=2,IT
X=X+XINC
YSCLD=ARRAY(I,K)+Y
CALL PLOT(X,YSCLD,1)
CCONTINUE
IF(INDEX .NE. 0) RETURN
K=K+1
INDEX=INDEX+1
Y=13.5
GO TO 50
END
SUBROUTINE DECPL2(V,U,ANG,NO,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,
NXTC,*
EXTERNAL CRDEF,MC1MC2,CECS,DU2DD,DRAW
COMPLEX Y(1I,V(1I,SCHED(1)
INTEGER*2 CFCNT I11,NXTRI1),RPONTI1),NXTCI1),NO 11),TYPE(1)
INTEGER SECI1,11STEP
REAL Ull KANC
LOGICAL*1 BATCH,ALG,10NGS(80),PLCT,PRINT
COMMON /ININF/ I1,I2,I3,ITMAX,TOLERN,ZERO,BMVRA,L1NE,L1ST,1END
COMMON /ADS/ IA,1D,IS,ALG,BATCH
COMM1N /OUTINF/ NPAGE,1LINE,10NGS,1STNO
COMM1N /TYM/ RES
EQUIVALENCE (II,ABUS)
IA=1
GC TC 10
ENTRY DECPL2(V,U,ANG,NO,SCHED,TYPE,SEQ,Y,CPONT,NXTR,RPONT,NXTC,*)
CCONTINUE
CALL TIME(0)
IF(PLOT) REWIND 3
CALL EMPTYF(3)
PRINT=1ABS(LIST) .GT. 10
NSTEF=IABS(LISTI/10
N1=(IEND-1)/32+1
NN=N1*IEND*4
CALL GSPACE(IACCNN,N1)
CALL GSPACE(ATEMP,IEND*2)
CALL CALLER(CRCER,ACCNN,IPTR(N1),IPTR(CPONT),IPTR(NXTR),IPTR(RPONT
*),IPTR(NXTC),IPTR(SEQ),ATEMP,IPTR(ACPN),IPTR(NPV),IPTR(TYPE))
CALL GSPACE(ATEMP)
CALL GSPACE(ACCNN)
THE STORAGE IS SLIGHTLY OVER - ALLOCATED.
CALL GSPACE(NUMB,NBUS*2)
CALL GSPACE(EIST,NBUS*2)
CALL GSPACE(E2ST,NBUS*2)
IND = 1
CALL CALLER(MC1MC2,B1,AWR,AWI,B1ST,IPTR(Y),IPTR(CPNT),IPTR(NXT)
*),IPTR(RPONT),IPTR(NXTC),IPTR(SEO),IPTR(TYPE),NUMB,IPTR(IND))
IND = 2
CALL CALLER(MC1MC2,B2,AWR,AWI,B2ST,IPTR(Y),IPTR(CPNT),IPTR(NXT)
*),IPTR(RPONT),IPTR(NXTC),IPTR(SEO),IPTR(TYPE),NUMB,IPTR(IND))
CALL FSPACE(AWR)
CALL FSPACE(AWI)
CALL GSPACE(CEL1,IND*4)
CALL GSPACE(CEL2,IND*4)
IT = 0
LIMIT = 0
ITLAST = ITMAX*2
IT = 1
KP = 1
K2 = 1
IND = 1
DO 110 I = 1,1END
AI = U(I)*COS(ANG(I))
A2 = U(I)*SIN(ANG(I))
110 V(I) = CMPLX(A1,A2)
DO 140 I = 1,1END
A1 = U(I)*COS(ANG(I))
A2 = U(I)*SIN(ANG(I))
140 V(I) = CMPLX(A1,A2)
CALL CALLER(DECS,DEL1,IPTR(Y),IPTR(CPNT),IPTR(NXT),IPTR(RPONT),
*),IPTR(NXTC),IPTR(SEO),IPTR(TYPE),IPTR(U),IPTR(V),IPTR(SCHED),IPTR(U),IPTR(
*),IPTR(ERR1),IPTR(ERR2),IPTR(ERR3),IPTR(KP),IPTR(IND))
IERR2 = NO(IERR2)
IF(IERR2 .LT. 0) IERR2 = -IERR2
IF(IPLOT) CALL PLOTR(U,ANG,ERR1,ERR2,ERR3,NO)
IF(.NOT. PRINT .OR. IT .NE. LIMIT) GO TO 120
CALL PRTTR(U,ANG,ERR1,ERR2,ERR3,NO)
LIMIT = LIMIT-NSTEP
IF(KP .EQ. 0) GO TO 300
IF(IT .GE. ITLAST) GO TO 350
CALL CALLER(DU2DD,B1,BlST,DEL1,IPTR(ANG),IPTR(TYPE),IPTR(SEO),
*NUMB,IPTR(INC))
IT = IT+1
120 IF(KP .NE. C) GO TO 200
130 IF(IT .GE. ITLAST) GO TO 350
CALL CALLER(DU2DD,B1,BlST,DEL1,IPTR(ANG),IPTR(TYPE),IPTR(SEO),
*NUMB,IPTR(INC))
130 IF(KP .NE. C) GO TO 200
GO TO 100
200 IF(KP .NE. C) GO TO 400
GO TO 130
GO TO 300
300 IF(KP .NE. C) GO TO 400
2640 GC TC 100
2641 350 IST=350
2642 40C CALL FSPACE(B1)
2643 CALL FSPACE(B2)
2644 CALL FSPACE(B1ST)
2645 CALL FSPACE(B2ST)
2646 CALL FSPACE(NUMB)
2647 CALL TIME(3,C,RES)
2648 IF(ISTNO, PLOT) GO TO 450
2649 IT=IT+1
2650 ITMAX=-ITMAX
2651 NN=(IEND*2+3)^IT^4
2652 CALL GSPACE(APR, NN)
2653 CALL CALLER(DRAW, IPTR(IT), IPTR(INC), ARRAY)
2654 ITMAX=-ITMAX
2655 CALL FSPACE(ARRAY)
2656 IT=IT-1
2657 450 CALL OUT(I(IT, ERR1, ERR2, ERR2, ERR3)
2658 IF(ISTNO .NE. 0) RETURN 1
2659 RETURN
2660 END
2661 SUBROUTINE MC1MC2IBR,Bl,WRR,WRI,BST,Y,CPONT,NXTR,RPONT,NXTC,SEQ,
2662 TYPE,NUMB,INC)
2663 REAL BP (I), WP, RP.
2664 INTEGER Bl(I), FIRST, VP, PP
2665 COMPLEX Y(I)
2666 INTEGER*2 CPONT(I), NXTR(I), RPONT(I), IDX(I), SEC(I), TYPE(I)
2667 integer*2 WRI(I), BST(I), NUMB(I)
2668 COMMON /ININF/I1, I2, I3, ITMAX, TOLER, ZERO, BMVA, LINE, LIST, IEND
2669 COMMON /ADS/ IA, I0, IS, ALG, BATCH
2670 LOGICAL*1 INLONG, INLGBI, I1, I2, I3, ITMAX, TOLER, ZERO, BMVA, LINE, LIST, IEND
2671 EQUIVALENCE(I1, NBUSI)
2672 LOGICAL*1 DPRIME
2673 DPRIME= IND .NE. 1
2674 IF(DPRIME) GC TO 15
2675 D C 10 I=1, IENC
2676 K=SEC(I)
2677 NUMB(K)=I
2678 10 CONTINUE
2679 15 BST(I)=1
2680 DC 1000 II=1, IEND
2681 IF(SEC(IFIRST)) GO TO 700
2682 K=2
2683 FIRST=I-8-7
2684 SUM=C.
2685 PP=NXTA(FIRST)
2686 IF(PP .EQ. FIRST) GO TO 50
2687 WP=(PP+7)/4
2688 J=CPONT(PP)
2689 BIJ=AIMAG(Y(VP))
2690 IF(IA .EQ. 1) GO TO 23
2691 GIJ=REAL(Y(VP))
2692 RATIC=BIJ/GIJ
2693 BIJ=BIJ-1*RATIC**2
2694 SUM=SUM+BIJ
2695 IF(DPRIME .AND. TYPE(J) .NE. 1) OR. J .GT. IEND) GO TO 30
2696 WR(K)=-BIJ
2697 WR(K)=J
2698 K=K+1
2700  30  PP=NXTTR(PP)
2701    GO TO 20
2702  50  PP=NXTC(FIRST)
2703  70  IF(PP .EQ. FIRST) GO TO 100
2704     VP=(PP-7)/4
2705    J=RPCNT(PP)
2706   BIJ=AIMAG(Y(VP))
2707  73  IF(IA .EQ. 1 .OR. DPRIME) GO TO 73
2708   GIJ=REAL(Y(VP))
2709   RATI=GIJ/BIJ
2710  80  BIJ=BIJ*(1.+RATI**2)
2711  100  SUM=SUM+BIJ
2712  125  WORKING ROW FORMATION IS COMPLETED
2713  130 工作的行组合过程完成
2714  140  0
BR(INDEX)=RATIO
INDEX=INDEX+1
IF(K .EQ. 1) GO TO 650
DO 600 KK=2,K
IF(WR(KK) .EQ. 0) GO TO 600
BI(INDEX)=WR(KK)
BR(INDEX+1)=WR(KK)*RATIO
INDEX=INDEX+2
CONTINUE
600 CONTEINUE
650 BST(II+1)=INDEX
600 GC TO 1000
700 BST(II+1)=BST(II)
700 BST(II)=0
1000 CONTINUE
RETURN
END

SUBCUTINE DECIDE(Y,CPCNT,NXTR,RFCNT,NXTC,SEQ,TYPE,V,SCHED,U,ERR)
*1,ERR2,ERR3,KPC,IND)
REAL C1(1),U(1)
COMPLEX Y(1),V(11,SCHED(1),I,K,SK,DS
INTEGER*2 CPCNT(1),NXTR(1),RFONT(1),NXTC(1),SEQ(1),TYPE(1)
COMMON /ININF/ I1,I2,I3,ITMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
EQUIVALENCE (II,NBUS)
LOGICAL MPMME
INTEGER FIRST,VP,
KPQ=0
ERR2=0.
ERR3=0.
DV1=0.
DV2=0.
DO 200 I=1, IEND
I=SEC(I)
FIRST=I*8-7
IK=Y(VP)*V(J)
PP=NXTR(PF)
GC TO 30
50 PP=NXTC(PP)
200 GC TO 50
200 IF(PP .EQ. FIRST) GO TO 50
257 VP=(PP+7)/4
298 J=CPCNT(PP)
299 IK=IK+Y(VP)*V(J)
280 PP=NXTR(PP)
281 GC TO 30
282 PP=NXTC(PP)
283 IF(PP .EQ. FIRST) GO TO 100
284 VP=(PP+7)/4
285 J=RFONT(PP)
286 IK=IK+Y(VP)*V(J)
287 PP=NXTC(PP)
288 GC TO 80
289 SK=V(I)*CONJG(IK)
290 CS=SCHED(I)-SK
291 DP=REAL(DS)
292 ACD=ABS(DP)
293 IF(ACD .LT. TCLERN) KPQ=1
294 DEL(I)=DP
295 IF(ACD .LT. TCLERN) KPQ=1
296 DV1=DV1+DP
297 ERR3=ERR3+ADD
298 IF(ERR2 .GE. ADD) GO TO 130
299 ERR2=ADD
156.

2820 IERR2=I
2821 130 IF(TYPE(I) .EQ. 2) GO TO 200
2822 DO=AIMAG(DO)
2823 ADD=ABS(DO)
2824 IF(.NOT. DPRIME) GO TO 140
2825 DEL(I)=DO
2826 IF(ADD .GT. TOLERN) KPQ=1
2827 14C DV2=CV2*DO
2828 EPR3=ERR3*ADD
2829 IF(ERR2 .GE. ADD) GO TO 200
2830 IERR2=ADD
2831 200 CONTINUE
2832 ERR1=SORT(DV1**2+DV2**2)
2833 RETURN
2834 END
2835 SUBROUTINE DUZCD(BR,RI,BST,DEL,SEQ,NUMB,IND)
2837 REAL BR(1),DEL(1),SEQ(1)
2838 INTEGER BI(1)
2839 INTEGER*2 BST(I),SEQ(I),NUMBI(1)
2840 COMMON /ININF/ II,I2,I3,IMAX,TOLERN,ZERO,BMVA,LINE,LIST,IEND
2841 EQUIVALENCE (II,NBUS)
2842 LCGICAL*1 DPRIME
2843 DPRIME= IND .EQ. 2
2844 DO 200 II=1,IEND
2845 I=SEQ(I)
2846 4C IFDPRIME .AND. TYPE(I) .EQ. 2) GO TO 200
2847 INDEX=BST(I)
2848 II=II+1
2849 4O IF(INDEX .EQ. 0) GO TO 40
2850 LAST=BST(I)
2851 5O IF(INDEX .GE. LAST) GO TO 100
2852 JJ=BI(INDEX)
2853 JJ=NUMB(JJ)
2854 DEL(JJ)=DEL(JJ)+BR(INDEX+1)
2855 INDEX=INDEX+2
2856 GO TO 50
2857 100 INDEX=BST(I)
2858 DEL(I)=DEL(I)+BR(INDEX)
2860 200 CONTINUE
2861 C FORWARD PROCESS COMPLETED
2862 C START BACK SUBSTITUTION.
2863 C DO 400 JJ=IEND
2864 JJ=IEND-I-1
2865 I=SEQ(JJ)
2866 IFDPRIME .AND. TYPE(I) .EQ. 2) GO TO 400
2867 SUM=C.
2868 I=SEQ(JJ)
2869 INDEX=BST(JJ)+1
2870 JJ=JJ+1
2871 JJ=JJ+1
2872 JJ=JJ+1
2873 24C JJ=JJ+1
2874 LAST=BST(JJ)
2875 IF(LAST .EQ. 0) GO TO 24C
2876 25O IF(INDEX .GE. LAST) GO TO 300
2877 J=BI(INDEX)
2878 J=NUMB(J)
2879 SUM=SUM+BR(INDEX+1)*DEL(J)
INDEX=INDEX+2
GO TO 250
DEL(JJI)=DEL(JJI)-SUM
CONTINUE
C    BACK SUBSTITUTION ENDED.
C
DC 500 I=1,IEND
J=SEQ(I)
IF(DPRIME) GC TO 450
UANG(J)=UANG(J)*DEL(I)
GC TO 500
45C IF(TYPE(J) EQ. 2) GC TO 500
UANG(J)=SORT(UANG(J)**2+2.*DEL(I))
CONTINUE
RETURN
END
**GTNAME CSECT**

***********************************************************************
CALLING SEQUENCE FROM FORTRAN:
CALL GTNAME(MTU,NAME,LEN)
WHERE
MTU IS THE LOCATION OF AN ARRAY THAT CONTAINS THE NAME
OF THE MTS UNIT. NAME MUST BE 8 CHARACTERS. IF IT IS LESS
THAN 8, IT MUST BE LEFT JUSTIFIED WITH TRAILING BLANKS.
EG. "SCARDS"
NAME() IS A LOGICAL*1 ARRAY THAT IN RETURN WILL CONTAIN
THE NAME OF THE FILE ASSIGNED TO THAT UNIT.
LEN IS AN INTEGER VARIABLE THAT IN RETURN WILL CONTAIN
THE NUMBER OF CHARACTERS IN NAME().
***********************************************************************

LSN #,9
ST 14,12,12(13)  STORE ALL THE REGISTERS BUT 13.
LR 9,15  SET BASE REGISTER
LA 10,SAVE
ST 10,8(0,13)  SET THE FORWARD LINK
ST 13,4(0,10)  SET THE BACKWARD LINK
LR 13,10  ESTABLISH THE SAVE AREA
L 5,(1C,1)  GR5=AIMTU
L 6,(1C,1)  GR6=AINAME
L 7,(1C,1)  GR7=ALEN
LW 0,1,(5)  GET THE NAME OF THE UNIT.
L 15,=V(GDINFO)  THEN
PALR 14,15  CALL THE SUBROUTINE GDINFO
LTR 15,15  CHECK RETURN CODE
BNZ .ERRC  AND BRANCH IF NON-ZERO
L 1,36(0,1)  GR1=ALENGTH AND NAME
LH 2,010(1)  GR2=LENGTH OF NAME
LA 1,210(1)  GR1=FIRST LOCATION OF NAME
ST 2,010(7)  SET LEN = LENGTH
RCTR 2,.C  SUBTRACT ONE FROM LENGTH
EX 2,MOVE  EXECUTE MOVE
L 13,4(0,13)  RESTORE SAVE AREA ADDRESS
LM 14,12,12(13)  RESTORE GENERAL REGISTERS
SR 15,15  INDICATE ZERO RETURN CODE
BCR 15,14  RETURN
ERROR  
L 13,4(0,13)  RESTORE SAVE AREA ADDRESS
L 14,12(0,13)  RESTORE RETURN ADDRESS
LM 0,12,20(13)  RESTORE GRO TO GR12
BC 15,14  RETURN
MOVE MVC 0,(0,61),(0,11)  MOVE 'FILE NAME' TO 'NAME'
NAME DS 4F
SAVE DS 18F  SAVE AREA
END

**NUMBER CSECT**

USING *,9
STM 14,12,12(13)
LR 9,15
LA 11,SAVE
ST 11,6(0,13)  SET BACKWARD LINK
ST 13,4(0,11)  SET FORWARD LINK
LR 13,11  ESTABLISH SAVE AREA
L 4,(1C,1)  LOAD A(ARRAY)
L 5,(1C,1)  LOAD A(END)
L 5,0(0,5)  LOAD END

*
<table>
<thead>
<tr>
<th>Line</th>
<th>Comment</th>
<th>Machine Instruction Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>* TO INITIALIZE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td></td>
<td>LA 7,0(0,0)</td>
<td>WORD INDICATOR=0</td>
</tr>
<tr>
<td>62</td>
<td></td>
<td>LA 3,33(0,0)</td>
<td>BIT INDICATOR=33</td>
</tr>
<tr>
<td>63</td>
<td></td>
<td>LR 6,7</td>
<td>COUNT=0</td>
</tr>
<tr>
<td>64</td>
<td></td>
<td>L 2,0(7,4)</td>
<td>LOAD PROPER WORD.</td>
</tr>
<tr>
<td>65</td>
<td></td>
<td>GCCN BCT 3,NEXT</td>
<td>DECREMENT BIT INDICATOR:GO TO NEXT IF NON-ZE</td>
</tr>
<tr>
<td>66</td>
<td></td>
<td>LA 3,32(0,0)</td>
<td>RE-INIT. BIT INDICATOR.</td>
</tr>
<tr>
<td>67</td>
<td></td>
<td>LA 7,4(0,71)</td>
<td>MODIFY WORD INDICATOR.</td>
</tr>
<tr>
<td>68</td>
<td></td>
<td>L 2,0(7,4)</td>
<td>LOAD PROPER WORD.</td>
</tr>
<tr>
<td>69</td>
<td>NEXT</td>
<td>LA 3,320,0)</td>
<td>MODIFY WORD INDICATOR.</td>
</tr>
<tr>
<td>70</td>
<td></td>
<td>LTR 2,2</td>
<td>TEST REGISTER 2</td>
</tr>
<tr>
<td>71</td>
<td></td>
<td>BC 11,ZERO</td>
<td>BRANCH IF NON-NEGATIVE.</td>
</tr>
<tr>
<td>72</td>
<td></td>
<td>LA 6,1(0,61)</td>
<td>MODIFY COUNT</td>
</tr>
<tr>
<td>73</td>
<td>ZERO</td>
<td>SLL 2,1(0)</td>
<td>SHIFT LEFT.</td>
</tr>
<tr>
<td>74</td>
<td></td>
<td>BCT 5,GCCN</td>
<td>DECREMENT IEND; BRANCH IF NOT ZERO.</td>
</tr>
<tr>
<td>75</td>
<td>FINISHED</td>
<td>LR 0,6</td>
<td>STORE COUNT.</td>
</tr>
<tr>
<td>76</td>
<td></td>
<td>L 13,4(0,13)</td>
<td>RESTORE SAVE AREA ADDRESS.</td>
</tr>
<tr>
<td>77</td>
<td></td>
<td>LM 14,15,12(13)</td>
<td>RESTORE GENERAL REGISTERS.</td>
</tr>
<tr>
<td>78</td>
<td></td>
<td>LM 1,12,24(13)</td>
<td>EXCEPT GR 0.</td>
</tr>
<tr>
<td>79</td>
<td></td>
<td>SR 15,15</td>
<td>INDICATE ZERO RETURN CODE.</td>
</tr>
<tr>
<td>80</td>
<td></td>
<td>BCR 15,14</td>
<td>RETURN.</td>
</tr>
<tr>
<td>81</td>
<td>SAVE</td>
<td>DS 18F</td>
<td>END OF FILE</td>
</tr>
</tbody>
</table>
APPENDIX B

DATA FOR TEST SYSTEMS 6, 9 AND 10

The data for Test Systems 6, 9 and 10 are given below. This data was obtained from B.C. Hydro and is published here with their permission. The data for all the other test systems, are published in the respective references, mentioned in Table 2.2.

For the sake of clarity, the listing of input data, produced by LFP, is used. The format of the input data (for LFP) is as explained in Appendix A, section A.4. In the listing that follows, the names SAMPLE (4001), SAMPLE (5001) and BCH.138 in the heading refer, respectively, to Test Systems 6, 9 and 10.
## Bus Information

<table>
<thead>
<tr>
<th>Bus #</th>
<th>Bus Name</th>
<th>Type</th>
<th>Generation MW</th>
<th>MVAR</th>
<th>Load MW</th>
<th>MVAR</th>
<th>Base KV</th>
<th>Voltage KV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ING230</td>
<td>3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>235.00</td>
</tr>
<tr>
<td>3</td>
<td>JHT13</td>
<td>2</td>
<td>127.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>4</td>
<td>LDR13</td>
<td>1</td>
<td>47.00</td>
<td>5.00</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>SCA13</td>
<td>1</td>
<td>25.80</td>
<td>5.00</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>PUN13</td>
<td>2</td>
<td>24.50</td>
<td>0.0</td>
<td>17.80</td>
<td>8.15</td>
<td>13.80</td>
<td>14.30</td>
</tr>
<tr>
<td>7</td>
<td>ASH13</td>
<td>2</td>
<td>26.70</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>8</td>
<td>GGA13</td>
<td>2</td>
<td>72.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>10</td>
<td>JHT132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>11</td>
<td>CRY132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>64.00</td>
<td>6.60</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>12</td>
<td>LDR132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>13</td>
<td>SCA132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>14</td>
<td>GLO132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>5.00</td>
<td>2.50</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>16</td>
<td>TAF132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>18.00</td>
<td>5.40</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>17</td>
<td>CTI132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>19</td>
<td>CBL132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>10.60</td>
<td>5.30</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>20</td>
<td>PUN132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>10.00</td>
<td>3.00</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>22</td>
<td>CMT132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>23</td>
<td>PAL132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>107.30</td>
<td>36.30</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>24</td>
<td>GGL132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>6.90</td>
<td>2.60</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>25</td>
<td>ASH132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>26</td>
<td>NKR132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>9.60</td>
<td>4.80</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>27</td>
<td>HMC132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>24.00</td>
<td>7.20</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>29</td>
<td>JPT132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>30</td>
<td>VIT230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>31</td>
<td>VIT12</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>12.60</td>
<td>0.0</td>
</tr>
<tr>
<td>33</td>
<td>JPT60</td>
<td>1</td>
<td>0.0</td>
<td>0.86</td>
<td>27.80</td>
<td>15.90</td>
<td>60.00</td>
<td>0.0</td>
</tr>
<tr>
<td>34</td>
<td>JPT12</td>
<td>1</td>
<td>0.0</td>
<td>15.00</td>
<td>0.0</td>
<td>0.0</td>
<td>12.00</td>
<td>0.0</td>
</tr>
<tr>
<td>35</td>
<td>GGA132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>36</td>
<td>CFT132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>65.40</td>
<td>20.30</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>37</td>
<td>ARN230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>38</td>
<td>ARN132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>39</td>
<td>STO132</td>
<td>1</td>
<td>25.00</td>
<td>93.30</td>
<td>197.50</td>
<td>140.50</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>40</td>
<td>VIT132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>132.00</td>
<td>0.0</td>
</tr>
<tr>
<td>TX #</td>
<td>TX NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>LEAKAGE IMPEDANCE</td>
<td>S.E. TAP</td>
<td>R.E. TAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>------</td>
<td>------</td>
<td>-------------------</td>
<td>----------</td>
<td>----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SCA132</td>
<td>5</td>
<td>13</td>
<td>0.0063</td>
<td>1.13</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LDR132</td>
<td>4</td>
<td>12</td>
<td>0.0095</td>
<td>1.13</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>JHT132</td>
<td>3</td>
<td>10</td>
<td>0.0027</td>
<td>1.15</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PUN122</td>
<td>6</td>
<td>20</td>
<td>0.0353</td>
<td>1.05</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ASH132</td>
<td>7</td>
<td>25</td>
<td>0.0220</td>
<td>1.10</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>JPT 60</td>
<td>29</td>
<td>33</td>
<td>0.0110</td>
<td>1.03</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>JPT 12</td>
<td>33</td>
<td>34</td>
<td>0.0064</td>
<td>1.00</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>GGA132</td>
<td>8</td>
<td>35</td>
<td>0.0039</td>
<td>1.10</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>VIT122</td>
<td>30</td>
<td>40</td>
<td>0.0030</td>
<td>1.02</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>VIT 12</td>
<td>30</td>
<td>31</td>
<td>0.0033</td>
<td>0.98</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>VIT 12</td>
<td>30</td>
<td>31</td>
<td>0.0011</td>
<td>1.03</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ARN132</td>
<td>37</td>
<td>38</td>
<td>0.0011</td>
<td>1.03</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
<td>R.E. SUSEP.</td>
<td>TX?</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
<td>-----</td>
</tr>
<tr>
<td>15</td>
<td>TAS132</td>
<td>14</td>
<td>16</td>
<td>0.0157</td>
<td>0.0393</td>
<td>0.00880</td>
<td>0.00880</td>
<td>F</td>
</tr>
<tr>
<td>17</td>
<td>DRY132</td>
<td>10</td>
<td>11</td>
<td>0.0017</td>
<td>0.0052</td>
<td>0.00440</td>
<td>0.00440</td>
<td>F</td>
</tr>
<tr>
<td>19</td>
<td>GLD132</td>
<td>13</td>
<td>14</td>
<td>0.0516</td>
<td>0.1290</td>
<td>0.02863</td>
<td>0.02860</td>
<td>F</td>
</tr>
<tr>
<td>20</td>
<td>SCA132</td>
<td>12</td>
<td>13</td>
<td>0.0168</td>
<td>0.0448</td>
<td>0.00900</td>
<td>0.00900</td>
<td>F</td>
</tr>
<tr>
<td>21</td>
<td>LDR132</td>
<td>10</td>
<td>12</td>
<td>0.0017</td>
<td>0.0086</td>
<td>0.00840</td>
<td>0.00840</td>
<td>F</td>
</tr>
<tr>
<td>22</td>
<td>GBL132</td>
<td>17</td>
<td>19</td>
<td>0.0042</td>
<td>0.0123</td>
<td>0.00280</td>
<td>0.00280</td>
<td>F</td>
</tr>
<tr>
<td>23</td>
<td>CT1132</td>
<td>10</td>
<td>17</td>
<td>0.0021</td>
<td>0.0061</td>
<td>0.00140</td>
<td>0.00140</td>
<td>F</td>
</tr>
<tr>
<td>24</td>
<td>PUN132</td>
<td>17</td>
<td>20</td>
<td>0.0410</td>
<td>0.1209</td>
<td>0.02660</td>
<td>0.02660</td>
<td>F</td>
</tr>
<tr>
<td>25</td>
<td>DMR132</td>
<td>10</td>
<td>22</td>
<td>0.0410</td>
<td>0.1227</td>
<td>0.10420</td>
<td>0.10420</td>
<td>F</td>
</tr>
<tr>
<td>26</td>
<td>DMR132</td>
<td>20</td>
<td>22</td>
<td>0.0404</td>
<td>0.1194</td>
<td>0.02620</td>
<td>0.02620</td>
<td>F</td>
</tr>
<tr>
<td>27</td>
<td>ASH132</td>
<td>24</td>
<td>25</td>
<td>0.0161</td>
<td>0.0427</td>
<td>0.00840</td>
<td>0.00840</td>
<td>F</td>
</tr>
<tr>
<td>28</td>
<td>GCL132</td>
<td>23</td>
<td>24</td>
<td>0.0176</td>
<td>0.0458</td>
<td>0.00920</td>
<td>0.00920</td>
<td>F</td>
</tr>
<tr>
<td>29</td>
<td>PAL132</td>
<td>22</td>
<td>23</td>
<td>0.0127</td>
<td>0.0326</td>
<td>0.02760</td>
<td>0.02760</td>
<td>F</td>
</tr>
<tr>
<td>30</td>
<td>JPT132</td>
<td>22</td>
<td>29</td>
<td>0.0254</td>
<td>0.0750</td>
<td>0.06560</td>
<td>0.06560</td>
<td>F</td>
</tr>
<tr>
<td>31</td>
<td>JPT132</td>
<td>26</td>
<td>29</td>
<td>0.0165</td>
<td>0.0431</td>
<td>0.00860</td>
<td>0.00860</td>
<td>F</td>
</tr>
<tr>
<td>32</td>
<td>JPT132</td>
<td>27</td>
<td>29</td>
<td>0.0165</td>
<td>0.0431</td>
<td>0.00860</td>
<td>0.00860</td>
<td>F</td>
</tr>
<tr>
<td>33</td>
<td>STO132</td>
<td>29</td>
<td>39</td>
<td>0.0440</td>
<td>0.1299</td>
<td>0.02840</td>
<td>0.02840</td>
<td>F</td>
</tr>
<tr>
<td>34</td>
<td>CGA132</td>
<td>29</td>
<td>35</td>
<td>0.0416</td>
<td>0.1231</td>
<td>0.02680</td>
<td>0.02680</td>
<td>F</td>
</tr>
<tr>
<td>35</td>
<td>CFT112</td>
<td>35</td>
<td>36</td>
<td>0.0128</td>
<td>0.0380</td>
<td>0.00820</td>
<td>0.00820</td>
<td>F</td>
</tr>
<tr>
<td>36</td>
<td>STO132</td>
<td>36</td>
<td>39</td>
<td>0.0094</td>
<td>0.0279</td>
<td>0.00620</td>
<td>0.00620</td>
<td>F</td>
</tr>
<tr>
<td>37</td>
<td>VIT112</td>
<td>39</td>
<td>40</td>
<td>0.0034</td>
<td>0.0031</td>
<td>0.00360</td>
<td>0.00360</td>
<td>F</td>
</tr>
<tr>
<td>39</td>
<td>VIT112</td>
<td>38</td>
<td>40</td>
<td>0.0284</td>
<td>0.0643</td>
<td>0.81900</td>
<td>0.81900</td>
<td>F</td>
</tr>
<tr>
<td>40</td>
<td>ARN230</td>
<td>1</td>
<td>37</td>
<td>0.0026</td>
<td>0.0158</td>
<td>0.03060</td>
<td>0.03060</td>
<td>F</td>
</tr>
</tbody>
</table>
### Bus Information

<table>
<thead>
<tr>
<th>BUS #</th>
<th>BUS NAME</th>
<th>TYPE</th>
<th>GENERATION MW</th>
<th>MVAR</th>
<th>LOAD MW</th>
<th>MVAR</th>
<th>BASE KV</th>
<th>VOLTAGE KV</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>BUS 21</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>0.0</td>
</tr>
<tr>
<td>22</td>
<td>BUS 22</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>0.0</td>
</tr>
<tr>
<td>23</td>
<td>BUS 23</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>0.0</td>
</tr>
<tr>
<td>24</td>
<td>BUS 24</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>0.0</td>
</tr>
<tr>
<td>25</td>
<td>BUS 25</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>0.0</td>
</tr>
<tr>
<td>26</td>
<td>BUS 26</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>44.40</td>
<td>21.50</td>
<td>136.00</td>
<td>0.0</td>
</tr>
<tr>
<td>27</td>
<td>BUS 27</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>44.40</td>
<td>21.50</td>
<td>136.00</td>
<td>0.0</td>
</tr>
<tr>
<td>28</td>
<td>BUS 28</td>
<td>2</td>
<td>333.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>14.15</td>
</tr>
<tr>
<td>29</td>
<td>BUS 29</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>30</td>
<td>BUS 30</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>259.00</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>31</td>
<td>BUS 31</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>32</td>
<td>BUS 32</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>360.00</td>
<td>0.0</td>
</tr>
<tr>
<td>33</td>
<td>BUS 33</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>360.00</td>
<td>0.0</td>
</tr>
<tr>
<td>34</td>
<td>BUS 34</td>
<td>2</td>
<td>208.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>14.15</td>
</tr>
<tr>
<td>35</td>
<td>BUS 35</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>36</td>
<td>BUS 36</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>360.00</td>
<td>0.0</td>
</tr>
<tr>
<td>37</td>
<td>BUS 37</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>124.00</td>
<td>29.00</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>38</td>
<td>BUS 38</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>39</td>
<td>BUS 39</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>40</td>
<td>BUS 40</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>66.40</td>
<td>32.10</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>41</td>
<td>BUS 41</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>42</td>
<td>BUS 42</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>43</td>
<td>BUS 43</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>44</td>
<td>BUS 44</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>83.50</td>
<td>40.40</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>45</td>
<td>BUS 45</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>111.70</td>
<td>54.10</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>46</td>
<td>BUS 46</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>47</td>
<td>BUS 47</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>48</td>
<td>BUS 48</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>682.00</td>
<td>45.00</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>49</td>
<td>BUS 49</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>23.60</td>
<td>22.20</td>
<td>230.00</td>
<td>0.0</td>
</tr>
<tr>
<td>50</td>
<td>BUS 50</td>
<td>2</td>
<td>972.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>14.15</td>
</tr>
<tr>
<td>51</td>
<td>BUS 51</td>
<td>2</td>
<td>306.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>14.15</td>
</tr>
<tr>
<td>52</td>
<td>BUS 52</td>
<td>2</td>
<td>42.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>53</td>
<td>BUS 53</td>
<td>2</td>
<td>22.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>54</td>
<td>BUS 54</td>
<td>2</td>
<td>60.00</td>
<td>0.0</td>
<td>12.00</td>
<td>5.80</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>55</td>
<td>BUS 55</td>
<td>2</td>
<td>140.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>14.15</td>
</tr>
<tr>
<td>56</td>
<td>BUS 56</td>
<td>2</td>
<td>50.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>57</td>
<td>BUS 57</td>
<td>2</td>
<td>92.50</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>58</td>
<td>BUS 58</td>
<td>1</td>
<td>105.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
<td>13.80</td>
</tr>
<tr>
<td>59</td>
<td>BUS 59</td>
<td>2</td>
<td>3241.00</td>
<td>756.00</td>
<td>1955.00</td>
<td>70.00</td>
<td>230.00</td>
<td>241.30</td>
</tr>
<tr>
<td>60</td>
<td>BUS 60</td>
<td>3</td>
<td>11337.30</td>
<td>-886.50</td>
<td>9529.00</td>
<td>-3645.00</td>
<td>100.00</td>
<td>113.00</td>
</tr>
<tr>
<td>61</td>
<td>BUS 61</td>
<td>2</td>
<td>826.00</td>
<td>292.00</td>
<td>411.00</td>
<td>-24.00</td>
<td>100.00</td>
<td>110.00</td>
</tr>
<tr>
<td>62</td>
<td>BUS 62</td>
<td>2</td>
<td>936.00</td>
<td>711.00</td>
<td>389.00</td>
<td>-290.00</td>
<td>100.00</td>
<td>115.00</td>
</tr>
<tr>
<td>63</td>
<td>BUS 63</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>12.30</td>
<td>12.60</td>
</tr>
<tr>
<td>BUS #</td>
<td>BUS NAME</td>
<td>TYPE</td>
<td>GENERATION (Mw)</td>
<td>LOAD (Mw)</td>
<td>BASE (KV)</td>
<td>VOLTAGE (KV)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------</td>
<td>------</td>
<td>----------------</td>
<td>----------</td>
<td>----------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>BUS 64</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>BUS 65</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>BUS 66</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>BUS 67</td>
<td>2</td>
<td>250.00</td>
<td>200.00</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>BUS 68</td>
<td>1</td>
<td>0.0</td>
<td>70.20</td>
<td>17.50</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>BUS 69</td>
<td>1</td>
<td>0.0</td>
<td>150.00</td>
<td>61.00</td>
<td>66.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>BUS 70</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>76.87</td>
<td>12.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>BUS 71</td>
<td>1</td>
<td>0.0</td>
<td>3.70</td>
<td>1.80</td>
<td>66.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>BUS 72</td>
<td>1</td>
<td>0.0</td>
<td>9.30</td>
<td>4.50</td>
<td>66.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>BUS 73</td>
<td>1</td>
<td>0.0</td>
<td>0.40</td>
<td>0.20</td>
<td>66.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>BUS 74</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>360.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>BUS 75</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>BUS 76</td>
<td>1</td>
<td>0.0</td>
<td>140.60</td>
<td>43.50</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>BUS 77</td>
<td>1</td>
<td>0.0</td>
<td>47.20</td>
<td>16.40</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>BUS 78</td>
<td>1</td>
<td>0.0</td>
<td>31.60</td>
<td>14.50</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>BUS 79</td>
<td>1</td>
<td>0.0</td>
<td>106.80</td>
<td>49.90</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>BUS 80</td>
<td>1</td>
<td>0.0</td>
<td>3.80</td>
<td>1.30</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>BUS 81</td>
<td>1</td>
<td>0.0</td>
<td>5.70</td>
<td>1.90</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>BUS 82</td>
<td>1</td>
<td>0.0</td>
<td>81.60</td>
<td>39.50</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>BUS 83</td>
<td>1</td>
<td>0.0</td>
<td>95.20</td>
<td>46.10</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>BUS 84</td>
<td>1</td>
<td>0.0</td>
<td>110.10</td>
<td>53.30</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>BUS 85</td>
<td>1</td>
<td>0.0</td>
<td>55.70</td>
<td>69.80</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>BUS 86</td>
<td>1</td>
<td>0.0</td>
<td>20.50</td>
<td>12.30</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>BUS 87</td>
<td>1</td>
<td>0.0</td>
<td>29.00</td>
<td>12.40</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>BUS 88</td>
<td>1</td>
<td>0.0</td>
<td>711.00</td>
<td>-343.00</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>BUS 89</td>
<td>2</td>
<td>0.0</td>
<td>3152.00</td>
<td>-756.00</td>
<td>503.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>BUS 90</td>
<td>1</td>
<td>0.0</td>
<td>312.00</td>
<td>243.09</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>BUS 91</td>
<td>1</td>
<td>0.0</td>
<td>200.00</td>
<td>48.11</td>
<td>132.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>BUS 92</td>
<td>1</td>
<td>0.0</td>
<td>33.90</td>
<td>11.20</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>BUS 93</td>
<td>1</td>
<td>0.0</td>
<td>20.90</td>
<td>11.30</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>BUS 94</td>
<td>1</td>
<td>0.0</td>
<td>36.70</td>
<td>17.80</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>BUS 95</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>BUS 96</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>-69.02</td>
<td>12.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>BUS 97</td>
<td>1</td>
<td>0.0</td>
<td>18.90</td>
<td>8.10</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>BUS 98</td>
<td>1</td>
<td>0.0</td>
<td>7.90</td>
<td>3.60</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>BUS 99</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>BUS 100</td>
<td>1</td>
<td>0.0</td>
<td>85.80</td>
<td>18.90</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>BUS 101</td>
<td>1</td>
<td>0.0</td>
<td>7.30</td>
<td>3.50</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>BUS 102</td>
<td>1</td>
<td>0.0</td>
<td>9.30</td>
<td>4.50</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>BUS 103</td>
<td>1</td>
<td>0.0</td>
<td>31.30</td>
<td>15.10</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>BUS 104</td>
<td>1</td>
<td>0.0</td>
<td>8.60</td>
<td>4.20</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>BUS 105</td>
<td>1</td>
<td>0.0</td>
<td>16.40</td>
<td>28.42</td>
<td>60.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- MVAR values are not shown.
- BUS numbers 1 through 89 are listed.
- BUS numbers 90 through 107 are not listed.
- VOLTS values are not shown.
### BUS INFORMATION

<table>
<thead>
<tr>
<th>BUS #</th>
<th>BUS NAME</th>
<th>TYPE</th>
<th>GENERATION</th>
<th>LOAD</th>
<th>BASE</th>
<th>VOLTAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MW</td>
<td>MW</td>
<td>MVAR</td>
<td>KV</td>
</tr>
<tr>
<td>108</td>
<td>BUS108</td>
<td>1</td>
<td>0.0</td>
<td>36.90</td>
<td>27.86</td>
<td>60.00</td>
</tr>
<tr>
<td>109</td>
<td>BUS109</td>
<td>1</td>
<td>0.0</td>
<td>38.60</td>
<td>13.20</td>
<td>360.00</td>
</tr>
<tr>
<td>110</td>
<td>BUS110</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>-76.87</td>
<td>12.00</td>
</tr>
<tr>
<td>111</td>
<td>BUS111</td>
<td>2</td>
<td>275.00</td>
<td>0.0</td>
<td>0.0</td>
<td>13.80</td>
</tr>
<tr>
<td>112</td>
<td>BUS112</td>
<td>1</td>
<td>0.0</td>
<td>21.40</td>
<td>8.50</td>
<td>230.00</td>
</tr>
<tr>
<td>113</td>
<td>BUS113</td>
<td>1</td>
<td>0.0</td>
<td>30.30</td>
<td>10.70</td>
<td>60.00</td>
</tr>
<tr>
<td>114</td>
<td>BUS114</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>230.00</td>
</tr>
<tr>
<td>115</td>
<td>BUS115</td>
<td>1</td>
<td>0.0</td>
<td>360.00</td>
<td>360.68</td>
<td>13.80</td>
</tr>
<tr>
<td>270</td>
<td>BUS270</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>113.01</td>
</tr>
<tr>
<td>TX #</td>
<td>TX NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>LEAKAGE IMPEDANCE</td>
<td>S.E. TAP</td>
<td>R.E. TAP</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>------</td>
<td>------</td>
<td>--------------------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>96</td>
<td>21</td>
<td>28</td>
<td>0.0</td>
<td>0.0558</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>97</td>
<td>21</td>
<td>50</td>
<td>0.0</td>
<td>0.0143</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>98</td>
<td>69</td>
<td>63</td>
<td>0.0</td>
<td>0.0399</td>
<td>1.00</td>
<td>0.93</td>
</tr>
<tr>
<td>99</td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0968</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>100</td>
<td>90</td>
<td>92</td>
<td>0.0</td>
<td>0.2253</td>
<td>1.00</td>
<td>1.02</td>
</tr>
<tr>
<td>101</td>
<td>46</td>
<td>85</td>
<td>0.0</td>
<td>0.0714</td>
<td>1.00</td>
<td>1.02</td>
</tr>
<tr>
<td>102</td>
<td>46</td>
<td>85</td>
<td>0.0</td>
<td>0.0768</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td>103</td>
<td>31</td>
<td>76</td>
<td>0.0</td>
<td>0.1108</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>104</td>
<td>24</td>
<td>31</td>
<td>0.0</td>
<td>0.0193</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>105</td>
<td>35</td>
<td>82</td>
<td>0.0</td>
<td>0.2280</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>106</td>
<td>39</td>
<td>82</td>
<td>0.0</td>
<td>0.2280</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>107</td>
<td>22</td>
<td>29</td>
<td>0.0</td>
<td>0.0305</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>108</td>
<td>29</td>
<td>69</td>
<td>0.0</td>
<td>0.1112</td>
<td>1.00</td>
<td>1.05</td>
</tr>
<tr>
<td>109</td>
<td>29</td>
<td>69</td>
<td>0.0</td>
<td>0.1132</td>
<td>1.00</td>
<td>1.05</td>
</tr>
<tr>
<td>110</td>
<td>21</td>
<td>27</td>
<td>0.0</td>
<td>0.0648</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>111</td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>112</td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>113</td>
<td>96</td>
<td>91</td>
<td>0.0</td>
<td>0.0750</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>114</td>
<td>23</td>
<td>30</td>
<td>0.0</td>
<td>0.0309</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>115</td>
<td>30</td>
<td>98</td>
<td>0.0</td>
<td>0.0720</td>
<td>1.00</td>
<td>0.97</td>
</tr>
<tr>
<td>116</td>
<td>37</td>
<td>99</td>
<td>0.0</td>
<td>0.4285</td>
<td>1.00</td>
<td>0.97</td>
</tr>
<tr>
<td>117</td>
<td>37</td>
<td>99</td>
<td>0.0</td>
<td>0.4110</td>
<td>1.00</td>
<td>0.97</td>
</tr>
<tr>
<td>118</td>
<td>101</td>
<td>97</td>
<td>0.0</td>
<td>0.1147</td>
<td>1.00</td>
<td>0.91</td>
</tr>
<tr>
<td>119</td>
<td>38</td>
<td>102</td>
<td>0.0</td>
<td>0.0404</td>
<td>1.00</td>
<td>0.97</td>
</tr>
<tr>
<td>121</td>
<td>101</td>
<td>97</td>
<td>0.0</td>
<td>0.1147</td>
<td>1.00</td>
<td>0.91</td>
</tr>
<tr>
<td>123</td>
<td>22</td>
<td>29</td>
<td>0.0</td>
<td>0.0300</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>124</td>
<td>21</td>
<td>111</td>
<td>0.0</td>
<td>0.0558</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>126</td>
<td>29</td>
<td>110</td>
<td>0.0</td>
<td>0.0715</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td>128</td>
<td>44</td>
<td>83</td>
<td>0.0</td>
<td>0.1335</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>129</td>
<td>25</td>
<td>47</td>
<td>0.0</td>
<td>0.0117</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>130</td>
<td>75</td>
<td>55</td>
<td>0.0</td>
<td>0.0835</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>131</td>
<td>40</td>
<td>67</td>
<td>0.0</td>
<td>0.0154</td>
<td>1.00</td>
<td>0.95</td>
</tr>
<tr>
<td>132</td>
<td>42</td>
<td>66</td>
<td>0.0</td>
<td>0.1137</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>133</td>
<td>44</td>
<td>83</td>
<td>0.0</td>
<td>0.1333</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>134</td>
<td>46</td>
<td>85</td>
<td>0.0</td>
<td>0.0768</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td>135</td>
<td>31</td>
<td>76</td>
<td>0.0</td>
<td>0.1108</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>136</td>
<td>32</td>
<td>31</td>
<td>0.0</td>
<td>0.0198</td>
<td>1.00</td>
<td>1.08</td>
</tr>
<tr>
<td>137</td>
<td>24</td>
<td>31</td>
<td>0.0</td>
<td>0.0193</td>
<td>1.00</td>
<td>1.03</td>
</tr>
<tr>
<td>138</td>
<td>39</td>
<td>82</td>
<td>0.0</td>
<td>0.2385</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>139</td>
<td>56</td>
<td>80</td>
<td>0.0</td>
<td>0.1180</td>
<td>1.00</td>
<td>1.09</td>
</tr>
<tr>
<td>140</td>
<td>57</td>
<td>79</td>
<td>0.0</td>
<td>0.3206</td>
<td>1.00</td>
<td>1.16</td>
</tr>
<tr>
<td>141</td>
<td>58</td>
<td>77</td>
<td>0.0</td>
<td>0.1734</td>
<td>1.00</td>
<td>1.14</td>
</tr>
<tr>
<td>142</td>
<td>74</td>
<td>54</td>
<td>0.0</td>
<td>0.1948</td>
<td>1.00</td>
<td>0.97</td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>1</td>
<td>21 22</td>
<td>0.0028</td>
<td>0.0239</td>
<td>0.51290</td>
<td>0.51290</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>22 23</td>
<td>0.0031</td>
<td>0.0281</td>
<td>0.84650</td>
<td>0.84650</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>23 24</td>
<td>0.0030</td>
<td>0.0410</td>
<td>0.78113</td>
<td>2.03560</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td>24 25</td>
<td>0.0004</td>
<td>0.0056</td>
<td>0.19344</td>
<td>0.19344</td>
<td>F</td>
</tr>
<tr>
<td>5</td>
<td>32 109</td>
<td>0.0023</td>
<td>0.0218</td>
<td>0.20660</td>
<td>0.20660</td>
<td>F</td>
</tr>
<tr>
<td>6</td>
<td>33 36</td>
<td>0.0067</td>
<td>0.0653</td>
<td>0.63040</td>
<td>0.63040</td>
<td>F</td>
</tr>
<tr>
<td>7</td>
<td>36 74</td>
<td>0.0003</td>
<td>0.0031</td>
<td>0.02710</td>
<td>0.02710</td>
<td>F</td>
</tr>
<tr>
<td>8</td>
<td>35 37</td>
<td>0.0221</td>
<td>0.0768</td>
<td>0.11510</td>
<td>0.11510</td>
<td>F</td>
</tr>
<tr>
<td>9</td>
<td>30 39</td>
<td>0.0008</td>
<td>0.0048</td>
<td>0.00440</td>
<td>0.00440</td>
<td>F</td>
</tr>
<tr>
<td>10</td>
<td>31 46</td>
<td>0.0032</td>
<td>0.0195</td>
<td>0.01890</td>
<td>0.01890</td>
<td>F</td>
</tr>
<tr>
<td>11</td>
<td>31 90</td>
<td>0.0026</td>
<td>0.0158</td>
<td>0.01540</td>
<td>0.01540</td>
<td>F</td>
</tr>
<tr>
<td>12</td>
<td>31 47</td>
<td>0.0055</td>
<td>0.0330</td>
<td>0.03190</td>
<td>0.03190</td>
<td>F</td>
</tr>
<tr>
<td>13</td>
<td>37 39</td>
<td>0.0109</td>
<td>0.0651</td>
<td>0.05750</td>
<td>0.05750</td>
<td>F</td>
</tr>
<tr>
<td>14</td>
<td>39 40</td>
<td>0.0021</td>
<td>0.0131</td>
<td>0.01290</td>
<td>0.01290</td>
<td>F</td>
</tr>
<tr>
<td>15</td>
<td>37 75</td>
<td>0.0030</td>
<td>0.0190</td>
<td>0.01700</td>
<td>0.01700</td>
<td>F</td>
</tr>
<tr>
<td>16</td>
<td>37 97</td>
<td>0.0083</td>
<td>0.0573</td>
<td>0.05560</td>
<td>0.05560</td>
<td>F</td>
</tr>
<tr>
<td>17</td>
<td>40 43</td>
<td>0.0035</td>
<td>0.0240</td>
<td>0.02320</td>
<td>0.02320</td>
<td>F</td>
</tr>
<tr>
<td>18</td>
<td>41 43</td>
<td>0.0014</td>
<td>0.0087</td>
<td>0.00650</td>
<td>0.00650</td>
<td>F</td>
</tr>
<tr>
<td>19</td>
<td>42 43</td>
<td>0.0002</td>
<td>0.0017</td>
<td>0.00140</td>
<td>0.00140</td>
<td>F</td>
</tr>
<tr>
<td>21</td>
<td>46 96</td>
<td>0.0018</td>
<td>0.0052</td>
<td>0.36700</td>
<td>0.36700</td>
<td>F</td>
</tr>
<tr>
<td>22</td>
<td>31 42</td>
<td>0.0013</td>
<td>0.0079</td>
<td>0.00770</td>
<td>0.00770</td>
<td>F</td>
</tr>
<tr>
<td>23</td>
<td>40 41</td>
<td>0.0015</td>
<td>0.0046</td>
<td>0.00370</td>
<td>0.00370</td>
<td>F</td>
</tr>
<tr>
<td>24</td>
<td>39 44</td>
<td>0.0004</td>
<td>0.0028</td>
<td>0.19730</td>
<td>0.19730</td>
<td>F</td>
</tr>
<tr>
<td>25</td>
<td>39 40</td>
<td>0.0024</td>
<td>0.0155</td>
<td>0.01520</td>
<td>0.01520</td>
<td>F</td>
</tr>
<tr>
<td>26</td>
<td>44 45</td>
<td>0.0005</td>
<td>0.0028</td>
<td>0.02020</td>
<td>0.02020</td>
<td>F</td>
</tr>
<tr>
<td>27</td>
<td>30 35</td>
<td>0.0102</td>
<td>0.0704</td>
<td>0.06800</td>
<td>0.06800</td>
<td>F</td>
</tr>
<tr>
<td>28</td>
<td>30 49</td>
<td>0.0203</td>
<td>0.1401</td>
<td>0.13560</td>
<td>0.13560</td>
<td>F</td>
</tr>
<tr>
<td>29</td>
<td>29 113</td>
<td>0.0122</td>
<td>0.0649</td>
<td>0.06320</td>
<td>0.06320</td>
<td>F</td>
</tr>
<tr>
<td>30</td>
<td>29 68</td>
<td>0.0077</td>
<td>0.0467</td>
<td>0.03650</td>
<td>0.03650</td>
<td>F</td>
</tr>
<tr>
<td>31</td>
<td>82 83</td>
<td>0.0111</td>
<td>0.0417</td>
<td>0.02380</td>
<td>0.02380</td>
<td>F</td>
</tr>
<tr>
<td>32</td>
<td>83 84</td>
<td>0.0107</td>
<td>0.0394</td>
<td>0.02280</td>
<td>0.02280</td>
<td>F</td>
</tr>
<tr>
<td>33</td>
<td>84 85</td>
<td>0.0113</td>
<td>0.0417</td>
<td>0.00140</td>
<td>0.00140</td>
<td>F</td>
</tr>
<tr>
<td>34</td>
<td>84 85</td>
<td>0.0107</td>
<td>0.0394</td>
<td>0.02280</td>
<td>0.02280</td>
<td>F</td>
</tr>
<tr>
<td>35</td>
<td>79 82</td>
<td>0.0124</td>
<td>0.0481</td>
<td>0.00120</td>
<td>0.00120</td>
<td>F</td>
</tr>
<tr>
<td>36</td>
<td>85 87</td>
<td>0.0542</td>
<td>0.1679</td>
<td>0.00110</td>
<td>0.00110</td>
<td>F</td>
</tr>
<tr>
<td>37</td>
<td>79 86</td>
<td>0.0356</td>
<td>0.1255</td>
<td>0.00080</td>
<td>0.00080</td>
<td>F</td>
</tr>
<tr>
<td>38</td>
<td>86 87</td>
<td>0.0089</td>
<td>0.0347</td>
<td>0.00220</td>
<td>0.00220</td>
<td>F</td>
</tr>
<tr>
<td>39</td>
<td>80 81</td>
<td>0.1233</td>
<td>0.2051</td>
<td>0.01000</td>
<td>0.01000</td>
<td>F</td>
</tr>
<tr>
<td>40</td>
<td>81 82</td>
<td>0.0543</td>
<td>0.0980</td>
<td>0.00050</td>
<td>0.00050</td>
<td>F</td>
</tr>
<tr>
<td>41</td>
<td>79 80</td>
<td>0.0632</td>
<td>0.1151</td>
<td>0.00230</td>
<td>0.00230</td>
<td>F</td>
</tr>
<tr>
<td>42</td>
<td>76 79</td>
<td>0.0068</td>
<td>0.0130</td>
<td>0.00130</td>
<td>0.00130</td>
<td>F</td>
</tr>
<tr>
<td>43</td>
<td>78 79</td>
<td>0.0535</td>
<td>0.1697</td>
<td>0.00850</td>
<td>0.00850</td>
<td>F</td>
</tr>
<tr>
<td>44</td>
<td>77 78</td>
<td>0.0226</td>
<td>0.0682</td>
<td>0.00040</td>
<td>0.00040</td>
<td>F</td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E. R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
<td>R.E. SUSEP.</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>76 77</td>
<td>0.1539</td>
<td>0.4232</td>
<td>0.00280</td>
<td>0.00280</td>
</tr>
<tr>
<td>46</td>
<td></td>
<td>72 73</td>
<td>0.1457</td>
<td>0.2555</td>
<td>0.00220</td>
<td>0.00220</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td>71 73</td>
<td>0.2248</td>
<td>0.6099</td>
<td>0.00480</td>
<td>0.00340</td>
</tr>
<tr>
<td>48</td>
<td></td>
<td>61 88</td>
<td>-15.3670</td>
<td>36.6340</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>49</td>
<td></td>
<td>47 48</td>
<td>0.0029</td>
<td>0.0178</td>
<td>0.00570</td>
<td>0.00570</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>48 60</td>
<td>-0.2625</td>
<td>0.4722</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>51</td>
<td></td>
<td>62 89</td>
<td>-21.4680</td>
<td>16.0660</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>52</td>
<td></td>
<td>60 88</td>
<td>-0.0732</td>
<td>0.1962</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>53</td>
<td></td>
<td>88 89</td>
<td>-0.8718</td>
<td>2.4920</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>54</td>
<td></td>
<td>48 59</td>
<td>-0.0174</td>
<td>0.2617</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>55</td>
<td></td>
<td>59 88</td>
<td>0.0038</td>
<td>0.0348</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>56</td>
<td></td>
<td>48 89</td>
<td>-0.0131</td>
<td>0.1389</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>57</td>
<td></td>
<td>62 88</td>
<td>-0.0207</td>
<td>0.1314</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>58</td>
<td></td>
<td>59 60</td>
<td>-0.0096</td>
<td>0.0388</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>59</td>
<td></td>
<td>60 62</td>
<td>-1.0183</td>
<td>0.7056</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>60</td>
<td></td>
<td>59 89</td>
<td>-0.0036</td>
<td>0.0509</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>61</td>
<td></td>
<td>48 61</td>
<td>-0.0181</td>
<td>0.2965</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>62</td>
<td></td>
<td>61 89</td>
<td>-0.0397</td>
<td>0.1729</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>227</td>
<td></td>
<td>25 89</td>
<td>0.0017</td>
<td>0.0363</td>
<td>0.88630</td>
<td>0.88600</td>
</tr>
<tr>
<td>65</td>
<td></td>
<td>59 61</td>
<td>-0.0695</td>
<td>0.3615</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>66</td>
<td></td>
<td>60 89</td>
<td>-0.0144</td>
<td>0.0303</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>67</td>
<td></td>
<td>60 61</td>
<td>-0.3306</td>
<td>0.4649</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>201</td>
<td></td>
<td>21 22</td>
<td>0.0028</td>
<td>0.0239</td>
<td>0.51290</td>
<td>0.51290</td>
</tr>
<tr>
<td>202</td>
<td></td>
<td>22 23</td>
<td>0.0031</td>
<td>0.0281</td>
<td>0.04650</td>
<td>0.04650</td>
</tr>
<tr>
<td>203</td>
<td></td>
<td>35 37</td>
<td>0.0104</td>
<td>0.0592</td>
<td>0.15280</td>
<td>0.15280</td>
</tr>
<tr>
<td>204</td>
<td></td>
<td>31 90</td>
<td>0.0027</td>
<td>0.0137</td>
<td>0.01950</td>
<td>0.01950</td>
</tr>
<tr>
<td>205</td>
<td></td>
<td>31 90</td>
<td>0.0026</td>
<td>0.0158</td>
<td>0.01520</td>
<td>0.01520</td>
</tr>
<tr>
<td>206</td>
<td></td>
<td>30 35</td>
<td>0.0100</td>
<td>0.0690</td>
<td>0.06670</td>
<td>0.06670</td>
</tr>
<tr>
<td>207</td>
<td></td>
<td>82 93</td>
<td>0.0113</td>
<td>0.0425</td>
<td>0.02420</td>
<td>0.02420</td>
</tr>
<tr>
<td>208</td>
<td></td>
<td>76 77</td>
<td>0.1537</td>
<td>0.4226</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>209</td>
<td></td>
<td>25 95</td>
<td>0.0038</td>
<td>0.0267</td>
<td>0.02590</td>
<td>0.02590</td>
</tr>
<tr>
<td>210</td>
<td></td>
<td>41 96</td>
<td>0.3012</td>
<td>0.0380</td>
<td>0.68700</td>
<td>0.68700</td>
</tr>
<tr>
<td>211</td>
<td></td>
<td>31 45</td>
<td>0.0033</td>
<td>0.0204</td>
<td>0.07320</td>
<td>0.07320</td>
</tr>
<tr>
<td>212</td>
<td></td>
<td>38 97</td>
<td>0.0030</td>
<td>0.0194</td>
<td>0.01899</td>
<td>0.01899</td>
</tr>
<tr>
<td>213</td>
<td></td>
<td>23 24</td>
<td>0.0227</td>
<td>0.0353</td>
<td>0.48223</td>
<td>1.70670</td>
</tr>
<tr>
<td>214</td>
<td></td>
<td>99 100</td>
<td>0.5305</td>
<td>0.7527</td>
<td>0.03360</td>
<td>0.03360</td>
</tr>
<tr>
<td>215</td>
<td></td>
<td>100 101</td>
<td>0.0799</td>
<td>0.1183</td>
<td>0.00360</td>
<td>0.00360</td>
</tr>
<tr>
<td>216</td>
<td></td>
<td>101 103</td>
<td>0.3042</td>
<td>0.0164</td>
<td>0.00010</td>
<td>0.00010</td>
</tr>
<tr>
<td>217</td>
<td></td>
<td>103 104</td>
<td>0.0164</td>
<td>0.0642</td>
<td>0.00050</td>
<td>0.00050</td>
</tr>
<tr>
<td>218</td>
<td></td>
<td>101 106</td>
<td>0.0236</td>
<td>0.0786</td>
<td>0.00060</td>
<td>0.00060</td>
</tr>
<tr>
<td>219</td>
<td></td>
<td>101 105</td>
<td>0.0128</td>
<td>0.0498</td>
<td>0.00030</td>
<td>0.00030</td>
</tr>
<tr>
<td>220</td>
<td></td>
<td>105 102</td>
<td>0.0250</td>
<td>0.0922</td>
<td>0.00060</td>
<td>0.00060</td>
</tr>
<tr>
<td>221</td>
<td></td>
<td>106 107</td>
<td>0.0095</td>
<td>0.0350</td>
<td>0.00020</td>
<td>0.00020</td>
</tr>
</tbody>
</table>
**LINE INFORMATION:**

<table>
<thead>
<tr>
<th>LINE #</th>
<th>LINE NAME</th>
<th>S.E.</th>
<th>R.E.</th>
<th>PER UNIT R</th>
<th>PER UNIT X</th>
<th>S.E. SUSEP.</th>
<th>R.E. SUSEP.</th>
<th>TX?</th>
</tr>
</thead>
<tbody>
<tr>
<td>222</td>
<td></td>
<td>107</td>
<td>108</td>
<td>0.0143</td>
<td>0.0529</td>
<td>0.00030</td>
<td>0.00030</td>
<td>F</td>
</tr>
<tr>
<td>223</td>
<td></td>
<td>108</td>
<td>102</td>
<td>0.0082</td>
<td>0.0322</td>
<td>0.00023</td>
<td>0.00020</td>
<td>F</td>
</tr>
<tr>
<td>224</td>
<td></td>
<td>108</td>
<td>102</td>
<td>0.0088</td>
<td>0.0343</td>
<td>0.00020</td>
<td>0.00020</td>
<td>F</td>
</tr>
<tr>
<td>225</td>
<td></td>
<td>48</td>
<td>88</td>
<td>-9.7693</td>
<td>35.6993</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>226</td>
<td></td>
<td>59</td>
<td>112</td>
<td>-2.4809</td>
<td>8.9636</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>228</td>
<td></td>
<td>25</td>
<td>89</td>
<td>0.0017</td>
<td>0.0363</td>
<td>0.08600</td>
<td>0.08600</td>
<td>F</td>
</tr>
<tr>
<td>229</td>
<td></td>
<td>49</td>
<td>113</td>
<td>0.0114</td>
<td>0.0794</td>
<td>0.07730</td>
<td>0.07700</td>
<td>F</td>
</tr>
<tr>
<td>230</td>
<td></td>
<td>30</td>
<td>112</td>
<td>0.0115</td>
<td>0.0800</td>
<td>0.07760</td>
<td>0.07760</td>
<td>F</td>
</tr>
<tr>
<td>231</td>
<td></td>
<td>46</td>
<td>90</td>
<td>0.0023</td>
<td>0.0146</td>
<td>0.01440</td>
<td>0.01440</td>
<td>F</td>
</tr>
<tr>
<td>232</td>
<td></td>
<td>36</td>
<td>109</td>
<td>0.0005</td>
<td>0.0052</td>
<td>0.00900</td>
<td>0.00900</td>
<td>F</td>
</tr>
<tr>
<td>233</td>
<td></td>
<td>47</td>
<td>114</td>
<td>0.0012</td>
<td>0.0103</td>
<td>0.01050</td>
<td>0.01050</td>
<td>F</td>
</tr>
<tr>
<td>234</td>
<td></td>
<td>47</td>
<td>114</td>
<td>0.0011</td>
<td>0.0097</td>
<td>0.00980</td>
<td>0.00980</td>
<td>F</td>
</tr>
</tbody>
</table>
### B.3 Data for Test System 10

<table>
<thead>
<tr>
<th>BUS #</th>
<th>BUS NAME TYPE</th>
<th>LOAD MW</th>
<th>VOLTAGE KV</th>
<th>GENERATION MW</th>
<th>BASE KV</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>02</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>03</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>04</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>05</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>06</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>07</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>08</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>09</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>10</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>11</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>12</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>13</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>14</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>15</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>16</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>17</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>18</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>19</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>20</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>21</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>22</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>23</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>24</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>25</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>26</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>27</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>28</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>29</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>30</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>31</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>32</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>33</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>34</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>35</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>36</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>37</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>38</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>39</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>40</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>41</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>42</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>43</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>44</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>45</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>46</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>47</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>48</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>49</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>50</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>51</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>52</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>53</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>54</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>55</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>56</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>57</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>58</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>59</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>60</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>61</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>62</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>63</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>64</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>65</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>66</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
<tr>
<td>67</td>
<td>CHS 500</td>
<td>0.0</td>
<td>24.60</td>
<td>0.0</td>
<td>24.60</td>
</tr>
</tbody>
</table>

**NOTE:** Data is for testing purposes only.
### BUS INFORMATION

<table>
<thead>
<tr>
<th>BUS #</th>
<th>BUS NAME</th>
<th>TYPE</th>
<th>GENERATION</th>
<th>LOAD</th>
<th>BASE</th>
<th>VOLTAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MW</td>
<td>MW</td>
<td>MW</td>
<td>MW</td>
</tr>
<tr>
<td>68</td>
<td>GLN138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>69</td>
<td>WSN 66</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>70</td>
<td>WSNRZ2</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>71</td>
<td>LAJ 66</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>72</td>
<td>SCN 66</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>73</td>
<td>BRT 66</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>74</td>
<td>WSN136</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>75</td>
<td>CMS230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>76</td>
<td>ING 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>77</td>
<td>RUS 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>78</td>
<td>SFL 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>79</td>
<td>BND 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>80</td>
<td>LBN 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>81</td>
<td>SHL 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>82</td>
<td>HPN 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>83</td>
<td>MUR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>84</td>
<td>SPG 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>85</td>
<td>KL 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>86</td>
<td>NMR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>87</td>
<td>RYL 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>90</td>
<td>ARN230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>91</td>
<td>ARN132</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>92</td>
<td>ARN 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>95</td>
<td>SVY230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>96</td>
<td>CSN230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>97</td>
<td>CYP230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>98</td>
<td>KLYR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>99</td>
<td>CKY 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>100</td>
<td>HSB 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>101</td>
<td>CYP 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>102</td>
<td>WLT 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>103</td>
<td>MCT 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>104</td>
<td>GLR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>105</td>
<td>CAP 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>106</td>
<td>JLN 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>107</td>
<td>NDR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>108</td>
<td>NVR 60</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>109</td>
<td>ATZ360</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>110</td>
<td>WSN273</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>111</td>
<td>GMS 13</td>
<td>2</td>
<td>825.0</td>
<td>0.0</td>
<td>0.0</td>
<td>14.5</td>
</tr>
<tr>
<td>112</td>
<td>HM1230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>22.70</td>
<td>230.00</td>
</tr>
<tr>
<td>113</td>
<td>RLW230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>42.70</td>
<td>230.00</td>
</tr>
<tr>
<td>BUS #</td>
<td>BUS NAME</td>
<td>TYPE</td>
<td>GENERATION</td>
<td>LOAD</td>
<td>BASE</td>
<td>VOLTAGE</td>
</tr>
<tr>
<td>-------</td>
<td>----------</td>
<td>------</td>
<td>------------</td>
<td>------</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MW</td>
<td>MVAR</td>
<td>MW</td>
<td>MVAR</td>
</tr>
<tr>
<td>116</td>
<td>SVA230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>117</td>
<td>SVA138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>7.60</td>
<td>2.50</td>
</tr>
<tr>
<td>118</td>
<td>KWD138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>119</td>
<td>DUG138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>29.00</td>
<td>9.50</td>
</tr>
<tr>
<td>120</td>
<td>VVW138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>111.10</td>
<td>36.50</td>
</tr>
<tr>
<td>121</td>
<td>SAM138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>51.40</td>
<td>16.90</td>
</tr>
<tr>
<td>122</td>
<td>VNT138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>91.30</td>
<td>30.00</td>
</tr>
<tr>
<td>123</td>
<td>WSK138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>11.40</td>
<td>3.80</td>
</tr>
<tr>
<td>124</td>
<td>WSH138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>125</td>
<td>WSH 13</td>
<td>2</td>
<td>57.60</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>126</td>
<td>SLN138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>127</td>
<td>SLN 7</td>
<td>2</td>
<td>198.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>128</td>
<td>REL138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>132</td>
<td>MLD138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>206.90</td>
<td>45.90</td>
</tr>
<tr>
<td>133</td>
<td>NIC138</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>74.10</td>
<td>19.60</td>
</tr>
<tr>
<td>134</td>
<td>NIC230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>135</td>
<td>NIC500</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>136</td>
<td>MCA500</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>137</td>
<td>MCA 1</td>
<td>2</td>
<td>300.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>138</td>
<td>VNT230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>139</td>
<td>KCM230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>140</td>
<td>KCN 13</td>
<td>2</td>
<td>375.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>141</td>
<td>SM1230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>142</td>
<td>REL161</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>143</td>
<td>BLT 13</td>
<td>2</td>
<td>120.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>144</td>
<td>BLT 69</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>35.00</td>
<td>15.00</td>
</tr>
<tr>
<td>145</td>
<td>CLV161</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>146</td>
<td>SSL 69</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>172.00</td>
<td>31.00</td>
</tr>
<tr>
<td>148</td>
<td>TRL161</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>149</td>
<td>TRL 69</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>500.00</td>
<td>210.00</td>
</tr>
<tr>
<td>150</td>
<td>WAN 14</td>
<td>2</td>
<td>360.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>151</td>
<td>WAN230</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>152</td>
<td>WAN 69</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>153</td>
<td>GLN500</td>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>154</td>
<td>CBK230</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>234.70</td>
<td>84.70</td>
</tr>
<tr>
<td>155</td>
<td>GMS 9</td>
<td>2</td>
<td>300.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>156</td>
<td>MCA 2</td>
<td>2</td>
<td>300.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>251</td>
<td>CST230</td>
<td>2</td>
<td>12.70</td>
<td>6.20</td>
<td>1507.14</td>
<td>130.63</td>
</tr>
<tr>
<td>252</td>
<td>CST500</td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>5.56</td>
<td>-42.19</td>
</tr>
<tr>
<td>253</td>
<td>SCN 138</td>
<td>2</td>
<td>192.00</td>
<td>18.00</td>
<td>3460.70</td>
<td>-790.68</td>
</tr>
<tr>
<td>254</td>
<td>JGC230</td>
<td>2</td>
<td>3497.10</td>
<td>507.20</td>
<td>2266.02</td>
<td>30.26</td>
</tr>
<tr>
<td>255</td>
<td>REL230</td>
<td>2</td>
<td>104.60</td>
<td>193.20</td>
<td>2144.87</td>
<td>-250.98</td>
</tr>
<tr>
<td>KV</td>
<td>MM</td>
<td>MW</td>
<td>MW</td>
<td>MV</td>
<td>BASE</td>
<td>LOAD</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>120.74</td>
<td>100.00</td>
<td>07.40</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
<tr>
<td>19.00</td>
<td>103.00</td>
<td>10.00</td>
<td>01.50</td>
<td>00.00</td>
<td>10.00</td>
<td>00.00</td>
</tr>
</tbody>
</table>

---

**BUS INFORMATION:**

**BUS # BUS NAME TYPE**

**AEC-3T5, DATA=CH-138, START=4 09:25 P.M., JULY 24, 1977**

---

**PAGE 4**
### Transformer Information

<table>
<thead>
<tr>
<th>TX #</th>
<th>TX Name</th>
<th>S.E.</th>
<th>R.E.</th>
<th>Leakage Impedance</th>
<th>S.E. Tap</th>
<th>R.E. Tap</th>
</tr>
</thead>
<tbody>
<tr>
<td>311</td>
<td></td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.00</td>
</tr>
<tr>
<td>312</td>
<td></td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0500</td>
<td>1.00</td>
</tr>
<tr>
<td>313</td>
<td></td>
<td>90</td>
<td>91</td>
<td>0.0</td>
<td>0.0550</td>
<td>1.00</td>
</tr>
<tr>
<td>314</td>
<td></td>
<td>23</td>
<td>30</td>
<td>0.0</td>
<td>0.0309</td>
<td>1.00</td>
</tr>
<tr>
<td>315</td>
<td></td>
<td>30</td>
<td>98</td>
<td>0.0</td>
<td>0.0720</td>
<td>1.00</td>
</tr>
<tr>
<td>316</td>
<td></td>
<td>37</td>
<td>99</td>
<td>0.0</td>
<td>0.4285</td>
<td>1.00</td>
</tr>
<tr>
<td>317</td>
<td></td>
<td>37</td>
<td>99</td>
<td>0.0</td>
<td>0.4110</td>
<td>1.00</td>
</tr>
<tr>
<td>318</td>
<td></td>
<td>101</td>
<td>97</td>
<td>0.0</td>
<td>0.1147</td>
<td>1.00</td>
</tr>
<tr>
<td>319</td>
<td></td>
<td>38</td>
<td>102</td>
<td>0.0</td>
<td>0.0404</td>
<td>1.00</td>
</tr>
<tr>
<td>321</td>
<td></td>
<td>101</td>
<td>97</td>
<td>0.0</td>
<td>0.1147</td>
<td>1.00</td>
</tr>
<tr>
<td>322</td>
<td></td>
<td>22</td>
<td>29</td>
<td>0.0</td>
<td>0.0300</td>
<td>1.00</td>
</tr>
<tr>
<td>324</td>
<td></td>
<td>21</td>
<td>111</td>
<td>0.0</td>
<td>0.0186</td>
<td>1.00</td>
</tr>
<tr>
<td>326</td>
<td></td>
<td>29</td>
<td>110</td>
<td>0.0</td>
<td>0.0715</td>
<td>1.00</td>
</tr>
<tr>
<td>344</td>
<td></td>
<td>153</td>
<td>68</td>
<td>0.0</td>
<td>0.0300</td>
<td>1.00</td>
</tr>
<tr>
<td>328</td>
<td></td>
<td>116</td>
<td>117</td>
<td>0.0</td>
<td>0.2233</td>
<td>1.00</td>
</tr>
<tr>
<td>329</td>
<td></td>
<td>124</td>
<td>125</td>
<td>0.0</td>
<td>0.2343</td>
<td>1.00</td>
</tr>
<tr>
<td>330</td>
<td></td>
<td>147</td>
<td>126</td>
<td>0.0</td>
<td>0.1010</td>
<td>1.00</td>
</tr>
<tr>
<td>343</td>
<td></td>
<td>156</td>
<td>136</td>
<td>0.0</td>
<td>0.2073</td>
<td>1.00</td>
</tr>
<tr>
<td>332</td>
<td></td>
<td>24</td>
<td>31</td>
<td>0.0</td>
<td>0.0193</td>
<td>1.00</td>
</tr>
<tr>
<td>348</td>
<td></td>
<td>47</td>
<td>48</td>
<td>0.0</td>
<td>0.0129</td>
<td>1.00</td>
</tr>
<tr>
<td>349</td>
<td></td>
<td>47</td>
<td>48</td>
<td>0.0</td>
<td>0.0129</td>
<td>1.00</td>
</tr>
<tr>
<td>353</td>
<td></td>
<td>21</td>
<td>155</td>
<td>0.0</td>
<td>0.0465</td>
<td>1.00</td>
</tr>
<tr>
<td>333</td>
<td></td>
<td>135</td>
<td>134</td>
<td>0.0</td>
<td>0.0151</td>
<td>1.00</td>
</tr>
<tr>
<td>334</td>
<td></td>
<td>134</td>
<td>133</td>
<td>0.0</td>
<td>0.0450</td>
<td>1.00</td>
</tr>
<tr>
<td>335</td>
<td></td>
<td>139</td>
<td>140</td>
<td>0.0</td>
<td>0.0310</td>
<td>1.00</td>
</tr>
<tr>
<td>336</td>
<td></td>
<td>138</td>
<td>122</td>
<td>0.0</td>
<td>0.0450</td>
<td>1.00</td>
</tr>
<tr>
<td>337</td>
<td></td>
<td>137</td>
<td>136</td>
<td>0.0</td>
<td>0.0073</td>
<td>1.00</td>
</tr>
<tr>
<td>338</td>
<td></td>
<td>142</td>
<td>128</td>
<td>0.0</td>
<td>0.1480</td>
<td>1.00</td>
</tr>
<tr>
<td>339</td>
<td></td>
<td>143</td>
<td>144</td>
<td>0.0</td>
<td>0.0439</td>
<td>1.00</td>
</tr>
<tr>
<td>340</td>
<td></td>
<td>127</td>
<td>147</td>
<td>0.0</td>
<td>0.0167</td>
<td>1.00</td>
</tr>
<tr>
<td>341</td>
<td></td>
<td>148</td>
<td>149</td>
<td>0.0</td>
<td>0.1540</td>
<td>1.00</td>
</tr>
<tr>
<td>342</td>
<td></td>
<td>150</td>
<td>152</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>343</td>
<td></td>
<td>153</td>
<td>149</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>344</td>
<td></td>
<td>154</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>345</td>
<td></td>
<td>155</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>346</td>
<td></td>
<td>156</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>347</td>
<td></td>
<td>157</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>348</td>
<td></td>
<td>158</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>349</td>
<td></td>
<td>159</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>350</td>
<td></td>
<td>160</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>351</td>
<td></td>
<td>161</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>352</td>
<td></td>
<td>162</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>353</td>
<td></td>
<td>163</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>354</td>
<td></td>
<td>164</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>355</td>
<td></td>
<td>165</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>356</td>
<td></td>
<td>166</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>357</td>
<td></td>
<td>167</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>358</td>
<td></td>
<td>168</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>359</td>
<td></td>
<td>169</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>360</td>
<td></td>
<td>170</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>361</td>
<td></td>
<td>171</td>
<td>145</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>362</td>
<td></td>
<td>172</td>
<td>146</td>
<td>0.0</td>
<td>0.0181</td>
<td>1.00</td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>-------------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>21</td>
<td>22</td>
<td>0.0028</td>
<td>0.0179</td>
<td>0.51290</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>22</td>
<td>23</td>
<td>0.0031</td>
<td>0.0211</td>
<td>0.84650</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>23</td>
<td>24</td>
<td>0.0030</td>
<td>0.0205</td>
<td>0.70100</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>24</td>
<td>25</td>
<td>0.0034</td>
<td>0.0056</td>
<td>0.19340</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>32</td>
<td>109</td>
<td>0.0023</td>
<td>0.0218</td>
<td>0.20660</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>33</td>
<td>36</td>
<td>0.0067</td>
<td>0.0653</td>
<td>0.63040</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>36</td>
<td>76</td>
<td>0.0033</td>
<td>0.0031</td>
<td>0.02710</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>35</td>
<td>37</td>
<td>0.0221</td>
<td>0.0768</td>
<td>0.11150</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>38</td>
<td>39</td>
<td>0.0008</td>
<td>0.0049</td>
<td>0.02290</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>31</td>
<td>46</td>
<td>0.0032</td>
<td>0.0195</td>
<td>0.01890</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>31</td>
<td>60</td>
<td>0.0026</td>
<td>0.0108</td>
<td>0.01540</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>24</td>
<td>25</td>
<td>0.3034</td>
<td>0.0047</td>
<td>0.22700</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>37</td>
<td>38</td>
<td>0.0095</td>
<td>0.0573</td>
<td>0.05610</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>38</td>
<td>48</td>
<td>0.0037</td>
<td>0.0236</td>
<td>0.02320</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>37</td>
<td>75</td>
<td>0.0030</td>
<td>0.0190</td>
<td>0.01730</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>37</td>
<td>97</td>
<td>0.0083</td>
<td>0.0573</td>
<td>0.05560</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>42</td>
<td>43</td>
<td>0.0034</td>
<td>0.0222</td>
<td>0.00220</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>42</td>
<td>48</td>
<td>0.0022</td>
<td>0.0153</td>
<td>0.00140</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>41</td>
<td>48</td>
<td>0.0035</td>
<td>0.0218</td>
<td>0.00980</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>46</td>
<td>96</td>
<td>0.0018</td>
<td>0.0052</td>
<td>0.03670</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>31</td>
<td>43</td>
<td>0.0009</td>
<td>0.0056</td>
<td>0.00550</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>40</td>
<td>41</td>
<td>0.0015</td>
<td>0.0096</td>
<td>0.00770</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>44</td>
<td>48</td>
<td>0.0048</td>
<td>0.0303</td>
<td>0.25880</td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>39</td>
<td>40</td>
<td>0.0024</td>
<td>0.0155</td>
<td>0.01520</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td>44</td>
<td>45</td>
<td>0.0005</td>
<td>0.0028</td>
<td>0.02060</td>
</tr>
<tr>
<td>26</td>
<td></td>
<td>30</td>
<td>35</td>
<td>0.0022</td>
<td>0.0704</td>
<td>0.05800</td>
</tr>
<tr>
<td>27</td>
<td></td>
<td>30</td>
<td>49</td>
<td>0.0203</td>
<td>0.1401</td>
<td>0.13580</td>
</tr>
<tr>
<td>28</td>
<td></td>
<td>29</td>
<td>113</td>
<td>0.0122</td>
<td>0.0849</td>
<td>0.09820</td>
</tr>
<tr>
<td>29</td>
<td></td>
<td>22</td>
<td>153</td>
<td>0.0016</td>
<td>0.0217</td>
<td>0.20710</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>31</td>
<td>83</td>
<td>0.0111</td>
<td>0.4417</td>
<td>0.25880</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>83</td>
<td>86</td>
<td>0.3107</td>
<td>0.0394</td>
<td>0.30780</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>84</td>
<td>85</td>
<td>0.3113</td>
<td>0.0417</td>
<td>0.10340</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td>76</td>
<td>85</td>
<td>0.0567</td>
<td>0.2352</td>
<td>0.02140</td>
</tr>
<tr>
<td>34</td>
<td></td>
<td>79</td>
<td>82</td>
<td>0.0124</td>
<td>0.0481</td>
<td>0.00120</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td>85</td>
<td>87</td>
<td>0.0452</td>
<td>0.1679</td>
<td>0.00110</td>
</tr>
<tr>
<td>36</td>
<td></td>
<td>79</td>
<td>86</td>
<td>0.0356</td>
<td>0.1255</td>
<td>0.02080</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>86</td>
<td>87</td>
<td>0.0089</td>
<td>0.3475</td>
<td>0.00020</td>
</tr>
<tr>
<td>38</td>
<td></td>
<td>80</td>
<td>81</td>
<td>0.1233</td>
<td>0.2051</td>
<td>0.00100</td>
</tr>
<tr>
<td>39</td>
<td></td>
<td>81</td>
<td>82</td>
<td>0.0543</td>
<td>0.0980</td>
<td>0.00200</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>79</td>
<td>80</td>
<td>0.0632</td>
<td>0.1151</td>
<td>0.00290</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td>76</td>
<td>79</td>
<td>0.0869</td>
<td>0.2090</td>
<td>0.00130</td>
</tr>
<tr>
<td>42</td>
<td></td>
<td>78</td>
<td>79</td>
<td>0.0535</td>
<td>0.1697</td>
<td>0.00850</td>
</tr>
<tr>
<td>43</td>
<td></td>
<td>77</td>
<td>78</td>
<td>0.0226</td>
<td>0.0682</td>
<td>0.00040</td>
</tr>
<tr>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line #</td>
<td>Line Name</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. Susp.</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>76</td>
<td>77</td>
<td>0.1539</td>
<td>0.4232</td>
<td>0.00280</td>
</tr>
<tr>
<td>46</td>
<td></td>
<td>72</td>
<td>73</td>
<td>0.1457</td>
<td>0.2555</td>
<td>0.00220</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td>71</td>
<td>73</td>
<td>0.2248</td>
<td>0.6099</td>
<td>0.00480</td>
</tr>
<tr>
<td>48</td>
<td></td>
<td>266</td>
<td>267</td>
<td>-2.9023</td>
<td>2.6004</td>
<td>0.0</td>
</tr>
<tr>
<td>49</td>
<td></td>
<td>265</td>
<td>267</td>
<td>-0.5588</td>
<td>0.8411</td>
<td>0.0</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>24</td>
<td>47</td>
<td>0.0002</td>
<td>0.0030</td>
<td>0.14260</td>
</tr>
<tr>
<td>51</td>
<td></td>
<td>40</td>
<td>48</td>
<td>0.0316</td>
<td>0.0105</td>
<td>0.01030</td>
</tr>
<tr>
<td>52</td>
<td></td>
<td>40</td>
<td>48</td>
<td>0.0016</td>
<td>0.0105</td>
<td>0.01030</td>
</tr>
<tr>
<td>907</td>
<td></td>
<td>265</td>
<td>266</td>
<td>-0.6650</td>
<td>1.5140</td>
<td>0.0</td>
</tr>
<tr>
<td>908</td>
<td></td>
<td>264</td>
<td>267</td>
<td>-0.1943</td>
<td>0.3147</td>
<td>0.0</td>
</tr>
<tr>
<td>909</td>
<td></td>
<td>264</td>
<td>267</td>
<td>-2.2094</td>
<td>2.3464</td>
<td>0.0</td>
</tr>
<tr>
<td>910</td>
<td></td>
<td>263</td>
<td>267</td>
<td>-0.175</td>
<td>0.4398</td>
<td>0.0</td>
</tr>
<tr>
<td>911</td>
<td></td>
<td>263</td>
<td>265</td>
<td>-0.0890</td>
<td>0.2360</td>
<td>0.0</td>
</tr>
<tr>
<td>917</td>
<td></td>
<td>262</td>
<td>267</td>
<td>-0.8873</td>
<td>0.8362</td>
<td>0.0</td>
</tr>
<tr>
<td>918</td>
<td></td>
<td>262</td>
<td>266</td>
<td>-4.6097</td>
<td>4.7822</td>
<td>0.0</td>
</tr>
<tr>
<td>919</td>
<td></td>
<td>262</td>
<td>265</td>
<td>-1.1300</td>
<td>1.5223</td>
<td>0.0</td>
</tr>
<tr>
<td>920</td>
<td></td>
<td>262</td>
<td>264</td>
<td>-0.5653</td>
<td>0.7038</td>
<td>0.0</td>
</tr>
<tr>
<td>921</td>
<td></td>
<td>262</td>
<td>263</td>
<td>-0.2321</td>
<td>0.3430</td>
<td>0.0</td>
</tr>
<tr>
<td>925</td>
<td></td>
<td>261</td>
<td>267</td>
<td>-0.9667</td>
<td>0.9315</td>
<td>0.0</td>
</tr>
<tr>
<td>201</td>
<td></td>
<td>21</td>
<td>22</td>
<td>0.0298</td>
<td>0.0179</td>
<td>0.51290</td>
</tr>
<tr>
<td>202</td>
<td></td>
<td>22</td>
<td>23</td>
<td>0.0031</td>
<td>0.0211</td>
<td>0.84650</td>
</tr>
<tr>
<td>203</td>
<td></td>
<td>35</td>
<td>37</td>
<td>0.0194</td>
<td>0.0592</td>
<td>0.15280</td>
</tr>
<tr>
<td>204</td>
<td></td>
<td>31</td>
<td>90</td>
<td>0.0027</td>
<td>0.0137</td>
<td>0.01950</td>
</tr>
<tr>
<td>205</td>
<td></td>
<td>31</td>
<td>90</td>
<td>0.0026</td>
<td>0.0158</td>
<td>0.01520</td>
</tr>
<tr>
<td>206</td>
<td></td>
<td>30</td>
<td>35</td>
<td>0.0100</td>
<td>0.0690</td>
<td>0.06670</td>
</tr>
<tr>
<td>207</td>
<td></td>
<td>82</td>
<td>83</td>
<td>0.0113</td>
<td>0.0425</td>
<td>0.02420</td>
</tr>
<tr>
<td>208</td>
<td></td>
<td>76</td>
<td>77</td>
<td>0.1537</td>
<td>0.6216</td>
<td>0.00280</td>
</tr>
<tr>
<td>209</td>
<td></td>
<td>29</td>
<td>95</td>
<td>0.0038</td>
<td>0.0267</td>
<td>0.02590</td>
</tr>
<tr>
<td>210</td>
<td></td>
<td>41</td>
<td>96</td>
<td>0.0012</td>
<td>0.0080</td>
<td>0.68700</td>
</tr>
<tr>
<td>211</td>
<td></td>
<td>31</td>
<td>45</td>
<td>0.0033</td>
<td>0.0206</td>
<td>0.07320</td>
</tr>
<tr>
<td>212</td>
<td></td>
<td>38</td>
<td>97</td>
<td>0.0030</td>
<td>0.0194</td>
<td>0.01890</td>
</tr>
<tr>
<td>213</td>
<td></td>
<td>23</td>
<td>47</td>
<td>0.0026</td>
<td>0.0344</td>
<td>0.38940</td>
</tr>
<tr>
<td>214</td>
<td></td>
<td>99</td>
<td>100</td>
<td>0.5305</td>
<td>0.7527</td>
<td>0.00380</td>
</tr>
<tr>
<td>215</td>
<td></td>
<td>106</td>
<td>101</td>
<td>0.0739</td>
<td>0.1183</td>
<td>0.00060</td>
</tr>
<tr>
<td>216</td>
<td></td>
<td>101</td>
<td>103</td>
<td>0.0042</td>
<td>0.0164</td>
<td>0.00010</td>
</tr>
<tr>
<td>217</td>
<td></td>
<td>103</td>
<td>104</td>
<td>0.0164</td>
<td>0.0642</td>
<td>0.00050</td>
</tr>
<tr>
<td>218</td>
<td></td>
<td>101</td>
<td>106</td>
<td>0.0236</td>
<td>0.0786</td>
<td>0.00060</td>
</tr>
<tr>
<td>219</td>
<td></td>
<td>104</td>
<td>105</td>
<td>0.0128</td>
<td>0.0494</td>
<td>0.00330</td>
</tr>
<tr>
<td>220</td>
<td></td>
<td>105</td>
<td>102</td>
<td>0.0250</td>
<td>0.0922</td>
<td>0.00060</td>
</tr>
<tr>
<td>221</td>
<td></td>
<td>106</td>
<td>107</td>
<td>0.0095</td>
<td>0.0350</td>
<td>0.00020</td>
</tr>
</tbody>
</table>
### Line Information

<table>
<thead>
<tr>
<th>Line #</th>
<th>Line Name</th>
<th>S.E.</th>
<th>R.E.</th>
<th>Per Unit R</th>
<th>Per Unit X</th>
<th>S.E. Suscept.</th>
<th>R.E. Suscept.</th>
<th>TX?</th>
</tr>
</thead>
<tbody>
<tr>
<td>222</td>
<td></td>
<td>0.0143</td>
<td>0.0052</td>
<td>0.19230</td>
<td>0.0030</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>223</td>
<td></td>
<td>0.0062</td>
<td>0.0322</td>
<td>0.10220</td>
<td>0.0020</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td></td>
<td>0.0058</td>
<td>0.0343</td>
<td>0.00020</td>
<td>0.0020</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>261</td>
<td>0.0285</td>
<td>0.0087</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0501</td>
<td>0.4320</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td></td>
<td>0.0991</td>
<td>0.3906</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>239</td>
<td></td>
<td>0.0114</td>
<td>0.0794</td>
<td>0.007700</td>
<td>0.07700</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>239</td>
<td>49</td>
<td>0.0115</td>
<td>0.0800</td>
<td>0.007760</td>
<td>0.07760</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>112</td>
<td>0.0023</td>
<td>0.0146</td>
<td>0.004400</td>
<td>0.04400</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>36</td>
<td>0.0035</td>
<td>0.0052</td>
<td>0.004903</td>
<td>0.04900</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>261</td>
<td>0.2277</td>
<td>0.4451</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>263</td>
<td>0.1985</td>
<td>1.6585</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>267</td>
<td>0.5780</td>
<td>0.6528</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td>117</td>
<td>0.0246</td>
<td>0.0886</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>119</td>
<td>0.0276</td>
<td>0.0992</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td>118</td>
<td>0.0139</td>
<td>0.0498</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>120</td>
<td>0.0036</td>
<td>0.0131</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>120</td>
<td>0.1118</td>
<td>0.2918</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>121</td>
<td>0.0537</td>
<td>0.1397</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>122</td>
<td>0.1136</td>
<td>0.2964</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>123</td>
<td>0.1020</td>
<td>0.2663</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>124</td>
<td>0.0678</td>
<td>0.2048</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>124</td>
<td>0.0055</td>
<td>0.0166</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>128</td>
<td>0.0496</td>
<td>0.1511</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td>260</td>
<td>0.4103</td>
<td>1.1718</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>265</td>
<td>0.1067</td>
<td>0.4018</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>266</td>
<td>0.1749</td>
<td>0.3764</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>117</td>
<td>0.0295</td>
<td>0.1062</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>117</td>
<td>0.0276</td>
<td>0.1043</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>117</td>
<td>0.0157</td>
<td>0.1091</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>132</td>
<td>0.0374</td>
<td>0.1348</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>134</td>
<td>0.0156</td>
<td>0.1025</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>138</td>
<td>0.0276</td>
<td>0.1813</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>141</td>
<td>0.0079</td>
<td>0.0518</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>141</td>
<td>0.0191</td>
<td>0.0125</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>136</td>
<td>0.0026</td>
<td>0.0355</td>
<td>1.78622</td>
<td>0.56107</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>135</td>
<td>0.0025</td>
<td>0.0332</td>
<td>1.47260</td>
<td>0.44103</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>132</td>
<td>0.1197</td>
<td>0.3397</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>145</td>
<td>0.0666</td>
<td>0.2680</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
<td>R.E. SUSEP.</td>
<td>Text</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>263</td>
<td></td>
<td>144</td>
<td>149</td>
<td>0.0244</td>
<td>0.0857</td>
<td>0.00700</td>
<td>0.00700</td>
<td>F</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td>145</td>
<td>148</td>
<td>0.0364</td>
<td>0.1460</td>
<td>0.03430</td>
<td>0.03430</td>
<td>F</td>
</tr>
<tr>
<td>265</td>
<td></td>
<td>147</td>
<td>149</td>
<td>0.0229</td>
<td>0.0804</td>
<td>0.01820</td>
<td>0.01820</td>
<td>F</td>
</tr>
<tr>
<td>266</td>
<td></td>
<td>149</td>
<td>152</td>
<td>0.0055</td>
<td>0.0323</td>
<td>1.25560</td>
<td>0.00560</td>
<td>F</td>
</tr>
<tr>
<td>267</td>
<td></td>
<td>151</td>
<td>256</td>
<td>0.0201</td>
<td>0.0214</td>
<td>0.02180</td>
<td>0.02180</td>
<td>F</td>
</tr>
<tr>
<td>271</td>
<td></td>
<td>40</td>
<td>44</td>
<td>0.0032</td>
<td>0.0203</td>
<td>0.22390</td>
<td>0.22390</td>
<td>F</td>
</tr>
<tr>
<td>801</td>
<td></td>
<td>251</td>
<td>252</td>
<td>0.0004</td>
<td>0.0098</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>802</td>
<td></td>
<td>251</td>
<td>253</td>
<td>-0.0227</td>
<td>0.1683</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>803</td>
<td></td>
<td>251</td>
<td>254</td>
<td>-0.0422</td>
<td>0.3900</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>804</td>
<td></td>
<td>251</td>
<td>255</td>
<td>-20.0650</td>
<td>59.5974</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>806</td>
<td></td>
<td>251</td>
<td>257</td>
<td>-0.0700</td>
<td>0.4537</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>807</td>
<td></td>
<td>251</td>
<td>258</td>
<td>-102.5900</td>
<td>182.2950</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>808</td>
<td></td>
<td>251</td>
<td>259</td>
<td>-2.8125</td>
<td>5.7180</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>809</td>
<td></td>
<td>251</td>
<td>260</td>
<td>-0.5662</td>
<td>1.4469</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>810</td>
<td></td>
<td>251</td>
<td>261</td>
<td>-4.5883</td>
<td>8.9837</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>811</td>
<td></td>
<td>251</td>
<td>262</td>
<td>-3.0584</td>
<td>4.4074</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>812</td>
<td></td>
<td>251</td>
<td>263</td>
<td>-4.3832</td>
<td>4.0890</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>813</td>
<td></td>
<td>251</td>
<td>264</td>
<td>-6.0393</td>
<td>6.6111</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>814</td>
<td></td>
<td>251</td>
<td>265</td>
<td>-6.5004</td>
<td>10.6569</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>815</td>
<td></td>
<td>251</td>
<td>266</td>
<td>-35.8660</td>
<td>40.8093</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>816</td>
<td></td>
<td>251</td>
<td>267</td>
<td>-12.9350</td>
<td>8.3706</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>817</td>
<td></td>
<td>252</td>
<td>253</td>
<td>0.0012</td>
<td>0.0240</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>818</td>
<td></td>
<td>253</td>
<td>254</td>
<td>-0.0075</td>
<td>0.0079</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>819</td>
<td></td>
<td>253</td>
<td>255</td>
<td>-1.0302</td>
<td>3.0674</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>820</td>
<td></td>
<td>253</td>
<td>256</td>
<td>-0.0443</td>
<td>0.1832</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>821</td>
<td></td>
<td>253</td>
<td>257</td>
<td>-6.3519</td>
<td>9.9404</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>822</td>
<td></td>
<td>253</td>
<td>258</td>
<td>-0.2185</td>
<td>0.4428</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>823</td>
<td></td>
<td>253</td>
<td>259</td>
<td>-0.0220</td>
<td>0.0789</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>824</td>
<td></td>
<td>253</td>
<td>260</td>
<td>-0.0955</td>
<td>0.3380</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>825</td>
<td></td>
<td>253</td>
<td>261</td>
<td>-0.1424</td>
<td>0.2520</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>826</td>
<td></td>
<td>253</td>
<td>262</td>
<td>-0.1436</td>
<td>0.1915</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>827</td>
<td></td>
<td>253</td>
<td>263</td>
<td>-0.1667</td>
<td>0.2929</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>828</td>
<td></td>
<td>253</td>
<td>264</td>
<td>-0.1666</td>
<td>0.4468</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>829</td>
<td></td>
<td>253</td>
<td>265</td>
<td>-1.3147</td>
<td>1.9227</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>830</td>
<td></td>
<td>253</td>
<td>266</td>
<td>-0.4159</td>
<td>0.4190</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>831</td>
<td></td>
<td>254</td>
<td>255</td>
<td>0.0038</td>
<td>0.0350</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>832</td>
<td></td>
<td>254</td>
<td>256</td>
<td>-0.0892</td>
<td>0.4263</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>833</td>
<td></td>
<td>254</td>
<td>258</td>
<td>-0.3450</td>
<td>3.3338</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>834</td>
<td></td>
<td>254</td>
<td>259</td>
<td>-0.5214</td>
<td>1.1066</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>835</td>
<td></td>
<td>254</td>
<td>260</td>
<td>-0.0074</td>
<td>0.0752</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>836</td>
<td></td>
<td>254</td>
<td>261</td>
<td>-0.0560</td>
<td>0.5254</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>837</td>
<td></td>
<td>254</td>
<td>262</td>
<td>-0.2510</td>
<td>0.5082</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>838</td>
<td></td>
<td>254</td>
<td>263</td>
<td>-0.1070</td>
<td>0.2710</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSP.</td>
<td>R.E. SUSP.</td>
<td>TX?</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>-----</td>
</tr>
<tr>
<td>839</td>
<td>254 264</td>
<td>-0.2274</td>
<td>0.5160</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>840</td>
<td>254 265</td>
<td>-0.1163</td>
<td>0.5713</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>841</td>
<td>254 266</td>
<td>-0.4738</td>
<td>1.7287</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>842</td>
<td>254 267</td>
<td>-0.6873</td>
<td>0.8224</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>843</td>
<td>255 256</td>
<td>-0.0075</td>
<td>0.0665</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>844</td>
<td>255 257</td>
<td>-22.0660</td>
<td>50.7000</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>845</td>
<td>255 258</td>
<td>-0.0146</td>
<td>0.0997</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>846</td>
<td>255 259</td>
<td>-35.9370</td>
<td>56.8130</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>847</td>
<td>255 260</td>
<td>-0.1595</td>
<td>1.4119</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>848</td>
<td>255 261</td>
<td>-2.3240</td>
<td>7.3432</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>849</td>
<td>255 262</td>
<td>-7.9582</td>
<td>10.6685</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>850</td>
<td>255 263</td>
<td>-0.3532</td>
<td>1.0663</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>851</td>
<td>255 264</td>
<td>-3.8132</td>
<td>5.5421</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>852</td>
<td>255 265</td>
<td>-0.5188</td>
<td>2.4778</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>853</td>
<td>255 266</td>
<td>-0.1765</td>
<td>0.4482</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>854</td>
<td>255 267</td>
<td>-6.5988</td>
<td>7.0145</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>855</td>
<td>255 268</td>
<td>-105.6200</td>
<td>152.6340</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>856</td>
<td>255 269</td>
<td>-2.7975</td>
<td>4.7670</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>857</td>
<td>255 270</td>
<td>-0.5803</td>
<td>1.2263</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>858</td>
<td>255 271</td>
<td>-4.4030</td>
<td>7.1198</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>859</td>
<td>255 272</td>
<td>-2.8699</td>
<td>4.7670</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>860</td>
<td>255 273</td>
<td>-3.9949</td>
<td>2.4778</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>861</td>
<td>255 274</td>
<td>-5.5379</td>
<td>3.7067</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>862</td>
<td>255 275</td>
<td>-6.1980</td>
<td>6.7961</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>863</td>
<td>255 276</td>
<td>-33.4670</td>
<td>33.0671</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>864</td>
<td>255 277</td>
<td>-11.5410</td>
<td>6.4912</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>865</td>
<td>255 278</td>
<td>-168.8700</td>
<td>176.5590</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>866</td>
<td>255 279</td>
<td>-1.5420</td>
<td>4.3306</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>867</td>
<td>255 280</td>
<td>-22.0740</td>
<td>26.1033</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>868</td>
<td>255 281</td>
<td>-48.4210</td>
<td>29.7899</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>869</td>
<td>255 282</td>
<td>-3.3237</td>
<td>3.7931</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>870</td>
<td>255 283</td>
<td>-27.7870</td>
<td>6.8697</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>871</td>
<td>255 284</td>
<td>-5.5765</td>
<td>9.7306</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>872</td>
<td>255 285</td>
<td>-1.0127</td>
<td>0.7724</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>873</td>
<td>255 286</td>
<td>-40.7720</td>
<td>7.3724</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>874</td>
<td>255 287</td>
<td>-0.9370</td>
<td>1.5647</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>875</td>
<td>255 288</td>
<td>-5.4770</td>
<td>7.5895</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>876</td>
<td>255 289</td>
<td>-2.0136</td>
<td>2.8698</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>877</td>
<td>255 290</td>
<td>-4.1062</td>
<td>3.0298</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>878</td>
<td>255 291</td>
<td>-5.9510</td>
<td>5.1445</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>879</td>
<td>255 292</td>
<td>-7.5768</td>
<td>9.0937</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>880</td>
<td>255 293</td>
<td>-40.6000</td>
<td>32.8438</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>881</td>
<td>255 294</td>
<td>-11.5520</td>
<td>5.9102</td>
<td>0.0</td>
<td>0.0</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINE #</td>
<td>LINE NAME</td>
<td>S.E.</td>
<td>R.E.</td>
<td>PER UNIT R</td>
<td>PER UNIT X</td>
<td>S.E. SUSEP.</td>
<td>R.E. SUSEP.</td>
<td>TXT</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
<td>-----</td>
</tr>
<tr>
<td>882</td>
<td></td>
<td>260</td>
<td>261</td>
<td>-0.0730</td>
<td>0.3250</td>
<td>0.00</td>
<td>0.00</td>
<td>F</td>
</tr>
<tr>
<td>883</td>
<td></td>
<td>260</td>
<td>262</td>
<td>-0.4514</td>
<td>0.6506</td>
<td>0.00</td>
<td>0.00</td>
<td>F</td>
</tr>
<tr>
<td>884</td>
<td></td>
<td>260</td>
<td>263</td>
<td>-0.1047</td>
<td>0.2208</td>
<td>0.00</td>
<td>0.00</td>
<td>F</td>
</tr>
<tr>
<td>99</td>
<td></td>
<td>90</td>
<td>91</td>
<td>0.00</td>
<td>0.0968</td>
<td>0.00</td>
<td>0.40600</td>
<td>T</td>
</tr>
</tbody>
</table>