University of Alberta

REAL-TIME DIGITAL SIMULATION OF LARGE POWER SYSTEMS BASED ON A ROBUST TWO-LAYER NETWORK EQUIVALENT

by

Xin Nie ©

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

Department of Electrical and Computer Engineering

Edmonton, Alberta
Fall 2005

Your file Votre référence
ISBN: 0-494-09247-5
Our file Notre référence
ISBN: 0-494-09247-5

Abstract

In this thesis, a robust approach of Two-Layer Network Equivalent (TLNE) is elaborated. Featuring low-order Marti's line model in surface layer, global searching of low-order deep region networks by Genetic Algorithms (GAs) with passivity and stability constraints and compensation, and Constrained Linearized Least-Square (CLLSQ) optimization with guaranteed convergence as well as the stability and passivity of the obtained model, the robust approach is able to generate the Robust TLNE model with high accuracy and efficiency.

Simulation results and computational time analysis show that the Robust TLNE model is highly efficient in the order reduction of the external system and is suitable for real-time implementation of larger power systems. Two examples and a case study of a realistic large power system — the Alberta Interconnected Electric System (AIES) verify the proposed Robust TLNE approach.

Furthermore, real-time implementation of the second example and AIES is accomplished at $20\mu s$ time-step size in a Xeon cluster based real-time simulator in the Real-Time eXperimental LABoratory (RTX-LAB) of the Power Engineering Group at the University of Alberta. A MATLAB/SIMULINK C++ S-function implementing EMTP is programmed to accommodate the Robust TLNE model in the real-time simulator. Real-time simulation results observed from oscilloscope are identical compared to off-line results.

Acknowledgements

I wish to express my deep gratitude to Dr. Venkata Dinavahi, my supervisor, for his constant and kind support, excellent advice and guidance during the whole research work.

I extend my thanks to the members of my M. Sc. committee members for their careful reviews of the thesis and for many useful comments.

Appreciations are also due to Dr. Bjørn Gustavsen from SINTEF Energy Research Norway for providing Vector Fitting related routines and helpful suggestions; to Dr. Washington Neves for providing the nonlinear fitting routine for transmission line parameters and kind assistance; to the Ms. Pamela Mclean from Alberta Electric System Operator (AESO), Calgary for providing Transmission Alberta System Model (TASMo) database, Alberta Interconnected Electric System (AIES) Map and valuable advice; to European EMTP-ATP User Group (EEUG) for their passion on working with, developing and enhancing ATP; to technical support of Opal-RT Technologies Inc. during the integration, testing and application of the RTX-LAB real-time simulator.

I am wholly indebted to my parents for their support and wisdom that has brought me this far and will hopefully carry me further.

Contents

List of Tables

List of Figures

List of Symbols

A, B, P, ...	Capital boldface letters denote matrices and phaser vectors
u, y, z, ...	Lowercase boldface letters denote vectors
u, y, z, ...	Lowercase italics denote scalar-valued function or scalars
$\overrightarrow{V}$	Voltage phasor
$\overrightarrow{I}$	Current phasor
$\overrightarrow{S}$	Complex power
$\mathbf{Y}^T$	Transpose of $\mathbf{Y}$
$\mathbf{Y}^*$	Complex conjugate and transpose of $\mathbf{Y}$
$\mathrm{Re}(\mathbf{Y})$	Real part of $\mathbf{Y}$
$\mathrm{Im}(\mathbf{Y})$	Imaginary part of $\mathbf{Y}$
$\|\mathbf{Y}\|_F$	Frobenius norm of $\mathbf{Y}$
$\mathrm{eig}(\mathbf{Y})$	Eigenvalues of $\mathbf{Y}$
Y or $\mathbf{Y}$	Admittance or admittance matrix
Z or $\mathbf{Z}$	Impedance or Impedance matrix
V or $\mathbf{V}$	Voltage phaser or phaser vector
I or $\mathbf{I}$	Current phaser or phaser vector
$v(t)$ or $\mathbf{v}(t)$	Instantaneous voltage or voltage vector
$i(t)$ or $\mathbf{i}(t)$	Instantaneous current or current vector
$\mathbf{Y}_{input}$	Original external system input admittance matrix
$\tilde{\mathbf{Y}}_{input}$	Admittance matrix approximating external system
$\tilde{\mathbf{Y}}^0_{input}$	First approximation of external system input admittance matrix
$\mathbf{Y}_{surface}$	Original surface layer admittance matrix
$\tilde{\mathbf{Y}}_{surface}$	Admittance matrix approximating surface layer

$\tilde{\mathbf{Y}}^0_{surface}$	First approximation of surface layer admittance matrix
$\mathbf{Y}_{deep}$	Original deep region admittance matrix
$\tilde{\mathbf{Y}}^{VF}_{deep}$	Original deep region admittance matrix fitted by Vector Fitting
$\tilde{\mathbf{Y}}_{deep}$	Admittance matrix approximating deep region
$\tilde{\mathbf{Y}}^0_{deep}$	First approximation of deep region admittance matrix

List of Acronyms

EMTP	Electro-Magnetic Transients Program
ATP	Alternative Transients Program
FDNE	Frequency-Dependent Network Equivalent
TLNE	Two-Layer Network Equivalent
GA	Genetic Algorithm
VF	Vector Fitting
RMS	Root-Mean-Square
SISO	Single-Input-Single-Output
MIMO	Multiple-Input-Multiple-Output
QP	Quadratic Programming
SQP	Sequential Quadratic Programming
NLP	Nonlinear Programming
CLLSQ	Constrained Linearized Least-Square
LTI	Linear and Time-Invariant
AIES	The Alberta Interconnected Electric System
TASMo	Transmission Administrator System Model
RTW	Real-Time Workshop
RTOS	Real-Time Operating System
FPGA	Field-Programmable Gate Array

1 Introduction

Electromagnetic transients in power systems, induced by switchings, surges, faults and other topology changes in the network, occur in a very short period of time. These transients can activate control and protection systems, lead to power interruptions, or even result in component failures. Studies of such transients are thus of great importance.

Via time-domain, the simulation can be carried out in off-line or real-time. The off-line digital simulations by Electro-Magnetic Transients Program (EMTP) [1,2], *e.g.*, ATP, PSCAD/EMTDC, EMTP-RV, MICROTRAN, NETOMAC, *etc.*, are able to verify the design of line and station insulation and selection of equipment. Nonetheless, real-time simulation is required in order to realize

- Accurate design and testing of new apparatus and schemes such as controllers, relays, and other protective equipment like surge arrestor, spark gaps, *etc.*,
- Closed loop behavior study of the device on the full system,
- Studying the high frequency phenomenon due to the introduction of different disturbance and switching actions, and
- Training and education purposes.

A variety of analogue and digital real-time simulators have so far been developed and used at research institutes, universities, power companies and manufacturers all over the world, such as Micro-Network of EDF (France), AC-DC power system simulator at CRIEPI (Japan), Hydro-Québec real-time simulator (Canada), HVDC simulator of CEPEL (Brazil), APSA of Kansai Electric Power Company (Japan), RTDS of Manitoba HVDC Research Center (Canada), *etc.*. However, the real-time digital simulation of large power systems are limited to only analogue or hybrid (mixed digital and analogue) simulators. This is due to the excessive computational time during the simulation of the full representation of large power system. In such a case, even detailed off-line simulation is computationally prohibitive, since the system model is too complicated. Simulation of transients for large power systems, especially in real-time, requires not only blazing computational power but also simpler models. In order to find more computational efficient representation of power systems, considerable efforts have been made by the past three decades. The main achievement is study zone *v.s.* external system, Frequency-Dependent Network Equivalents (FDNE), and Two-Layer Network Equivalents (TLNE).

1.1 Frequency-Dependent Network Equivalent

In electromagnetic transient studies, due to system complexities, it is a common practice that the system is divided into a *study zone*, a restricted part of the system where the transient phenomena occur and whose components must be fully characterized including any nonlinear and time-variant elements, and an *external system* which encompasses the rest of the system. External system is considered to be electrically remote from the transient location. Higher frequency electromagnetic waves, due to higher attenuation, propagate shorter electrical distances. Therefore, the external system is represented by a linear equivalent network, *i.e.*, Frequency-Dependent Network Equivalent (FDNE), which is shown in Fig. 1.1. There is no well-defined approach for the division of study zones and external systems. Instead, engineering judgement plays an important role. CIGRÉ provides guidelines in [4].

There have been quite a few successful achievements in constructing FDNE which greatly reduce the computational burden of transient simulations [5–20]. Those FDNE models can mainly be classified as frequency-domain model relying on convolutions in time-domain simulation [5], z-domain models [6–9] and s-domain models [10–22] for EMTP. The latest development in s-domain fitting is Vector Fitting (VF) by Gustavsen *et al.* [17–20]. Time-domain fitting for Sparse Network Equivalent (SNE) due to Boaventura *et al.* [8,9] is the up-to-date fitting method in z-domain.

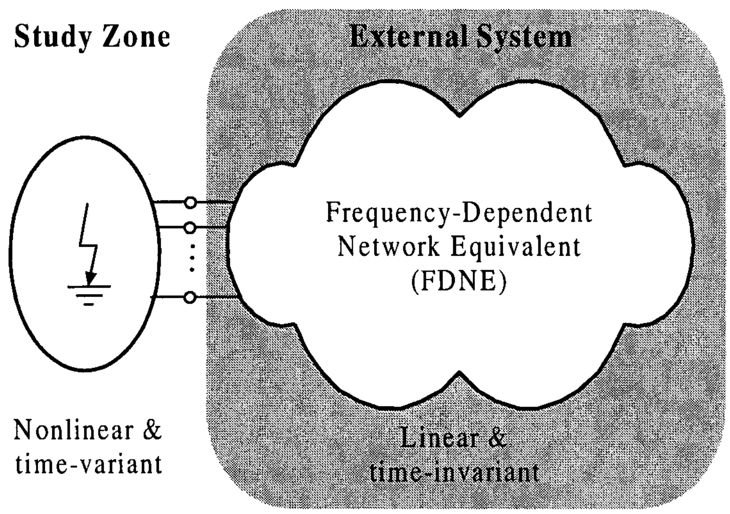


Figure 1.1: Study zone and external system

1.2 Two-Layer Network Equivalent

In real-time simulation of large power systems, however, the bottleneck still remains due to the efficiency concern in convolutions and the high order of the fitted FDNE mathematical model of external systems. It is obvious that an alternative way must be found out to reduce the model complexity without significant effect on accuracy. Proposed by Abdel-Rahman *et al.* [21,22], the Two-Layer Network Equivalent (TLNE) model is another milestone in overcoming the obstacle of real-time digital simulations. It has been noticed that in frequency domain, the fact that external system has the property of numerous resonance peaks is primarily due to frequency-dependent transmission lines. If the leading part of the external system is retained as reduced-order line models, it is possible to find an FDNE model to compensate the deviations due to the order reduction of the line models and the rest of the external system. Thus, illustrated in Fig. 1.2, we have a TLNE model consisting of a *surface layer* represented by reduced-order frequency-dependent transmission line models and a *deep region* as a low-order

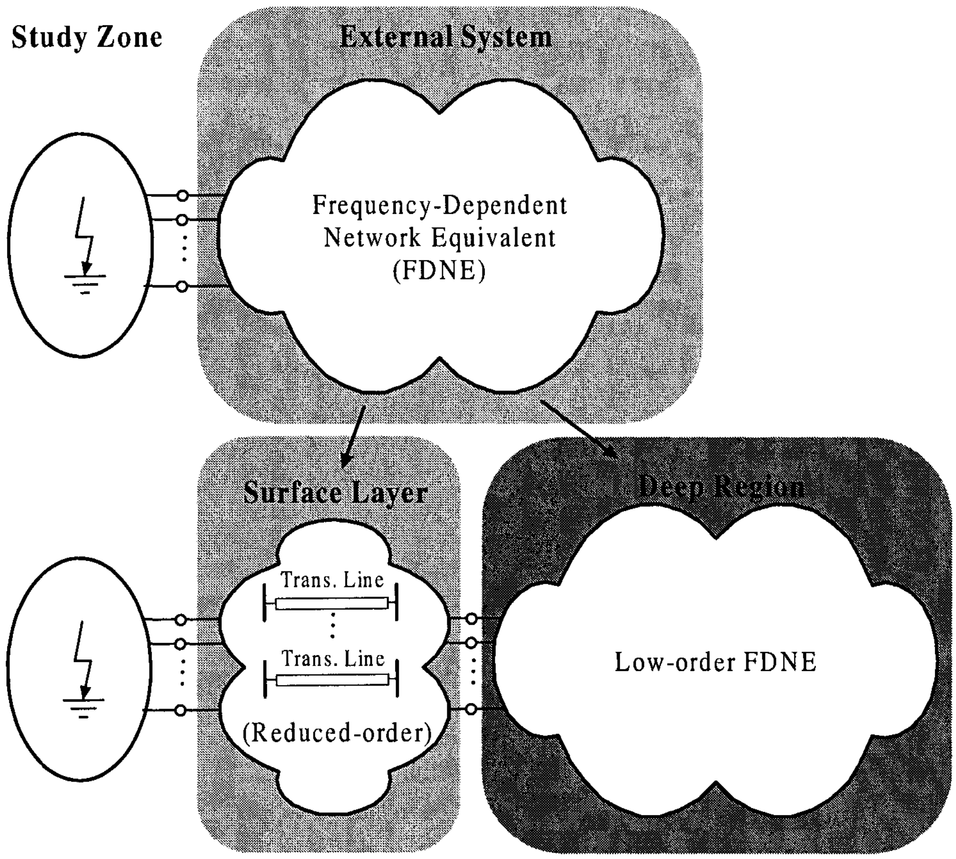


Figure 1.2: Two-layer network equivalent for external system

FDNE. The impact of surface layer and deep region on the external system input admittance varies with frequency. At low frequencies, both surface layer and deep region contributes to the admittance. When the frequency increases, however, the contribution of deep region diminishes drastically due to high attenuation. Therefore, the FDNE model for deep region is able to be obtained in low order. The boundary between surface layer and deep region also heavily relies on engineering judgement and experience. Some guidelines should be obeyed to obtain better and simpler possible equivalent [21]:

- The topology of surface layer, which includes loads and other shunt connections, is retained. This is because removing such elements in surface layer will cause major deviations from the original system in frequency response, which is very unlikely to be compensated by the developed model with stability and passivity constraints.

- Deep region is obtained based on attenuation in the transmission lines of surface layer. Such damping effects depend on the length of transmission lines and the loads in surface layer. Therefore, if lines are short and surface layer does not contain loads or other elements that produce damping, the surface layer is considered to include more lines.

- If difficulties arises in finding deep region parameters, more buses should be included in surface layer, *i.e.*, the surface layer is made deeper.

In addition,

- Depending on the size and complexities, big and complicated external systems result in complex frequency responses. Therefore more lines should be included in surface layer (surface layer is deeper). For example, in modeling a large power system, only one transmission line in surface layer will result in a high-order deep region model, which cancels off the benefits of TLNE model.

1.3 Existing Issues Associated with TLNE

In obtaining TLNE by existing approach [21, 22], it is however revealed following concerns:

- With low-order VF in obtaining the deep region, it is difficult to control deviations with respect to the original deep region, which Sequential Quadratic Programming (SQP) may not be able to compensate.

- Frequency response at DC is not specifically accentuated although it affects transient DC offset.

- SQP is prone to divergence. If better first approximations of input admittance of external systems can be found, SQP can be replaced by Constrained Linearized Least-Square (CLLSQ) optimization, whose the convergence is guaranteed.

- Optimization of surface layer is especially helpful in increasing accuracy of frequency response in the low frequency range, *e.g.*, DC and power frequency, with little cost of computational time. Thus, parameters of surface layer are optimized.

- In multi-port external systems with complex frequency response, the passivity constraint is very strong, so the freedom for changing the parameters is small. Therefore, the first approximation in a multi-port case, is required to be closer to the original than that for single-port case. Thus, transmission line parameters in surface layer require higher accuracy but low-order realization methods.
- The order of deep region is not determined in an optimal way. Optimal deep region order can be obtained by comparing a series orders of deep region with each other.

1.4 Thesis Objectives

Considering the requirement of developing appropriate an external system model for real-time digital simulations and possible improvements in the TLNE model, the main objectives of this thesis are:

- To extend the concept of TLNE model, *i.e.*, Robust TLNE model, which is robust in terms of stability and passivity, more accurate and computationally efficient.
- To compare the performance of the Robust TLNE model with that of the full EMTP model and the FDNE model based on case studies.
- To obtain an TLNE model for the Alberta Interconnected Electric System (AIES) which is suitable for real-time digital simulation of electromagnetic transients.
- To implement a customized EMTP program in C++ for real-time simulation purpose.
- To realize real-time digital simulation of large power systems such as AIES.

1.5 Thesis Outline

The rest of this thesis consists of the following chapters:

- Chapter 2 formulates the Robust TLNE model for passive networks. The background concepts and methods such as passivity criterion, frequency scan, Vector

Fitting (VF), Marti's frequency-dependent line model, and Genetic Algorithms (GAs) are covered as well. One simple system is examined.

- Chapter 3 extends the Robust TLNE model to three-phase active networks. A modified benchmark system which is both multi-port and multi-phase verifies the approach. Performance analysis is carried out in terms of accuracy and computational efficiency.
- Chapter 4 constructs the full model of AIES Area 50 (Backbone) and its robust TLNE model in ATP. Procedures and methods in obtaining the models are explained in detail. Computational performance and accuracy are also analyzed.
- Chapter 5 presents the real-time digital simulation of AIES at Bus 524 Genesee by a Xeon-cluster based real-time simulator at RTX-LAB, which is implemented by a MATLAB/SIMULINK C++ S-function program.
- The conclusions of the thesis and future work are summarized in Chapter 6.

2

Robust Two-Layer Network Equivalent for Passive Networks

In this chapter, detailed theory and formulation of existing TLNE model and the Robust TLNE model are explained. Divided into a surface layer and a deep region, an external system is further reduced to a Robust TLNE with high accuracy and efficiency. The most significant improvements include the implementation of Genetic Algorithms (GAs), Constrained Linearized Least-Square (CLLSQ) optimization and optimal deep region order determination [23].

2.1 Models and Concepts in the Robust TLNE

Before going further into the model, the following topics are briefly explained as the TLNE model is founded on the following methods, concepts and models: passivity criterion, frequency scan, Vector Fitting (VF) used in obtaining FDNE, Marti's frequency-dependent transmission line model and Genetic Algorithms (GAs).

2.1.1 Positive-Realness and Passivity Criterion

An electrical network is passive if it consumes real power for any active sources applied to input terminals, and does not deliver real power. Passivity affects the stability of time-domain simulation. An obvious example is a linear network without any active sources. In realistic networks, passivity is not of concern. However, in fitting the frequency response of passive networks, especially large networks, passivity is of great importance. The fitted model representing such network is required to be passive as well. The electrical network of FDNE model with passivity violation will more likely result in unstable and erroneous simulations.

For an electrical network whose admittance matrix is $\mathbf{Y}$,

$$\mathbf{YV} = \mathbf{I} \tag{2.1}$$

defines the relationship between current vector $\mathbf{I}$ applied to network terminals and voltage vector $\mathbf{V}$ corresponding to each terminal. The real power absorbed by the network is

$$P = \mathrm{Re}(\mathbf{V}^*\mathbf{I}) = \mathrm{Re}(\mathbf{V}^*\mathbf{YV}) = \mathrm{Re}(\mathbf{V}^*(\mathbf{G} + j\mathbf{B})\mathbf{V}) = \mathrm{Re}(\mathbf{V}^*\mathbf{GV}) \tag{2.2}$$

where the asterisk * stands for transpose and conjugate. Notice that the final result of (2.2) is a quadratic form. Therefore, the real power is absorbed or $P > 0$ if and only if the conductance matrix $\mathbf{G} = \mathrm{Re}\{\mathbf{Y}\}$ is positive-definite. It follows that since $\mathbf{G}$ matrix representing a linear network is real and symmetric, all eigenvalues of $\mathbf{G}$ must be real. Thus, passivity criterion can be equivalenced to requiring all eigenvalues of $\mathbf{G}$ to be positive or

$$\mathrm{eig}(\mathbf{G}) > \mathbf{0}. \tag{2.3}$$

The passivity criterion of electrical networks is also denoted as *positive-real* criterion in linear system theory.

2.1.2 Frequency Scan of Linear Passive Networks

To obtain the frequency response of a linear network, frequency scan is the appropriate method. In single-phase single-port network case, a current source with unity magnitude and zero phase angle is applied to the designated port, which is shown in Fig. 2.1. The node voltage of the port is measured. At a particular frequency, based on the phasor equation $Z = V/I$, since the current has unity magnitude and zero phase shift, the voltage phasor measured is equal to impedance Z. Thus frequency response of impedance $Z(\omega)$ is obtained by measuring all voltage phasors when applying the current source

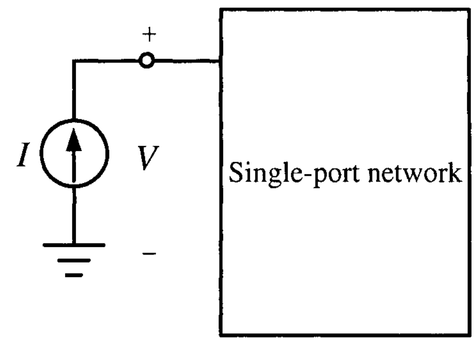


Figure 2.1: Frequency scan of single-phase single-port network

with a range of frequencies of interest. The admittance $Y(\omega)$ is found by taking the inverse of $Z(\omega)$ at each frequency point.

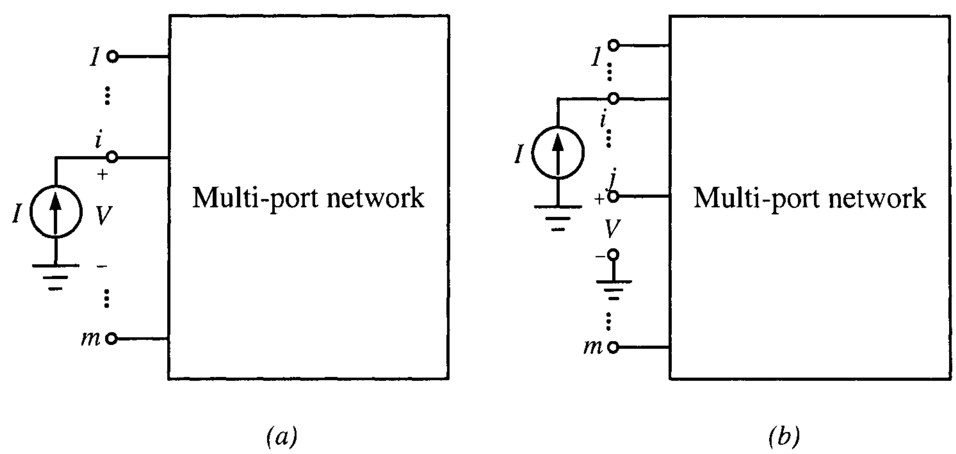


Figure 2.2: Frequency scan of single-phase multi-port network

When frequency scan is applied to a multi-port network, the situation is a little bit different. For a m-port network, we are looking for an impedance matrix that has following form

$$\mathbf{Z}(\omega) = \begin{bmatrix} Z_{11}(\omega) & \cdots & Z_{1m}(\omega) \\ \vdots & \ddots & \vdots \\ Z_{m1}(\omega) & \cdots & Z_{mm}(\omega) \end{bmatrix} \tag{2.4}$$

Since the system is linear, $\mathbf{Z}(\omega)$ is symmetric. Thus off-diagonal elements $Z_{ij}(\omega) = Z_{ji}(\omega)$ where $1 < (i,j) < m, i \neq j$. For diagonal elements $Z_{ii}(\omega)$ where $1 < i < m$, they can be obtained by applying a current with unity magnitude and zero phase angle at i-th port while leaving all other ports as open circuit and measuring the voltage phasor at i-th port, as shown in Fig. 2.2(a). The same method is applied to the off-diagonal element $Z_{ij}(\omega)$ where $1 < (i,j) < m$, $i \neq j$, except the voltage phasor is measured at j-th port instead of i-th port. Finally, admittance matrix $\mathbf{Y}(\omega)$ is obtained by inverting $\mathbf{Z}(\omega)$ matrix in (2.4) at each frequency point, which is

$$\mathbf{Y}(\omega) = \begin{bmatrix} Y_{11}(\omega) & \cdots & Y_{1m}(\omega) \\ \vdots & \ddots & \vdots \\ Y_{m1}(\omega) & \cdots & Y_{mm}(\omega) \end{bmatrix} \tag{2.5}$$

where $Y_{ij}(\omega) = Y_{ji}(\omega)$.

2.1.3 Vector Fitting

Provided Linear and Time-Invariant (LTI), an external system can be reproduced by a rational transfer function in frequency domain for the Single-Input-Single-Output (SISO) case, or in the Multiple-Input-Multiple-Output (MIMO) case, by a rational transfer function matrix whose elements share a common denominator, *i.e.*, same poles. Such procedures require curve fitting in frequency domain. Vector Fitting (VF) [17–20] is the appropriate technique used in s-domain. Since expressing the rational transfer function (matrix) in partial fraction form is suitable for order reduction as well as its robustness and effectiveness, VF is applied in FDNE and TLNE model [21,22]. In MIMO case, the fitted rational matrix having a common set of poles is very useful not only in improving the computational efficiency, but also in maintaining the simplicities of the realistic electrical network branches in EMTP. The method is able to fit the frequency responses with very high accuracy on the RMS-error% basis. For an m-port network system, the fitted proper (the order of denominator is equal to that of numerator) n-th order rational function matrix with common poles is expressed in partial fraction form as

$$\mathbf{Y}_{VF} = \begin{bmatrix} d_{11} + \sum_{i_{11},j=1}^{n} \frac{c_{i_{11}}}{s-a_j} & d_{12} + \sum_{i_{12},j=1}^{n} \frac{c_{i_{12}}}{s-a_j} & \cdots & d_{1m} + \sum_{i_{1m},j=1}^{n} \frac{c_{i_{1m}}}{s-a_j} \\ d_{21} + \sum_{i_{21},j=1}^{n} \frac{c_{i_{21}}}{s-a_j} & d_{22} + \sum_{i_{22},j=1}^{n} \frac{c_{i_{22}}}{s-a_j} & \cdots & d_{2m} + \sum_{i_{2m},j=1}^{n} \frac{c_{i_{2m}}}{s-a_j} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m1} + \sum_{i_{m1},j=1}^{n} \frac{c_{i_{m1}}}{s-a_j} & d_{m2} + \sum_{i_{m2},j=1}^{n} \frac{c_{i_{m2}}}{s-a_j} & \cdots & d_{mm} + \sum_{i_{mm},j=1}^{n} \frac{c_{i_{mm}}}{s-a_j} \end{bmatrix} \tag{2.6}$$

where d_{ij} $(1 \leq i,j \leq m)$ is constant term; $c_{i_{jk}}$ $(1 \leq i \leq n,\ 1 \leq j,k \leq m)$ and a_i $(1 \leq i \leq n)$ are residue and pole, respectively. Notice that the poles and residues may come in complex conjugate pairs.

2.1.4 Electrical Network Realization

The fitted admittance matrix $\mathbf{Y}_{VF}$ in (2.6) can be converted to a realistic electrical network, which has branches between nodes and between nodes and the ground. Branch admittance between node i and ground is given as [19]

$$Y_i(s) = \sum_{j=1}^{m} (\mathbf{Y}_{VF})_{ij} \tag{2.7}$$

and branch admittance between node i and node j is

$$Y_{ij}(s) = -(\mathbf{Y}_{VF})_{ij} \,. \tag{2.8}$$

Since all elements of $\mathbf{Y}_{VF}$ share the same set of poles, the summation in (2.7) becomes the sum of d terms as well as the sum of c terms in the corresponding poles in (2.6).

Each branch calculated in (2.7) and (2.8) gives a rational function

$$Y(s) = d + \sum_{i=1}^{n} \frac{c_i}{s - a_j}. \tag{2.9}$$

Electrical realization generates RL and $RLCG$ branches [10, 11, 19], shown in Fig. 2.3. The R_0 is calculated as

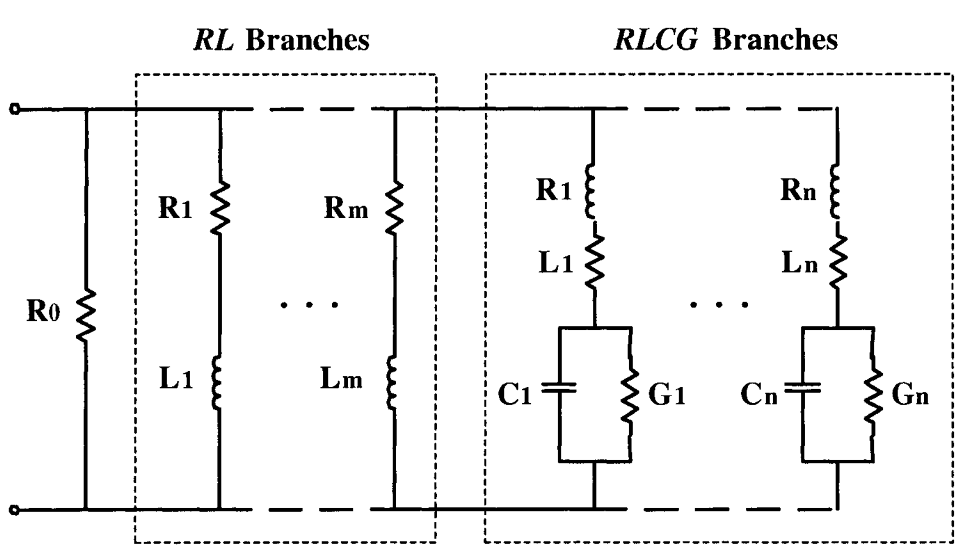


Figure 2.3: Synthesis of RL and $RLCG$ branches in FDNE

$$R_0 = 1/d \tag{2.10}$$

and each $\frac{c}{s-a}$ of real pole gives an RL branch

$$R = -a/c, \; L = 1/c \tag{2.11}$$

whereas each complex conjugate pair $\frac{c'+jc''}{s-(a'+ja'')} + \frac{c'-jc''}{s-(a'-ja'')}$ gives an $RLCG$ branch

$$\begin{aligned} R &= [-2a' + 2(c'a' + c''a'')L]\,L \\ L &= 1/(2c') \\ C &= \frac{1}{[(a')^2 + (a'')^2 + 2(c'a' + c''a'')R]\,L} \\ G &= -2(c'a' + c''a'')CL \end{aligned} \tag{2.12}$$

The R, L, C, G in (2.10), (2.11), (2.12) may appear as negative values. However, EMTP packages such as ATP and PSCAD/EMTDC accept those values. As long as the passivity of overall FDNE is guaranteed, correct and stable simulation is ensured.

2.1.5 Frequency-Dependent Transmission Line Model

In transient simulation of power system networks, accurate modeling requires both frequency dependent factors due to skin effect and traveling wave effects in transmission lines be taken into account. The constant-parameter line model [1] tends to exaggerate the transients somehow. Many efforts have been devoted in the past three decades to the development of frequency-dependent line models in EMTP [24–28] and simulations of transmission line transients. The line models can be categorized as modal-domain models [24, 25] and phase-domain models [26–29]. Since there have been successful approaches in the real-time application of the low-order model in [38–41], Marti's line model [25] is employed in the Robust TLNE model.

When the frequency dependence of parameters and the distributed nature of losses in transmission lines are taken into account, it is very difficult to write the line equations directly in time domain. However, the solution can be readily obtained in frequency domain by the well-known equations:

$$V_k(\omega) = \cosh[\gamma(\omega)\ell]V_m(\omega) - Z_c(\omega)\sinh[\gamma(\omega)\ell]I_m(\omega) \tag{2.13a}$$

$$I_k(\omega) = \frac{1}{Z_c(\omega)}\sinh[\gamma(\omega)\ell]V_m(\omega) - \cosh[\gamma(\omega)\ell]I_m(\omega) \tag{2.13b}$$

where V_k, V_m, I_k and I_m are the frequency-domain quantities corresponding to sending-end and receiving-end voltages and currents, respectively; ℓ is the line length; $Z_c(\omega)$

and $\gamma(\omega)$ are frequency-dependent characteristic impedance and propagation function defined as

$$Z_c(\omega) = \sqrt{\frac{R(\omega) + j\omega L(\omega)}{G + j\omega C}}; \;\; \gamma(\omega) = \sqrt{(R(\omega) + j\omega L(\omega))(G + j\omega C)} \tag{2.14}$$

with

$$\begin{aligned} R(\omega) &= \text{series resistance} & L(\omega) &= \text{series inductance} \\ G &= \text{shunt conductance} & C &= \text{shunt capacitance.} \end{aligned}$$

To relate currents and voltages in frequency domain, new functions are defined [25]: Forward traveling functions

$$F_k(\omega) = V_k(\omega) + Z_c(\omega)I_k(\omega) \tag{2.15a}$$

$$F_m(\omega) = V_m(\omega) + Z_c(\omega)I_m(\omega) \tag{2.15b}$$

and backward traveling functions

$$B_k(\omega) = V_k(\omega) - Z_c(\omega)I_k(\omega) \tag{2.16a}$$

$$B_m(\omega) = V_m(\omega) - Z_c(\omega)I_m(\omega). \tag{2.16b}$$

By eliminating $V_k(\omega)$, $V_m(\omega)$, $I_k(\omega)$ and $I_m(\omega)$ from (2.13a), (2.13b), (2.16a) and (2.16b), we obtain

$$B_k(\omega) = A_1(\omega)F_m(\omega) \tag{2.17a}$$

$$B_m(\omega) = A_1(\omega)F_k(\omega) \tag{2.17b}$$

where

$$A_1(\omega) = e^{-\gamma(\omega)\ell} = \frac{1}{\cosh[\gamma(\omega)\ell] + \sinh[\gamma(\omega)\ell]} \tag{2.18}$$

the time-domain form of which is defined as the *weighting function* $a_1(t)$, obtained by the inverse Fourier transform of $A_1(\omega)$. Equations (2.16a) and (2.16b) gives the Thevenin equivalent network shown in Fig. 2.4. The voltage sources $b_k(t)$ and $b_m(t)$ are the time domain forms of (2.17a) and (2.17b), which are convolution integrals

$$b_k(t) = \int_{\tau}^{\infty} f_m(t-u)a_1(u)du \tag{2.19a}$$

$$b_m(t) = \int_{\tau}^{\infty} f_k(t-u)a_1(u)du \tag{2.19b}$$

where

$$f_k(t) = 2V_k(t) - b_k(t) \tag{2.20a}$$

$$f_m(t) = 2V_m(t) - b_m(t). \tag{2.20b}$$

The reason that the lower limit of those integrals is the propagation delay τ is that in time domain an impulse on one end of the line will not reach the other end until the time of τ [25].

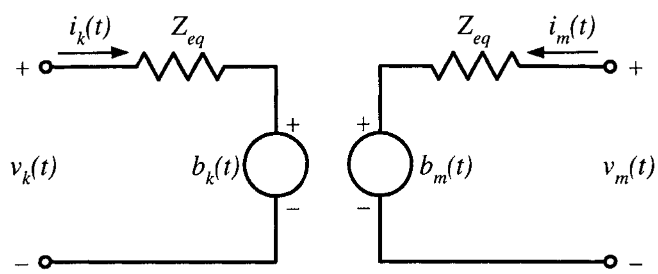


Figure 2.4: Marti's frequency-dependent line model

Accuracy of Marti's line model greatly depends on the fitting of $Z_c(\omega)$ and $A_1(\omega)$. The appropriate techniques used are Bode's asymptotic fitting technique [25] and non-linear Levenberg-Marquardt (LM) fitting method due to Fernandes *et al.* [42,43]. $Z_c(\omega)$ is fitted by an n-th order rational function of the form

$$Z_{eq}(s) = k_0 \frac{(s+z_1)(s+z_2)\cdots(s+z_n)}{(s+p_1)(s+p_2)\cdots(s+p_n)} \tag{2.21}$$

with all poles and zeros are real and simple and lie on the left hand side of the complex plane (a minimal phase system) and is further expanded to partial fraction form

$$Z_{eq}(s) = k_0 + \sum_{i=1}^{n} \frac{r_i}{s+p_i} \tag{2.22}$$

which is realized by a series of RC parallel blocks (Foster I network realization, Fig. 2.5). The parameters are calculated as

$$R_0 = k_0,\ R_i = k_i/p_i,\ C_i = 1/k_i. \tag{2.23}$$

One RC parallel block shown in Fig. 2.6(a) can be discretized into an equivalent resistor in series with a voltage source (Fig. 2.6(b)) [1]. The discrete nodal equation is

$$v(t) = R_{RC}i(t) + V(t - \Delta t) \tag{2.24}$$

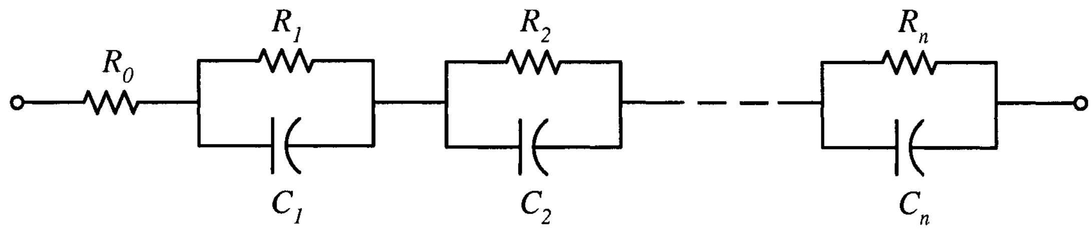


Figure 2.5: RC network realization of $Z_{eq}(\omega)$ approximating $Z_c(\omega)$

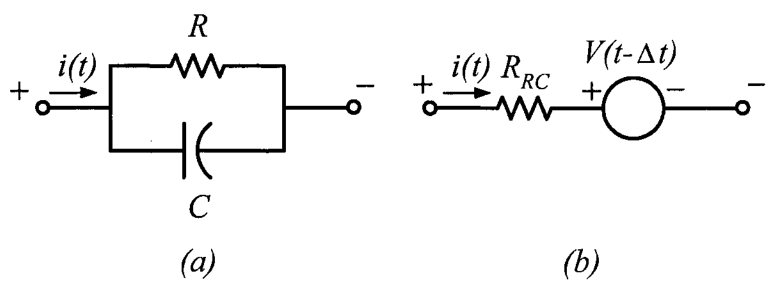


Figure 2.6: One RC block and its discretization

where

$$R_{RC} = \frac{R\frac{\Delta t}{2C}}{R + \frac{\Delta t}{2C}} \tag{2.25}$$

$$V(t - \Delta t) = \frac{R^2\frac{\Delta t}{C}}{(R + \frac{\Delta t}{2C})^2} i(t - \Delta t) + \frac{R - \frac{\Delta t}{2C}}{R + \frac{\Delta t}{2C}} V(t - 2\Delta t). \tag{2.26}$$

Since the past history term $V(t - \Delta t)$ is only related to the current $i(t - \Delta t)$ and history term $V(t - 2\Delta t)$ of the previous time step, the RC parallel block networks representing Z_{eq} in Fig. 2.5 can be further reduced into an equivalent resistor

$$R_{eq} = R_0 + \sum_{i=1}^{n} \frac{R_i\frac{\Delta t}{2C_i}}{R_i + \frac{\Delta t}{2C_i}} \tag{2.27}$$

in series with a voltage source

$$V_{eq}(t - \Delta t) = \sum_{i=1}^{n} V_i(t - \Delta t) \tag{2.28}$$

where

$$V_i(t - \Delta t) = \frac{R_i^2\frac{\Delta t}{C_i}}{(R_i + \frac{\Delta t}{2C_i})^2} i_i(t - \Delta t) + \frac{R_i - \frac{\Delta t}{2C_i}}{R_i + \frac{\Delta t}{2C_i}} V_i(t - 2\Delta t) \tag{2.29}$$

and $1 \leq i \leq n$, n is total number of RC parallel blocks or the order of the approximation of $Z_c(\omega)$.

Computational efficiency of the convolutions of (2.19a) and (2.19b) may be greatly increased if the weighting function $a_1(t)$ has the form of sum of exponential terms [24]. To do so, the same fitting techniques for $Z_c(\omega)$ are applied in the approximation of weighting function $A_1(\omega)$. However, rather than a proper form of $Z_{eq}(s)$, the $A_1(\omega)$ is first back-winded by the propagation delay τ to produce

$$P(\omega) = A_1(\omega)e^{j\omega\tau} \tag{2.30}$$

and then approximated in a strictly proper manner by a m-th order rational function

$$P_a(s) = k\frac{(s+z_1)(s+z_2)\cdots(s+z_q)}{(s+p_1)(s+p_2)\cdots(s+p_m)} \quad (q < m) \tag{2.31}$$

Partial fraction expansion gives

$$P_a(s) = \sum_{i=1}^{m} \frac{r_i}{s+p_i}. \tag{2.32}$$

Thus we obtain a sum of exponentials from the inverse Fourier transformation as

$$a_{1a}(t) = u(t-\tau)\sum_{i=1}^{m} k_i e^{-p_i(t-\tau)} \tag{2.33}$$

where $u(t-\tau)$ is step response with τ delay.

In order to obtain the current sources $b_k(t)$ and $b_m(t)$ in Fig. 2.4, it is necessary to evaluate convolution integrals from (2.19a) and (2.19b). To reduce the computational burden, the weighting function $a_1(t)$ is expressed as sum of exponential terms. For one generic term $ke^{-p(t-\tau)}u(t-\tau)$ in approximated function $a_{1a}(t)$ of (2.33), the convolution integral is

$$s(t) = \int_{\tau}^{\infty} f(t-u)ke^{-p(u-\tau)}du. \tag{2.34}$$

$s(t)$ in (2.34) can be directly obtained from recursive convolution [24] by

$$s(t) = M\cdot s(t-\Delta t) + P\cdot f(t-\tau) + Q\cdot f(t-\tau-\Delta t) \tag{2.35}$$

where

$$M = e^{-p\Delta t} = \alpha \tag{2.36a}$$

$$P = \frac{k}{p}(1 - \frac{1-\alpha}{p\Delta t}) \tag{2.36b}$$

$$Q = \frac{k}{p}(\frac{1-\alpha}{p\Delta t} - \alpha). \tag{2.36c}$$

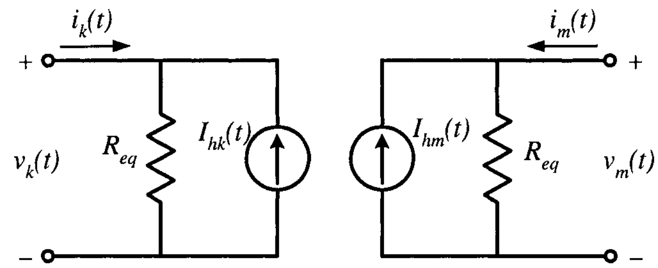


Figure 2.7: EMTP equivalent circuit for Marti's frequency-dependent line model

Consequently, combining (2.28) (2.29), $b_k(t)$ and $b_m(t)$, an EMTP equivalent circuit is obtained, shown in Fig. 2.7. History terms $I_h(t)$ (as both $I_{hk}(t)$ and $I_{hm}(t)$ have the same form of equations) is

$$I_h(t) = \left(V_{eq}(t-\Delta t) + \sum_{j=1}^{m} s_j(t) \right) / R_{eq} \tag{2.37}$$

$$= \left(\sum_{i=1}^{n} V_i(t-\Delta t) + \sum_{j=1}^{m} s_j(t) \right) / R_{eq} \tag{2.38}$$

where

$$V_i(t-\Delta t) = \frac{R_i^2 \frac{\Delta t}{C_i}}{(R_i + \frac{\Delta t}{2C_i})^2} i_i(t-\Delta t) + \frac{R_i - \frac{\Delta t}{2C_i}}{R_i + \frac{\Delta t}{2C_i}} V_i(t-2\Delta t) \tag{2.39}$$

$$s_j(t) = M \cdot s_j(t-\Delta t) + P \cdot f(t-\tau) + Q \cdot f(t-\tau-\Delta t). \tag{2.40}$$

2.1.6 Genetic Algorithms

The Genetic Algorithm (GA) is a probabilistic-rule based global search method mimicking natural biological evolution [48]. As generation-based algorithms, GAs operate on a population of potential solutions, or individuals. By applying the principle of survival of the fittest individuals, GAs are to produce better and better results to a solution in the following generations.

Borrowed from natural genetics by GAs, individuals are encoded as strings, or *chromosomes*, so that unique mappings are created between the *chromosome values* (*genotypes*) and *decision variables* (*phenotype*) of the problem. Chromosomes can be represented by binary, ternary, integer, real value, *etc.* [47, 48]. In order to access the performance, or in genetic terminology — fitness of the individuals, phenotypes obtained by decoding chromosomes are evaluated in the problem domain by objective functions. The objective functions are used to characterize the individual's performance from decision variables. They establish the basis for selection of pairs of individuals that will be mated together during reproduction [47, 48]. This can be explained as an individual's ability to survive in natural world.

Depending on the problem size and characteristics, a set of individuals or a population is first generated and evaluated by objective functions and each individual is given a fitness value. At each generation, selection phase and reproduction phase are carried out. Based on the fitness values of individuals, the selection biases towards more fit individuals. Relatively more highly fit individuals have a higher probability of being chosen for mating or recombination. The most commonly used methods in selection are Roulette Wheel Selection (RWS) and Stochastic Universal Sampling (SUS) selection [47,48].

Reproduction phase commonly includes two steps — recombination and mutation. In the recombination step, or crossover step, operators are applied to exchange genetic information between pairs or groups of individuals. Such operations are not necessarily performed on all chromosomes of the population. Instead, a probability is applied. Depending on the encodings of chromosomes, available recombination operation methods are single-point crossover, multi-point crossover, uniform crossover, intermediate recombination, line recombination, *etc.* [48]. Mutation, which is used to converge GAs to global optimum, is applied to new chromosomes generated after recombination. It causes individual chromosomes to change according to probabilistic rules.

Evaluations of the decoded chromosomes of individuals by objective functions are performed to assign fitness values to each individual, so that the selection process for the next generation is to be carried out. After generations of GA procedures, the average performance of individuals of a population is expected to increase, since the less fit individuals are superseded, as what happens in the natural world. Flowchart in Fig. 2.8 illustrates generic procedures.

GAs finish when certain termination criteria are met, *e.g.*, a certain number of generations, or the discovery of the expected chromosomes satisfying the problem [47, 48].

The above discussion shows that GAs has significant difference from the traditional analytical search and optimization methods. They have the merits of [47,48]

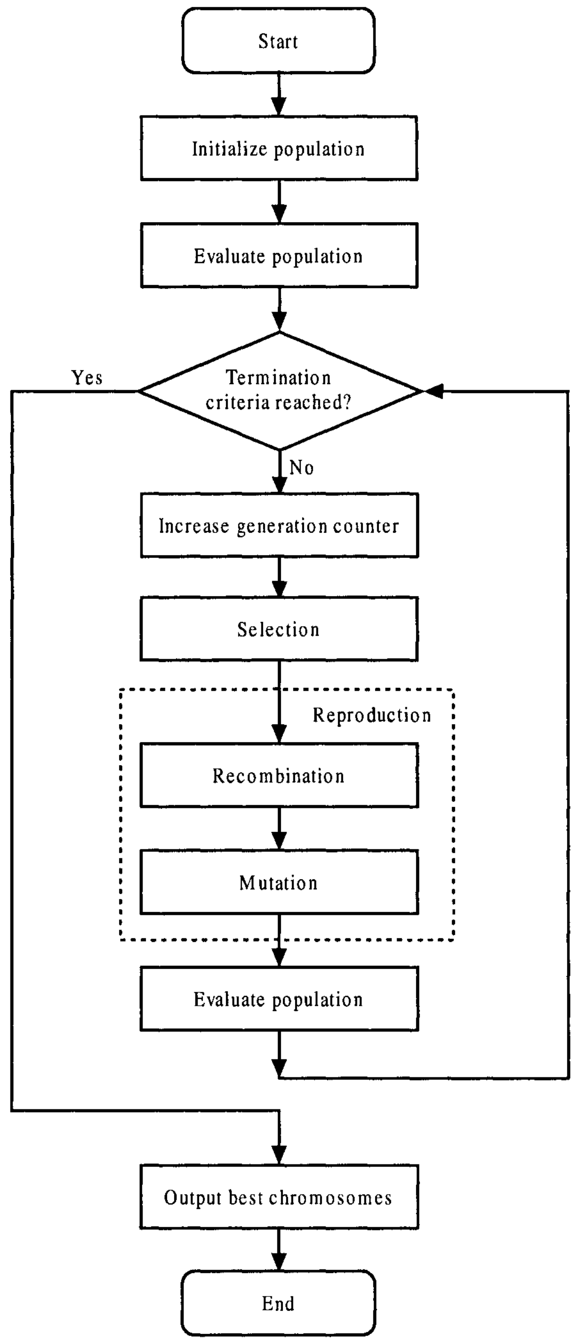


Figure 2.8: Flowchart of Genetic Algorithms

- Search of a population of points instead of single-point search.
- Utilization of objective functions rather than derivative information or other aux-

iliary knowledge.

- Probabilistic rules instead of deterministic ones.

- Encoding the parameters rather than using the parameters themselves.

In the multi-objective optimization problems, GAs are potentially useful for identifying alternative solutions simultaneously. This is particularly helpful in finding best suitable deep region in multi-port external systems. Moreover, its globalism in searching is also important advantage to be utilized from GAs.

2.2 Surface Layer

The surface layer comprises reduced-order or simplified Marti's frequency-dependent line model, which was first proposed by L. Marti [37] used for the simulation of secondary and less important transmission lines. The reduced-order model implies reduced simulation time. Later successful approaches in real-time simulation [38–41] show that since it keeps much of the accuracy with respect to the full-order model, the low-order model is also suitable for real-time implementation.

The order of Marti's line model comprises the order of $Z_{eq}(s)$ from (2.21) approximating characteristic impedance $Z_c(\omega)$ and that of $P_a(s)$ from (2.31) approximating back-winded weighting function $P(\omega)$. For a typical 280km single-phase overhead transmission line of one two-bundle Drake conductor, the frequency response of $Z_{eq}(s)$ and $P_a(s)$ is shown in Figures 2.9 and 2.10. To fit such frequency response by rational functions, Bode's asymptotic technique is employed. Featuring automatic order determination, the ATP Line Constant Program generates an 18th order $Z_{eq}(s)$ and a 16th order $P_a(s)$ for the transmission line within a very low RMS-error% (below 0.3% by default). The generated output data file is listed in Appendix A.3 and the data file format is interpreted in [44].

In this case, the low-order approximation of $Z_c(\omega)$ and that of $P(\omega)$ are also done by applying the same asymptotic technique. The difference is lower order and relatively larger RMS-error% tolerance. The fitting results from ATP listed in Appendix A.4 are 2nd order of $Z_{eq}(s)$ and 1st order of $P_a(s)$, also shown in Figures 2.9 and 2.10. It is obvious that low-order model still maintains much of accuracy. For the purpose of

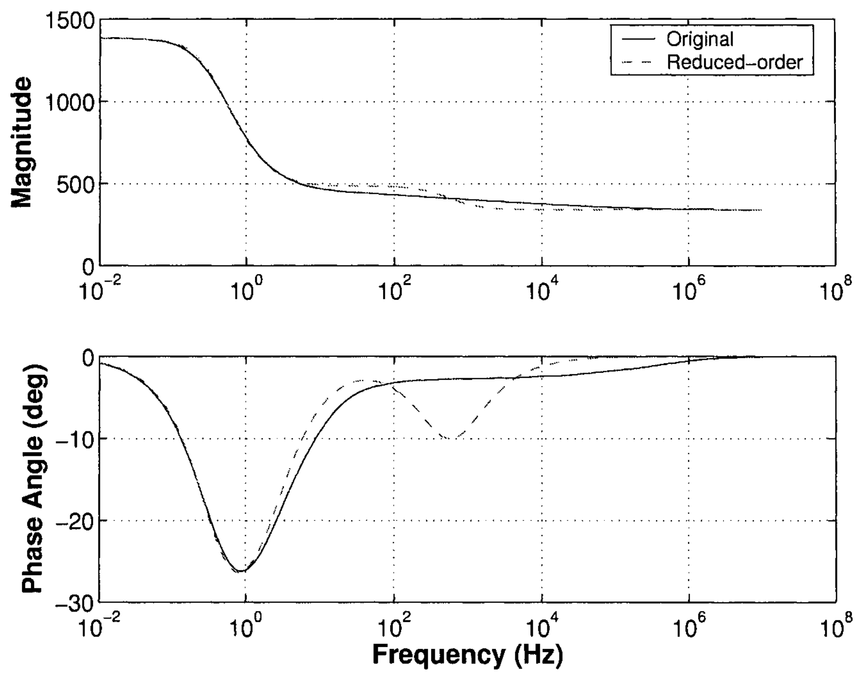


Figure 2.9: Frequency response of characteristic impedance $Z_c(\omega)$ and its low-order approximation

demonstrating the compensation technique of Robust TLNE model discussed in Section 2.4.6, $P_a(s)$ only gives 1st order, which results relatively higher deviations of phase angles in the high frequency range. The deviations due to reduced-order fitting is to be compensated by the deep region of TLNE. In addition, surface layer may also make up the deviations due to reduced-order deep region, especially in low frequency range such as DC and power frequency.

From individual lines which have the nodal equations (2.13a) and (2.13b), the admittance matrix of the original surface layer network is

$$\mathbf{Y}_{surface}(\omega) = \begin{bmatrix} \mathbf{Y}_{AA}(\omega) & \mathbf{Y}_{AB}(\omega) \\ \mathbf{Y}_{BA}(\omega) & \mathbf{Y}_{BB}(\omega) \end{bmatrix} \tag{2.41a}$$

and that of the reduced-order surface layer network

$$\tilde{\mathbf{Y}}_{surface}(\omega) = \begin{bmatrix} \tilde{\mathbf{Y}}_{AA}(\omega) & \tilde{\mathbf{Y}}_{AB}(\omega) \\ \tilde{\mathbf{Y}}_{BA}(\omega) & \tilde{\mathbf{Y}}_{BB}(\omega) \end{bmatrix} \tag{2.41b}$$

where subscript $_A$ stands for the ports connected to study zone, subscript $_B$ stands for those connected to deep region, and $\tilde{}$ designates simplification or approximation.

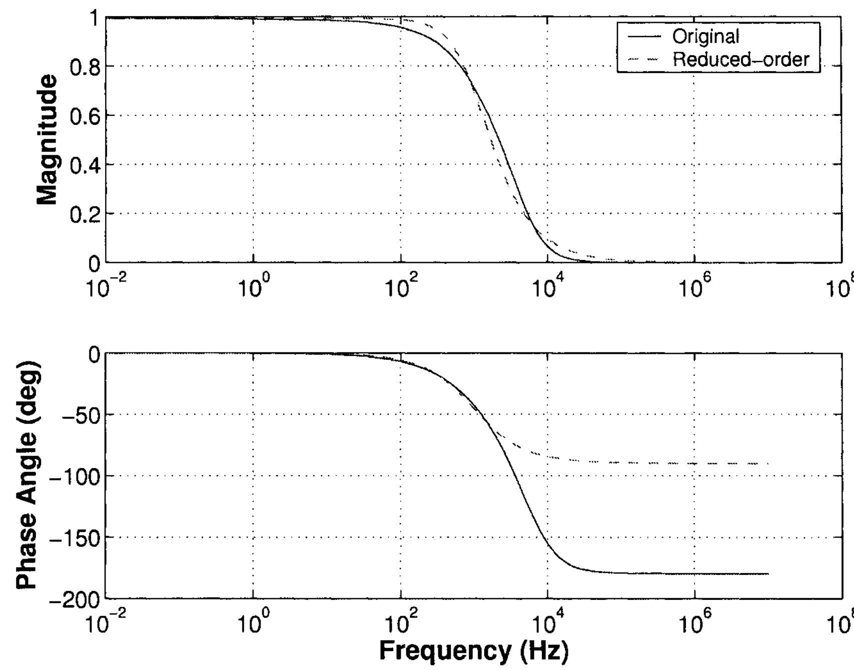


Figure 2.10: Frequency response of back-winded weighting function $P(\omega)$ and its low-order approximation

Since the external system is LTI, $\mathbf{Y}_{AB}(\omega) = [\mathbf{Y}_{BA}(\omega)]^T$ and $\tilde{\mathbf{Y}}_{AB}(\omega) = [\tilde{\mathbf{Y}}_{BA}(\omega)]^T$ where superscript ^{T} denotes transpose.

2.3 Deep Region

The fitting of external system by VF is stressed on relatively lower frequency range since high frequency transients do not travel very far in the external system. In TLNE, the deep region is further "insulated" from the study zone by the surface layer. Thus, the order of deep region may be significantly reduced. This gives the idea that the deep region can be represented by a low-order rational function (matrix) of FDNE model in (2.6), which not only represents the transient behavior of the original external system excluding the transmission network in surface layer, but also compensates for the differences due to the order reduction of transmission lines in surface layer. The deep region can be obtained by low-order VF approximation with emphasis on lower frequency range and Sequential Quadratic Programming (SQP) [21,22], or by the proposed robust approach which utilizes GAs and CLLSQ with optimal order selection introduced in the subsequent sections.

2.4 Finding the First Approximations with GAs

The first approximation of external system input admittance [21,22], is the initial mathematical combination of admittance matrix $\tilde{\mathbf{Y}}_{surface}(\omega)$ of the surface layer constituting reduced-order lines and $\tilde{\mathbf{Y}}_{deep}(\omega)$ of deep region comprising low-order FDNE generated by VF. It is obtained from [21,22]

$$\tilde{\mathbf{Y}}^{0}_{input}(\omega) = \tilde{\mathbf{Y}}^{0}_{AA}(\omega) - \tilde{\mathbf{Y}}^{0}_{AB}(\omega)(\tilde{\mathbf{Y}}^{0}_{BB}(\omega) + \tilde{\mathbf{Y}}^{0}_{deep}(\omega))^{-1}\tilde{\mathbf{Y}}^{0}_{BA}(\omega) \tag{2.42}$$

where the superscript ^{0} denotes "first" since the subsequent optimizations (*e.g.*, SQP) are to be carried out; $\tilde{\mathbf{Y}}^{0}_{AA}(\omega)$, $\tilde{\mathbf{Y}}^{0}_{AB}(\omega)$, $\tilde{\mathbf{Y}}^{0}_{BA}(\omega)$ and $\tilde{\mathbf{Y}}^{0}_{BB}(\omega)$ corresponds to the blocks of the first approximation of surface layer admittance $\tilde{\mathbf{Y}}^{0}_{surface}(\omega)$ in (2.41b); the first approximation of deep region admittance $\tilde{\mathbf{Y}}^{0}_{deep}(\omega)$ has the form of (2.6), and $\tilde{\mathbf{Y}}^{0}_{input}(\omega)$ is the first approximation of external system input admittance. The ultimate goal of building TLNE is to match $\tilde{\mathbf{Y}}_{input}(\omega)$ with the original external system admittance $\mathbf{Y}_{input}(\omega)$ as close as possible while ensuring stability and passivity of the model and frequency response at DC and power frequency.

2.4.1 Problems in Finding the First Approximations

In frequency domain, compared to the original $\mathbf{Y}_{input}(\omega)$, the $\tilde{\mathbf{Y}}^{0}_{input}(\omega)$ will, to some extent, exhibit deviations. Reducing those deviations necessitates the further optimizations to be carried out. Due to the nonlinear nature of the parameters in surface layer and deep region with respect to $\mathbf{Y}_{input}(\omega)$, (2.42) must first be linearized with respect to the parameters to be optimized, and Nonlinear Programming (NLP) such as SQP is implemented since passivity criteria are also nonlinear constraints [21,22]. However, NLP method does not guarantee convergence and is computational expensive. Moreover, the first approximation of the deep region obtained by low-order VF with emphasis on lower frequency range is not able to ensure best choice of the deep region. This is because that both the selection of weighting function and that of order in VF are very arbitrary. In Example 1 in Section 2.6, VF produce pretty good results. However, VF fails to produce good first approximation for deep region in Example 2 in Section 3.3, whose external system has very complicated frequency response. Thus, using VF alone can not always ensure a relatively good first approximation. It is necessary to find out alternative methods that are able to generate better first approximations for TLNE, if VF is unsuccessful to do so. A close-to-original first approximation also leads to only requiring a small fine-tuning in the subsequent optimizations, which can probably be fulfilled by least-square optimization instead of NLP. The least-square optimization is faster and more importantly, has guaranteed convergence.

2.4.2 Application of GAs

The idea for finding reduce-order deep region is based on the fact that in frequency domain, each resonant peak of original deep region admittance $\mathbf{Y}_{deep}(\omega)$ are likely to produce a resonance peak in the input admittance of external system $\mathbf{Y}_{input}(\omega)$. However, due to the insulation of surface layer, some peaks in $\mathbf{Y}_{deep}(\omega)$ are insensitive to $\mathbf{Y}_{input}(\omega)$. This gives the idea that removing some resonant peaks in deep region will impose little effect on $\mathbf{Y}_{input}(\omega)$. Thus, with the utilization of GAs, a better first approximation can be found by globally selecting the resonant peaks of the original deep region that have more significant effect on $\mathbf{Y}_{input}(\omega)$ in the designated low orders of deep region FDNE model.

This is a multi-variable optimization with nonlinear passivity constraint in single-port external systems and a multi-variable multi-objective optimization with the constraints in multi-port external systems. The GAs introduced in Section 2.1.6 are well-suited for solving the problem. Especially in case of multi-port external systems, the merits of GAs in multi-objective optimizations are fully utilized. Other methods such as Simulated Annealing (SA) are also able to solve this problem. However, in this thesis, GAs are applied. In fact, one of the most significant improvements in the robust approach compared to the existing one [21, 22] is the application of GAs in obtaining better first approximations. In order to apply GAs in global searching of best deep regions, data pre-processing steps must be carried out.

2.4.3 Data Preparations For GAs

The sum-of-partial-fraction form in (2.6) is desirable for GAs' global selection. First, the frequency response of the original deep region $\mathbf{Y}_{deep}(\omega)$ is obtained by frequency scan in ATP [44], which is explained in Section 2.1.2. In order to stress on lower frequency range, during frequency scan, frequency points are logarithmically distributed from low frequency, *e.g.*, 10Hz, to a high frequency limit f_{max}. In common practice, the upper frequency limit f_{max} is where frequency response becomes virtually flat and constant, *e.g.*, 1MHz. It is however demonstrated later in Example 1 (Section 2.6) and Example 2 (Section 3.3) that due to the insulation of surface layer, resonant peaks of obtained deep region are mainly distributed in lower frequency range. Therefore, the upper frequency limit in frequency scan can be reduced to relatively lower frequencies, *e.g.*, 10kHz, which is demonstrated in the modeling of AIES in Section 4.4. The main advantage of lowering upper frequency limit is the lower order rational function (matrix) generated by full-order VF, which in turn reduces computational time of GAs and subsequent optimizations since unnecessary high-frequency resonant peaks in deep region are eliminated. With this method, the constant term(s) in (2.6) of the fitted rational function (matrix) may not be very accurate since the frequency response is not constant in the higher frequency range. Nonetheless, this can be compensated by surface layer during later optimizations.

In addition to the above frequency points in frequency scan, one extra frequency point at DC is considered, since DC offset is important in time-domain simulation as well. In ATP, frequency scan can not be done at DC, so a frequency point close to DC, *e.g.*, 10^{-10}Hz is supplied.

In the next step, the sum-of-partial-fraction form is obtained by applying VF to deep region admittance (matrix) $\mathbf{Y}_{deep}(\omega)$ in frequency domain with low RMS-error% to produce a high-order proper rational function (matrix) $\mathbf{Y}_{deep}^{VF}(s)$ in s-domain which is, in MIMO case, a matrix that shares a common set of poles. In this step, frequency point at DC is not included, since it is to be matched in the later optimizations. Moreover, the inclusion of this point will very likely generate rank deficiency warnings during using VF in MATLAB, which result in an undesirable $\mathbf{Y}_{deep}^{VF}(s)$. Then the $\mathbf{Y}_{deep}^{VF}(s)$ is to be fed into GAs.

2.4.4 GA Parameters and Flowchart

It is noticed that $\mathbf{Y}_{deep}^{VF}(s)$ commonly has both real poles and complex poles. In our experience, conforming the logarithmical distribution of frequency points, after adequate iterations, VF only generates less than 6 real poles which belong to lower frequency range, *e.g.*, below 200Hz and all other complex poles which are commonly not in low frequency range. Therefore, the partial fractions of complex conjugate pairs are considered to be processed by GAs, whereas the ones of real poles are all chosen to be the partial fractions for deep region. Keeping the partial fractions of real poles is also very helpful in ensuring response at DC and power frequency with a little price of possibly increasing the deep region order by 2 to 4. Thus, each complex conjugate pair of partial fractions is treated as one entity and all pairs are indexed. Considering the searching of best-suited complex conjugate partial fractions, decision variables in GAs are designated to the integer indices of those partial fractions, by which chromosomes are encoded. To facilitate evolution process in GAs, real-valued representation is used, since it is a more efficient encoding in GAs [47]. Then the integers can be obtained by rounding the real-valued chromosomes. However, after rounding, some of the integers may appear as duplicate ones. To solve this problem, population initialization and mutation routines are extended for corrections to make sure all integers are not same with each other in every chromosome.

In this particular problem, GA parameters have to be properly set up for better efficiency. According to Section 2.1.6, the flowchart in Fig. 2.8 and the problem characteristics, the following guidelines are provided

- Population size is in accordance with the order of $\mathbf{Y}_{deep}^{VF}(s)$. In Example 1 in Section 2.6, the population size is 500 corresponding to $\mathbf{Y}_{deep}^{VF}(s)$'s order of 60. In Example 2 in Section 3.3, since the order of $\mathbf{Y}_{deep}^{VF}(s)$ is 220, the population size is at least 2000. The larger the population size generated, the faster the GAs will converge.

- Number of generations is not essential since other criteria dominantly control the termination of GAs. However, a too small number may lead to unacceptable results. In this problem, the number of generations should be at least 50.

- Generation gap is recommended to be the default value 0.9.

- Roulette Wheel Selection (RWS) is used since it more obeys the probabilistic rules than SUS [48].

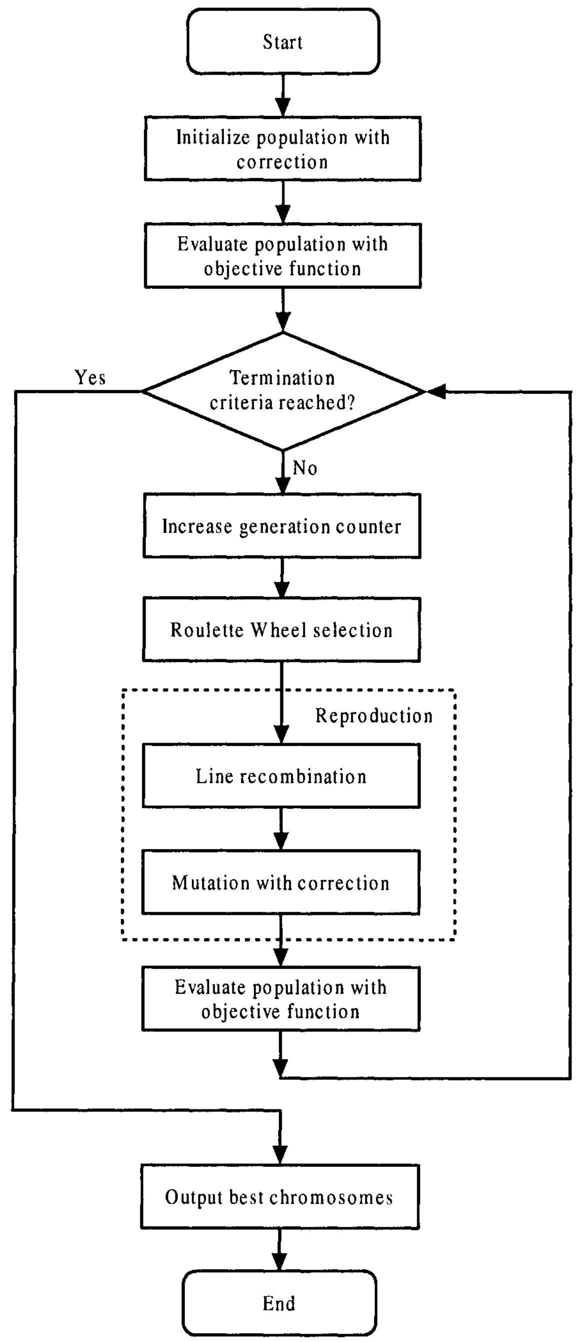


Figure 2.11: Flowchart of Genetic Algorithm for Robust TLNE

- Line recombination method with the probability of 0.8-0.95 is used. This is because the chromosomes are encoded by real numbers and the boundaries of selection scope must also be reached [48].

- Mutation with the probability of less than 0.7% is recommended.
- GA termination criteria are such emphasized that for a number of generations, the best value of objective function does not improve any more. A number between 30 to 50 is recommended.
- The best fitted individual generated by GAs does not guarantee the best deep region since further optimizations are to be carried out. Thus, the first N_{best} (*e.g.*, 5) number of best individuals are selected for future optimizations. Especially in multi-port system case, this strategy is more useful in finding better final results, which is examined in Example 2 in Section 3.3.

A detailed flowchart specific to this problem is shown in Fig. 2.11.

2.4.5 GA Objective Function

From above explanations on GAs' configurations, the admittance (SISO) or the elements of admittance matrix (MIMO) of the deep region $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ generated by GAs have the following form

$$Y(s) = d + \sum_{i=1}^{n_r} \frac{c_i}{s - a_i} + \sum_{k=1}^{n_c/2} \left(\frac{c_k}{s - a_k} + \frac{c_k^*}{s - a_k^*} \right) \tag{2.43}$$

where for complex conjugate pairs,

$$a_k = a_k' + ja_k'',\ a_k^* = a_k' - ja_k'',\ c_k = c_k' + jc_k'',\ c_k^* = c_k' - jc_k'' \tag{2.44}$$

n_r is the number of real poles, n_c is that of complex poles, and the order of deep region $n_{deep} = n_r + n_c$. It is also noticed that $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ still shares a common set of poles.

Calculation of $\tilde{\mathbf{Y}}^0_{input}(\omega)$ is based on the following equation

$$\tilde{\mathbf{Y}}^0_{input}(\omega) = \mathbf{Y}_{AA}(\omega) - \mathbf{Y}_{AB}(\omega)(\mathbf{Y}_{BB}(\omega) + \tilde{\mathbf{Y}}^0_{deep}(\omega))^{-1}\mathbf{Y}_{BA}(\omega) \tag{2.45}$$

Frequency points do not include DC, which will be optimized later. In (2.45), the surface layer is represented by the original admittance network $\mathbf{Y}_{surface}(\omega)$. This is because that in GAs, the search of the most significant resonant peaks is the goal, rather than compensate the simplified surface layer at same time. Both surface layer and deep region undergo further optimization later.

Since GAs try to find out the best low-order deep region $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ that minimizes the difference between $\tilde{\mathbf{Y}}^0_{input}(\omega)$ and $\mathbf{Y}_{input}(\omega)$, the RMS-error or Frobenius Norm $\left\|\mathbf{Y}_{input}(\omega) - \tilde{\mathbf{Y}}^0_{input}(\omega)\right\|^2_F$ defines one part of the objective function. In addition, GAs generated $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ must be positive-real. Thus the objective function is defined as [23]

$$f_{obj} = \left\|\mathbf{Y}_{input}(\omega) - \tilde{\mathbf{Y}}^0_{input}(\omega)\right\|^2_F + \mu = \sum_{i,j=1}^{m} \left|\mathbf{Y}_{input,ij}(\omega) - \tilde{\mathbf{Y}}^0_{input,ij}(\omega)\right|^2 + \mu \quad (2.46)$$

where $\mathbf{Y}_{input,ij}(\omega)$ is the ij-th elements of $\mathbf{Y}_{input}(\omega)$; μ denotes a penalty term when passivity criterion violation occurs in deep region. If this criterion is violated, μ will a big enough positive number, otherwise $\mu = 0$. This ensures that the outputs from GAs are the best fitted deep regions, which are both stable and passivity. For passivity violation case, the value of μ is such a big value that individuals with the violation are very likely to be eliminated during selection step in later generations. In our experience, the value of 0.1 to 0.5 is used since the RMS-error% is not expected to be very high.

2.4.6 Compensation Technique for GAs

Removing partial fractions from $\tilde{\mathbf{Y}}^{VF}_{deep}(s)$ may result in the system frequency response especially in lower frequency range showing pronounced deviation from $\mathbf{Y}_{deep}(\omega)$. Assuming the following n_{cs} number of partial fractions $\sum_{k=1}^{n_{cs}/2}\left(\frac{c_k}{s-a_k} + \frac{c_k^*}{s-a_k^*}\right)$ are chosen by GAs, *i.e.*, the number of $n_c - n_{cs}$ partial fractions $\sum_{i=1}^{(n_c-n_{cs})/2}\left(\frac{c_i}{s-a_i} + \frac{c_i^*}{s-a_i^*}\right)$ are deselected for deep region, we have the elements of deep region admittance matrix as

$$\tilde{Y}(s) = d + \sum_{i=1}^{n_r} \frac{c_i}{s-a_i} + \sum_{k=1}^{n_{cs}/2}\left(\frac{c_k}{s-a_k} + \frac{c_k^*}{s-a_k^*}\right) \quad (2.47)$$

where a_k and c_k have the same meanings as in (2.44). Then the deviations with respect to full-order $Y(\omega)$ in frequency response are most significant in the lower frequency range and have the DC value of

$$Y_{dc} = -\sum_{k=1}^{(n_c-n_{cs})/2}\left(\frac{2(a'_k c'_k + a''_k c''_k)}{(a'_k)^2 + (a''_k)^2}\right) \quad (2.48)$$

which will deteriorate the corresponding $\tilde{\mathbf{Y}}^0_{input}(\omega)$. This can be compensated by adding one extra partial fraction of a real pole and a real residue $\frac{c_{kc}}{s-a_{kc}}$ where a_{kc} and c_{kc}

are [23]

$$a_{kc} = \max(a_k''), \quad 1 \leq k \leq n_{cs} \tag{2.49}$$

$$c_{kc} = -a_{kc} Y_{dc} \tag{2.50}$$

By adding one more order to the deep region, deviations resulted from removing partial fractions are minimized. Moreover, this partial fraction also increases the flexibilities of later CLLSQ optimizations. It is observed that the partial fraction may not be positive-real. However, this is not considered as an issue since the generated $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ is positive-real.

2.4.7 Building the First Approximation of Input Admittance

After $\tilde{\mathbf{Y}}^0_{surface}(\omega)$ and $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ are obtained, the first approximation of external system input admittance is generated through (2.42). Notice that in building the first approximation, reduced-order transmission line models in surface layer $\tilde{\mathbf{Y}}^0_{surface}(\omega)$ are used. Since deep regions generated by GAs are multiple, a series of first approximations of input admittance are built.

2.5 Constrained Linearized Least-Square Optimization

Theoretically, the generated input admittance $\tilde{\mathbf{Y}}^0_{input}(\omega)$ from Section 2.4 is very close to the original $\mathbf{Y}_{input}(\omega)$. Nonetheless, reduced-order surface layer $\tilde{\mathbf{Y}}^0_{surface}(\omega)$ and GAs generated deep region $\tilde{\mathbf{Y}}^0_{deep}(\omega)$ are subject to further fine-tunings or optimizations to minimize the deviations between $\tilde{\mathbf{Y}}_{input}(\omega)$ and $\mathbf{Y}_{input}(\omega)$. Frequency points include DC since frequency response at DC is to be matched during optimization.

2.5.1 Parameters Subject to Optimization

Parameters eligible for optimization include the following:

- In surface layer, k_0 terms, all poles p_i and residues r_i of $Z_{eq}(s)$ in (2.22); all poles p_i and residues r_i of $P_a(s)$ in (2.32); propagation constants τ.
- In deep region, all constant terms d_{ik}, poles a_i and residues c_{ik} in (2.6), the elements of which have the form of (2.43).

However, the propagation constants τ represents the length of the line and not recommended for changing, so all τ in surface layer are not chosen for optimization. Our experience shows that modifying the poles shared by all elements of deep region admittance matrix does not help to improve $\tilde{\mathbf{Y}}_{input}(\omega)$ with respect to $\mathbf{Y}_{input}(\omega)$, rather, sometimes optimizing those poles may cause larger RMS-error% with respect to the cases without optimization of those poles. Constructing surface layer admittance matrix from Marti's line model parameters also shows that the partial derivatives with respect to parameters of $P_a(s)$ are very complicated, which is more likely to cause convergence problems and result in larger RMS-error% in optimization than other parameters. So the parameters of $Z_{eq}(s)$ are more suitable for optimization. Therefore, parameters considered for further optimization are

- In surface layer, k_0 terms, all poles p_i and residues r_i of $Z_{eq}(s)$ in (2.22).
- In deep region, all constant terms d_{ik} and residues c_{ik} in (2.6), the elements of which have the form of (2.43).

It has been pointed out in [21,22] that optimization of surface layer parameters are considered only a last resort for accuracy. However, in our experience, since it particularly improves the frequency response at DC and power frequency, optimizing surface layer parameters is chosen by default in the Robust TLNE.

2.5.2 Linearization

Equation (2.42) expressed in s-domain as

$$\tilde{\mathbf{Y}}_{input}(s) = \tilde{\mathbf{Y}}_{AA}(s) - \tilde{\mathbf{Y}}_{AB}(s)(\tilde{\mathbf{Y}}_{BB}(s) + \tilde{\mathbf{Y}}_{deep}(s))^{-1}\tilde{\mathbf{Y}}_{BA}(s) \tag{2.51}$$

is linearized in frequency domain with respect to parameters of surface layer and deep region at the point of $\tilde{\mathbf{Y}}^0_{input}(\omega)$ to produce [21,22]

$$\mathbf{Y}_{input}(\omega) = \tilde{\mathbf{Y}}^0_{input}(\omega) + \mathbf{J}(\omega)\Delta\mathbf{x} \tag{2.52}$$

i.e.,

$$\Delta\mathbf{Y}_{input}(\omega) = \mathbf{Y}_{input}(\omega) - \tilde{\mathbf{Y}}^0_{input}(\omega) = \mathbf{J}(\omega)\Delta\mathbf{x} \tag{2.53}$$

with

$$\mathbf{x} = \mathbf{x}_0 + \Delta\mathbf{x} \tag{2.54}$$

where $\tilde{\mathbf{Y}}^0_{input}(\omega)$ stands for the first approximation of input admittance matrix discussed in last section, $\mathbf{J}(\omega)$ is the Jacobian matrix, $\mathbf{x}$ is the model parameter column vector considered for optimization and $\Delta\mathbf{x}$ is the model parameter change column vector. The overdetermined linear equation (2.53) shows a least-square sense, which implies that further improvements of $\tilde{\mathbf{Y}}^0_{input}(\omega)$ with respect to $\mathbf{Y}_{input}(\omega)$ can be obtained through recursive evaluation of (2.53) with (2.54).

2.5.3 The Jacobian Matrix

To evaluate the Jacobian Matrix, partial derivatives with respect to parameters to be optimized are obtained first. From (2.51), in surface layer, the partial derivatives with respect to a real parameter ρ are given as [21,22]

$$\begin{aligned}\frac{\partial\tilde{\mathbf{Y}}_{input}(s)}{\partial\rho} = &\frac{\partial\tilde{\mathbf{Y}}_{AA}(s)}{\partial\rho} - \frac{\partial\tilde{\mathbf{Y}}_{AB}(s)}{\partial\rho}[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\tilde{\mathbf{Y}}_{BA}(s)\\ &-\tilde{\mathbf{Y}}_{AB}(s)[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\frac{\partial\tilde{\mathbf{Y}}_{BA}(s)}{\partial\rho}\\ &+\tilde{\mathbf{Y}}_{AB}(s)[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\frac{\partial\tilde{\mathbf{Y}}_{BB}(s)}{\partial\rho}[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\tilde{\mathbf{Y}}_{BA}(s)\end{aligned} \tag{2.55}$$

Consider a transmission line connecting between Bus N_1 and Bus N_2 in surface layer which has parameters of τ, $Z_{eq}(s)$ and $P_a(s)$ in s-domain, the elements of its admittance matrix $\mathbf{Y}_{TL}(s)$ in surface layer are given as

$$(\mathbf{Y}_{TL}(s))_{ij} = \begin{cases} -\dfrac{e^{-\tau s}}{Z_{eq}(s)}\left(\dfrac{P_a(s)+P_a^{-1}(s)}{P_a(s)-P_a^{-1}(s)}\right) & \text{when } i=j=N_1 \text{ or } i=j=N_2\\ \dfrac{e^{-\tau s}}{Z_{eq}(s)}\left(\dfrac{2}{P_a(s)-P_a^{-1}(s)}\right) & \text{when } i=N_1, j=N_2 \text{ or } i=N_2, j=N_1\\ 0 & \text{otherwise}\end{cases} \tag{2.56}$$

Thus, partial derivatives of $\tilde{\mathbf{Y}}_{surface}(s)$ with respect to the parameters of a specific line $Z_{eq}(s)$ in (2.22) are obtained, which is shown in Table 2.1. $\partial\tilde{\mathbf{Y}}_{AA}(s)/\partial\rho$, $\partial\tilde{\mathbf{Y}}_{AB}(s)/\partial\rho$, $\partial\tilde{\mathbf{Y}}_{BA}(s)/\partial\rho$ and $\partial\tilde{\mathbf{Y}}_{BB}(s)/\partial\rho$ in (2.55) are given by partitioning $\partial\tilde{\mathbf{Y}}_{surface}(s)/\partial\rho$ according to the buses connecting to study zone (subscript $_A$) and those connecting to deep region (subscript $_B$).

Parameter ρ in $Z_{eq}(s)$	Partial derivative $\partial\tilde{\mathbf{Y}}_{surface}(s)/\partial\rho$
k_0	$-\dfrac{1}{Z_{eq}(s)}\mathbf{Y}_{TL}(s)$
r_i	$-\dfrac{1}{(s+p_i)Z_{eq}(s)}\mathbf{Y}_{TL}(s)$
p_i	$\dfrac{r_i}{(s+p_i)^2 Z_{eq}(s)}\mathbf{Y}_{TL}(s)$

Table 2.1: Partial derivatives with respect to parameters of $Z_{eq}(s)$ in surface layer

Partial derivatives with respect to deep region parameters are obtained by [21,22]

$$\frac{\partial\tilde{\mathbf{Y}}_{input}(s)}{\partial\rho} = \tilde{\mathbf{Y}}_{AB}(s)[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\frac{\partial\tilde{\mathbf{Y}}_{deep}(s)}{\partial\rho}[\tilde{\mathbf{Y}}_{BB}(s)+\tilde{\mathbf{Y}}_{deep}(s)]^{-1}\tilde{\mathbf{Y}}_{BA}(s) \tag{2.57}$$

Parameter ρ in the N-th diagonal elements of $\tilde{\mathbf{Y}}_{deep}(s)$	The ij-th matrix elements of partial derivative $\partial\tilde{\mathbf{Y}}_{deep}(s)/\partial\rho$
d	$\begin{cases} 1 & \text{when } i=j=N \\ 0 & \text{otherwise} \end{cases}$
c_k	$\begin{cases} \dfrac{1}{s-a_k} & \text{when real and } i=j=N \\ f_k(s) & \text{when complex and } i=j=N \\ 0 & \text{otherwise} \end{cases}$

Table 2.2: Partial derivatives with respect to parameters of diagonal elements in deep region admittance matrix

!h However, for deep region defined by (2.6), in order to find out $\partial\tilde{\mathbf{Y}}_{deep}(s)/\partial\rho$, diagonal and off-diagonal elements in (2.6) have to be considered separately. Partial derivatives are with respect to real parameters. Therefore, the complex conjugate pairs in (2.43) are considered as real and imaginary parts shown in (2.44). For the N-th diagonal element, partial derivatives of $\tilde{\mathbf{Y}}_{deep}(s)$ with respect to a real parameter ρ are shown in Table 2.2, where $f_k(s)$ is defined as

$$f_k(s) = \begin{cases} \dfrac{1}{s-a_k} + \dfrac{1}{s-a_k^*} & \text{for real part } c' \\ j\left(\dfrac{1}{s-a_k} - \dfrac{1}{s-a_k^*}\right) & \text{for imaginary part } c'' \end{cases} \tag{2.58}$$

Partial derivatives of $\tilde{\mathbf{Y}}_{deep}(s)$ with respect to parameters of off-diagonal elements are very similar to diagonal ones except that two equal elements exist in the matrix due to the symmetry of $\tilde{\mathbf{Y}}_{deep}(s)$. For an off-diagonal element at N_1 row and N_2 column, partial derivatives $\partial\tilde{\mathbf{Y}}_{deep}(s)/\partial\rho$ are given in Table 2.3, where $f_k(s)$ is also defined by (2.58).

Parameter ρ in the N_1N_2-th off-diagonal elements of $\tilde{\mathbf{Y}}_{deep}(s)$	The ij-th matrix elements of partial derivative $\partial\tilde{\mathbf{Y}}_{deep}(s)/\partial\rho$
d	$\begin{cases} 1 & \text{when } i=N_1, j=N_2 \text{ or } i=N_2, j=N_1 \\ 0 & \text{otherwise} \end{cases}$
c_k	$\begin{cases} \dfrac{1}{s-a_k} & \text{when real and } i=N_1, j=N_2 \text{ or } i=N_2, j=N_1 \\ f_k(s) & \text{when complex and } i=N_1, j=N_2 \text{ or } i=N_2, j=N_1 \\ 0 & \text{otherwise} \end{cases}$

Table 2.3: Partial derivatives with respect to parameters of off-diagonal elements in deep region admittance matrix

With the partial derivatives formulated above, the Jacobian matrix in s-domain is formed corresponding to the parameters to be optimized as

$$\mathbf{J}_1(s) = \left[\begin{array}{cccc} \dfrac{\partial\tilde{\mathbf{Y}}_{input}(s)}{\partial\rho_1} & \dfrac{\partial\tilde{\mathbf{Y}}_{input}(s)}{\partial\rho_2} & \cdots & \dfrac{\partial\tilde{\mathbf{Y}}_{input}(s)}{\partial\rho_{N_p}} \end{array} \right] \tag{2.59}$$

where N_p is the total number of parameters to be optimized. However, in multi-port external system case, each matrix $\partial\tilde{\mathbf{Y}}_{input}(s)/\partial\rho_k$ $(1 \leq k \leq N_p)$ must be expanded to obtain the form suitable for least-square sense. Because of the symmetrical nature of the external system admittance matrix, only the upper or lower triangular parts are expended, the purpose of which is to reduce computational burdens. Therefore, we have

$$\mathbf{J}(s) = \begin{bmatrix} \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_1}\right)_{11} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_2}\right)_{11} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_{N_p}}\right)_{11} \\ \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_1}\right)_{12} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_2}\right)_{12} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_{N_p}}\right)_{12} \\ \vdots & \vdots & \ddots & \vdots \\ \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_1}\right)_{N_{es}N_{es}} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_2}\right)_{N_{es}N_{es}} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(s)}{\partial \rho_{N_p}}\right)_{N_{es}N_{es}} \end{bmatrix} \tag{2.60}$$

where N_{es} is the total number of elements in upper/lower triangular part of external system admittance matrix, which has the same dimension of its partial derivative matrices. Since frequency points are discrete, (2.60) is further evaluated in frequency domain at discrete frequency points and stacked to form the final Jacobian matrix as

$$\mathbf{J}(\omega) =$$

$$\left[\begin{array}{cccc}
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_1}\right)_{11} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_2}\right)_{11} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_{N_p}}\right)_{11} \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_1}\right)_{11} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_2}\right)_{11} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_{N_p}}\right)_{11} \\
\vdots & \vdots & \ddots & \vdots \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_1}\right)_{11} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_2}\right)_{11} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_{N_p}}\right)_{11} \\
\hline
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_1}\right)_{12} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_2}\right)_{12} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_{N_p}}\right)_{12} \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_1}\right)_{12} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_2}\right)_{12} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_{N_p}}\right)_{12} \\
\vdots & \vdots & \ddots & \vdots \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_1}\right)_{12} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_2}\right)_{12} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_{N_p}}\right)_{12} \\
\hline
\vdots & \vdots & \ddots & \vdots \\
\hline
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_1}\right)_{N_{es}N_{es}} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_2}\right)_{N_{es}N_{es}} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_1)}{\partial \rho_{N_p}}\right)_{N_{es}N_{es}} \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_1}\right)_{N_{es}N_{es}} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_2}\right)_{N_{es}N_{es}} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_2)}{\partial \rho_{N_p}}\right)_{N_{es}N_{es}} \\
\vdots & \vdots & \ddots & \vdots \\
\left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_1}\right)_{N_{es}N_{es}} & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_2}\right)_{N_{es}N_{es}} & \cdots & \left(\frac{\partial \tilde{\mathbf{Y}}_{input}(\omega_{N_f})}{\partial \rho_{N_p}}\right)_{N_{es}N_{es}}
\end{array}\right] \tag{2.61}$$

where N_f is the number of discrete frequency points.

2.5.4 Iterative Least-Square Optimization

To apply least-square optimization, in multi-port external system case, all elements of the original input admittance matrix $\mathbf{Y}_{input}(\omega)$ and the calculated $\tilde{\mathbf{Y}}_{input}(\omega)$ of each iteration in (2.53) are also evaluated at the same frequency points as the Jacobian matrix and stacked to form a column vector as

$$\begin{aligned}\Delta\mathbf{Y}_{input}(\omega) = \\ \big[(\Delta\mathbf{Y}_{input}(\omega_1))_{11} \quad & (\Delta\mathbf{Y}_{input}(\omega_2))_{11} \quad \cdots \quad (\Delta\mathbf{Y}_{input}(\omega_{N_f}))_{11} \\ (\Delta\mathbf{Y}_{input}(\omega_1))_{12} \quad & (\Delta\mathbf{Y}_{input}(\omega_2))_{12} \quad \cdots \quad (\Delta\mathbf{Y}_{input}(\omega_{N_f}))_{12} \\ (\Delta\mathbf{Y}_{input}(\omega_1))_{N_{es}N_{es}} \quad & (\Delta\mathbf{Y}_{input}(\omega_2))_{N_{es}N_{es}} \quad \cdots \quad (\Delta\mathbf{Y}_{input}(\omega_{N_f}))_{N_{es}N_{es}}\big]^T\end{aligned} \tag{2.62}$$

Also due to the symmetry of $\Delta\mathbf{Y}_{input}(\omega)$, the same method as stacking the Jacobian matrix is applied. In addition, the sequence of stacking $\Delta\mathbf{Y}_{input}(\omega)$ must be in accordance with that of stacking the Jacobian matrix.

Thus, from (2.53), (2.61) and (2.62), an overdetermined linear equation $\Delta\mathbf{Y}_{input}(\omega) = \mathbf{J}(\omega)\Delta\mathrm{x}$ of finding $\Delta\mathrm{x}$ to minimize $\Delta\mathbf{Y}_{input}(\omega)$ in the least-square sense is constructed. However, the least-square sense does not apply to complex quantities. Nonetheless, in (2.53), $\Delta\mathbf{Y}_{input}(\omega)$ and $\mathbf{J}(\omega)$ are complex quantities. Therefore, real and imaginary parts are separated and stacked to produce

$$\begin{bmatrix} \mathrm{Re}(\Delta\mathbf{Y}_{input}(\omega)) \\ \mathrm{Im}(\Delta\mathbf{Y}_{input}(\omega)) \end{bmatrix} = \begin{bmatrix} \mathrm{Re}(\mathbf{J}(\omega)) \\ \mathrm{Im}(\mathbf{J}(\omega)) \end{bmatrix} \Delta\mathrm{x} \tag{2.63}$$

During the optimization, both positive-real criterion and algorithm convergence must be guaranteed, which leads to the concept of constrained optimizations. It has been discussed in [21, 22] that the NLP methods such as SQP are required to accomplish this task. Consequently, (2.63) has to be converted to the more complex quadratic form. However, the NLP methods are prone to divergence and are computationally expensive. In the Robust TLNE, since GAs are able to obtain better first approximations that are both stable and passive, *i.e.*, the first approximations are already in the feasible region in optimization sense and close to original, only minor fine-tunings are required. Therefore, instead of NLP, Constrained Linearized Least-Square (CLLSQ) optimization is employed.

In each iteration of evaluating $\Delta\mathrm{x}$, both stability and passivity conditions for surface layer and deep region are checked after adding parameter changes. Checking positive-real criterion for $\tilde{\mathbf{Y}}_{input}(\omega)$ is not necessary since the surface layer and deep region belongs to different models. As far as both surface layer and deep region are stable and positive-real, stable time-domain simulations are guaranteed. During the condition check, each transmission line in surface layer and the whole deep region are treated as separate entities. The entities with either condition violation result a positive number δ ($0 < \delta < 1$) is multiplied with the entities' parameters change (a part of $\Delta\mathbf{x}$), since they have changed too much. Another essential condition in the optimization is the decrease of RMS-error% in the model input admittance during subsequent iterations. If RMS-error% does not decrease compared to the last iteration, the whole parameter change vector $\Delta\mathbf{x}$ is also multiplied by δ. The optimal point or termination criterion is that for a consecutive number N_{op} times of multiplication of δ with $\Delta\mathbf{x}$, the RMS-error% of input admittance does not decrease. Therefore, (2.54) is re-written as

$$\mathbf{x} = \begin{cases} \mathbf{x}_0 + (\delta)^q \Delta\mathbf{x}_1 + \Delta\mathbf{x}_2 & \text{when PR or stability violations,} \\ \mathbf{x}_0 + (\delta)^q \Delta\mathbf{x} & \text{when larger RMS-error,} \\ \mathbf{x}_0 + \Delta\mathbf{x} & \text{otherwise (no violations).} \end{cases} \tag{2.64}$$

where $0 < q < N_{op}$, $\Delta\mathbf{x}_1$ are parameter changes of criterion violation and $\Delta\mathbf{x}_2$ are the parameters not violating criteria. δ is recommended for values from 0.1 to 0.5. The value of 5 to 10 for N_{op} is recommended.

2.5.5 Optimal Deep Region Order Determination

So far, the discussion in the Robust TLNE is only limited to a specific deep region order. However, to obtain the best suitable order of deep region, a number of deep region orders are applied to this problem. The intuitive idea is to construct a loop from n_{low} (n_{low} is greater than the number of real partial fractions) to n_{high} (n_{high} is less than the order of $\tilde{\mathbf{Y}}^{VF}_{deep}$) for deep region orders and find the deep region with lowest RMS-error% in each iteration of loops. Since GAs only choose complex conjugate pairs of deep region, in the loop, the order increases by two. Thus we obtain the order of deep region *v.s.* RMS-error% from n_{low} to n_{high}. The acceptable RMS-error% range is below 10% in our practice. Within this range, the optimal order is the one where in the orders lower than it, the RMS-error% increases dramatically, whereas in the orders higher than it, the RMS-error% does not decrease significantly. The later examples and case study illustrate this idea. In Example 1, Fig. 2.13 shows that the order of 13 for deep region are the optimal order. In Example 2, 21 is the best suitable order for deep region in TLNE model.

2.6 Example 1

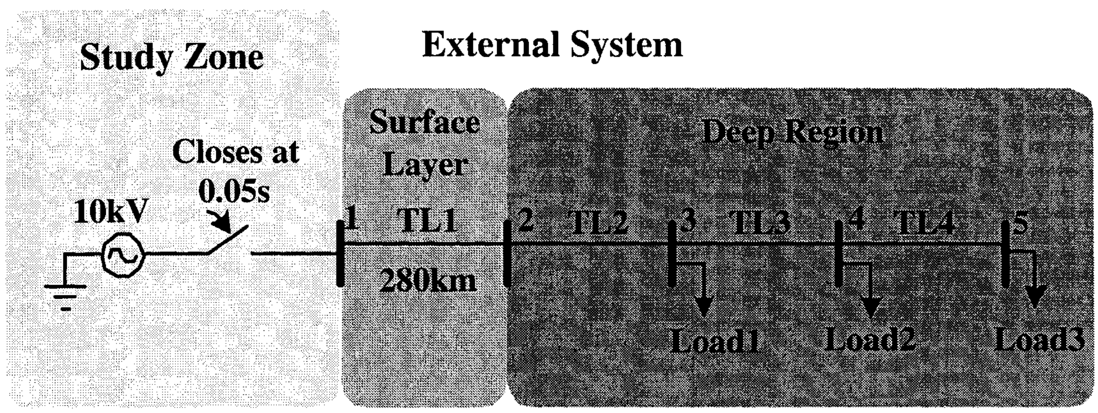


Figure 2.12: Example 1 system diagram and its partitioning

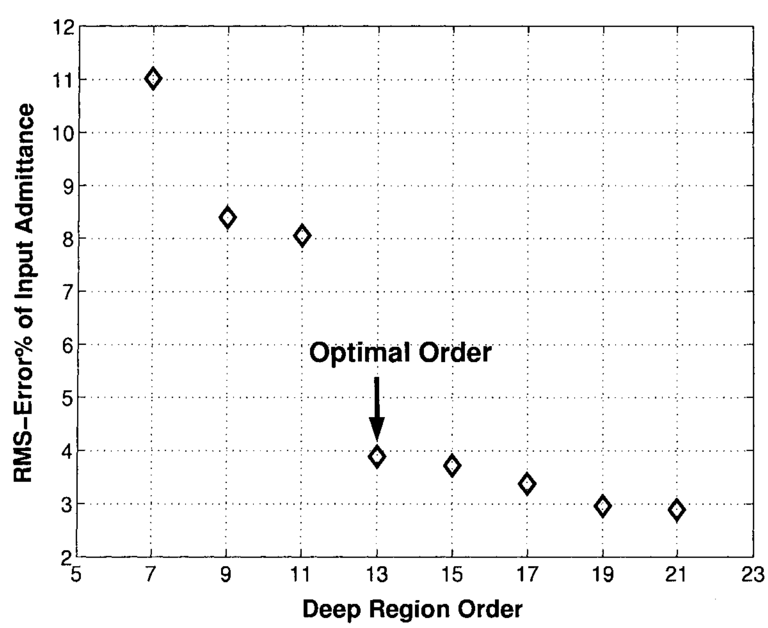


Figure 2.13: Example 1 RMS-error% of input admittance versus deep region order

Example 1, whose external system is a single-phase single-port passive network, shown in Fig. 2.12, is a transmission line energization case. Detailed parameters for the system are listed in Appendix A.1. The same Drake conductor and tower configuration are assumed in all transmission lines. A 60Hz 10kV ideal voltage source is switched to the passive part after 0.05s. Switching current is measured at Bus 1. Fig. 2.12 also shows partitioning the system into a study zone and an external system.

2.6.1 Generation of the Robust TLNE

Also shown in Fig. 2.12, the external system is further partitioned into a surface layer and a deep region. The surface layer consists of only one 280km transmission line TL1, since the deep region is less complex. The original deep region admittance (shown in Fig. 2.15) is fitted by a 40th-order rational function in s-domain with 0.346% RMS-error, which has 4 real poles all below 200Hz and 18 complex conjugate pole pairs. Then, the 18 pairs of complex conjugates are processed by GAs. Fig. 2.13 shows RMS-error% *v.s.* deep region order ranging from 7 to 21. It has been clearly identified that 13 is optimal order for deep region, which has the RMS-error of 3.893% in input admittance after CLLSQ optimization. At DC, the input admittance only has 0.0012% deviation. With 13th order for deep region, Fig. 2.15 shows the deep region frequency response generated by GAs and after CLLSQ optimization. Some resonant peaks in deep region are not selected due to their insensitivity to input admittance. On the other hand, the low-order deep region catches some major peaks of the original frequency response. Moreover, CLLSQ also compensates the deviations due to reduced-order surface layer models. Fig. 2.14 shows TL1 characteristic impedance in surface layer of reduced order and after CLLSQ optimization. It is observed that the change of surface layer parameters during CLLSQ helps to increase the accuracy of Robust TLNE. The resultant input admittance of Robust TLNE model, shown in Fig. 2.16, is very accurate. It is also noticed that by applying GAs, the first approximation of input admittance is already close to original. This verifies the accuracy of GAs in obtaining optimal deep region. Appendix A.3 and A.4 lists ATP data file of full system and Robust TLNE model, respectively.

2.6.2 Transient Simulations

Shown in Fig. 2.17, transient simulation of Bus 1 branch current at 10μs step size further verifies the indistinguishability between full model and Robust TLNE model. The total

Transient event	Simulation time	Full model	Robust TLNE	FDNE(18th)
Line energization	0.2s	0.173s	0.061s	0.092s

Table 2.4: Example 1 computational time comparison at time-step size $10\mu s$

simulation time T_{max} is 0.2s in this case. In order to emphasize the transient, only simulation time from 0.03s to 0.15s is displayed. In a Pentium IV 1.6GHz computer, the simulation of full model in ATP requires 0.173s, whereas the simulation of Robust TLNE model needs 0.061s to accomplish, which is about almost three times of save on computational time. For a similar RMS-error% FDNE model for the external system, VF generates an 18th-order rational function of 3.774% RMS-error, which requires 0.092s for the whole simulation. This is slightly higher than the one of Robust TLNE model. Table 2.4 shows simulation time comparison for different external system models. In a small system with less complex frequency response, the Robust TLNE model does not demonstrate significant computational save with respect to FDNE model. Later in Example 2, a much bigger computational save with high accuracy is observed. With existing TLNE approach, at 13th order for deep region, the model achieves 4.629% RMS-error on input admittance with respect to the original, which is slightly higher that the proposed approach. However, in Example 2, the RMS-error% of existing TLNE model is much higher than that of Robust TLNE model.

2.7 Summary

In this chapter, with discussions on some basic models, concepts and methods, the building of a Robust TLNE model for passive networks is explained in detail, . The external system is divided into a surface layer of reduced-order frequency-dependent transmission lines and a deep region of low-order FDNE model. GAs are applied to find best suitable deep region and CLLSQ optimization is used to fine-tune model parameters and further improve accuracy including DC frequency. One single-phase single-port example with a passive external system is presented. Comparisons between full model, Robust TLNE and FDNE are made in terms of computational time and accuracy. Transient simulation result verifies the accuracy of the Robust TLNE model and simu-

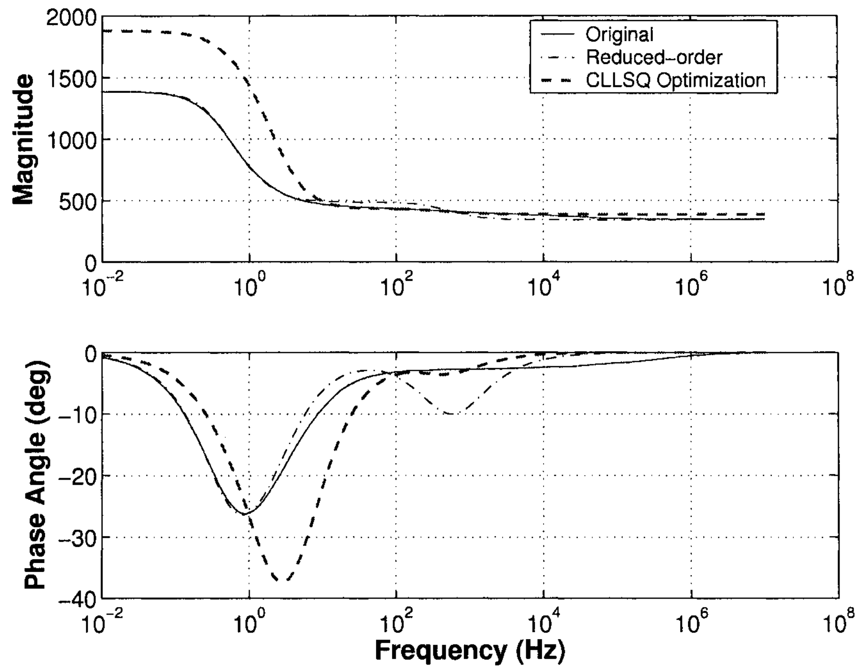


Figure 2.14: Example 1 TL1 characteristic impedance

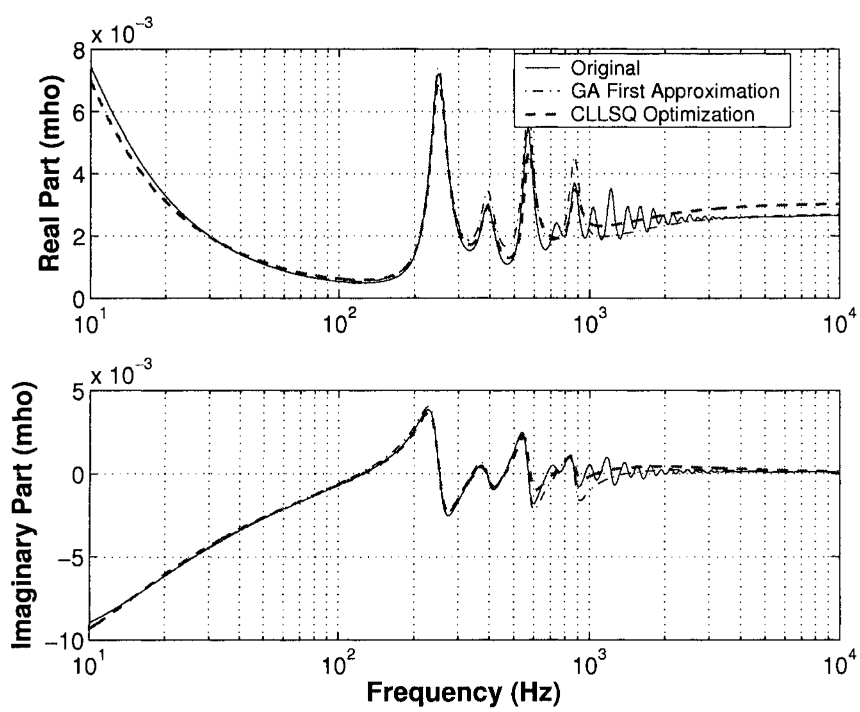


Figure 2.15: Example 1 deep region admittance

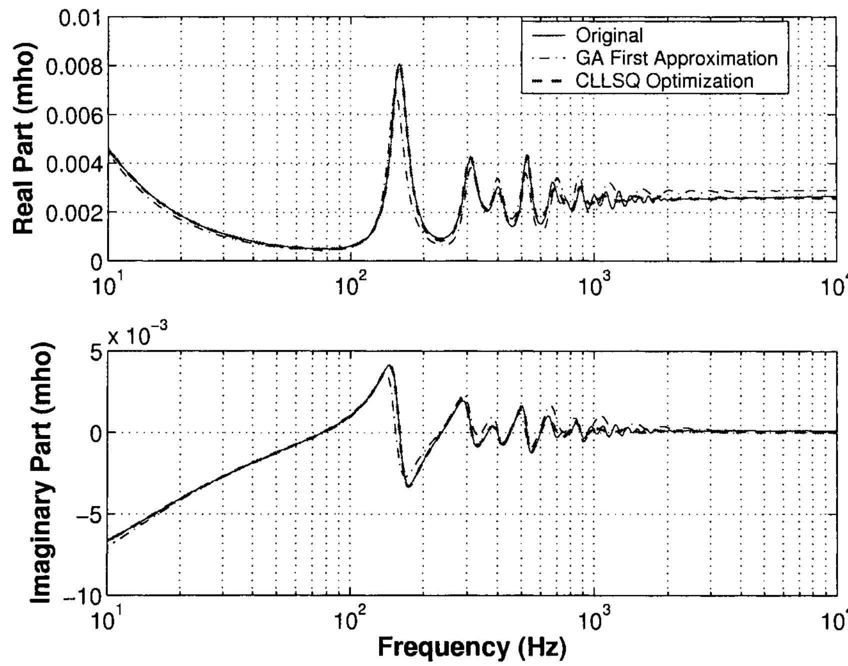


Figure 2.16: Example 1 input admittance

lation time analysis demonstrates computational burden reduction. In the next chapter, the Robust TLNE model is extended to active and multi-phase networks.

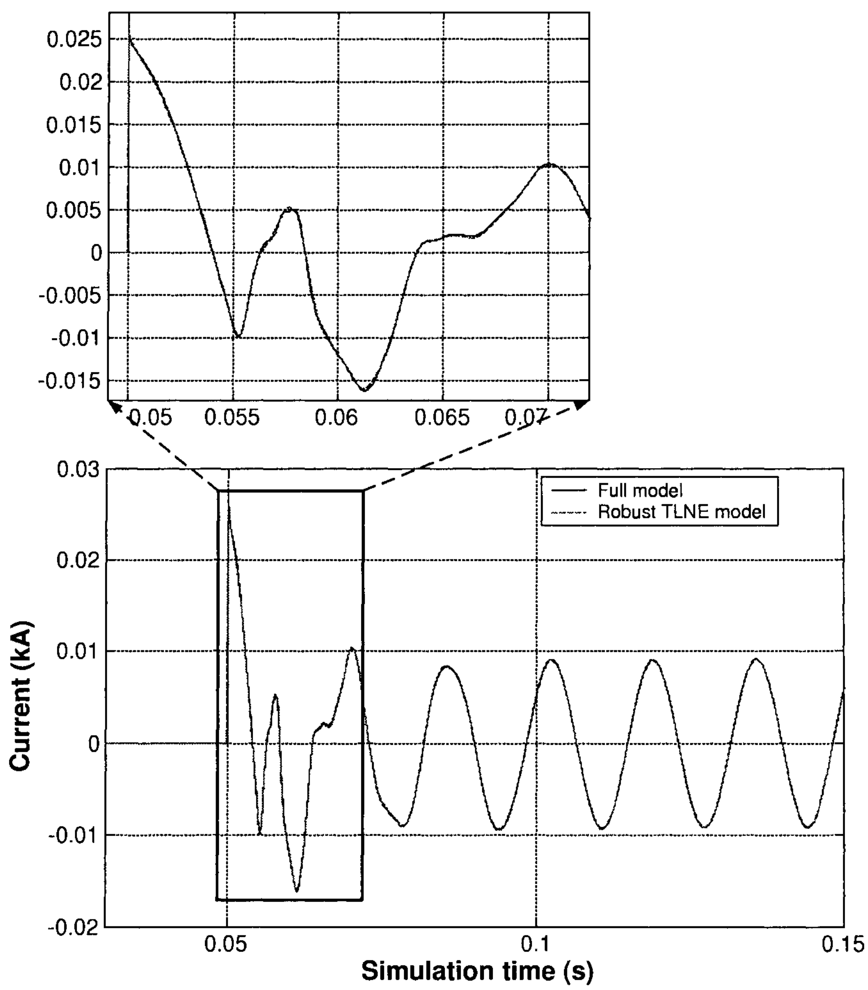


Figure 2.17: Example 1 transient simulation and comparison

3

Robust Two-Layer Network Equivalent for Active Networks

In the previous chapter, the Robust TLNE for passive networks is explained. However, realistic power systems, especially large power systems, include generators and other active elements. Thus, it is necessary to extend the Robust TLNE approach to active networks. Moreover, a three-phase multi-port example system with an active external system will be modeled by Robust TLNE.

3.1 External System with Active Elements

Since external system is LTI, the contribution of active elements in the external system to the transients is only limited to power frequency. This gives the idea that we only need to consider the external system in power frequency and construct Norton equivalent current sources for active elements at external system input ports, as shown in Fig. 3.1. The Norton equivalent current sources are found by either using an analytical method [21, 22], or by measuring short-circuit current.

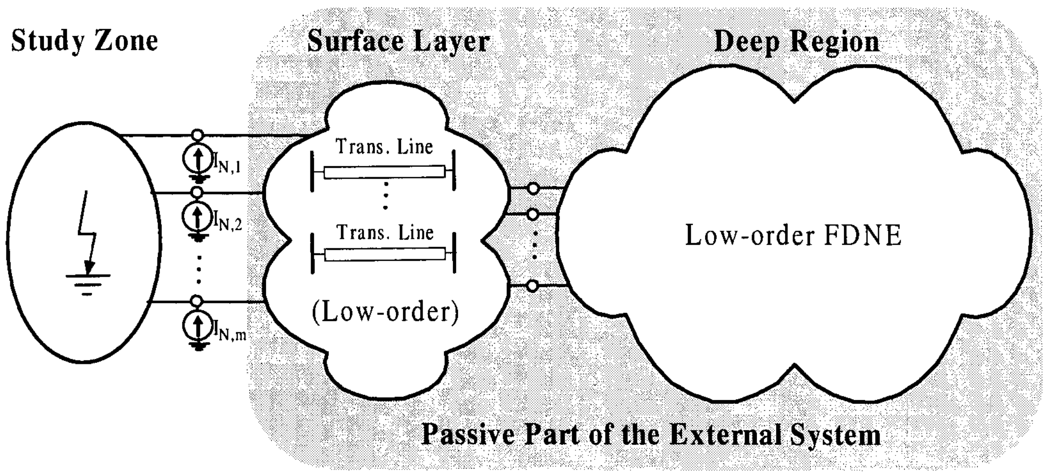


Figure 3.1: Robust TLNE model for external system with active elements

3.1.1 The Analytical Method

In power system transient studies, a generator can be treated as an ideal voltage source in series with an impedance. In an active external system, we denote the buses that have generators connected with subscript of $_G$, interfacing buses to the study zone with subscript of $_A$ and all other buses with subscript of $_B$. Then nodal equations are [21,22]

$$\begin{bmatrix} \mathbf{I}_A \\ \mathbf{I}_G \\ \mathbf{I}_B = 0 \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{AA} & \mathbf{Y}_{AG} & \mathbf{Y}_{AB} \\ \mathbf{Y}_{GA} & \mathbf{Y}_{GG} & \mathbf{Y}_{GB} \\ \mathbf{Y}_{BA} & \mathbf{Y}_{BG} & \mathbf{Y}_{BB} \end{bmatrix} \begin{bmatrix} \mathbf{V}_A \\ \mathbf{V}_G \\ \mathbf{V}_B \end{bmatrix} \tag{3.1}$$

The third row of (3.1) gives

$$\mathbf{V}_B = -\mathbf{Y}_{BB}^{-1}(\mathbf{Y}_{BA}\mathbf{V}_A + \mathbf{Y}_{BG}\mathbf{V}_G). \tag{3.2}$$

Substituting into first two rows of (3.1), we have

$$\begin{bmatrix} \mathbf{I}_A \\ \mathbf{I}_G \end{bmatrix} = \begin{bmatrix} \mathbf{Y}'_{AA} & \mathbf{Y}'_{AG} \\ \mathbf{Y}'_{GA} & \mathbf{Y}'_{GG} \end{bmatrix} \begin{bmatrix} \mathbf{V}_A \\ \mathbf{V}_G \end{bmatrix} \tag{3.3}$$

where

$$\begin{aligned} \mathbf{Y}'_{AA} &= \mathbf{Y}_{AA} - \mathbf{Y}_{AB}\mathbf{Y}_{BB}^{-1}\mathbf{Y}_{BA} \\ \mathbf{Y}'_{AG} &= \mathbf{Y}_{AG} - \mathbf{Y}_{AB}\mathbf{Y}_{BB}^{-1}\mathbf{Y}_{BG} \\ \mathbf{Y}'_{GA} &= \mathbf{Y}_{GA} - \mathbf{Y}_{GB}\mathbf{Y}_{BB}^{-1}\mathbf{Y}_{BA} \\ \mathbf{Y}'_{GG} &= \mathbf{Y}_{GG} - \mathbf{Y}_{AB}\mathbf{Y}_{BB}^{-1}\mathbf{Y}_{BG} \end{aligned} \tag{3.4}$$

The generator model is represented as

$$\mathbf{I}_G = \mathbf{Y}_G(\mathbf{E}_G - \mathbf{V}_G) \tag{3.5}$$

Substituting into (3.3), we obtain Norton equivalent

$$\mathbf{I}_A = \mathbf{Y}_N \mathbf{V}_A + \mathbf{I}_N \tag{3.6}$$

where

$$\begin{aligned} \mathbf{Y}_N &= \mathbf{Y}'_{AA} - \mathbf{Y}'_{AG}(\mathbf{Y}'_{GG} + \mathbf{Y}_G)^{-1}\mathbf{Y}'_{GA} \\ \mathbf{I}_N &= \mathbf{Y}'_{AG}(\mathbf{Y}'_{GG} + \mathbf{Y}_G)^{-1}\mathbf{Y}_G\mathbf{E}_G \end{aligned} \tag{3.7}$$

Thus, vector $\mathbf{I}_N$ is the Norton equivalent current sources at external system input ports; $\mathbf{Y}_N$ is the passive part of external system input admittance matrix at power frequency.

3.1.2 Measuring Short-Circuit Current

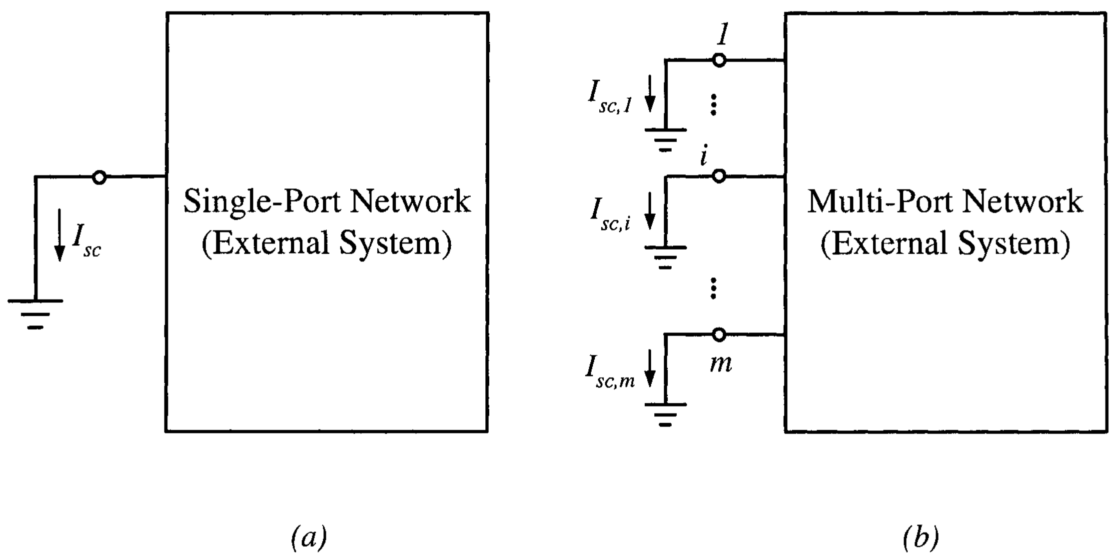


Figure 3.2: Obtaining Norton equivalent current sources for external system

Since the analytical method requires the reconstruction of external system admittance matrix, when the external system is very big, the procedure is error prone. Measuring short-circuit current is more reliable since the same circuits and data files as the ones for simulation in EMTP are used. In single-port external systems, the Norton equivalent current source is found by measuring short-circuit current at that port, shown in Fig. 3.2(a), where

$$I_N = I_{sc} \tag{3.8}$$

In multi-port external system case, shown in Fig. 3.2(b), the Norton equivalent current sources are obtained by measuring the short-circuit current at all ports at the same time.

In an m-port system, we have

$$\mathbf{I}_N = \mathbf{I}_{sc} \tag{3.9}$$

where $\mathbf{I}_N = [\, I_{N,1} \quad I_{N,2} \quad \cdots \quad I_{N,m} \,]^T$, $\mathbf{I}_{sc} = [\, I_{sc,1} \quad I_{sc,2} \quad \cdots \quad I_{sc,m} \,]^T$; $\mathbf{I}_N$ and $\mathbf{I}_{sc}$ correspond to the quantities in Figures 3.1 and 3.2(b), respectively.

The passive part of the external system is obtained by traditional method of eliminating all active elements, *i.e.*, voltage sources are considered short-circuit and current sources are considered open-circuit. The Robust TLNE model discussed in the previous chapter is applied to the passive part. Thus the procedures for building the Robust TLNE model for external system of active networks are obtained. Now, based on above explanation, it is necessary to illustrate the procedures to obtain Robust TLNE model, as shown in Fig. 3.3.

3.2 Robust TLNE for Three-Phase Multi-Port Systems

In three-phase systems, greatly attributed to transmission lines, mutual couplings exist. Therefore, some specific issues associated with three-phase systems requires to be discussed. In transient analysis, a three-phase system is commonly decoupled into three separate systems. Such decoupling is done by transformation between phase domain and modal domain. Since the external system is balanced and transposed, Clarke's transformation is more appropriate [45]. In addition, an alternative fitting routine for transmission line parameters is employed in multi-port systems with a strong constraint on passivity.

3.2.1 Clarke's Transformation

In a balanced and transposed power system, by Clarke's transformation, a three-phase system is decoupled into three separate single-phase systems called *modes*. The transformation matrix

$$\mathbf{T} = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & \sqrt{2} & 0 \\ 1 & -1/\sqrt{2} & \sqrt{3}/\sqrt{2} \\ 1 & -1/\sqrt{2} & -\sqrt{3}/\sqrt{2} \end{bmatrix} \tag{3.10}$$

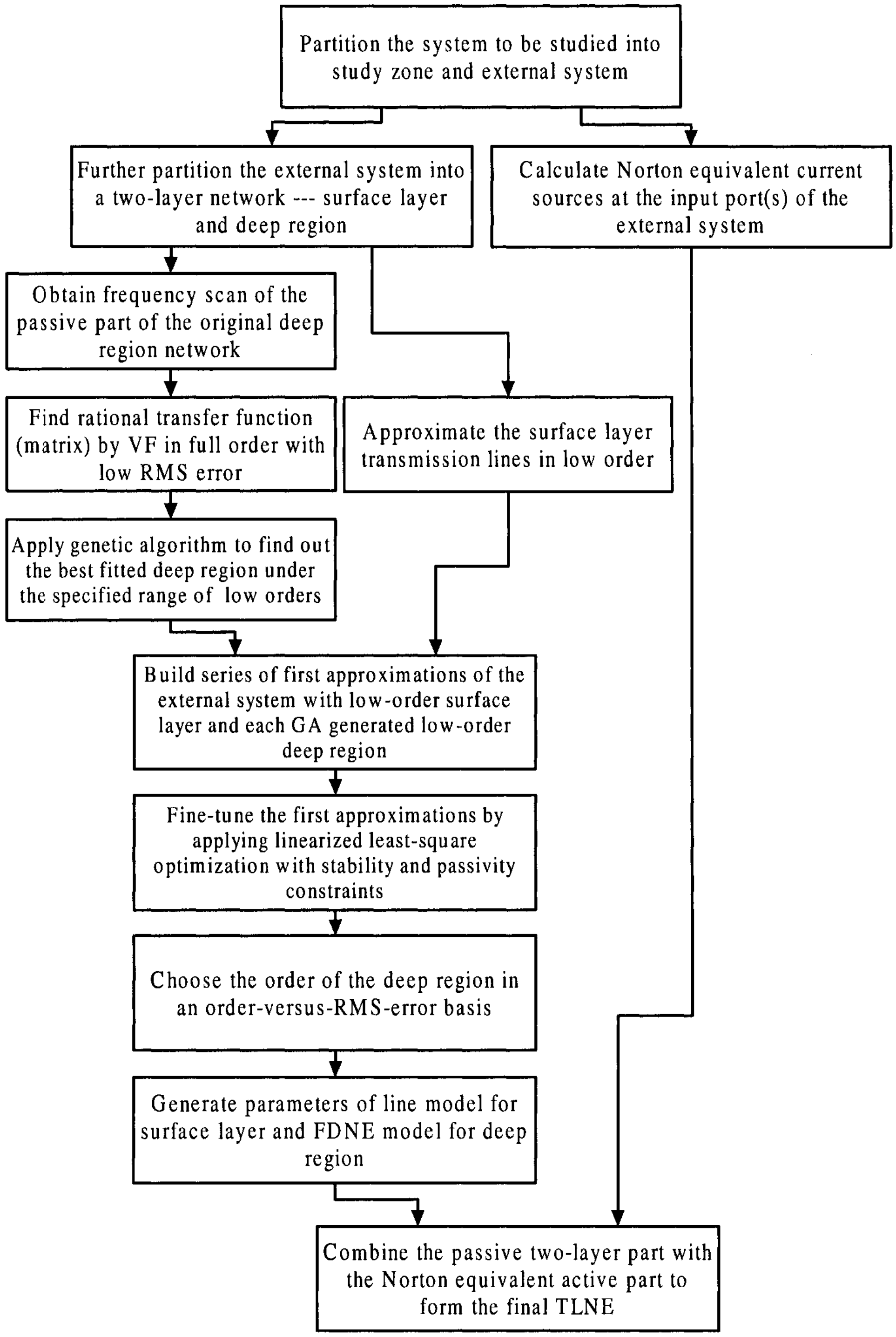


Figure 3.3: Flowchart for obtaining Robust TLNE model for generic external systems

defines Clarke's transformation [45] as

$$\mathbf{i}_{mode} = \mathbf{T}^{-1}\mathbf{i}_{phase} \tag{3.11a}$$

$$\mathbf{v}_{mode} = \mathbf{T}^{-1}\mathbf{v}_{phase} \tag{3.11b}$$

$$\mathbf{Y}_{mode} = \mathbf{T}^{-1}\mathbf{Y}_{phase}\mathbf{T} \tag{3.11c}$$

where

$$\mathbf{i}_{phase} = [\; i_A \quad i_B \quad i_C \;]^T \tag{3.12a}$$

$$\mathbf{v}_{phase} = [\; v_A \quad v_B \quad v_C \;]^T \tag{3.12b}$$

$$\mathbf{Y}_{phase} = \begin{bmatrix} Y_s & Y_m & Y_m \\ Y_m & Y_s & Y_m \\ Y_m & Y_m & Y_s \end{bmatrix} \tag{3.12c}$$

$$\mathbf{i}_{mode} = [\; i_0 \quad i_\alpha \quad i_\beta \;]^T \tag{3.13a}$$

$$\mathbf{v}_{mode} = [\; v_0 \quad v_\alpha \quad v_\beta \;]^T \tag{3.13b}$$

$$\mathbf{Y}_{mode} = \begin{bmatrix} Y_0 & 0 & 0 \\ 0 & Y_\alpha & 0 \\ 0 & 0 & Y_\beta \end{bmatrix} \tag{3.13c}$$

and

$$Y_0 = Y_s + 2Y_m, \; Y_\alpha = Y_\beta = Y_s - Y_m \tag{3.14}$$

T is orthogonal, which means

$$\mathbf{T}^{-1} = \mathbf{T}^T = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 1 & 1 \\ \sqrt{2} & -1/\sqrt{2} & -1/\sqrt{2} \\ 0 & \sqrt{3}/\sqrt{2} & -\sqrt{3}/\sqrt{2} \end{bmatrix} \tag{3.15}$$

Thus, a balanced m-port three-phase system defined by equation

$$\begin{bmatrix} \mathbf{i}_{P,1} \\ \mathbf{i}_{P,2} \\ \vdots \\ \mathbf{i}_{P,m} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{P,11} & \mathbf{Y}_{P,12} & \cdots & \mathbf{Y}_{P,1m} \\ \mathbf{Y}_{P,21} & \mathbf{Y}_{P,22} & \cdots & \mathbf{Y}_{P,2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Y}_{P,m1} & \mathbf{Y}_{P,m2} & \cdots & \mathbf{Y}_{P,mm} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{P,1} \\ \mathbf{v}_{P,2} \\ \vdots \\ \mathbf{v}_{P,m} \end{bmatrix} \tag{3.16}$$

is transformed to modal domain using Clarke's transformation as

$$\begin{bmatrix} \mathbf{i}_{M,1} \\ \mathbf{i}_{M,2} \\ \vdots \\ \mathbf{i}_{M,m} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{M,11} & \mathbf{Y}_{M,12} & \cdots & \mathbf{Y}_{M,1m} \\ \mathbf{Y}_{M,21} & \mathbf{Y}_{M,22} & \cdots & \mathbf{Y}_{M,2m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Y}_{M,m1} & \mathbf{Y}_{M,m2} & \cdots & \mathbf{Y}_{M,mm} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{M,1} \\ \mathbf{v}_{M,2} \\ \vdots \\ \mathbf{v}_{M,m} \end{bmatrix} \tag{3.17}$$

where subscript $_P$ stands for phase-domain quantities, $\mathbf{i}_{P,i}$, $\mathbf{v}_{P,i}$ $(1 \leq i \leq m)$ are phase-domain ABC current and voltage vectors, and $\mathbf{Y}_{P,ij}$ $(1 \leq i, j \leq m)$ are symmetrical admittance matrices of phase domain as (3.12c); subscript $_M$ denotes modal-domain quantities, $\mathbf{i}_{M,i}$, $\mathbf{v}_{M,i}$ $(1 \leq i \leq m)$ are modal-domain $0\alpha\beta$ current and voltage vectors, and $\mathbf{Y}_{M,ij}$ $(1 \leq i, j \leq m)$ are diagonal admittance matrices of modal domain as (3.13c). Relationships between modal-domain and phase-domain quantities are

$$\mathbf{i}_{M,i} = \mathbf{T}^{-1}\mathbf{i}_{P,i} \tag{3.18a}$$

$$\mathbf{v}_{M,i} = \mathbf{T}^{-1}\mathbf{v}_{P,i} \tag{3.18b}$$

$$\mathbf{Y}_{M,ij} = \mathbf{T}^{-1}\mathbf{Y}_{P,ij}\mathbf{T} \tag{3.18c}$$

After Clarke's transformation, there are only two separate networks required solving during simulation, since α network is the same as β network in modal domain due to balanced systems. The α and β modes are equal and denoted as *aerial modes* or *sky modes*; the 0/zero mode is also called *ground mode.*

During EMTP simulation of three-phase systems, system equations are solved in phase domain to obtain nodal voltages. Then history current terms are updated in modal domain and transformed back to phase domain.

3.2.2 Nonlinear Fitting Method for Transmission Line Parameters

Example 1 in Section 2.6 is a single-port network, in which positive-real criterion is not a strong constraint during the final fine-tuning via CLLSQ optimization. However, in multi-port systems, our experience shows that system margin associated with positive-real constraint is very limited, *i.e.*, parameter change is significantly confined by positive-real criterion. In such cases, although surface layer and deep region compensate each other, they may not be able to compensate larger deviations, which is mostly caused by low-order fitting of surface layer, since deep regions generated by GAs are very accurate. Thus, RMS-error% during fitting transmission line $Z_c(\omega)$ and $P(\omega)$ must be kept very low, *e.g.*, below 1%, with more accuracy stressed at lower frequencies and power frequency. Nonetheless, within this RMS-error% range, Bode's asymptotic fitting technique implemented by most of EMTP packages often produces high-order rational functions. In Robust TLNE to make transmission line models more suitable for real-time simulation, a nonlinear fitting technique due to Fernandes *et al.* [42, 43] is used. This method is an iterative nonlinear least-square optimization procedure based on Levenberg-Marquardt approach.

In multi-phase systems, after Clarke's transformation, commonly the aerial mode exhibits more resonant peaks in frequency response, whereas the frequency response of the ground mode appears to be smoother. Our experience also shows that after VF the system with smoother frequency response normally has more margin for optimization than the one with a lot of resonant peaks. Thus, during fitting for surface layer transmission line parameters, RMS-error% allowance for ground mode is set to less than 1% and for aerial mode is only 0.5% max.

3.3 Example 2

The 240kV system used here [23] is a modification of a standard system for transient stability studies [50]. Fig. 3.4(a) shows the system single-line diagram. The transient phenomena to be analyzed are balanced capacitor switching event at Bus 15 and three-phase to ground fault at Bus 16. In both cases, voltage and current are measured at Bus 16. System elements' parameters are shown in Appendix B.1. All transmission towers are assumed to have the same configurations and profiles. To simplify the original model construction, all parameters of generators and transformers were converted to 240kV base.

3.3.1 Generation of the Robust TLNE

The system is first partitioned into a study zone and an external system, as shown in Fig. 3.4(a). Fig. 3.4(b) shows the passive part of the external system. According to guidelines in Section 1.2 with engineering judgement, in the passive part of the external system, transmission line TL1, TL2 and TL3 are considered for the surface layer and the remaining system of a two-port network comprises the deep region.

Frequency scan (discussed in Section 2.1.2) of the original deep region network provided the phase-domain admittance matrix $\mathbf{Y}_{deep,P}(\omega)$. Clarke's transformation further decoupled $\mathbf{Y}_{deep,P}(\omega)$ to ground mode $\mathbf{Y}_{deep,0}(\omega)$ and aerial mode $\mathbf{Y}_{deep,\alpha}(\omega)$ in modal domain. Shown in Figures 3.6 through 3.8, due to ground return, the ground mode tends to be smoother. Full-order VF generates a 60th-order rational function matrix

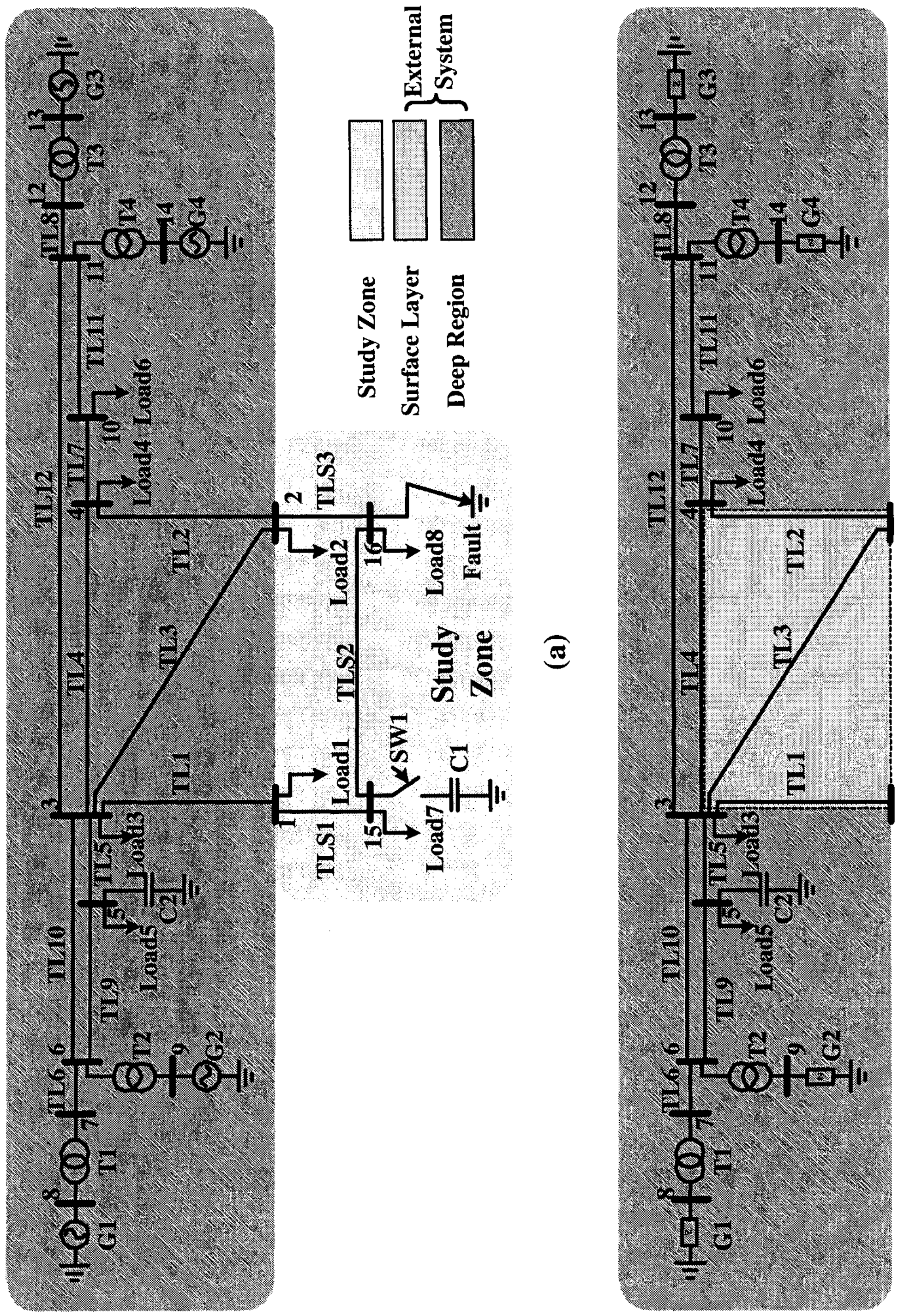


Figure 3.4: Example 2 system. (a) Example 2 system diagram and its partitioning. (b) Example 2 passive part of the external system.

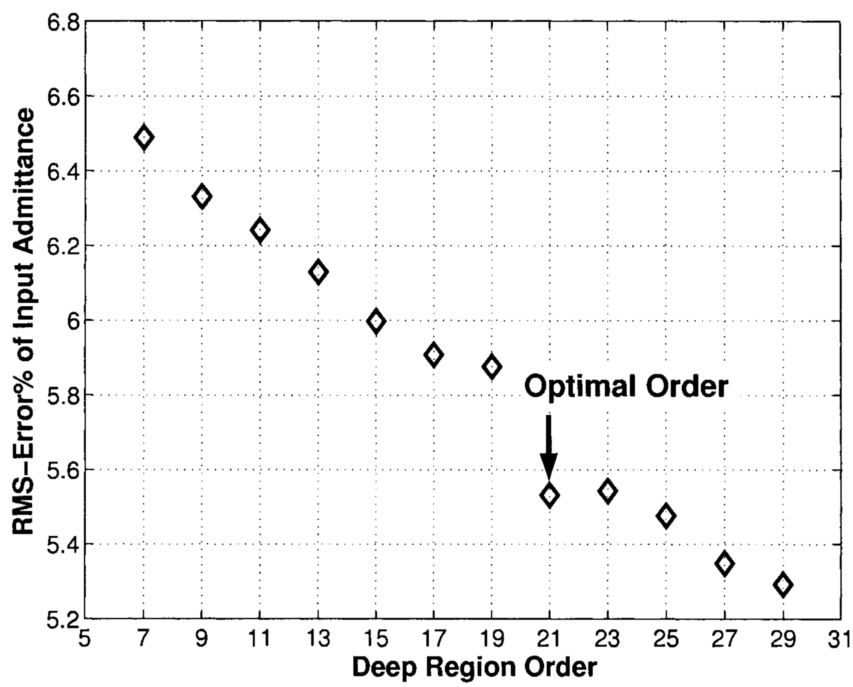


Figure 3.5: Example 2 aerial mode RMS-error% of input admittance *v.s.* deep region order

with 0.43% RMS-error. On the contrary, the aerial mode exhibits a lot of resonant peaks, as shown in Figures 3.12 through 3.14. Fitting such frequency response, VF generates a 220th-order rational function matrix of 4 real poles and 108 complex conjugate pole pairs with 4.38% RMS-error. Following the rules of Robust TLNE, the 4 partial fractions with real poles are selected for deep region and all partial fractions with complex pairs are to be processed by GAs.

Due to the multi-port nature and passivity constraints of the system, in fitting the surface layer parameters $Z_c(\omega)$ and $P(\omega)$, non-linear fitting technique [42, 43] is used. Since the transients to be analyzed are balanced, only aerial mode is considered for both surface layer transmission lines and deep region networks. The RMS-error% of input admittance *v.s.* deep region order is shown in Fig. 3.5, from which, order 21 was found to be the optimal order for the deep region with low RMS-error of 5.533%. Figures 3.12 through 3.14 show the deep region frequency response generated by GAs and after CLLSQ optimization. Figures 3.9 through 3.11 show the input admittances of ex-

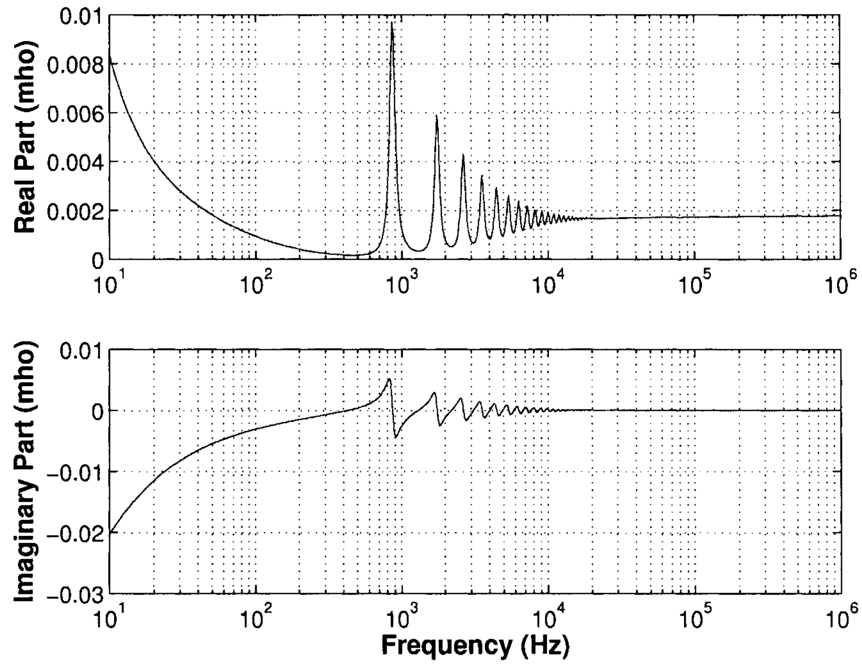


Figure 3.6: Example 2 ground mode input admittance $\mathbf{Y}_{input,0,11}$

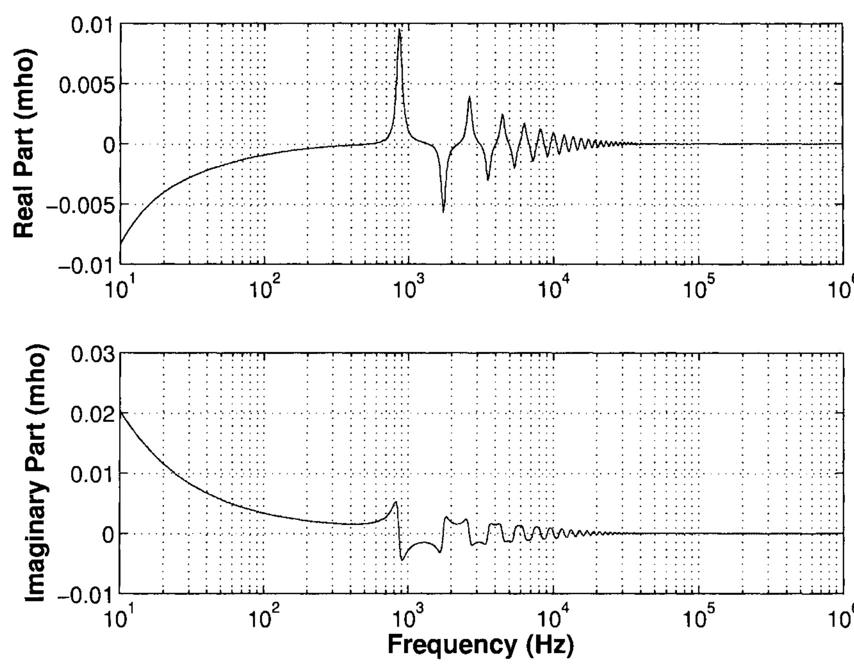


Figure 3.7: Example 2 ground mode input admittance $\mathbf{Y}_{input,0,12}$

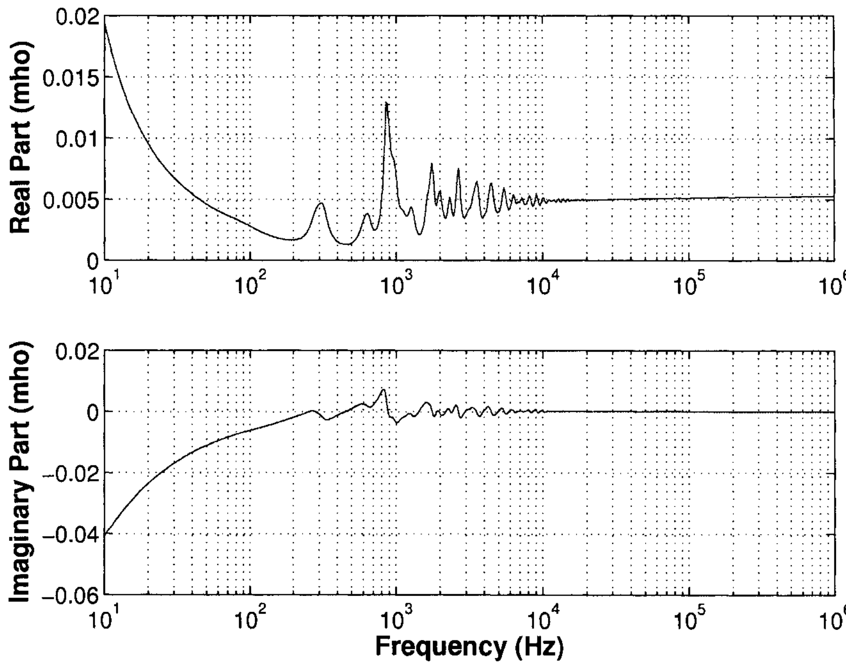


Figure 3.8: Example 2 ground mode input admittance $\mathbf{Y}_{input,0,22}$

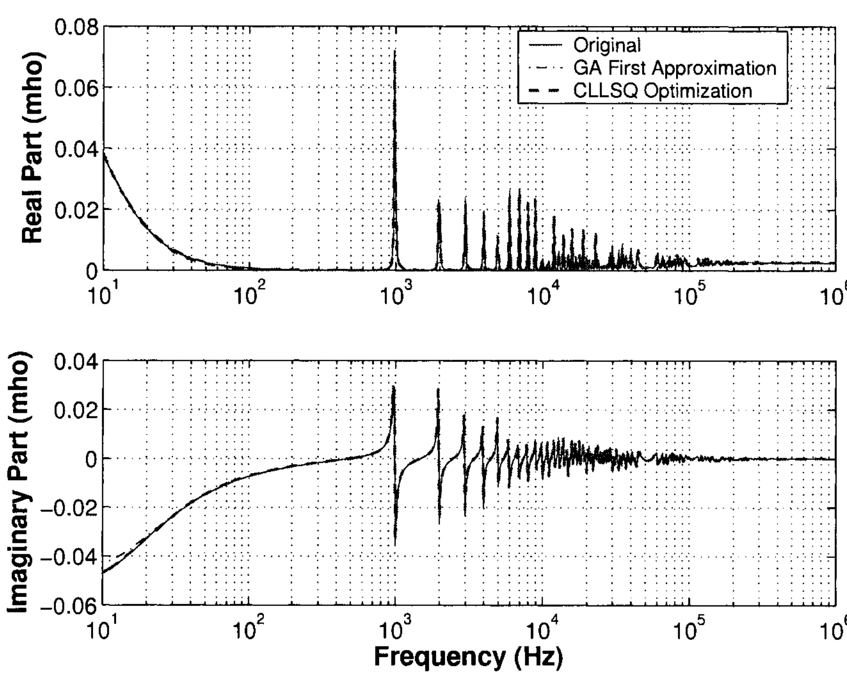


Figure 3.9: Example 2 aerial mode input admittance $\mathbf{Y}_{input,\alpha,11}$

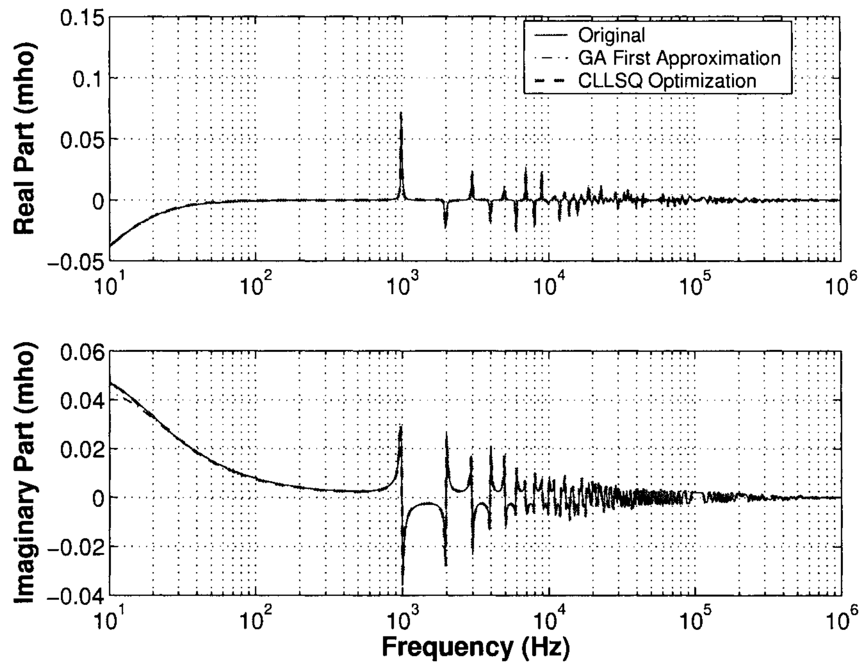


Figure 3.10: Example 2 aerial mode input admittance $\mathbf{Y}_{input,\alpha,12}$

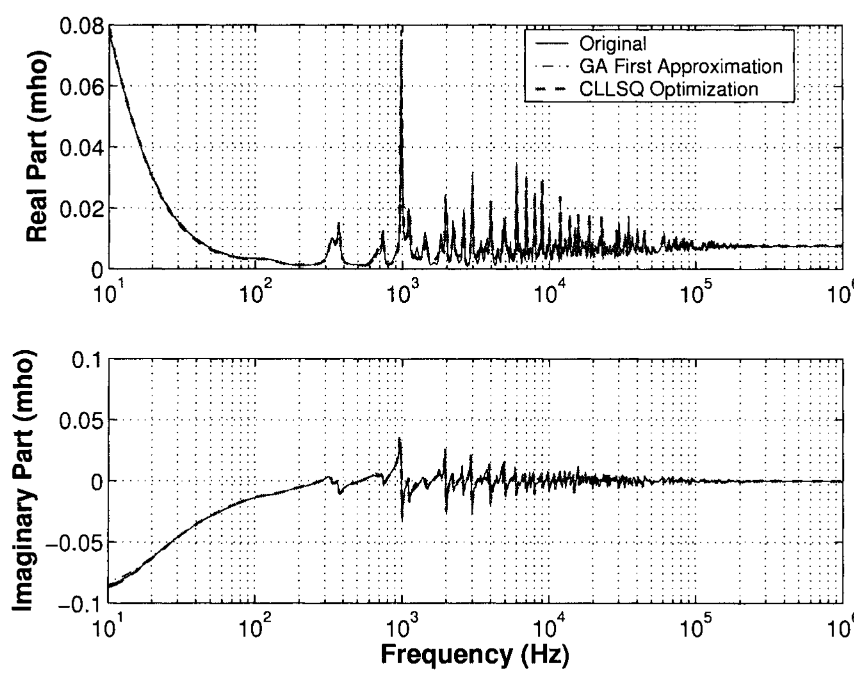


Figure 3.11: Example 2 aerial mode input admittance $\mathbf{Y}_{input,\alpha,22}$

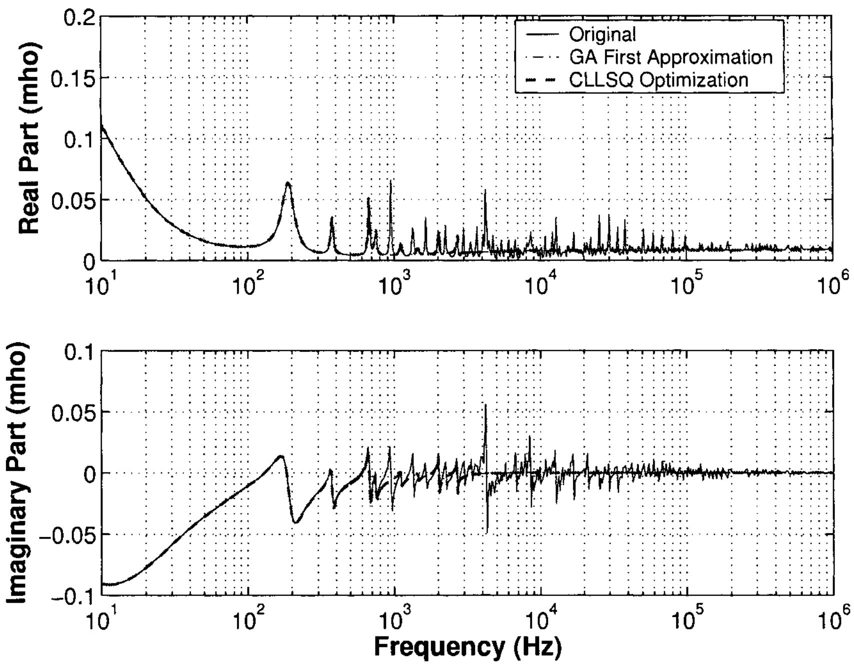


Figure 3.12: Example 2 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,11}$

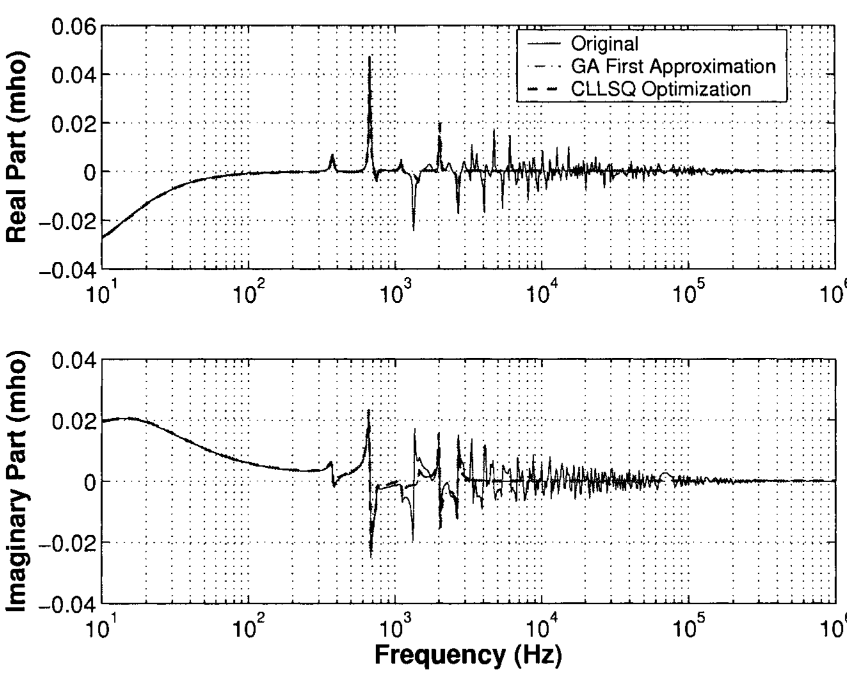


Figure 3.13: Example 2 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,12}$

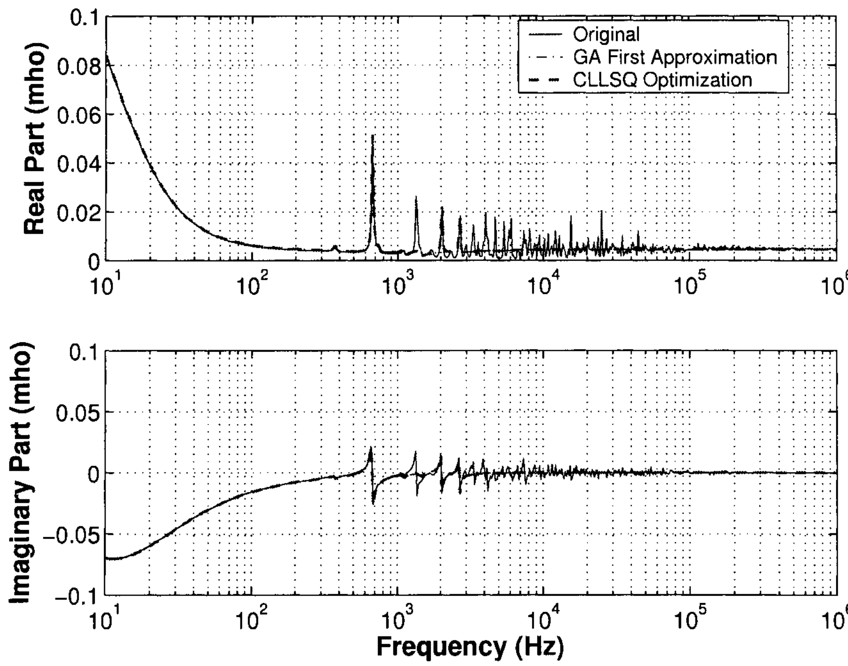


Figure 3.14: Example 2 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,22}$

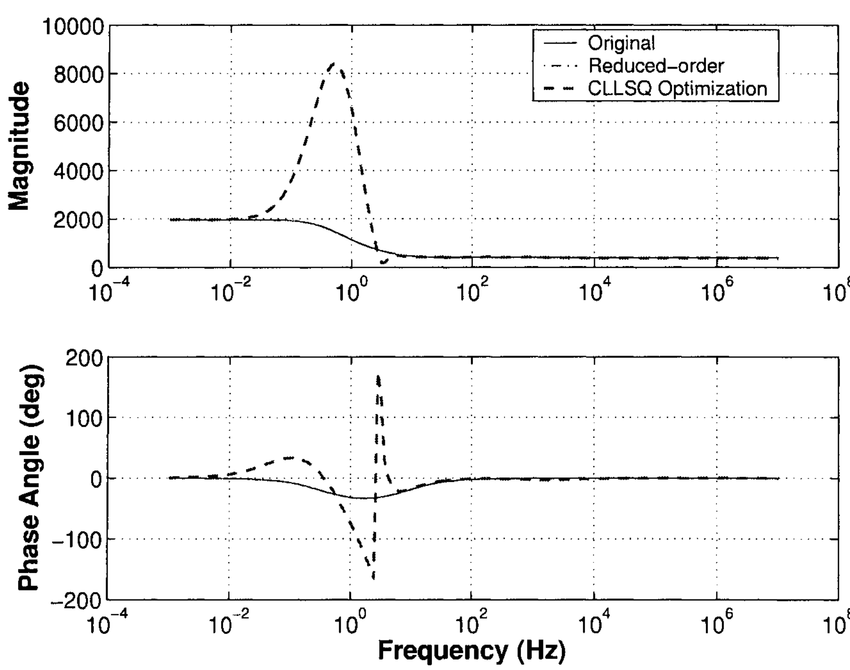


Figure 3.15: Example 2 TL1 aerial mode characteristic impedance

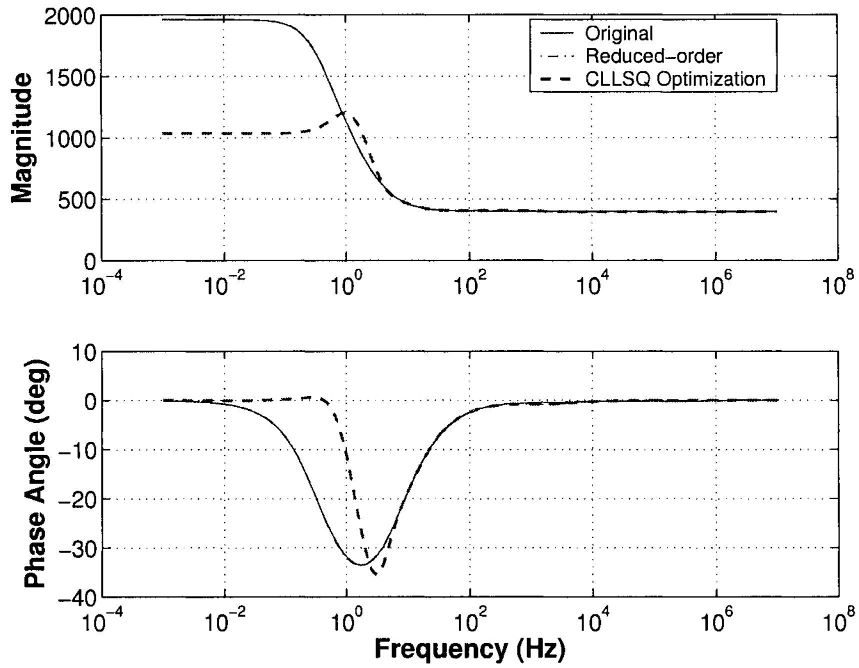


Figure 3.16: Example 2 TL2 aerial mode characteristic impedance

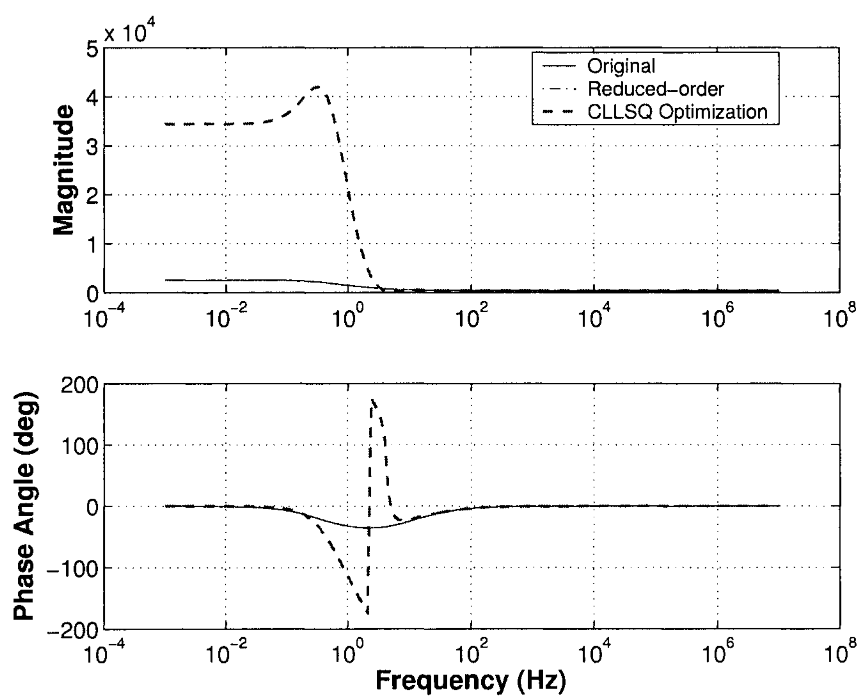


Figure 3.17: Example 2 TL3 aerial mode characteristic impedance

ternal system. It can be observed that the first approximations of input admittances are very close to original. Frequency responses of both the external system and deep region after CLLSQ optimization overlap those generated from GAs. It is shown that due to their relative insensitivity to the input admittance, some pronounced resonant peaks in deep region are not chosen by GAs. Figures 3.15 through 3.17 show characteristic impedance of surface layer transmissions TL1, TL2 and TL3 respectively. Further CLLSQ optimization mostly enhances the accuracy in the low frequency range, especially at DC, where the maximum RMS-error% is only 2.4%. Aerial mode transmission line parameters in surface layer are changed to compensate deviations in input admittance. Norton equivalent current sources for the external system are obtained through measuring short circuit current (Section 3.1.2) at input terminals. The phase-A phasors at each port are 1.648∠-86.21°kA and 2.284∠-93.66°kA, respectively. Converting to modal domain, the Norton equivalent current source phasor vectors $I_{eq,0\alpha\beta}$ are

$$\begin{bmatrix} 0\angle 0^\circ & 2.018\angle\text{-}86.21^\circ & 2.018\angle\text{-}176.21^\circ \end{bmatrix}^T \text{kA}$$

and

$$\begin{bmatrix} 0\angle 0^\circ & 2.797\angle\text{-}93.66^\circ & 2.797\angle\text{-}183.66^\circ \end{bmatrix}^T \text{kA}$$

at each port, respectively.

3.3.2 Transient Simulations

Transient events	Total time	Full model	Robust TLNE	160th FDNE
C1 switching	0.15s	1.034s	0.082s	1.024s
Balanced fault	0.20s	1.072s	0.117s	1.064s

Table 3.1: Example 2 computational time comparison at time step size 20μs

Fig. 3.18 shows phase-A voltage and current transients at Bus 15 where capacitor C1 is switched at 0.05s. Three-phase transients of this event are shown Fig. 3.19. Figures 3.20 and 3.21 show phase-A and three-phase fault current and voltage transients also at Bus 15, when balanced three-phase to ground fault is induced at Bus 16 with a 2Ω fault resistance per phase. The fault occurs at 0.05s. All transients are verified via ATP with time step size $20\mu s$. Detailed agreement between the full model of the system and the the robust TLNE model is observed. Computational time is a major saving in robust TLNE. Table 3.1 shows computational time comparison among full model, Robust TLNE model and FDNE model on a Pentium IV 1.6GHz computer. The FDNE model with 160th order has 6.859% RMS-error (higher than the TLNE model) and does not demonstrate great savings on computational time due to its high order. As seen from Table 3.1, the Robust TLNE model is 9 to 12 times faster than the full model, which makes it highly attractive for real-time digital simulation. Application of the existing method [21,22] to obtain TLNE of the same order produces 10.53% RMS-error, which is higher than the new approach. The ATP data files for the full model and Robust TLNE model are shown in Appendix B.3 and B.4, respectively.

3.4 Summary

In this chapter, the Robust TLNE model is extended to active networks and multi-phase multi-port systems. Procedures to obtain Norton equivalent current sources and interaction between phase domain and modal domain via Clarke's transformation are explained in detail. An accurate alternative technique in fitting transmission line parameters by nonlinear Levenberg-Marquardt method is also discussed. A three-phase multi-port example is modeled by the Robust TLNE to further verify the validity and stability of proposed approach.

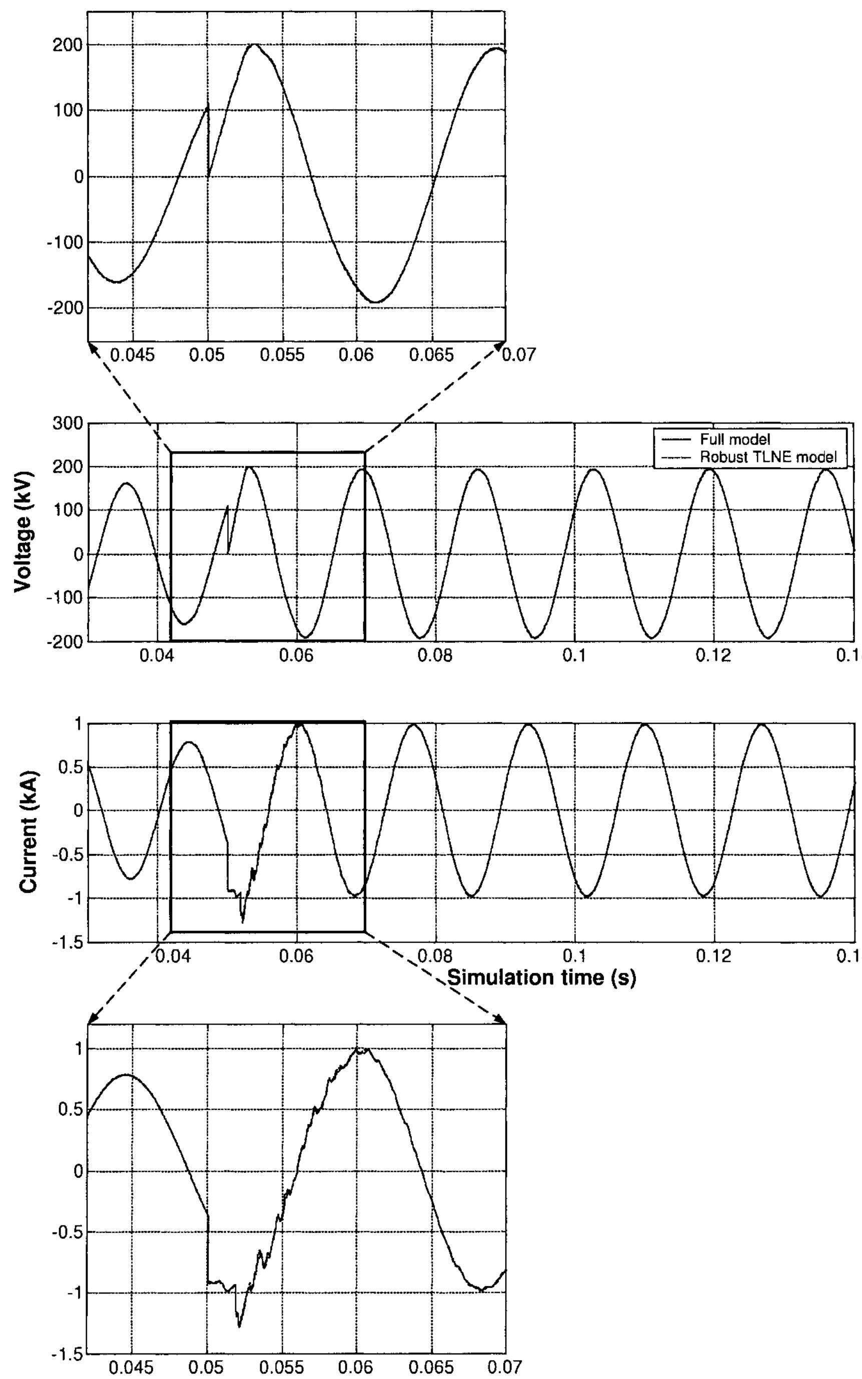


Figure 3.18: Example 2 $C1$ switching transient simulation and comparison (phase A)

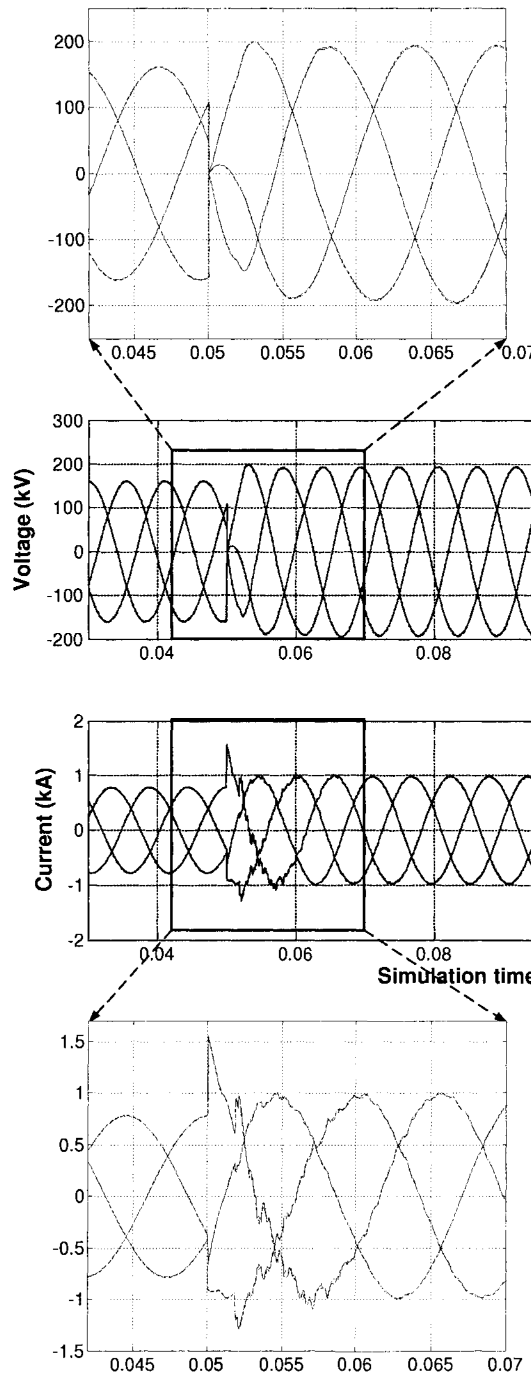


Figure 3.19: Example 2 $C1$ switching transient simulation and comparison (three phase)

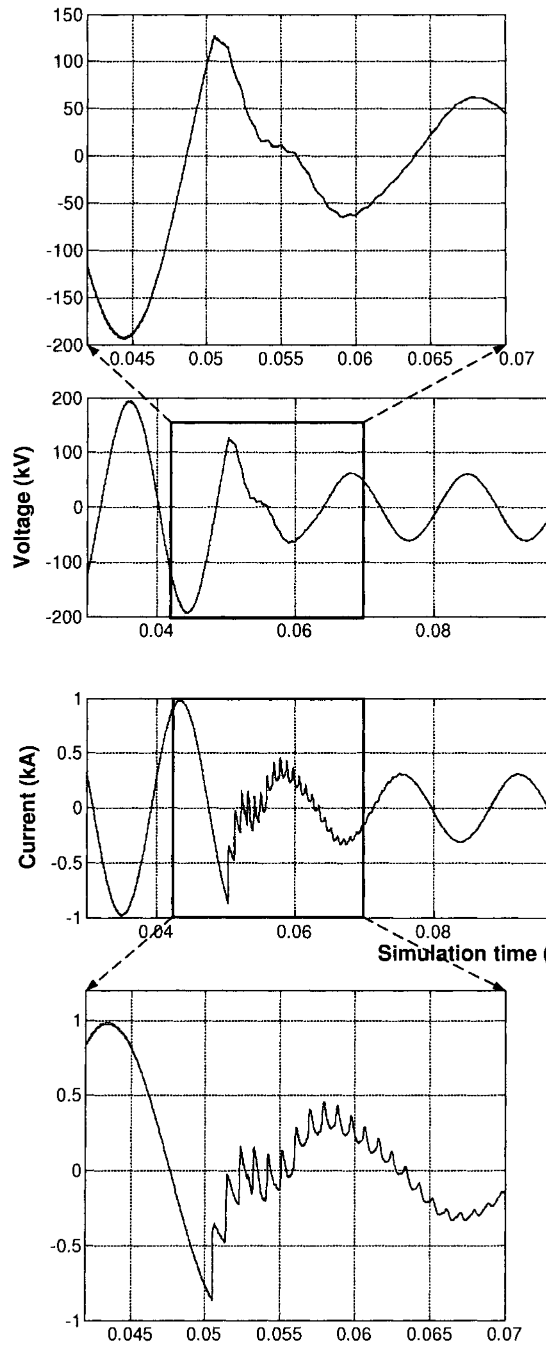


Figure 3.20: Example 2 three-phase to ground fault transient simulation and comparison (phase A)

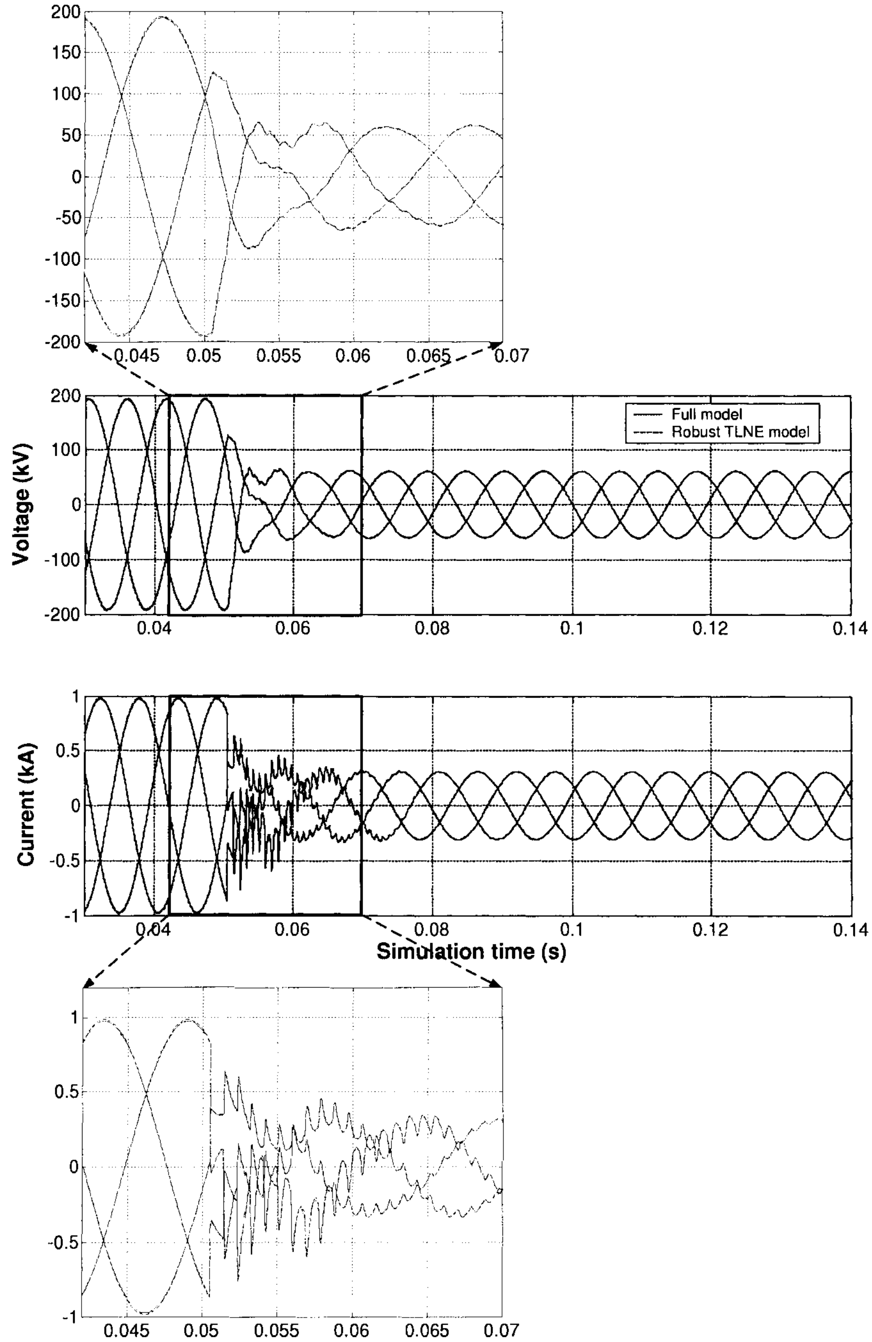


Figure 3.21: Example 2 three-phase to ground fault transient simulation and comparison (three phase)

4

Case Study — the Alberta Interconnected Electric System

With procedures for modeling Robust TLNE expanded and two examples explained in previous chapters, the accuracy and computational efficiency of the Robust TLNE model make it well-suited for real-time simulation of large power systems. In this chapter, a realistic large scale power system — the Alberta Interconnected Electric System (AIES) is to be modeled by a Robust TLNE at Bus 524 Genesee. Balanced three-phase to ground fault transients are to be examined and compared between the original full model and Robust TLNE model.

4.1 The Alberta Interconnected Electric System

The AIES, shown in Fig. 4.1, is a trans-province transmission and generation network with over 8,000MW generation capacities. As of 2003, it consists of over 17,000km of transmission lines and more than 400 substations ranging in voltage level from 69kV to 500kV. Two inter-province connections, which are HVDC transmission to SaskPower (Saskatchewan Province) and 500kV transmission lines to BC Hydro (British Columbia Province), connect AIES to the North American electric power grid. The AIES keeps growing along with Alberta's economy.

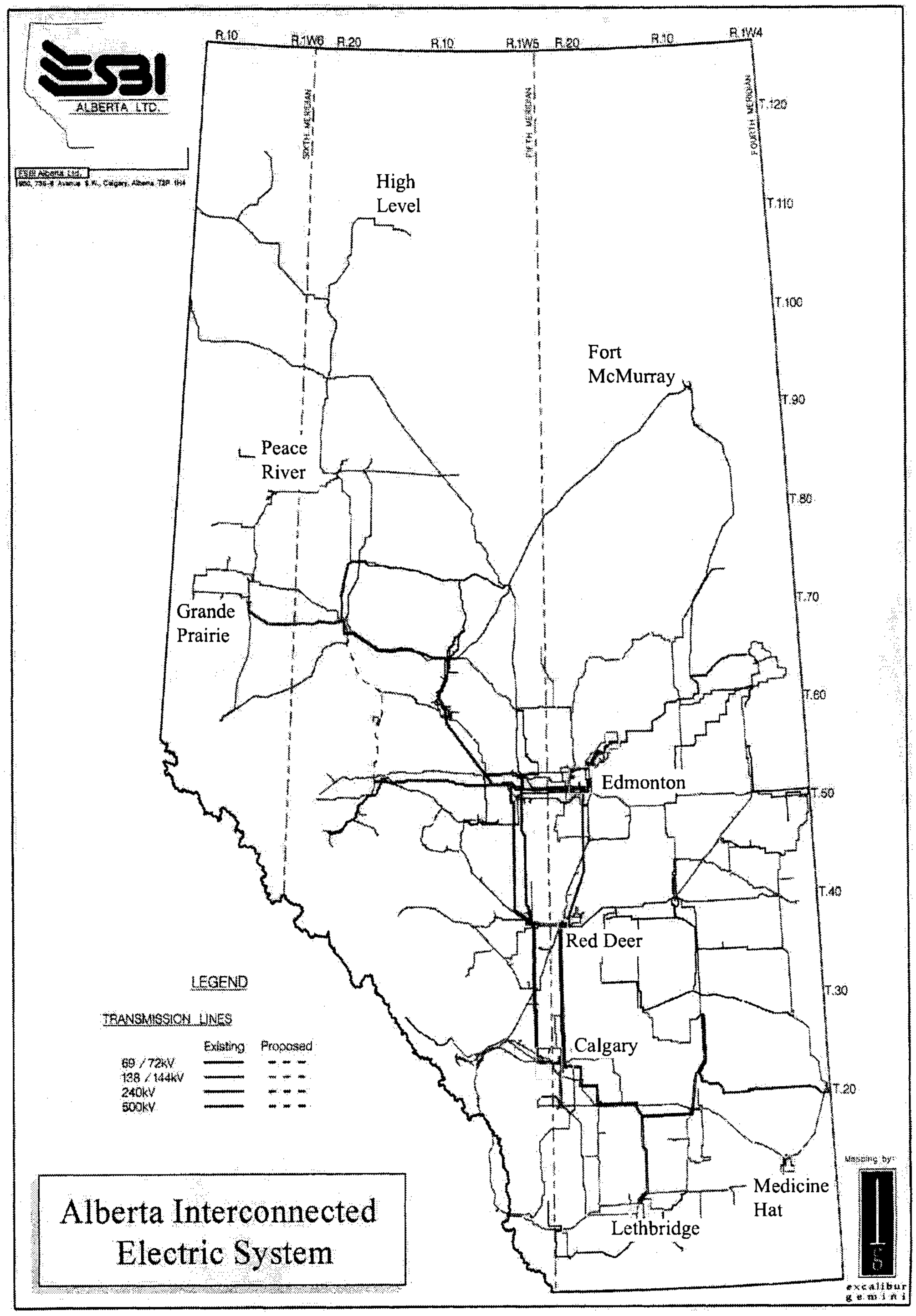


Figure 4.1: Map of AIES (Courtesy of AESO)

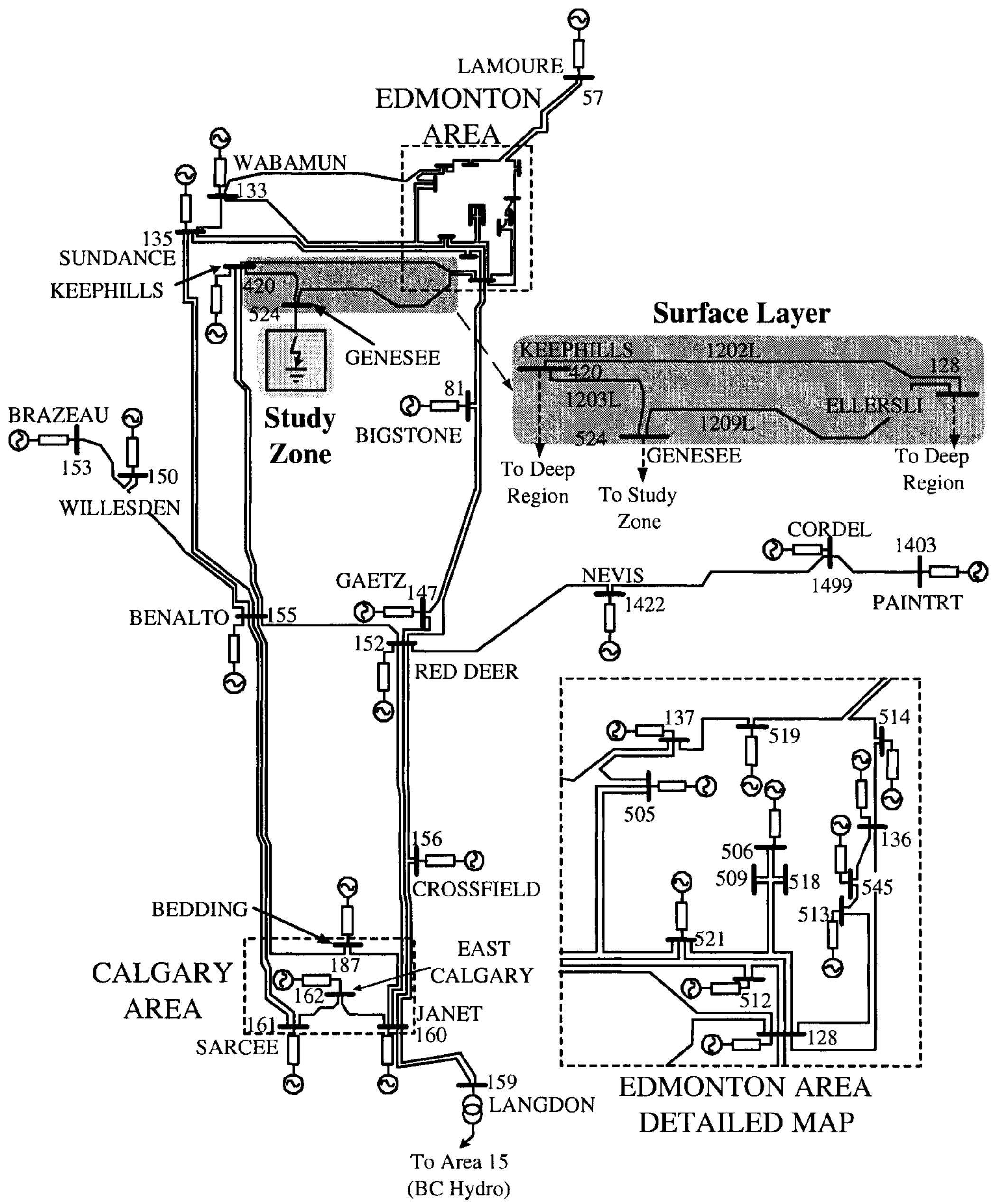


Figure 4.2: AIES Area 50 Backbone Single-Line Diagram

The system to be studied is 2003 Spring medium load operational case. In PSS/E, this case study of AIES has over 1600 buses with unique names, which are classified into approximately fifty numbered areas. Each area is also designated a name, *e.g.*, "Backbone" referring to Area 50, "WSCC" referring to Area 15, "Edmonton" referring to Area 60, "Calgary" referring to Area 6, *etc.*. SaskPower and BC Hydro connections are modeled as Thevenin equivalents. The Area 50 Backbone is an essential part of AIES, most buses of which are 240kV voltage level, as shown in Fig. 4.2. Mainly, the Area 50 includes one large ring network, which is from Edmonton Area to Calgary Area.

The transients to be analyzed are at Bus 524 Genesee, which belongs to Area 50 Backbone. It connects a number of generation stations together to the power grid of AIES. Due to the excessive complexities in modeling of AIES, network reduction is carried out while retaining the model's high accuracy.

4.2 Transmission Administrator System Model of AIES

The Transmission Administrator System Model or TASMo of AIES, is a model of all elements and facilities, that affect electricity flows through the Alberta Interconnected grid. Realized in Microsoft® Access, the TASMo database[1] provides detailed information of the following system elements and facilities:

- Transmission lines and line segments of each transmission line,
- Substations,
- Transformers,
- Machines including generators and motors,
- Shunts including capacitors and reactors,
- Loads.

A relational database is a collection of tables among which relationships exist. The TASMo database is constructed in the same way. All elements and facilities in AIES appear relationally in the corresponding tables. For example, table `TOPO_BUSSES` provides bus name, voltage level, ownership, PSS/E area code, facility name the bus resides as well as typical voltage and angle. The information provided in the TASMo database is suitable for both power flow and dynamic studies.

[1]The latest development of TASMo is called TASMo2 based on Oracle® Database.

4.3 Modeling and Network Reduction of AIES

In order to reduce the difficulties in modeling such a big power system in EMTP, the AIES is first equivalenced to only include Area 50 240kV Backbone, since transients occur in this area. In PSS/E, the appropriate activity for equivalencing AIES is `eeqv`, which allows to eliminate buses on the area and bus basis without affecting power flow. All buses in other areas except Area 50 are equivalenced. Appendix. C.2 shows detailed procedures to obtain equivalents for other areas excluding Area 50. Fig. 4.2 also illustrates the equivalents for outer areas with respect to Area 50. The equivalents obtained in PSS/E are all in per unit format for branches and MW/MVar format for loads and shunts. However, ATP requires values in Ω, mH, or μF. Thus, it is necessary to convert those equivalents into ATP readable style data. The following provides the methodology for the conversion:

- Equivalents appear in PSS/E with circuit number 99. Therefore, retained elements in Area 50 and equivalents can be distinguished.

- The base MVA for AIES is 100MVA. At 240kV voltage level, the base value for resistance, reactance or impedance is

  $$R_{base,240} = X_{base,240} = Z_{base,240} = 576\Omega$$

  and the base value for conductance, susceptance or admittance is

  $$G_{base,240} = B_{base,240} = Y_{base,240} = 0.001736\mho$$

  Then actual values for equivalent branches in Ω are obtained from PSS/E through above equations and are input into ATPDraw.

- Since loads are assumed to have constant real and reactive power, a constant current source is used for each load. The current source phasor is calculated by

  $$I\angle\theta_I = -\frac{\sqrt(2)\vec{S}^*}{240\sqrt{3}\vec{V}^*}$$

  where I is the current source magnitude, θ_I is the current source phase shift, $\vec{S}$ is the complex power absorbed by the load, $\vec{V}$ is the voltage phasor at the bus calculated in load flow, and superscript * denotes complex conjugate.

- Shunts are represented as RLC circuits and obtained from

$$Z_{shunt} = \frac{V^2}{\overrightarrow{S}*}$$

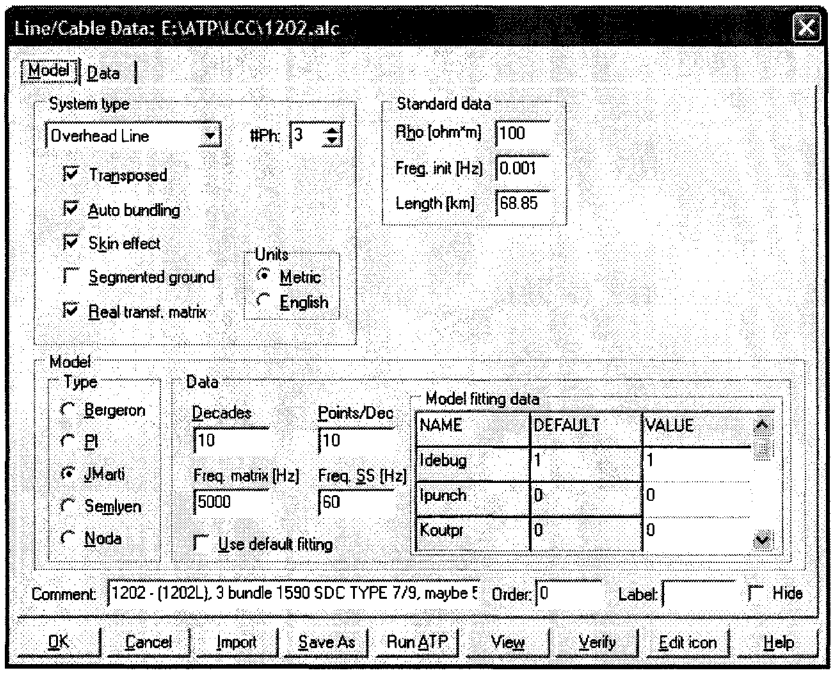
where V is base voltage level at the bus which is 240kV, and $\overrightarrow{S}$ is the complex power absorbed by the shunt.

- For convenience purpose, in ATP, the inductance is expressed as a reactance in Ω, while the capacitance is expressed as a susceptance in $\mu\mho$. Thus XOPT and COPT variables in ATP are both set to be 60.

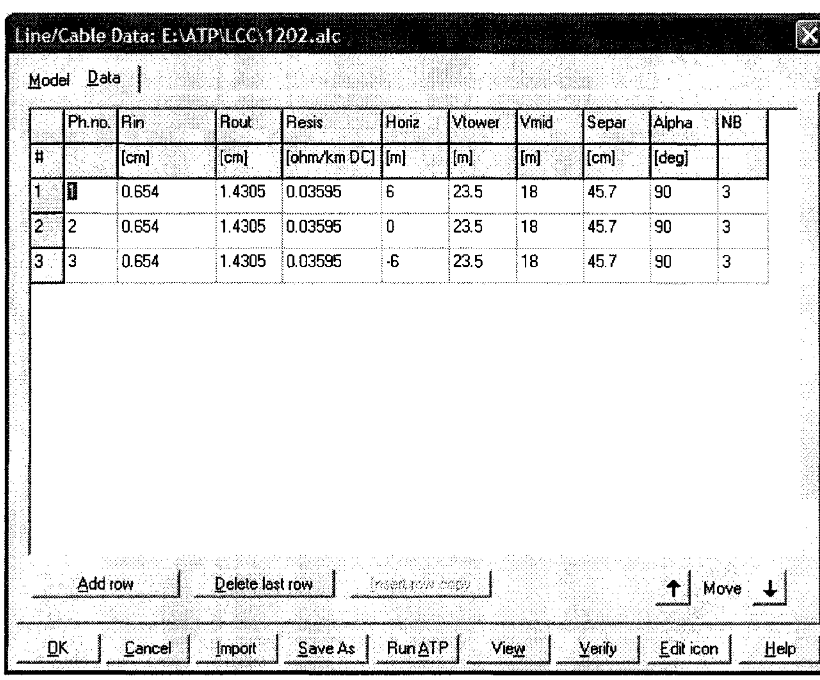
More accurate analysis of Area 50 requires all transmissions lines in the system modeled with frequency-dependence. This also enables the consideration of mutual couplings between some double circuit lines sharing the same right-of-way. In ATP, such frequency-dependent lines are modeled based on conductor geometry, which is not available in power flow data provided in PSS/E. Nonetheless, the AIES TASMo database provides adequate information for building a detailed model of frequency-dependent lines for Area 50 in ATP. In TASMo database

- Table STD_TOWERCOORDS provides tower code and phase coordinations.
- Conductor type and characteristics are given in table STD_CONDUCTORS.
- The names and connections of transmission line segments can be found in table TOPO_BRANCH_LINESEGS.
- Table TOPO_BRANCH_LINESEGS supplies detailed information for each line segment, such as conductor name, number of bundles, bundle spacing, tower code, height of the tower and length of the line segment.

Therefore, frequency-dependent lines in AIES Area 50 can be built in ATPDraw, which is shown in Fig. 4.3. Conductor characteristics can be found in the appendix of [51]. With above explanation, by combining the peripheral equivalent circuits generated from PSS/E with detailed model provided from TASMo, the AIES Area 50 Backbone model in ATP for electromagnetic transient analysis is obtained. Fig. 4.4 shows a part of actual EMTP model built in ATPDraw. The full diagram is shown in Appendix C.1. The corresponding ATP data file is shown in Appendix C.3.1, which is the base file for transient studies in the following sections.



(a) The line model information



(b) The line data information

Figure 4.3: Building a frequency-dependent transmission line from TASMo database in ATPDraw

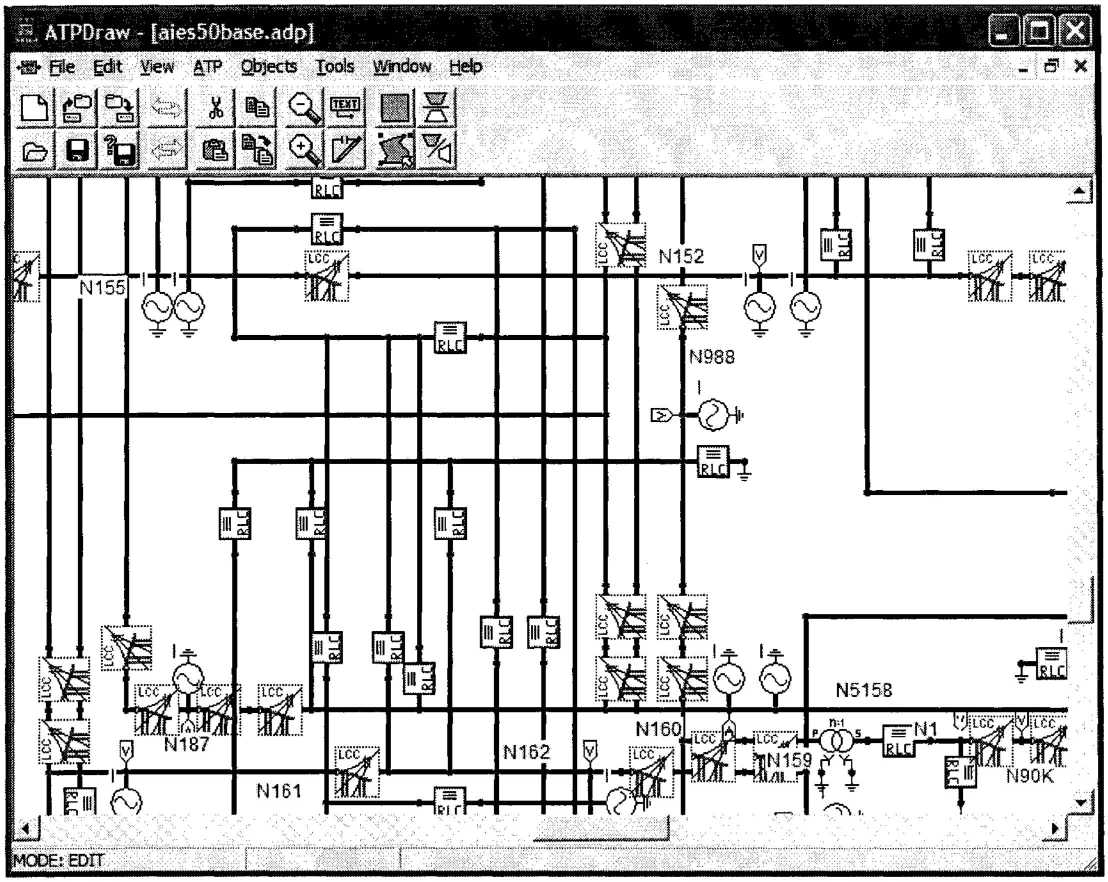


Figure 4.4: A part of AIES Area 50 diagram in ATPDraw

4.4 The Robust TLNE model for AIES

Since transients to be analyzed are around Bus 524 (Genesee), the generation stations are treated as a study zone and the rest of the network is considered as an external system. Furthermore, shown in Fig. 4.2, after applying TLNE model to the external system, transmission lines 1202L, 1203L, 1209L belong to the surface layer and the remaining of the external system forms the deep region. Shown in Fig. 4.6, due to ground return, the ground-mode input admittance $\mathbf{Y}_{input,0}(\omega)$ is very smooth. Full-order VF only generates a 36th-order FDNE model with 1.28% RMS-error. Therefore, the Robust TLNE model is only applied to aerial-mode admittance $\mathbf{Y}_{deep,\alpha}(\omega)$, since very high order rational function is required to achieve low RMS-error% in VF due to large amount of resonant peaks, as shown in Fig. 4.7.

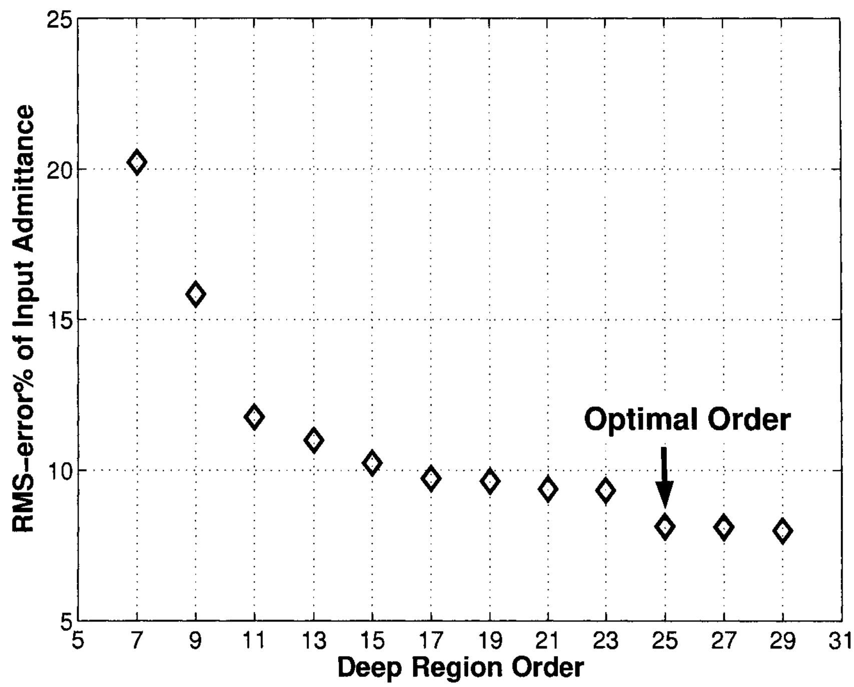


Figure 4.5: AIES Area 50 aerial mode RMS-error% of input admittance versus deep region order

The deep region is a two-port three-phase network. After removing all sources in the deep region, frequency scan of original deep region network gives phase-domain frequency response of deep region admittance matrix $\mathbf{Y}_{deep,P}(\omega)$. Clarke's transformation produces aerial mode $\mathbf{Y}_{deep,\alpha}(\omega)$ in modal domain. The aerial mode exhibits a lot of resonant peaks, as shown in Figures 4.8 through 4.10. In VF, the upper limit is 10kHz, since resonant peaks in higher frequency range are of less significant effects than those in lower frequency range, which is illustrated in both Example 1 and Example 2. With this method, VF generates a 240th-order rational function matrix of 2 real poles below 60Hz and 119 complex conjugate pole pairs with 1.93% RMS-error at discrete frequency points from 10Hz to 10kHz. In fitting of the whole frequency range from 10Hz to 1MHz, with the same level of RMS-error%, VF generates a 560th-order rational function matrix. Following the rules of Robust TLNE, the partial fractions with real poles are selected for deep region and all partial fractions of complex pairs are to be processed by GAs.

Due to the multi-port nature and complex frequency response of the system, in fitting the surface layer transmission line parameters $Z_c(\omega)$ and $P(\omega)$, non-linear fitting

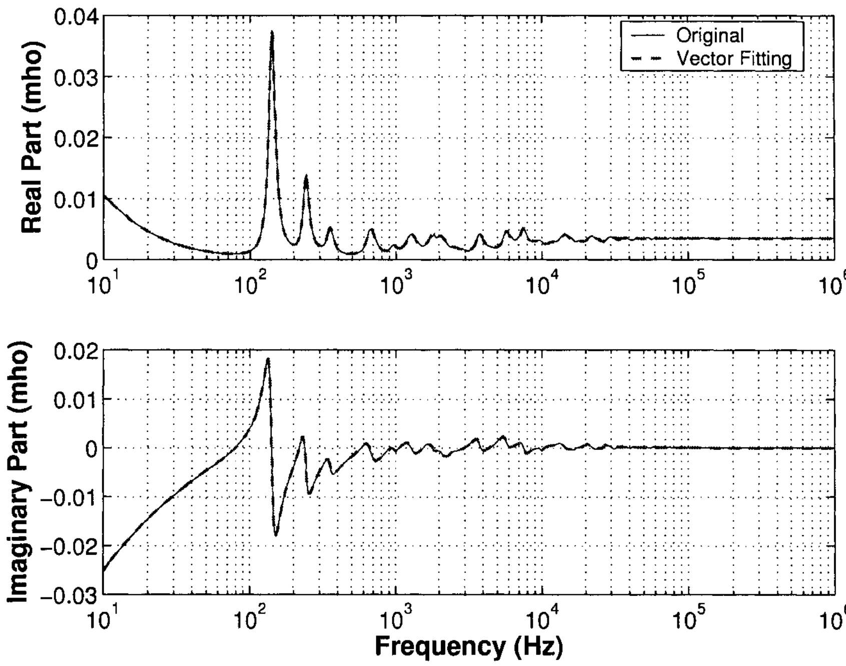


Figure 4.6: AIES Area 50 ground mode input admittance $\mathbf{Y}_{input,0}$

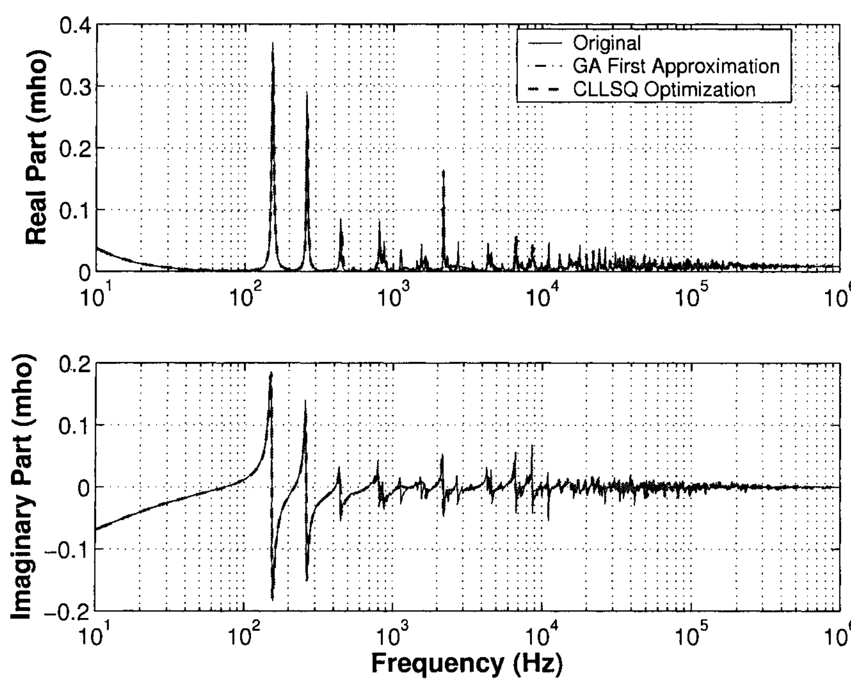


Figure 4.7: AIES Area 50 aerial mode input admittance $\mathbf{Y}_{input,\alpha}$

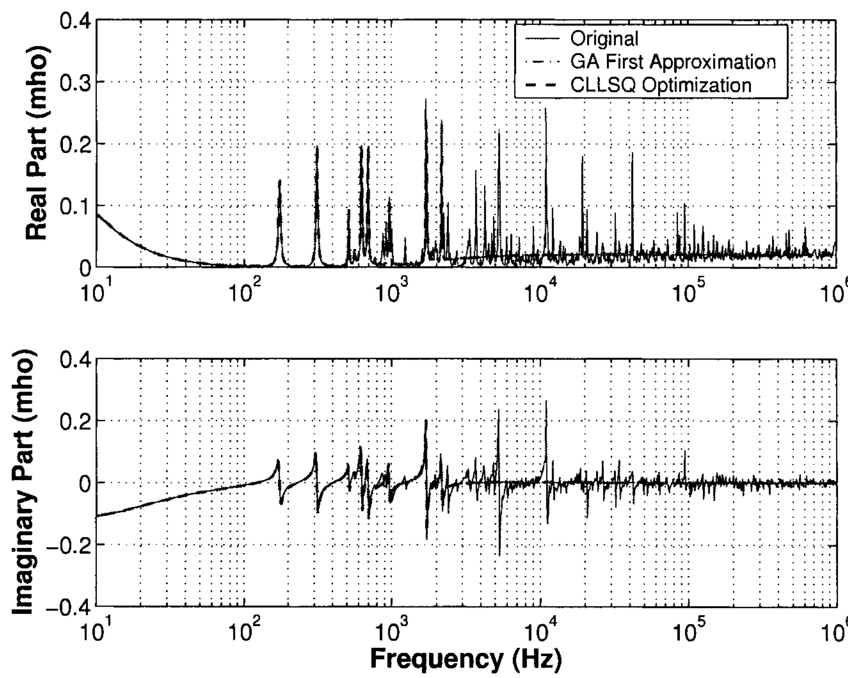


Figure 4.8: AIES Area 50 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,11}$

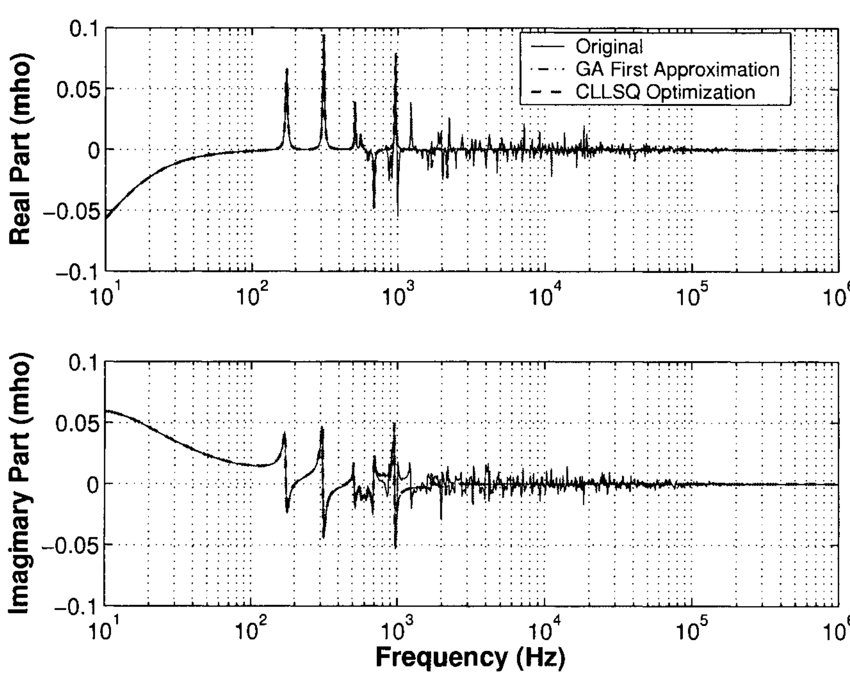


Figure 4.9: AIES Area 50 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,12}$

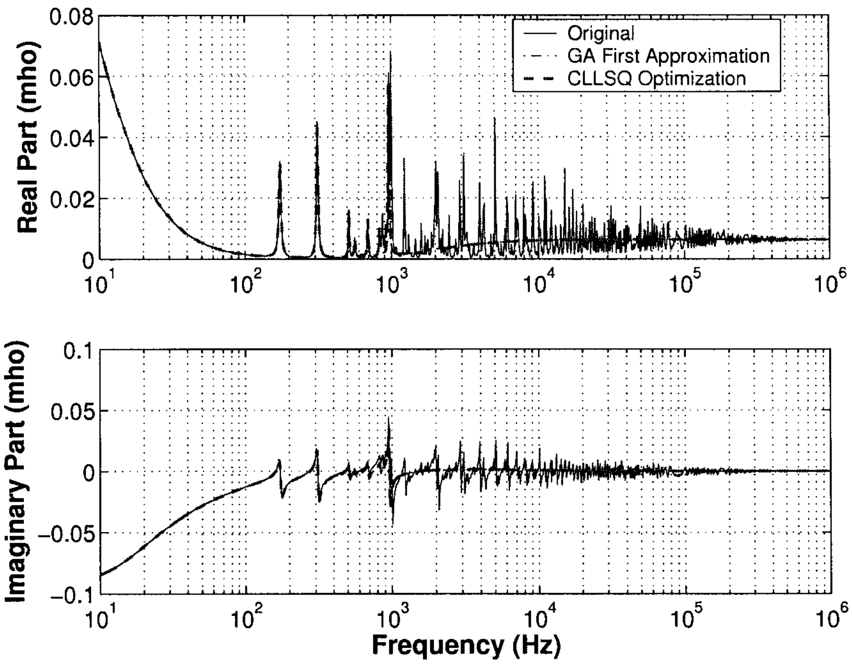


Figure 4.10: AIES Area 50 aerial mode deep region admittance $\mathbf{Y}_{deep,\alpha,22}$

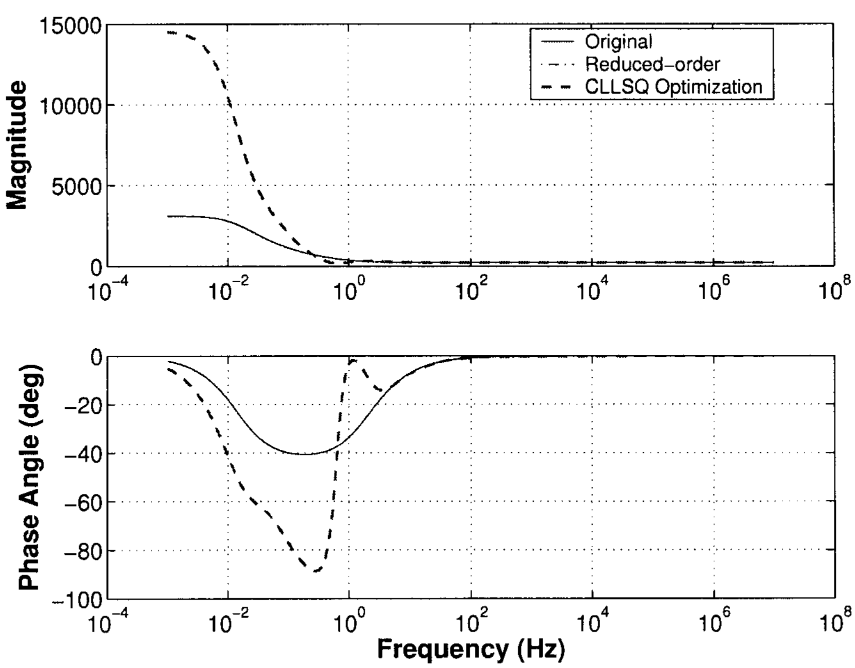


Figure 4.11: AIES Area 50 1202L aerial mode characteristic impedance

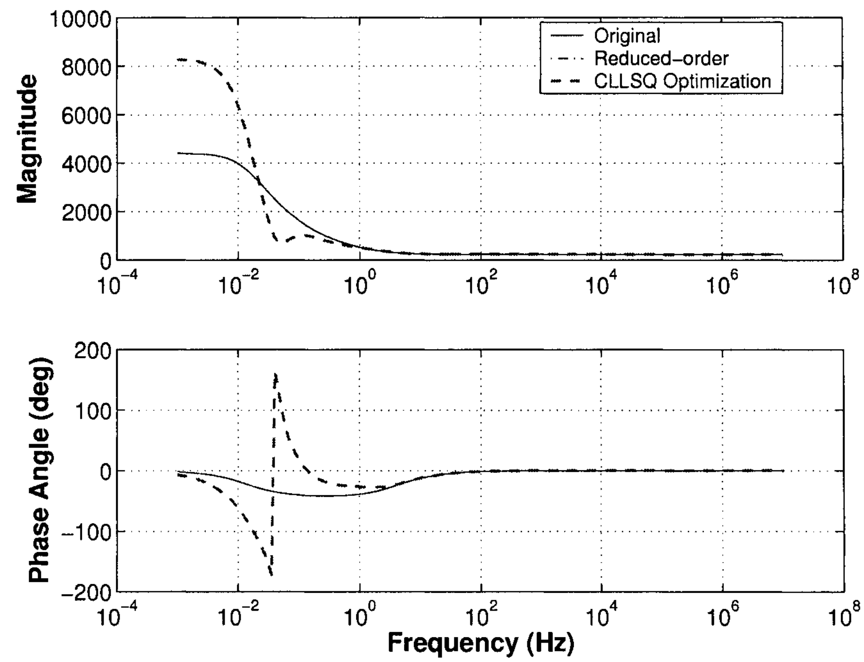


Figure 4.12: AIES Area 50 1203L aerial mode characteristic impedance

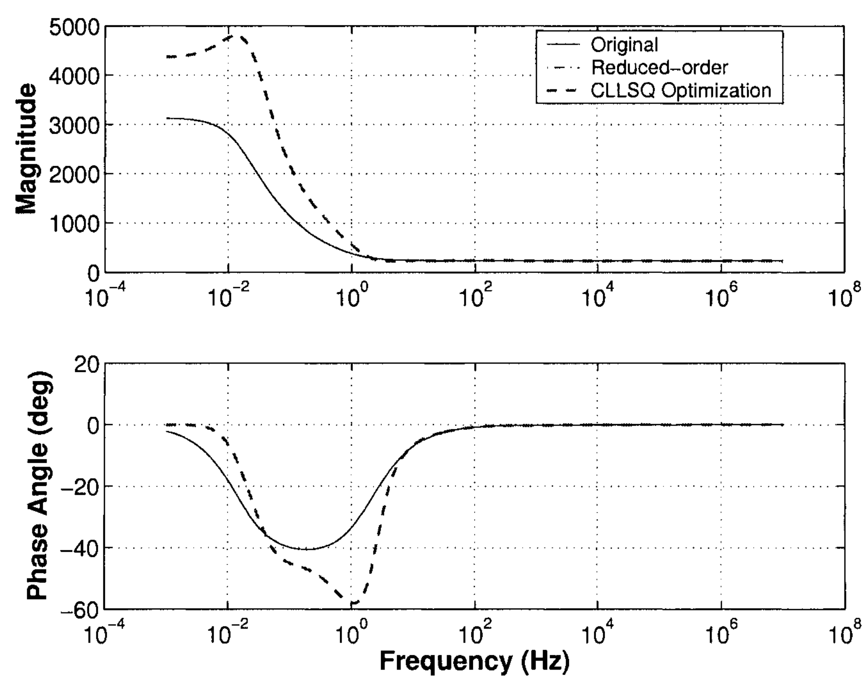


Figure 4.13: AIES Area 50 1209L aerial mode characteristic impedance

technique [42, 43] is used for aerial mode. Combining the surface layer with the results computed by GAs, CLLSQ is applied to find out the best suitable deep regions. The RMS-error% of input admittance versus deep region order is shown in Fig. 4.5 for aerial mode. The order of 25 was recognized as the optimal deep region order where with the higher orders of deep region, RMS-error% of input admittance does not decrease dramatically and is less than 10% (8.144%). Figures 4.8 through 4.10 show the aerial-mode deep region frequency response generated by GAs and after CLLSQ optimization. Fig. 4.7 shows the aerial-mode input admittance of external system. It can be observed that the first approximation of input admittance is very close to original. Frequency response of the external system input admittance after CLLSQ optimization overlaps that of GA first approximations. It is shown that due to their relative insensitivity to the input admittance, some pronounced resonant peaks in deep region are not chosen by GAs, which is salient feature of Robust TLNE. Further CLLSQ optimization mostly enhances the accuracy in the low frequency range, especially at DC, where the maximum RMS-error% is only 1.089%. This can be noticed from the frequency response of the surface layer characteristic impedance (shown in Figures 4.11 through 4.13), since in lower frequency range, it changes for the compensations on the frequency response at DC. The Norton equivalent current source for the external system is obtained through measuring short circuit current (Section 3.1.2) at input terminal, *i.e.*, Bus 524. The phase-A phasor at the input port is $1.881\angle\text{-}125.63°$ kA. Converting to modal domain, the Norton equivalent current source phasor vector $I_{eq,0\alpha\beta}$ is

$$\left[\ 0\angle 0^\circ \quad 2.304\angle\text{-}125.63^\circ \quad 2.304\angle\text{-}215.63^\circ\ \right]^T \text{kA}$$

It is observed that the order of deep region is drastically reduced and thus the obtained Robust TLNE model is very compact in computational aspect. Consequently, the Robust TLNE model for AIES is constructed. The ATP data file of Robust TLNE model is shown in Appendix C.3.2.

To achieve the same level of RMS-error% in aerial mode as Robust TLNE, the FDNE model requires 220th-order rational function. This is much higher than the combined order of the Robust TLNE model.

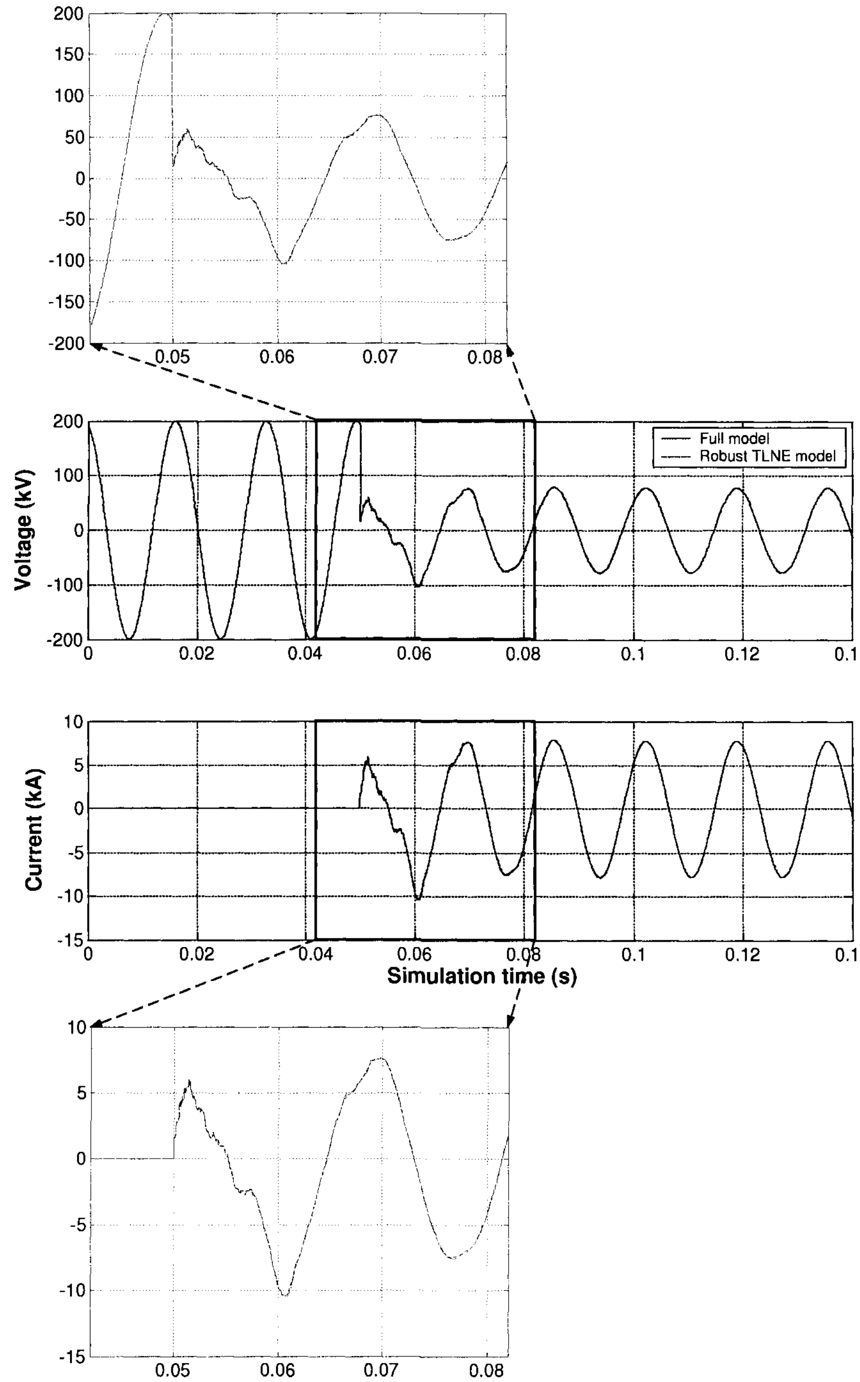


Figure 4.14: AIES Area 50 three-phase to ground fault transient simulation and comparison (phase A)

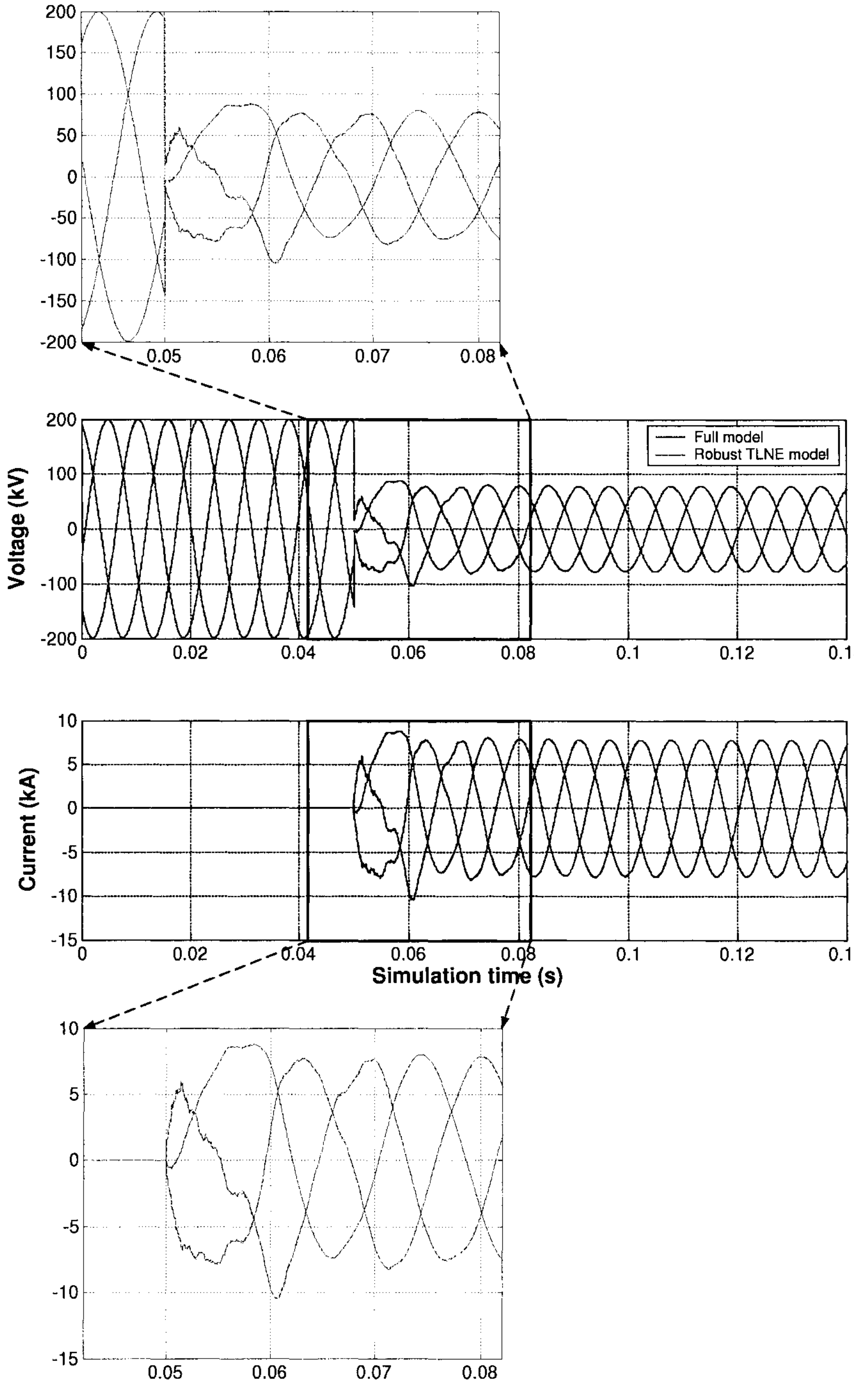


Figure 4.15: AIES Area 50 three-phase to ground fault transient simulation and comparison (three phase)

Transient events	Total time	Full model	Robust TLNE	220th FDNE
Balanced fault	0.15s	17.604s	0.081s	0.352s

Table 4.1: AIES Area 50 computational time comparison at time step size 20μs

4.5 Transient Simulations

Figures 4.14 and 4.15 show phase-A and three-phase voltage and current transients at Bus 524 Genesee, respectively. Only phase A is shown, since the same pattern is followed for all three phases. The balanced three-phase to ground fault with 10Ω resistance occurs at 0.05s. Total simulation time is 0.15s. All transients are verified via ATP with time-step size 20μs. Detailed agreement between the full model of the system and the the Robust TLNE model is observed. Computational time is a major saving in Robust TLNE. Table 4.1 shows computational time comparison among full model, Robust TLNE model and FDNE model on the same Pentium IV 1.6GHz computer. The 220th-order FDNE model does demonstrate great savings on computational time. However, computational savings of the Robust TLNE model are much more substantial. As observed from Table 4.1, the robust TLNE model is about 50 times faster than the full model. The more complex the system is, the larger computational saving we obtain in Robust TLNE model. It is also observed that simulation time of the Robust TLNE model is already well below 0.15s. With the precalculation of the inverse of the system conductance matrix in EMTP, more computational savings during real-time simulation of AIES Area 50 can be obtained.

4.6 Summary

In this chapter, the ATP model for a realistic large power system — AIES was developed. The AIES Area 50 Backbone model for electromagnetic transient studies in ATP was constructed based on detailed transmission line information from the TASMo database and equivalent circuits obtained using PSS/E. Then the Robust TLNE model for AIES Area 50 was developed and validated via time-domain simulations of three-phase to ground fault. Great computational saving in Robust TLNE was observed with respect to full model and FDNE model. Based on the computational time, the suitability of real-time implementation of large power systems by the Robust TLNE model is verified.

5

Real-Time Simulations Based on the Robust TLNE Model

In the previous chapters, a Robust TLNE model for large power systems has been developed and a realistic large power system — AIES has been modeled by the Robust TLNE. Transient simulations further verified accuracy and computational efficiency of the proposed model. In this chapter, the systems including Example 2 and AIES modeled by TLNE are to be implemented in the Real-Time Simulator in Real-Time eXperimental LABoratory (RTX-LAB), Power Engineering Group, University of Alberta.

5.1 Opal-RT Real-Time Simulator at RTX-LAB

The simulator in RTX-LAB [33] is a cluster-based, fully digital, parallel real-time simulator for power engineering research. It features high-performance computation units with high flexibility and scalability, high-speed communication links, Linux-based Real-Time Operating System (RTOS), fast FPGA (Field-Programmable Gate Array)-based analog and digital I/Os, MATLAB/SIMULINK-based development platform with customizable models, and highly accurate and efficient models and algorithms for power electronic applications.

The hardware and software architecture of the real-time simulator is shown in Fig-

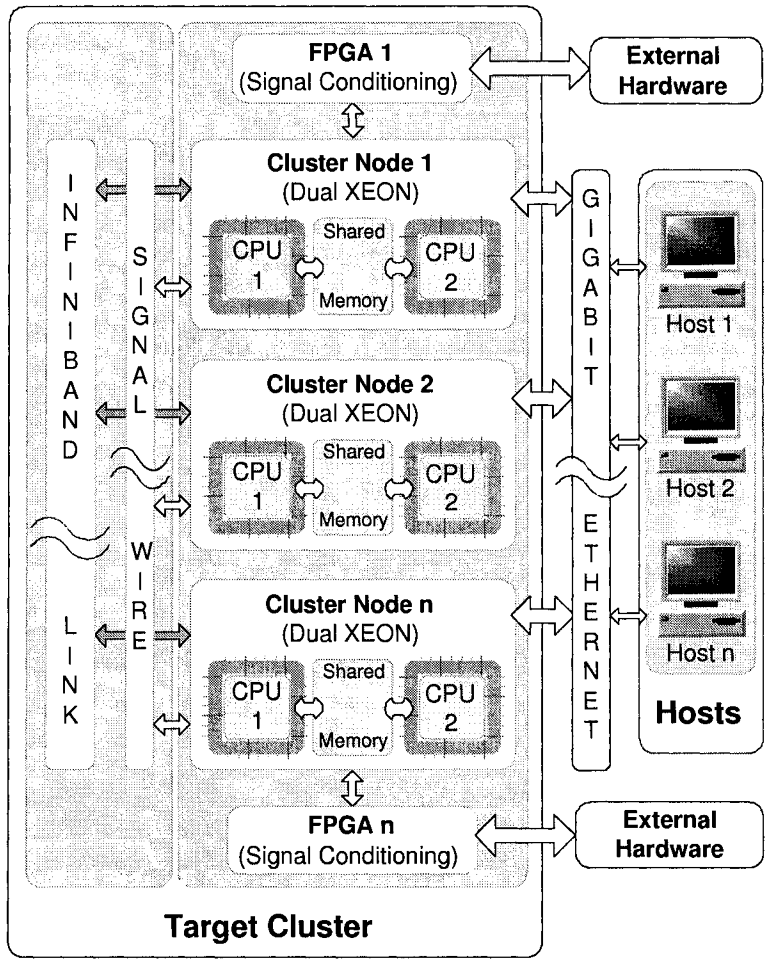


Figure 5.1: Hardware architecture of the RTX-LAB real-time simulator [33]

ures 5.1 and 5.2, respectively. It mainly comprises two groups of computers known as *target nodes* or *targets*, and *hosts*. In addition, high-speed communication links connect targets, as well as hosts and targets. External hardware is connected via FPGA-based analog/digital I/Os. Currently, the simulator has eight targets, each of which is powered by dual 3.0GHz Intel® Xeon™ processors. The two processors or CPUs in one target communicate with each other through shared memory. The targets is also capable of eXtreme High Performance (XHP) mode execution, in which one CPU is dedicated entirely to computation while the other CPU is running RTOS tasks and schedulers. The targets are also required to compile source code generated by MATLAB/SIMULINK Real-Time Workshop (RTW) to executables. The hosts are installed with a real-time

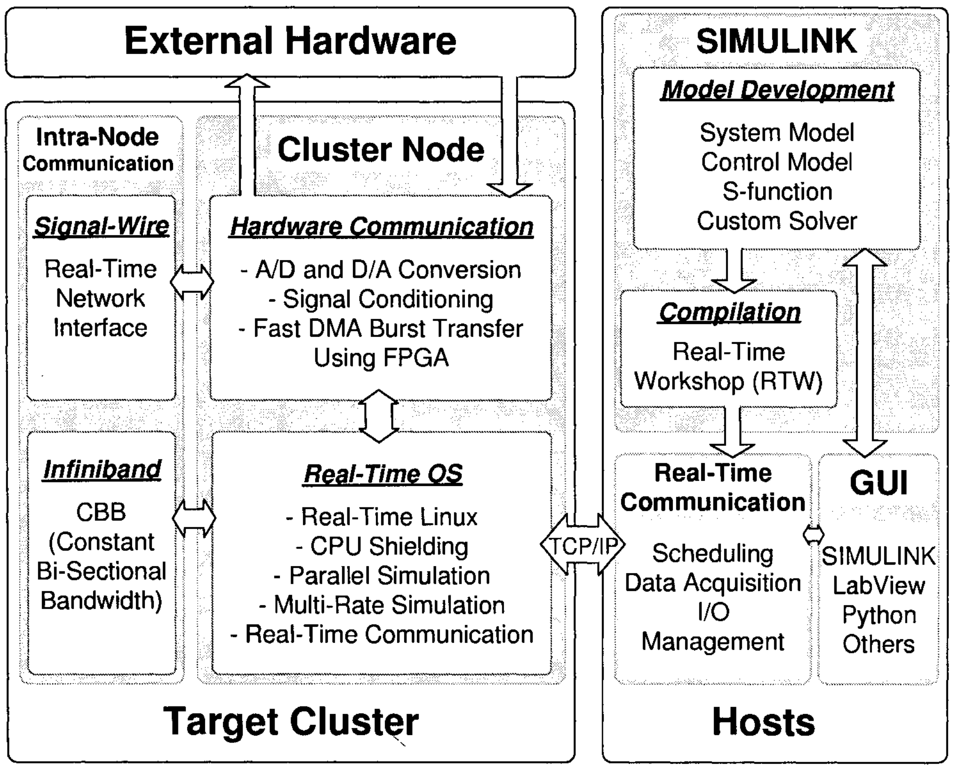


Figure 5.2: Software architecture of the RTX-LAB real-time simulator [33]

interfacing software called RT-LAB provided by Opal-RT Technologies Inc. to coordinate all hardware engaged for the simulation. The hosts are mainly used to create, edit and verify models in SIMULINK, compile SIMULINK blocks into C code by RTW, control and configure real-time simulations in targets, manipulate model parameters in real time as well as acquire real-time simulation results. In case that models are not available in SIMULINK, use-defined S-function block implemented by high-level programming language, *e.g.*, C/C++ and Fortran, *etc.*, can also be incorporated into the models. Again, each host is a high-performance computer which has an 3.0GHz Intel® Pentium® IV CPU to offer fast loading and compilation of the developed models in MATLAB/SIMULINK. Several state-of-the-art computer networking technologies have been utilized to achieve the best communication throughput:

- Shared memory for inter-processor communication in one target. It has the lowest latency.

- InfiniBand architecture for inter-target communication. It has low latency (from several to several-ten microsecond) depending on communication data size.

- SignalWire which only links adjacent two targets. It has only several-microsecond level of latency.
- Giga-speed Ethernet which mainly connects between targets and hosts, or among hosts.

Based on above introductions, this high-performance real-time simulation platform enables real-time implementation of large power systems modeled by Robust TLNE.

5.2 Implementation of EMTP in MATLAB/SIMULINK

In MATLAB/SIMULINK, due to the fact that Marti's frequency-dependent line model is not available, it is not straightforward to apply Robust TLNE model in SIMULINK. The models in MATLAB/SIMULINK are based on State-Space (SS) approach. However, EMTP uses nodal method. This leads to the solution on implementing EMTP with customized function blocks in SIMULINK. The SIMULINK S-function block, which is implemented by C/C++ and Fortran languages in S-function code format [49], is well-suited in this scenario. For enhanced efficiency, scalability and portability, the S-function is programmed in C++ language. The program mainly includes two parts — reading ATP data files with initializations and simulating obtained electrical networks based on EMTP nodal solutions.

5.2.1 C++ Implementation of EMTP

The C++ is a high-performance Object-Oriented Programming (OOP) language, which features encapsulation, inheritance and polymorphism [52]. Inherited from the nature of OOP, it is efficient, reliable, portable and scalable. The EMTP S-function program fully utilizes the merits of C++ language.

Overview of C++ Classes and Structures for EMTP

In C++, *classes* [52] are the basic elements for OOP programming. The functions inside a class is called *class methods*. An *object* is the instantiation of a class. A class can inherit or be derived from other classes, whose properties will be possessed by the class. The class

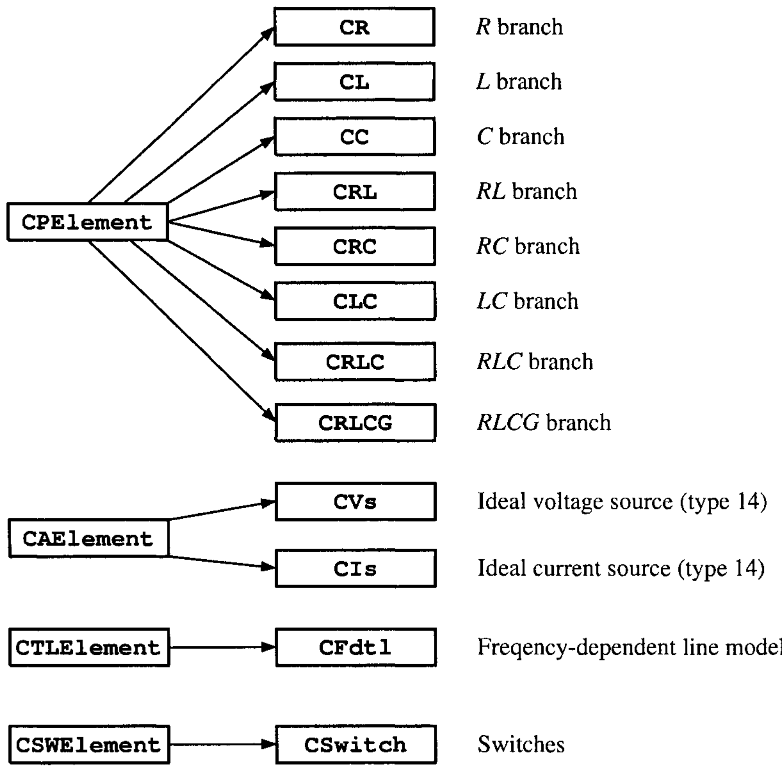


Figure 5.3: C++ class hierarchy for EMTP

for derivations is called *base class*. The C++ concepts can be directly applied in EMTP. Every element of an electrical network is considered as an object instantiated from a class. In a linear network, elements are classified as four base classes — `CPElement`, passive elements such as R, L, C and their combinations; `CAElement`, active elements such as ideal voltage and current sources; `CTLElement`, transmission line models with the consideration of traveling wave effect; `CSWElement`, switches. All realistic element models are derived from the four base classes. Fig. 5.3 shows the class hierarchy for EMTP. Due to the requirement in the Robust TLNE, Marti's frequency-dependent line model is implemented as a class called `CFdtl`, which is derived from `CTLElement`.

One of the most useful features in C++ is polymorphism, in which, a method of a derived class is can be invoked in the base class level by casting the objects of derived class to the ones of base class. For instance, all passive elements derived from `CPElement` class have a method called `update`, which is used to update history current terms in discretized equivalents. In writing the code to update history terms with polymorphism, we only need to call the method in base class level instead of calling the same methods in each derived class individually. Thus polymorphism reduces programming complexities and potential bugs. To enable this feature so that the C++ compiler knows exactly the method belonging to which class should be called, a keyword `virtual` is used in the declaration of the methods in base classes.

Class `CPElement` and its Derived Classes

In EMTP, passive elements such as R, L, C mainly have one action to accomplish in each simulation loop — updating history current terms. This is done by calling the method `void update(double&)` in `CPElement` class, where the method parameter is the nodal voltage. In addition, the class must be able to provide the element properties including discretized equivalent conductance, equivalent history current term, and the current that flows through the element. This is realized via methods `double getG()`,`double getIh()` and `double getib()` in `CPElement` class, respectively. The above four class methods form the basic interface of passive elements for the EMTP nodal solution part to invoke. For increased efficiency during simulation, R, L, C, RL, RC, LC, RLC elements are modeled by separate C++ classes derived from `CPElement`. Besides above well-known branch models, $RLCG$ branch model discussed in Section 2.1.4 used in FDNE is also implemented as a separate class called `CRLCG`.

As illustrated in Fig. 2.3, the $RLCG$ branch includes an internal node. Discretization of this branch requires the elimination of the internal node so that obtained equivalent circuit is not only efficient but also can be implemented by derived class of `CPElement`. To realize this approach, both RL and CG blocks are first discretized as an equivalent resistor in series with a history voltage term. The voltage term is only dependent on branch current. Then two equivalent resistors and two voltage terms are merged and grouped. Finally, the circuit is converted to a Norton equivalent — an equivalent resistor in parallel with a history current term, which is suitable for the implementation in a derived class of `CPElement`. Updating current terms requires the calculation of branch current. Therefore, in class method `update`, branch current is first calculated. Since the internal node is eliminated, computational efficiency will increase during matrix computations due to reduced dimension. The same discretization method is also applied to

LC and *RLC* branches for increased efficiency.

During initialization for passive elements in C++, the discretized equivalent resistor and the coefficients of the equation for updating history current terms are calculated and stored based on the class initialization method parameters including the value of all branch elements as well as simulation time-step size. Therefore, those values can be readily accessed during simulation. Appendix D lists the equations for obtaining Norton equivalent and updating history current terms. Those equations are the key to the implementation of passive element classes.

Class `CAElement` and its Derived Classes

The classes for ideal current sources `CIs` and voltage sources `CVs` are derived from `CAElement` class. Three interfaces are provided, which are used to obtain internal conductance by the method `double getG()`, equivalent history current term by the method `double getIeq(double&)`, and branch current by the class method `double getib(double&, double&)`. Current sources are assumed to have very large resistance, *e.g.*, $10^{10}\Omega$. Meanwhile, voltage sources are assumed to have very small resistance, *e.g.*, $10^{-10}\Omega$. The small resistance allows voltage source to convert to Norton equivalent, which is suitable for EMTP. It is verified that the inclusion of the resistance in current and voltage sources has virtually no effect in simulations.

The current simulation time is the parameter of class method `getIeq`, where the instantaneous value of the source is calculated in every time step. Two parameters of `getib` are nodal voltage and current simulation time, respectively. Initialization requires three parameters — frequency, peak magnitude and phase shift.

Class `CTLElement` and its Derived Classes

The `CTLElement` class is used to implement transmission lines with traveling wave effects. Referred to Fig. 2.7, it has six methods — `void update(double&, double&)` for updating history current terms, `double getG()` to obtain discretized equivalent conductance, `double getIhk()` and `double getIhm()` to obtain equivalent current sources of both sides, `getik()` and `getik()` to obtain branch current of both sides. In Robust TLNE model, only frequency-dependent line model is required to be implemented. The class `CFdtl` implements this line model. It strictly conforms to the equations developed in Section 2.1.5. In addition to the methods defined in the base class, the class method `void updateBkm()` is used to update backward traveling functions $b_k(t)$ and $b_m(t)$ shown in Fig. 2.4 via recursive convolution [24]. The function `void ArrangeF()` is invoked by the class method `update` internally to arrange the forward traveling functions values stored due to traveling delay in every time step. The two parameters of the class method `update` are the nodal voltages of both sides.

Initialization of `CFdtl` class requires poles, residues and constant term of $Z_{eq}(s)$ in (2.22), poles and residues of $P_a(s)$ in (2.32), traveling time τ, and time step size. Then discretized equivalent resistor R_{eq} in (2.27), the coefficients of the equations to update RC parallel blocks in (2.29), and the coefficients of the equations for recursive convolution in (2.36) are calculated and saved.

Class `CSWElement` and its Derived Classes

Class `CSWElement` is the base class for switch classes to derive. In this program, only ideal switch class called `CSwitch` is implemented. The switch class uses very large and very small resistance to simulate switch open and close. Three class methods are provided — `double getG()` for getting the current conductance of the switch, `double getib(double&)` to obtain the branch current flowing through the switch, `void update(bool&)` to update the resistance of the switch according to the current switch state. The parameter of `getib` is the nodal voltage. Logic-type switch state is the parameter of the class method `update`. Logic "1/true" stands for switch open, while logic "0/false" stands for switch close.

C++ Matrix Template Class

The class for matrix manipulation and operation is not available in C++ standard libraries. Thus, a third-party C++ matrix template class, called Matrix TCL Lite by Tech-Soft Pvt. Ltd. [54], is used. The Matrix TCL Lite is only capable of basic linear algebra operations such as matrix addition, substraction, multiplication as well as inversion. Since only matrix multiplication and inversion are required in EMTP nodal solution, the matrix class is well-suited for C++ EMTP implementation.

Type-defined by `matrix<double>`, `CMatrix` is the class used for matrix computations. In C++, taking the advantage of operator overloading [52], the operators for matrix addition, substraction, multiplication and inversion are "+", "-", "*", and "!", respectively. The C++ header file matrix.h is required to be included in the program.

C++ Structures for Storing Electrical Networks

In order to readily and efficiently construct the conductance matrix and current source vector of an electrical network during simulation, several C++ structures are created in this program to store the network in this fashion.

The CBranch and CShunt structures are the key structures in this EMTP implementation. The CBranch structure includes all branch-type elements in the network which connect between two nodes. The CShunt structure encompasses all shunt-type elements in the network which connect to ground. All linear elements are able to classified into either type of the structures. The EMTP element objects of the EMTP classes discussed in this section is stored as void *data in both CBranch and CShunt. In addition to the key structures, structures CIout, CVout and CSWNode are used to save current output, voltage output and switch information, correspondingly.

In order to dynamically allocate memory based on the electrical networks to be simulated, vector class available in C++ Standard Template Library (STL) [53] is employed. It features high performance, scalability and portability and is seamlessly incorporated with other C++ classes.

Several vectors are defined to provide complete information for the electrical network to be simulated:

- CNVec. Type-defined by vector<string>, it is used to store all node names in the network.
- CBVec. Type-defined by vector<CBranch>, it is used to store all branch-type elements defined in structure CBranch.
- CSVec. Type-defined by vector<CShunt>, it is used to store all shunt-type elements defined in structure CShunt.
- CIVec. Type-defined by vector<CIout>, it is used to store all current outputs defined in structure CIout.

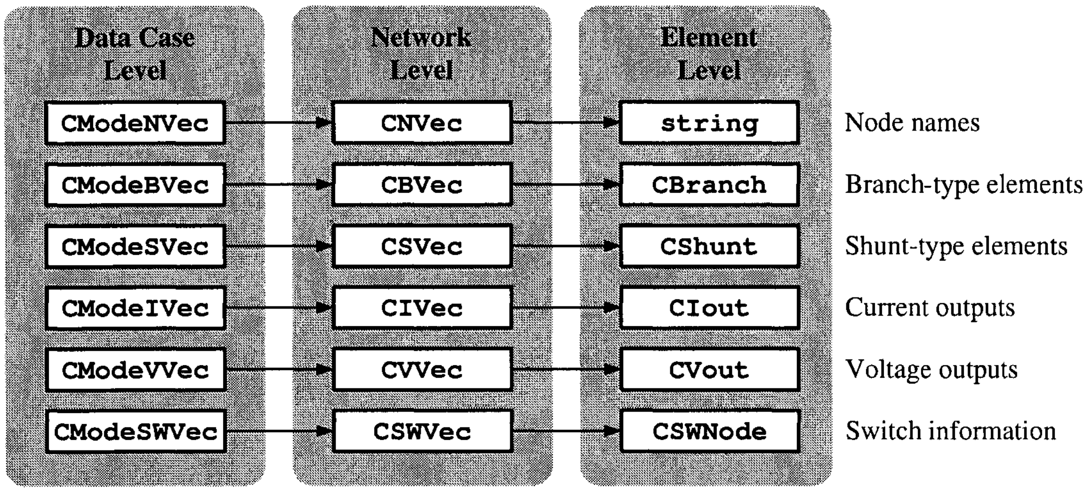


Figure 5.4: C++ data type hierarchy for EMTP

- `CVVec`. Type-defined by `vector<CVout>`, it is used to store all voltage outputs defined in structure `CVout`.
- `CSWVec`. Type-defined by `vector<CSWNode>`, it is used to store all switch information defined in structure `CSWNode`.
- `CMVec`. Type-defined by `vector<CMatrix>`, it is used to store all inverse conductance matrices defined in class `CMatrix`.

All elements and information are properly stored in the above vectors during reading the ATP data files.

To support for multiple data cases, the following vectors are also defined:

- `CModeNVec`, the vector of data type `CNVec`.
- `CModeBVec`, the vector of data type `CBVec`.
- `CModeSVec`, the vector of data type `CSVec`.
- `CModeIVec`, the vector of data type `CIVec`.
- `CModeVVec`, the vector of data type `CVVec`.
- `CModeSWVec`, the vector of data type `CSWVec`.

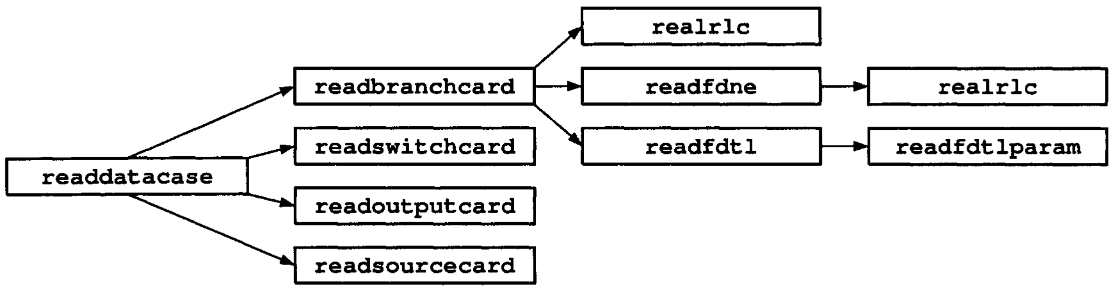


Figure 5.5: C++ function hierarchy for reading ATP data files

- `CModeMVec`, the vector of data type `CMVec`.

Then each data case representing a specific network is saved corresponds to the index in above vectors. The data type hierarchy is shown in Fig. 5.4.

Reading ATP Data Files

In order to generalize the network to be simulated, ATP data files are used as standard input data files for electrical networks. In this fashion, the program is fully adaptable for different electrical networks. Fig. 5.5 illustrates the function hierarchy for reading ATP data files. The main function to read data files is `readdatacase`, inside of which, it calls the corresponding functions to read sorting cards in data files:

- Function `readbranchcard` reads branch cards beginning with `/BRANCH`. Inside this function, three functions are called — `readrlc` to read *RLC* branch, `readfdne` to read FDNE model circuit including *RLCG* branches, and `readfdtl` to read Marti's frequency-dependent line model. The function `readfdne` actually invokes `readrlc` to obtain FDNE model elements. Function `readfdtlparam` reads parameter blocks of Marti's frequency-dependent line model.
- Function `readswitchcard` reads switch cards beginning with `/SWITCH`.
- Function `readoutputcard` reads voltage output cards beginning with `/OUTPUT`.
- Function `readsourcecard` reads source cards beginning with `/SOURCE`.

After invoking the functions for reading ATP data files, all vectors and objects are initialized corresponding to the electrical network to be simulated.

The data file format strictly follows the definitions in ATP Rule Book [44]. Currently, only four types of sorting cards — `/BRANCH`, `/SWITCH`, `/SOURCE` and `/OUTPUT` are supported, inside which, the support for EMTP elements are:

- Linear RLC branch with branch current output only.
- Linear $RLCG$ branch, which is not originally available in ATP. By using $RLCG$ branch, total number of nodes in EMTP can be greatly reduced. Thus computational time is potentially improved. To enable the program to read $RLCG$ branch instead of separate R, L, C branches, two guidelines must be followed:
  1. The $RLCG$-branch format is required to follow the format generated by VF for FDNE models. Examples are available in data files in Appendices A.4 and B.4.
  2. In ATP data files, statements `C BEGIN FDNE` and `C END FDNE` are required to immediately precede and immediately follow the FDNE network block, respectively. The first statement is especially important.
- Marti's frequency-dependent line model, whose parameters are generated by ATP LINE CONSTANT routine. Only branch current output is available.
- Type 14 ideal current and voltage sources with current output.
- Ideal switches with current output.

Nodal voltage outputs is supported via `/OUTPUT` card. Current measurement is not explicitly supported. However, this can be done by including an R branch of very low resistance and enable branch current output. The sequence of the outputs is that branch current outputs come first and then nodal voltage. The program only support node names of exactly six characters. Node names of less than six characters will lead to unexpected results.

The program is also capable of simulating multiple independent networks simultaneously. This is done by including multiple data cases in one ATP data file with `BEGIN DATA CASE` and `END DATA CASE` statements. This feature is also supported in ATP [44].

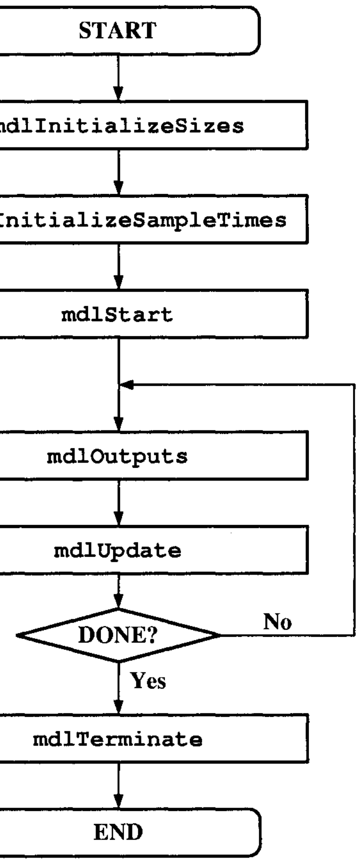


Figure 5.6: SIMULINK S-function flowchart for EMTP

Simulation of Robust TLNE model of three-phase systems is only done through Clarke's transformation of study zone from phase domain to modal domain. However, this is not suitable for general scenarios due to the fact that the study zone may include nonlinear and time-variant elements. Thus, it is a future work to integrate direct three-phase solution method [1] into the EMTP program so that it is suitable for any generic case of transient studies in three-phase systems with the Robust TLNE model.

Limitations on supporting other elements can be gradually reduced by including more types of network elements and incorporating more functionalities in the program in the future, since the C++ code is written in a flexible and scalable way.

5.2.2 SIMULINK S-Function Program Implementing EMTP

The SIMULINK S-Function block is invoked in every simulation loop during simulation. As shown in Fig. 5.6 flowchart, the essential functions in C++ S-function to be used are the follows.

Function `static void mdlInitializeSizes(SimStruct *S)`

This is the function to specify the basic characteristics of the block, such as how many inputs, outputs, the port width of each input and output, and the number of parameters of the S-function block. In this case, one input, one output and two parameters are defined. The input corresponds to switches in the network to be simulated — "1" for switch close, "0" for switch open. This logic is different from the one used in the class method `update` of the `CSwitch` class explained in the previous section. Therefore the logic NOT operating is applied to correct the discrepancies. The output is voltage and current outputs requested in data files for display. Default port width for input and output is both 3, which can be modified by `MAXINPUTPORTWIDTH` and `MAXOUTPUTPORTWIDTH` macros defined in the beginning of the program, if the port width is not sufficient. Two parameters are used in the block — sample time/time-step size and the full file name of the ATP data file for real-time simulation.

Function `static void mdlInitializeSampleTimes(SimStruct *S)`

Sample time of the S-function block is initialized here, which is retrieved from the first S-function parameter by C function `mxGetScalar(ssGetSFcnParam(S, 0))`.

Function `static void mdlStart(SimStruct *S)`

All initializations of EMTP models and matrices are placed in this function. Via functions in Fig. 5.5, ATP data files are read and interpreted, and corresponding models are initialized. Moreover, in order to reduce computational time in the simulation loop, all matrices corresponding to the switching states are built and their inverses are precalculated via `CMatrix` class and saved for future access in simulation loop.

Function `static void mdlOutputs(SimStruct *S, int_T tid)`

This is the function invoked in every simulation loop. Outputs requested in ATP data file are sent to S-function block output. The outputs are calculated in `mdlUpdate` function discussed below.

Function `static void mdlUpdate(SimStruct *S, int_T tid)`

This is the main function implementing EMTP. Invoked in each time step, the function first reads switch states from the S-function block input and retrieves matching inverse system conductance matrix precalculated during initialization in `mdlStart`, then obtains current source vector, calculates nodal voltage vector and updates the history terms (current sources) of EMTP models, and finally calculates outputs requested in the data file and saves them to be used by the function `mdlOutputs` for S-function block output.

Function `static void mdlTerminate(SimStruct *S)`

In this function, memory blocks allocated for storing EMTP models are freed to ensure no memory leakage in the C++ program.

5.2.3 Working with EMTP S-Function Block in Real-Time Simulator

To incorporate EMTP S-function block into RTX-LAB simulator, the following files are required to transfer to targets in RT-LAB: `emtp.cpp`, `emtp.h`, `matrix.h` and the ATP data file for real-time simulation. Together with existing C code generated by RTW, the C++ code is compiled by GCC/G++ in targets to generate real-time executables. However, the S-function program is only able to run in a single target. Implementation of distributed and parallel computation for EMTP is one of the future works to extend in this C++ program.

The complete C++ source code of S-function program is listed in Appendix E. To compile the source code into the MEX-function (executable for MATLAB with extension `.dll` in Microsoft Windows platform), the following command is used in MATLAB:

```
>> mex -g emtp.cpp
```

or

```
>> mex emtp.cpp
```

The -g option is used to include debugging information in the MEX-function. More information on debugging SIMULINK S-function is available in [49].

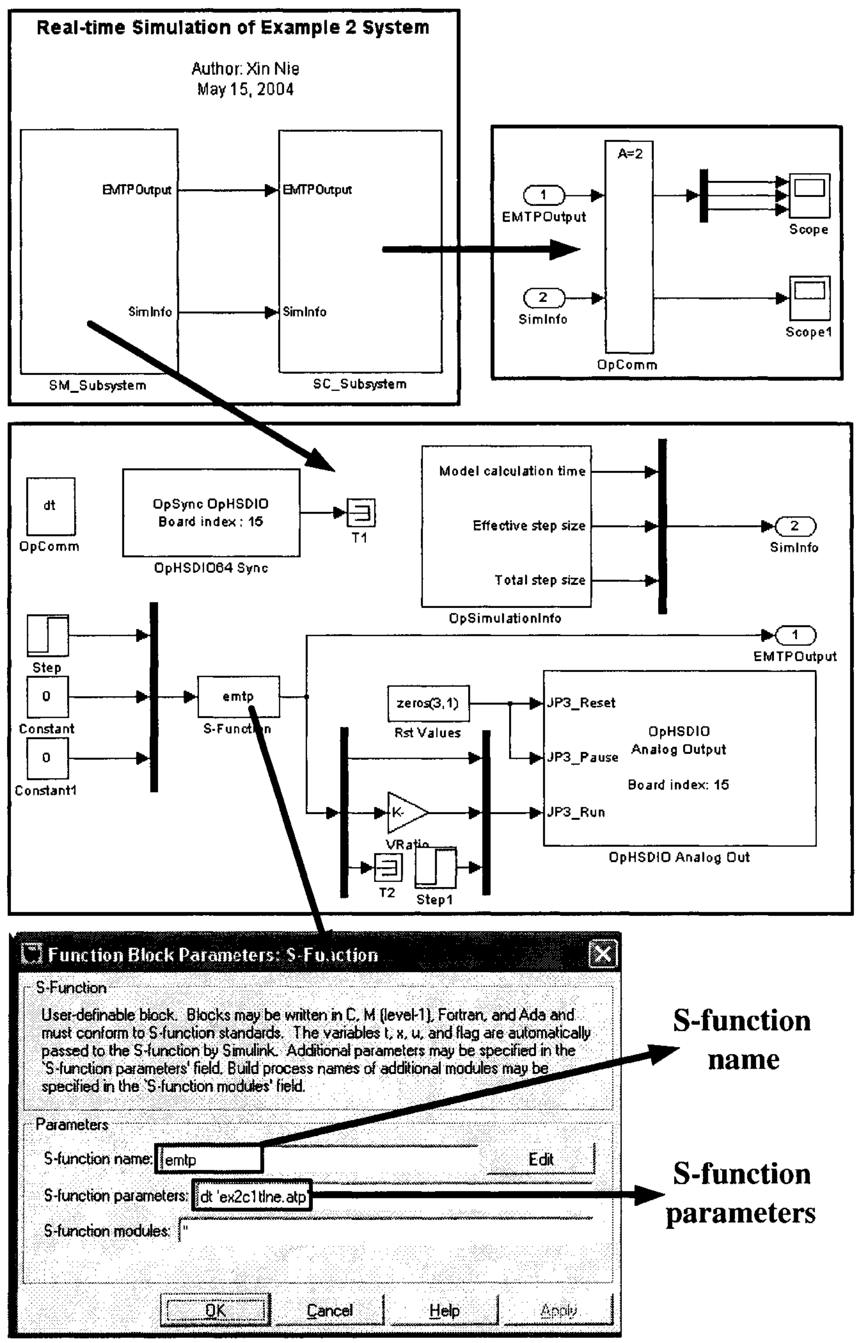


Figure 5.7: Example 2 SIMULINK diagram for S-function based EMTP

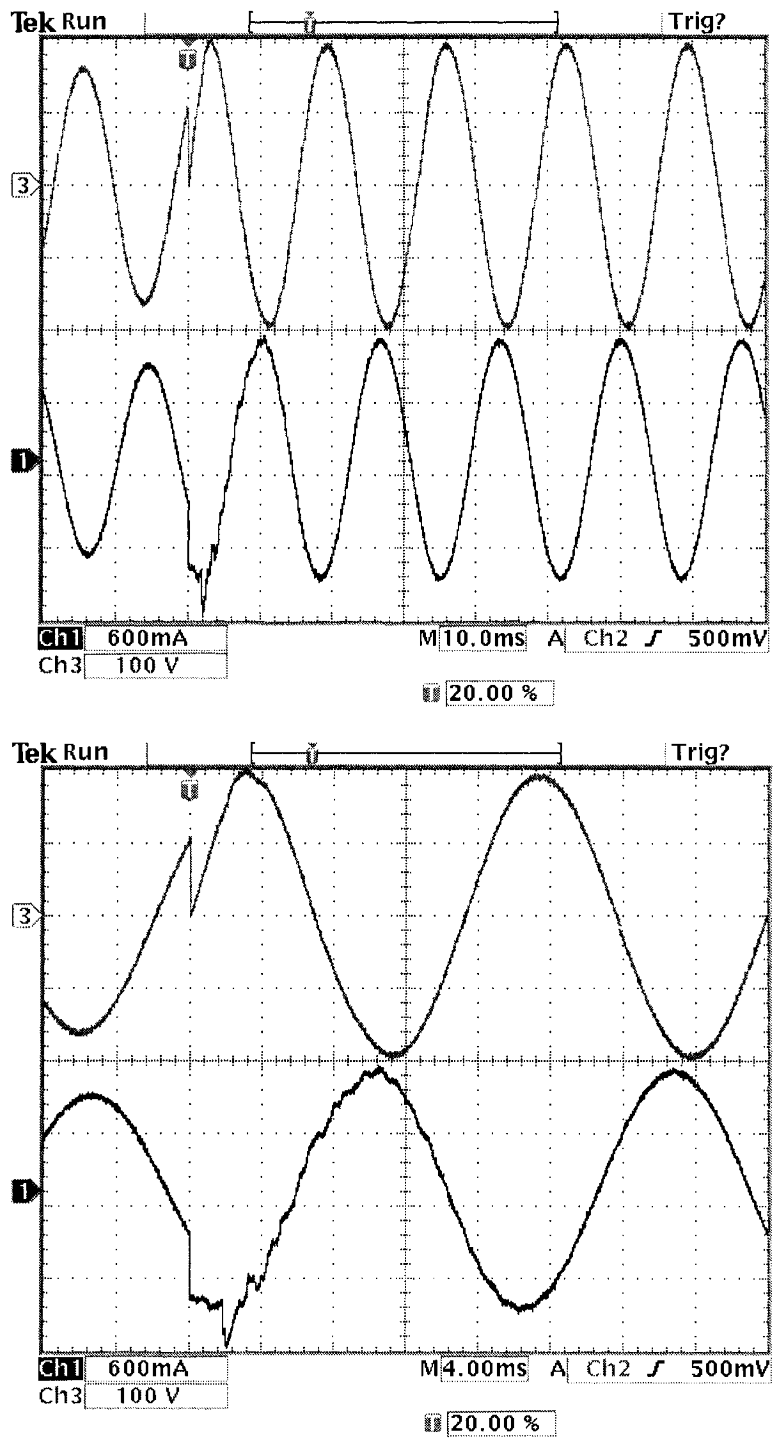


Figure 5.8: Example 2 real-time simulation result ($\times$1k) of capacitor switching (phase A)

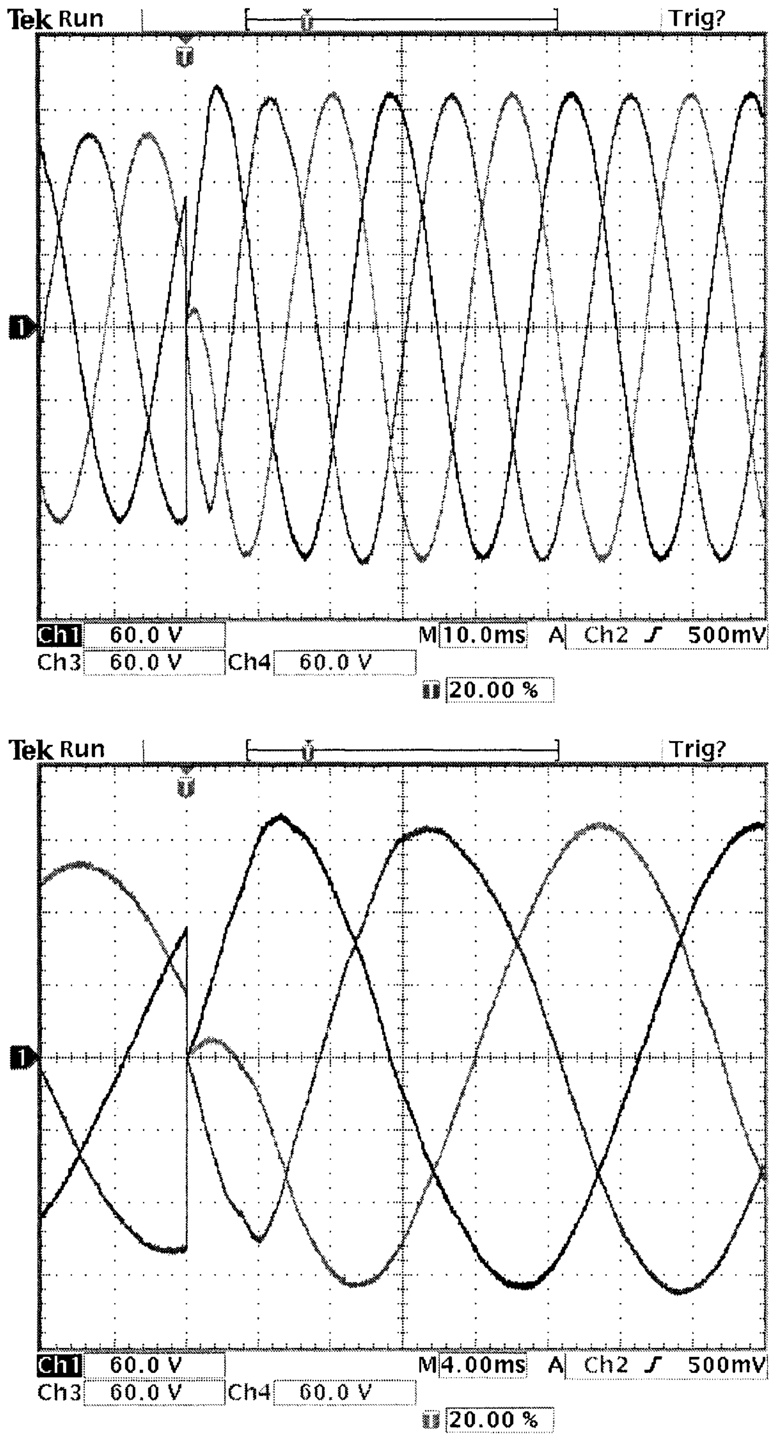


Figure 5.9: Example 2 real-time simulation voltage result (×1k) of capacitor switching (three-phase)

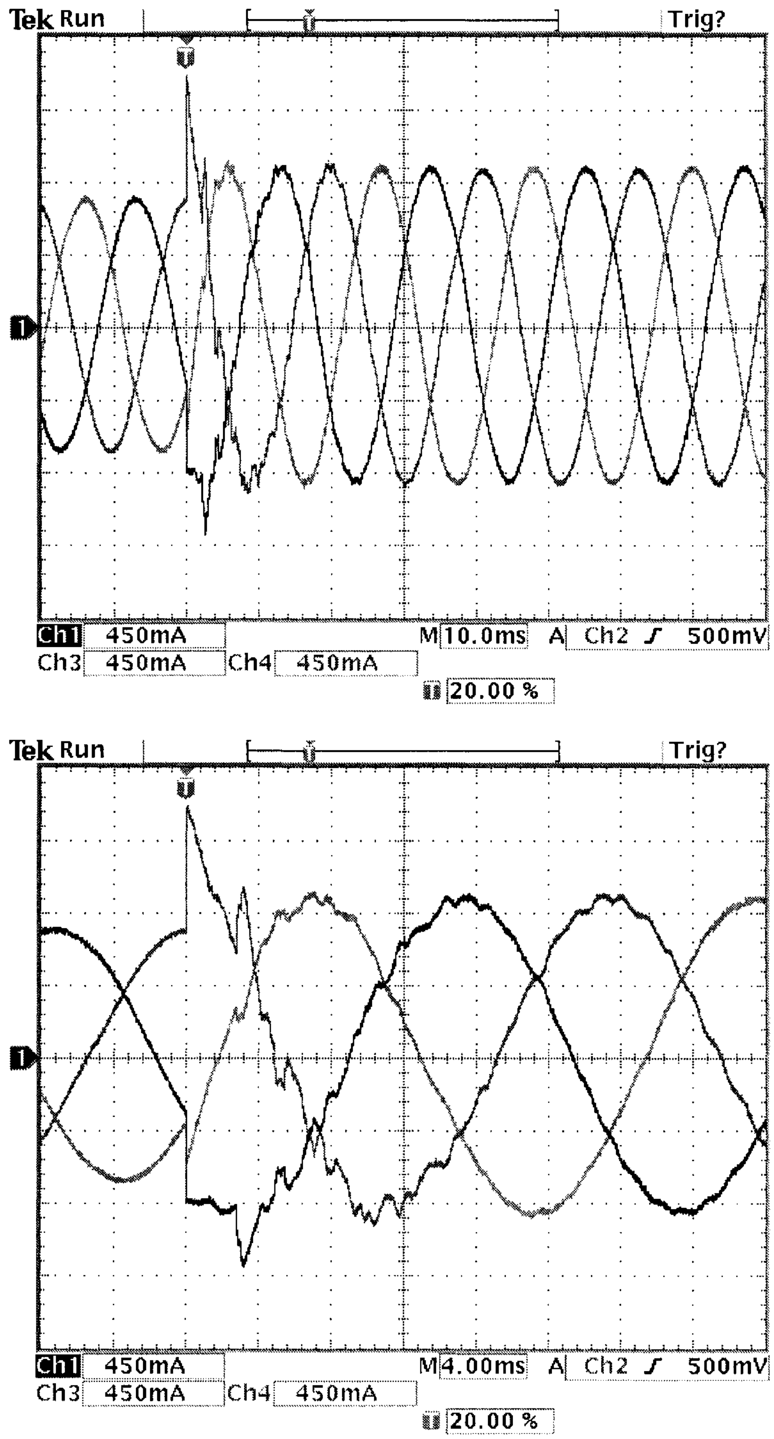


Figure 5.10: Example 2 real-time simulation current result (×1k) of capacitor switching (three-phase)

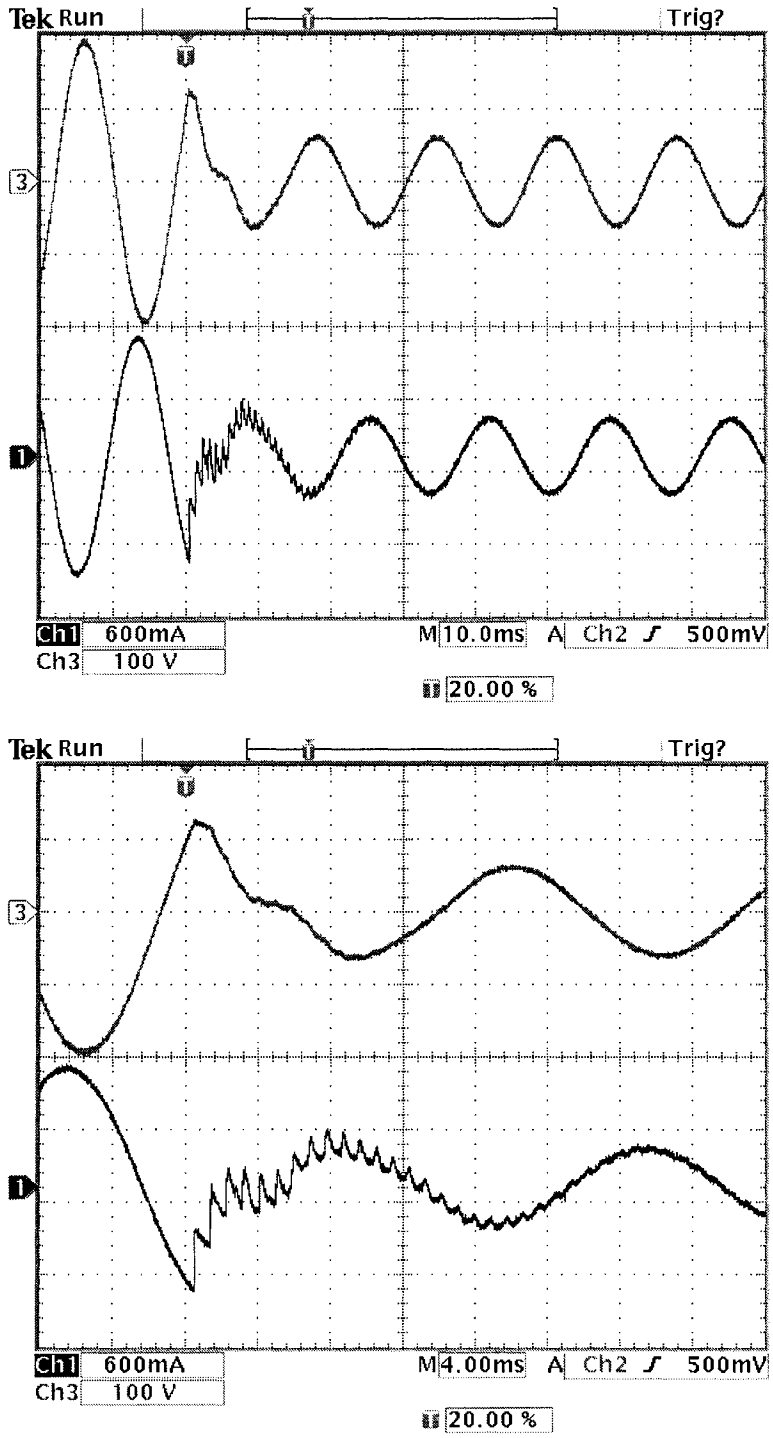


Figure 5.11: Example 2 real-time simulation result (×1k) of balanced fault (phase A)

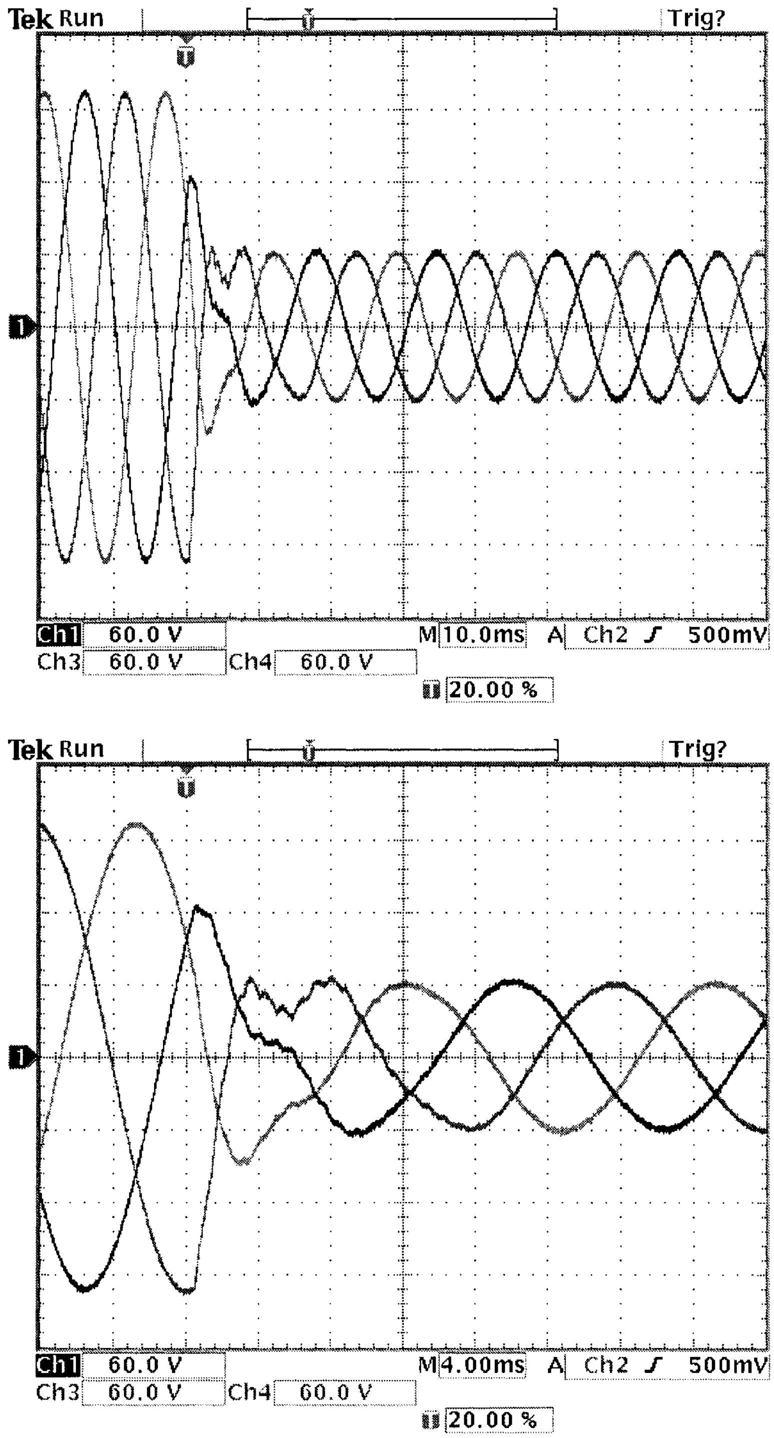


Figure 5.12: Example 2 real-time simulation voltage result (×1k) of balanced fault (three-phase)

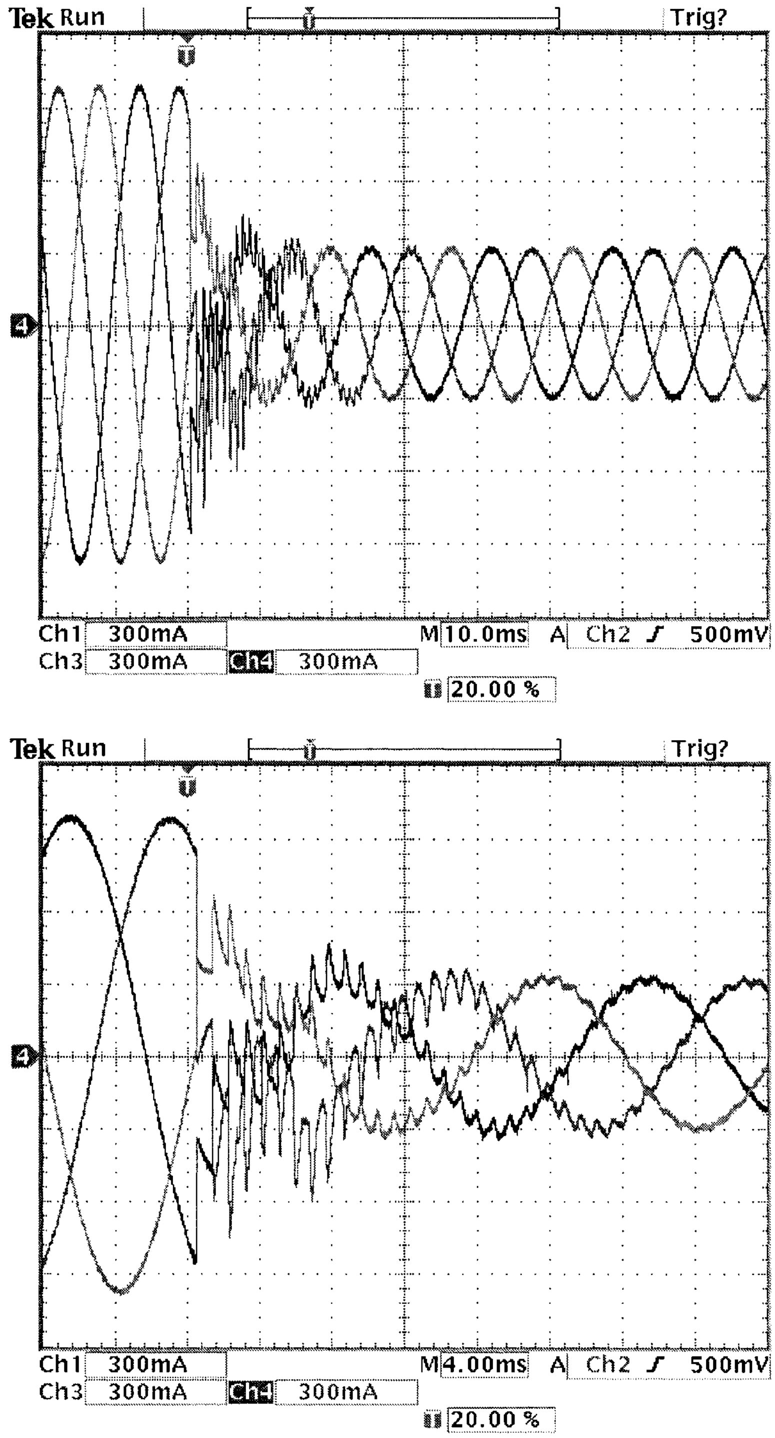


Figure 5.13: Example 2 real-time simulation current result (×1k) of balanced fault (three-phase)

5.3 Real-Time Simulation of Example 2

The system of Example 2 in Section 3.3 is implemented in RTX-LAB real-time simulator at 40μs time step size. Fig. 5.7 illustrates how the diagram is built in SIMULINK for the real-time implementation of the system. Since two events — capacitor switching and balanced three-phase to ground fault use different ATP data files (`ex2c1tlne.atp` and `ex2c2tlne.atp` listed in Appendix B), corresponding file names are entered in the "S-function parameter" field, as shown in Fig. 5.7. Figures 5.8 through 5.10 show capacitor switching (the first transient event) waveforms captured by oscilloscope of phase-A voltage and current, three-phase voltage, and three-phase current, respectively. Figures 5.11 through 5.13 show three-phase ground fault (the second transient event) waveforms captured by oscilloscope of phase-A voltage and current, three-phase voltage, and three-phase current, respectively. The waveforms are indistinguishable to the corresponding off-line simulation results shown in Figures 3.18 through 3.21.

5.4 Real-Time Simulation of AIES

The Robust TLNE model for AIES Area 50 is efficient and suitable for real-time simulation. The real-time simulation diagram in SIMULINK is shown in Fig. 5.14. The file name `aies50tlne.atp` is supplied in the second S-function parameter. Time-step size of 40μs is achieved during real-time simulation. Fig. 5.15 through 5.17 show transient waveforms captured by oscilloscope of phase-A voltage and current, three-phase voltage, and three-phase current, respectively. They are also indistinguishable to the off-line simulation results of AIES Area 50 in Figures 4.14 and 4.15. This case study fully verifies the efficiency and accuracy of the Robust TLNE model and its suitability of real-time simulation of large power systems as well as the efficiency and accuracy of the C++ SIMULINK S-function program implementing EMTP.

5.5 Summary

This chapter introduced the hardware and software architecture of a versatile and fully digital real-time simulator based on the MATLAB/SIMULINK development platform in RTX-LAB, Power Engineering Group, University of Alberta. A C++ S-function im-

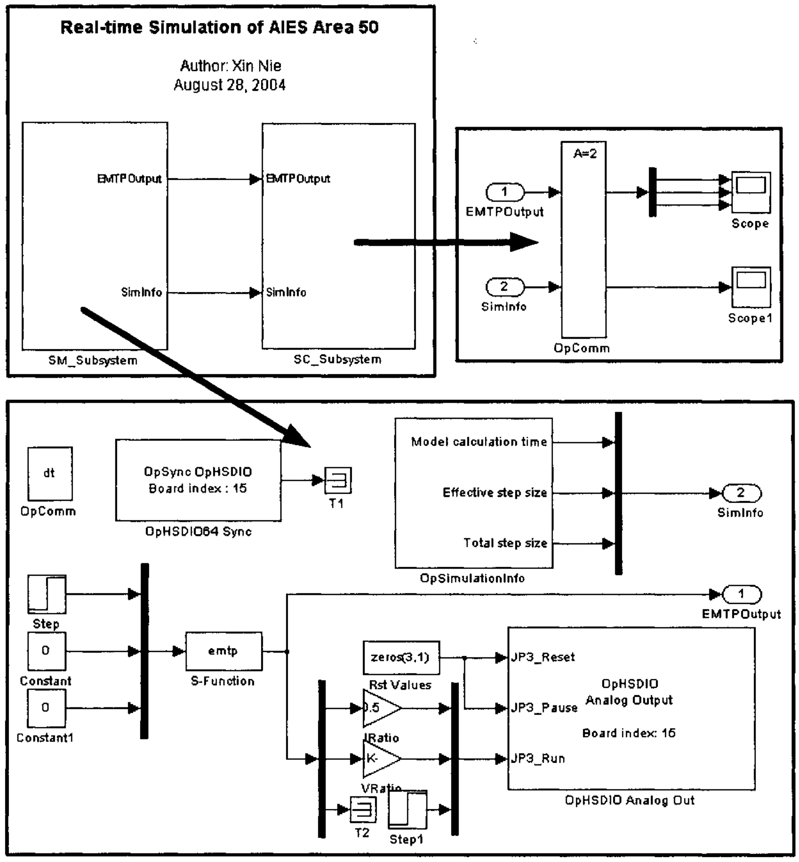


Figure 5.14: AIES Area 50 SIMULINK diagram for S-function based EMTP

plementation of EMTP in SIMULINK, including C++ classes for network elements, C++ structures for EMTP, and C++ functions to read ATP data files, is explained in detail. The merits of the S-function program are the OOP modeling of electrical network elements, the capability of reading ATP data files, the ability to dynamically allocate the memory for different networks, and the efficiency in computation using C++ `vector` class. Then the systems of Example 2 in Chapter 3 and a realistic large power system — AIES in Chapter 4 are implemented in real time, further validating the computational efficiency and accuracy of the Robust TLNE model.

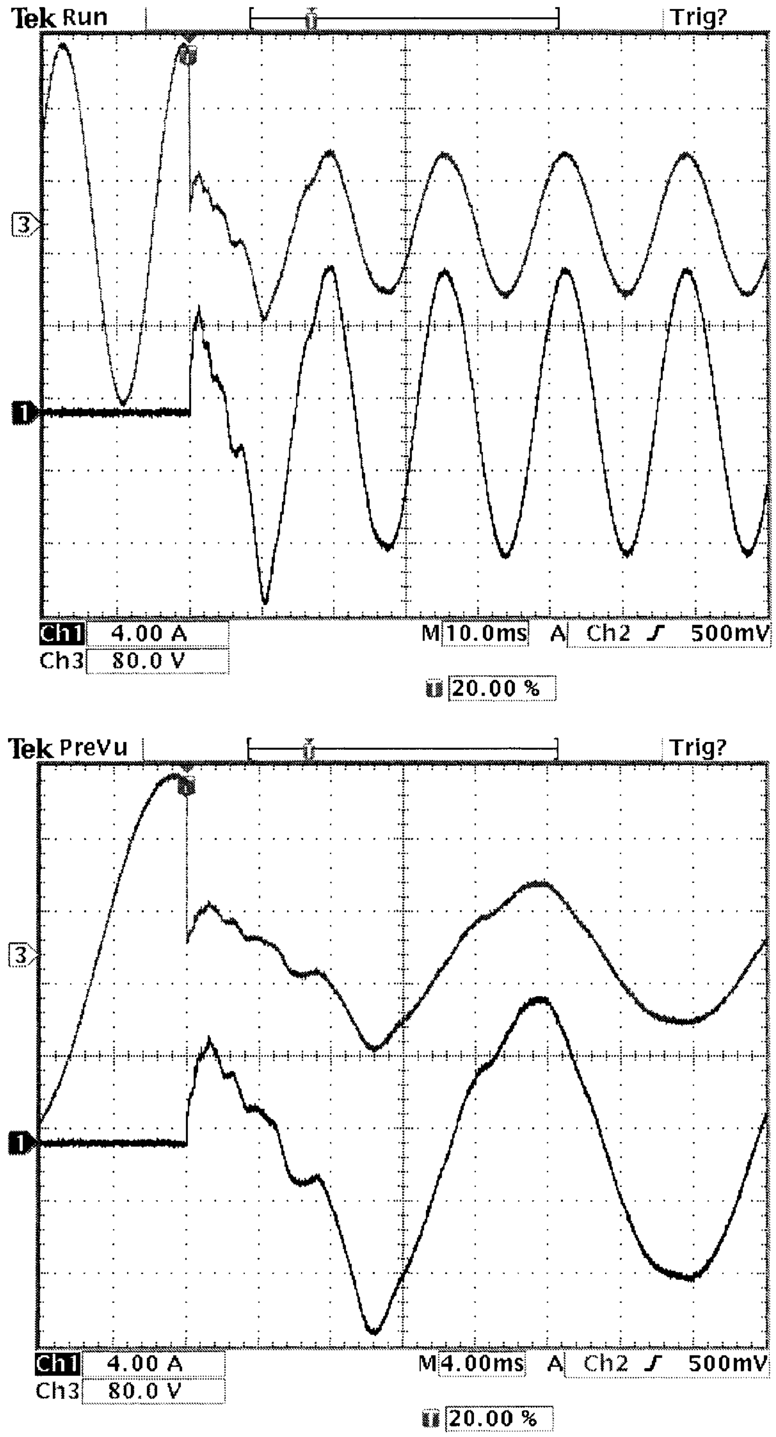


Figure 5.15: AIES Area 50 real-time simulation result (×1k) of balanced fault (phase A)

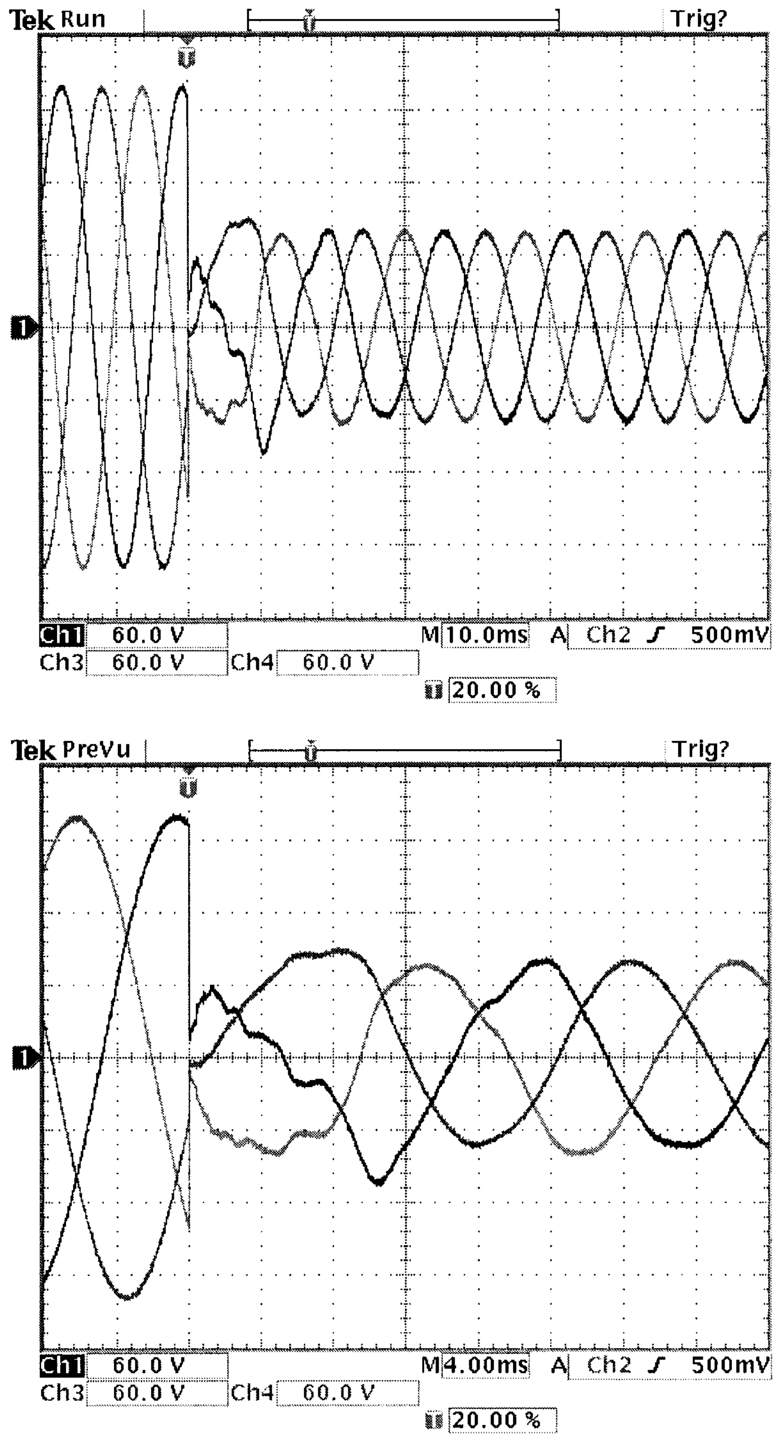


Figure 5.16: AIES Area 50 real-time simulation voltage result (×1k) of balanced fault (three phase)

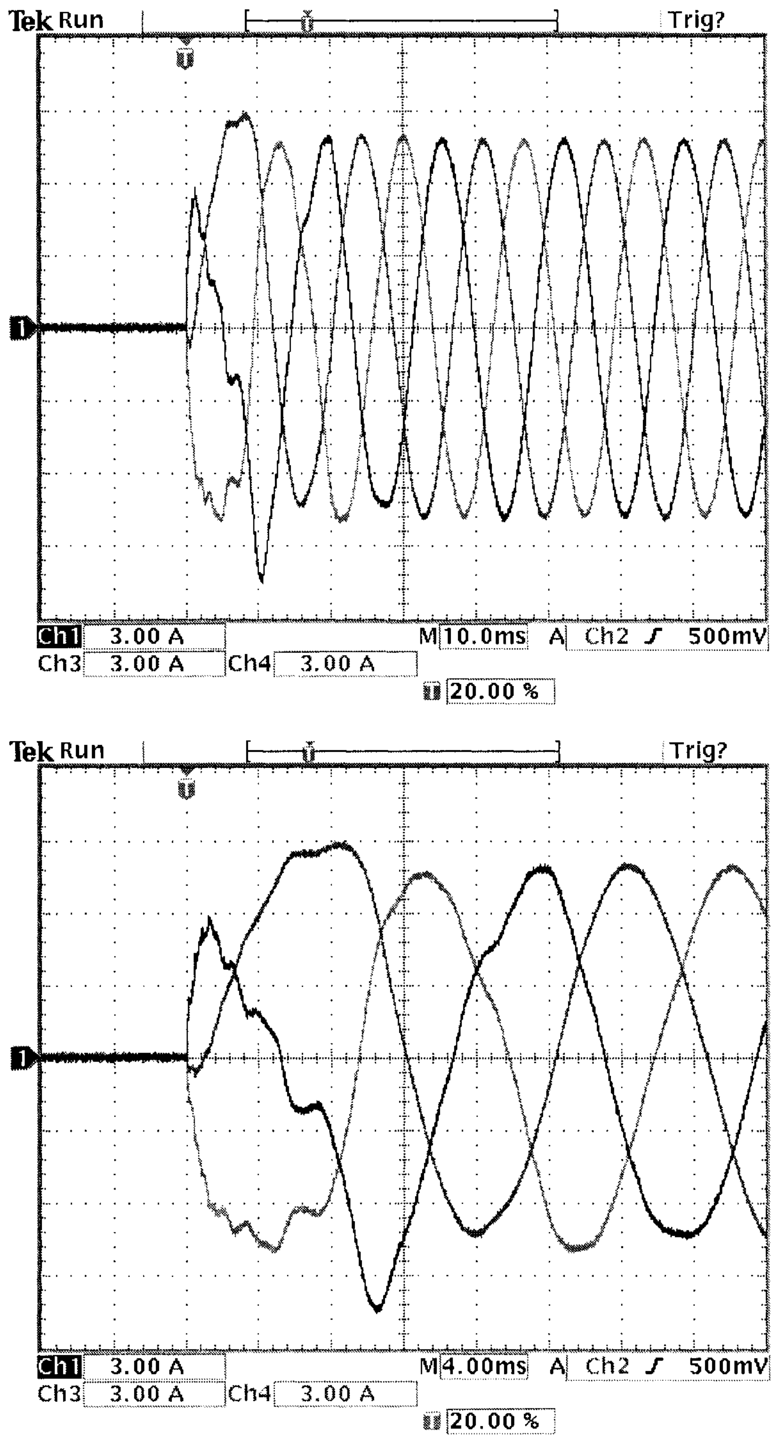


Figure 5.17: AIES Area 50 real-time simulation current result (×1k) of balanced fault (three phase)

6 Conclusions and Future Work

This thesis has proposed a new systematic approach for constructing a TLNE for external systems suitable for electromagnetic transient simulation. With full-order VF, GAs, CLLSQ optimization, and accurate low-order line parameter fitting routine, the generated low-order model is of high accuracy compared to its full model in frequency domain. The merits of this method are its robustness in terms of stability and passivity, its accuracy in not only transient frequencies but also at DC and power frequency, and its optimal deep region order determination feature.

Three detailed systems including a realistic system — the Alberta Interconnected Electric System (AIES) have been model by Robust TLNE and simulated in ATP for verification. Time-domain simulation results with respect to the original system in ATP illustrate the accuracy and computational efficiency of the proposed approach.

Real-time implementation of Robust TLNE models of AIES in RTX-LAB real-time digital simulator is carried out via MATLAB/Simulink C++ S-function. The S-function utilizes the merits of C++ language to ensure high efficiency, scalability and portability of the program. Low time-step size is achieved during real-time simulation with identical results compared to off-line simulation.

The main contributions of this thesis can be summarized as follows:

1. In the modeling of Robust TLNE, this thesis provides that

- Application of real-time Marti's frequency-dependent line model.
- A common set of poles based low-order deep region FDNE model.
- Application of Genetic Algorithms (GAs) with constraints in obtaining first approximations.
- Compensations in GAs for increased accuracies.
- A numerically stable and more efficient CLLSQ optimization algorithm.
- Ensuring accuracies of frequencies at DC of the obtained equivalent.
- Optimal deep region order determination.
- The model with optimal order, for both surface layer and deep region.
- Procedures to construct Robust TLNE model.

2. In constructing an EMTP model for AIES, this thesis contributes

   - An accurate EMTP model for AIES Area 50 Backbone in which all major 240kV transmission lines are represented with frequency dependence.
   - Transferring PSS/E power flow data along with Transmission Alberta System Models (TASMo) database into EMTP model.
   - Procedures to build Robust TLNE for large power systems.

3. In real-time digital simulation, the thesis gives

   - A MATLAB/SIMULINK C++ S-function based EMTP program with the capability of reading ATP data files.
   - Implementation of Marti's frequency dependent line model as well as FDNE including $RLCG$ branch in C++.
   - Implementation of $RLCG$ branch in EMTP for increased computational efficiency.
   - A collection of C++ class libraries for EMTP.
   - A collection of C++ libraries capable of reading ATP data files.
   - A MATLAB/SIMULINK approach in solving nodal equations of EMTP.

- Real-time digital simulation of large power systems implemented in RTX-LAB real-time simulator.
- A practical example to analyze electromagnetic transients in AIES.

The following topics are proposed for future work:

- Built a detailed EMTP model for AIES and implement the model in the RTX-LAB real-time simulator. Existing utilities such as SEQ2ATP for ATP and E-Tran for PSCAD/EMTDC are capable of converting data of IEEE standard power-flow format to corresponding EMTP packages. This facilitates the implementation of the whole AIES in EMTP.

- Include considerations on un-transposed transmission lines in surface layer. The study zone requires accurate representation of electrical elements in power systems. If more accurate line models are taken into account in surface layer, the surface layer is a part of the study zone and the study zone can be smaller.

- Enforce passivity on passive Robust TLNE model in the entire frequency range. In the proposed Robust TLNE, passivity criterion is only validated in limited frequency range and discrete frequency points during model generation, which is still potential stability issue for TLNE model.

- Initiate more comprehensive transient studies on AIES based on Robust TLNE model. Only a phase to ground fault is studies in this thesis. More transient studies can be done with the existing Robust TLNE model for AIES.

- Integrate direct support for three-phase systems in the EMTP S-function program. Transient studies are only limited to single-phase systems in the program. Three-phase systems are only supported via conversion to single-phase systems. Therefore, it is necessary to implement the support for generic transient studies of three-phase systems.

- Implement the Marti's frequency-dependent line model in MATLAB/SIMULINK. The Robust TLNE model can be directly incorporated into SIMULINK with State-Space solutions if the frequency-dependent line model is available in SIMULINK.

- Incorporate more network elements and functionalities into the C++ EMTP S-function program to support models such as nonlinear devices, power electronics, induction machines and control systems.

- Realize distributed and parallel computation for the EMTP in S-function program. Successful achievements has been noticed in partitioning power system networks for parallel simulation in EMTP [34]. This is done via transmission line models.

Bibliography

[1] H. W. Dommel, "Digital computer solution of electromagnetic transients in single and multiphase networks", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-88, No. 4, April 1969, pp. 388-399.

[2] H. W. Dommel, "Nonlinear and time-varying elements in digital simulation of electromagnetic transients", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-90, No. 6, November/December 1971, pp. 2561-2567.

[3] J. R. Marti, J. Lin, "Suppression of numerical oscillations in the EMTP", *IEEE Trans. on Power Systems,* Vol. 4, No. 2, May 1989, pp. 739-747.

[4] CIGRÉ WG13-05 III, "Transmission line representation of energization and re-energization studies for complex feeding networks", *Electra,* Vol. 62, January 1979, pp. 45-78.

[5] A. Semlyen, M. R. Iravani, "Frequency domain modeling of external systems in an electromagnetic transients program", *IEEE Trans. on Power Systems,* Vol. 8, No. 2, May 1993, pp. 527-533.

[6] M. Kizilcay, "Low-order network equivalents for electromagnetic transients studies", *European Trans. on Electrical Power Engineering,* Vol. 3, No. 2, March/April 1993, pp. 123-129.

[7] M. Kizilcay, "Computation of switching transients using low-order multi-port network equivalents", *Proc. of IPST'97,* Seattle, WA., June 1997, pp. 125-130.

[8] W. C. Boaventura, A. Semlyen, M. R. Iravani, A. Lopes, "Robust sparse network equivalent for large systems: part I-methodology", *IEEE Trans. on Power Systems,* Vol. 19, No. 1, Feb. 2004, pp. 157 - 163.

[9] W. C. Boaventura, A. Semlyen, M. R. Iravani, A. Lopes, "Robust sparse network equivalent for large systems: part II-performance evaluation", *IEEE Trans. on Power Systems,* Vol. 19, No. 1, Feb. 2004, pp. 157 - 163.

[10] A. S. Morched, V. Brandwajn, "Transmission network equivalents for electromagnetic transients studies", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-102, No. 9, September 1983, pp. 2984-2994.

[11] A. S. Morched, J. H. Ottevangers, L. Marti, "Multi-port frequency dependent network equivalents for the EMTP", *IEEE Trans. on Power Delivery,* Vol 8, No. 3, July 1993, pp. 1402-1412.

[12] V. Q. Do, M. M. Gavrilovic, "An iterative pole removal method for synthesis of power systems equivalent networks", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-103, No. 8, August 1984, pp. 2065-2070.

[13] V. Q. Do, M. M. Gavrilovic, "A synthesis method for one-port and multiport equivalent networks for analysis of power systems transient", *IEEE Trans. on Power Systems,* Vol. PWRD-1, No. 2, April 1986, pp. 103-113.

[14] N. R. Watson, J. Arrillaga, "Frequency-dependent AC system equivalents for harmonic and transient converter simulation", *IEEE Trans. on Power Delivery,* Vol. 3, No. 3, July 1988, pp. 1196-1202.

[15] A. Abur, H. Singh, "Time domain modeling of external systems for electromagnetic transients programs", *IEEE Trans. on Power Systems,* Vol. 8, No. 2, May 1993, pp. 671-677.

[16] H. Singh, A. Abur, "Multi-port equivalencing of external systems for simulation of switching transients", *IEEE Trans. on Power Delivery,* Vol. 10, No. 1, January 1995, pp. 374-380.

[17] B. Gustavsen, A. Semlyen, "Rational approximation of frequency domain responses by vector fitting", *IEEE Trans. on Power Delivery,* Vol. 14, No. 3, July 1999, pp. 1052-1061.

[18] B. Gustavsen, A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions", *IEEE Trans. on Power Systems,* Vol. 16, No. 1, February 2001, pp. 97-104.

[19] B. Gustavsen, "Computer code for rational approximation of frequency dependent admittance matrices", *IEEE Trans. on Power Delivery,* Vol. 17, No. 4, October 2002, pp. 1093-1098.

[20] B. Gustavsen, A. Semlyen, "A robust approach for system identification in the frequency domain", *IEEE Trans. on Power Delivery,* Accepted for future publication.

[21] M. Abdel-Rahman, A. Semlyen, M. R. Iravani, "Two-layer network equivalent for electromagnetic transients", *IEEE Trans. on Power Delivery,* Vol. 18, No. 4, October 2003, pp. 1328-1335.

[22] M. Abdel-Rahman, *Frequency Dependent Hybrid Equivalents of Large Networks,* Ph.D. Thesis, University of Toronto, 2001.

[23] X. Nie, V. Dinavahi, "A robust two-layer network equivalent for transient studies", *Proc. of IPST'05,* Montréal, Canada, June 19-23 2005.

[24] A. Semlyen, A. Dabuleanu, "Fast and accurate switching transient calculations on transmission lines with ground return using recursive convolutions", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-94, No. 2, March/April 1975, pp. 561-571.

[25] J. R. Marti, "Accurate modeling of fequency-dependent transmission lines in electromagnetic transient simulations", *IEEE Trans. on Power Apparatus and Systems,* Vol. PAS-101, No. 1, January 1982, pp. 147-155.

[26] F. Castellanos, J. R. Marti, "Full frequency-dependent phase-domain transmission line model", *IEEE Trans. on Power Systems,* Vol. 12, No. 3, August 1997, pp. 1331-1339.

[27] T. Noda, N. Nagaoka, A. Ametani, "Phase domain modeling of frequency-dependent transmission lines by means of an ARMA model", *IEEE Trans. on Power Delivery,* Vol. 11, No. 1, January 1996, pp. 401-411.

[28] T. Noda, N. Nagaoka, A. Ametani, "Further improvements to a phase domain ARMA line model in terms of convolution, steady-state initialization and stability", *IEEE Trans. on Power Delivery,* Vol. 12, No. 3, July 1997, pp. 1327-1332.

[29] A. B. Fernandes, W. L. A. Neves, "Phase-domain transmission line models considering frequency-dependent transformation matrices", *IEEE Trans. on Power Delivery,* Vol. 19, No. 2, April 2004, pp. 708-714.

[30] A. Semlyen, M. H. Abdel-Rahman, "A state variable approach for the calculation of switching transients on a power transmission line", *IEEE Trans. on Circuits and Systems,* Vol. CAS-29, No. 9, September 1982, pp. 624-633.

[31] B. Gustavsen, A. Semlyen, "Simulation of transmission line transients using vector fitting and model decomposition", *IEEE Trans. on Power Delivery,* Vol. 13, No. 2, April 1998, pp. 605-614.

[32] B. Gustavsen, A. Semlyen, "Combined phase and model domain calculation of transmission line transients based on vector fitting", *IEEE Trans. on Power Delivery,* Vol. 13, No. 2, April 1998, pp. 596-604.

[33] L. Pak, M. O. Faruque, X. Nie, V. Dinavahi, "A versatile cluster-based real-time digital simulator for power engineering research", *Submitted to IEEE Trans. on Power Systems.* July 2005.

[34] J. R. Marti, L. R. Linares, "Real-time EMTP-based transients simulation", *IEEE Trans. on Power Systems,* Vol. 9, No. 3, August 1994, pp. 1309-1317.

[35] A. Semlyen, F. de León, "Computation of electromagnetic transients using dual or multiple time steps", *IEEE Trans. on Power Systems,* Vol. 8, No. 3, August 1993, pp. 1274-1281.

[36] R. M. Mathur, X. Wang, "Real-time digital simulator of the electromagnetic transients of power transmission lines", *IEEE Trans. on Power Delivery,* Vol. 4, No. 2, April 1989, pp. 1275-1280.

[37] L. Marti, "Low-order approximation of transmission line parameters for frequency-dependent models", *IEEE Trans. on Power Apparatus and Systems*, Vol. PAS-102, No. 11, November 1983, pp. 3582-3589.

[38] X. Wang, R. M. Mathur, "Real-time digital simulator of the electromagnetic transients of transmission lines with frequency dependence", *IEEE Trans. on Power Delivery*, Vol. 4, No. 4, October 1989, pp. 2249-2255.

[39] N. Zhang, X. Wang, J. F. Eggleston, R. M. Mathur, "Improvements in the realization of a real-time digital simualtor of a power transmission line", *IEE International Conference on AC and DC Power Transmission*, London, U. K., September 17-20 1991, pp. 356-361.

[40] C. Dufour, H. Le-Huy, J. Soumagne, A. E. Hakimi, "Real-time simulation of power transmission lines using Marti model with optimal fitting on dual-DSP card", *IEEE Trans. on Power Delivery*, Vol. 11, No. 1, January 1996, pp. 412-419.

[41] X. Wang, D. A. Woodford, R. Kuffel, R. Wierckx, "A real-time transmission lines model for a digital TNA", *IEEE Trans. on Power Delivery*, Vol. 11, No. 2, April 1996, pp. 1092-1097.

[42] A. B. Fernandes, W. L. A. Neves, "Transmission lines: fitting technique optimization", *Proc. of IPST'97*, Seattle, U. S. A., June 22-26 1997.

[43] A. B. Fernandes, W. L. A. Neves, "Frequency-dependent low order approximation of transmission line parameters", *Proc. of IPST'99*, Budapest, Hungary, June 20-24 1999.

[44] Canadian/American EMTP user group, *Alternative Transient Program Rule Book*, 1999.

[45] W. Scott-Meyer, *EMTP Theory Book*, Bonneville Power Administration, 1984.

[46] M. Kizilcay, "A new branch in the ATP-EMTP high-order, linear admittance model", *ATP News*, 1993.

[47] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, January 1989.

[48] A. Chipperfield, P. Fleming, H. Pohlheim, C. Fonseca, *Genetic Algorithm Toolbox for MATLAB,* Department of Automatic Control and Systems Engineering, Univerisity of Sheffield.

[49] *MATLAB User Guides* The MathWorks Inc., Natick, MA..

[50] P. Kundur, *Power System Stability and Control,* McGraw-Hill, 1994.

[51] J. Duncan Glover, Mulukutla S. Sarma., *Power System Analysis and Design (3rd Edition)* Wadsworth/Thomson Learning, 2002.

[52] B. Eckel, *Thinking in C++, Volume 1: Introduction to Standard C++ (2nd Edition),* Prentice Hall, 2000.

[53] B. Eckel, C. Allison, *Thinking in C++, Volume 2: Practical Programming (2nd Edition),* Prentice Hall, 2003.

[54] *Matrix TCL Reference Manual,* Techsoft Pvt. Ltd..

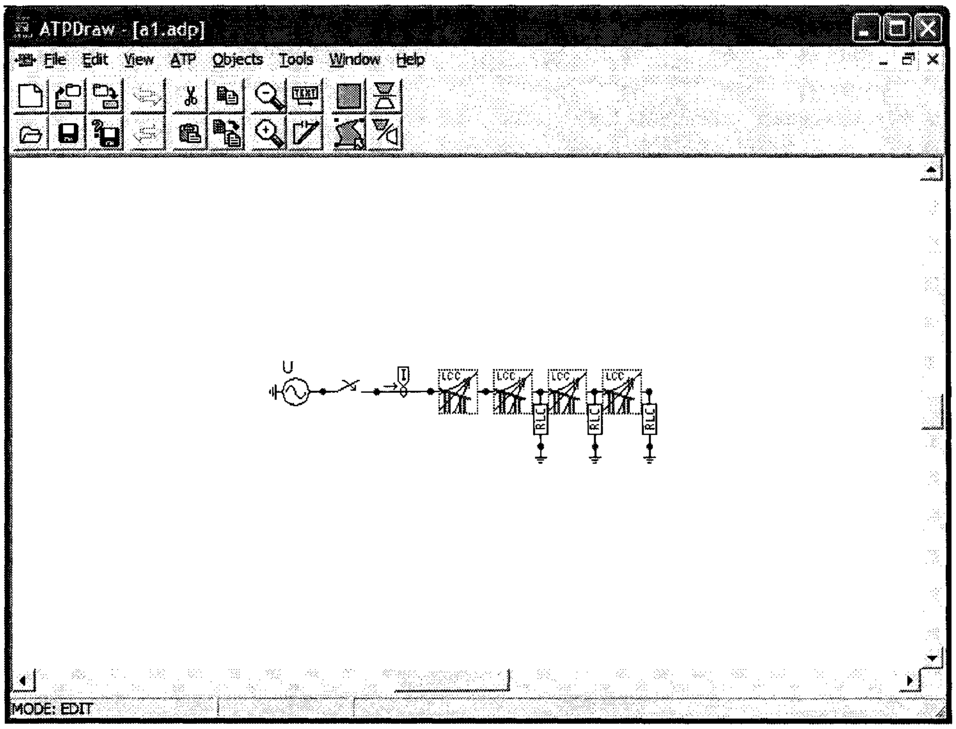
A

Example 1

A.1 System Parameters

Elements	Parameters
Source	60Hz, $10\angle 0^{\circ}$kV
TL1	280km transmission line
TL2	400km transmission line
TL3	220km transmission line
TL4	80km transmission line
Conductor	Two-bundle Drake conductor with 50cm distance
Tower	Height: 15m, ground resistivity: 100Ωm, homogeneous earth assumed
Load1	*RL* Load: 100Ω, 150mH
Load2	*RL* Load: 60Ω, 80mH
Load3	*RL* Load: 120Ω, 100mH

A.2 System Diagram in ATPDraw



A.3 ATP Data File for Full Model

```
BEGIN NEW DATA CASE
C -----------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C June 22, 2004
C -----------------------------------------------------------
POWER FREQUENCY                        60.
$DUMMY, XYZ000
C  dT   >< Tmax >< Xopt >< Copt >
   2.E-5      .2
     500       1       0       0       0       0       0       1       0
C         1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
C Load 1
        XX0006                60.   80.                                        0
C Load 2
        XX0004               100.  150.                                        0
C Load 3
        XX0012               120.  100.                                        0
C TL1
-1A____1XX0002                 2.  0.00                -2 1
      18       3.43904357882401600000E+02
   2.75748917329525200E+02   2.09556456539013700E+03  -3.02120107577198200E+02
   1.32522038307429600E+02   5.34206778326990500E+02   7.77786264529477200E+02
   1.49408699070299500E+03   7.42950753414815600E+03   3.11676509575282100E+04
   1.27236036995146400E+05   5.08023732184690700E+05   2.32321224243896200E+06
   5.09434464995741600E+06   3.65432987658662500E+06   2.66960449636756200E+06
   2.78771199529035400E+06   2.85557150987219100E+06   3.28340044978634400E+06
   1.55157367465512800E+00   3.13344553898543100E+00   3.37321729335057800E+00
   4.11467825483759500E+00   8.24190385834596400E+00   1.44386945071827500E+01
   4.50761403850135000E+01   3.97846218782673400E+02   1.72512452765138800E+03
   7.33827995192348500E+03   3.07002204634114500E+04   1.46775367026548200E+05
   6.61870409143599800E+05   1.37066806499039800E+06   1.35751400884485100E+06
   3.13588034329753700E+06   2.94525940129055000E+06   3.70207919969885800E+06
      16       9.92561082855090700000E-04
   1.21611384045855800E-02   3.17820537468208000E-02   1.63475935146936700E-01
   4.50277411304402400E-01   6.24587870467199700E-01   9.62547231237992800E-01
   2.11492778892177200E+00   1.76164573866439500E+01   9.33425314822263600E+01
   5.94613595758890300E+02   1.64245992775248200E+03   2.02960381955791800E+03
   1.56675491715680900E+04   3.56471480032259900E+04   1.59460983021337400E+07
  -1.60017949953998000E+07
   4.47470371651519400E+00   1.17382684613682300E+01   5.99242290097617800E+01
   1.59436483822033800E+02   2.23499770666861100E+02   3.22868571196113200E+02
   3.70637016742645200E+02   7.30820641779627400E+02   1.71955068882627000E+03
   3.96286728577050700E+03   6.25989363073368600E+03   9.88370958996017500E+03
   1.95251223260011500E+04   4.43329805258528100E+04   2.98053414854056900E+04
   2.98351468268911200E+04
  1.00000000
  0.00000000
C TL2
-1XX0002XX0004                 2.  0.00                -2 1
      18       3.43904357882401600000E+02
   2.75748917329525200E+02   2.09556456539013700E+03  -3.02120107577198200E+02
   1.32522038307429600E+02   5.34206778326990500E+02   7.77786264529477200E+02
   1.49408699070299500E+03   7.42950753414815600E+03   3.11676509575282100E+04
   1.27236036995146400E+05   5.08023732184690700E+05   2.32321224243896200E+06
   5.09434464995741600E+06   3.65432987658662500E+06   2.66960449636756200E+06
   2.78771199529035400E+06   2.85557150987219100E+06   3.28340044978634400E+06
   1.55157367465512800E+00   3.13344553898543100E+00   3.37321729335057800E+00
   4.11467825483759500E+00   8.24190385834596400E+00   1.44386945071827500E+01
   4.50761403850135000E+01   3.97846218782673400E+02   1.72512452765138800E+03
   7.33827995192348500E+03   3.07002204634114500E+04   1.46775367026548200E+05
   6.61870409143599800E+05   1.37066806499039800E+06   1.35751400884485100E+06
   3.13588034329753700E+06   2.94525940129055000E+06   3.70207919969885800E+06
      12       1.43447271640600000000E-03
   2.24741581214468300E-02   6.14193666189440300E-01   1.73631140036276900E-01
  -1.88996292345958600E+00   1.05136587209149200E+01   1.79732027306505200E+02
   2.80020207640768800E+02   6.47272395784606100E+02   5.42303119579675400E+03
   4.13756337182865700E+05   4.26608490676978000E+07  -4.30811448947022200E+07
   4.40522127150029800E+00   1.16980102042384100E+02   3.42279960201415600E+01
   3.24436046529355400E+02   3.21222961401006300E+02   2.02563796805317100E+03
   2.44012031826036300E+03   3.54015992949590200E+03   8.66028848648272500E+03
   2.30461334935895600E+04   2.09372081176427900E+04   2.09581453257604500E+04
  1.00000000
  0.00000000
C TL3
-1XX0004XX0006                 2.  0.00                -2 1
      18       3.43904357882401600000E+02
   2.75748917329525200E+02   2.09556456539013700E+03  -3.02120107577198200E+02
   1.32522038307429600E+02   5.34206778326990500E+02   7.77786264529477200E+02
   1.49408699070299500E+03   7.42950753414815600E+03   3.11676509575282100E+04
   1.27236036995146400E+05   5.08023732184690700E+05   2.32321224243896200E+06
   5.09434464995741600E+06   3.65432987658662500E+06   2.66960449636756200E+06
   2.78771199529035400E+06   2.85557150987219100E+06   3.28340044978634400E+06
   1.55157367465512800E+00   3.13344553898543100E+00   3.37321729335057800E+00
   4.11467825483759500E+00   8.24190385834596400E+00   1.44386945071827500E+01
   4.50761403850135000E+01   3.97846218782673400E+02   1.72512452765138800E+03
   7.33827995192348500E+03   3.07002204634114500E+04   1.46775367026548200E+05
   6.61870409143599800E+05   1.37066806499039800E+06   1.35751400884485100E+06
   3.13588034329753700E+06   2.94525940129055000E+06   3.70207919969885800E+06
      12       7.75205848884722700000E-04
   5.16575344691927400E-02   1.53232837264703700E+00   2.37724925263830500E+00
   3.30486779211416800E+00   1.36818812823423600E+01   1.62481417148063300E+02
  -8.80925175507769900E+02   2.64168245174528800E+03   1.05532018711274100E+04
   4.46227461788609200E+06   1.84254666467924900E+08  -1.88729438474360000E+08
   9.86927346002103800E+00   2.75447572089606500E+02   4.20801419765296900E+02
   5.40238212815512500E+02   6.41977109380031700E+02   2.73672970771993400E+03
   6.16757885803951600E+03   5.72202239134570900E+03   1.66032197737507600E+04
   4.27963575320475200E+04   4.10830083290002900E+04   4.11240913373293300E+04
  1.00000000
  0.00000000
C TL4
-1XX0006XX0012                 2.  0.00                -2 1
      18       3.43904357882401600000E+02
   2.75748917329525200E+02   2.09556456539013700E+03  -3.02120107577198200E+02
   1.32522038307429600E+02   5.34206778326990500E+02   7.77786264529477200E+02
   1.49408699070299500E+03   7.42950753414815600E+03   3.11676509575282100E+04
   1.27236036995146400E+05   5.08023732184690700E+05   2.32321224243896200E+06
   5.09434464995741600E+06   3.65432987658662500E+06   2.66960449636756200E+06
   2.78771199529035400E+06   2.85557150987219100E+06   3.28340044978634400E+06
   1.55157367465512800E+00   3.13344553898543100E+00   3.37321729335057800E+00
   4.11467825483759500E+00   8.24190385834596400E+00   1.44386945071827500E+01
   4.50761403850135000E+01   3.97846218782673400E+02   1.72512452765138800E+03
   7.33827995192348500E+03   3.07002204634114500E+04   1.46775367026548200E+05
   6.61870409143599800E+05   1.37066806499039800E+06   1.35751400884485100E+06
   3.13588034329753700E+06   2.94525940129055000E+06   3.70207919969885800E+06
      27       2.75716715953531900000E-04
   7.41855023793343800E-03   6.89364462199310900E-02   3.25985424621389200E-01
   4.37349506980829100E-01   5.33890733252180900E-01   6.63903346282451200E-01
   7.27352105811201600E-01   9.03208844048479000E-01   9.50187765078895200E-01
   1.25245027938036600E+00   3.52881270255085800E+00   1.26610600550988400E+01
   6.21970156366083600E+01   3.04227613082559300E+02   6.05579666456830200E+02
   1.13692588827181600E+03   2.18134235607163100E+03   2.55204251163790000E+03
   1.17699737622563400E+04  -6.30737653062193900E+04   9.35305340262357300E+04
   1.80726010396839700E+06  -1.83885262811079000E+06   8.69613895477062600E+05
  -8.57336616619472200E+05   8.19093664541233600E+05  -8.48869537345589800E+05
   5.83002815611244100E+00   5.40430558878495500E+01   2.51143324984160300E+02
   3.34941363004325500E+02   4.12846528154363100E+02   5.04528748011672400E+02
   5.79140621318636800E+02   6.59481393962392100E+02   7.64831178018867800E+02
   9.32370343345793200E+02   1.28049005297752600E+03   2.13574579051440000E+03
   2.76751077409838300E+03   6.01981074582288400E+03   1.05351913699554100E+04
   1.53693948316460300E+04   1.91191202315414500E+04   2.54555964205854500E+04
   4.10060803429461300E+04   9.20941067880203900E+04   7.30370500245080900E+04
   1.47396919251189900E+05   1.47544316170440800E+05   3.67181057879015600E+05
   3.67548238936894000E+05   1.81744201094532400E+05   1.81925945295626900E+05
  1.00000000
  0.00000000
/SWITCH
```

```
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
  XX0015XX0017       .05        .5                                      1
  XX0017A____1                                       MEASURING          1
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
C Ideal voltage source
14XX0015 0       10.       60.                                  -1.        1.
/OUTPUT
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

A.4 ATP Data File for Robust TLNE Model

```
BEGIN NEW DATA CASE
C -----------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C June 20, 2004
C -----------------------------------------------------------
POWER FREQUENCY                         60.
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   1.E-5      .2
     100       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C Surface layer
C TL1 (reduced-order, after CLLSQ optimization)
-1IN___1A____1                    2.  0.00                 -2 1
       3         3.864633776250925100E+002
   1.314702435456360lE+005   1.113044259871725 8E+004   7.146643589811817 5E+002
   3.087007541852243 8E+003   9.400320958020543 6E+000   2.679942459277918 9E+000
       1         1.022400031718859000E-003
    6.117460669092771 2E+003
    6.161905839764758 6E+003
   1.00000000
   0.00000000
C Deep region
C Low-order FDNE model
C BEGIN FDNE
$VINTAGE,1
C <BUS1><BUS2><BUS3><BUS4><   OHM         ><   milliH      ><   microF      >
C
C (1,1)
  A____1                       3.283409e+002
  A____1                       5.773775e+001   1.814903e+003
  A____1                       2.270264e+002   3.735426e+003
  A____1                       9.378893e+002   3.250459e+003
  A____1                       6.449830e+003   7.655092e+003
  A____1                      -3.774552e+002  -6.912906e+001
  A____1A 6__1                 1.923339e+002   5.019495e+002
  A 6__1                      -1.770025e+004
  A 6__1                                                       8.039527e-001
  A____1A 8__1                 2.399467e+002   1.253622e+003
  A 8__1                       3.599651e+004
  A 8__1                                                       1.305689e-001
  A____1A A__1                 1.171002e+003   6.912165e+002
  A A__1                      -8.148504e+003
  A A__1                                                       9.737904e-002
  A____1A C__1                 2.611214e+003   1.069758e+003
  A C__1                      -2.018949e+004
  A C__1                                                       2.721165e-002
$VINTAGE,0
C END FDNE
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
  XX0015IN___1       .05        .5                                      1
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
14XX0015 0       10.       60.                                  -1.        1.
/OUTPUT
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK INITIAL
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

B
Example 2

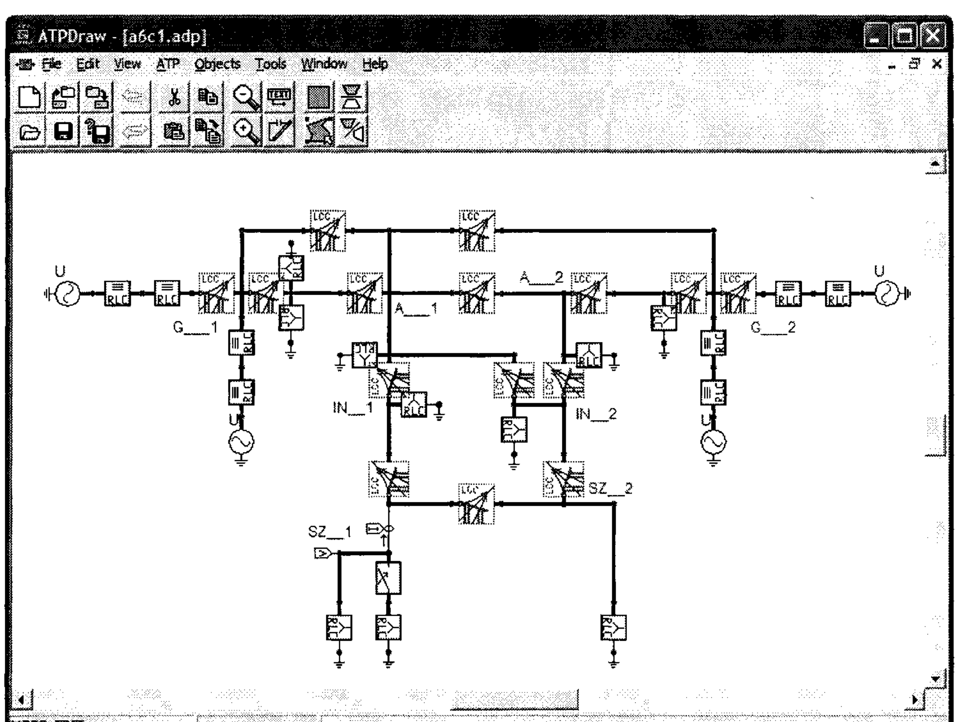
B.1 System Parameters

Elements	Parameters
TLS1	120km transmission line, one Hawk conductor per phase
TLS2	136km transmission line, one Drake conductor per phase
TLS3	165km transmission line, one Hawk conductor per phase
TL1	150km transmission line, one Drake conductor per phase
TL2	120km transmission line, one Drake conductor per phase
TL3	400km transmission line, one Hawk conductor per phase
TL4	220km transmission line, one Hawk conductor per phase
TL5	35km transmission line, one Hawk conductor per phase
TL6	10km transmission line, one Hawk conductor per phase
TL7	35km transmission line, one Hawk conductor per phase
TL8	15km transmission line, one Hawk conductor per phase
TL9	65km transmission line, one Hawk conductor per phase
TL10	133km transmission line, one Hawk conductor per phase
TL11	42km transmission line, one Hawk conductor per phase
TL12	375km transmission line, one Hawk conductor per phase
Tower	Conductors: horizontal -6, 0, 6m, vertical 15, 24, 15m; ground wires: horizontal -3.932, 3.932m, vertical 30, 30m, ground resistivity: 100Ωm
Load1	*RL* Load: 1200Ω, 500mH per phase
Load2	*RL* Load: 2150Ω, 380mH per phase
Load3	*RL* Load: 250Ω, 25mH per phase

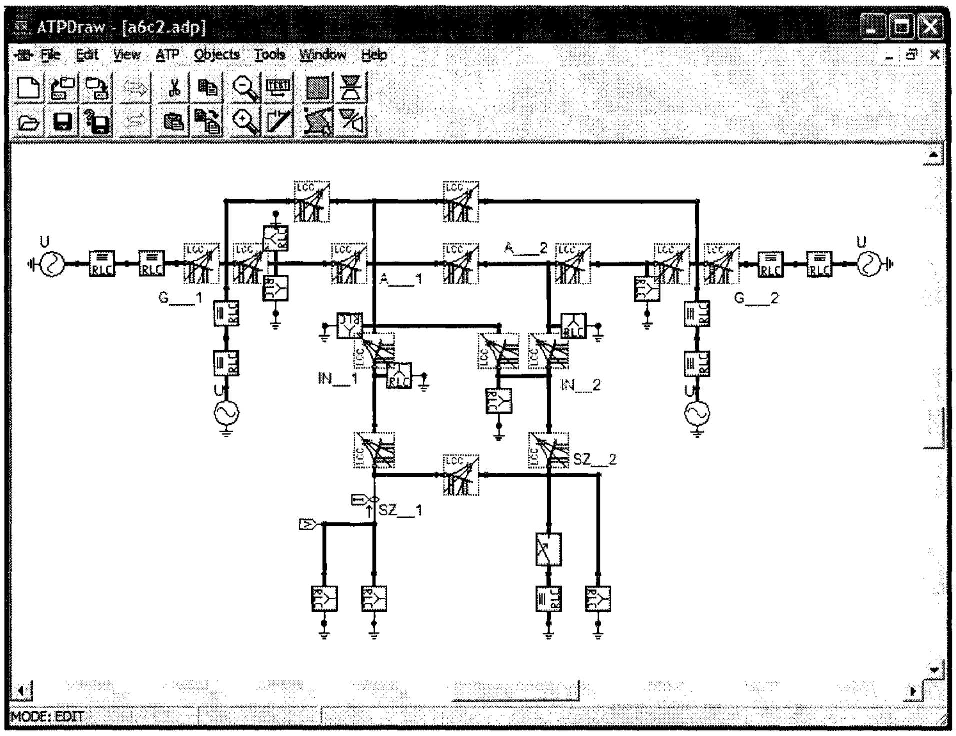
Load4	RL Load: 350Ω, 60mH per phase
Load5	RL Load: 250Ω, 25mH per phase
Load6	RL Load: 420Ω, 30mH per phase
Load7	RL Load: 200Ω, 130mH per phase
Load8	RL Load: 650Ω, 250mH per Phase
C1	Capacitor bank 5μF per phase
C2	Capacitor bank 20μF per phase
G1	Generator: 1.03∠20.2°, Z_{G1}: 1.2Ω, 38.98mH per phase
G2	Generator: 1.01∠10.5°, Z_{G2}: 1.1Ω, 45.52mH per phase
G3	Generator: 1.03∠-6.8°, Z_{G3}: 0.9Ω, 38.98mH per phase
G4	Generator: 1.01∠-17.0°, Z_{G4}: 0.8Ω, 35.23mH per phase
T1	Transformer: Z_{T1}: 1.5Ω, 23.4mH per phase
T2	Transformer: Z_{T2}: 0.8Ω, 29.5mH per phase
T3	Transformer: Z_{T3}: 1.6Ω, 23.4mH per phase
T4	Transformer: Z_{T4}: 0.6Ω, 20.8mH per phase

B.2 System Diagram in ATPDraw

B.2.1 Capacitor $C1$ Switching Case



B.2.2 Balanced Three-Phase to Ground Fault Case



B.3 ATP Data File for Full Model

B.3.1 Capacitor $C1$ Switching Case

```
BEGIN NEW DATA CASE
C ---------------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C September 10, 2004
C ---------------------------------------------------------------
POWER FREQUENCY                                 60.
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   2.E-5    .15
       1       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
  SZ__1A                    200.  130.                                         0
  X0007AX0009A               1.1 45.52                                         0
  X0010AX0012A               1.2 38.98                                         0
  X0009AX0046A                .8  29.5                                         0
  X0012AG___1A               1.5  23.4                                         0
  X0017A                                  8.                                   0
  SZ__2A                    650.  250.                                         0
  X0030AX0032A                .8 35.23                                         0
  X0032AX0049A                .6  20.8                                         0
  X0036AX0035A                .9 38.98                                         0
  G___2AX0036A               1.6  23.4                                         0
  X0042A                    420.   30.                                         0
  X0044A                    250.   25.                                         0
  X0044A                                 20.                                   0
  IN__1A                   1200.  500.                                         0
  IN__2A                   2150.  380.                                         0
  A___1A                    250.   25.                                         0
  A___2A                    350.   60.                                         0
C TLS1 HAWK conductor
-1SZTMPAIN__1A                2.  0.00                -2
     12      4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     19      4.00519444461175000000E-04
   1.56976974488308500E-02   5.96931801697355600E-02   7.65710632981902200E-02
   1.23025310978241300E-01   7.74362618176230500E+00   1.15631326792783200E+01
   1.51793395253545000E+01   2.45460099809200000E+01   2.17562524566291500E+02
   4.75157557302470000E+03   7.17277231675696400E+03   2.15420855113000800E+02
   1.92780440925090000E+05   1.00130204456972800E+05   6.67694141945910300E+05
   4.49850806862179600E+09  -4.59220311195214400E+09   3.24980116228040600E+09
  -3.15707914037557500E+09
   5.46006799865154000E+00   2.06382999204778400E+01   2.69410363626236900E+01
   4.36617508325875400E+01   2.72889153410192200E+03   3.96969851665451300E+03
   5.15623359578923200E+03   1.03865827369060600E+04   9.26732459557961600E+03
   1.12469458616561000E+05   6.93234200200993500E+04   4.98920839522332300E+05
   4.15754453085636900E+05   7.76422380585483400E+05   1.43944837446228700E+06
   4.02521957768449700E+06   4.02924479726218400E+06   4.38129540251903900E+06
   4.38567669792155100E+06
  1.00000000
  0.00000000
C TLS2 DRAKE conductor
-1SZTMPASZ__2A                2.  0.00                -2
     11      3.93897327703557300000E+02
   2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
   9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
   6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
   5.35709730600259600E+04   2.76198969356064600E+06
   2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
   2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
   8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
   2.24249725179149800E+04   1.15971094692565100E+06
     19      4.53915497928795800000E-04
   1.71275917005395300E-02   3.98691337390333500E-02   1.60864260029983000E-01
   6.89003618290678800E+00   8.78632925717092800E+00   1.18708400084055100E+01
   1.46650428128307100E+01   4.55132169053748200E+01   1.92681981165319200E+02
   2.21326495567518500E+03   1.11000476132130400E+04  -9.21432832870185500E+04
   2.81776544230162200E+05   1.67841941653263700E+04   1.22535036976167300E+06
  -3.16876240455654700E+07   2.93464632821155900E+07   3.77888719163484300E+07
  -3.68930729156446700E+07
   6.29920182790418300E+00   1.47535442461695500E+01   5.98892062931638800E+01
   2.46878104720658700E+03   3.24688225671564200E+03   4.20547568033697400E+03
   5.30166238127239600E+03   1.66566947044242800E+04   8.86571988667755600E+03
   4.65746128476995800E+04   1.13398074824504100E+05   4.43515608473281600E+05
   3.98417841600631700E+05   5.62068679963815700E+05   1.44682872910978600E+06
   2.87151967320063000E+06   2.87439119287383300E+06   3.65427188965252700E+06
   3.65792616154218300E+06
  1.00000000
  0.00000000
C TLS3 HAWK conductor
-1SZ__2AIN__2A                2.  0.00                -2
     12      4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     19      5.50554384535761800000E-04
   9.79465605855859600E-03   4.06212733123785000E-02   7.44854903179096900E-02
   4.00229870144877300E-02   6.56254564933406700E-02   7.79345690027440800E-02
   7.11151466287055800E-01   6.74005236856138300E+00   8.01620767358119800E+00
   1.17220670074905700E+01   3.50297130275519900E+01   7.05494675388751300E+01
   7.13600486817041200E+02   4.43545346148581900E+03   6.11781278301569400E+04
   6.79300571244501800E+04   7.49764722063926500E+05   2.36469459307105400E+07
  -2.45311009688204900E+07
   3.94587715068155200E+00   1.57035953406104900E+01   3.03393666019717600E+01
   1.67066122294504600E+01   2.60870054659298800E+01   3.32492107284800500E+01
   2.91988641610697900E+02   2.66461815651150000E+03   3.31965063922072700E+03
   4.56609698304445300E+03   7.47335672283102600E+03   6.69903629701688300E+03
   1.75902539791774200E+04   5.33686195521447800E+04   2.15476314218399300E+05
   3.13824776054083900E+05   9.37530861396163600E+05   1.82325148034484500E+06
   1.82507473182518700E+06
  1.00000000
  0.00000000
C TL1 DRAKE conductor
-1IN__1AA___1A                2.  0.00                -2
     11      3.93897327703557300000E+02
   2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
   9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
   6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
   5.35709730600259600E+04   2.76198969356064600E+06
   2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
   2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
   8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
   2.24249725179149800E+04   1.15971094692565100E+06
     18      5.00569525809250100000E-04
   1.43902995689640600E-02   4.33458044156565500E-02   5.16186088796879400E-02
   2.60473078409012900E+00   6.36887547988259200E+00   8.77520515952783600E+00
   1.07715571586440900E+01   1.62456122440872200E+01   1.49294354917248900E+02
   3.28434390136963700E+03   5.49853561056992600E+03   5.34533157308277900E+04
   2.31574716012179000E+05   3.27298197639052900E+05   3.14159278130439400E+08
  -3.10595574404512300E+08   9.30706221006033400E+08  -9.34891228010527500E+08
   5.61936471587928700E+00   1.68305760407245200E+01   2.05250926175730500E+01
```

```
    1.01726311455150700E+03   2.52403027449716200E+03   3.28509515605487000E+03
    4.12398597622061600E+03   7.90009211967655900E+03   7.14159016302577300E+03
    8.66107182139554900E+04   5.76718145371582000E+04   2.43513293383362700E+05
    5.26418551705884900E+05   9.59660805850347500E+05   3.87278673075127000E+06
    3.87665951748202500E+06   2.75397849079348300E+06   2.75673246928427900E+06
   1.00000000
   0.00000000
C TL2 DRAKE conductor
-1IN__2AA___2A                 2.  0.00             -2
      11        3.93897327703557300000E+02
    2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
    9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
    6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
    5.35709730600259600E+04   2.76198969356064600E+06
    2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
    2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
    8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
    2.24249725179149800E+04   1.15971094692565100E+06
      20        4.00317427133653900000E-04
    2.08891430395107500E-02   4.32593489018147900E-02   1.15251517569450200E+00
    8.84981542682858600E+00   1.06444954332436300E+01   1.39766096840854400E+01
    1.91363487218951700E+01   6.03314183666120600E+01   3.04976279498481800E+02
    2.56135740054043300E+03   1.65083598049797800E+04   5.70650702060662600E+04
   -1.53112682223131700E+05   3.16998984679543100E+05   4.92484468183444900E+04
    1.19008457608240200E+06   1.68350028281018700E+08  -1.72202925551848900E+08
    1.16734633689862100E+08  -1.14361509663435200E+08
    7.46197475944493800E+00   1.55748584486930600E+01   4.15231927538063100E+02
    3.07835555158135900E+03   3.88583225013656500E+03   4.85587931971029500E+03
    6.73334411727069400E+03   2.14696936317370400E+04   1.34300288947473000E+04
    5.33603599572152600E+04   1.53615723524379400E+05   3.54735233931288400E+05
    5.75236151700719600E+05   5.40507595967298000E+05   7.45893945162507200E+05
    1.68236204302255000E+06   3.62728264518077400E+06   3.63090992782594900E+06
    4.48069217682420800E+06   4.48517286900102500E+06
   1.00000000
   0.00000000
C TL3 HAWK conductor
-1IN__2AA___1A                 2.  0.00             -2
      12        4.09034706490576600000E+02
    2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
    4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
    7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
    1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
    2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
    1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
    7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
    4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
      30        1.33643854753835200000E-03
    9.68976713503962700E-03   1.33120204768106500E-02   2.18568427218494000E-02
    2.19675564248809300E-02   3.38078085190960200E-02   4.33486771049049300E-02
    6.76106246957764400E-02   5.43140411789459800E-02   1.41460389799945100E-01
    9.11002157670505200E-02   8.01180983546494100E-02   1.12453334559639800E-01
    1.38076692919730400E-01   1.82463659623661200E-01   2.64099618447538500E+00
    3.87852196950510400E+00   6.12938827733858700E+00   1.90993810391674400E+01
    7.96538243161341800E+01   1.29056641368434100E+02   9.01650840605030900E+02
    7.99973827313519500E+03   4.45255469141922400E+03   1.42270434787099700E+04
    1.50844403119674400E+04   4.48375328644470000E+05  -8.32700885975779600E+07
    8.26595716973833400E+07   6.59442835173351500E+06  -6.47519367811138600E+06
    3.72635837689990400E+00   5.16894174539967600E+00   8.23817361522780300E+00
    8.84390655470486900E+00   1.32755619142013700E+01   1.68748913128125700E+01
    2.32643778079367300E+01   2.37675495625943900E+01   2.83107374370852200E+01
    3.46158684277817600E+01   3.57418804477292800E+01   4.53482047733419300E+01
    5.77262025912380300E+01   7.59404155027318700E+01   1.04881658190294000E+03
    1.64478990131605700E+03   2.36646385732968900E+03   3.56011487741156300E+03
    5.83982434202618200E+03   6.42072869092559000E+03   1.13892174160352400E+04
    5.35511456962960400E+04   9.56093676362913700E+04   7.29646502751217500E+04
    1.31776060655731200E+05   3.46569378060718900E+05   6.03290569663039200E+05
    6.03893860232702600E+05   7.50635238095463700E+05   7.51385873333557900E+05
   1.00000000
   0.00000000
C TL4 HAWK conductor
-1A___1AA___2A                 2.  0.00             -2
      12        4.09034706490576600000E+02
    2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
    4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
    7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
    1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
    2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
    1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
    7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
    4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
      14        7.34268486806543900000E-04
    3.29705233563703400E-02   1.06110147200249700E-01   1.49699996946430600E-01
    4.54005314668431200E-01   2.29894736051669500E+01   2.00264225444867500E+01
    3.98594237404368400E+01   2.45759513008771500E+03   2.07126101614158800E+03
    3.62588430407428400E+04   8.92966801187376800E+04   8.32467519726971600E+05
   -1.44068517734813900E+08   1.43105882217675000E+08
    5.87892499398299700E+00   1.88886536977137700E+01   2.73362791240876300E+01
    8.32906170850658600E+01   3.44044452144373500E+03   4.07461831915967500E+03
    3.66919194505697000E+03   5.23907878298744000E+04   2.21598491116926500E+04
    1.45187998152569500E+05   2.73030963103455100E+05   7.70543131940325700E+05
    1.29689934755590900E+06   1.29819624690346300E+06
   1.00000000
   0.00000000
C TL5 HAWK conductor
-1X0044AA___1A                 2.  0.00             -2
      12        4.09034706490576600000E+02
    2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
    4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
    7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
    1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
    2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
    1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
    7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
    4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
      24        1.16743539295552900000E-04
    1.40338144245651000E-02   3.39703490032430100E-02   7.69056681810012900E+00
    3.58110106599894800E+01   4.59434851394437100E+01   2.84917356432648400E+02
    3.53388521104483100E+01  -2.01507841110737700E+02   6.39382485292421200E+01
    8.88535598132061800E+01   3.01026729828361200E+03   1.26069665984102700E+04
    6.12738155584614600E+04   2.34189902042343900E+05   1.31935990946075200E+05
    4.94675134450245700E+05   5.84569299761196800E+05   2.68740374944970800E+06
   -2.54293806583983400E+05   2.05396861193992900E+06   2.67442299542458200E+07
   -2.64255488222761200E+07   1.13743438258904500E+09  -1.14376276468576100E+09
    9.70708063378627600E+00   2.35778474391517000E+01   2.65822018541962300E+03
    1.20880840340937600E+04   1.54883460011065300E+04   1.90230929114263600E+04
    2.53178115338652200E+04   1.90298438667389800E+04   2.24482361683339700E+04
    3.11892517616459600E+04   1.21614891415593700E+05   2.55166226779757100E+05
    6.08626632094715300E+05   1.50779229197664000E+06   1.83794319683241000E+06
    2.82950218682721500E+06   4.40043917681811400E+06   7.89472474983174600E+06
    1.11535384043682500E+07   1.93763328756605500E+07   2.25132290200842800E+08
    2.25357422491043800E+08   3.24932297557221100E+07   3.25257229854777800E+07
C TL6 HAWK conductor
-1G___1AX0046A                 2.  0.00             -2
      12        4.09034706490576600000E+02
    2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
    4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
    7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
    1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
    2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
    1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
    7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
    4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
      18        3.33580387547205000000E-05
    6.58877224092557400E+00   9.21132217460473900E+01   1.64236821912810800E+02
    3.01784074030806600E+02   4.07285148771138300E+02   7.20209715494018000E+02
    2.11600688286696500E+03   9.54417393977620700E+03  -7.22282835612318400E+05
    1.00592080081464800E+06   1.90612425927318100E+06   1.86832492641856100E+06
    1.20340357323744300E+07   3.93702748363217000E+07  -3.29562023743535300E+07
    1.18172079621027700E+08   9.03548751341195200E+10  -9.04955667614826500E+10
    2.33712518728992700E+03   3.24110385762308600E+04   5.80692968645576200E+04
    1.04952805612402300E+05   1.49680184774320500E+05   2.39617718503239400E+05
    3.37048463932246700E+05   4.26888044830299700E+05   2.18258538211766000E+06
    2.15901592579414600E+06   9.35283838216233300E+06   1.57160031279052000E+07
    4.27483849792290000E+07   6.31753569155142500E+07   6.84598541345712800E+07
    4.85147552098229100E+08   2.94499563544426600E+08   2.94794063107971300E+08
   1.00000000
```

```
  0.00000000
C TL7 HAWK conductor
-1A___2AX0042A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
    24      1.16743539295552900000E-04
   1.4033814424565100OE-02   3.3970349003243010OE-02   7.6905668181001290OE+00
   3.5811010659989480OE+01   4.5943485139443710OE+01   2.8491735643264840OE+02
   3.5338852110448310OE+01  -2.0150784111073770OE+02   6.3938248529242120OE+01
   8.8853559813206180OE+01   3.0102672982836120OE+03   1.2606966598410270OE+04
   6.1273815558461460OE+04   2.3418990204234390OE+05   1.3193599094607520OE+05
   4.9467513445024570OE+05   5.8456929976119680OE+05   2.6874037494497080OE+06
  -2.5429380658398340OE+05   2.0539686119399290OE+06   2.6744229954245820OE+07
  -2.6425548822276120OE+07   1.1374343825890450OE+09  -1.1437627646857610OE+09
   9.7070806337862760OE+00   2.3577847439151700OE+01   2.6582201854196230OE+03
   1.2088084034093760OE+04   1.5488346001106530OE+04   1.9023092911426360OE+04
   2.5317811533865220OE+04   1.9029843866738980OE+04   2.2448236168333970OE+04
   3.1189251761645960OE+04   1.2161489141559370OE+05   2.5516622677975710OE+05
   6.0862663209471530OE+05   1.5077922919766400OE+06   1.8379431968324100OE+06
   2.8295021868272150OE+06   4.4004391768181140OE+06   7.8947247498317460OE+06
   1.1153538404368250OE+07   1.9376332875660550OE+07   2.2513229020084280OE+08
   2.2535742249104380OE+08   3.2493229755722110OE+07   3.2525722985477780OE+07
  1.00000000
  0.00000000
C TL8 HAWK conductor
-1X0049AG___2A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
    12      5.00031399666535300000E-05
   2.2006505466685240OE+02   1.9917133079851560OE+01   3.0324587682859780OE+02
   1.3331946931657280OE+03   6.1065621476310030OE+03   4.9073160329080080OE+04
   8.2420464512539900OE+04   1.7290284880488390OE+06   1.9256172891326790OE+06
   8.0633573471674240OE+06   1.9562820318725090OE+10  -1.9574677798459200OE+10
   4.1365564306689630OE+04   3.8106280777490570OE+03   5.9144670933026060OE+04
   2.2567118658118070OE+05   2.8630129819420180OE+05   9.0132352748558780OE+05
   1.1637935076559950OE+06   1.3601737115924910OE+07   5.8380034486487250OE+06
   2.6427968698049180OE+07   1.1148336231140280OE+08   1.1159484567371430OE+08
  1.00000000
  0.00000000
C TL9 HAWK conductor
-1X0046AX0044A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
    19      2.16779123239017300000E-04
   2.5668812019169910OE-02   7.4573586542535580OE-02   2.6158242533535060OE+00
   1.5821400034897010OE+01   2.0157791621581910OE+01   2.3095225558142560OE+01
   3.7513625899986710OE+01   9.5468282926668350OE+01   6.0412567556937340OE+02
   9.7648615523174430OE+03   1.4706488559565730OE+04   3.5425302215796070OE+05
   3.5923220693707160OE+04   2.7768142142824660OE+05   7.8167462058354530OE+05
   7.7471224487183280OE+09  -7.5980069769264870OE+09   1.3551323622494650OE+10
  -1.3701913896819820OE+10
   9.2789686020372990OE+00   2.7107207914849560OE+01   9.5104437069529050OE+02
   5.5849222080702620OE+03   7.2943745131051480OE+03   8.3167404305381530OE+03
   1.3336790252125830OE+04   3.6099635343242440OE+04   2.6834670922832760OE+04
   1.5436613667813580OE+05   1.9357236334150650OE+05   8.8420482900030070OE+05
   1.0837417027655030OE+06   1.7693252262584120OE+06   2.7931417931620310OE+06
   1.2403798307821410OE+07   1.2416202106129210OE+07   1.0787161250841890OE+07
   1.0797948412092740OE+07
  1.00000000
  0.00000000
C TL10 HAWK conductor
-1X0046AA___1A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
    19      4.43851439830561100000E-04
   1.3846733280850070OE-02   6.3612025723329370OE-02   4.2065566089219500OE-02
   8.4294061998856490OE-02   2.7830324593105840OE-01   8.6339420458983720OE+00
   1.0416845703890280OE+01   2.2640770995110300OE+01   1.6322421017560130OE+02
   5.1764139806445930OE+03   4.7461012003364690OE+03   2.6632381229083540OE+04
   1.2830609309403920OE+05   4.4148861677650630OE+04   1.1532636208966570OE+06
  -8.2870306359694590OE+07   8.1501201114640980OE+07   9.0724858998812210OE+03
  -2.4461108141898670OE+03
   5.0543630218745620OE+00   1.9228451428876380OE+01   1.9481517270651050OE+01
   3.1383007112171700OE+01   1.0383346454392840OE+02   3.0841295424481410OE+03
   3.8898944731064800OE+03   8.8983138974601060OE+03   7.3625455474570110OE+03
   1.1080299227483260OE+05   5.0677539433029600OE+04   4.1283788130115310OE+05
   3.3046947651639940OE+05   5.7194355582472170OE+05   1.3941564599775770OE+06
   2.6404700803351120OE+06   2.6431105504154470OE+06   3.7351243444744360OE+06
   3.7388594688189050OE+06
  1.00000000
  0.00000000
C TL11 HAWK conductor
-1X0042AX0049A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
    21      1.40129890129590700000E-04
   2.8240946504097680OE-02   5.7366675514210650OE-01   2.1753629144096640OE+01
   3.4696998477248620OE+01   2.8632263850624500OE+01   2.7235842112372180OE+01
   3.3038477467983360OE+01   5.4427858119154380OE+01  -7.8362274009826730OE+02
   9.5361101287814850OE+03   4.5284789330850410OE+04   2.0970219147071450OE+05
   7.3401243046790370OE+05   9.4868597680161660OE+04   1.0952882976537170OE+06
   1.2760211070831240OE+06   4.4668032040495750OE+05   8.9249120212932580OE+06
  -8.8115910486319770OE+06   1.1015575983390880OE+09  -1.1055817299201960OE+09
   1.0564236361150320OE+01   2.1511772811952130OE+02   8.0017796856278630OE+03
   1.1745390196181120OE+04   1.2583604888609420OE+04   9.8231506988077850OE+03
   1.2245023335568170OE+04   2.0449323410736870OE+04   1.3556176382890870OE+05
   1.3101506677864060OE+05   4.6551910546953580OE+05   1.0789797923685850OE+06
   2.3984025536325340OE+06   2.9307829917931130OE+06   4.8713494886691970OE+06
   8.3024972985157260OE+06   1.2301888261565410OE+07   2.6484991346448700OE+08
   2.6511476337795100OE+08   2.1506538424976190OE+07   2.1528044963401180OE+07
  1.00000000
  0.00000000
C TL12 HAWK conductor
-1A___1AX0049A              2.  0.00               -2
    12      4.09034706490576600000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
```

```
  7.78217102800857800E+01  1.29727612927171000E+02  2.26805464551456900E+02
  4.14424917798641100E+03  2.15580737175207600E+04  8.57837925748202800E+05
    30       1.25283427372755600000E-03
  9.08757311529024800E-03  1.24921965405884400E-02  2.04684013014478000E-02
  2.06615327397690500E-02  3.31759823948442500E-02  4.20176524185137500E-02
  4.23787906612785400E-02  1.01879007311903300E-01  7.78394046512750100E-02
 -2.02349559986073500E-01  3.10375115852991000E-01  7.86215129679403700E-02
  1.05978609404068000E-01  1.27267670612558800E-01  1.71998694905671200E-01
  2.79964668391564400E+00  3.63380499208956800E+00  5.74442869220301300E+00
  1.78597780990942000E+01  7.03758038651224200E+01  1.33287316651978400E+02
  8.08711023086552800E+02  7.71126319206399800E+03  1.70597935435073100E+04
  1.55185819572756500E+04  3.50512237615357300E+05  9.59090873569393800E+11
 -5.01489359681364200E+12  5.01280173512390700E+12 -9.56999403724883900E+11
  3.72666260258944100E+00  5.16936374583739600E+00  8.23884462804631900E+00
  8.84463026449296800E+00  1.37960479431099500E+01  1.82225815658297100E+01
  1.71156600972840500E+01  2.87974024832269900E+01  3.34578016143075200E+01
  2.91435191536744300E+01  2.91280076907280200E+01  3.57447959438362200E+01
  4.53519895960392500E+01  5.66336588690298900E+01  7.59465867562610400E+01
  1.17688664181622500E+03  1.64491287601993100E+03  2.36665475861263800E+03
  3.56068851790910000E+03  5.84176701978976700E+03  6.67630768468816300E+03
  1.09890912038616700E+04  8.27565673224283100E+04  6.29060985548070500E+04
  1.32473146492676300E+05  3.40564995673745100E+05  7.37142997742820400E+05
  7.37880140740563900E+05  7.38228667513777500E+05  7.38966896181291800E+05
 1.00000000
 0.00000000
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
  X0017ASZ__1A       .05        1.                                         0
  SZ__1ASZTMPA                                        MEASURING            1
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
14X0007A 0      195.      60.       10.5                          -1.        1.
14X0010A 0      200.      60.       20.2                          -1.        1.
14X0030A 0      195.      60.       -17.                          -1.        1.
14X0035A 0      200.      60.       -6.8                          -1.        1.
/OUTPUT
  SZ__1ASZ__2A
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

B.3.2 Balanced Three-Phase to Ground Fault Case

```
BEGIN NEW DATA CASE
C -----------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C September 22, 2004
C -----------------------------------------------------------
POWER FREQUENCY                        60.
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   2.E-5      .2
       1       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
  SZ__1A                  200.  130.                                           0
  X0007AX0009A             1.1 45.52                                           0
  X0010AX0012A             1.2 38.98                                           0
  X0009AX0046A              .8  29.5                                           0
  X0012AG___1A             1.5  23.4                                           0
  SZ__1A                              8.                                       0
  SZ__2A                  650.  250.                                           0
  X0030AX0032A              .8 35.23                                           0
  X0032AX0049A              .6  20.8                                           0
  X0036AX0035A              .9 38.98                                           0
  G___2AX0036A             1.6  23.4                                           0
  X0042A                  420.   30.                                           0
  X0044A                  250.   25.                                           0
  X0044A                             20.                                       0
  IN__1A                 1200.  500.                                           0
  IN__2A                 2150.  380.                                           0
  A___1A                  250.   25.                                           0
  A___2A                  350.   60.                                           0
        X0017A              2.                                                 0
C TLS1 HAWK conductor
-1SZTMPAIN__1A              2.  0.00             -2
    12       4.09034706490576600000E+02
   2.45890513600235700E+03  1.00691916473651100E+03  3.73552293959034800E+03
   4.94910864283924900E+03  1.80922017453526000E+03  1.62638997544022700E+03
   7.23590211201397200E+02  9.98061571653958900E+02  5.16274050708064800E+02
   1.17955959812405300E+04  6.12653756748480700E+04  2.42883914569856300E+06
   2.53613213618031100E+00  3.67827403252360000E+00  7.35825287386416900E+00
   1.84026248477307900E+01  3.58022844932382000E+01  5.94599892702880700E+01
   7.78217102800857800E+01  1.29727612927171000E+02  2.26805464551456900E+02
   4.14424917798641100E+03  2.15580737175207600E+04  8.57837925748202800E+05
    19       4.00519444461175000000E-04
   1.56976974488308500E-02  5.96931801697355600E-02  7.65710632981902200E-02
   1.23025310978241300E-01  7.74362618176230500E+00  1.15631326792783200E+01
   1.51793395253545000E+01  2.45460099809200000E+01  2.17562524566291500E+02
   4.75157557302470000E+03  7.17277231675696400E+03  2.15420855113000800E+02
   1.92780440925090000E+05  1.00130204456972800E+05  6.67694141945910300E+05
   4.49850806862179600E+09 -4.59220311195214400E+09  3.24980116228040600E+09
 -3.15707914037557500E+09
   5.46006799865154000E+00  2.06382999204778400E+01  2.69410363626236900E+01
   4.36617508325875400E+01  2.72889153410192200E+03  3.96969851665451300E+03
   5.15623359578923200E+03  1.03865827369060600E+04  9.26732459557961600E+03
   1.12469458616561000E+05  6.93234200200993500E+04  4.98920839522332300E+05
   4.15754453085636900E+05  7.76422380585483400E+05  1.43944837446228700E+06
   4.02521957768449700E+06  4.02924479726218400E+06  4.38129540251903900E+06
   4.38567669792155100E+06
 1.00000000
 0.00000000
C TLS2 DRAKE conductor
-1SZTMPASZ__2A              2.  0.00             -2
    11       3.93897327703557300000E+02
   2.27109078735675900E+03  2.46873836980220700E+03  2.76728330884469200E+03
   9.27201327857635600E+02  8.13652385489715700E+02  4.84386453347630200E+02
   6.00660454747077000E+02  3.19123006877189800E+02  1.47413498397660100E+04
   5.35709730600259600E+04  2.76198969356064600E+06
   2.56520429418621500E+00  6.23867915240623700E+00  1.37315501948703500E+01
   2.44411566219346600E+01  3.88234599365499000E+01  5.22046486070647100E+01
   8.94985384307681800E+01  1.61248544547484800E+02  6.10024281549984200E+03
   2.24249725179149800E+04  1.15971094692565100E+06
    19       4.53915497928795800000E-04
   1.71275917005395300E-02  3.98691337390333500E-02  1.60864260029983000E-01
   6.89003618290678800E+00  8.78632925717092800E+00  1.18708400084055100E+01
   1.46650428128307100E+01  4.55132169053748200E+01  1.92681981165319200E+02
   2.21326495567518500E+03  1.11000476132130400E+04 -9.21432832870185500E+04
   2.81776544230162200E+05  1.67841941653263700E+04  1.22535036976167300E+06
 -3.16876240455654700E+07  2.93464632821155900E+07  3.77888719163484300E+07
 -3.68930729156446700E+07
   6.29920182790418300E+00  1.47535442461695500E+01  5.98892062931638800E+01
   2.46878104720658700E+03  3.24688225671564200E+03  4.20547568033697400E+03
   5.30166238127239600E+03  1.66566947044242800E+04  8.86571988667755600E+03
   4.65746128476995800E+04  1.13398074824504100E+05  4.43515608473281600E+05
   3.98417841600631700E+05  5.62068679963815700E+05  1.44682872910978600E+06
   2.87151967320063000E+06  2.87439119287383300E+06  3.65427188965252700E+06
   3.65792616154218300E+06
 1.00000000
 0.00000000
C TLS3 HAWK conductor
-1SZ__2AIN__2A              2.  0.00             -2
    12       4.09034706490576600000E+02
```

```
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     19        5.50554384535761800000E-04
   9.79465605855859600E-03   4.06212733123785000E-02   7.44854903179096900E-02
   4.00229870144877300E-02   6.56254564933406700E-02   7.79345690027440800E-02
   7.11151466287055800E-01   6.74005236856138300E+00   8.01620767358119800E+00
   1.17220670074905700E+01   3.50297130275519900E+01   7.05494675388751300E+01
   7.13600486817041200E+02   4.43545346148581900E+03   6.11781278301569400E+04
   6.79300571244501800E+04   7.49764722063926500E+05   2.36469459307105400E+07
  -2.45311009688204900E+07
   3.94587715068155200E+00   1.57035953406104900E+01   3.03393666019717600E+01
   1.67066122294504600E+01   2.60870054659298800E+01   3.32492107284800500E+01
   2.91988641610697900E+02   2.66461815651150000E+03   3.31965063922072700E+03
   4.56609698304445300E+03   7.47335672283102600E+03   6.69903629701688300E+03
   1.75902539791774200E+04   5.33686195521447800E+04   2.15476314218399300E+05
   3.13824776054083900E+05   9.37530861396163600E+05   1.82325148034484500E+06
   1.82507473182518700E+06
  1.00000000
  0.00000000
C TL1 DRAKE conductor
-1IN__1AA___1A                  2.  0.00                 -2
     11        3.93897327703557300000E+02
   2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
   9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
   6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
   5.35709730600259600E+04   2.76198969356064600E+06
   2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
   2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
   8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
   2.24249725179149800E+04   1.15971094692565100E+06
     18        5.00569525809250100000E-04
   1.43902995689640600E-02   4.33458044156565500E-02   5.16186088796879400E-02
   2.60473078409012900E+00   6.36887547988259200E+00   8.77520515952783600E+00
   1.07715571586440900E+01   1.62456122440872200E+01   1.49294354917248900E+02
   3.28434390136963700E+03   5.49853561056992600E+03   5.34533157308277900E+04
   2.31574716012179000E+05   3.27298197639052900E+05   3.14159278130439400E+08
  -3.10595574404512300E+08   9.30706221006033400E+08  -9.34891228010527500E+08
   5.61936471587928700E+00   1.68305760407245200E+01   2.05250926175730500E+01
   1.01726311455150700E+03   2.52403027449716200E+03   3.28509515605487000E+03
   4.12398597622061600E+03   7.90009211967655900E+03   7.14159016302577300E+03
   8.66107182139554900E+04   5.76718145371582000E+04   2.43513293383362700E+05
   5.26418551705884900E+05   9.59660805850347500E+05   3.87278673075127000E+06
   3.87665951748202500E+06   2.75397849079348300E+06   2.75673246928427900E+06
  1.00000000
  0.00000000
C TL2 DRAKE conductor
-1IN__2AA___2A                  2.  0.00                 -2
     11        3.93897327703557300000E+02
   2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
   9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
   6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
   5.35709730600259600E+04   2.76198969356064600E+06
   2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
   2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
   8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
   2.24249725179149800E+04   1.15971094692565100E+06
     20        4.00317427133653900000E-04
   2.08891430395107500E-02   4.32593489018147900E-02   1.15251517569450200E+00
   8.84981542682858600E+00   1.06444954332436300E+01   1.39766096840854400E+01
   1.91363487218951700E+01   6.03314183666120600E+01   3.04976279498481800E+02
   2.56135740054043300E+03   1.65083598049797800E+04   5.70650702060662600E+04
  -1.53112682223131700E+05   3.16998984679543100E+05   4.92484468183444900E+04
   1.19008457608240200E+06   1.68350028281018700E+08  -1.72202925551848900E+08
   1.16734633689862100E+08  -1.14361509663435200E+08
   7.46197475944493800E+00   1.55748584486930600E+01   4.15231927538063100E+02
   3.07835555158135900E+03   3.88583225013656500E+03   4.85587931971029500E+03
   6.73334411727069400E+03   2.14696936317370400E+04   1.34300288947473000E+04
```

```
   5.33603599572152600E+04   1.53615723524379400E+05   3.54735233931288400E+05
   5.75236151700719600E+05   5.40507595967298000E+05   7.45893945162507200E+05
   1.68236204302255000E+06   3.62728264518077400E+06   3.63090992782594900E+06
   4.48069217682420800E+06   4.48517286900102500E+06
  1.00000000
  0.00000000
C TL3 HAWK conductor
-1IN__2AA___1A                  2.  0.00                 -2
     12        4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     30        1.33643854753835200000E-03
   9.68976713503962700E-03   1.33120204768106500E-02   2.18568427218494000E-02
   2.19675564248809300E-02   3.38078085190960200E-02   4.33486771049049300E-02
   6.76106246957764400E-02   5.43140411789459800E-02   1.41460389799945100E-01
   9.11002157670505200E-02   8.01180983546494100E-02   1.12453334559639800E-01
   1.38076692919730400E-01   1.82463659623661200E-01   2.64099618447538500E+00
   3.87852196950510400E+00   6.12938827733858700E+00   1.90993810391674400E+01
   7.96538243161341800E+01   1.29056641368434100E+02   9.01650840605030900E+02
   7.99973827313519500E+03   4.45255469141922400E+03   1.42270434787099700E+04
   1.50844403119674400E+04   4.48375328644470000E+05  -8.32700885975779600E+07
   8.26595716973833400E+07   6.59442835173351500E+06  -6.47519367811138600E+06
   3.72635837689990400E+00   5.16894174539967600E+00   8.23817361522780300E+00
   8.84390655470486900E+00   1.32755619142013700E+01   1.68748913128125700E+01
   2.32643778079367300E+01   2.37675495625943900E+01   2.83107374370852200E+01
   3.46158684277817600E+01   3.57418804477292800E+01   4.53482047733419300E+01
   5.77262025912380300E+01   7.59404155027318700E+01   1.04881658190294000E+03
   1.64478990131605700E+03   2.36646385732968900E+03   3.56011487741156300E+03
   5.83982434202618200E+03   6.42072869092559000E+03   1.13892174160352400E+04
   5.35511456962960400E+04   9.56093676362913700E+04   7.29646502751217500E+04
   1.31776060655731200E+05   3.46569378060718900E+05   6.03290569663039200E+05
   6.03893860232702600E+05   7.50635238095463700E+05   7.51385873333557900E+05
  1.00000000
  0.00000000
C TL4 HAWK conductor
-1A___1AA___2A                  2.  0.00                 -2
     12        4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     14        7.34268486806543900000E-04
   3.29705233563703400E-02   1.06110147200249700E-01   1.49699996946430600E-01
   4.54005314668431200E-01   2.29894736051669500E+01   2.00264225444867500E+01
   3.98594237404368400E+01   2.45759513008771500E+03   2.07126101614158800E+03
   3.62588430407428400E+04   8.92966801187376800E+04   8.32467519726971600E+05
  -1.44068517734813900E+08   1.43105882217675000E+08
   5.87892499398299700E+00   1.88886536977137700E+01   2.73362791240876300E+01
   8.32906170850658600E+01   3.44044452144373500E+03   4.07461831915967500E+03
   3.66919194505697000E+03   5.23907878298744000E+04   2.21598491116926500E+04
   1.45187998152569500E+05   2.73030963103455100E+05   7.70543131940325700E+05
   1.29689934755590900E+06   1.29819624690346300E+06
  1.00000000
  0.00000000
C TL5 HAWK conductor
-1X0044AA___1A                  2.  0.00                 -2
     12        4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
```

```
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     24       1.16743539295552900000E-04
   1.40338144245651000E-02   3.39703490032430100E-02   7.69056681810012900E+00
   3.58110106599894800E+01   4.59434851394437100E+01   2.84917356432648400E+02
   3.53388521104483100E+01  -2.01507841110737700E+02   6.39382485292421200E+01
   8.88535598132061800E+01   3.01026729828361200E+03   1.26069665984102700E+04
   6.12738155584614600E+04   2.34189902042343900E+05   1.31935990946075200E+05
   4.94675134450245700E+05   5.84569299761196800E+05   2.68740374944970800E+06
  -2.54293806583983400E+05   2.05396861193992900E+06   2.67442299542458200E+07
  -2.64255488222761200E+07   1.13743438258904500E+09  -1.14376276468576100E+09
   9.70708063378627600E+00   2.35778474391517000E+01   2.65822018541962300E+03
   1.20880840340937600E+04   1.54883460011065300E+04   1.90230929114263600E+04
   2.53178115338652200E+04   1.90298438667389800E+04   2.24482361683339700E+04
   3.11892517616459600E+04   1.21614891415593700E+05   2.55166226779757100E+05
   6.08626632094715300E+05   1.50779229197664000E+06   1.83794319683241000E+06
   2.82950218682721500E+06   4.40043917681811400E+06   7.89472474983174600E+06
   1.11535384043682500E+07   1.93763328756605500E+07   2.25132290200842800E+08
   2.25357422491043800E+08   3.24932297557221100E+07   3.25257229854777800E+07
C TL6 HAWK conductor
-1G___1AX0046A                2.  0.00                -2
     12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     18       3.33580387547205000000E-05
   6.58877224092557400E+00   9.21132217460473900E+01   1.64236821912810800E+02
   3.01784074030806600E+02   4.07285148771138300E+02   7.20209715494018000E+02
   2.11600688286696500E+03   9.54417393977620700E+03  -7.22282835612318400E+05
   1.00592080081464800E+06   1.90612425927318100E+06   1.86832492641856100E+06
   1.20340357323744300E+07   3.93702748363217000E+07  -3.29562023743535300E+07
   1.18172079621027700E+08   9.03548751341195200E+10  -9.04955667614826500E+10
   2.33712518728992700E+03   3.24110385762308600E+04   5.80692968645576200E+04
   1.04952805612402300E+05   1.49680184774320500E+05   2.39617718503239400E+05
   3.37048463932246700E+05   4.26888044830299700E+05   2.18258538211766000E+06
   2.15901592579414600E+06   9.35283838216233300E+06   1.57160031279052000E+07
   4.27483849792290000E+07   6.31753569155142500E+07   6.84598541345712800E+07
   4.85147552098229100E+08   2.94499563544426600E+08   2.94794063107971300E+08
  1.00000000
  0.00000000
C TL7 HAWK conductor
-1A___2AX0042A                2.  0.00                -2
     12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     24       1.16743539295552900000E-04
   1.40338144245651000E-02   3.39703490032430100E-02   7.69056681810012900E+00
   3.58110106599894800E+01   4.59434851394437100E+01   2.84917356432648400E+02
   3.53388521104483100E+01  -2.01507841110737700E+02   6.39382485292421200E+01
   8.88535598132061800E+01   3.01026729828361200E+03   1.26069665984102700E+04
   6.12738155584614600E+04   2.34189902042343900E+05   1.31935990946075200E+05
   4.94675134450245700E+05   5.84569299761196800E+05   2.68740374944970800E+06
  -2.54293806583983400E+05   2.05396861193992900E+06   2.67442299542458200E+07
  -2.64255488222761200E+07   1.13743438258904500E+09  -1.14376276468576100E+09
   9.70708063378627600E+00   2.35778474391517000E+01   2.65822018541962300E+03
   1.20880840340937600E+04   1.54883460011065300E+04   1.90230929114263600E+04
   2.53178115338652200E+04   1.90298438667389800E+04   2.24482361683339700E+04
   3.11892517616459600E+04   1.21614891415593700E+05   2.55166226779757100E+05
   6.08626632094715300E+05   1.50779229197664000E+06   1.83794319683241000E+06
   2.82950218682721500E+06   4.40043917681811400E+06   7.89472474983174600E+06
   1.11535384043682500E+07   1.93763328756605500E+07   2.25132290200842800E+08
   2.25357422491043800E+08   3.24932297557221100E+07   3.25257229854777800E+07
  1.00000000
  0.00000000
C TL8 HAWK conductor
-1X0049AG___2A                2.  0.00                -2
     12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     12       5.00031399666535300000E-05
   2.20065054666852400E+02   1.99171330798515600E+01   3.03245876828597800E+02
   1.33319469316572800E+03   6.10656214763100300E+03   4.90731603290800800E+04
   8.24204645125399000E+04   1.72902848804883900E+06   1.92561728913267900E+06
   8.06335734716742400E+06   1.95628203187250900E+10  -1.95746777984592000E+10
   4.13655643066896300E+04   3.81062807774905700E+03   5.91446709330260600E+04
   2.25671186581180700E+05   2.86301298194201800E+05   9.01323527485587800E+05
   1.16379350765599500E+06   1.36017371159249100E+07   5.83800344864872500E+06
   2.64279686980491800E+07   1.11483362311402800E+08   1.11594845673714300E+08
  1.00000000
  0.00000000
C TL9 HAWK conductor
-1X0046AX0044A                2.  0.00                -2
     12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     19       2.16779123239017300000E-04
   2.56688120191699100E-02   7.45735865425355800E-02   2.61582425335350600E+00
   1.58214000348970100E+01   2.01577916215819100E+01   2.30952255581425600E+01
   3.75136258999867100E+01   9.54682829266683500E+01   6.04125675569373400E+02
   9.76486155231744300E+03   1.47064885595657300E+04   3.54253022157960700E+05
   3.59232206937071600E+04   2.77681421428246600E+05   7.81674620583545300E+05
   7.74712244871832800E+09  -7.59800697692648700E+09   1.35513236224946500E+10
  -1.37019138968198200E+10
   9.27896860203729900E+00   2.71072079148495600E+01   9.51044370695290500E+02
   5.58492220807026200E+03   7.29437451310514800E+03   8.31674043053815300E+03
   1.33367902521258300E+04   3.60996353432424400E+04   2.68346709228327600E+04
   1.54366136678135800E+05   1.93572363341506500E+05   8.84204829000300700E+05
   1.08374170276550300E+06   1.76932522625841200E+06   2.79314179316203100E+06
   1.24037983078214100E+07   1.24162021061292100E+07   1.07871612508418900E+07
   1.07979484120927400E+07
  1.00000000
  0.00000000
C TL10 HAWK conductor
-1X0046AA___1A                2.  0.00                -2
     12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
     19       4.43851439830561100000E-04
   1.38467332808500700E-02   6.36120257233293700E-02   4.20655660892195000E-02
   8.42940619988564900E-02   2.78303245931058400E-01   8.63394204589837200E+00
   1.04168457038902800E+01   2.26407709951103000E+01   1.63224210175601300E+02
   5.17641398064459300E+03   4.74610120033646900E+03   2.66323812290835400E+04
   1.28306093094039200E+05   4.41488616776506300E+04   1.15326362089665700E+06
  -8.28703063596945900E+07   8.15012011146409800E+07   9.07248589988122100E+03
  -2.44611081418986700E+03
   5.05436302187456200E+00   1.92284514288763800E+01   1.94815172706510500E+01
   3.13830071121717000E+01   1.03833464543928400E+02   3.08412954244814100E+03
```

```
   3.8898944731064800E+03   8.8983138974601060E+03   7.3625455474570110E+03
   1.1080299227483260E+05   5.0677539433029600E+04   4.1283788130115310E+05
   3.3046947651639940E+05   5.7194355582472170E+05   1.3941564599775770E+06
   2.6404700803351120E+06   2.6431105504154470E+06   3.7351243444744360E+06
   3.7388594688189050E+06
  1.00000000
  0.00000000
C TL11 HAWK conductor
-1X0042AX0049A                 2.  0.00                 -2
     12        4.0903470649057660000E+02
   2.4589051360023570E+03   1.0069191647365110E+03   3.7355229395903480E+03
   4.9491086428392490E+03   1.8092201745352600E+03   1.6263899754402270E+03
   7.2359021120139720E+02   9.9806157165395890E+02   5.1627405070806480E+02
   1.1795595981240530E+04   6.1265375674848070E+04   2.4288391456985630E+06
   2.5361321361803110E+00   3.6782740325236000E+00   7.3582528738641690E+00
   1.8402624847730790E+01   3.5802284493238200E+01   5.9459989270288070E+01
   7.7821710280085780E+01   1.2972761292717100E+02   2.2680546455145690E+02
   4.1442491779864110E+03   2.1558073717520760E+04   8.5783792574820280E+05
     21        1.4012989012959070000E-04
   2.8240946504097680E-02   5.7366675514210650E-01   2.1753629144096640E+01
   3.4696998477248620E+01   2.8632263850624500E+01   2.7235842112372180E+01
   3.3038477467983360E+01   5.4427858119154380E+01  -7.8362274009826730E+02
   9.5361101287814850E+03   4.5284789330850410E+04   2.0970219147071450E+05
   7.3401243046790370E+05   9.4868597680161660E+04   1.0952882976537170E+06
   1.2760211070831240E+06   4.4668032040495750E+05   8.9249120212932580E+06
  -8.8115910486319770E+06   1.1015575983390880E+09  -1.1055817299201960E+09
   1.0564236361150320E+01   2.1511772811952130E+02   8.0017796856278630E+03
   1.1745390196181120E+04   1.2583604888609420E+04   9.8231506988077850E+03
   1.2245023335568170E+04   2.0449323410736870E+04   1.3556176382890870E+05
   1.3101506677864060E+05   4.6551910546953580E+05   1.0789797923685850E+06
   2.3984025536325340E+06   2.9307829917931130E+06   4.8713494886691970E+06
   8.3024972985157260E+06   1.2301888261565410E+07   2.6484991346448700E+08
   2.6511476337795100E+08   2.1506538424976190E+07   2.1528044963401180E+07
  1.00000000
  0.00000000
C TL12 HAWK conductor
-1A___1AX0049A                 2.  0.00                 -2
     12        4.0903470649057660000E+02
   2.4589051360023570E+03   1.0069191647365110E+03   3.7355229395903480E+03
   4.9491086428392490E+03   1.8092201745352600E+03   1.6263899754402270E+03
   7.2359021120139720E+02   9.9806157165395890E+02   5.1627405070806480E+02
   1.1795595981240530E+04   6.1265375674848070E+04   2.4288391456985630E+06
   2.5361321361803110E+00   3.6782740325236000E+00   7.3582528738641690E+00
   1.8402624847730790E+01   3.5802284493238200E+01   5.9459989270288070E+01
   7.7821710280085780E+01   1.2972761292717100E+02   2.2680546455145690E+02
   4.1442491779864110E+03   2.1558073717520760E+04   8.5783792574820280E+05
     30        1.2528342737275560000E-03
   9.0875731152902480E-03   1.2492196540588440E-02   2.0468401301447800E-02
   2.0661532739769050E-02   3.3175982394844250E-02   4.2017652418513750E-02
   4.2378790661278540E-02   1.0187900731190330E-01   7.7839404651275010E-02
  -2.0234955998607350E-01   3.1037511585299100E-01   7.8621512967940370E-02
   1.0597860940406800E-01   1.2726767061255880E-01   1.7199869490567120E-01
   2.7996466839156440E+00   3.6338049920895680E+00   5.7444286922030130E+00
   1.7859778099094200E+01   7.0375803865122420E+01   1.3328731665197840E+02
   8.0871102308655280E+02   7.7112631920639980E+03   1.7059793543507310E+04
   1.5518581957275650E+04   3.5051223761535730E+05   9.5909087356939380E+11
  -5.0148935968136420E+12   5.0128017351239070E+12  -9.5699940372488390E+11
   3.7266626025894410E+00   5.1693637458373960E+00   8.2388446280463190E+00
   8.8446302644929680E+00   1.3796047943109950E+01   1.8222581565829710E+01
   1.7115660097284050E+01   2.8797402483226990E+01   3.3457801614307520E+01
   2.9143519153674430E+01   2.9128007690728020E+01   3.5744795943836220E+01
   4.5351989596039250E+01   5.6633658869029890E+01   7.5946586756261040E+01
   1.1768866418162250E+03   1.6449128760199310E+03   2.3666547586126380E+03
   3.5606885179091000E+03   5.8417670197897670E+03   6.6763076846881630E+03
   1.0989091203861670E+04   8.2756567322428310E+04   6.2906098554807050E+04
   1.3247314649267630E+05   3.4056499567374510E+05   7.3714299774282040E+05
   7.3788014074056390E+05   7.3822866751377750E+05   7.3896689618129180E+05
  1.00000000
  0.00000000
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
  X0017ASZ__2A        .05       .15                                       0
  SZ__1ASZTMPA                                      MEASURING             1
```

```
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
14X0007A 0      195.       60.      10.5                                -1.        1.
14X0010A 0      200.       60.      20.2                                -1.        1.
14X0030A 0      195.       60.      -17.                                -1.        1.
14X0035A 0      200.       60.      -6.8                                -1.        1.
/OUTPUT
  SZ__1ASZ__2A
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

B.4 ATP Data File for Robust TLNE Model

B.4.1 Capacitor $C1$ Switching Case

```
BEGIN NEW DATA CASE
C -----------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C September 22, 2004
C -----------------------------------------------------------
POWER FREQUENCY                               60.
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   2.E-5    .15
      50       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
C ********************************* Alpha Mode ***********************************
C *************
C study zone
C *************
  SZ__1ASZTMPA              1.E-9                                            1
  SZ__1A                     200.  130.                                      0
  SZ__2A                     650.  250.                                      0
  IN__1A                    1200.  500.                                      0
  IN__2A                    2150.  380.                                      0
  X0018A                                 8.                                  0
C HAWK conductor
-1SZTMPAIN__1A                 2.  0.00                 -2 1
     12        4.0903470649057660000E+02
   2.4589051360023570E+03   1.0069191647365110E+03   3.7355229395903480E+03
   4.9491086428392490E+03   1.8092201745352600E+03   1.6263899754402270E+03
   7.2359021120139720E+02   9.9806157165395890E+02   5.1627405070806480E+02
   1.1795595981240530E+04   6.1265375674848070E+04   2.4288391456985630E+06
   2.5361321361803110E+00   3.6782740325236000E+00   7.3582528738641690E+00
   1.8402624847730790E+01   3.5802284493238200E+01   5.9459989270288070E+01
   7.7821710280085780E+01   1.2972761292717100E+02   2.2680546455145690E+02
   4.1442491779864110E+03   2.1558073717520760E+04   8.5783792574820280E+05
     19        4.0051944446117500000E-04
   1.5697697448830850E-02   5.9693180169735560E-02   7.6571063298190220E-02
   1.2302531097824130E-01   7.7436261817623050E+00   1.1563132679278320E+01
   1.5179339525354500E+01   2.4546009980920000E+01   2.1756252456629150E+02
   4.7515755730247000E+03   7.1727723167569640E+03   2.1542085511300080E+02
   1.9278044092509000E+05   1.0013020445697280E+05   6.6769414194591030E+05
   4.4985080686217960E+09  -4.5922031119521440E+09   3.2498011622804060E+09
```

```
  -3.1570791403755750OE+09
   5.4600679986515400OE+00   2.0638299920477840OE+01   2.6941036362623690OE+01
   4.3661750832587540OE+01   2.7288915341019220OE+03   3.9696985166545130OE+03
   5.1562335957892320OE+03   1.0386582736906060OE+04   9.2673245955796160OE+03
   1.1246945861656100OE+05   6.9323420020099350OE+04   4.9892083952233230OE+05
   4.1575445308563690OE+05   7.7642238058548340OE+05   1.4394483744622870OE+06
   4.0252195776844970OE+06   4.0292447972621840OE+06   4.3812954025190390OE+06
   4.3856766979215510OE+06
  1.00000000
  0.00000000
C DRAKE conductor
-1SZTMPASZ__2A                 2.  0.00               -2 1
     11        3.9389732770355730000E+02
   2.2710907873567590OE+03   2.4687383698022070OE+03   2.7672833088446920OE+03
   9.2720132785763560OE+02   8.1365238548971570OE+02   4.8438645334763020OE+02
   6.0066045474707700OE+02   3.1912300687718980OE+02   1.4741349839766010OE+04
   5.3570973060025960OE+04   2.7619896935606460OE+06
   2.5652042941862150OE+00   6.2386791524062370OE+00   1.3731550194870350OE+01
   2.4441156621934660OE+01   3.8823459936549900OE+01   5.2204648607064710OE+01
   8.9498538430768180OE+01   1.6124854454748480OE+02   6.1002428154998420OE+03
   2.2424972517914980OE+04   1.1597109469256510OE+06
     19        4.5391549792879580000E-04
   1.7127591700539530OE-02   3.9869133739033350OE-02   1.6086426002998300OE-01
   6.8900361829067880OE+00   8.7863292571709280OE+00   1.1870840008405510OE+01
   1.4665042812830710OE+01   4.5513216905374820OE+01   1.9268198116531920OE+02
   2.2132649556751850OE+03   1.1100047613213040OE+04  -9.2143283287018550OE+04
   2.8177654423016220OE+05   1.6784194165326370OE+04   1.2253503697616730OE+06
  -3.1687624045565470OE+07   2.9346463282115590OE+07   3.7788871916348430OE+07
  -3.6893072915644670OE+07
   6.2992018279041830OE+00   1.4753544246169550OE+01   5.9889206293163880OE+01
   2.4687810472065870OE+03   3.2468822567156420OE+03   4.2054756803369740OE+03
   5.3016623812723960OE+03   1.6656694704424280OE+04   8.8657198866775560OE+03
   4.6574612847699580OE+04   1.1339807482450410OE+05   4.4351560847328160OE+05
   3.9841784160063170OE+05   5.6206867996381570OE+05   1.4468287291097860OE+06
   2.8715196732006300OE+06   2.8743911928738330OE+06   3.6542718896525270OE+06
   3.6579261615421830OE+06
  1.00000000
  0.00000000
C HAWK conductor
-1SZ__2AIN__2A                 2.  0.00               -2 1
     12        4.0903470649057660000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
   7.2359021120139720OE+02   9.9806157165395890OE+02   5.1627405070806480OE+02
   1.1795595981240530OE+04   6.1265375674848070OE+04   2.4288391456985630OE+06
   2.5361321361803110OE+00   3.6782740325236000OE+00   7.3582528738641690OE+00
   1.8402624847730790OE+01   3.5802284493238200OE+01   5.9459989270288070OE+01
   7.7821710280085780OE+01   1.2972761292717100OE+02   2.2680546455145690OE+02
   4.1442491779864110OE+03   2.1558073717520760OE+04   8.5783792574820280OE+05
     19        5.5055438453576180000E-04
   9.7946560585585960OE-03   4.0621273312378500OE-02   7.4485490317909690OE-02
   4.0022987014487730OE-02   6.5625456493340670OE-02   7.7934569002744080OE-02
   7.1115146628705580OE-01   6.7400523685613830OE+00   8.0162076735811980OE+00
   1.1722067007490570OE+01   3.5029713027551990OE+01   7.0549467538875130OE+01
   7.1360048681704120OE+02   4.4354534614858190OE+03   6.1178127830156940OE+04
   6.7930057124450180OE+04   7.4976472206392650OE+05   2.3646945930710540OE+07
  -2.4531100968820490OE+07
   3.9458771506815520OE+00   1.5703595340610490OE+01   3.0339366601971760OE+01
   1.6706612229450460OE+01   2.6087005465929880OE+01   3.3249210728480050OE+01
   2.9198864161069790OE+02   2.6646181565115000OE+03   3.3196506392207270OE+03
   4.5660969830444530OE+03   7.4733567228310260OE+03   6.6990362970168830OE+03
   1.7590253979177420OE+04   5.3368619552144780OE+04   2.1547631421839930OE+05
   3.1382477605408390OE+05   9.3753086139616360OE+05   1.8232514803448450OE+06
   1.8250747318251870OE+06
  1.00000000
  0.00000000
C *************
C surface layer
C *************
C DRAKE conductor
-1IN__1AA____1                 2.  0.00               -2 1
      8        3.7502383477891607OOE+002
   5.1182350308961829E+005  -7.7220400150216330E+004   5.3836081387381419E+005
  -1.3130787111864449E+006   1.4106854758738480E+006  -8.7056744342053682E+005
   3.9738886193076416E+005  -8.1086902030084631E+004
   1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
   3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
   6.0037967902148388E+000   2.6655609959826490E+000
     10        5.0097255352752625OOE-004
   1.6270863906469899E+008  -1.6431596001157719E+008   1.4214204455473491E+006
   1.0885261434708411E+005   6.4174528600579331E+004   1.1562808837095919E+004
   1.0498665383127150E+003   1.2301409456296111E+002   1.3744260229970391E+002
   2.2630797117004681E-001
   2.1851996518619689E+006   2.1830166352267410E+006   1.2797282522085649E+006
   4.1814911890264828E+005   2.5794236314651690E+005   1.1288345220778709E+005
   2.5365200913558881E+004   5.8946984100303853E+003   1.3005384419092390E+004
   2.2225616072406790E+001
  1.00000000
  0.00000000
C DRAKE conductor
-1IN__2AA____2                 2.  0.00               -2 1
      8        3.9390329733500556OOE+002
   1.0511045492931569E+005   2.4164547148414579E+002  -1.5017186955810123E+004
   6.3476046493852729E+004  -8.0135889472332128E+004   4.9970378687677192E+004
  -8.5537264822298675E+003   3.3880808573122692E+002
   1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
   3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
   6.0037967902148388E+000   2.6655609959826490E+000
     12        4.0042605096120450OOE-004
   2.9821418453288162E+008  -3.0044193674812359E+008   1.9495373574076470E+006
   1.8119051123979481E+005   4.5287095013004116E+004   4.9416715953395367E+004
   1.8523396032005680E+003   3.6333402652528878E+002   5.8013994064875767E+001
   3.9268948601835461E+001   7.5171522529347543E+000   6.1905967511505823E-002
   3.1923605038688411E+006   3.1891713325363020E+006   1.8517660946908800E+006
   6.0844300818797620E+005   3.0829532807651232E+005   2.4818127081645271E+005
   4.0789847713431889E+004   1.5961619255950680E+004   9.9200042766566548E+003
   6.7969813906110430E+003   1.3438258169018341E+003   1.1100966125235569E+001
  1.00000000
  0.00000000
C HAWK conductor
-1IN__2AA____1                 2.  0.00               -2 1
     12        4.0903348945255431OOE+002
   2.4288461303353016E+006   6.1273663687205080E+004   1.1791906944042370E+004
   5.1956569709320877E+002   3.3236418590387979E+002   5.7164357367498978E+003
   3.2634004169326912E+004  -1.5586524316327844E+005   4.8412147680448723E+005
  -1.2384748760169270E+006   1.3944189855436631E+006  -5.0557916661077325E+005
   8.5783792574820283E+005   2.1558073717520761E+004   4.1442491779864113E+003
   2.2680546455145691E+002   1.2972761292717101E+002   7.7821710280085782E+001
   5.9459989270288069E+001   3.5802284493238197E+001   1.8402624847730792E+001
   7.3582528738641688E+000   3.6782740325235999E+000   2.5361321361803109E+000
     11        1.3355385859350800OOE-003
  -1.7748345869520440E+009   1.7507732888229799E+009   2.4023866400058411E+007
   2.1781893643545041E+004   1.4510008190993110E+004   5.9933232436985 78E+002
   5.3905705626991960E+002   9.0291025531296631E-001   2.8882885870059560E-001
   1.8241089263411739E-001   6.3630822885285973E-002
   4.9455971431356872E+005   4.9406564866490458E+005   5.2859557145165885E+005
   9.3790496018900856E+004   5.8281852345950239E+004   7.0495362844468582E+003
   1.4090256362095870E+004   9.2024301665763190E+001   2.9266337352851991E+001
   1.7828841875301030E+001   6.1809013182452661E+000
  1.00000000
  0.00000000
C ******************************** Beta Mode ********************************
C *************
C study zone
C *************
  SZ__1BSZTMPB                1.E-9                                          1
  SZ__1B                       200.  130.                                    0
  SZ__2B                       650.  250.                                    0
  IN__1B                      1200.  500.                                    0
  IN__2B                      2150.  380.                                    0
  X0018B                                  8.                                 0
C HAWK conductor
-1SZTMPBIN__1B                 2.  0.00               -2 1
     12        4.0903470649057660000E+02
   2.4589051360023570OE+03   1.0069191647365110OE+03   3.7355229395903480OE+03
   4.9491086428392490OE+03   1.8092201745352600OE+03   1.6263899754402270OE+03
```

```
  7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
  1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
  2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
  1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
  7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
  4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
    19        4.00519444461175000000E-04
  1.56976974488308500E-02   5.96931801697355600E-02   7.65710632981902200E-02
  1.23025310978241300E-01   7.74362618176230500E+00   1.15631326792783200E+01
  1.51793395253545000E+01   2.45460099809200000E+01   2.17562524566291500E+02
  4.75157557302470000E+03   7.17277231675696400E+03   2.15420855113000800E+02
  1.92780440925090000E+05   1.00130204456972800E+05   6.67694141945910300E+05
  4.49850806862179600E+09  -4.59220311195214400E+09   3.24980116228040600E+09
 -3.15707914037557500E+09
  5.46006799865154000E+00   2.06382999204778400E+01   2.69410363626236900E+01
  4.36617508325875400E+01   2.72889153410192200E+03   3.96969851665451300E+03
  5.15623359578923200E+03   1.03865827369060600E+04   9.26732459557961600E+03
  1.12469458616561000E+05   6.93234200200993500E+04   4.98920839522332300E+05
  4.15754453085636900E+05   7.76422380585483400E+05   1.43944837446228700E+06
  4.02521957768449700E+06   4.02924479726218400E+06   4.38129540251903900E+06
  4.38567669792155100E+06
 1.00000000
 0.00000000
C DRAKE conductor
-1SZTMPBSZ__2B                2.  0.00                -2 1
    11        3.93897327703557300000E+02
  2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
  9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
  6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
  5.35709730600259600E+04   2.76198969356064600E+06
  2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
  2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
  8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
  2.24249725179149800E+04   1.15971094692565100E+06
    19        4.53915497928795800000E-04
  1.71275917005395300E-02   3.98691337390333500E-02   1.60864260029983000E-01
  6.89003618290678800E+00   8.78632925717092800E+00   1.18708400084055100E+01
  1.46650428128307100E+01   4.55132169053748200E+01   1.92681981165319200E+02
  2.21326495567518500E+03   1.11000476132130400E+04  -9.21432832870185500E+04
  2.81776544230162200E+05   1.67841941653263700E+04   1.22535036976167300E+06
 -3.16876240455654700E+07   2.93464632821155900E+07   3.77888719163484300E+07
 -3.68930729156446700E+07
  6.29920182790418300E+00   1.47535442461695500E+01   5.98892062931638800E+01
  2.46878104720658700E+03   3.24688225671564200E+03   4.20547568033697400E+03
  5.30166238127239600E+03   1.66566947044242800E+04   8.86571988667755600E+03
  4.65746128476995800E+04   1.13398074824504100E+05   4.43515608473281600E+05
  3.98417841600631700E+05   5.62068679963815700E+05   1.44682872910978600E+06
  2.87151967320063000E+06   2.87439119287383300E+06   3.65427188965252700E+06
  3.65792616154218300E+06
 1.00000000
 0.00000000
C HAWK conductor
-1SZ__2BIN__2B                2.  0.00                -2 1
    12        4.09034706490576600000E+02
  2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
  4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
  7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
  1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
  2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
  1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
  7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
  4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
    19        5.50554384535761800000E-04
  9.79465605855859600E-03   4.06212733123785000E-02   7.44854903179096900E-02
  4.00229870144877300E-02   6.56254564933406700E-02   7.79345690027440800E-02
  7.11151466287055800E-01   6.74005236856138300E+00   8.01620767358119800E+00
  1.17220670074905700E+01   3.50297130275519900E+01   7.05494675388751300E+01
  7.13600486817041200E+02   4.43545346148581900E+03   6.11781278301569400E+04
  6.79300571244501800E+04   7.49764722063926500E+05   2.36469459307105400E+07
 -2.45311009688204900E+07
  3.94587715068155200E+00   1.57035953406104900E+01   3.03393666019717600E+01
  1.67066122294504600E+01   2.60870054659298800E+01   3.32492107284800500E+01
  2.91988641610697900E+02   2.66461815651150000E+03   3.31965063922072700E+03
  4.56609698304445300E+03   7.47335672283102600E+03   6.69903629701688300E+03
  1.75902539791774200E+04   5.33686195521447800E+04   2.15476314218399300E+05
  3.13824776054083900E+05   9.37530861396163600E+05   1.82325148034484500E+06
  1.82507473182518700E+06
 1.00000000
 0.00000000
C *************
C surface layer
C *************
C DRAKE conductor
-1IN__1BB____1                2.  0.00                -2 1
     8        3.75023834778916070OE+002
  5.1182350308961829E+005  -7.7220400150216330E+004   5.3836081387381419E+005
 -1.3130787111864449E+006   1.4106854758738480E+006  -8.7056744342053682E+005
  3.9738886193076416E+005  -8.1086902030084631E+004
  1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
  3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
  6.0037967902148388E+000   2.6655609959826490E+000
    10        5.0097255352752625OOE-004
  1.6270863906469899E+008  -1.6431596001157719E+008   1.4214204455473491E+006
  1.0885261434708411E+005   6.4174528600579331E+004   1.1562808837095919E+004
  1.0498665383127150E+003   1.2301409456296111E+002   1.3744260229970391E+002
  2.2630797117004681E-001
  2.1851996518619689E+006   2.1830166352267410E+006   1.2797282522085649E+006
  4.1814911890264828E+005   2.5794236314651690E+005   1.1288345220778709E+005
  2.5365200913558881E+004   5.8946984100303853E+003   1.3005384419092390E+004
  2.2225616072406790E+001
 1.00000000
 0.00000000
C DRAKE conductor
-1IN__2BB____2                2.  0.00                -2 1
     8        3.9390329733500556OOE+002
  1.0511045492931569E+005   2.4164547148414579E+002  -1.5017186955810123E+004
  6.3476046493852729E+004  -8.0135889472332128E+004   4.9970378687677192E+004
 -8.5537264822298675E+003   3.3880808573122692E+002
  1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
  3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
  6.0037967902148388E+000   2.6655609959826490E+000
    12        4.0042605096120450OOE-004
  2.9821418453288162E+008  -3.0044193674812359E+008   1.9495373574076470E+006
  1.8119051123979481E+005   4.5287095013004116E+004   4.9416715953395367E+004
  1.8523396032005680E+003   3.6333402652528878E+002   5.8013994064875767E+001
  3.9268948601835461E+001   7.5171522529347543E+000   6.1905967511505823E-002
  3.1923605038688411E+006   3.1891713325363020E+006   1.8517660946908800E+006
  6.0844300818797620E+005   3.0829532807651232E+005   2.4818127081645271E+005
  4.0789847713431889E+004   1.5961619255950680E+004   9.9200042766566548E+003
  6.7969813906110430E+003   1.3438258169018341E+003   1.1100966125235569E+001
 1.00000000
 0.00000000
C HAWK conductor
-1IN__2BB____1                2.  0.00                -2 1
    12        4.0903348945255431OOE+002
  2.4288461303353016E+006   6.1273663687205080E+004   1.1791906944042370E+004
  5.1956569709320877E+002   3.3236418590387979E+002   5.7164357367498978E+003
  3.2634004169326912E+004  -1.5586524316327844E+005   4.8412147680448723E+005
 -1.2384748760169270E+006   1.3944189855436631E+006  -5.0557916661077325E+005
  8.5783792574820283E+005   2.1558073717520761E+004   4.1442491779864113E+003
  2.2680546455145691E+002   1.2972761292717101E+002   7.7821710280085782E+001
  5.9459989270288069E+001   3.5802284493238197E+001   1.8402624847730792E+001
  7.3582528738641688E+000   3.6782740325235999E+000   2.5361321361803109E+000
    11        1.3355385859350800OOE-003
 -1.7748345869520440E+009   1.7507732888229799E+009   2.4023866400058411E+007
  2.1781893643545041E+004   1.4510008190993110E+004   5.9933233243698578E+002
  5.3905705626991960E+002   9.0291025531296631E-001   2.8882885870059560E-001
  1.8241089263411739E-001   6.3630822885285973E-002
  4.9455971431356872E+005   4.9406564866490458E+005   5.2859557145165885E+005
  9.3790496018900856E+004   5.8281852345950239E+004   7.0495362844468582E+003
  1.4090256362095870E+004   9.2024301665763190E+001   2.9266337352851991E+001
  1.7828841875301030E+001   6.1809013182452661E+000
 1.00000000
 0.00000000
C ********************************* Alpha Mode *********************************
C *************
```

```
C deep region
C *************
C BEGIN FDNE
$VINTAGE,1
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><    microF     >
C
C (1,1)
  A____1                       1.179990e+002
  A____1                      -1.817485e+001  -1.663768e+003
  A____1                       1.139488e+001   2.460232e+002
  A____1                       1.235152e+001   1.618993e+002
  A____1                       1.198652e+003   5.511484e+002
  A____1                      -1.989105e+002  -1.174363e+001
  A____1A 6__1                 6.337633e+000   6.022723e+001
  A 6__1                       4.738628e+002
  A 6__1                                                       1.181832e+001
  A____1A 8__1                 1.102904e+001   2.229382e+002
  A 8__1                       2.072046e+004
  A 8__1                                                       8.153085e-001
  A____1A A__1                 1.013795e+001   7.765019e+001
  A A__1                       1.249981e+006
  A A__1                                                       7.261159e-001
  A____1A C__1                 2.569920e+002   3.941270e+002
  A C__1                      -1.787782e+004
  A C__1                                                       1.139597e-001
  A____1A E__1                -1.620042e+002   2.269678e+002
  A E__1                       1.168918e+004
  A E__1                                                       8.943689e-002
  A____1A10__1                 2.320907e+004   1.290952e+003
  A10__1                      -2.935271e+004
  A10__1                                                       1.945724e-003
  A____1A12__1                 4.713920e+001   6.899455e+001
  A12__1                      -2.446764e+004
  A12__1                                                       8.993800e-002
  A____1A14__1                -2.288147e+006  -1.622627e+004
  A14__1                       2.321225e+006
  A14__1                                                      -3.060986e-006
C (1,2)
  A____1A____2                -6.862856e+003
  A____1A____2                 5.790700e+001   5.300940e+003
  A____1A____2                -3.452911e+001  -7.455070e+002
  A____1A____2                 1.627967e+001   2.133881e+002
  A____1A____2                 3.760290e+003   1.729007e+003
  A____1A____2                 2.107372e+004   1.244187e+003
  A____1A 6_12                 3.509147e+004  -5.910586e+004
  A 6_12A____2                -1.309887e+005
  A 6_12A____2                                                -8.700025e-003
  A____1A 8_12                 2.942102e+002  -1.138949e+003
  A 8_12A____2                -1.737797e+004
  A 8_12A____2                                                -1.568034e-001
  A____1A A_12                -2.195788e+001  -1.550764e+002
  A A_12A____2                 2.769217e+005
  A A_12A____2                                                -3.635504e-001
  A____1A C_12                -1.356027e+003   9.905119e+002
  A C_12A____2                 1.556051e+004
  A C_12A____2                                                 4.199693e-002
  A____1A E_12                 2.534693e+003  -8.123033e+002
  A E_12A____2                -1.426832e+004
  A E_12A____2                                                -2.083931e-002
  A____1A10_12                -3.480429e+003   5.651609e+002
  A10_12A____2                 1.058648e+004
  A10_12A____2                                                 1.425341e-002
  A____1A12_12                -1.427467e+002  -1.380378e+002
  A12_12A____2                 2.771293e+004
  A12_12A____2                                                -4.480791e-002
  A____1A14_12                -1.835352e+002   1.407996e+002
  A14_12A____2                 2.579641e+004
  A14_12A____2                                                 2.457868e-002
C (2,2)
  A____2                       2.136131e+002
  A____2                      -1.697350e+001  -1.553794e+003
  A____2                       1.209292e+001   2.610945e+002
  A____2                       2.347825e+001   3.077445e+002
  A____2                       5.817348e+002   2.674856e+002
  A____2                      -4.503580e+002  -2.658903e+001
  A____2A 6__2                 1.269424e+005  -4.880319e+004
  A 6__2                      -1.510270e+005
  A 6__2                                                      -2.295168e-003
  A____2A 8__2                -2.646082e+002   8.977345e+002
  A 8__2                       1.251412e+004
  A 8__2                                                       1.980824e-001
  A____2A A__2                 9.729605e+000   7.751308e+001
  A A__2                       2.239360e+005
  A A__2                                                       7.274259e-001
  A____2A C__2                 4.703554e+002  -1.343231e+003
  A C__2                      -5.811050e+004
  A C__2                                                      -3.365080e-002
  A____2A E__2                -3.314680e+003   6.780535e+002
  A E__2                       9.734108e+003
  A E__2                                                       2.002066e-002
  A____2A10__2                -2.573017e+002  -6.882284e+002
  A10__2                      -6.031784e+005
  A10__2                                                      -1.744483e-002
  A____2A12__2                 4.967693e+001   6.967468e+001
  A12__2                      -2.319546e+004
  A12__2                                                       8.904087e-002
  A____2A14__2                -1.198030e+006   1.281169e+004
  A14__2                       1.237224e+006
  A14__2                                                       8.618294e-006
C ********************************* Beta Mode ********************************
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><    microF     >
C
C (1,1)
  B____1                       1.179990e+002
  B____1                      -1.817485e+001  -1.663768e+003
  B____1                       1.139488e+001   2.460232e+002
  B____1                       1.235152e+001   1.618993e+002
  B____1                       1.198652e+003   5.511484e+002
  B____1                      -1.989105e+002  -1.174363e+001
  B____1B 6__1                 6.337633e+000   6.022723e+001
  B 6__1                       4.738628e+002
  B 6__1                                                       1.181832e+001
  B____1B 8__1                 1.102904e+001   2.229382e+002
  B 8__1                       2.072046e+004
  B 8__1                                                       8.153085e-001
  B____1B A__1                 1.013795e+001   7.765019e+001
  B A__1                       1.249981e+006
  B A__1                                                       7.261159e-001
  B____1B C__1                 2.569920e+002   3.941270e+002
  B C__1                      -1.787782e+004
  B C__1                                                       1.139597e-001
  B____1B E__1                -1.620042e+002   2.269678e+002
  B E__1                       1.168918e+004
  B E__1                                                       8.943689e-002
  B____1B10__1                 2.320907e+004   1.290952e+003
  B10__1                      -2.935271e+004
  B10__1                                                       1.945724e-003
  B____1B12__1                 4.713920e+001   6.899455e+001
  B12__1                      -2.446764e+004
  B12__1                                                       8.993800e-002
  B____1B14__1                -2.288147e+006  -1.622627e+004
  B14__1                       2.321225e+006
  B14__1                                                      -3.060986e-006
C (1,2)
  B____1B____2                -6.862856e+003
  B____1B____2                 5.790700e+001   5.300940e+003
  B____1B____2                -3.452911e+001  -7.455070e+002
  B____1B____2                 1.627967e+001   2.133881e+002
  B____1B____2                 3.760290e+003   1.729007e+003
  B____1B____2                 2.107372e+004   1.244187e+003
  B____1B 6_12                 3.509147e+004  -5.910586e+004
  B 6_12B____2                -1.309887e+005
  B 6_12B____2                                                -8.700025e-003
  B____1B 8_12                 2.942102e+002  -1.138949e+003
  B 8_12B____2                -1.737797e+004
  B 8_12B____2                                                -1.568034e-001
```

```
  B____1B A_12              -2.195788e+001  -1.550764e+002
  B A_12B____2               2.769217e+005
  B A_12B____2                                              -3.635504e-001
  B____1B C_12              -1.356027e+003   9.905119e+002
  B C_12B____2               1.556051e+004
  B C_12B____2                                               4.199693e-002
  B____1B E_12               2.534693e+003  -8.123033e+002
  B E_12B____2              -1.426832e+004
  B E_12B____2                                              -2.083931e-002
  B____1B10_12              -3.480429e+003   5.651609e+002
  B10_12B____2               1.058648e+004
  B10_12B____2                                               1.425341e-002
  B____1B12_12              -1.427467e+002  -1.380378e+002
  B12_12B____2               2.771293e+004
  B12_12B____2                                              -4.480791e-002
  B____1B14_12              -1.835352e+002   1.407996e+002
  B14_12B____2               2.579641e+004
  B14_12B____2                                               2.457868e-002
C (2,2)
  B____2                     2.136131e+002
  B____2                    -1.697350e+001  -1.553794e+003
  B____2                     1.209292e+001   2.610945e+002
  B____2                     2.347825e+001   3.077445e+002
  B____2                     5.817348e+002   2.674856e+002
  B____2                    -4.503580e+002  -2.658903e+001
  B____2B 6__2               1.269424e+005  -4.880319e+004
  B 6__2                    -1.510270e+005
  B 6__2                                                    -2.295168e-003
  B____2B 8__2              -2.646082e+002   8.977345e+002
  B 8__2                     1.251412e+004
  B 8__2                                                     1.980824e-001
  B____2B A__2               9.729605e+000   7.751308e+001
  B A__2                     2.239360e+005
  B A__2                                                     7.274259e-001
  B____2B C__2               4.703554e+002  -1.343231e+003
  B C__2                    -5.811050e+004
  B C__2                                                    -3.365080e-002
  B____2B E__2              -3.314680e+003   6.780535e+002
  B E__2                     9.734108e+003
  B E__2                                                     2.002066e-002
  B____2B10__2              -2.573017e+002  -6.882284e+002
  B10__2                    -6.031784e+005
  B10__2                                                    -1.744483e-002
  B____2B12__2               4.967693e+001   6.967468e+001
  B12__2                    -2.319546e+004
  B12__2                                                     8.904087e-002
  B____2B14__2              -1.198030e+006   1.281169e+004
  B14__2                     1.237224e+006
  B14__2                                                     8.618294e-006
$VINTAGE,0
C END FDNE
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
C ********************************* Alpha Mode *********************************
  X0018ASZ__1A       .05        1.                                             0
C *********************************  Beta Mode *********************************
  X0018BSZ__1B       .05        1.                                             0
/SOURCE
C *********************************
C Norton equivalent current sources
C *********************************
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
C ********************************* Alpha Mode *********************************
14IN__1A-12.01843989      60.    -86.21                          -1.         1.
14IN__2A-12.79744876      60.    -93.66                          -1.         1.
C *********************************  Beta Mode *********************************
14IN__1B-12.01843989      60.   -176.21                          -1.         1.
14IN__2B-12.79744876      60.   -183.66                          -1.         1.
/OUTPUT
  SZ__1ASZ__2ASZ__1BSZ__2B
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

B.4.2 Balanced Three-Phase to Ground Fault Case

```
BEGIN NEW DATA CASE
C --------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C September 22, 2004
C --------------------------------------------------------
POWER FREQUENCY                           60.
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   2.E-5      .2
       1       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
C ********************************* Alpha Mode *********************************
C *************
C study zone
C *************
  SZ__1ASZTMPA            1.E-9                                                1
  SZ__1A                   200.  130.                                          0
  SZ__2A                   650.  250.                                          0
  SZ__1A                                    8.                                 0
  IN__1A                  1200.  500.                                          0
  IN__2A                  2150.  380.                                          0
  X0018A                     2.                                                0
C HAWK conductor
-1SZTMPAIN__1A               2.  0.00                -2 1
       12       4.09034706490576600000E+02
   2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
   4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
   7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
   1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
   2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
   1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
   7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
   4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
       19       4.00519444461175000000E-04
   1.56976974488308500E-02   5.96931801697355600E-02   7.65710632981902200E-02
   1.23025310978241300E-01   7.74362618176230500E+00   1.15631326792783200E+01
   1.51793395253545000E+01   2.45460099809200000E+01   2.17562524566291500E+02
   4.75157557302470000E+03   7.17277231675696400E+03   2.15420855113000800E+02
   1.92780440925090000E+05   1.00130204456972800E+05   6.67694141945910300E+05
   4.49850806862179600E+09  -4.59220311195214400E+09   3.24980116228040600E+09
  -3.15707914037557500E+09
   5.46006799865154000E+00   2.06382999204778400E+01   2.69410363626236900E+01
   4.36617508325875400E+01   2.72889153410192200E+03   3.96969851665451300E+03
   5.15623359578923200E+03   1.03865827369060600E+04   9.26732459557961600E+03
   1.12469458616561000E+05   6.93234200200993500E+04   4.98920839522332300E+05
   4.15754453085636900E+05   7.76422380585483400E+05   1.43944837446228700E+06
   4.02521957768449700E+06   4.02924479726218400E+06   4.38129540251903900E+06
   4.38567669792155100E+06
  1.00000000
  0.00000000
C DRAKE conductor
-1SZTMPASZ__2A               2.  0.00                -2 1
       11       3.93897327703557300000E+02
   2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
   9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
   6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
```

```
  5.35709730600259600E+04  2.76198969356064600E+06
  2.56520429418621500E+00  6.23867915240623700E+00  1.37315501948703500E+01
  2.44411566219346600E+01  3.88234599365499000E+01  5.22046486070647100E+01
  8.94985384307681800E+01  1.61248544547484800E+02  6.10024281549984200E+03
  2.24249725179149800E+04  1.15971094692565100E+06
    19        4.53915497928795800000E-04
  1.71275917005395300E-02  3.98691337390333500E-02  1.60864260029983000E-01
  6.89003618290678800E+00  8.78632925717092800E+00  1.18708400084055100E+01
  1.46650428128307100E+01  4.55132169053748200E+01  1.92681981165319200E+02
  2.21326495567518500E+03  1.11000476132130400E+04 -9.21432832870185500E+04
  2.81776544230162200E+05  1.67841941653263700E+04  1.22535036976167300E+06
 -3.16876240455654700E+07  2.93464632821155900E+07  3.77888719163484300E+07
 -3.68930729156446700E+07
  6.29920182790418300E+00  1.47535442461695500E+01  5.98892062931638800E+01
  2.46878104720658700E+03  3.24688225671564200E+03  4.20547568033697400E+03
  5.30166238127239600E+03  1.66566947044242800E+04  8.86571988667755600E+03
  4.65746128476995800E+04  1.13398074824504100E+05  4.43515608473281600E+05
  3.98417841600631700E+05  5.62068679963815700E+05  1.44682872910978600E+06
  2.87151967320063000E+06  2.87439119287383300E+06  3.65427188965252700E+06
  3.65792616154218300E+06
 1.00000000
 0.00000000
C HAWK conductor
-1SZ__2AIN__2A                2.  0.00                -2 1
    12        4.09034706490576600000E+02
  2.45890513600235700E+03  1.00691916473651100E+03  3.73552293959034800E+03
  4.94910864283924900E+03  1.80922017453526000E+03  1.62638997544022700E+03
  7.23590211201397200E+02  9.98061571653958900E+02  5.16274050708064800E+02
  1.17955959812405300E+04  6.12653756748480700E+04  2.42883914569856300E+06
  2.53613213618031100E+00  3.67827403252360000E+00  7.35825287386416900E+00
  1.84026248477307900E+01  3.58022844932382000E+01  5.94599892702880700E+01
  7.78217102800857800E+01  1.29727612927171000E+02  2.26805464551456900E+02
  4.14424917798641100E+03  2.15580737175207600E+04  8.57837925748202800E+05
    19        5.50554384535761800000E-04
  9.79465605855859600E-03  4.06212733123785000E-02  7.44854903179096900E-02
  4.00229870144877300E-02  6.56254564933406700E-02  7.79345690027440800E-02
  7.11151466287055800E-01  6.74005236856138300E+00  8.01620767358119800E+00
  1.17220670074905700E+01  3.50297130275519900E+01  7.05494675388751300E+01
  7.13600486817041200E+02  4.43545346148581900E+03  6.11781278301569400E+04
  6.79300571244501800E+04  7.49764722063926500E+05  2.36469459307105400E+07
 -2.45311009688204900E+07
  3.94587715068155200E+00  1.57035953406104900E+01  3.03393666019717600E+01
  1.67066122294504600E+01  2.60870054659298800E+01  3.32492107284800500E+01
  2.91988641610697900E+02  2.66461815651150000E+03  3.31965063922072700E+03
  4.56609698304445300E+03  7.47335672283102600E+03  6.69903629701688300E+03
  1.75902539791774200E+04  5.33686195521447800E+04  2.15476314218399300E+05
  3.13824776054083900E+05  9.37530861396163600E+05  1.82325148034484500E+06
  1.82507473182518700E+06
 1.00000000
 0.00000000
C *************
C surface layer
C *************
C DRAKE conductor
-1IN__1AA____1                2.  0.00                -2 1
     8        3.75023834778916070OE+002
  5.118235030896l829E+005 -7.722040015021633OE+004  5.383608138738l419E+005
 -1.313078711l864449E+006  1.41068547587384 80E+006 -8.7056744342053682E+005
  3.9738886193076416E+005 -8.1086902030084631E+004
  1.0945782546902021E+004  8.9498538430768178E+001  5.2204648607064712E+001
  3.6651709608150732E+001  2.5397350977380938E+001  1.3731550194870350E+001
  6.0037967902148388E+000  2.6655609959826490E+000
    10        5.00972553527526250OE-004
  1.6270863906469899E+008 -1.6431596001157719E+008  1.4214204455473491E+006
  1.0885261434708411E+005  6.4174528600579331E+004  1.1562808837095919E+004
  1.0498665383127150E+003  1.2301409456296111E+002  1.3744260229970391E+002
  2.2630797117004681E-001
  2.1851996518619689E+006  2.1830166352267410E+006  1.2797282522085649E+006
  4.1814911890264828E+005  2.5794236314651690E+005  1.1288345220778709E+005
  2.5365200913558881E+004  5.8946984100303853E+003  1.3005384419092390E+004
  2.2225616072406790E+001
 1.00000000
 0.00000000
```

```
C DRAKE conductor
-1IN__2AA____2                2.  0.00                -2 1
     8        3.93903297335005560OE+002
  1.0511045492931569E+005  2.4164547148414579E+002 -1.5017186955810123E+004
  6.3476046493852729E+004 -8.0135889472332128E+004  4.9970378687677192E+004
 -8.5537264822298675E+003  3.3880808573122692E+002
  1.0945782546902021E+004  8.9498538430768178E+001  5.2204648607064712E+001
  3.6651709608150732E+001  2.5397350977380938E+001  1.3731550194870350E+001
  6.0037967902148388E+000  2.6655609959826490E+000
    12        4.00426050961204500OE-004
  2.9821418453288162E+008 -3.0044193674812359E+008  1.9495373574076470E+006
  1.8119051123979481E+005  4.5287095013004116E+004  4.9416715953395367E+004
  1.8523396032005680E+003  3.6333402652528878E+002  5.8013994064875767E+001
  3.9268948601835461E+001  7.5171522529347543E+000  6.1905967511505823E-002
  3.1923605038688411E+006  3.1891713325363020E+006  1.8517660946908800E+006
  6.0844300818797620E+005  3.0829532807651232E+005  2.4818127081645271E+005
  4.0789847713431889E+004  1.5961619255950680E+004  9.9200042766566548E+003
  6.7969813906110430E+003  1.3438258169018341E+003  1.1100966125235569E+001
 1.00000000
 0.00000000
C HAWK conductor
-1IN__2AA____1                2.  0.00                -2 1
    12        4.09033489452554310OE+002
  2.4288461303353016E+006  6.1273663687205080E+004  1.1791906944042370E+004
  5.1956569709320877E+002  3.3236418590387979E+002  5.7164357367498978E+003
  3.2634004169326912E+004 -1.5586524316327844E+005  4.8412147680448723E+005
 -1.2384748760169270E+006  1.3944189855436631E+006 -5.0557916661077325E+005
  8.5783792574820283E+005  2.1558073717520761E+004  4.1442491779864113E+003
  2.2680546455145691E+002  1.2972761292717101E+002  7.7821710280085782E+001
  5.9459989270288069E+001  3.5802284493238197E+001  1.8402624847730792E+001
  7.3582528738641688E+000  3.6782740325235999E+000  2.5361321361803109E+000
    11        1.33553858593508000OE-003
 -1.7748345869520440E+009  1.7507732888229799E+009  2.4023866400058411E+007
  2.1781893643545041E+004  1.4510008190993110E+004  5.9933233243698578E+002
  5.3905705626991960E+002  9.0291025531296631E-001  2.8882885870059560E-001
  1.8241089263411739E-001  6.3630822885285973E-002
  4.9455971431356872E+005  4.9406564866490458E+005  5.2859557145165885E+005
  9.3790496018900856E+004  5.8281852345950239E+004  7.0495362844468582E+003
  1.4090256362095870E+004  9.2024301665763190E+001  2.9266337352851991E+001
  1.7828841875301030E+001  6.1809013182452661E+000
 1.00000000
 0.00000000
C ******************************  Beta Mode ******************************
C *************
C study zone
C *************
 SZ__1BSZTMPB               1.E-9                                        1
 SZ__1B                      200.  130.                                  0
 SZ__2B                      650.  250.                                  0
 SZ__1B                                  8.                              0
 IN__1B                     1200.  500.                                  0
 IN__2B                     2150.  380.                                  0
 X0018B                        2.                                        0
C HAWK conductor
-1SZTMPBIN__1B                2.  0.00                -2 1
    12        4.09034706490576600000E+02
  2.45890513600235700E+03  1.00691916473651100E+03  3.73552293959034800E+03
  4.94910864283924900E+03  1.80922017453526000E+03  1.62638997544022700E+03
  7.23590211201397200E+02  9.98061571653958900E+02  5.16274050708064800E+02
  1.17955959812405300E+04  6.12653756748480700E+04  2.42883914569856300E+06
  2.53613213618031100E+00  3.67827403252360000E+00  7.35825287386416900E+00
  1.84026248477307900E+01  3.58022844932382000E+01  5.94599892702880700E+01
  7.78217102800857800E+01  1.29727612927171000E+02  2.26805464551456900E+02
  4.14424917798641100E+03  2.15580737175207600E+04  8.57837925748202800E+05
    19        4.00519444461175000000E-04
  1.56976974488308500E-02  5.96931801697355600E-02  7.65710632981902200E-02
  1.23025310978241300E-01  7.74362618176230500E+00  1.15631326792783200E+01
  1.51793395253545000E+01  2.45460099809200000E+01  2.17562524566291500E+02
  4.75157557302470000E+03  7.17277231675696400E+03  2.15420855113000800E+02
  1.92780440925090000E+05  1.00130204456972800E+05  6.67694141945910300E+05
  4.49850806862179600E+09 -4.59220311195214400E+09  3.24980116228040600E+09
 -3.15707914037557500E+09
  5.46006799865154000E+00  2.06382999204778400E+01  2.69410363626236900E+01
```

```
    4.36617508325875400E+01   2.72889153410192200E+03   3.96969851665451300E+03
    5.15623359578923200E+03   1.03865827369060600E+04   9.26732459557961600E+03
    1.12469458616561000E+05   6.93234200200993500E+04   4.98920839522332300E+05
    4.15754453085636900E+05   7.76422380585483400E+05   1.43944837446228700E+06
    4.02521957768449700E+06   4.02924479726218400E+06   4.38129540251903900E+06
    4.38567669792155100E+06
  1.00000000
  0.00000000
C DRAKE conductor
-1SZTMPBSZ__2B                2.  0.00                 -2 1
      11        3.93897327703557300000E+02
    2.27109078735675900E+03   2.46873836980220700E+03   2.76728330884469200E+03
    9.27201327857635600E+02   8.13652385489715700E+02   4.84386453347630200E+02
    6.00660454747077000E+02   3.19123006877189800E+02   1.47413498397660100E+04
    5.35709730600259600E+04   2.76198969356064600E+06
    2.56520429418621500E+00   6.23867915240623700E+00   1.37315501948703500E+01
    2.44411566219346600E+01   3.88234599365499000E+01   5.22046486070647100E+01
    8.94985384307681800E+01   1.61248544547484800E+02   6.10024281549984200E+03
    2.24249725179149800E+04   1.15971094692565100E+06
      19        4.53915497928795800000E-04
    1.71275917005395300E-02   3.98691337390333500E-02   1.60864260029983000E-01
    6.89003618290678800E+00   8.78632925717092800E+00   1.18708400084055100E+01
    1.46650428128307100E+01   4.55132169053748200E+01   1.92681981165319200E+02
    2.21326495567518500E+03   1.11000476132130400E+04  -9.21432832870185500E+04
    2.81776544230162200E+05   1.67841941653263700E+04   1.22535036976167300E+06
   -3.16876240455654700E+07   2.93464632821155900E+07   3.77888719163484300E+07
   -3.68930729156446700E+07
    6.29920182790418300E+00   1.47535442461695500E+01   5.98892062931638800E+01
    2.46878104720658700E+03   3.24688225671564200E+03   4.20547568033697400E+03
    5.30166238127239600E+03   1.66566947044242800E+04   8.86571988667755600E+03
    4.65746128476995800E+04   1.13398074824504100E+05   4.43515608473281600E+05
    3.98417841600631700E+05   5.62068679963815700E+05   1.44682872910978600E+06
    2.87151967320063000E+06   2.87439119287383300E+06   3.65427188965252700E+06
    3.65792616154218300E+06
  1.00000000
  0.00000000
C HAWK conductor
-1SZ__2BIN__2B                2.  0.00                 -2 1
      12        4.09034706490576600000E+02
    2.45890513600235700E+03   1.00691916473651100E+03   3.73552293959034800E+03
    4.94910864283924900E+03   1.80922017453526000E+03   1.62638997544022700E+03
    7.23590211201397200E+02   9.98061571653958900E+02   5.16274050708064800E+02
    1.17955959812405300E+04   6.12653756748480700E+04   2.42883914569856300E+06
    2.53613213618031100E+00   3.67827403252360000E+00   7.35825287386416900E+00
    1.84026248477307900E+01   3.58022844932382000E+01   5.94599892702880700E+01
    7.78217102800857800E+01   1.29727612927171000E+02   2.26805464551456900E+02
    4.14424917798641100E+03   2.15580737175207600E+04   8.57837925748202800E+05
      19        5.50554384535761800000E-04
    9.79465605855859600E-03   4.06212733123785000E-02   7.44854903179096900E-02
    4.00229870144877300E-02   6.56254564933406700E-02   7.79345690027440800E-02
    7.11151466287055800E-01   6.74005236856138300E+00   8.01620767358119800E+00
    1.17220670074905700E+01   3.50297130275519900E+01   7.05494675388751300E+01
    7.13600486817041200E+02   4.43545346148581900E+03   6.11781278301569400E+04
    6.79300571244501800E+04   7.49764722063926500E+05   2.36469459307105400E+07
   -2.45311009688204900E+07
    3.94587715068155200E+00   1.57035953406104900E+01   3.03393666019717600E+01
    1.67066122294504600E+01   2.60870054659298800E+01   3.32492107284800500E+01
    2.91988641610697900E+02   2.66461815651150000E+03   3.31965063922072700E+03
    4.56609698304445300E+03   7.47335672283102600E+03   6.69903629701688300E+03
    1.75902539791774200E+04   5.33686195521447800E+04   2.15476314218399300E+05
    3.13824776054083900E+05   9.37530861396163600E+05   1.82325148034484500E+06
    1.82507473182518700E+06
  1.00000000
  0.00000000
C *************
C surface layer
C *************
C DRAKE conductor
-1IN__1BB____1                2.  0.00                 -2 1
       8        3.75023834778916070 0E+002
    5.1182350308961829E+005  -7.7220400150216330E+004   5.3836081387381419E+005
   -1.3130787111864449E+006   1.4106854758738480E+006  -8.7056744342053682E+005
    3.9738886193076416E+005  -8.1086902030084631E+004
```

```
    1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
    3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
    6.0037967902148388E+000   2.6655609959826490E+000
      10        5.0097255352752625 00E-004
    1.6270863906469899E+008  -1.6431596001157719E+008   1.4214204455473491E+006
    1.0885261434708411E+005   6.4174528600579331E+004   1.1562808837095919E+004
    1.0498665383127150E+003   1.2301409456296111E+002   1.3744260229970391E+002
    2.2630797117004681E-001
    2.1851996518619689E+006   2.1830166352267410E+006   1.2797282522085649E+006
    4.1814911890264828E+005   2.5794236314651690E+005   1.1288345220778709E+005
    2.5365200913558881E+004   5.8946984100303853E+003   1.3005384419092390E+004
    2.2225616072406790E+001
  1.00000000
  0.00000000
C DRAKE conductor
-1IN__2BB____2                2.  0.00                 -2 1
       8        3.9390329733500556 00E+002
    1.0511045492931569E+005   2.4164547148414579E+002  -1.5017186955810123E+004
    6.3476046493852729E+004  -8.0135889472332128E+004   4.9970378687677192E+004
   -8.5537264822298675E+003   3.3880808573122692E+002
    1.0945782546902021E+004   8.9498538430768178E+001   5.2204648607064712E+001
    3.6651709608150732E+001   2.5397350977380938E+001   1.3731550194870350E+001
    6.0037967902148388E+000   2.6655609959826490E+000
      12        4.0042605096120450 00E-004
    2.9821418453288162E+008  -3.0044193674812359E+008   1.9495373574076470E+006
    1.8119051123979481E+005   4.5287095013004116E+004   4.9416715953395367E+004
    1.8523396032005680E+003   3.6333402652528878E+002   5.8013994064875767E+001
    3.9268948601835461E+001   7.5171522529347543E+000   6.1905967511505823E-002
    3.1923605038688411E+006   3.1891713325363020E+006   1.8517660946908800E+006
    6.0844300818797620E+005   3.0829532807651232E+005   2.4818127081645271E+005
    4.0789847713431889E+004   1.5961619255950680E+004   9.9200042766566548E+003
    6.7969813906110430E+003   1.3438258169018341E+003   1.1100966125235569E+001
  1.00000000
  0.00000000
C HAWK conductor
-1IN__2BB____1                2.  0.00                 -2 1
      12        4.0903348945255431 00E+002
    2.4288461303353016E+006   6.1273663687205080E+004   1.1791906944042370E+004
    5.1956569709320877E+002   3.3236418590387979E+002   5.7164357367498978E+003
    3.2634004169326912E+004  -1.5586524316327844E+005   4.8412147680448723E+005
   -1.2384748760169270E+006   1.3944189855436631E+006  -5.0557916661077325E+005
    8.5783792574820283E+005   2.1558073717520761E+004   4.1442491779864113E+003
    2.2680546455145691E+002   1.2972761292717101E+002   7.7821710280085782E+001
    5.9459989270288069E+001   3.5802284493238197E+001   1.8402624847730792E+001
    7.,3582528738641688E+000   3.6782740325235999E+000   2.5361321361803109E+000
      11        1.3355385859350800 00E-003
   -1.7748345869520440E+009   1.7507732888229799E+009   2.4023866400058411E+007
    2.1781893643545041E+004   1.4510008190993110E+004   5.9933233243698578E+002
    5.3905705626991960E+002   9.0291025531296631E-001   2.8882885870059560E-001
    1.8241089263411739E-001   6.3630822885285973E-002
    4.9455971431356872E+005   4.9406564866490458E+005   5.2859557145165885E+005
    9.3790496018900856E+004   5.8281852345950239E+004   7.0495362844468582E+003
    1.4090256362095870E+004   9.2024301665763190E+001   2.9266337352851991E+001
    1.7828841875301030E+001   6.1809013182452661E+000
  1.00000000
  0.00000000
C ********************************* Alpha Mode *********************************
C *************
C deep region
C *************
C BEGIN FDNE
$VINTAGE,1
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><   microF     >
C
C (1,1)
  A_____1                     1.179990e+002
  A_____1                    -1.817485e+001  -1.663768e+003
  A_____1                     1.139488e+001   2.460232e+002
  A_____1                     1.235152e+001   1.618993e+002
  A_____1                     1.198652e+003   5.511484e+002
  A_____1                    -1.989105e+002  -1.174363e+001
  A_____1A 6__1               6.337633e+000   6.022723e+001
  A 6__1                      4.738628e+002
```

```
A 6__1                                                  1.181832e+001
A____1A 8__1          1.102904e+001   2.229382e+002
A 8__1                2.072046e+004
A 8__1                                                  8.153085e-001
A____1A A__1          1.013795e+001   7.765019e+001
A A__1                1.249981e+006
A A__1                                                  7.261159e-001
A____1A C__1          2.569920e+002   3.941270e+002
A C__1               -1.787782e+004
A C__1                                                  1.139597e-001
A____1A E__1         -1.620042e+002   2.269678e+002
A E__1                1.168918e+004
A E__1                                                  8.943689e-002
A____1A10__1          2.320907e+004   1.290952e+003
A10__1               -2.935271e+004
A10__1                                                  1.945724e-003
A____1A12__1          4.713920e+001   6.899455e+001
A12__1               -2.446764e+004
A12__1                                                  8.993800e-002
A____1A14__1         -2.288147e+006  -1.622627e+004
A14__1                2.321225e+006
A14__1                                                 -3.060986e-006
C (1,2)
A____1A____2         -6.862856e+003
A____1A____2          5.790700e+001   5.300940e+003
A____1A____2         -3.452911e+001  -7.455070e+002
A____1A____2          1.627967e+001   2.133881e+002
A____1A____2          3.760290e+003   1.729007e+003
A____1A____2          2.107372e+004   1.244187e+003
A____1A 6_12          3.509147e+004  -5.910586e+004
A 6_12A____2         -1.309887e+005
A 6_12A____2                                           -8.700025e-003
A____1A 8_12          2.942102e+002  -1.138949e+003
A 8_12A____2         -1.737797e+004
A 8_12A____2                                           -1.568034e-001
A____1A A_12         -2.195788e+001  -1.550764e+002
A A_12A____2          2.769217e+005
A A_12A____2                                           -3.635504e-001
A____1A C_12         -1.356027e+003   9.905119e+002
A C_12A____2          1.556051e+004
A C_12A____2                                            4.199693e-002
A____1A E_12          2.534693e+003  -8.123033e+002
A E_12A____2         -1.426832e+004
A E_12A____2                                           -2.083931e-002
A____1A10_12         -3.480429e+003   5.651609e+002
A10_12A____2          1.058648e+004
A10_12A____2                                            1.425341e-002
A____1A12_12         -1.427467e+002  -1.380378e+002
A12_12A____2          2.771293e+004
A12_12A____2                                           -4.480791e-002
A____1A14_12         -1.835352e+002   1.407996e+002
A14_12A____2          2.579641e+004
A14_12A____2                                            2.457868e-002
C (2,2)
A____2                2.136131e+002
A____2               -1.697350e+001  -1.553794e+003
A____2                1.209292e+001   2.610945e+002
A____2                2.347825e+001   3.077445e+002
A____2                5.817348e+002   2.674856e+002
A____2               -4.503580e+002  -2.658903e+001
A____2A 6__2          1.269424e+005  -4.880319e+004
A 6__2               -1.510270e+005
A 6__2                                                 -2.295168e-003
A____2A 8__2         -2.646082e+002   8.977345e+002
A 8__2                1.251412e+004
A 8__2                                                  1.980824e-001
A____2A A__2          9.729605e+000   7.751308e+001
A A__2                2.239360e+005
A A__2                                                  7.274259e-001
A____2A C__2          4.703554e+002  -1.343231e+003
A C__2               -5.811050e+004
A C__2                                                 -3.365080e-002
A____2A E__2         -3.314680e+003   6.780535e+002
A E__2                9.734108e+003
A E__2                                                  2.002066e-002
A____2A10__2         -2.573017e+002  -6.882284e+002
A10__2               -6.031784e+005
A10__2                                                 -1.744483e-002
A____2A12__2          4.967693e+001   6.967468e+001
A12__2               -2.319546e+004
A12__2                                                  8.904087e-002
A____2A14__2         -1.198030e+006   1.281169e+004
A14__2                1.237224e+006
A14__2                                                  8.618294e-006
C ******************************** Beta Mode ********************************
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><   microF      >
C
C (1,1)
B____1                1.179990e+002
B____1               -1.817485e+001  -1.663768e+003
B____1                1.139488e+001   2.460232e+002
B____1                1.235152e+001   1.618993e+002
B____1                1.198652e+003   5.511484e+002
B____1               -1.989105e+002  -1.174363e+001
B____1B 6__1          6.337633e+000   6.022723e+001
B 6__1                4.738628e+002
B 6__1                                                  1.181832e+001
B____1B 8__1          1.102904e+001   2.229382e+002
B 8__1                2.072046e+004
B 8__1                                                  8.153085e-001
B____1B A__1          1.013795e+001   7.765019e+001
B A__1                1.249981e+006
B A__1                                                  7.261159e-001
B____1B C__1          2.569920e+002   3.941270e+002
B C__1               -1.787782e+004
B C__1                                                  1.139597e-001
B____1B E__1         -1.620042e+002   2.269678e+002
B E__1                1.168918e+004
B E__1                                                  8.943689e-002
B____1B10__1          2.320907e+004   1.290952e+003
B10__1               -2.935271e+004
B10__1                                                  1.945724e-003
B____1B12__1          4.713920e+001   6.899455e+001
B12__1               -2.446764e+004
B12__1                                                  8.993800e-002
B____1B14__1         -2.288147e+006  -1.622627e+004
B14__1                2.321225e+006
B14__1                                                 -3.060986e-006
C (1,2)
B____1B____2         -6.862856e+003
B____1B____2          5.790700e+001   5.300940e+003
B____1B____2         -3.452911e+001  -7.455070e+002
B____1B____2          1.627967e+001   2.133881e+002
B____1B____2          3.760290e+003   1.729007e+003
B____1B____2          2.107372e+004   1.244187e+003
B____1B 6_12          3.509147e+004  -5.910586e+004
B 6_12B____2         -1.309887e+005
B 6_12B____2                                           -8.700025e-003
B____1B 8_12          2.942102e+002  -1.138949e+003
B 8_12B____2         -1.737797e+004
B 8_12B____2                                           -1.568034e-001
B____1B A_12         -2.195788e+001  -1.550764e+002
B A_12B____2          2.769217e+005
B A_12B____2                                           -3.635504e-001
B____1B C_12         -1.356027e+003   9.905119e+002
B C_12B____2          1.556051e+004
B C_12B____2                                            4.199693e-002
B____1B E_12          2.534693e+003  -8.123033e+002
B E_12B____2         -1.426832e+004
B E_12B____2                                           -2.083931e-002
B____1B10_12         -3.480429e+003   5.651609e+002
B10_12B____2          1.058648e+004
B10_12B____2                                            1.425341e-002
B____1B12_12         -1.427467e+002  -1.380378e+002
B12_12B____2          2.771293e+004
B12_12B____2                                           -4.480791e-002
```

```
  B_____1B14_12                -1.835352e+002   1.407996e+002
  B14_12B_____2                 2.579641e+004
  B14_12B_____2                                                  2.457868e-002
C (2,2)
  B_____2                       2.136131e+002
  B_____2                      -1.697350e+001  -1.553794e+003
  B_____2                       1.209292e+001   2.610945e+002
  B_____2                       2.347825e+001   3.077445e+002
  B_____2                       5.817348e+002   2.674856e+002
  B_____2                      -4.503580e+002  -2.658903e+001
  B_____2B 6__2                 1.269424e+005  -4.880319e+004
  B 6__2                       -1.510270e+005
  B 6__2                                                        -2.295168e-003
  B_____2B 8__2                -2.646082e+002   8.977345e+002
  B 8__2                        1.251412e+004
  B 8__2                                                         1.980824e-001
  B_____2B A__2                 9.729605e+000   7.751308e+001
  B A__2                        2.239360e+005
  B A__2                                                         7.274259e-001
  B_____2B C__2                 4.703554e+002  -1.343231e+003
  B C__2                       -5.811050e+004
  B C__2                                                        -3.365080e-002
  B_____2B E__2                -3.314680e+003   6.780535e+002
  B E__2                        9.734108e+003
  B E__2                                                         2.002066e-002
  B_____2B10__2                -2.573017e+002  -6.882284e+002
  B10__2                       -6.031784e+005
  B10__2                                                        -1.744483e-002
  B_____2B12__2                 4.967693e+001   6.967468e+001
  B12__2                       -2.319546e+004
  B12__2                                                         8.904087e-002
```

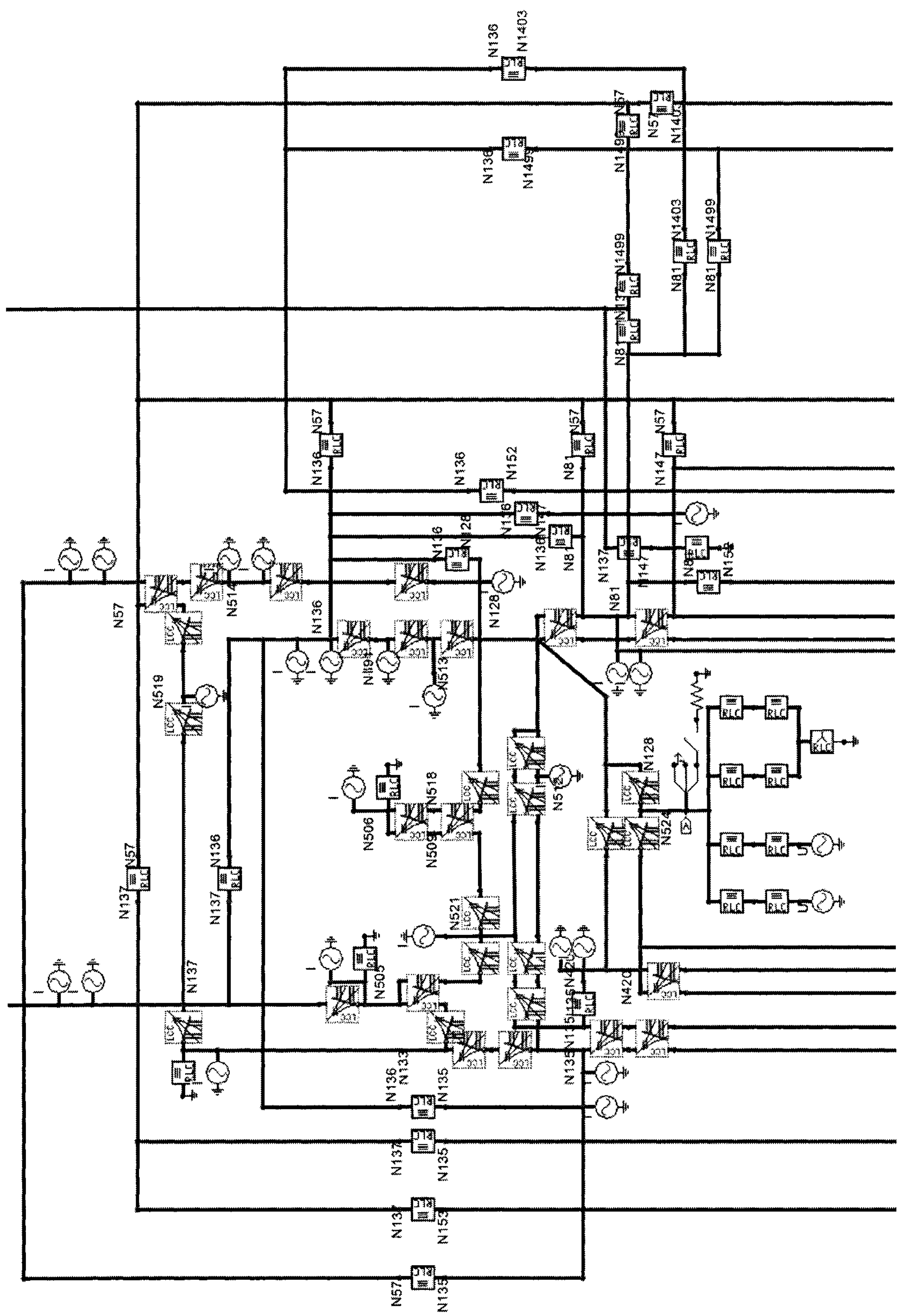
```
  B_____2B14__2                -1.198030e+006   1.281169e+004
  B14__2                        1.237224e+006
  B14__2                                                         8.618294e-006
$VINTAGE,0
C END FDNE
/SWITCH
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
C ********************************** Alpha Mode ************************************
  X0018ASZ__2A       .05        .15                                                 0
C **********************************  Beta Mode ************************************
  X0018BSZ__2B       .05        .15                                                 0
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
C **********************************
C Norton equivalent current sources
C **********************************
C ********************************** Alpha Mode ************************************
14IN__1A-12.01843989      60.    -86.21                                -1.        1.
14IN__2A-12.79744876      60.    -93.66                                -1.        1.
C **********************************  Beta Mode ************************************
14IN__1B-12.01843989      60.   -176.21                                -1.        1.
14IN__2B-12.79744876      60.   -183.66                                -1.        1.
/OUTPUT
  SZ__1ASZ__2ASZ__1BSZ__2B
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

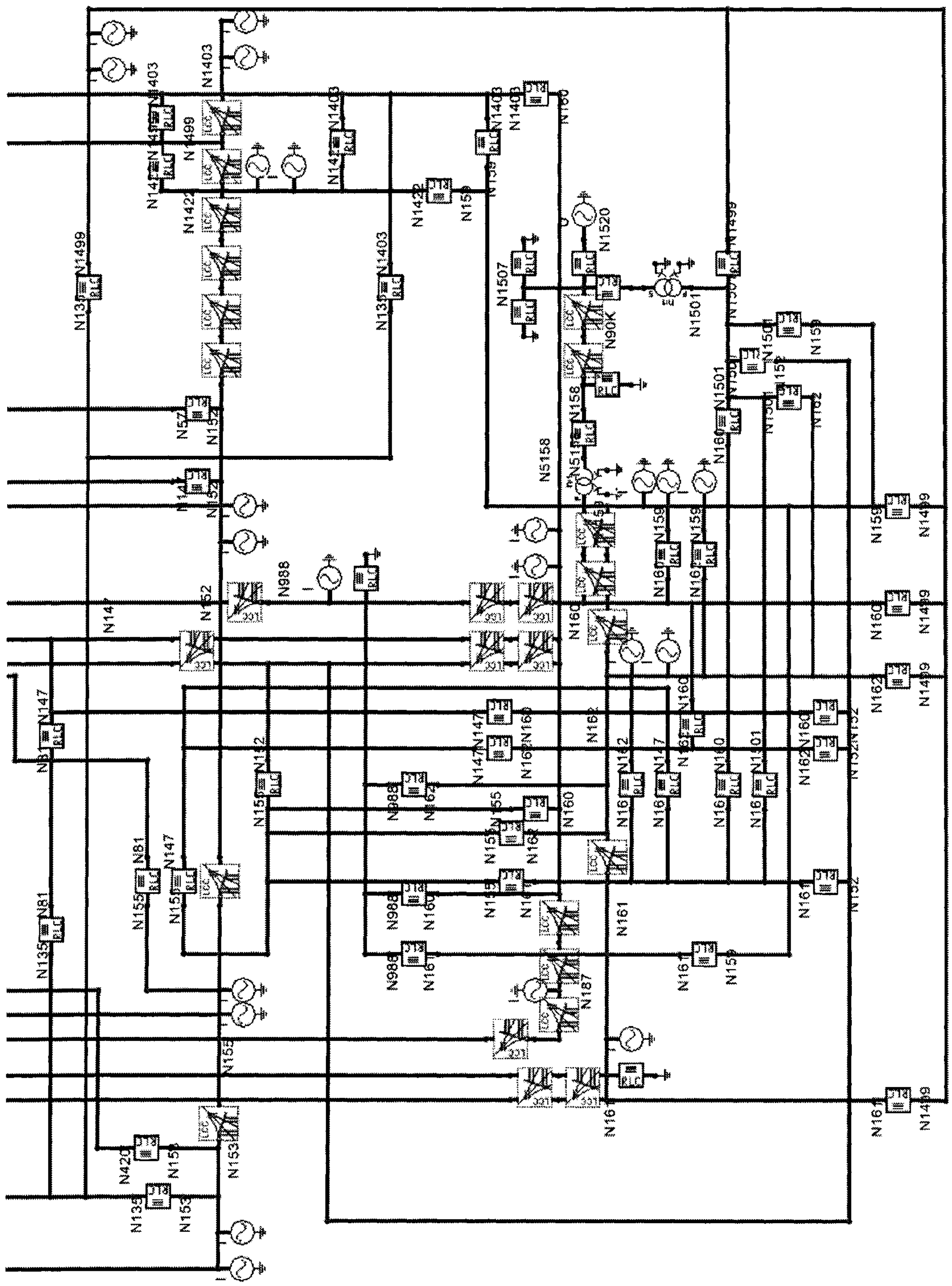
C

AIES Area 50 Backbone

C.1 AIES Area 50 Backbone Diagram in ATP

The diagram in ATPDraw is split into two figures, which are the upper and lower parts.




N1403
N1499
N1520
N1507
N1501
N90K
N5158
N988
N160
N161
N187

C.2 PSS/E Procedures in Obtaining Equivalents for Area 50 Backbone

```
                 POWER TECHNOLOGIES INCORPORATED

           4000 BUS POWER SYSTEM SIMULATOR--PSS/E-26.1

 INITIATED AT LOAD FLOW ENTRY POINT ON SUN, JUL 31 2005  20:42

 Executing activity read

read
 ENTER INPUT FILE NAME (0 TO EXIT, 1 FOR TERMINAL): d:\aies\aies.raw

 ENTER IC, SBASE

 ENTER TWO LINE HEADING

 ENTER BUS DATA

 ENTER LOAD DATA

 ENTER GENERATOR DATA

 MESSAGES FOR MACHINE 1  AT BUS  4185 [SCOTF GT13.800]:
  WARNING: MACHINE IS OFF-LINE--BUS TYPE CODE IS  1

 MESSAGES FOR MACHINE 1  AT BUS  4187 [CECGT   18.000]:
  WARNING: MACHINE IS OFF-LINE--BUS TYPE CODE IS  1

 MESSAGES FOR MACHINE 2  AT BUS  7185 [SCOT4 ST13.800]:
  WARNING: MACHINE IS OFF-LINE--BUS TYPE CODE IS  1

 ENTER BRANCH DATA

 ENTER TRANSFORMER ADJUSTMENT DATA

 ENTER AREA INTERCHANGE DATA

 ENTER TWO-TERMINAL DC LINE DATA

 ENTER SWITCHED SHUNT DATA

 ENTER TRANSFORMER IMPEDANCE CORRECTION DATA

 ENTER MULTI-TERMINAL DC LINE DATA

 ENTER MULTI-SECTION LINE DATA

 ENTER ZONE NAME DATA

 ENTER INTER-AREA TRANSFER DATA

 ENTER OWNER NAME DATA

 ENTER FACTS CONTROL DEVICE DATA

 ACTIVITY?
 Executing activity fnsl

fnsl

 ORDERING NETWORK
 DIAGONALS =  1686  OFF-DIAGONALS =  2466  MAX SIZE =  3694

 ENTER ITERATION NUMBER FOR VAR LIMITS
  0 FOR IMMEDIATELY, -1 TO IGNORE COMPLETELY:

 ITER   DELTAP    BUS      DELTAQ    BUS      DELTA/V/   BUS   DELTAANG   BUS
   0    0.0237(  229)     0.0548( 3069)      0.05482( 3069)   0.01289( 4264)
   1    0.0010(   69)     0.2816( 3069)      0.04936( 3069)   0.01318( 4264)
   2    0.0011(  111)     0.0134( 3069)      0.00278( 3069)   0.00065(19124)
   3    0.0005(  299)     0.0020(  825)      0.00030(  379)   0.00033(  299)
   4    0.0005(  299)     0.0010(  924)

 REACHED TOLERANCE IN   4 ITERATIONS

 LARGEST MISMATCH:    0.03 MW   -0.12 MVAR      0.12 MVA-BUS   825 [CGETAP  138.00]
 SYSTEM TOTAL ABSOLUTE MISMATCH:                2.75 MVA

 SWING BUS SUMMARY:
  BUS X--- NAME ---X      PGEN     PMAX    PMIN       QGEN     QMAX    QMIN
  1520 WSCC GEN500.00     177.0 158154.7-87991.2     -139.5  63541.1-48593.3

 ACTIVITY?
 Executing activity eeqv,area

eeqv,area

 USER SPECIFIES SUBSYSTEM TO BE EQUIVALENCED
 ENTER UP TO 20 AREA NUMBERS
4,6,13,17,18,19,20,21,22,23,24,25,26,27,28,29,30
 ENTER UP TO 20 AREA NUMBERS
31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49
 ENTER UP TO 20 AREA NUMBERS
52,53,54,55,56,57,60,88,89,91,92,97
 ENTER UP TO 20 AREA NUMBERS


 ENTER 1 TO RETAIN AREA BOUNDARY BUSES:

 ENTER 1 TO RETAIN ZONE BOUNDARY BUSES:

 ENTER 1 TO SUPPRESS EQUIVALENCING OF PHASE SHIFTERS:

 ENTER 1 TO RETAIN BUSES CONTROLLED BY REMOTE GENERATION OR SWITCHED SHUNT:

 ENTER MINIMUM GENERATION FOR RETAINING GENERATOR BUSES
   (CARRIAGE RETURN TO KEEP ALL ON-LINE GENERATOR BUSES): 1000

 ENTER 1 TO RETAIN EXISTING BRANCHES BETWEEN RETAINED BUSES: 1
 1492 RADIAL AND TWO POINT BUSES EQUIVALENCED

 ENTER BRANCH THRESHOLD TOLERANCE:
 DIAGONALS =   155  OFF-DIAGONALS =    742  MAX SIZE =  1045

 ENTER 1 TO NET LOAD AND SHUNT AT RETAINED BUSES:

 ACTIVITY?
 Executing activity eeqv

eeqv

 USER SPECIFIES SUBSYSTEM TO BE EQUIVALENCED
 ENTER UP TO 20 BUS NUMBERS
36,40,41,63,87,89,90,99,109,117,118,129,130,131,132,145,146,148
 ENTER UP TO 20 BUS NUMBERS
151,154,156,163,165,202,207,208,281,338,342,345,348,350,374,422,423,424
 ENTER UP TO 20 BUS NUMBERS
491,492,495,496,516,542,545,805,806,943,1260,1280,1484,1489,1497,2144,3150
 ENTER UP TO 20 BUS NUMBERS
3187,4158,4187,5159,5161,5505,5506,5526,6135,6144,6506,6526,10226,10228,10312
 ENTER UP TO 20 BUS NUMBERS
10422,11228,11312,18403,25053,25054,25123,25124,25135,25136,25193,25194
 ENTER UP TO 20 BUS NUMBERS
25213,25214,138,1226,1228,1312,1348,19226
 ENTER UP TO 20 BUS NUMBERS


 ENTER 1 TO RETAIN AREA BOUNDARY BUSES:

 ENTER 1 TO RETAIN ZONE BOUNDARY BUSES:
```

```
ENTER 1 TO RETAIN BUSES CONTROLLED BY REMOTE GENERATION OR SWITCHED SHUNT:

ENTER MINIMUM GENERATION FOR RETAINING GENERATOR BUSES
  (CARRIAGE RETURN TO KEEP ALL ON-LINE GENERATOR BUSES): 1000

ENTER 1 TO RETAIN EXISTING BRANCHES BETWEEN RETAINED BUSES: 1
  54 RADIAL AND TWO POINT BUSES EQUIVALENCED

ENTER BRANCH THRESHOLD TOLERANCE:

MATRIX TOO BIG IN ORDR AT ROW     30--INCREASING MATRIX GROWTH FACTOR TO  2.25

MATRIX TOO BIG IN ORDR AT ROW     30--INCREASING MATRIX GROWTH FACTOR TO  2.50
DIAGONALS =     66  OFF-DIAGONALS =   414  MAX SIZE =   605

ENTER 1 TO NET LOAD AND SHUNT AT RETAINED BUSES:

ACTIVITY?
```

C.3 Area 50 Backbone ATP Data Files

C.3.1 Full Model

```
BEGIN NEW DATA CASE
C -----------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C July 19, 2005
C -----------------------------------------------------
C Alberta Interconnected Electric System 240kV Area 50 Backbone
C
C
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   1.E-5     .15      60.      60.
     500       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
  N1507AN1520A               5.574.175                                         0
  N1507BN1520B               5.574.175                                         0
  N1507CN1520C               5.574.175                                         0
  N1422AN1403A            1645.3 3347.                                         0
  N1422BN1403B            1645.3 3347.                                         0
  N1422CN1403C            1645.3 3347.                                         0
  N137A N57A               11.87 99.23                                         0
  N137B N57B               11.87 99.23                                         0
  N137C N57C               11.87 99.23                                         0
  N81A  N137A              73.45623.32                                         0
  N81B  N137B              73.45623.32                                         0
  N81C  N137C              73.45623.32                                         0
  N135A N137A              55.94275.21                                         0
  N135B N137B              55.94275.21                                         0
  N135C N137C              55.94275.21                                         0
  N137A N136A                66.292.98                                         0
  N137B N136B                66.292.98                                         0
  N137C N136C                66.292.98                                         0
  N153A N137A             1581.54387.3                                         0
  N153B N137B             1581.54387.3                                         0
  N153C N137C             1581.54387.3                                         0
  N137A N1499A            1041.3242.4                                          0
  N137B N1499B            1041.3242.4                                          0
  N137C N1499C            1041.3242.4                                          0
  N81A  N57A              162.3465.31                                          0
  N81B  N57B              162.3465.31                                          0
  N81C  N57C              162.3465.31                                          0
  N135A N57A              38.15171.02                                          0
  N135B N57B              38.15171.02                                          0
  N135C N57C              38.15171.02                                          0
  N136A N57A                9.21 50.75                                         0
  N136B N57B                9.21 50.75                                         0
  N136C N57C                9.21 50.75                                         0
  N147A N57A             2105.33105.4                                          0
  N147B N57B             2105.33105.4                                          0
  N147C N57C             2105.33105.4                                          0
  N152A N57A              3076.3887.3                                          0
  N152B N57B              3076.3887.3                                          0
  N152C N57C              3076.3887.3                                          0
  N1403AN57A              244.51054.3                                          0
  N1403BN57B              244.51054.3                                          0
  N1403CN57C              244.51054.3                                          0
  N1499AN57A              135.6499.92                                          0
  N1499BN57B              135.6499.92                                          0
  N1499CN57C              135.6499.92                                          0
  N81A  N136A             34.12255.58                                          0
  N81B  N136B             34.12255.58                                          0
  N81C  N136C             34.12255.58                                          0
  N135A N136A            286.43773.28                                          0
  N135B N136B            286.43773.28                                          0
  N135C N136C            286.43773.28                                          0
  N147A N136A            733.98 1790.                                          0
  N147B N136B            733.98 1790.                                          0
  N147C N136C            733.98 1790.                                          0
  N152A N136A            1141.32287.8                                          0
  N152B N136B            1141.32287.8                                          0
  N152C N136C            1141.32287.8                                          0
  N1403AN136A            855.062615.2                                          0
  N1403BN136B            855.062615.2                                          0
  N1403CN136C            855.062615.2                                          0
  N1499AN136A            186.84768.65                                          0
  N1499BN136B            186.84768.65                                          0
  N1499CN136C            186.84768.65                                          0
  N506A                                    184.03                              0
  N506B                                    184.03                              0
  N506C                                    184.03                              0
        N133A            2438.616583.                                          0
        N133B            2438.616583.                                          0
        N133C            2438.616583.                                          0
  N505A                                    197.92                              0
  N505B                                    197.92                              0
  N505C                                    197.92                              0
  N135A N1403A           980.243957.6                                          0
  N135B N1403B           980.243957.6                                          0
  N135C N1403C           980.243957.6                                          0
  N152A N147A              3.97 59.55                                          0
  N152B N147B              3.97 59.55                                          0
  N152C N147C              3.97 59.55                                          0
  N128A N136A               .56  3.17                                          0
  N128B N136B               .56  3.17                                          0
  N128C N136C               .56  3.17                                          0
  N135A N81A             295.611145.6                                          0
  N135B N81B             295.611145.6                                          0
  N135C N81C             295.611145.6                                          0
  N153A N135A             84.72283.61                                          0
  N153B N135B             84.72283.61                                          0
  N153C N135C             84.72283.61                                          0
  N135A N420A            305.11836.69                                          0
  N135B N420B            305.11836.69                                          0
  N135C N420C            305.11836.69                                          0
  N135A N1499A           485.391860.5                                          0
  N135B N1499B           485.391860.5                                          0
  N135C N1499C           485.391860.5                                          0
  N153A N420A               84.371.24                                          0
```

```
N153B N420B           84.371.24                 0
N153C N420C           84.371.24                 0
N81A  N147A          101.92616.29               0
N81B  N147B          101.92616.29               0
N81C  N147C          101.92616.29               0
N152A N81A           194.21 802.4               0
N152B N81B           194.21 802.4               0
N152C N81C           194.21 802.4               0
N155A N81A           1363.53550.3               0
N155B N81B           1363.53550.3               0
N155C N81C           1363.53550.3               0
N81A  N1499A         244.09 1335.               0
N81B  N1499B         244.09 1335.               0
N81C  N1499C         244.09 1335.               0
N81A  N1403A          1238.4688.3               0
N81B  N1403B          1238.4688.3               0
N81C  N1403C          1238.4688.3               0
N155A N147A           53.49269.67               0
N155B N147B           53.49269.67               0
N155C N147C           53.49269.67               0
N155A N152A           10.13147.82               0
N155B N152B           10.13147.82               0
N155C N152C           10.13147.82               0
N160A N155A          807.252542.6               0
N160B N155B          807.252542.6               0
N160C N155C          807.252542.6               0
N162A N155A          952.28 3631.               0
N162B N155B          952.28 3631.               0
N162C N155C          952.28 3631.               0
N161A N155A          261.13 1203.               0
N161B N155B          261.13 1203.               0
N161C N155C          261.13 1203.               0
N162A N147A          2025.54586.3               0
N162B N147B          2025.54586.3               0
N162C N147C          2025.54586.3               0
N160A N147A          1602.13173.9               0
N160B N147B          1602.13173.9               0
N160C N147C          1602.13173.9               0
N161A N147A          599.361531.4               0
N161B N147B          599.361531.4               0
N161C N147C          599.361531.4               0
N147A                 2487.      105.81         0
N147B                 2487.      105.81         0
N147C                 2487.      105.81         0
N152A N160A          639.42 1810.               0
N152B N160B          639.42 1810.               0
N152C N160C          639.42 1810.               0
N152A N162A          768.572591.3               0
N152B N162B          768.572591.3               0
N152C N162C          768.572591.3               0
N152A N161A          215.49859.85               0
N152B N161B          215.49859.85               0
N152C N161C          215.49859.85               0
X0895AN1507A           125.  625.               0
X0895BN1507B           125.  625.               0
X0895CN1507C           125.  625.               0
N152A N1501A         1576.94549.4               0
N152B N1501B         1576.94549.4               0
N152C N1501C         1576.94549.4               0
N159A N1422A         397.432121.8               0
N159B N1422B         397.432121.8               0
N159C N1422C         397.432121.8               0
N1422AN1499A         132.34 694.6               0
N1422BN1499B         132.34 694.6               0
N1422CN1499C         132.34 694.6               0
N159A N1403A         195.261023.1               0
N159B N1403B         195.261023.1               0
N159C N1403C         195.261023.1               0
N160A N1403A          1259.4107.8               0
N160B N1403B          1259.4107.8               0
N160C N1403C          1259.4107.8               0
N1499AN1403A          39.95  245.               0
N1499BN1403B          39.95  245.               0
```

```
N1499CN1403C          39.95  245.               0
N1499AN159A           12.57 110.3               0
N1499BN159B           12.57 110.3               0
N1499CN159C           12.57 110.3               0
N1499AN160A           88.25506.02               0
N1499BN160B           88.25506.02               0
N1499CN160C           88.25506.02               0
N1499AN162A          1255.8 4321.               0
N1499BN162B          1255.8 4321.               0
N1499CN162C          1255.8 4321.               0
N1499AN161A          870.262598.9               0
N1499BN161B          870.262598.9               0
N1499CN161C          870.262598.9               0
N1501AN1499A         591.961601.7               0
N1501BN1499B         591.961601.7               0
N1501CN1499C         591.961601.7               0
N160A N988A           67.11274.57               0
N160B N988B           67.11274.57               0
N160C N988C           67.11274.57               0
N161A N988A           78.04305.68               0
N161B N988B           78.04305.68               0
N161C N988C           78.04305.68               0
N162A N988A          129.16 562.6               0
N162B N988B          129.16 562.6               0
N162C N988C          129.16 562.6               0
N988A                             234.37        0
N988B                             234.37        0
N988C                             234.37        0
N160A N159A           30.14191.53               0
N160B N159B           30.14191.53               0
N160C N159C           30.14191.53               0
N162A N160A              1.  71.92              0
N162B N160B              1.  71.92              0
N162C N160C              1.  71.92              0
N161A N160A            2.32  43.9               0
N161B N160B            2.32  43.9               0
N161C N160C            2.32  43.9               0
N160A N1501A          81.45  305.               0
N160B N1501B          81.45  305.               0
N160C N1501C          81.45  305.               0
N161A N162A             .03 60.89               0
N161B N162B             .03 60.89               0
N161C N162C             .03 60.89               0
N159A N161A          391.361167.2               0
N159B N161B          391.361167.2               0
N159C N161C          391.361167.2               0
N161A N1501A         123.93556.27               0
N161B N1501B         123.93556.27               0
N161C N1501C         123.93556.27               0
N161A                 92.17468.35               0
N161B                 92.17468.35               0
N161C                 92.17468.35               0
N162A N159A          565.661933.4               0
N162B N159B          565.661933.4               0
N162C N159C          565.661933.4               0
N162A N1501A         442.571583.1               0
N162B N1501B         442.571583.1               0
N162C N1501C         442.571583.1               0
N159A N1501A         187.97534.71               0
N159B N1501B         187.97534.71               0
N159C N1501C         187.97534.71               0
N147A N137A          1999.54829.8               0
N147B N137B          1999.54829.8               0
N147C N137C          1999.54829.8               0
N1507A                     2222.2               0
N1507B                     2222.2               0
N1507C                     2222.2               0
      N1507A                        200.        0
      N1507B                        200.        0
      N1507C                        200.        0
N158A                      2222.2               0
N158B                      2222.2               0
N158C                      2222.2               0
```

```
N5158AN158A              .275 24.08                     0
N5158BN158B              .275 24.08                     0
N5158CN158C              .275 24.08                     0
N524A X0376A             .38621.715                     0
N524B X0376B             .38621.715                     0
N524C X0376C             .38621.715                     0
N524A X0378A              .3821.882                     0
N524B X0378B              .3821.882                     0
N524C X0378C              .3821.882                     0
N524A X0380A            1.733871.654                    0
N524B X0380B            1.733871.654                    0
N524C X0380C            1.733871.654                    0
N524A X0382A            1.693472.236                    0
N524B X0382B            1.693472.236                    0
N524C X0382C            1.693472.236                    0
X0376AX0384A             .245432.287                    0
X0376BX0384B             .245432.287                    0
X0376CX0384C             .245432.287                    0
X0378AX0386A             .245432.287                    0
X0378BX0386B             .245432.287                    0
X0378CX0386C             .245432.287                    0
X1071A                  14615.10167.                    0
X1071B                  14615.10167.                    0
X1071C                  14615.10167.                    0
X0380AX1071A            1.733871.654                    0
X0380BX1071B            1.733871.654                    0
X0380CX1071C            1.733871.654                    0
X0382AX1071A            1.693472.236                    0
X0382BX1071B            1.693472.236                    0
X0382CX1071C            1.693472.236                    0
C 1201-1,2 - (1201L), CROWSNEST (4 bundle)
$INCLUDE, E:\ATP\LCC\1201_12.lib, N158A#, N158B#, N158C#, N90KA#, N90KB# $$
  , N90KC#
C 1201-3 - (1201L), 666.9 (4 bundle)
$INCLUDE, E:\ATP\LCC\1201_3.lib, N90KA#, N90KB#, N90KC#, N1507A, N1507B $$
  , N1507C
C Segment 1 : 936-1 - (936L), 937-1 - (937L), 1590 SDC TYPE 9
$INCLUDE, E:\ATP\LCC\936937_1.lib, N160A#, N160B#, N160C#, N160A#, N160B# $$
  , N160C#, X0014A, X0014B, X0014C, X0015A, X0015B, X0015C
C Segment 2 : 936-2 - (936L), 937-2 - (937L), CURLEW
$INCLUDE, E:\ATP\LCC\923924_2.lib, X0014A, X0014B, X0014C, X0015A, X0015B $$
  , X0015C, N159A#, N159B#, N159C#, N159A#, N159B#, N159C#
C 9L20, 397.5 (similar to IBIS)
$INCLUDE, E:\ATP\LCC\9L20.lib, N1422A, N1422B, N1422C, N1499A, N1499B $$
  , N1499C
C 9L27 - (9L27), IBIS
$INCLUDE, E:\ATP\LCC\9L27.lib, N1499A, N1499B, N1499C, N1403A, N1403B $$
  , N1403C
C 918-6,7a,8a - (918L), HAWK and 477 SDC 5005
$INCLUDE, E:\ATP\LCC\918_678.lib, X0401A, X0401B, X0401C, N187A#, N187B# $$
  , N187C#
C Segment 1 : 906-134 - (906L), 477 SDC EC, 928-134 - (928L), SDC 5005
$INCLUDE, E:\ATP\LCC\906928_1.lib, X0026A, X0026B, X0026C, X0027A, X0027B $$
  , X0027C, N155A#, N155B#, N155C#, N155A#, N155B#, N155C#
C Segment 2 : 906-2,5 - (906L), 928-2,5 - (928L), CURLEW, (CARDINAL/SD merged)
$INCLUDE, E:\ATP\LCC\906928_2.lib, N161A#, N161B#, N161C#, N161A#, N161B# $$
  , N161C#, X0026A, X0026B, X0026C, X0027A, X0027B, X0027C
C Segment 1 : 925-12 - (925L), SDC 5005, 929-123 - (929L), SDC 5005
$INCLUDE, E:\ATP\LCC\925929_1.lib, X0034A, X0034B, X0034C, X0035A, X0035B $$
  , X0035C, N152A#, N152B#, N152C#, N152A#, N152B#, N152C#
C Segment 2 : 925-345 - (925L), 929-4,5 - (929L), CURLEW
$INCLUDE, E:\ATP\LCC\925929_2.lib, N160A#, N160B#, N160C#, N160A#, N160B# $$
  , N160C#, X0034A, X0034B, X0034C, X0035A, X0035B, X0035C
C 901-1-5 - (901L), SDC 5005 and HAWK
$INCLUDE, E:\ATP\LCC\901_1_5.lib, N988A#, N988B#, N988C#, N152A#, N152B# $$
  , N152C#
C 901-6 - (901L), HAWK
$INCLUDE, E:\ATP\LCC\901_6.lib, X0044A, X0044B, X0044C, N988A#, N988B# $$
  , N988C#
C 901-7 - (901L), CURLEW
$INCLUDE, E:\ATP\LCC\901_7.lib, N160A#, N160B#, N160C#, X0044A, X0044B $$
  , X0044C
C 900 - (900L), HAWK
$INCLUDE, E:\ATP\LCC\900.lib, N155A#, N155B#, N155C#, N152A#, N152B#, N152C#
C 995 - (995L), HAWK and SDC 5005 (19.97km), N150 (68S) and N995 ignored
$INCLUDE, E:\ATP\LCC\995.lib, N153A#, N153B#, N153C#, N155A#, N155B#, N155C#
C 912-123 - (912L), SDC 5005
$INCLUDE, E:\ATP\LCC\912_123.lib, N152A#, N152B#, N152C#, X0053A, X0053B $$
  , X0053C
C 917-12 - (917L), CURLEW
$INCLUDE, E:\ATP\LCC\917.lib, N162A#, N162B#, N162C#, N160A#, N160B#, N160C#
C 916 - (916L), DRAKE
$INCLUDE, E:\ATP\LCC\916.lib, N161A#, N161B#, N161C#, N162A#, N162B#, N162C#
C 912-4 - (912L), IBIS
$INCLUDE, E:\ATP\LCC\912_4.lib, X0053A, X0053B, X0053C, X0059A, X0059B $$
  , X0059C
C 918-1-5 - (918L), HAWK, 3.08km CURLEW and  SDC 6101 was merged
$INCLUDE, E:\ATP\LCC\918_1_5.lib, X0401A, X0401B, X0401C, N155A#, N155B# $$
  , N155C#
C 912-5 - (912L), 477 SDC 6101
$INCLUDE, E:\ATP\LCC\912_5.lib, X0059A, X0059B, X0059C, X0063A, X0063B $$
  , X0063C
C 912-6 - (912L), IBIS
$INCLUDE, E:\ATP\LCC\912_6.lib, X0063A, X0063B, X0063C, N1422A, N1422B $$
  , N1422C
C Segment 2 : 922-34789 - (922L), 926-34789 - (926L), SDC 5005 and HAWK
$INCLUDE, E:\ATP\LCC\922926_2.lib, N155A#, N155B#, N155C#, N155A#, N155B# $$
  , N155C#, X0070A, X0070B, X0070C, X0071A, X0071B, X0071C
C Segment 1 : 922-1256 - (922L), 926-1256 - (926L), CURLEW
$INCLUDE, E:\ATP\LCC\922926_1.lib, X0070A, X0070B, X0070C, X0071A, X0071B $$
  , X0071C, N135A#, N135B#, N135C#, N135A#, N135B#, N135C#
C 903 - (903L), 190 - (190L), HAWK
$INCLUDE, E:\ATP\LCC\903190.lib, N155A#, N155B#, N155C#, N155A#, N155B# $$
  , N155C#, N420A#, N420B#, N420C#, N420A#, N420B#, N420C#
C Segment 3 : 910-34 - (910L), IBIS, 914-456 - (914L), 447 SDC 6101
$INCLUDE, E:\ATP\LCC\910914_3.lib, N152A#, N152B#, N152C#, N152A#, N152B# $$
  , N152C#, X0681A, X0681B, X0681C, N147A#, N147B#, N147C#
C Segment 2 : 910-2 - (910L), 447 SDC 5005, 914-3 - (914L), 447 SDC 6101
$INCLUDE, E:\ATP\LCC\910914_2.lib, X0681A, X0681B, X0681C, N147A#, N147B# $$
  , N147C#, X0443A, X0443B, X0443C, N81A##, N81B##, N81C##
C Segment 1 : 910-1 - (910L), 447 SDC 5005, 914-12 - (914L), 447 SDC 6101
$INCLUDE, E:\ATP\LCC\910914_1.lib, X0443A, X0443B, X0443C, N81A##, N81B## $$
  , N81C##, N128A#, N128B#, N128C#, N128A#, N128B#, N128C#
C Segment 2 : 908-2-5 - (908L), 909-2-6 - (909L), 477 SDC TYPE 6.4
$INCLUDE, E:\ATP\LCC\908909_2.lib, X0092A, X0092B, X0092C, X0093A, X0093B $$
  , X0093C, N521A#, N521B#, N521C#, X0537A, X0537B, X0537C
C Segment 1 : 908-1 - (908L), 909-1 - (909L), CURLEW
$INCLUDE, E:\ATP\LCC\908909_1.lib, N135A#, N135B#, N135C#, N135A#, N135B# $$
  , N135C#, X0092A, X0092B, X0092C, X0093A, X0093B, X0093C
C Segment 3 : 908-6 - (908L), 909-7 - (909L), 477 SDC TYPE 6.4
$INCLUDE, E:\ATP\LCC\908909_3.lib, N521A#, N521B#, N521C#, X0537A, X0537B $$
  , X0537C, X0102A, X0102B, X0102C, N512A#, N512B#, N512C#
C Segment 4 : 908-7,8 - (908L), 909-8-12 - (909L), 477 SDC TYPE 6.4
$INCLUDE, E:\ATP\LCC\908909_4.lib, X0102A, X0102B, X0102C, N512A#, N512B# $$
  , N512C#, N128A#, N128B#, N128C#, N128A#, N128B#, N128C#
C 1202 - (1202L), 3 bundle 1590 SDC TYPE 7/9, maybe 500kV line
$INCLUDE, E:\ATP\LCC\1202.lib, N420A#, N420B#, N420C#, N128A#, N128B# $$
  , N128C#
C 1203 - (1203L), 3 bundle DRAKE, maybe 500kV line
$INCLUDE, E:\ATP\LCC\1203.lib, N420A#, N420B#, N420C#, N524A#, N524B# $$
  , N524C#
C 1209 - (1209L), 3 bundle 1590 SDC TYPE 7/9, TRILLIUM, maybe 500kV line
$INCLUDE, E:\ATP\LCC\1209.lib, N524A#, N524B#, N524C#, N128A#, N128B# $$
  , N128C#
C 902-34 - (902L), HAWK and SDC 5005
$INCLUDE, E:\ATP\LCC\902_34.lib, X0114A, X0114B, X0114C, N133A#, N133B# $$
  , N133C#
C 902-1256 - (902L), CURLEW
$INCLUDE, E:\ATP\LCC\902_1256.lib, N135A#, N135B#, N135C#, X0114A, X0114B $$
  , X0114C
C 904-134 - (904L), CURLEW
$INCLUDE, E:\ATP\LCC\904_134.lib, X0118A, X0118B, X0118C, X0475A, X0475B $$
  , X0475C, N505A#, N505B#, N505C#, N505A#, N505B#, N505C#
C 904-2 - (904L), DRAKE
$INCLUDE, E:\ATP\LCC\904_2.lib, N133A#, N133B#, N133C#, X0118A, X0118B $$
  , X0118C
```

```
C 904-567 - (904L), DRAKE
$INCLUDE, E:\ATP\LCC\904_567.lib, X0475A, X0475B, X0475C, N521A#, N521B# $$
  , N521C#
C 904-8,9,10 - (904L), DRAKE
$INCLUDE, E:\ATP\LCC\904_8910.lib, N521A#, N521B#, N521C#, X0531A, X0531B $$
  , X0531C
C 904-11~14 - (904L), CURLEW
$INCLUDE, E:\ATP\LCC\904_11_4.lib, X0531A, X0531B, X0531C, X0129A, X0129B $$
  , X0129C, N509A#, N509B#, N509C#, N518A#, N518B#, N518C#
C 904-15~17 - (904L), 447 SDC
$INCLUDE, E:\ATP\LCC\904_15_7.lib, X0129A, X0129B, X0129C, N128A#, N128B# $$
  , N128C#
C 904-18~21 - (904L), 447 SDC
$INCLUDE, E:\ATP\LCC\904_18_2.lib, N128A#, N128B#, N128C#, N513A#, N513B# $$
  , N513C#
C 904-22~24 - (904L), 447 SDC
$INCLUDE, E:\ATP\LCC\904_22_4.lib, N513A#, N513B#, N513C#, N891A#, N891B# $$
  , N891C#
C 904-25~26 - (904L), 447 SDC
$INCLUDE, E:\ATP\LCC\904_25_6.lib, N891A#, N891B#, N891C#, N136A#, N136B# $$
  , N136C#
C 908-9~12 - (908L), 447 SDC 6.4
$INCLUDE, E:\ATP\LCC\908_9_12.lib, N128A#, N128B#, N128C#, N136A#, N136B# $$
  , N136C#
C 905 - (905L), DRAKE
$INCLUDE, E:\ATP\LCC\905.lib, N133A#, N133B#, N133C#, N137A#, N137B#, N137C#
C 920-12 - (920L), DRAKE
$INCLUDE, E:\ATP\LCC\920_12.lib, N137A#, N137B#, N137C#, N519A#, N519B# $$
  , N519C#
C 240BA2 & 240BA3, 2500 MCM U/G CABLE, 1590 SDC assumed
$INCLUDE, E:\ATP\LCC\240BA2_3.lib, N509A#, N509B#, N509C#, N518A#, N518B# $$
  , N518C#, N506A#, N506B#, N506C#, N506A#, N506B#, N506C#
C 920-3 - (920L), DRAKE
$INCLUDE, E:\ATP\LCC\920_3.lib, N519A#, N519B#, N519C#, X0151A, X0151B $$
  , X0151C
C 920-4 - (920L), HORNBILL, 921-4 - (921L), HORNBILL
$INCLUDE, E:\ATP\LCC\920921.lib, X0151A, X0151B, X0151C, X0153A, X0153B $$
  , X0153C, N57A##, N57B##, N57C##, N57A##, N57B##, N57C##
C 921-123 - (921L), CURLEW
$INCLUDE, E:\ATP\LCC\921_123.lib, N514A#, N514B#, N514C#, X0153A, X0153B $$
  , X0153C
C 915L-1,2 (915L), CURLEW
$INCLUDE, E:\ATP\LCC\915.lib, N136A#, N136B#, N136C#, N514A#, N514B#, N514C#
C 930-12 - (930L), CURLEW
$INCLUDE, E:\ATP\LCC\930.lib, N505A#, N505B#, N505C#, N137A#, N137B#, N137C#
C 932-1,2 - (932L), 447 SDC 5005
$INCLUDE, E:\ATP\LCC\932_12.lib, N187A#, N187B#, N187C#, X0163A, X0163B $$
  , X0163C
C 932-3 - (932L), CURLEW
$INCLUDE, E:\ATP\LCC\932_3.lib, X0163A, X0163B, X0163C, N160A#, N160B# $$
  , N160C#
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1    ><    T1    >< TSTART >< TSTOP  >
14N159A        1.E-20       60.                                    -1.        10.
18              .478N5158A
14N159B        1.E-20       60.                                    -1.        10.
18              .478N5158B
14N159C        1.E-20       60.                                    -1.        10.
18              .478N5158C
14N1520A 0     416.41       60.                                    -1.         1.
14N1520B 0     416.41       60.      -120.                         -1.         1.
14N1520C 0     416.41       60.       120.                         -1.         1.
14N137A -1       .252       60.     -82.72                         -1.         1.
14N137B -1       .252       60.    -202.72                         -1.         1.
14N137C -1       .252       60.      37.28                         -1.         1.
14N137A -1      .6163       60.     150.89                         -1.         1.
14N137B -1      .6163       60.      30.89                         -1.         1.
14N137C -1      .6163       60.     270.89                         -1.         1.
14N57A  -1     1.9733       60.     -80.77                         -1.         1.
14N57B  -1     1.9733       60.    -200.77                         -1.         1.
14N57C  -1     1.9733       60.      39.23                         -1.         1.
14N57A  -1      1.777       60.      114.5                         -1.         1.
14N57B  -1      1.777       60.       -5.5                         -1.         1.
14N57C  -1      1.777       60.      234.5                         -1.         1.
14N519A -1       .153       60.     171.19                         -1.         1.
14N519B -1       .153       60.      51.19                         -1.         1.
14N519C -1       .153       60.     291.19                         -1.         1.
14N514A -1       .133       60.     -75.01                         -1.         1.
14N514B -1       .133       60.    -195.01                         -1.         1.
14N514C -1       .133       60.      44.99                         -1.         1.
14N514A -1      1.209       60.      16.68                         -1.         1.
14N514B -1      1.209       60.    -103.32                         -1.         1.
14N514C -1      1.209       60.     136.68                         -1.         1.
14N136A -1       .347       60.     -75.62                         -1.         1.
14N136B -1       .347       60.    -195.62                         -1.         1.
14N136C -1       .347       60.      44.38                         -1.         1.
14N136A -1      .9516       60.     154.14                         -1.         1.
14N136B -1      .9516       60.      34.14                         -1.         1.
14N136C -1      .9516       60.     274.14                         -1.         1.
14N891A -1      .0755       60.     171.52                         -1.         1.
14N891B -1      .0755       60.      51.52                         -1.         1.
14N891C -1      .0755       60.     291.52                         -1.         1.
14N506A -1      .7193       60.     160.46                         -1.         1.
14N506B -1      .7193       60.      40.46                         -1.         1.
14N506C -1      .7193       60.     280.46                         -1.         1.
14N505A -1      .5534       60.     166.22                         -1.         1.
14N505B -1      .5534       60.      46.22                         -1.         1.
14N505C -1      .5534       60.     286.22                         -1.         1.
14N133A -1      .7857       60.      24.76                         -1.         1.
14N133B -1      .7857       60.     -95.24                         -1.         1.
14N133C -1      .7857       60.     144.76                         -1.         1.
14N521A -1      .1503       60.     173.85                         -1.         1.
14N521B -1      .1503       60.      53.85                         -1.         1.
14N521C -1      .1503       60.     293.85                         -1.         1.
14N513A -1      .2826       60.     166.27                         -1.         1.
14N513B -1      .2826       60.      46.27                         -1.         1.
14N513C -1      .2826       60.     286.27                         -1.         1.
14N128A -1       .046       60.     -75.45                         -1.         1.
14N128B -1       .046       60.    -195.45                         -1.         1.
14N128C -1       .046       60.      44.55                         -1.         1.
14N512A -1      .1423       60.     169.24                         -1.         1.
14N512B -1      .1423       60.      49.24                         -1.         1.
14N512C -1      .1423       60.     289.24                         -1.         1.
14N135A -1        4.3       60.     -80.67                         -1.         1.
14N135B -1        4.3       60.    -200.67                         -1.         1.
14N135C -1        4.3       60.      39.33                         -1.         1.
14N135A -1      5.538       60.       59.1                         -1.         1.
14N135B -1      5.538       60.      -60.9                         -1.         1.
14N135C -1      5.538       60.      179.1                         -1.         1.
14N187A -1     .01737       60.     157.43                         -1.         1.
14N187B -1     .01737       60.      37.43                         -1.         1.
14N187C -1     .01737       60.     277.43                         -1.         1.
14N420A -1      2.093       60.      21.41                         -1.         1.
14N420B -1      2.093       60.     -98.59                         -1.         1.
14N420C -1      2.093       60.     141.41                         -1.         1.
14N420A -1      .1066       60.     -71.28                         -1.         1.
14N420B -1      .1066       60.    -191.28                         -1.         1.
14N420C -1      .1066       60.      48.72                         -1.         1.
14N81A  -1      .3994       60.     152.88                         -1.         1.
14N81B  -1      .3994       60.      32.88                         -1.         1.
14N81C  -1      .3994       60.     272.88                         -1.         1.
14N81A  -1       .107       60.     -86.15                         -1.         1.
14N81B  -1       .107       60.    -206.15                         -1.         1.
14N81C  -1       .107       60.      33.85                         -1.         1.
14N153A -1      .2909       60.    -162.31                         -1.         1.
14N153B -1      .2909       60.    -282.31                         -1.         1.
14N153C -1      .2909       60.     -42.31                         -1.         1.
14N153A -1      .1803       60.     -84.83                         -1.         1.
14N153B -1      .1803       60.    -204.83                         -1.         1.
14N153C -1      .1803       60.      35.17                         -1.         1.
14N155A -1      .3235       60.     158.68                         -1.         1.
14N155B -1      .3235       60.      38.68                         -1.         1.
14N155C -1      .3235       60.     278.68                         -1.         1.
14N155A -1       .253       60.     -84.47                         -1.         1.
14N155B -1       .253       60.    -204.47                         -1.         1.
14N155C -1       .253       60.      35.53                         -1.         1.
```

```
14N147A -1      .322        60.      162.84                                        -1.        1.
14N147B -1      .322        60.       42.84                                        -1.        1.
14N147C -1      .322        60.      282.84                                        -1.        1.
14N152A -1     .4045        60.      158.28                                        -1.        1.
14N152B -1     .4045        60.       38.28                                        -1.        1.
14N152C -1     .4045        60.      278.28                                        -1.        1.
14N152A -1      .198        60.       -87.9                                        -1.        1.
14N152B -1      .198        60.      -207.9                                        -1.        1.
14N152C -1      .198        60.        32.1                                        -1.        1.
14N1501A      1.E-20        60.                                                    -1.       10.
18              .468X0895A
14N1501B      1.E-20        60.                                                    -1.       10.
18              .468X0895B
14N1501C      1.E-20        60.                                                    -1.       10.
18              .468X0895C
14N1422A-1     .1848        60.      144.06                                        -1.        1.
14N1422B-1     .1848        60.       24.06                                        -1.        1.
14N1422C-1     .1848        60.      264.06                                        -1.        1.
14N1422A-1      .068        60.      -93.92                                        -1.        1.
14N1422B-1      .068        60.     -213.92                                        -1.        1.
14N1422C-1      .068        60.       26.08                                        -1.        1.
14N1403A-1     .6883        60.      134.84                                        -1.        1.
14N1403B-1     .6883        60.       14.84                                        -1.        1.
14N1403C-1     .6883        60.      254.84                                        -1.        1.
14N1403A-1      .462        60.      -97.78                                        -1.        1.
14N1403B-1      .462        60.     -217.78                                        -1.        1.
14N1403C-1      .462        60.       22.22                                        -1.        1.
14N1499A-1     .9033        60.       72.09                                        -1.        1.
14N1499B-1     .9033        60.      -47.91                                        -1.        1.
14N1499C-1     .9033        60.      192.09                                        -1.        1.
14N1499A-1      1.14        60.       -95.1                                        -1.        1.
14N1499B-1      1.14        60.      -215.1                                        -1.        1.
14N1499C-1      1.14        60.        24.9                                        -1.        1.
14N988A -1     .2742        60.       153.5                                        -1.        1.
14N988B -1     .2742        60.        33.5                                        -1.        1.
14N988C -1     .2742        60.       273.5                                        -1.        1.
14N160A -1    1.6443        60.      142.02                                        -1.        1.
14N160B -1    1.6443        60.       22.02                                        -1.        1.
14N160C -1    1.6443        60.      262.02                                        -1.        1.
14N160A -1     1.222        60.      -95.32                                        -1.        1.
14N160B -1     1.222        60.     -215.32                                        -1.        1.
14N160C -1     1.222        60.       24.68                                        -1.        1.
14N161A -1    1.6625        60.       147.1                                        -1.        1.
14N161B -1    1.6625        60.        27.1                                        -1.        1.
14N161C -1    1.6625        60.       267.1                                        -1.        1.
14N162A -1     .8402        60.      145.08                                        -1.        1.
14N162B -1     .8402        60.       25.08                                        -1.        1.
14N162C -1     .8402        60.      265.08                                        -1.        1.
14N162A -1      .733        60.      -96.92                                        -1.        1.
14N162B -1      .733        60.     -216.92                                        -1.        1.
14N162C -1      .733        60.       23.08                                        -1.        1.
14N159A -1     .1061        60.       83.93                                        -1.        1.
14N159B -1     .1061        60.      -36.07                                        -1.        1.
14N159C -1     .1061        60.      203.93                                        -1.        1.
14N159A -1     1.016        60.      -98.66                                        -1.        1.
14N159B -1     1.016        60.     -218.66                                        -1.        1.
14N159C -1     1.016        60.       21.34                                        -1.        1.
14N159A -1     .8152        60.      110.73                                        -1.        1.
14N159B -1     .8152        60.       -9.27                                        -1.        1.
14N159C -1     .8152        60.      230.73                                        -1.        1.
14X0384A 0     207.7        60.        28.5                                        -1.        1.
14X0384B 0     207.7        60.       -91.5                                        -1.        1.
14X0384C 0     207.7        60.       148.5                                        -1.        1.
14X0386A 0     207.7        60.        29.9                                        -1.        1.
14X0386B 0     207.7        60.       -90.1                                        -1.        1.
14X0386C 0     207.7        60.       149.9                                        -1.        1.
/OUTPUT
  N524A
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

C.3.2 Robust TLNE Model

```
BEGIN NEW DATA CASE
C -------------------------------------------------------
C Xin Nie
C Power Engineering Group
C Dept. of Electrical and Computer Engineering
C University of Alberta
C August 25, 2005
C -------------------------------------------------------
C Alberta Interconnected Electric System 240kV Area 50 Backbone
C
C
$DUMMY, XYZ000
C  dT  >< Tmax >< Xopt >< Copt >
   2.E-5     .15
       1       1       0       0       0       0       0       1       0
C        1         2         3         4         5         6         7         8
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
/BRANCH
C < n 1>< n 2><ref1><ref2>< R  >< L  >< C  >
C < n 1>< n 2><ref1><ref2>< R  >< A  >< B  ><Leng><><>0
C *************
C study zone
C *************
C ********************************** Alpha Mode ********************************
  NN524AX0010A                .386  57.6                                       0
  NN524AX0012A                 .38 58.04                                       0
  NN524AX0014A              1.7338190.07                                       0
  NN524AX0016A              1.6934 191.6                                       0
  X0010AX0018A               .2454 85.64                                       0
  X0012AX0020A               .2454 85.64                                       0
  X0047A                    14615.26969.                                       0
  X0014AX0047A              1.7338190.07                                       0
  X0016AX0047A              1.6934 191.6                                       0
C Fault resistance is 10ohm
  X0030A                      10.                                              0
C ********************************** Beta Mode *********************************
  NN524BX0010B                .386  57.6                                       0
  NN524BX0012B                 .38 58.04                                       0
  NN524BX0014B              1.7338190.07                                       0
  NN524BX0016B              1.6934 191.6                                       0
  X0010BX0018B               .2454 85.64                                       0
  X0012BX0020B               .2454 85.64                                       0
  X0047B                    14615.26969.                                       0
  X0014BX0047B              1.7338190.07                                       0
  X0016BX0047B              1.6934 191.6                                       0
C Fault resistance is 10ohm
  X0030B                      10.                                              0
C *************
C surface layer
C *************
C ********************************** Alpha Mode ********************************
C 1202 - (1202L), 3 bundle 1590 SDC TYPE 7/9
-1A____2A____1                 2.  0.00              -2 1
       6        2.28629697636174260OE+002
   1.0930875266152098E+004   3.7816977451926283E+002   7.2244742454667050E+002
```

```
  1.00000000
  0.00000000
C 1203 - (1203L), 3 bundle DRAKE
-1A____2NN524A                 2.  0.00               -2 1
     6        2.35229445783656100OE+002
   3.9505913954208889E+004   8.0197412225411199E+002   1.4725142377331936E+003
   8.2201676052107996E+002   4.4413522729278338E+002   2.7794179375099776E+002
   2.6680936098286791E+004   4.4921141400484473E+001   7.7832278591362201E+000
   1.8693797599852731E+000   4.2769483703231187E-001   1.1187292926531707E-001
     5        6.1042236888132892OOE-005
   2.4151609673450780E+006  -2.8230692343018622E+004   7.4449451575206308E+002
   7.2745462670418738E+002   6.1464436080477562E+001
   2.4411496466817860E+006   4.5797148524818365E+006   1.3891888773426731E+005
   1.2317685627274060E+005   1.1287966569326491E+004
  1.00000000
  0.00000000
C 1209 - (1209L), 3 bundle 1590 SDC TYPE 7/9, TRILLIUM
-1A____1NN524A                 2.  0.00               -2 1
     6        2.2862969763617426OOE+002
   1.0930875266152098E+004   3.7816977451926283E+002   7.2244742454667050E+002
   4.1950782676470499E+002   2.2555335480015390E+002   1.5690211858979180E+002
   6.1738261875301032E+003   2.4272139132891308E+001   4.2730997978570935E+000
   1.1090996364568724E+000   2.8428629943126288E-001   1.0228943543026825E-001
     7        2.2455520048261961OOE-004
   9.9887410724068910E+005   1.1008647372556110E+005   1.3520092765397260E+003
   2.8928746877012838E+002   2.3800045703456291E+002   7.0667870778097210E+001
   4.3505966674159673E+000
   1.1944215242197730E+006   8.7173695782165113E+005   6.9547573155737497E+004
   7.6282894016341219E+004   4.9061961187422588E+004   1.5429848538226939E+004
   9.6482408646867452E+002
  1.00000000
  0.00000000
C ********************************* Beta Mode *******************************
C 1202 - (1202L), 3 bundle 1590 SDC TYPE 7/9
-1B____2B____1                 2.  0.00               -2 1
     6        2.2862969763617426OOE+002
   1.0930875266152098E+004   3.7816977451926283E+002   7.2244742454667050E+002
   4.1950782676470499E+002   2.2555335480015390E+002   1.5690211858979180E+002
   6.1738261875301032E+003   2.4272139132891308E+001   4.2730997978570935E+000
   1.1090996364568724E+000   2.8428629943126288E-001   1.0228943543026825E-001
     7        2.3079513625135739OOE-004
   1.1105422223173680E+006   3.8383326840361638E+004   1.3390360725888700E+003
   3.2319994632951602E+002   2.5118915993528501E+002   7.7028931875867855E+001
   4.4701114841863907E+000
   1.2399131976713850E+006   5.8232972072919679E+005   6.8208797858970356E+004
   7.7755873710647822E+004   5.0009371993671411E+004   1.6343106099887820E+004
   9.6476396587423778E+002
  1.00000000
  0.00000000
C 1203 - (1203L), 3 bundle DRAKE
-1B____2NN524B                 2.  0.00               -2 1
     6        2.35229445783656100OE+002
   3.9505913954208889E+004   8.0197412225411199E+002   1.4725142377331936E+003
   8.2201676052107996E+002   4.4413522729278338E+002   2.7794179375099776E+002
   2.6680936098286791E+004   4.4921141400484473E+001   7.7832278591362201E+000
   1.8693797599852731E+000   4.2769483703231187E-001   1.1187292926531707E-001
     5        6.1042236888132892OOE-005
   2.4151609673450780E+006  -2.8230692343018622E+004   7.4449451575206308E+002
   7.2745462670418738E+002   6.1464436080477562E+001
   2.4411496466817860E+006   4.5797148524818365E+006   1.3891888773426731E+005
   1.2317685627274060E+005   1.1287966569326491E+004
  1.00000000
  0.00000000
C 1209 - (1209L), 3 bundle 1590 SDC TYPE 7/9, TRILLIUM
-1B____1NN524B                 2.  0.00               -2 1
     6        2.2862969763617426OOE+002
   1.0930875266152098E+004   3.7816977451926283E+002   7.2244742454667050E+002
   4.1950782676470499E+002   2.2555335480015390E+002   1.5690211858979180E+002
   6.1738261875301032E+003   2.4272139132891308E+001   4.2730997978570935E+000
   1.1090996364568724E+000   2.8428629943126288E-001   1.0228943543026825E-001
     7        2.2455520048261961OOE-004
   9.9887410724068910E+005   1.1008647372556110E+005   1.3520092765397260E+003
   2.8928746877012838E+002   2.3800045703456291E+002   7.0667870778097210E+001
```

```
   4.3505966674159673E+000
   1.1944215242197730E+006   8.7173695782165113E+005   6.9547573155737497E+004
   7.6282894016341219E+004   4.9061961187422588E+004   1.5429848538226939E+004
   9.6482408646867452E+002
  1.00000000
  0.00000000
C *************
C deep region
C *************
C BEGIN FDNE
C ********************************** Alpha Mode ******************************
$VINTAGE,1
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><   microF     >
C
C (1,1)
  A_____1                   4.794415e+001
  A_____1                   9.493358e+000   1.713512e+002
  A_____1                   2.749832e+003   2.027777e+004
  A_____1                  -4.959225e+001  -3.649759e+000
  A_____1A 4__1            -7.177667e+001  -7.209291e+002
  A 4__1                    1.144283e+002
  A 4__1                                                  -3.579021e+002
  A_____1A 6__1             4.891120e+000   1.096120e+002
  A 6__1                   -1.460729e+005
  A 6__1                                                   7.690014e+000
  A_____1A 8__1             5.273031e+000   5.655519e+001
  A 8__1                   -6.603677e+003
  A 8__1                                                   4.595994e+000
  A_____1A A__1            -8.781766e-001   1.404059e+002
  A A__1                    2.389376e+004
  A A__1                                                   6.859620e-001
  A_____1A C__1             1.605895e+002   3.123332e+002
  A C__1                   -9.180542e+003
  A C__1                                                   2.509075e-001
  A_____1A E__1             5.796004e+000   5.743626e+001
  A E__1                   -4.518826e+004
  A E__1                                                   1.123840e+000
  A_____1A10__1             5.347305e+000   7.876362e+001
  A10__1                    7.120424e+004
  A10__1                                                   6.714215e-001
  A_____1A12__1            -4.031501e+002   1.320121e+003
  A12__1                    8.496503e+004
  A12__1                                                   2.756715e-002
  A_____1A14__1             9.837555e+000   4.307456e+001
  A14__1                   -1.224322e+004
  A14__1                                                   6.348458e-001
  A_____1A16__1             2.723430e+001   2.055907e+001
  A16__1                   -2.016991e+003
  A16__1                                                   4.175572e-001
  A_____1A18__1             9.151733e+000   3.943101e+001
  A18__1                   -7.115812e+004
  A18__1                                                   1.373400e-001
C (1,2)
  A_____1A_____2           -2.412135e+004
  A_____1A_____2            1.264011e+001   2.281487e+002
  A_____1A_____2            1.319321e+002   9.728918e+002
  A_____1A_____2           -8.499067e+003  -6.254918e+002
  A_____1A 4_12             2.894614e+001   5.290562e+002
  A 4_12A_____2             8.461279e+000
  A 4_12A_____2                                            5.784619e+003
  A_____1A 6_12            -1.400855e+001  -3.397235e+002
  A 6_12A_____2            -1.614046e+005
  A 6_12A_____2                                           -2.481487e+000
  A_____1A 8_12            -1.649082e+001  -1.749592e+002
  A 8_12A_____2             1.981734e+004
  A 8_12A_____2                                           -1.485595e+000
  A_____1A A_12             4.669506e+000  -4.794408e+002
  A A_12A_____2            -7.718285e+004
  A A_12A_____2                                           -2.008815e-001
  A_____1A C_12            -3.786232e+002  -9.427005e+002
  A C_12A_____2             3.712853e+004
  A C_12A_____2                                           -8.374724e-002
  A_____1A E_12             1.583677e+002   1.118122e+003
```

```
A E_12A____2           -2.868322e+005
A E_12A____2                                           5.770550e-002
A____1A10_12            2.495367e+001   2.351903e+002
A10_12A____2           -2.572280e+005
A10_12A____2                                           2.248158e-001
A____1A12_12            5.012797e+002  -2.021988e+003
A12_12A____2           -1.501703e+005
A12_12A____2                                          -1.802356e-002
A____1A14_12           -2.394826e+001  -1.030946e+002
A14_12A____2            2.843980e+004
A14_12A____2                                          -2.652384e-001
A____1A16_12           -3.029497e+003   4.894610e+002
A16_12A____2            1.191969e+004
A16_12A____2                                           1.326025e-002
A____1A18_12           -2.675602e+002   3.302770e+002
A18_12A____2            6.514827e+004
A18_12A____2                                           1.633147e-002
C (2,2)
 A____2                 1.561783e+002
 A____2                 3.399620e+001   6.136174e+002
 A____2                -1.611083e+003  -1.188042e+004
 A____2                -1.649564e+002  -1.214002e+001
 A____2A 4__2          -1.284818e+001   1.715270e+003
 A 4__2                 4.283411e+001
 A 4__2                                                2.825206e+002
 A____2A 6__2           9.128618e+000   2.302783e+002
 A 6__2                 6.678028e+004
 A 6__2                                                3.661054e+000
 A____2A 8__2           1.133786e+001   1.184714e+002
 A 8__2                -1.287173e+004
 A 8__2                                                2.193829e+000
 A____2A A__2          -1.158493e+000   3.389292e+002
 A A__2                 6.048886e+004
 A A__2                                                2.841737e-001
 A____2A C__2           2.355020e+002   6.303375e+002
 A C__2                -2.715400e+004
 A C__2                                                1.254410e-001
 A____2A E__2          -1.029387e+002  -1.176880e+003
 A E__2                 2.918482e+006
 A E__2                                               -5.485280e-002
 A____2A10__2          -2.393039e+001  -3.143175e+002
 A10__2                -4.690046e+005
 A10__2                                               -1.682449e-001
 A____2A12__2           1.814293e+002   6.146666e+002
 A12__2                -9.700521e+004
 A12__2                                                5.937703e-002
 A____2A14__2           1.484779e+001   6.005530e+001
 A14__2                -1.489116e+004
 A14__2                                                4.552538e-001
 A____2A16__2           4.956775e+003  -5.222964e+002
 A16__2                -1.119085e+004
 A16__2                                               -9.281440e-003
 A____2A18__2           2.830843e+002  -3.804930e+002
 A18__2                -8.068415e+004
 A18__2                                               -1.418462e-002
$VINTAGE,0
C ********************************* Beta Mode ******************************
$VINTAGE,1
C <BUS1><BUS2><BUS3><BUS4><   OHM        ><   milliH     ><   microF     >
C
C (1,1)
 B____1                 4.794415e+001
 B____1                 9.493358e+000   1.713512e+002
 B____1                 2.749832e+003   2.027777e+004
 B____1                -4.959225e+001  -3.649759e+000
 B____1B 4__1          -7.177667e+001  -7.209291e+002
 B 4__1                 1.144283e+002
 B 4__1                                               -3.579021e+002
 B____1B 6__1           4.891120e+000   1.096120e+002
 B 6__1                -1.460729e+005
 B 6__1                                                7.690014e+000
 B____1B 8__1           5.273031e+000   5.655519e+001
 B 8__1                -6.603677e+003
 B 8__1                                                4.595994e+000
 B____1B A__1          -8.781766e-001   1.404059e+002
 B A__1                 2.389376e+004
 B A__1                                                6.859620e-001
 B____1B C__1           1.605895e+002   3.123332e+002
 B C__1                -9.180542e+003
 B C__1                                                2.509075e-001
 B____1B E__1           5.796004e+000   5.743626e+001
 B E__1                -4.518826e+004
 B E__1                                                1.123840e+000
 B____1B10__1           5.347305e+000   7.876362e+001
 B10__1                 7.120424e+004
 B10__1                                                6.714215e-001
 B____1B12__1          -4.031501e+002   1.320121e+003
 B12__1                 8.496503e+004
 B12__1                                                2.756715e-002
 B____1B14__1           9.837555e+000   4.307456e+001
 B14__1                -1.224322e+004
 B14__1                                                6.348458e-001
 B____1B16__1           2.723430e+001   2.055907e+001
 B16__1                -2.016991e+003
 B16__1                                                4.175572e-001
 B____1B18__1           9.151733e+000   3.943101e+001
 B18__1                -7.115812e+004
 B18__1                                                1.373400e-001
C (1,2)
 B____1B____2          -2.412135e+004
 B____1B____2           1.264011e+001   2.281487e+002
 B____1B____2           1.319321e+002   9.728918e+002
 B____1B____2          -8.499067e+003  -6.254918e+002
 B____1B 4_12           2.894614e+001   5.290562e+002
 B 4_12B____2           8.461279e+000
 B 4_12B____2                                          5.784619e+003
 B____1B 6_12          -1.400855e+001  -3.397235e+002
 B 6_12B____2          -1.614046e+005
 B 6_12B____2                                         -2.481487e+000
 B____1B 8_12          -1.649082e+001  -1.749592e+002
 B 8_12B____2           1.981734e+004
 B 8_12B____2                                         -1.485595e+000
 B____1B A_12           4.669506e+000  -4.794408e+002
 B A_12B____2          -7.718285e+004
 B A_12B____2                                         -2.008815e-001
 B____1B C_12          -3.786232e+002  -9.427005e+002
 B C_12B____2           3.712853e+004
 B C_12B____2                                         -8.374724e-002
 B____1B E_12           1.583677e+002   1.118122e+003
 B E_12B____2          -2.868322e+005
 B E_12B____2                                          5.770550e-002
 B____1B10_12           2.495367e+001   2.351903e+002
 B10_12B____2          -2.572280e+005
 B10_12B____2                                          2.248158e-001
 B____1B12_12           5.012797e+002  -2.021988e+003
 B12_12B____2          -1.501703e+005
 B12_12B____2                                         -1.802356e-002
 B____1B14_12          -2.394826e+001  -1.030946e+002
 B14_12B____2           2.843980e+004
 B14_12B____2                                         -2.652384e-001
 B____1B16_12          -3.029497e+003   4.894610e+002
 B16_12B____2           1.191969e+004
 B16_12B____2                                          1.326025e-002
 B____1B18_12          -2.675602e+002   3.302770e+002
 B18_12B____2           6.514827e+004
 B18_12B____2                                          1.633147e-002
C (2,2)
 B____2                 1.561783e+002
 B____2                 3.399620e+001   6.136174e+002
 B____2                -1.611083e+003  -1.188042e+004
 B____2                -1.649564e+002  -1.214002e+001
 B____2B 4__2          -1.284818e+001   1.715270e+003
 B 4__2                 4.283411e+001
 B 4__2                                                2.825206e+002
 B____2B 6__2           9.128618e+000   2.302783e+002
 B 6__2                 6.678028e+004
```

```
  B 6__2                                                       3.661054e+000
  B_____2B 8__2              1.133786e+001   1.184714e+002
  B 8__2                    -1.287173e+004
  B 8__2                                                       2.193829e+000
  B_____2B A__2             -1.158493e+000   3.389292e+002
  B A__2                     6.048886e+004
  B A__2                                                       2.841737e-001
  B_____2B C__2              2.355020e+002   6.303375e+002
  B C__2                    -2.715400e+004
  B C__2                                                       1.254410e-001
  B_____2B E__2             -1.029387e+002  -1.176880e+003
  B E__2                     2.918482e+006
  B E__2                                                      -5.485280e-002
  B_____2B10__2             -2.393039e+001  -3.143175e+002
  B10__2                    -4.690046e+005
  B10__2                                                      -1.682449e-001
  B_____2B12__2              1.814293e+002   6.146666e+002
  B12__2                    -9.700521e+004
  B12__2                                                       5.937703e-002
  B_____2B14__2              1.484779e+001   6.005530e+001
  B14__2                    -1.489116e+004
  B14__2                                                       4.552538e-001
  B_____2B16__2              4.956775e+003  -5.222964e+002
  B16__2                    -1.119085e+004
  B16__2                                                      -9.281440e-003
  B_____2B18__2              2.830843e+002  -3.804930e+002
  B18__2                    -8.068415e+004
  B18__2                                                      -1.418462e-002
$VINTAGE,0
C END FDNE
/SWITCH
```

```
C < n 1>< n 2>< Tclose ><Top/Tde ><   Ie   ><Vf/CLOP ><  type  >
C ********************************** Alpha Mode *******************************
  NN524AX0030A        .05        1.                                           1
C ********************************** Beta Mode *******************************
  NN524BX0030B        .05        1.                                           1
/SOURCE
C < n 1><>< Ampl.  >< Freq.  ><Phase/T0><   A1   ><   T1   >< TSTART >< TSTOP  >
C ********************************** Alpha Mode *******************************
14X0018A 0    254.38        60.      28.5                         -1.        1.
14X0020A 0    254.38        60.      29.9                         -1.        1.
C ************************************
C Norton equivalent current source
C ************************************
14NN524A-12.30377594        60. -125.6348                         -1.        1.
C ********************************** Beta Mode *******************************
14X0018B 0    254.38        60.     -61.5                         -1.        1.
14X0020B 0    254.38        60.     -60.1                         -1.        1.
C ************************************
C Norton equivalent current source
C ************************************
14NN524B-12.30377594        60. -215.6348                         -1.        1.
/OUTPUT
  NN524ANN524B
BLANK BRANCH
BLANK SWITCH
BLANK SOURCE
BLANK OUTPUT
BLANK PLOT
BEGIN NEW DATA CASE
BLANK
```

D

EMTP Models of Passive Elements

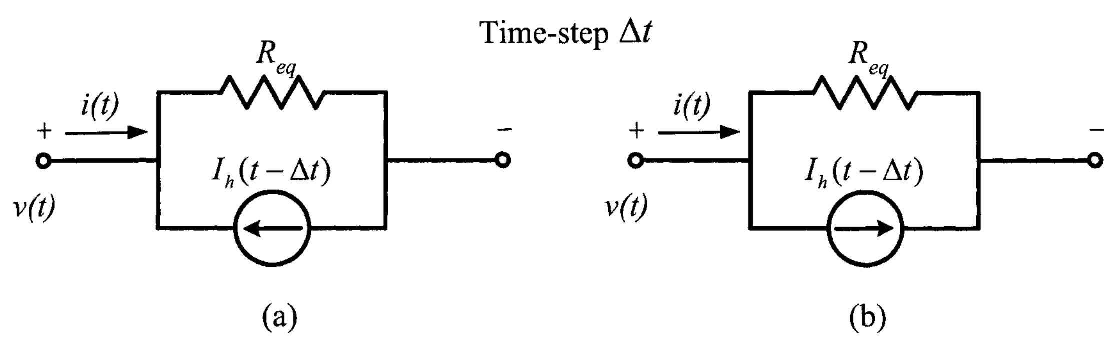


Figure D.1: Norton equivalent circuit of the discretized branch of passive elements

D.1 L Branch

Shown in Fig. D.1(b)

$$i(t) = \frac{1}{R_{eq}} v(t) - I(t - \Delta t)$$

where

$$R_{eq} = \frac{2L}{\Delta t}$$

$$I_h(t - \Delta t) = I_h(t - 2\Delta t) + \frac{2}{R_{eq}} v(t - \Delta t)$$

D.2 C Branch

Shown in Fig. D.1(a)

$$i(t) = \frac{1}{R_{eq}} v(t) + I(t - \Delta t)$$

where

$$R_{eq} = \frac{\Delta t}{2C}$$

$$I_h(t - \Delta t) = -I_h(t - 2\Delta t) + \frac{2}{R_{eq}} v(t - \Delta t)$$

D.3 RL Branch

Shown in Fig. D.1(b)

$$i(t) = \frac{1}{R_{eq}} v(t) - I(t - \Delta t)$$

where

$$R_{eq} = R + \frac{2L}{\Delta t}$$

$$I_h(t - \Delta t) = -\frac{R - \frac{2L}{\Delta t}}{R_{eq}} I_h(t - 2\Delta t) + \frac{\frac{4L}{\Delta t}}{R_{eq}^2} v(t - \Delta t)$$

D.4 RC Branch

Shown in Fig. D.1(a)

$$i(t) = \frac{1}{R_{eq}} v(t) + I(t - \Delta t)$$

where

$$R_{eq} = R + \frac{\Delta t}{2C}$$

$$I_h(t - \Delta t) = -\frac{R - \frac{\Delta t}{2C}}{R_{eq}} I_h(t - 2\Delta t) + \frac{2R}{R_{eq}^2} v(t - \Delta t)$$

D.5 LC Branch

Shown in Fig. D.1(a)

$$i(t) = \frac{1}{R_{eq}} v(t) + I(t - \Delta t)$$

where

$$R_{eq} = \frac{2L}{\Delta t} + \frac{\Delta t}{2C}$$

$$I_h(t-\Delta t) = \frac{V_{h,L}(t-\Delta t) + V_{h,C}(t-\Delta t)}{R_{eq}}$$

$$V_{h,L}(t-\Delta t) = -V_{h,L}(t-2\Delta t) - \frac{4L}{\Delta t} i(t-\Delta t)$$

$$V_{h,C}(t-\Delta t) = V_{h,C}(t-2\Delta t) + \frac{\Delta t}{C} i(t-\Delta t)$$

D.6 RLC Branch

Shown in Fig. D.1(a)

$$i(t) = \frac{1}{R_{eq}} v(t) + I(t-\Delta t)$$

where

$$R_{eq} = R + \frac{2L}{\Delta t} + \frac{\Delta t}{2C}$$

$$I_h(t-\Delta t) = \frac{V_{h,L}(t-\Delta t) + V_{h,C}(t-\Delta t)}{R_{eq}}$$

$$V_{h,L}(t-\Delta t) = -V_{h,L}(t-2\Delta t) - \frac{4L}{\Delta t} i(t-\Delta t)$$

$$V_{h,C}(t-\Delta t) = V_{h,C}(t-2\Delta t) + \frac{\Delta t}{C} i(t-\Delta t)$$

D.7 $RLCG$ Branch

Shown in Fig. D.1(a)

$$i(t) = \frac{1}{R_{eq}} v(t) + I(t-\Delta t)$$

where

$$R_{eq} = R + \frac{2L}{\Delta t} + \frac{G\frac{\Delta t}{2C}}{G + \frac{\Delta t}{2C}}$$

$$I_h(t-\Delta t) = \frac{V_{h,L}(t-\Delta t) + V_{h,CG}(t-\Delta t)}{R_{eq}}$$

$$V_{h,L}(t-\Delta t) = -V_{h,L}(t-2\Delta t) - \frac{4L}{\Delta t} i(t-\Delta t)$$

$$V_{h,CG}(t-\Delta t) = \frac{G - \frac{\Delta t}{2C}}{G + \frac{\Delta t}{2C}} V_{h,CG}(t-2\Delta t) + \frac{G^2 \frac{\Delta t}{C}}{(G + \frac{\Delta t}{2C})^2} i(t-\Delta t)$$

Note that G is in Ω.

E

C++ EMTP S-function Complete Source Code

The S-function program includes two files:

- `emtp.h`. This is the header file that defines all class and structures, and implements all functions including reading ATP data files and auxiliary functions.
- `emtp.cpp`. This is the main S-function program that implements S-function initialization, simulation loop and output, as well as EMTP initialization, simulation loop and solutions.

E.1 emtp.h

```
/*  File    : emtp.h
 *  Abstract:
 *
 *      Header file for C++ implementation of SimuLINK S-function
 *      for Real-time EMTP
 *
 *  by Xin Nie, Power Engineering Group
 *  Dept. of Electrical and Computer Engineering
 *  University of Alberta, Edmonton, Canada
 *  email: xnie@ece.ualberta.ca
 *  May 10, 2005
 */

#ifndef EMTP_H
#define EMTP_H

#include <math.h>
#include "matrix.h"

typedef math::matrix<double> CMatrix;

static const double PI = 3.1415927;
static const double Rinfs = 1.0e-10;        // infinite small resistance
static const double Rinfl = 1.0e10;         // infinite large resistance

// the abstract base class for other circuit element to derive
// this is an abstract class and cannot be instantiated

// Passive circuit element
class CPElement {
public:
    virtual void update(double&) = 0;       // update history terms
    virtual double getG() = 0;              // get conductance G
    virtual double getIh() = 0;             // get history term Ih
    virtual double getib() = 0;             // get branch current ib
};

// Active circuit element
class CAElement {
public:
    virtual double getG() = 0;              // get conductance G
    virtual double getIeq(double&) = 0;     // get history term I
    virtual double getib(double&, double&) = 0; // get branch current
};

// Transmission line element
class CTLElement {
public:
    virtual double getIhk() = 0;            // get history term Ihk
    virtual double getIhm() = 0;            // get history term Ihm
    virtual void update(double&, double&) = 0;  // update history terms
    virtual double getG() = 0;              // get conductance G
    virtual double getik() = 0;             // get branch current ik
    virtual double getim() = 0;             // get branch current im
};

// Switch element
class CSWElement {
public:
    virtual double getG() = 0;              // get conductance G
    virtual double getib(double&) = 0;      // get branch current
    virtual void update(bool&) = 0;         // update history terms
};

// get switch conductance G
/*
    st defines switch states
    0/false --- close
    1/true  --- open
*/
double getSWG(bool &st) {
    return st?(1.0/Rinfl):(1.0/Rinfs);
}

/*************************/
/*    Switch Elements    */
/*************************/

// class for switchs
class CSwitch : public CSWElement {
    bool st;
public:
    double getG();
    double getib(double &v);
    void update(bool &st);
};

inline void CSwitch::update(bool &sts) {
    st = sts;
}

inline double CSwitch::getib(double &v) {
    return v*getG();
}

inline double CSwitch::getG() {
    return getSWG(st);
}

/*************************/
/*    Passive Elements   */
/*************************/

// class for R branch
// o----R----o
class CR : public CPElement {
    double R;
    double ib;
public:
    CR(double &Rs);
    void update(double &v);
    double getG();
    double getIh();
    double getib();
};

inline CR::CR(double &Rs) {
    R = Rs;
}

inline void CR::update(double &v) {
    ib = v/R;
}

inline double CR::getG() {
    return 1.0/R;
}

inline double CR::getIh() {
    return 0.0;
}

inline double CR::getib() {
    return ib;
}

// class for L branch
// o----L----o
class CL : public CPElement {
    double A, B;
    double Req;
    double Ih;
    double ib;
```

```
public:
    CL(double &L, double &dt);
    void update(double &v);
    double getG();
    double getIh();
    double getib();
};

inline CL::CL(double &L, double &dt) {
    // discretization
    Req = 2.0*L/dt;
    // history term
    // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
    A = 1;
    B = 2.0/Req;
    Ih = 0.0;
}

inline void CL::update(double &v) {
    ib = v/Req + Ih;
    Ih = A * Ih + B * v;
}

inline double CL::getG() {
    return 1.0/Req;
}

inline double CL::getIh() {
    return -Ih;
}

inline double CL::getib() {
    return ib;
}

// class for C branch
// o----C----o
class CC : public CPElement {
    double A, B;
    double Req;
    double Ih;
    double ib;
public:
    CC(double &C, double &dt);
    void update(double &v);
    double getG();
    double getIh();
    double getib();
};

inline CC::CC(double &C, double &dt) {
    // discretization
    Req = dt/(2.0*C);
    // history term
    // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
    A = -1;
    B = 2.0/Req;
    Ih = 0.0;
}

inline void CC::update(double &v) {
    ib = v/Req - Ih;
    Ih = A * Ih + B * v;
}

inline double CC::getG() {
    return 1.0/Req;
}

inline double CC::getIh() {
    return Ih;
}

inline double CC::getib() {
    return ib;
}

// class for RL branch
// o----R--L----o
class CRL : public CPElement {
    double A, B;
    double Req;
    double Ih;
    double ib;
public:
    CRL(double &R, double &L, double &dt);
    void update(double &v);
    double getG();
    double getIh();
    double getib();
};

inline CRL::CRL(double &R, double &L, double &dt) {
    // discretization
    double Ld = 2.0*L/dt;
    Req = R+Ld;
    // history term
    // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
    A = -(R-Ld)/Req;
    B = (2.0*Ld)/(Req*Req);
    Ih = 0.0;
}

inline void CRL::update(double &v) {
    ib = v/Req + Ih;
    Ih = A * Ih + B * v;
}

inline double CRL::getG() {
    return 1.0/Req;
}

inline double CRL::getIh() {
    return -Ih;
}

inline double CRL::getib() {
    return ib;
}

// class for RC branch
// o----R--C----o
class CRC : public CPElement {
    double A, B;
    double Req;
    double Ih;
    double ib;
public:
    CRC(double &R, double &C, double &dt);
    void update(double &v);
    double getG();
    double getIh();
    double getib();
};

inline CRC::CRC(double &R, double &C, double &dt) {
    // discretization
    double Cd = dt/(2.0*C);
    Req = R+Cd;
    // history term
    // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
    A = -(R-Cd)/Req;
    B = (2.0*R)/(Req*Req);
    Ih = 0.0;
}
```

```
inline void CRC::update(double &v) {
     ib = v/Req - Ih;
     Ih = A * Ih + B * v;
}

inline double CRC::getG() {
     return 1.0/Req;
}

inline double CRC::getIh() {
     return Ih;
}

inline double CRC::getib() {
     return ib;
}

// class for LC branch
// o----L--C----o
class CLC : public CPElement {
     double A[2], B[2], Vh[2];
     double Req,ib;
public:
     CLC(double &L, double &C, double &dt);
     void update(double &v);
     double getIh();
     double getG();
     double getib();
};

inline CLC::CLC(double &L, double &C, double &dt) {
     // discretization
     double Ld = 2.0*L/dt;
     double Cd = dt/(2.0*C);
     Req = Ld+Cd;
     // history term
     // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
     A[0] = -1.0;
     B[0] = -2.0*Ld;
     Vh[0] = 0.0;
     A[1] = 1;
     B[1] = 2.0*Cd;
     Vh[1] = 0.0;
}

inline void CLC::update(double &v) {
     ib = v/Req - getIh();
     Vh[0] = A[0]*Vh[0] + B[0]*ib;
     Vh[1] = A[1]*Vh[1] + B[1]*ib;
}

inline double CLC::getIh() {
     return (Vh[0]+Vh[1])/Req;
}

inline double CLC::getG() {
     return 1.0/Req;
}

inline double CLC::getib() {
     return ib;
}

// class for RLC branch
// o----R--L--C----o
class CRLC : public CPElement {
     double A[2], B[2], Vh[2];
     double Req,ib;
public:
     CRLC(double &R, double &L, double &C, double &dt);
     void update(double &v);
     double getIh();
     double getG();
     double getib();
};

inline CRLC::CRLC(double &R, double &L, double &C, double &dt) {
     // discretization
     double Ld = 2.0*L/dt;
     double Cd = dt/(2.0*C);
     Req = R+Ld+Cd;
     // history term
     // I(t-dt) = A*I(t-2*dt)+B*v(t-dt)
     A[0] = -1.0;
     B[0] = -2.0*Ld;
     Vh[0] = 0.0;
     A[1] = 1;
     B[1] = 2.0*Cd;
     Vh[1] = 0.0;
}

inline void CRLC::update(double &v) {
     ib = v/Req - getIh();
     Vh[0] = A[0]*Vh[0] + B[0]*ib;
     Vh[1] = A[1]*Vh[1] + B[1]*ib;
}

inline double CRLC::getIh() {
     return (Vh[0]+Vh[1])/Req;
}

inline double CRLC::getG() {
     return 1.0/Req;
}

inline double CRLC::getib() {
     return ib;
}

// RLCG branch
//              ---C---
// o----R--L--|       |----o
//              ---G---
class CRLCG : public CPElement {
     double A[2], B[2], Vh[2];
     double Req,ib;
public:
     CRLCG(double &R, double &L, double &C, double &G, double &dt);
     void update(double &v);
     double getIh();
     double getG();
     double getib();
};

inline CRLCG::CRLCG(double &R, double &L, double &C, double &G, double &dt) {
     // discretization
     double Ld = 2.0*L/dt;
     double Cd = dt/(2.0*C);
     Req = R+Ld+G*Cd/(G+Cd);
     // history term
     // V(t-dt) = A*V(t-2*dt)+B*i(t-dt)
     A[0] = -1.0;
     B[0] = -2.0*Ld;
     Vh[0] = 0.0;
     A[1] = (G-Cd)/(G+Cd);
     B[1] = 2.0*G*G*Cd/((G+Cd)*(G+Cd));
     Vh[1] = 0.0;
}

inline void CRLCG::update(double &v) {
     ib = v/Req - getIh();
     Vh[0] = A[0]*Vh[0] + B[0]*ib;
     Vh[1] = A[1]*Vh[1] + B[1]*ib;
}

inline double CRLCG::getIh() {
```

```
    return (Vh[0]+Vh[1])/Req;
}

inline double CRLCG::getG() {
    return 1.0/Req;
}

inline double CRLCG::getib() {
    return ib;
}

/***************************/
/*  Trans. Line Elements  */
/***************************/

// Marti's frequency-dependent transmission line model
class CFdtl : public CTLElement {
    double *Vk, *Vm, *Fk, *Fm, *Sk, *Sm;
    double *A, *B, *M, *P, *Q, *K;
    double Vks, Vms, Bk, Bm;
    double dt;
    const int NZc, NP, NTau;
    void ArrangeF();
    double Zceq;
    double ik, im;
public:
    CFdtl(double &ZcCon, double* ZcRes, double* ZcPol, int sNZc,
        double &Tau, double* PRes, double* PPol, int sNP, double &dt);
    double getIhk();
    double getIhm();
    void update(double &vk, double &vm);
    void updateBkm();
    double getG();
    double getik();
    double getim();
    ~CFdtl();
};

CFdtl::CFdtl(double &ZcCon,                 // constant term of Zc
             double* ZcRes,                 // residues of Zc
             double* ZcPol,                 // poles of Zc
             int sNZc,                      // order of Zc
             double &Tau,                   // traveling time
             double* PRes,                  // residues of A1/P
             double* PPol,                  // poles of A1/P
             int sNP,                       // order of Zc
             double &sdt) : NZc(sNZc), NP(sNP), NTau((int) (Tau/sdt)) {
    double R, C, Cd, alp, p, Zt;
    int i;
    dt = sdt;
    A = new double[NZc];
    B = new double[NZc];
    M = new double[NP];
    P = new double[NP];
    Q = new double[NP];
    K = new double[NP];
    // history terms
    Vk = new double[NZc];
    Vm = new double[NZc];
    Fk = new double[NTau+1];
    Fm = new double[NTau+1];
    Sk = new double[NP];
    Sm = new double[NP];

    // discretized Characteristic Impedance coefficients
    Zt = ZcCon;
    for(i=0;i<NZc;i++) {
        // find equivalent circuit
        R = ZcRes[i]/ZcPol[i];
        C = 1/ZcRes[i];
        // discretization
        Cd = dt/(2.0*C);
        Zt += R*Cd/(R+Cd);
        // V(t-dt) = A*V(t-2*dt)+B*i(t-dt)
        A[i] = (R-Cd)/(R+Cd);
        B[i] = 2.0*R*R*Cd/((R+Cd)*(R+Cd));
        Vk[i] = 0.0;
        Vm[i] = 0.0;
    }
    Zceq=Zt;
    // Recursive Convolution coefficients
    for(i=0;i<NP;i++) {
        // b(t)=M*b(t-dt)+P*f(t-tau)+Q*f(t-tau-dt)
        p=PPol[i];
        alp = exp(-p*dt);
        M[i] = alp;
        P[i] = 1/p*(1-(1-alp)/(p*dt));
        Q[i] = 1/p*((1-alp)/(p*dt)-alp);
        Sk[i] = 0.0;
        Sm[i] = 0.0;
        K[i] = PRes[i];
    }
    for(i=0;i<NTau+1;i++) {
        Fk[i] = 0.0;
        Fm[i] = 0.0;
    }
    Vks = 0.0;
    Vms = 0.0;
    Bk = 0.0;
    Bm = 0.0;
}

// arrange the two Forward traveling function so that the first
// element of the array is always Fk(t-tau-dt) and Fm(t-tau-dt)
inline void CFdtl::ArrangeF() {
    double Fkt = Fk[0];
    double Fmt = Fm[0];
    for(int i=0;i<NTau;i++) {
        Fk[i] = Fk[i+1];
        Fm[i] = Fm[i+1];
    }
    Fk[NTau] = Fkt;
    Fm[NTau] = Fmt;
}

inline double CFdtl::getIhk() {
    return (Vks+Bk)/Zceq;
}

inline double CFdtl::getIhm() {
    return (Vms+Bm)/Zceq;
}

inline double CFdtl::getG() {
    return 1.0/Zceq;
}

inline double CFdtl::getik() {
    return ik;
}

inline double CFdtl::getim() {
    return im;
}

inline void CFdtl::update(double &vk, double &vm) {
    ik = vk/Zceq-getIhk();
    im = vm/Zceq-getIhm();
    Vks = 0.0;
    Vms = 0.0;
    // update history terms of Zc
    for(int i=0;i<NZc;i++) {
        Vk[i] = A[i] * Vk[i] + B[i] * ik;
        Vm[i] = A[i] * Vm[i] + B[i] * im;
        Vks += Vk[i];
        Vms += Vm[i];
```

```
    }
    Fk[0] = 2.0 * vk - Bk;
    Fm[0] = 2.0 * vm - Bm;
    ArrangeF();
}

// recursive convolution
inline void CFdtl::updateBkm() {
    Bk = 0.0;
    Bm = 0.0;
    for(int i=0;i<NP;i++) {
        Sk[i] = M[i] * Sk[i] + P[i] * Fm[1] + Q[i] * Fm[0];
        Bk += K[i] * Sk[i];
        Sm[i] = M[i] * Sm[i] + P[i] * Fk[1] + Q[i] * Fk[0];
        Bm += K[i] * Sm[i];
    }
}

inline CFdtl::~CFdtl() {
    delete []A;
    delete []B;
    delete []M;
    delete []P;
    delete []Q;
    delete []K;
    delete []Vk;
    delete []Vm;
    delete []Fk;
    delete []Fm;
    delete []Sk;
    delete []Sm;
}

/**************************/
/*    Active Elements     */
/**************************/

// voltage source with resistance
class CVsr : public CAElement {
    const double mag, f, pha;
    const double R;
public:
    CVsr(double &Rs, double &mags, double &fs, double &phas);
    double getIeq(double &t);
    double getG();
    double getib(double &v, double &t);
};

CVsr::CVsr(double &Rs, double &mags, double &fs,
           double &phas) : R(Rs), mag(mags), f(fs), pha(phas) {}

inline double CVsr::getIeq(double &t) {
    return (mag*cos(2.0*PI*f*t+pha*PI/180))/R;
}

inline double CVsr::getG() {
    return 1.0/R;
}

inline double CVsr::getib(double &v, double &t) {
    return (getIeq(t)-v/R);
}

// ideal voltage source
class CVs : public CAElement {
    const double mag, f, pha;
    const double R;
public:
    CVs(double &mags, double &fs, double &phas);
    double getIeq(double &t);
    double getG();
    double getib(double &v, double &t);
};

CVs::CVs(double &mags, double &fs,
           double &phas) : R(Rinfs), mag(mags), f(fs), pha(phas) {}

inline double CVs::getIeq(double &t) {
    return (mag*cos(2.0*PI*f*t+pha*PI/180))/R;
}

inline double CVs::getG() {
    return 1.0/R;
}

inline double CVs::getib(double &v, double &t) {
    return (getIeq(t)-v/R);
}

// ideal current source
class CIs : public CAElement {
    const double mag, f, pha;                 // phase shift in degree
    const double R;
public:
    CIs(double &mags, double &fs, double &phas);
    double getIeq(double &t);
    double getG();
    double getib(double &v, double &t);
};

CIs::CIs(double &mags, double &fs,
           double &phas) : R(Rinfl), mag(mags), f(fs), pha(phas) {}

inline double CIs::getIeq(double &t) {
    return mag*cos(2.0*PI*f*t+pha*PI/180);
}

inline double CIs::getG() {
    return 0.0;
}

inline double CIs::getib(double &v, double &t) {
    return getIeq(t);
}

/**************************/
/*  Auxiliary functions   */
/**************************/

// double to bool
bool dtob(const double &val) {
    if(int(val)>=1)
        return true;
    else
        return false;
}

// integer power function (2^x), which is not available in GCC math library
int pow2(const int &val) {
    int pow = 1;
    for(int i=0;i<val;i++) {
        pow *= 2;
    }
    return pow;
}

// integer value to bool bits
bool itoba(const int &val, const int &width, bool *a) {
    if(val<pow2(width)) {
        int rem=val;
        for(int i=width-1;i>=0;i--) {
            a[i]=(rem/pow2(i))?true:false;
            rem=rem%pow2(i);
        }
        return 0;                          // succeed
    }
```

```
    else
        return 1;                           // failed, val is too big
}

/*******************************************************/
/*  classes and functions for reading ATP data files  */
/*******************************************************/

enum etype {TR, TL, TC, TLC, TRL, TRC, TRLC, TRLCG,
            TVsr, TVs, TIs, TSW, TFDTL};    // type of elements
enum ecirtype {TBranch, TShunt};            // type of branch
// type of base class
enum ecls {TPElement, TAElement, TTLElement, TSWElement};

// an exception class derived from the class logic_error, which is used for
// exception handling in the EMTP program
class emtp_error : public logic_error {
public:
    emtp_error (const string &what_arg) : logic_error(what_arg) {}
};

// structure for branch elements (connecting between two nodes)
struct CBranch {                            // branch
    etype type;                             // class type of the element
    ecls cls;                               // base class type of the element
    string node1;                           // Node 1 name
    string node2;                           // Node 2 name
    int pos1;                               // position index of Node 1
    int pos2;                               // position index of Node 2
    void* data;                             // object of the element
};

// structure for shunt elements (connecting between one node to ground)
struct CShunt {                             // shunt or load
    etype type;                             // class type of the element
    ecls cls;                               // base class type of the element
    string node;                            // Node name
    int pos;                                // position index of Node
    void* data;                             // object of the element
};

struct CIout {                              // current output
    ecirtype type;                          // Branch / Shunt
    int index;                              // index in branch/shunt vector
};

struct CVout {                              // voltage output
    string node;                            // Node name
    int index;                              // index in voltage vector
};

struct CSWNode {                            // location of switches
    ecirtype type;                          // Branch / Shunt
    int index;                              // index in branch/shunt vector
};

typedef vector<string> CNVec;
typedef vector<CBranch> CBVec;
typedef vector<CShunt> CSVec;
typedef vector<CIout> CIVec;
typedef vector<CVout> CVVec;
typedef vector<CSWNode> CSWVec;
typedef vector<CMatrix> CMVec;

typedef vector<CNVec> CModeNVec;
typedef vector<CBVec> CModeBVec;
typedef vector<CSVec> CModeSVec;
typedef vector<CIVec> CModeIVec;
typedef vector<CVVec> CModeVVec;
typedef vector<CSWVec> CModeSWVec;
typedef vector<CMVec> CModeMVec;

// obtain the G matrix corresponding to the current switch states
CMatrix &getCurrentG(CMVec &Gmx, bool *cst, const int &width) {
    int idx = 0;
    for(int i=0;i<width;i++) {
        int tmp = (int) cst[i];
        int pow2 = 1;
        for(int j=0;j<i;j++) {
            pow2 *= 2;
        }
        idx += tmp*pow2;
    }
    return Gmx[idx];
}

// find position of the nodes in the node vector
bool findnode(const CNVec &nodes, const string &node, int &pos) {
    for(int i=0;i<nodes.size();i++) {
        if(nodes.at(i) == node) {
            pos = i;
            return true;
        }
    }
    return false;
}

// right trim the string if blank characters exist
string rtrim(const string &str) {
    string tstr(str);
    int len = tstr.length();
    while (len != 0) {
        if(tstr[len-1] == ' ') {
            tstr.erase(len-1,1);
            len--;
        }
        else
            break;
    }
    return tstr;
}

// left trim the string if blank characters exist
string ltrim(const string &str) {
    string tstr(str);
    int len = tstr.length();
    while (len != 0) {
        if(tstr[0] == ' ') {
            tstr.erase(0,1);
            len--;
        }
        else
            break;
    }
    return tstr;
}

// trim the string if blank characters exist
string trim(const string &str) {
    string tstr(str);
    return ltrim(rtrim(tstr));
}

// check blank strings
bool isblank(const string &str) {
    string tmpstr = string(str);
    return (ltrim(tmpstr)).length()?false:true;
}

// swap two strings
void swapstr(string &str1, string &str2) {
    string tmp;
    tmp = str1;
    str1 = str2;
    str2 = tmp;
}
```

```

// make an uppercase copy of s:
string uppercase(const string &s) {
  char* buf = new char[s.length()];
  s.copy(buf, s.length());
  for(int i = 0; i < s.length(); i++)
    buf[i] = toupper(buf[i]);
  string r(buf, s.length());
  delete buf;
  return r;
}

// make a lowercase copy of s:
string lowercase(const string &s) {
  char* buf = new char[s.length()];
  s.copy(buf, s.length());
  for(int i = 0; i < s.length(); i++)
    buf[i] = tolower(buf[i]);
  string r(buf, s.length());
  delete buf;
  return r;
}

/*  read an RLC branch for ATP data file
    return values:
        0 --- a RLC branch has been successfully read
        1 --- end of file
        2 --- end of branch card
        3 --- $include encountered
        4 --- C BEGIN FDNE encountered
        5 --- BEGIN NEW DATA CASE encountered
        6 --- C END FDNE encountered
        7 --- frequency-dependent line card found
*/
int readrlc(const string &fname,            // ATP data file name
            ifstream &dat,                  // file stream to read file
            int &linenum,                   // line number counter
            string &buf,                    // string buffer for the file
            string &node1,                  // node 1
            string &node2,                  // node 2
            double &r,                      // resistance
            double &l,                      // inductance
            double &c,                      // capacitance
            int &output,                    // request for output
            bool &vintage                   // precision format for RLC
            ) throw(emtp_error) {
    while(!getline(dat,buf).eof()) {
        // In Linux system, reading windows-format text files will have
        // a '\r' character in the end of string buffer. It has to be
        // removed in order to execute the program properly.
        if(buf[buf.size()-1] == '\r') buf = buf.erase(buf.size()-1,1);
        linenum++;
        output = 0;
        if(buf.size()>=80) {
            if(buf[79] == '1')
                output = 1;
            //else if(buf[79] == '2')       // voltage output in unsupported
            //  output = 2;
            buf = buf.substr(0,79);         // truncate
        }
        r=0.0;l=0.0;c=0.0;
        buf = rtrim(buf);
        int cnt = buf.length();
        if(cnt==0) continue;                // blank line
        // compare the first letter
        if(buf[0]=='C'||buf[0]=='c') {
            if(uppercase(buf.substr(1,11))==" BEGIN FDNE")
                return 4;                   // BEGIN FDNE found
            else if(uppercase(buf.substr(1,9))==" END FDNE")
                return 6;                   // END FDNE found
            else
                continue;                   // comment line
        }
        if(buf[0]=='/') return 2;           // end of branch card
        if(buf[0]=='-') return 7;           // frequency-dependent line found
        string tmpstr = uppercase(buf.substr(0,5));
        if(tmpstr=="BEGIN") {
            if(uppercase(rtrim(buf)) == "BEGIN NEW DATA CASE") {
                if(!getline(dat,buf).eof()) {
                    // find BLANK
                    if(buf[buf.size()-1] == '\r')
                        buf = buf.erase(buf.size()-1,1);
                    linenum++;
                    if(uppercase(rtrim(buf)) != "BLANK") {
                        if(buf[0]=='C' || buf[0]=='c')
                            return 5;   // BEGIN NEW DATA CASE encountered
                        else {
                            stringstream errstr;
                            errstr << fname << " - Invalid data read at line "
                                << linenum << " : \"" << buf << "\", "
                                << "comment card must be followed with "
                                << "BEGIN NEW DATA CASE";
                            throw emtp_error(errstr.str());
                        }
                    }
                    else
                        return 1;           // end of file
                }
                else {
                    stringstream errstr;
                    errstr << fname << " - Invalid data read at line "
                        << linenum << " : \"" << buf << "\"";
                    throw emtp_error(errstr.str());
                }
            }
            else
                continue;                   // ignore
        }
        if(tmpstr=="BLANK") {
            if(uppercase(rtrim(buf)) == "BLANK")
                return 1;                   // end of file
            else
                continue;
        }
        if(buf[0]=='$') {                   // ATP variables
            if(uppercase(buf.substr(1,7))=="INCLUDE")
                return 3;                   // $include branches
            else if(uppercase(buf.substr(1,7))=="VINTAGE") {
                string valstr = ltrim(buf.substr(8,buf.size()-8));
                if(valstr[0]==',') {
                    valstr.erase(0,1);
                    valstr=ltrim(valstr);
                }
                if(valstr=="1") {
                    vintage = true;
                }
                else if(valstr=="0") {
                    vintage = false;
                }
                else {
                    stringstream errstr;
                    errstr << fname << " - Invalid data read at line "
                        << linenum << " : \"" << buf << "\", "
                        << "invalid variable assignment";
                    throw emtp_error(errstr.str());
                }
                continue;
            }
            else
                continue;                   // ignore
        }
        node1 = uppercase(buf.substr(2,6));
        node2 = uppercase(buf.substr(8,6));
        string resis, induc, capac;
        if(vintage) {
            if(cnt>=74) {                   // R, L, C
```

```
                resis = buf.substr(26,16);
                induc = buf.substr(26+16,16);
                capac = buf.substr(26+2*16,16);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
                l = isblank(induc)?0.0:(1.0e-3*atof(ltrim(induc).c_str()));
                c = isblank(capac)?0.0:(1.0e-6*atof(ltrim(capac).c_str()));
            }
            else if(cnt>=58) {              // R, L
                resis = buf.substr(26,16);
                induc = buf.substr(26+16,16);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
                l = isblank(induc)?0.0:(1.0e-3*atof(ltrim(induc).c_str()));
            }
            else if(cnt>=42) {              // R
                resis = buf.substr(26,16);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
            }
            else {
                stringstream errstr;
                errstr << fname << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\"";
                throw emtp_error(errstr.str());
            }
        }
        else {
            if(cnt>=44) {                   // R, L, C
                resis = buf.substr(26,6);
                induc = buf.substr(26+6,6);
                capac = buf.substr(26+2*6,6);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
                l = isblank(induc)?0.0:(1.0e-3*atof(ltrim(induc).c_str()));
                c = isblank(capac)?0.0:(1.0e-6*atof(ltrim(capac).c_str()));
            }
            else if(cnt>=38) {              // R, L
                resis = buf.substr(26,6);
                induc = buf.substr(26+6,6);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
                l = isblank(induc)?0.0:(1.0e-3*atof(ltrim(induc).c_str()));
            }
            else if(cnt>=32) {              // R
                resis = buf.substr(26,6);
                r = isblank(resis)?0.0:atof(ltrim(resis).c_str());
            }
            else {
                stringstream errstr;
                errstr << fname << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\"";
                throw emtp_error(errstr.str());
            }
        }
        break;
    }
    if(dat.eof())
        return 1;                           // end of reading data file
    else
        return 0;
}


int readfdne(const string &fname,           // ATP data file name
             ifstream &fdnedat,             // file stream for reading file
             string &buf,                   // string buffer for the file
             double &dt,                    // time step size
             CNVec &nodes,                  // node vector
             CBVec &branches,               // branch vector
             CSVec &shunts,                 // shunt vector
             bool &vintage,                 // precision format for RLC
             int &linenum                   // line number counter
             ) throw(emtp_error) {
    if(!fdnedat.is_open()) throw emtp_error(fname+" - File open error.");
    string cn;
    CBranch cb;
    CShunt cs;
    // read file
    string node1, node2, node3, node4, node5, node6;
    int output = 0;
    double r, l, c, g, tmp1, tmp2;
    int pos;
    // three types of branches: R, RL, RLCG
    while(true) {
        int retval = readrlc(fname, fdnedat, linenum, buf, node1, node2,
            r, l, c, output, vintage);
        if(retval) return retval;
        if(!isblank(node1) && !isblank(node2)) {
            // possible a branch
            if(node1.substr(1,2)=="__" && node2.substr(1,2)=="__") {
                // a branch
                if(l==0.0) {                // R
                    cb.node1 = node1;
                    cb.node2 = node2;
                    cb.type = TR;
                    cb.cls = TPElement;
                    cb.data = new CR(r);
                    if(!findnode(nodes, node1, pos)) {
                        cn = string(node1);
                        nodes.push_back(cn);
                        cb.pos1 = nodes.size() - 1;
                    }
                    else
                        cb.pos1 = pos;
                    if(!findnode(nodes, node2, pos)) {
                        cn = string(node2);
                        nodes.push_back(cn);
                        cb.pos2 = nodes.size() - 1;
                    }
                    else
                        cb.pos2 = pos;
                    branches.push_back(cb);
                }
                else {                      // RL
                    cb.node1 = node1;
                    cb.node2 = node2;
                    cb.type = TRL;
                    cb.cls = TPElement;
                    cb.data = new CRL(r, l, dt);
                    if(!findnode(nodes, node1, pos)) {
                        cn = string(node1);
                        nodes.push_back(cn);
                        cb.pos1 = nodes.size() - 1;
                    }
                    else
                        cb.pos1 = pos;
                    if(!findnode(nodes, node2, pos)) {
                        cn = string(node2);
                        nodes.push_back(cn);
                        cb.pos2 = nodes.size() - 1;
                    }
                    else
                        cb.pos2 = pos;
                    branches.push_back(cb);
                }
            }
            else if(node1.substr(1,2)=="__" ||
                node2.substr(1,2)=="__") {  // RLCG
                // read one more line
                bool isbranch = false;
                if(!readrlc(fname, fdnedat, linenum, buf, node3, node4,
                    g, tmp1, c, output, vintage)) {
                    if(isblank(node3) || node3.substr(1,2)=="__")
                        swapstr(node3, node4);
                    if(g == 0.0) {          // read G
                        if(readrlc(fname, fdnedat, linenum, buf, node5, node6,
                            g, tmp1, tmp2, output, vintage)) {
                            stringstream errstr;
                            errstr << fname << " - Invalid data read at line "
                                << linenum << " : \"" << buf << "\"";
```

```
                        throw emtp_error(errstr.str());
                    }
                }
                else {                      // read C
                    if(readrlc(fname, fdnedat, linenum, buf, node5, node6,
                        tmp1, tmp2, c, output, vintage)) {
                        stringstream errstr;
                        errstr << fname << " - Invalid data read at line "
                            << linenum << " : \"" << buf << "\"";
                        throw emtp_error(errstr.str());
                    }
                }
                if(isblank(node5) || node5.substr(1,2)=="__")
                    swapstr(node5, node6);
                if(!isblank(node6))         // RLCG branch
                    isbranch = true;
                // else RLCG shunt
            }
            if(node4!=node6 || node3!=node5) {
                stringstream errstr;
                errstr << fname << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\"";
                throw emtp_error(errstr.str());
            }
            if(node1 == node3) node1 = node2;
            node2 = node4;
            if(isbranch) {                  // RLCG branch
                cb.node1 = node1;
                cb.node2 = node2;
                cb.type = TRLCG;
                cb.cls = TPElement;
                cb.data = new CRLCG(r, l, c, g, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
            }
            else {                          // RLCG shunt
                cs.node = node1;
                cs.type = TRLCG;
                cs.cls = TPElement;
                cs.data = new CRLCG(r, l, c, g, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
            }
        }
        else {
            stringstream errstr;
            errstr << fname << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
    }
    else {                                  // L
        if(isblank(node1)) node1 = node2;
        if(r == 0.0) {
            cs.node = node1;
            cs.type = TL;
            cs.cls = TPElement;
            cs.data = new CL(l, dt);
            if(!findnode(nodes, node1, pos)) {
                cn = string(node1);
                nodes.push_back(cn);
                cs.pos = nodes.size() - 1;
            }
            else
                cs.pos = pos;
            shunts.push_back(cs);
        }
        else if(l == 0.0) {                 // R
            cs.node = node1;
            cs.type = TR;
            cs.cls = TPElement;
            cs.data = new CR(r);
            if(!findnode(nodes, node1, pos)) {
                cn = string(node1);
                nodes.push_back(cn);
                cs.pos = nodes.size() - 1;
            }
            else
                cs.pos = pos;
            shunts.push_back(cs);
        }
        else {                              // RL
            cs.node = node1;
            cs.type = TRL;
            cs.cls = TPElement;
            cs.data = new CRL(r, l, dt);
            if(!findnode(nodes, node1, pos)) {
                cn = string(node1);
                nodes.push_back(cn);
                cs.pos = nodes.size() - 1;
            }
            else
                cs.pos = pos;
            shunts.push_back(cs);
        }
    }
  }
}

void readfdtlparam(const string &atpdatfn,  // ATP data file name
                   ifstream &atpdat,        // file stream for the file
                   string &buf,             // string buffer for the file
                   const int &N,            // order of Zc/A1
                   double *para,            // parameters
                   int &linenum             // line number counter
                   ) throw(emtp_error) {
    int i, j, k=0;
    for(i=0;i<N/3;i++) {
        if(!getline(atpdat,buf).eof()) {
            if(buf[buf.size()-1] == '\r')
                buf = buf.erase(buf.size()-1,1);
            linenum++;
            buf = rtrim(buf);
            for(j=0;j<3;j++) {
                para[k] = atof(ltrim(buf.substr(j*26,26)).c_str());
                k++;
            }
        }
        else {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
    }
    if(N%3!=0) {
        if(!getline(atpdat,buf).eof()) {
```

```cpp
            if(buf[buf.size()-1] == '\r')
                buf = buf.erase(buf.size()-1,1);
            linenum++;
            buf = rtrim(buf);
            for(j=0;j<N%3;j++) {
                para[k] = atof(ltrim(buf.substr(j*26,26)).c_str());
                k++;
            }
        }
        else {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
    }
}

void readfdtl(const string &atpdatfn,   // ATP data file name
              ifstream &atpdat,            // file stream for the file
              string &buf,                 // string buffer for the file
              double &dt,                  // time step size
              CNVec &nodes,                // node vector
              CBVec &branches,             // branch vector
              CSVec &shunts,               // shunt vector
              CIVec &curout,               // current output vector
              bool &vintage,               // precision format for RLC
              int &linenum                 // line number counter
              ) throw(emtp_error) {
    if(!atpdat.is_open()) throw emtp_error(atpdatfn+" - File open error.");
    string cn;
    CBranch cb;
    CShunt cs;
    CIout cio;
    // read file
    string node1, node2;
    int output = 0;
    bool isshunt;
    int pos;

    int NZc, NP;
    double Tau, ZcCon;
    double *ZcRes, *ZcPol, *PRes, *PPol;

    if(buf[1]!='1') {
        stringstream errstr;
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\", "
            << "multiphase lines are unsupported.";
        throw emtp_error(errstr.str());
    }
    if(buf.size()>=80) {
        if(buf[79] == '1')
            output = 1;
        buf = buf.substr(0,79);              // truncate
    }
    else if(rtrim(buf).size()<54) {
        stringstream errstr;
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\"";
        throw emtp_error(errstr.str());
    }
    if(buf[55]!='1') {
        stringstream errstr;
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\", "
            << "multiphase lines are unsupported.";
        throw emtp_error(errstr.str());
    }
    node1 = uppercase(buf.substr(2,6));
    node2 = uppercase(buf.substr(8,6));
    isshunt=false;
    if(isblank(node1)) {                     // shunt
        swapstr(node1, node2);
        isshunt=true;
    }
    else if(isblank(node2)) {                // still shunt
        isshunt=true;
    }

    // read Zc(s)
    if(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r')
            buf = buf.erase(buf.size()-1,1);
        linenum++;
        NZc = atoi(ltrim(buf.substr(0,8)).c_str());
        ZcCon = atof(ltrim(buf.substr(8,32)).c_str());
    }
    else {
        stringstream errstr;
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\"";
        throw emtp_error(errstr.str());
    }
    ZcRes = new double[NZc];
    ZcPol = new double[NZc];
    readfdtlparam(atpdatfn, atpdat, buf, NZc, ZcRes, linenum);
    readfdtlparam(atpdatfn, atpdat, buf, NZc, ZcPol, linenum);

    // read A1(s)/P(s)
    if(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r')
            buf = buf.erase(buf.size()-1,1);
        linenum++;
        NP = atoi(ltrim(buf.substr(0,8)).c_str());
        Tau = atof(ltrim(buf.substr(8,32)).c_str());
    }
    else {
        stringstream errstr;
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\"";
        throw emtp_error(errstr.str());
    }
    PRes = new double[NP];
    PPol = new double[NP];
    readfdtlparam(atpdatfn, atpdat, buf, NP, PRes, linenum);
    readfdtlparam(atpdatfn, atpdat, buf, NP, PPol, linenum);

    if(isshunt) {                            // shunt
        cs.node = node1;
        cs.type = TFDTL;
        cs.cls = TTLElement;
        cs.data = new CFdtl(ZcCon, ZcRes, ZcPol, NZc, Tau,
            PRes, PPol, NP, dt);
        if(!findnode(nodes, node1, pos)) {
            cn = string(node1);
            nodes.push_back(cn);
            cs.pos = nodes.size() - 1;
        }
        else
            cs.pos = pos;
        shunts.push_back(cs);
        if(output) {
            cio.type = TShunt;
            cio.index = shunts.size() - 1;
            curout.push_back(cio);
        }
    }
    else { // branch
        cb.node1 = node1;
        cb.node2 = node2;
        cb.type = TFDTL;
        cb.cls = TTLElement;
        cb.data = new CFdtl(ZcCon, ZcRes, ZcPol, NZc, Tau,
            PRes, PPol, NP, dt);
        if(!findnode(nodes, node1, pos)) {
```

```
            cn = string(node1);
            nodes.push_back(cn);
            cb.pos1 = nodes.size() - 1;
        }
        else
            cb.pos1 = pos;
        if(!findnode(nodes, node2, pos)) {
            cn = string(node2);
            nodes.push_back(cn);
            cb.pos2 = nodes.size() - 1;
        }
        else
            cb.pos2 = pos;
        branches.push_back(cb);
        if(output) {
            cio.type = TBranch;
            cio.index = branches.size() - 1;
            curout.push_back(cio);
        }
    }
    delete []ZcRes;
    delete []ZcPol;
    delete []PRes;
    delete []PPol;
    // ignore transformation matrix since this is a single-phase system
    for(int i=0;i<2;i++) {
        if(!getline(atpdat,buf).eof()) {
            linenum++;
        }
        else {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
    }
}

int readbranchcard(const string &atpdatfn,  // ATP data file name
                   ifstream &atpdat,        // file stream for reading file
                   string &buf,             // string buffer for the file
                   double &dt,              // time step size
                   CNVec &nodes,            // node vector
                   CBVec &branches,         // branch vector
                   CSVec &shunts,           // shunt vector
                   CIVec &curout,           // current output vector
                   int &linenum,            // line number counter
                   string &newcs            // new sorting card found
                   ) throw(emtp_error) {
    string cn;
    CBranch cb;
    CShunt cs;
    CIout cio;
    // read file
    string node1, node2;
    int output;
    double r, l, c;
    int retval, pos;
    bool searchfdne=false, isshunt, vintage=false;
    stringstream errstr;
    while(true) {
    retval=readrlc(atpdatfn, atpdat, linenum, buf, node1, node2,
        r, l, c, output, vintage);
    switch(retval) {
    case 0:                                 // succeed
        isshunt=false;
        if(isblank(node1)) {                // shunt
            swapstr(node1, node2);
            isshunt=true;
        }
        else if(isblank(node2)) {
            isshunt=true;
        }

    if(r==0.0) {                                // R=0
        if(l==0.0) {                            // C only
            if(isshunt) {                       // shunt
                cs.node = node1;
                cs.type = TC;
                cs.cls = TPElement;
                cs.data = new CC(c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
                if(output) {
                    cio.type = TShunt;
                    cio.index = shunts.size() - 1;
                    curout.push_back(cio);
                }
            }
            else {                              // branch
                cb.node1 = node1;
                cb.node2 = node2;
                cb.type = TC;
                cb.cls = TPElement;
                cb.data = new CC(c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
                if(output) {
                    cio.type = TBranch;
                    cio.index = branches.size() - 1;
                    curout.push_back(cio);
                }
            }
        }
        else if(c==0.0) {                       // L only
            if(isshunt) {                       // shunt
                cs.node = node1;
                cs.type = TL;
                cs.cls = TPElement;
                cs.data = new CL(l, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
                if(output) {
                    cio.type = TShunt;
                    cio.index = shunts.size() - 1;
                    curout.push_back(cio);
                }
            }
            else {                              // branch
                cb.node1 = node1;
                cb.node2 = node2;
```

```
                cb.type = TL;
                cb.cls = TPElement;
                cb.data = new CL(l, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
                if(output) {
                    cio.type = TBranch;
                    cio.index = branches.size() - 1;
                    curout.push_back(cio);
                }
            }
        }
        else {                              // LC
            if(isshunt) {                   // shunt
                cs.node = node1;
                cs.type = TLC;
                cs.cls = TPElement;
                cs.data = new CLC(l, c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
                if(output) {
                    cio.type = TShunt;
                    cio.index = shunts.size() - 1;
                    curout.push_back(cio);
                }
            }
            else {                          // branch
                cb.node1 = node1;
                cb.node2 = node2;
                cb.type = TLC;
                cb.cls = TPElement;
                cb.data = new CLC(l, c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
                if(output) {
                    cio.type = TBranch;
                    cio.index = branches.size() - 1;
                    curout.push_back(cio);
                }
            }
        }
        }
        else {                                  // R not equal to 0
            if(l==0.0) {                        // RC or R branch
                if(c==0.0) {                    // R
                    if(isshunt) {               // shunt
                        cs.node = node1;
                        cs.type = TR;
                        cs.cls = TPElement;
                        cs.data = new CR(r);
                        if(!findnode(nodes, node1, pos)) {
                            cn = string(node1);
                            nodes.push_back(cn);
                            cs.pos = nodes.size() - 1;
                        }
                        else
                            cs.pos = pos;
                        shunts.push_back(cs);
                        if(output) {
                            cio.type = TShunt;
                            cio.index = shunts.size() - 1;
                            curout.push_back(cio);
                        }
                    }
                    else {                      // branch
                        cb.node1 = node1;
                        cb.node2 = node2;
                        cb.type = TR;
                        cb.cls = TPElement;
                        cb.data = new CR(r);
                        if(!findnode(nodes, node1, pos)) {
                            cn = string(node1);
                            nodes.push_back(cn);
                            cb.pos1 = nodes.size() - 1;
                        }
                        else
                            cb.pos1 = pos;
                        if(!findnode(nodes, node2, pos)) {
                            cn = string(node2);
                            nodes.push_back(cn);
                            cb.pos2 = nodes.size() - 1;
                        }
                        else
                            cb.pos2 = pos;
                        branches.push_back(cb);
                        if(output) {
                            cio.type = TBranch;
                            cio.index = branches.size() - 1;
                            curout.push_back(cio);
                        }
                    }
                }
                else {                          // RC
                    if(isshunt) {               // shunt
                        cs.node = node1;
                        cs.type = TRC;
                        cs.cls = TPElement;
                        cs.data = new CRC(r, c, dt);
                        if(!findnode(nodes, node1, pos)) {
                            cn = string(node1);
                            nodes.push_back(cn);
                            cs.pos = nodes.size() - 1;
                        }
                        else
                            cs.pos = pos;
                        shunts.push_back(cs);
                        if(output) {
                            cio.type = TShunt;
                            cio.index = shunts.size() - 1;
                            curout.push_back(cio);
                        }
                    }
                    else {                      // branch
                        cb.node1 = node1;
```

```
                    cb.node2 = node2;
                    cb.type = TRC;
                    cb.cls = TPElement;
                    cb.data = new CRC(r, c, dt);
                    if(!findnode(nodes, node1, pos)) {
                        cn = string(node1);
                        nodes.push_back(cn);
                        cb.pos1 = nodes.size() - 1;
                    }
                    else
                        cb.pos1 = pos;
                    if(!findnode(nodes, node2, pos)) {
                        cn = string(node2);
                        nodes.push_back(cn);
                        cb.pos2 = nodes.size() - 1;
                    }
                    else
                        cb.pos2 = pos;
                    branches.push_back(cb);
                    if(output) {
                        cio.type = TBranch;
                        cio.index = branches.size() - 1;
                        curout.push_back(cio);
                    }
                }
            }
        }
        else if(c==0.0) {                   // RL
            if(isshunt) {                   // shunt
                cs.node = node1;
                cs.type = TRL;
                cs.cls = TPElement;
                cs.data = new CRL(r, l, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
                if(output) {
                    cio.type = TShunt;
                    cio.index = shunts.size() - 1;
                    curout.push_back(cio);
                }
            }
            else {                          // branch
                cb.node1 = node1;
                cb.node2 = node2;
                cb.type = TRL;
                cb.cls = TPElement;
                cb.data = new CRL(r, l, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
                if(output) {
                    cio.type = TBranch;
                    cio.index = branches.size() - 1;
                    curout.push_back(cio);
                }
            }
        }
        else {                              // RLC
            if(isshunt) {                   // shunt
                cs.node = node1;
                cs.type = TRLC;
                cs.cls = TPElement;
                cs.data = new CRLC(r, l, c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cs.pos = nodes.size() - 1;
                }
                else
                    cs.pos = pos;
                shunts.push_back(cs);
                if(output) {
                    cio.type = TShunt;
                    cio.index = shunts.size() - 1;
                    curout.push_back(cio);
                }
            }
            else {                          // branch
                cb.node1 = node1;
                cb.node2 = node2;
                cb.type = TRLC;
                cb.cls = TPElement;
                cb.data = new CRLC(r, l, c, dt);
                if(!findnode(nodes, node1, pos)) {
                    cn = string(node1);
                    nodes.push_back(cn);
                    cb.pos1 = nodes.size() - 1;
                }
                else
                    cb.pos1 = pos;
                if(!findnode(nodes, node2, pos)) {
                    cn = string(node2);
                    nodes.push_back(cn);
                    cb.pos2 = nodes.size() - 1;
                }
                else
                    cb.pos2 = pos;
                branches.push_back(cb);
                if(output) {
                    cio.type = TBranch;
                    cio.index = branches.size() - 1;
                    curout.push_back(cio);
                }
            }
        }
    }
    break;
case 1:                                     // end of file
    return 1;
case 2:                                     // end of branch card
    newcs = buf.substr(1,6);
    return 0;
case 3: // $include encountered
        // FDTL
        //readfdtl function
        // this functionaliy is not incorporated into this program
        // all frequency-dependent line model data must be included
        // in the main ATP data file.
    break;
case 4:                                     // C BEGIN FDNE encountered
    if(readfdne(atpdatfn, atpdat, buf, dt, nodes, branches, shunts,
        vintage, linenum) == 3) {
        string fdnefn=trim(buf.substr(8,buf.size()-8));
        if(fdnefn[0]==',') {
            fdnefn.erase(0,1);
            fdnefn=ltrim(fdnefn);
        }
        // read FDNE data file
```

```
            ifstream fdnedat(fdnefn.c_str());
            if(!fdnedat.is_open()) {
                throw emtp_error(fdnefn+" - File open error.");
            }
            int fdnelm=0;
            string fdnebuf;
            int val = readfdne(fdnefn, fdnedat, fdnebuf, dt, nodes, branches,
                shunts, vintage, fdnelm);
        }
        break;
    case 5:                                 // BEGIN NEW DATA CASE encountered
        newcs = "BNDC";
        return 1;
    case 6:                                 // C END FDNE encountered
        if(searchfdne) searchfdne = false;
        break;
    case 7: // fdtl card found
        readfdtl(atpdatfn, atpdat, buf, dt, nodes, branches, shunts, curout,
            vintage, linenum);
        break;
    default:
        errstr << atpdatfn << " - Invalid data read at line "
            << linenum << " : \"" << buf << "\"";
        throw emtp_error(errstr.str());
    }
    }
}

int readswitchcard(const string &atpdatfn,  // ATP data file name
                   ifstream &atpdat,        // file stream for reading file
                   string &buf,             // string buffer for the file
                   double &dt,              // time step size
                   CNVec &nodes,            // node vector
                   CBVec &branches,         // branch vector
                   CSVec &shunts,           // shunt vector
                   CIVec &curout,           // current output vector
                   CSWVec &sw,              // switch vector
                   int &linenum,            // line number counter
                   string &newcs            // sorting branch card found
                   ) throw(emtp_error) {
    string cn;
    CBranch cb;
    CShunt cs;
    CIout cio;
    CSWNode swnode;
    // read file
    string node1, node2;
    int output, pos;
    bool isshunt;
    while(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r')
            buf = buf.erase(buf.size()-1,1);
        linenum++;
        output = 0;
        if(buf.size()>=80) {
            if(buf[79] == '1')
                output = 1;
            //else if(buf[79] == '2')       // voltage output is unsupported
            //  output = 2;
            buf = buf.substr(0,79);         // truncate
        }
        buf = rtrim(buf);
        int cnt = buf.length();
        if(cnt==0) continue;                // blank line;
        // reading first letter
        if(buf[0]=='C'||buf[0]=='c')
            continue;                       // comment line
        if(buf[0]=='/') {
            newcs = buf.substr(1,6);
            return 0;                       // end of branch card
        }
        string tmpstr = uppercase(buf.substr(0,5));
        if(tmpstr=="BEGIN") {
            if(uppercase(rtrim(buf)) == "BEGIN NEW DATA CASE") {
                if(!getline(atpdat,buf).eof()) {
                    // find BLANK
                    if(buf[buf.size()-1] == '\r')
                        buf = buf.erase(buf.size()-1,1);
                    linenum++;
                    if(uppercase(rtrim(buf)) != "BLANK") {
                        if(buf[0]=='C' || buf[0]=='c')
                            newcs = "BNDC"; // BEGIN NEW DATA CASE encountered
                        else {
                            stringstream errstr;
                            errstr << atpdatfn
                                << " - Invalid data read at line "
                                << linenum << " : \"" << buf << "\", "
                                << "comment card must be followed with "
                                << "BEGIN NEW DATA CASE";
                            throw emtp_error(errstr.str());
                        }
                    }
                    return 1;
                }
                else {
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid data read at line "
                        << linenum << " : \"" << buf << "\"";
                    throw emtp_error(errstr.str());
                }
            }
            else
                continue;                       // ignore
        }
        if(tmpstr=="BLANK") {
            if(uppercase(rtrim(buf)) == "BLANK")
                return 1;                       // end of file
            else
                continue;
        }
        if(buf[0]=='$') {                       // ATP variables
            continue;                           // ignore
        }
        node1 = uppercase(buf.substr(2,6));
        node2 = uppercase(buf.substr(8,6));
        isshunt=false;
        if(isblank(node1)) {                    // shunt
            swapstr(node1, node2);
            isshunt=true;
        }
        else if(isblank(node2)) {
            isshunt=true;
        }
        if(isshunt) {                           // shunt
            cs.node = node1;
            cs.type = TSW;
            cs.cls = TSWElement;
            cs.data = new CSwitch;

            if(!findnode(nodes, node1, pos)) {
                cn = string(node1);
                nodes.push_back(cn);
                cs.pos = nodes.size() - 1;
            }
            else
                cs.pos = pos;
            shunts.push_back(cs);

            swnode.type = TShunt;
            swnode.index = shunts.size() - 1;
            sw.push_back(swnode);

            if(output) {
                cio.type = TShunt;
                cio.index = shunts.size() - 1;
                curout.push_back(cio);
```

```
            }
        }
        else {                                  // branch
            cb.node1 = node1;
            cb.node2 = node2;
            cb.type = TSW;
            cb.cls = TSWElement;
            cb.data = new CSwitch;

            if(!findnode(nodes, node1, pos)) {
                cn = string(node1);
                nodes.push_back(cn);
                cb.pos1 = nodes.size() - 1;
            }
            else
                cb.pos1 = pos;
            if(!findnode(nodes, node2, pos)) {
                cn = string(node2);
                nodes.push_back(cn);
                cb.pos2 = nodes.size() - 1;
            }
            else
                cb.pos2 = pos;
            branches.push_back(cb);

            swnode.type = TBranch;
            swnode.index = branches.size() - 1;
            sw.push_back(swnode);

            if(output) {
                cio.type = TBranch;
                cio.index = branches.size() - 1;
                curout.push_back(cio);
            }
        }
    }
    return 1;                                   // eof found

}

int readoutputcard(const string &atpdatfn,  // ATP data file name
                   ifstream &atpdat,        // file stream for reading file
                   string &buf,             // string buffer for the file
                   double &dt,              // time step size
                   CNVec &nodes,            // node vector
                   CBVec &branches,         // branch vector
                   CSVec &shunts,           // shunt vector
                   CVVec &volout,           // voltage output vector
                   int &linenum,            // line number counter
                   string &newcs             // new sorting card found
                   ) throw(emtp_error) {
    CVout vio;
    // read file
    string node;
    while(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r') buf = buf.erase(buf.size()-1,1);
        linenum++;
        buf = rtrim(buf);
        int cnt = buf.length();
        if(cnt==0) continue;                    // blank line;
        // first letter
        if(buf[0]=='C'||buf[0]=='c')
            continue;                           // comment line
        if(buf[0]=='/') {
            newcs = buf.substr(1,6);
            return 0;                           // end of branch card
        }
        string tmpstr = uppercase(buf.substr(0,5));
        if(tmpstr=="BEGIN") {
            if(uppercase(rtrim(buf)) == "BEGIN NEW DATA CASE") {
                if(!getline(atpdat,buf).eof()) {
                    // find BLANK
                    if(buf[buf.size()-1] == '\r')
                        buf = buf.erase(buf.size()-1,1);
                    linenum++;
                    if(uppercase(rtrim(buf)) != "BLANK") {
                        if(buf[0]=='C' || buf[0]=='c')
                            newcs = "BNDC"; // BEGIN NEW DATA CASE encountered
                        else {
                            stringstream errstr;
                            errstr << atpdatfn
                                << " - Invalid data read at line "
                                << linenum << " : \"" << buf << "\", "
                                << "comment card must be followed with "
                                << "BEGIN NEW DATA CASE";
                            throw emtp_error(errstr.str());
                        }
                    }
                    return 1;
                }
                else {
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid data read at line "
                        << linenum << " : \"" << buf << "\"";
                    throw emtp_error(errstr.str());
                }
            }
            else
                continue;                       // ignore
        }
        if(tmpstr=="BLANK") {
            if(uppercase(rtrim(buf)) == "BLANK")
                return 1;                       // end of file
            else
                continue;
        }
        if(buf[0]=='$') {                       // ATP variables
            continue;                           // ignore
        }
        buf = ltrim(buf);
        cnt = buf.size();
        if(cnt%6!=0) {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
        for(int i=0;i<cnt/6;i++) {
            node = uppercase(buf.substr(6*i,6));
            int pos;
            if(!findnode(nodes, node, pos)) {
                stringstream errstr;
                errstr << atpdatfn << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\", "
                    << "output node not exist.";
                throw emtp_error(errstr.str());
            }
            else {
                vio.node = node;
                vio.index = pos;
                volout.push_back(vio);
            }
        }
    }
    return 1;                                   // eof found
}

int readsourcecard(const string &atpdatfn,  // ATP data file name
                   ifstream &atpdat,        // file stream for reading file
                   string &buf,             // string buffer for the file
                   double &dt,              // time step size
                   CNVec &nodes,            // node vector
                   CBVec &branches,         // branch vector
                   CSVec &shunts,           // shunt vector
                   CIVec &curout,           // current output vector
                   int &linenum,            // line number counter
```

```
                    string &newcs              // new sorting card found
                    ) throw(emtp_error) {
    string cn;
    CShunt cs;
    // read file
    string node;
    int pos;
    while(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r')
            buf = buf.erase(buf.size()-1,1);
        linenum++;
        if(buf.size()>=80) {
            buf = buf.substr(0,40);            // truncate
        }
        buf = rtrim(buf);
        int cnt = buf.length();
        if(cnt==0) continue;                   // blank line;
        // first letter
        if(buf[0]=='C'||buf[0]=='c')
            continue;                          // comment line
        if(buf[0]=='/') {
            newcs = buf.substr(1,6);
            return 0;                          // end of branch card
        }
        string tmpstr = uppercase(buf.substr(0,5));
        if(tmpstr=="BEGIN") {
            if(uppercase(rtrim(buf)) == "BEGIN NEW DATA CASE") {
                if(!getline(atpdat,buf).eof()) {
                    // find BLANK
                    if(buf[buf.size()-1] == '\r')
                        buf = buf.erase(buf.size()-1,1);
                    linenum++;
                    if(uppercase(rtrim(buf)) != "BLANK") {
                        if(buf[0]=='C' || buf[0]=='c')
                            newcs = "BNDC"; // BEGIN NEW DATA CASE encountered
                        else {
                            stringstream errstr;
                            errstr << atpdatfn
                                << " - Invalid data read at line "
                                << linenum << " : \"" << buf << "\", "
                                << "comment card must be folllowed with "
                                << "BEGIN NEW DATA CASE";
                            throw emtp_error(errstr.str());
                        }
                    }
                    return 1;
                }
                else {
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid data read at line "
                        << linenum << " : \"" << buf << "\"";
                    throw emtp_error(errstr.str());
                }
            }
            else
                continue;                      // ignore
        }
        if(tmpstr=="BLANK") {
            if(uppercase(rtrim(buf)) == "BLANK")
                return 1;                      // end of file
            else
                continue;
        }
        if(buf[0]=='$') {                      // ATP variables
            continue;                          // ignore
        }
        if(buf.substr(0,2)!="14") {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\", "
                << "only voltage/current source type 14 supported";
            throw emtp_error(errstr.str());
        }
        node = uppercase(buf.substr(2,6));
        double mag = atof(ltrim(buf.substr(10,10)).c_str());
        double f = atof(ltrim(buf.substr(20,10)).c_str());
        double pha = atof(ltrim(buf.substr(30,10)).c_str());
        cs.node = node;
        if(buf.substr(8,2)=="-1") {            // current source
            cs.type = TIs;
            cs.data = new CIs(mag, f, pha);
        }
        else {                                 // voltage source
            cs.type = TVs;
            cs.data = new CVs(mag, f, pha);
        }
        cs.cls = TAElement;
        if(!findnode(nodes, node, pos)) {
            cn = string(node);
            nodes.push_back(cn);
            cs.pos = nodes.size() - 1;
        }
        else
            cs.pos = pos;
        shunts.push_back(cs);
    }
    return 1;                                  // eof encountered
}

int readdatacase(const string &atpdatfn,       // ATP data file name
                 ifstream &atpdat,             // file stream for reading file
                 double &dt,                   // time step size
                 double &f,                    // power frequency
                 CNVec &nodes,                 // node vector
                 CBVec &branches,              // branch vector
                 CSVec &shunts,                // shunt vector
                 CIVec &curout,                // current output vector
                 CVVec &volout,                // voltage output vector
                 CSWVec &sw,                   // switch vector
                 bool &searchdt,               // search dt or not
                 const bool &readdt,           // read dt or not
                 int &linenum
                 ) throw(emtp_error) {
    string buf, newcs;
    while(!getline(atpdat,buf).eof()) {
        if(buf[buf.size()-1] == '\r') buf = buf.erase(buf.size()-1,1);
        linenum++;
        buf = rtrim(buf);
        int cnt = buf.length();
        if(cnt==0) continue;                   // blank line;
        // first letter
        if(buf[0]=='C'||buf[0]=='c')
            continue;                          // comment line
        if(buf[0]=='$') {                      // ATP variables
            continue;                          // ignore
        }
        if(cnt < 5) {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
        string tmpstr = uppercase(buf.substr(0,5));
        if(tmpstr=="BEGIN") {
            if(uppercase(rtrim(buf)) == "BEGIN NEW DATA CASE") {
                if(!getline(atpdat,buf).eof()) {
                    // find BLANK
                    if(buf[buf.size()-1] == '\r')
                        buf = buf.erase(buf.size()-1,1);
                    linenum++;
                    if(uppercase(rtrim(buf)) == "BLANK")
                        return 1;              // end of file
                    else {
                        if(buf[0]=='C' || buf[0]=='c') {
                            searchdt = true;
                            continue;
```

```
                    }
                    else {
                        stringstream errstr;
                        errstr << atpdatfn
                            << " - Invalid data read at line "
                            << linenum << " : \"" << buf << "\", "
                            << "comment card must be folllowed with "
                            << "BEGIN NEW DATA CASE";
                        throw emtp_error(errstr.str());
                    }
                }
            }
            else {
                stringstream errstr;
                errstr << atpdatfn << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\"";
                throw emtp_error(errstr.str());
            }
        }
        continue;
    }
    if(tmpstr=="BLANK") {
        if(uppercase(rtrim(buf)) == "BLANK")
            return 1;                   // end of file
        else
            continue;
    }
    if(uppercase(buf.substr(0,15))=="POWER FREQUENCY") {
        // read power frequency f
        f = atof(ltrim(buf.substr(31,8)).c_str());
        continue;
    }
    if(searchdt) {
        // currently, the support of XOPT and COPT
        // is not incorporated into this program
        if(readdt) dt = atof(ltrim(buf.substr(0,8)).c_str());
        if(!getline(atpdat,buf).eof()) { // other settings are ignored
            if(buf[buf.size()-1] == '\r')
                buf = buf.erase(buf.size()-1,1);
            linenum++;
            searchdt = false;
            continue;
        }
        else {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\"";
            throw emtp_error(errstr.str());
        }
    }

    if(buf[0]=='/') {                   // beginning of cards
        if(dt == 0.0) {
            stringstream errstr;
            errstr << atpdatfn << " - Invalid data read at line "
                << linenum << " : \"" << buf << "\","
                << "simulation time-step is undefined.";
            throw emtp_error(errstr.str());
        }
        string cardspec=uppercase(buf.substr(1,6));
        while(true) {
            if(cardspec=="BRANCH") {
                // reading branches
                if(readbranchcard(atpdatfn, atpdat, buf, dt,
                    nodes, branches, shunts,
                    curout, linenum, newcs)) {
                    if(newcs=="BNDC") {
                        searchdt = true;
                        return 0;
                    }
                    else
                        return 1;
                }
                cardspec = newcs;
            }
            else if(cardspec=="SWITCH") {
                // reading switches
                if(readswitchcard(atpdatfn, atpdat, buf, dt,
                    nodes, branches, shunts,
                    curout, sw, linenum, newcs)) {
                    if(newcs=="BNDC") {
                        searchdt = true;
                        return 0;
                    }
                    else
                        return 1;
                }
                cardspec = newcs;
            }
            else if(cardspec=="OUTPUT") {
                // reading outputs
                if(readoutputcard(atpdatfn, atpdat, buf, dt,
                    nodes, branches, shunts,
                    volout, linenum, newcs)) {
                    if(newcs=="BNDC") {
                        searchdt = true;
                        return 0;
                    }
                    else
                        return 1;
                }
                cardspec = newcs;
            }
            else if(cardspec=="SOURCE") {
                // reading sources
                if(readsourcecard(atpdatfn, atpdat, buf, dt,
                    nodes, branches, shunts,
                    curout, linenum, newcs)) {
                    if(newcs=="BNDC") {
                        searchdt = true;
                        return 0;
                    }
                    else
                        return 1;
                }
                cardspec = newcs;
            }
            else {
                stringstream errstr;
                errstr << atpdatfn << " - Invalid data read at line "
                    << linenum << " : \"" << buf << "\"";
                throw emtp_error(errstr.str());
            }
        }
    }
}
return 1;                                   // end of file
}

#endif
```

E.2 emtp.cpp

```
/*  File    : emtp.cpp
 *  Abstract:
 *
 *      C++ implementation of SimuLINK S-function
 *      for Real-time EMTP
 *
 *  by Xin Nie, Power Engineering Group
 *  Dept. of Electrical and Computer Engineering
```

```
 *  University of Alberta, Edmonton, Canada
 *  email: xnie@ece.ualberta.ca
 *  May 10, 2005
 *
 *  This program is based on a C++ S-function template,
 *  which is the copyright of Mathworks Inc.
 *
 */

/*  this section of code is used to compile this program in
    Visual C++. It must be disabled when compile in MATLAB  */
/*#if (_MSC_VER <= 1200)  // including MSVC 6.0
    #pragma warning(disable:4786)
    #ifndef MATLAB_MEX_FILE
        #define MATLAB_MEX_FILE
    #endif
#endif*/

// defines max input and output port width for S-function block
#define MAXINPUTPORTWIDTH          3
#define MAXOUTPUTPORTWIDTH         3

#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include "matrix.h"
#include "emtp.h"

using namespace std;
using namespace math;

/****************************************/
/*    definition of static variables     */
/****************************************/

// simulation variables
static double dt;                            // simulation time-step size
static string atpdatfn;                      // ATP data file name
static int InputPortWidth;                   // input port width
static int OutputPortWidth;                  // output port width

// simulation objects to store network elements
static CModeNVec md_nodes;
static CModeBVec md_branches;
static CModeSVec md_shunts;
static CModeIVec md_curout;
static CModeVVec md_volout;
static CModeSWVec md_sw;
static CModeMVec md_Gmx;
static CMVec md_vnode;
static vector<bool> SFInput;
static vector<double> SFOutput;

void EMTPInitialize(CModeNVec &md_nodes, CModeBVec &md_branches,
                    CModeSVec &md_shunts, CModeIVec &md_curout,
                    CModeVVec &md_volout, CModeSWVec &md_sw,
                    CModeMVec &md_Gmx, CMVec &md_vnode,
                    const string &atpdatfn, double& f, double& dt,
                    const bool& readdt, int& InputPortWidth,
                    int& OutputPortWidth) throw(exception);

void EMTPUpdate(CModeNVec &md_nodes, CModeBVec &md_branches,
                CModeSVec &md_shunts, CModeIVec &md_curout,
                CModeVVec &md_volout, CModeSWVec &md_sw,
                CModeMVec &md_Gmx, CMVec &md_vnode,
                const string &atpdatfn, double& t, vector<bool> &SFInput,
                int& InputPortWidth, vector<double> &SFOutput,
                int& OutputPortWidth) throw(exception);

void EMTPTerminate(CModeBVec &md_branches, CModeSVec &md_shunts)
                throw(exception);

#ifdef __cplusplus
extern "C" { // use the C fcn-call standard for all functions
#endif       // defined within this scope

#define S_FUNCTION_LEVEL 2
#define S_FUNCTION_NAME  emtp

/*
 * Need to include simstruc.h for the definition of the SimStruct and
 * its associated macro definitions.
 */
#include "simstruc.h"

/*====================*
 * S-function methods *
 *====================*/

/* Function: mdlInitializeSizes ===========================================
 * Abstract:
 *    The sizes information is used by Simulink to determine the S-function
 *    block's characteristics (number of inputs, outputs, states, etc.).
 */
static void mdlInitializeSizes(SimStruct *S)
{
    /* See sfuntmpl.doc for more details on the macros below */

    int_T NInputPort = 1;
    int_T NOutputPort = 1;
    int_T NSampleTime = 1;

    ssSetNumSFcnParams(S, 2);  /* Number of expected parameters */
    if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S)) {
        /* Return if number of expected != number of actual parameters */
        return;
    }

    ssSetNumContStates(S, 0);
    ssSetNumDiscStates(S, 0);

    if (!ssSetNumInputPorts(S, NInputPort)) return;
    ssSetInputPortWidth(S, 0, MAXINPUTPORTWIDTH);

    if (!ssSetNumOutputPorts(S, NOutputPort)) return;
    ssSetOutputPortWidth(S, 0, MAXOUTPUTPORTWIDTH);
    ssSetInputPortDirectFeedThrough(S, 0, 1);

    ssSetNumSampleTimes(S, NSampleTime);
    ssSetNumRWork(S, 0);                    // reserve for real numbers
    ssSetNumIWork(S, 0);                    // reserve for integers
    ssSetNumPWork(S, 10);                   // reserve for pointers
    ssSetNumModes(S, 0);
    ssSetNumNonsampledZCs(S, 0);

    ssSetOptions(S, 0);
}


/* Function: mdlInitializeSampleTimes =====================================
 * Abstract:
 *    This function is used to specify the sample time(s) for your
 *    S-function. You must register the same number of sample times as
 *    specified in ssSetNumSampleTimes.
 */
static void mdlInitializeSampleTimes(SimStruct *S)
{
    ssSetSampleTime(S, 0, mxGetScalar(ssGetSFcnParam(S, 0)));
    ssSetOffsetTime(S, 0, 0.0);
}

// #undef MDL_START
#define MDL_START  /* Change to #undef to remove function */
#if defined(MDL_START)
  /* Function: mdlStart ===================================================
```

```
  * Abstract:
  *    This function is called once at start of model execution. If you
  *    have states that should be initialized once, this is the place
  *    to do it.
  */
 static void mdlStart(SimStruct *S)
 {
     // get simulation time-step from s-function parameters

     dt = (double) mxGetScalar(ssGetSFcnParam(S, 0));
     double f = 0.0;
     bool readdt = false;
     const mxArray *para2 = ssGetSFcnParam(S, 1);
     int len = (mxGetM(para2) * mxGetN(para2)) + 1;
     char *str = new char[len];
     if(mxGetString(para2, str, len)) {
         static const char *msg = "Parameter #2 - string is truncated.";
         ssSetErrorStatus(S, msg);
     }
     atpdatfn = trim(string(str));       // ATP data file
     delete []str;

     md_nodes.clear();
     md_branches.clear();
     md_shunts.clear();
     md_curout.clear();
     md_volout.clear();
     md_sw.clear();
     md_Gmx.clear();

     try {
         EMTPInitialize(md_nodes, md_branches, md_shunts, md_curout,
             md_volout, md_sw, md_Gmx, md_vnode, atpdatfn, f, dt, readdt,
             InputPortWidth, OutputPortWidth);
     }
     catch(exception &e) {
         // cout << e.what() << endl;
         // send the error message to Simulink
         static const char *msg = e.what();
         ssSetErrorStatus(S, msg);
     }

     // Initialize vectors for reading inputs and outputs
     SFInput.clear();
     int i;
     for(i=0;i<InputPortWidth;i++) {
         SFInput.push_back((bool)0);
     }
     SFOutput.clear();
     for(i=0;i<OutputPortWidth;i++) {
         SFOutput.push_back((double)0.0);
     }
 }
#endif /*  MDL_START */

/* Function: mdlOutputs =================================================
 * Abstract:
 *    In this function, you compute the outputs of your S-function
 *    block. Generally outputs are placed in the output vector, ssGetY(S).
 */
static void mdlOutputs(SimStruct *S, int_T tid)
{
    // get input
    InputRealPtrsType uPtrs = ssGetInputPortRealSignalPtrs(S,0);
    // get output
    real_T             *y = ssGetOutputPortRealSignal(S,0);

    // update input
    int i;
    for(i=0;i<InputPortWidth;i++) {
        SFInput.at(i) = !(dtob(*uPtrs[i]));
    }

    // update output
    for(i=0;i<OutputPortWidth;i++) {
        y[i] = SFOutput.at(i);
    }

}

//#undef MDL_UPDATE
#define MDL_UPDATE  /* Change to #undef to remove function */
#if defined(MDL_UPDATE)
    /* Function: mdlUpdate ==================================================
     * Abstract:
     *    This function is called once for every major integration time step.
     *    Discrete states are typically updated here, but this function is
     *    useful for performing any tasks that should only take place once per
     *    integration step.
     */
    static void mdlUpdate(SimStruct *S, int_T tid)
    {
        // get current simulation time and time step
        double t = ssGetT(S);

        try {
            EMTPUpdate(md_nodes, md_branches, md_shunts, md_curout, md_volout,
                md_sw, md_Gmx, md_vnode, atpdatfn, t, SFInput, InputPortWidth,
                SFOutput, OutputPortWidth);
        }
        catch(exception &e) {
            // cout << e.what() << endl;
            // send the error message to Simulink
            static const char *msg = e.what();
            ssSetErrorStatus(S, msg);
        }
    }
#endif /* MDL_UPDATE */

/* Function: mdlTerminate ================================================
 * Abstract:
 *    In this function, you should perform any actions that are necessary
 *    at the termination of a simulation.  For example, if memory was
 *    allocated in mdlStart, this is the place to free it.
 */
static void mdlTerminate(SimStruct *S)
{
    try {
        EMTPTerminate(md_branches, md_shunts);
    }
    catch(exception &e) {
        // cout << e.what() << endl;
        // send the error message to Simulink
        static const char *msg = e.what();
        ssSetErrorStatus(S, msg);
    }
}
/*======================================================*
 * See sfuntmpl.doc for the optional S-function methods *
 *======================================================*/

/*=============================*
 * Required S-function trailer *
 *=============================*/

#ifdef  MATLAB_MEX_FILE    /* Is this file being compiled as a MEX-file? */
#include "simulink.c"      /* MEX-file interface mechanism */
#else
#include "cg_sfun.h"       /* Code generation registration function */
#endif

#ifdef __cplusplus
} // end of extern "C" scope
#endif

/**************************/
```

```
/*   Main EMTP routines   */
/**************************/

void EMTPInitialize(CModeNVec &md_nodes,     // node vector for data cases
                    CModeBVec &md_branches,  // branch vector for data cases
                    CModeSVec &md_shunts,    // shunt vector for data cases
                    CModeIVec &md_curout,    // current output for data cases
                    CModeVVec &md_volout,    // voltage output for data cases
                    CModeSWVec &md_sw,       // switch vector for data cases
                    CModeMVec &md_Gmx,       // system conductance matrix
                    CMVec &md_vnode,         // system voltage vector
                    const string &atpdatfn,  // ATP data file name
                    double& f,               // power frequency
                    double& dt,              // time step size
                    const bool& readdt,      // whether read dt in data file
                    int& InputPortWidth,     // input port width
                    int& OutputPortWidth     // output port width
                    ) throw(exception) {
        ifstream atpdat(atpdatfn.c_str());
        if(!atpdat.is_open())
            throw emtp_error(atpdatfn+" - File open error.");
        bool searchdt = false;
        int linenum = 0;                     // line number counter
        int nsw = 0;                         // number of switch states
        int i, k;
        int ipw = 0, opw = 0;

        while(true) {
            CNVec nodes;
            CBVec branches;
            CSVec shunts;
            CIVec curout;
            CVVec volout;
            CSWVec sw;
            CMVec Gmx;
            nodes.clear();
            branches.clear();
            shunts.clear();
            curout.clear();
            volout.clear();
            sw.clear();
            Gmx.clear();

            double fs = 0.0;

            // read ATP data file
            int retval = readdatacase(atpdatfn, atpdat, dt, fs,
                nodes, branches, shunts, curout, volout, sw,
                searchdt, readdt, linenum);

            int Nnode = nodes.size();        // number of nodes in network

            md_nodes.push_back(nodes);
            md_branches.push_back(branches);
            md_shunts.push_back(shunts);
            md_curout.push_back(curout);
            md_volout.push_back(volout);
            md_sw.push_back(sw);

            // node voltage vector
            CMatrix vnode = CMatrix(Nnode, 1);
            // initialize to 0 to all elements
            int ii, jj;
            for(ii=0;ii<Nnode;ii++) {
                vnode(ii,0) = 0.0;
            }

            md_vnode.push_back(vnode);       // save it

            int nswt = sw.size();
            if(nsw != 0 && nswt != nsw) {
                stringstream errstr;
                errstr << atpdatfn
                    << " - number of switch states mismatch.";
                throw emtp_error(errstr.str());
            }
            else
                nsw = nswt;

            if(int(fs)!=0) {
                if(int(f)==0) {
                    f = fs;
                }
                else {
                    if(int(f)!=int(fs)) {
                        stringstream errstr;
                        errstr << atpdatfn
                            << " - power frequency mismatch.";
                        throw emtp_error(errstr.str());
                    }
                }
            }

            CMatrix Gc(Nnode, Nnode);        // conductance matrix
            for(ii=0;ii<Nnode;ii++) {        // initialize to 0
                for(jj=0;jj<Nnode;jj++) {
                    Gc(ii,jj) = 0.0;
                }
            }

            /**********************************/
            /*  filling the matrix elements  */
            /**********************************/

            // iterate all branches
            for(i=0;i<branches.size();i++) {
                CBranch &cb = branches.at(i);
                int pos1, pos2;
                pos1 = cb.pos1;
                pos2 = cb.pos2;
                double cg;
                switch(cb.cls) {
                    case TPElement:
                        cg = ((CPElement *) cb.data)->getG();
                        Gc(pos1, pos1) += cg;
                        Gc(pos2, pos2) += cg;
                        Gc(pos1, pos2) += -cg;
                        Gc(pos2, pos1) += -cg;
                        break;
                    case TAElement:
                        cg = ((CAElement *) cb.data)->getG();
                        Gc(pos1, pos1) += cg;
                        Gc(pos2, pos2) += cg;
                        Gc(pos1, pos2) += -cg;
                        Gc(pos2, pos1) += -cg;
                        break;
                    case TTLElement:
                        cg = ((CTLElement *) cb.data)->getG();
                        Gc(pos1, pos1) += cg;
                        Gc(pos2, pos2) += cg;
                        break;
                    case TSWElement:
                        break;
                    default:
                        stringstream errstr;
                        errstr << atpdatfn << " - Invalid cls at branch No. "
                            << i <<" .";
                        throw emtp_error(errstr.str());
                }
            }

            // iterate all shunts
            for(i=0;i<shunts.size();i++) {
                CShunt &cs = shunts.at(i);
                int pos;
                pos = cs.pos;
```

```
                double cg;
                switch(cs.cls) {
                    case TPElement:
                        cg = ((CPElement *) cs.data)->getG();
                        Gc(pos, pos) += cg;
                        break;
                    case TAElement:
                        cg = ((CAElement *) cs.data)->getG();
                        Gc(pos, pos) += cg;
                        break;
                    case TTLElement:
                        cg = ((CTLElement *) cs.data)->getG();
                        Gc(pos, pos) += cg;
                        break;
                    case TSWElement:
                        break;
                    default:
                        stringstream errstr;
                        errstr << atpdatfn << " - Invalid cls at branch No. "
                            << i <<" .";
                        throw emtp_error(errstr.str());
                }
            }

            // find G matrices based on different switching states
            bool *st = new bool[nsw];
            for(k=0;k<pow2(nsw);k++) {
                itoba(k, nsw, st);
                CMatrix Gnodei = CMatrix(Nnode, Nnode);
                Gnodei = Gc;

                // iterate all switches
                for(i=0;i<nsw;i++) {
                    CSWNode &cswn = sw.at(i);
                    if(cswn.type == TBranch) {
                        // TBranch, branch element
                        int pos1, pos2;
                        CBranch &cb = branches.at(cswn.index);
                        pos1 = cb.pos1;
                        pos2 = cb.pos2;
                        double cg = getSWG(st[i]);
                        Gnodei(pos1, pos1) += cg;
                        Gnodei(pos2, pos2) += cg;
                        Gnodei(pos1, pos2) += -cg;
                        Gnodei(pos2, pos1) += -cg;
                    }
                    else {
                        // TShunt, shunt element
                        int pos;
                        CShunt &cs = shunts.at(cswn.index);
                        pos = cs.pos;
                        double cg = getSWG(st[i]);
                        Gnodei(pos, pos) += cg;
                    }
                }

                Gnodei = !Gnodei;            // find inverse
                Gmx.push_back(Gnodei);       // save the matrix
            }

            md_Gmx.push_back(Gmx);           // save the matrix for data case

            delete []st;

            ipw += sw.size();
            opw += curout.size() + volout.size();

            if(retval) break;                // all data cases end
        }

        if(ipw > MAXINPUTPORTWIDTH) {
            stringstream errstr;
            errstr << atpdatfn
                << " - Input port width exceeds port width limit, "
                << "increase MAXINPUTPORTWIDTH.";
            throw emtp_error(errstr.str());
        }
        else if(opw > MAXOUTPUTPORTWIDTH) {
            stringstream errstr;
            errstr << atpdatfn
                << " - Output port width exceeds port width limit, "
                << "increase MAXOUTPUTPORTWIDTH.";
            throw emtp_error(errstr.str());
        }
        else {
            InputPortWidth = ipw;
            OutputPortWidth = opw;
        }
}

void EMTPUpdate(CModeNVec &md_nodes,        // node vector for data cases
                CModeBVec &md_branches,     // branch vector for data cases
                CModeSVec &md_shunts,       // shunt vector for data cases
                CModeIVec &md_curout,       // current output for data cases
                CModeVVec &md_volout,       // voltage output for data cases
                CModeSWVec &md_sw,          // switch vector for data cases
                CModeMVec &md_Gmx,          // system conductance matrix
                CMVec &md_vnode,            // system voltage vector
                const string &atpdatfn,     // ATP data file name
                double& t,                  // current simulation time
                vector<bool> &SFInput,      // input vector
                int& InputPortWidth,        // input port width
                vector<double> &SFOutput,   // output vector
                int& OutputPortWidth        // output port width
                ) throw(exception)
{
    for(int k=0;k<md_nodes.size();k++) {
        CNVec &nodes = md_nodes.at(k);
        CBVec &branches = md_branches.at(k);
        CSVec &shunts = md_shunts.at(k);
        CIVec &curout = md_curout.at(k);
        CVVec &volout = md_volout.at(k);
        CSWVec &sw = md_sw.at(k);
        CMVec &Gmx = md_Gmx.at(k);
        CMatrix &vnode = md_vnode.at(k);
        int i, j=0, m=0;

        /*************************/
        /*  update switch states  */
        /*************************/

        int nsw = sw.size();
        bool *st = new bool[nsw];
        int Nnode = nodes.size();

        // convert input to bool types, since the inputs are switch states
        for(i=0;i<nsw;i++) st[i] = SFInput.at(i);

        // iterate all switches
        for(i=0;i<sw.size();i++) {
            CSWNode &cswn = sw.at(i);
            if(cswn.type == TBranch) {
                CBranch &cb = branches.at(cswn.index);
                ((CSWElement *) cb.data)->update(st[i]);
            }
            else {                              // TShunt
                CShunt &cs = shunts.at(cswn.index);
                ((CSWElement *) cs.data)->update(st[i]);
            }
        }
        m += nsw;

        CMatrix Gnodei = getCurrentG(Gmx, st, nsw);

        delete []st;
```

```
        /**************************************/
        /*  filling current source vector I  */
        /**************************************/

        // construct Inode vector
        CMatrix Inode = CMatrix(Nnode, 1);
        for(int ii=0;ii<Nnode;ii++) {
            Inode(ii,0) = 0.0;
        }

        // iterate all branches
        for(i=0;i<branches.size();i++) {
            CBranch &cb = branches.at(i);
            int pos1, pos2;
            pos1 = cb.pos1;
            pos2 = cb.pos2;
            double cih;
            double vk = vnode(pos1,0), vm = vnode(pos2,0);
            double vdiff = vk - vm;
            switch(cb.cls) {
                case TPElement:
                    cih = ((CPElement *) cb.data)->getIh();
                    Inode(pos1, 0) += cih;
                    Inode(pos2, 0) += -cih;
                    break;
                case TAElement:
                    cih = ((CAElement *) cb.data)->getIeq(t);
                    Inode(pos1, 0) += cih;
                    Inode(pos2, 0) += -cih;
                    break;
                case TTLElement:
                    if(cb.type == TFDTL) {
                        ((CFdtl *) cb.data)->updateBkm();
                    }
                    cih = ((CTLElement *) cb.data)->getIhk();
                    Inode(pos1, 0) += cih;
                    cih = ((CTLElement *) cb.data)->getIhm();
                    Inode(pos2, 0) += cih;
                    break;
                case TSWElement:
                    break;
                default:
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid cls at branch No. "
                        << i <<" .";
                    throw emtp_error(errstr.str());
            }
        }

        // iterate all shunts
        for(i=0;i<shunts.size();i++) {
            CShunt &cs = shunts.at(i);
            int pos;
            pos = cs.pos;
            double cih;
            double tmp = 0.0;
            switch(cs.cls) {
                case TPElement:
                    cih = ((CPElement *) cs.data)->getIh();
                    Inode(pos, 0) += cih;
                    break;
                case TAElement:
                    cih = ((CAElement *) cs.data)->getIeq(t);
                    Inode(pos, 0) += cih;
                    break;
                case TTLElement:
                    if(cs.type == TFDTL) {
                        ((CFdtl *) cs.data)->updateBkm();
                    }
                    cih = ((CTLElement *) cs.data)->getIhk();
                    Inode(pos, 0) += cih;
                    break;
                case TSWElement:
                    break;
                default:
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid cls at branch No. "
                        << i <<" .";
                    throw emtp_error(errstr.str());
            }
        }

        /************************************/
        /*  solve the equation v=G^(-1)*I  */
        /*  to obtain node voltage vector  */
        /************************************/
        vnode = Gnodei * Inode;

        /************************************/
        /*      update histor terms of     */
        /*        all network elements     */
        /************************************/

        // iterate all branches
        for(i=0;i<branches.size();i++) {
            CBranch &cb = branches.at(i);
            int pos1, pos2;
            pos1 = cb.pos1;
            pos2 = cb.pos2;
            double vk = vnode(pos1,0), vm = vnode(pos2,0);
            double vdiff = vk - vm;
            switch(cb.cls) {
                case TPElement:
                    ((CPElement *) cb.data)->update(vdiff);
                    break;
                case TAElement:
                    break;
                case TTLElement:
                    ((CTLElement *) cb.data)->update(vk, vm);
                    break;
                case TSWElement:
                    break;
                default:
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid cls at branch No. "
                        << i <<" .";
                    throw emtp_error(errstr.str());
            }
        }

        // iterate all shunts
        for(i=0;i<shunts.size();i++) {
            CShunt &cs = shunts.at(i);
            int pos;
            pos = cs.pos;
            double tmp = 0.0;
            switch(cs.cls) {
                case TPElement:
                    ((CPElement *) cs.data)->update(vnode(pos,0));
                    break;
                case TAElement:
                    break;
                case TTLElement:
                    ((CTLElement *) cs.data)->update(vnode(pos,0), tmp);
                    break;
                case TSWElement:
                    break;
                default:
                    stringstream errstr;
                    errstr << atpdatfn << " - Invalid cls at branch No. "
                        << i <<" .";
                    throw emtp_error(errstr.str());
            }
        }

        /************************************/
```

```
        /* setup current and voltage output*/
        /************************************/

        // current outputs
        for(i=0;i<curout.size();i++) {
            CIout &cio = curout.at(i);
            if(cio.type == TBranch) {       // Branch
                int pos1, pos2;
                CBranch &cb = branches.at(cio.index);
                pos1 = cb.pos1;
                pos2 = cb.pos2;
                double vdiff = vnode(pos1,0)-vnode(pos2,0);
                switch(cb.cls) {
                    case TPElement:
                        SFOutput.at(j) =
                            ((CPElement *) cb.data)->getib();
                        break;
                    case TAElement:
                        SFOutput.at(j) =
                            ((CAElement *) cb.data)->getib(vdiff, t);
                        break;
                    case TTLElement:
                        SFOutput.at(j) =
                            ((CTLElement *) cb.data)->getik();
                        break;
                    case TSWElement:
                        SFOutput.at(j) =
                            ((CSWElement *) cb.data)->getib(vdiff);
                        break;
                    default:
                        stringstream errstr;
                        errstr << atpdatfn << " - Invalid cls at branch No. "
                            << i <<" .";
                        throw emtp_error(errstr.str());
                }
                j++;
            }
            else {                          // TShunt
                int pos;
                CShunt &cs = shunts.at(cio.index);
                pos = cs.pos;
                switch(cs.cls) {
                    case TPElement:
                        SFOutput.at(j) =
                            ((CPElement *) cs.data)->getib();
                        break;
                    case TAElement:
                        SFOutput.at(j) =
                            ((CAElement *) cs.data)->getib(vnode(pos,0), t);
                        break;
                    case TTLElement:
                        SFOutput.at(j) =
                            ((CTLElement *) cs.data)->getik();
                        break;
                    case TSWElement:
                        SFOutput.at(j) =
                            ((CSWElement *) cs.data)->getib(vnode(pos,0));
                        break;
                    default:
                        stringstream errstr;
                        errstr << atpdatfn << " - Invalid cls at branch No. "
                            << i <<" .";
                        throw emtp_error(errstr.str());
                }
                j++;
            }
        }                                   // end for (2) loop
        // voltage outputs
        for(i=0;i<volout.size();i++) {
            CVout &vio = volout.at(i);
            SFOutput.at(j) = vnode(vio.index, 0);
            j++;
        }
    }
}

void EMTPTerminate(CModeBVec &md_branches,  // branch vector for data cases
                   CModeSVec &md_shunts     // shunt vector for data cases
                   ) throw(exception)
{
    int i=0, k=0;
    for(k=0;k<md_nodes.size();k++) {
        CBVec &branches = md_branches.at(k);
        CSVec &shunts = md_shunts.at(k);

        // delete all branches
        for(i=0;i<branches.size();i++) {
            CBranch &cb = branches.at(i);
            switch(cb.type) {
                case TR:
                    delete ((CR *) cb.data);
                    break;
                case TL:
                    delete ((CL *) cb.data);
                    break;
                case TC:
                    delete ((CC *) cb.data);
                    break;
                case TLC:
                    delete ((CLC *) cb.data);
                        break;
                case TRL:
                    delete ((CRL *) cb.data);
                    break;
                case TRC:
                    delete ((CRC *) cb.data);
                    break;
                case TRLC:
                    delete ((CRLC *) cb.data);
                    break;
                case TRLCG:
                    delete ((CRLCG *) cb.data);
                    break;
                case TVsr:
                    delete ((CVsr *) cb.data);
                    break;
                case TVs:
                    delete ((CVs *) cb.data);
                    break;
                case TIs:
                    delete ((CIs *) cb.data);
                    break;
                case TSW:
                    delete ((CSwitch *) cb.data);
                    break;
                case TFDTL:
                    delete ((CFdtl *) cb.data);
                    break;
                default:;
            }
        }

        // delete all shunts
        for(i=0;i<shunts.size();i++) {
            CShunt &cs = shunts.at(i);
            switch(cs.type) {
                case TR:
                    delete ((CR *) cs.data);
                    break;
                case TL:
                    delete ((CL *) cs.data);
                    break;
                case TC:
                    delete ((CC *) cs.data);
                    break;
                case TLC:
```

```
                    delete ((CLC *) cs.data);
                    break;
                case TRL:
                    delete ((CRL *) cs.data);
                    break;
                case TRC:
                    delete ((CRC *) cs.data);
                    break;
                case TRLC:
                    delete ((CRLC *) cs.data);
                    break;
                case TRLCG:
                    delete ((CRLCG *) cs.data);
                    break;
                case TVsr:
                    delete ((CVsr *) cs.data);
                    break;
                case TVs:
                    delete ((CVs *) cs.data);
                    break;
                case TIs:
                    delete ((CIs *) cs.data);
                    break;
                case TSW:
                    delete ((CSwitch *) cs.data);
                    break;
                case TFDTL:
                    delete ((CFdtl *) cs.data);
                    break;
                default:;
            }
        }
    }
}
```