Robust Non-Matrix Based Power Flow Algorithm for Solving Integrated Transmission and Distribution Systems

Ahmad Tbaileh

Dissertation submitted to the faculty of Virginia Polytechnic Institute and State University in partial fulfillment of the requirements of the degree of

Doctor of Philosophy
In
Electrical Engineering

Robert P. Broadwater, Chair

Saifur Rahman

Virgilio A. Centeno

Binoy Ravindran

Christopher A. Beattie

October 2nd 2017
Blacksburg, VA

Keywords: Distributed Computation, Integrated System Modeling, Load Flow,

Renewable Energy Resources, Voltage Stability

Robust Non-Matrix Based Power Flow Algorithm for Solving Integrated Transmission and Distribution Systems

Ahmad Tbaileh

ABSTRACT (Academic)

This work presents an alternative approach to power system computations, Graph Trace Analysis (GTA), and applies GTA to the power flow problem. A novel power flow algorithm is presented, where GTA traces are used to implement a modified Gauss-Seidel algorithm coupled with a continuation method. GTA is derived from the Generic Programming Paradigm of computer science. It uses topology iterators to move through components in a model and perform calculations. Two advantages that GTA brings are the separation of system equations from component equations and the ability to distribute calculations across processors. The implementation of KVL and KCL in GTA is described. The GTA based power flow algorithm is shown to solve IEEE standard transmission models, IEEE standard distribution models, and integrated transmission and distribution models (hybrid models) constructed from modifying IEEE standard models. The GTA power flow is shown to solve a set of robustness testing circuits, and solutions are compared with other power flow algorithms. This comparison illustrates convergence characteristics of different power flow algorithms in the presence of voltage stability concerns. It is also demonstrated that the GTA power flow solves integrated transmission and distribution system models. Advantages that GTA power flow bring are the ability to solve realistic, complex circuit models that pose problems to many traditional algorithms; the ability to solve circuits that are operating far from nominal conditions; and the ability to solve transmission and distribution networks together in the same model.

Robust Non-Matrix Based Power Flow Algorithm
for Solving Integrated Transmission and Distribution Systems

Ahmad Tbaileh

ABSTRACT (General Audience)

Power system engineers rely on modeling and analysis of the electric grid to ensure reliable delivery of that electricity to consumers. The algorithms currently being used for this purpose are designed to simulate either the high voltage transmission networks, or the low voltage distribution networks. The rapid growth solar photovoltaics (PV) based distributed generation over the last few years, which is expected to continue in the near future, has demanded a change in this modeling and analysis approach. As the penetration levels of distributed generation increase, accurate analysis of such an electric grid cannot be performed if either the distribution or the transmission network topology is neglected in the models. Integrated transmission and distribution system modeling and simulation, where transmission and distribution networks are modeled as one single unit, has become an important research area in recent years.

This work contributes to this research area by presenting an algorithm that can be used to solve (find operating point) of integrated transmission and distribution system models. An algorithm that can find a solution of challenging network topologies and operating under severe conditions is also presented. Finally, an application of the algorithm is discussed where the impact of solar PV-based distributed generation on voltage stability limits of the electric grid is studied by using an integrated transmission and distribution system model. The dissertation shows that by solving integrated transmission and distribution system models using this algorithm, insights about the impact of solar PV-based distributed generation on the stability limits of the electric grid can be obtained, which the transmission only or distribution only models cannot provide.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Dr. Robert Broadwater, whose continuous support, guidance, and patience made this research work possible for me. He taught me to become an independent researcher while always questioning the applicability of proposed solutions in real world problems. His discipline, hard work, and deep knowledge set a good example for me to learn from. He will be always my role model not just as a great professor and advisor, but also as an outstanding person in academic, social, and personal life. I will always be grateful to him for giving me wise and precious advice in academic and non-academic matters.

I would like to thank Dr. Saifur Rahman, Dr. Virgilio Centeno, Dr. Binoy Ravindran, and Dr. Christopher Beattie who served as members of my committee. Thank you for your support, and great comments that helped me, improve my work from different perspectives.

I would like to thank all my friends at Virginia Tech, who made my Ph.D. life fun and enjoyable. I would like to thank Ibrahima, who has been like a close brother to me, for his selfless friendship, as well as teaching me to be strong. I would like also to thank my siblings for their continuous support and motivation.

I would like to thank my parents, Kafa Tbaileh and Anan Tbaileh. I would not have been able to do all of this without their unlimited support, encouragement, friendship, and love.

Ahmad Tbaileh
Virginia Tech
October 2017

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

CE – Component Equations

GS – Gauss-Seidel

GTA – Graph Trace Analysis

KCL – Kirchhoff Current Law

KVL – Kirchhoff Voltage Law

CHAPTER 1 – INTRODUCTION

1.1 PROBLEM STATEMENT

In power systems it is common to use different algorithms to solve the power flow problem for transmission and distribution systems. Algorithms used for solving transmission networks apply matrix based algorithms [1, 2]. These algorithms work well for meshed networks where line impedances have small R/X ratios. However, while convergence rates are excellent for these algorithms under normal operating conditions, they require a Jacobian matrix to be factorized at every iteration. This operation is computationally intensive, and can require large amounts of computational time for larger networks. In addition, they have poor convergence when the system is heavily loaded, and with systems that have large R/X ratios, such as distribution systems. Furthermore, convergence may not occur if initial conditions for the power flow algorithm are not sufficiently close to the solution, which can often be the case for extreme operating conditions. It is also common for these algorithms to restrict the size of a system that may be solved, with representative restrictions ranging from 15,000 to 100,000 nodes.

For distribution networks, transmission network algorithms show slow to no convergence [3, 4]. This is due to the radial, or lightly meshed, nature of the network, and large R/X ratios. Distribution networks are unbalanced in many ways: number of phases, unbalanced construction or line impedances, unbalanced loading, and unbalanced distributed generation. Power flow algorithms used for distribution networks are different from transmission system algorithms. The forward/backward sweep method for distribution networks has been used by many distribution power flow algorithms [5].

With the growth of generation resources at the distribution level, power flow solutions of hybrid, or integrated transmission and distribution models, are needed [6, 7], including the ability to solve hybrid models that include detailed substation models and even downtown network models.

Another matter of concern is the ability of the power flow algorithm to provide a solution to the system under extreme conditions. In recent years the increase in peak load demand and increase in power transfers among utilities has elevated concerns about system voltage security. A particular difficulty encountered is when the Jacobian of the Newton-Raphson power flow becomes singular at the steady-state voltage stability limit [8]. Thus, Newton-Raphson based power flow solutions near the steady-state voltage stability limit are prone to divergence and/or error.

1.2 RELATED WORK

Because of the nonlinear equations in the power flow problem, all of the algorithms are iterative, where most algorithms converge in a few iterations [1]. One of the most common power flow algorithms is the Fast Decoupled Power Flow [2]. Some algorithms employ sparse matrix techniques [9], which reduces the number of floating point operations, and thereby the computation time. Further reduction in computation time may be achieved via utilizing sparse matrices in parallel computing [10]. One approach [11] uses parallel processing employing a Jacobian matrix for the radial distribution systems. Still other methods use distributed computation for solving transmission systems [12].

In attempts to overcome the numerical instability in some algorithms, some methods have tried using double precision computation and anti-divergence algorithms [4]. Other work suggests modifications to the Newton-Raphson and Fast Decoupled power flow algorithms [3, 13]. However, these approaches suffer from limitations.

Another numerical approach to solving the power flow is the continuation method [14]. With this method the singularity of the Jacobian can be avoided by reformulating the power flow equations and applying a locally parameterized continuation technique. Using this method the reformulated set of equations remains well conditioned and divergence problems are not encountered.

Based on the sweep method, power flow solution methods have been developed for lightly meshed networks [15, 16]. With these algorithms the system is fragmented into radial networks and solved using sweep-based methods. A compensation technique is then used to solve the mesh equations. However, these algorithms become inefficient for heavily meshed systems. Improvements have been developed for this method by constructing a sensitivity matrix that saves time in execution [16, 17].

1.3 CONTRIBUTIONS

In this work an alternative approach for power systems computations, **G**raph **T**race **A**nalysis (GTA), is described and used to solve the power flow problem. GTA and topology iterators have been used for over 20 years to develop many different algorithms [18]. GTA based power flow algorithms have been implemented that solve transmission, radial distribution, lightly meshed distribution, and heavily meshed distribution, all in the same model [19]. It has been demonstrated that GTA algorithms can solve multi-domain models, and that the same algorithm can be used to

analyze across different engineering domains [20]. These abilities derive from the generic programming architecture that GTA is based on, and the object-oriented approach used in existing GTA algorithms, where system equations can have no knowledge of the internal behavior of the components.

This work provides a description of GTA topology iterators and graph traces, including the implementation of Kirchhoff voltage (**KVL**) and current (**KCL**) laws. A new power flow algorithm, based on Graph Trace Analysis (GTA) [21], is presented.

Here, a modified Gauss-Seidel (GS) algorithm is coupled with a continuation method based on line impedances and is implemented using GTA, referred to hereafter as the GS-GTA power flow algorithm. The GS-GTA algorithm does not employ any matrices, which allows for distributing the model and the algorithm calculations across multiple processors without the need for a master processor or parallel processing computer equipment [22]. It is shown that the GS-GTA algorithm's computational complexity is approximately linear with respect to the number of nodes in the system.

It is demonstrated that the GS-GTA power flow can solve a set of circuits, referred to as robustness testing circuits, which pose problems for many traditional power flow algorithms. It is also demonstrated that the GS-GTA power flow can solve transmission and distribution topologies in the same model.

1.4 OUTLINE

This dissertation is organized as follows. Chapter II provides a review of the most popular techniques for load flow computation. In chapter III, a review for GTA is presented. The concept of topology iterators is introduced and the application of GTA to circuit analysis is discussed. Chapter IV presents the GS-GTA algorithm with a continuation method, including a simple example of the GTA based power flow algorithm. Case studies are introduced in chapter V, including a set of robustness testing circuits. GTA power flow solutions for the robustness testing circuits, IEEE standard transmission models, IEEE standard distribution models, and models that include both transmission and distribution, are compared with other commonly used power flow solvers. Finally, chapter VI reviews contributions and proposes future work.

CHAPTER 2 – REVIEW OF LOAD FLOW METHODS

2.1 INTRODUCTION

The load flow, or power flow problem, is required to find the steady-state operating point of an electric power system. Given the load demand at consumption buses and the power supplied by generators, the aim is to obtain all bus voltages and complex power flowing through all network components.

The power flow solver is the most widely used application in power systems, both in operations and planning. The power flow solver can be either a stand-alone tool or a function within more complex processes, such as stability analysis, optimization problems, or contingency analysis.

During daily operations the load flow is the basic tool for security analysis, identifying potential unacceptable voltage deviations or component overloads. Power flow is also used by planning engineers to simulate different future scenarios that may be needed for a forecasted demand.

The power flow solution works in two steps. The first and most important one is to find the complex voltage at all buses. Conventional linear circuit analysis techniques are not useful because of complex powers being specified as binding constraints, leading to a set of nonlinear equations. The second step consists of computing values for the remaining variables of interest, such as active and reactive power flows, ohmic losses, currents, and many others. In this chapter, the most popular techniques for load flow computation are explained in detail.

2.2 NETWORK MODELING

All power system components (lines, cables, and transformers) connecting different buses can be represented with a two-port π model. Given the complex voltages of the terminal buses, sending and receiving power flows are calculated, which then allows for calculating power losses.

The power flow analysis involves solving the whole network. Traditionally this solution is approached with the node or bus admittance matrix. Formation of the bus admittance matrix is considered next.

Consider a generic bus *i*, as shown in Figure 1, connected through series admittances to other buses. In addition, a small shunt admittance directly connected to the neutral or ground node may exist, representing the total charging current. The net current injected into the bus by generators or loads, $\boldsymbol{I}_i$, follows Kirchhoff's current law, which is,

$$\boldsymbol{I}_i = \sum_{j \in i} y_{ij}\left(\boldsymbol{V}_i - \boldsymbol{V}_j\right) + y_{si}\boldsymbol{V}_i \tag{1}$$

,where $\boldsymbol{V}_j$ denotes the complex voltage at bus *j*, and $j \in i$ refers to the set of buses 1, 2, . . . ,*m* directly connected to bus *i*. Reordering the terms yields

$$\boldsymbol{I}_i = \left[\sum_{j \in i} y_{ij} + y_{si}\right]\boldsymbol{V}_i - \sum_{j \in i} y_{ij}\boldsymbol{V}_i \tag{2}$$

Repeating the above derivation for the whole set of *n* buses leads to the nodal equations, which can be written in matrix form as follows:

$$\boldsymbol{I} = \boldsymbol{Y}\boldsymbol{V} \tag{3}$$

, where $\boldsymbol{Y}$ is the $n \times n$ bus admittance matrix, and the elements of the column vectors $\boldsymbol{V}$ and $\boldsymbol{I}$ represent complex node voltages and net injected currents, respectively. The elements of matrix $\boldsymbol{Y}$ can be obtained by comparing the i^{th} row of Equation (3).

$$\boldsymbol{I}_i = \sum_{i=1}^{n} y_{ij}\boldsymbol{V}_j \quad i = 1,2,\dots,n \tag{4}$$

with Equation (2) leading to

$$\boldsymbol{Y}_{ii} = \left[\sum_{j \in i} y_{ij} + y_{si}\right]; \boldsymbol{Y}_{ij} = -y_{ij} \tag{5}$$

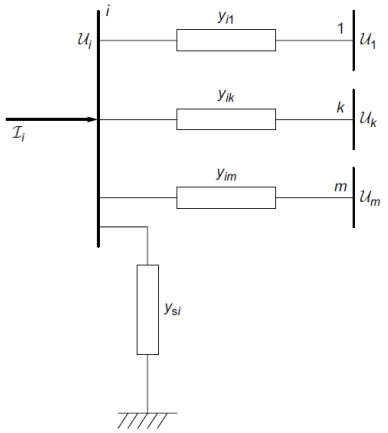


Figure 1: Elements connected to a generic bus i.

The diagonal elements of $\boldsymbol{Y}$ are calculated by adding all admittances connected to the respective bus, whereas the off-diagonal elements are simply the negative of the admittance between the

respective buses. In case there are several admittances in parallel, the equivalent admittance (that is, the sum of such admittances) should be previously computed. The great majority of off-diagonal elements will be zero because a bus is usually only directly connected to a few buses.

2.3 PROBLEM FORMULATION

In addition to the n linear equations of (4), representing the network topology and components, the following power constraint should be enforced at each bus:

$$\boldsymbol{S}_i = \boldsymbol{S}_{Gi} - \boldsymbol{S}_{Li} = \boldsymbol{V}_i \boldsymbol{I}_i^* \tag{6}$$

, where $\boldsymbol{S}_i$ is the net complex power injected to bus *i*, obtained as the difference between the complex power injected by generating elements, $\boldsymbol{S}_{Gi}$, and the complex power absorbed by loads, $\boldsymbol{S}_{Li}$. Also, $\boldsymbol{S}_{Li}$ can be used to reflect the effect of other passive components not included in matrix $\boldsymbol{Y}$. The net complex power injection applied to all buses can be written in matrix form as:

$$\boldsymbol{S} = diag(\boldsymbol{V})\boldsymbol{I}^* \tag{7}$$

, where $\boldsymbol{S}$ is the column vector consisting of bus complex powers, and $diag(\boldsymbol{V})$ represents a diagonal matrix whose elements are the elements of vector $\boldsymbol{V}$.

Given $\boldsymbol{Y}$, Equations (3) and (7) represent a system of $2n$ complex equations in terms of the $3n$ complex unknowns $\boldsymbol{S}$, $\boldsymbol{V}$, and $\boldsymbol{I}$. In theory, knowing n of such unknowns, the resulting nonlinear system can be solved to obtain the remaining $2n$ variables. However, in practice the complex nodal currents are rarely known or specified in advance. Therefore, they are usually removed from the

unknown set by substitution of Equation (3) into Equation (7). This leads to the following nonlinear set of *n* complex equations:

$$\boldsymbol{S} = diag(\boldsymbol{V})[\boldsymbol{Y}\boldsymbol{V}]^* \tag{8}$$

Expressing the complex power in terms of active and reactive powers, $\boldsymbol{S} = P + jQ$, and using rectangular coordinates for the elements of the admittance matrix, $\boldsymbol{Y} = G + jB$, the above expression can be written as

$$P + jQ = diag(\boldsymbol{V})[G - jB]\boldsymbol{V}^* \tag{9}$$

$$P_i + jQ_i = \boldsymbol{V}_i \sum_{j=1}^{n} (G_{ij} - jB_{ij})\boldsymbol{V}_j^* \qquad i = 1, 2, \ldots, n \tag{10}$$

The methods used to solve the load flow problem cannot directly work with the above equations, because the conjugate operator "$*$" prevents the application of derivatives in complex form. For this reason, it is common to split them into $2n$ real equations. Usually, complex voltages are expressed in polar form, $\boldsymbol{V} = V\angle\theta$, leading to

$$P_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} cos\theta_{ij} - B_{ij} sin\theta_{ij} \right) \qquad i = 1, 2, \ldots, n \tag{11}$$

$$Q_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij} \right) \qquad i = 1, 2, \ldots, n \tag{12}$$

Each node provides two equations and four unknowns, which means that two variables per node should be specified to solve the resulting load flow equations. Based on which variables are specified, buses can be categorized into two main types:

- *Load or PQ buses*: Both active and reactive power absorbed by the sum of loads connected at the bus are specified. Assuming the power generated locally is zero ($P_{Gi} = Q_{Gi} = 0$), this leads to the following bus constraints:

  $$P_i = P_i^{sp} = -P_{Li}^{sp}; Q_i = Q_i^{sp} = -Q_{Li}^{sp} \tag{13}$$

  leaving the voltage components, V_i and θ_i, as unknowns.

- *Generation or PV buses*: These are buses where the voltage regulator of a local generator keeps the voltage magnitude to a specified value (V_i^{sp}). Furthermore, the active power injected by the generator is usually set based on cost (economic dispatch). Taking into account the possible load demand, the resulting constraints are therefore

  $$P_i = P_i^{sp} = -P_{Gi}^{sp} - P_{Li}^{sp}; V_i = V_i^{sp} \tag{14}$$

  , leaving Q_i and θ_i as unknowns.

If only the above two types of buses are considered, all injected active powers should be specified in advance, which requires that ohmic losses be known in advance. However, power losses depend

on the resulting power flows and cannot be accurately determined until the load flow itself is solved. Therefore, the active power of at least one generator should be left as an unknown. Fortunately, when performing steady-state AC analysis, the voltage phase angle of an arbitrary bus needs to be set to zero. This represents the phase angle reference for the remaining sinusoidal waveforms. For convenience, the voltage phasor of the generating bus whose active power remains unspecified is taken as reference for phase angles. This particular PV bus, known as the *slack* or *swing bus*, is usually chosen from those generating buses with largest capacity, frequently being in charge of frequency regulation. In summary, for the slack bus the voltage magnitude and phase angle are fully specified, whereas active and reactive power components are unknowns.

For very large systems, power losses may exceed by far the capacity of certain generators. However, if an estimation of power losses is available, which is usually the case, then all generators can share a fraction of those losses. This way the slack bus will be responsible only for the power system imbalance, in addition to its own power.

Let n_L be the number of PQ buses. Then, the number of PV buses, excluding the slack bus, will be $n_G = n - n_L - 1$. It will be assumed that the first n_L buses correspond with PQ buses, followed by ordinary PV buses, and then the slack bus. Following this classification of buses, the load flow equations in polar form are

$$P_i^{sp} = V_i \sum_{j=1}^{n} V_j \left(G_{ij} cos\theta_{ij} - B_{ij} sin\theta_{ij} \right) \qquad i = 1, 2, \ldots, n_L + n_G \tag{15}$$

$$Q_i^{sp} = V_i \sum_{j=1}^{n} V_j \left(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij} \right) \qquad i = 1, 2, \ldots, n_L \tag{16}$$

Solving the load flow consists of finding the set of phase angles θ_i, $i = 1, 2, \ldots, n_L + n_G$, and the set of voltage magnitudes V_i, $i = 1, 2, \ldots, n_L$, satisfying the $2n_L + n_G$ equations of (15) and (16).

Specifying the complex voltage of the slack bus and leaving its complex power unknown simply means that its respective pair of equations will be ignored during the load flow process. Such equations will be used afterward to calculate the slack complex power.

In the same way, the n_G Equation (12) excluded from Equation (16) will provide the reactive power required by each generator to keep its voltage at the specified value. As the reactive power capability of generators is bounded, it is necessary to verify that none of the limits are violated, which complicates and slows down the solution process.

As the resulting system of equations is nonlinear, its solution necessarily involves an iterative process, for which appropriate initial values should be given to the voltages or state variables. Although finding suitable initial values may not be trivial in the general case, the so-called *flat start* is generally used for the load flow problem. It consists of setting $\theta_i^0 = 0$ for every bus and $V_i^0 = 1$ for PQ buses, reflecting the fact that voltage magnitudes normally fall within a relatively narrow band around 1 pu, while phase angle differences between adjacent buses are also small.

Once Equations (15) and (16) are solved, any other desired variable can be computed. The power flows leaving bus *i* for an element connected between buses *i* and *j* can be obtained from

$$P_{ij} = V_i V_j \left(G_{ij} cos\theta_{ij} - B_{ij} sin\theta_{ij} \right) - G_{ij} V_i^2 \quad (17)$$

$$Q_{ij} = V_i V_j \left(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij} \right) + V_i^2 (B_{ij} - b_{s,ij}) \quad (18)$$

where $b_{s,ij}$ represents the shunt susceptance associated with the π model. In case there are several components connected in parallel between buses *i* and *j*, it is necessary to use individual parameters associated to each π model, since the elements of the bus admittance matrix represent aggregated values.

Similarly, total network losses (both active and reactive) can be computed, either by adding the power injections at all buses or by adding the losses corresponding to each individual component.

2.4 SOLUTION METHODS

Because of the limited computing power and amount of memory available in primitive computers, simple methods iterating on a bus each time were adopted. The common feature of these methods is that just a single row is manipulated, instead of the entire admittance [23, 24] or impedance matrices [25-27]. Even though their practical feasibility is questionable these days, some of these methods are still used in commercial packages, and will be considered below.

2.4.1 Gauss–Seidel Method

This scheme sequentially sweeps each node, updating its complex voltage in terms of the voltages of neighboring buses. In general, finding the vector *x* satisfying the nonlinear system

$$f(x) = 0 \tag{19}$$

can be re-written as a fixed-point problem,

$$x = F(x) \tag{20}$$

, whose solution, starting from the initial value x^0, is iteratively obtained through the sequence:

$$x_i^{k+1} = F_i\left(x_1^{k+1}, \dots, x_{i-1}^{k+1}, x_i^k, \dots, x_n^k\right) \qquad i = 1,2, \dots, n \tag{21}$$

Most recent values of variables already updated ($i = 1,2, \dots, i-1$) are used when computing x_i. Relating to the load flow problem, from the many ways Equation (10) can be rewritten, the following has proven to be effective:

$$\boldsymbol{V}_i^{k+1} = \frac{1}{\boldsymbol{Y}_{ii}}\left[\frac{P_i^{sp} - jQ_i^{sp}}{(\boldsymbol{V}_i^k)^*} - \sum_{j=1}^{i-1} \boldsymbol{Y}_{ij}\boldsymbol{V}_j^{k+1} - \sum_{j=i+1}^{n} \boldsymbol{Y}_{ij}\,\boldsymbol{V}_j^k\right] \qquad i = 1,2, \dots, n-1 \tag{22}$$

The iterative process is stopped when the voltage satisfies the convergence criteria

$$\max_i \left|\boldsymbol{V}_i^{k+1} - \boldsymbol{V}_i^k\right| \leq \varepsilon \tag{23}$$

, where ε is a sufficiently small threshold (for example, 0.0001).

Although the computational cost per iteration is moderate, the convergence of this method is linear. This means that the tolerance decreases almost linearly with the number of iterations, and tends to increase as the dimension *n* of the system increases. This presents an important limitation for large

systems, as the computational burden and corresponding solution time increase considerably as larger and larger systems are solved. The number of iterations can be significantly reduced by using an acceleration factor α, as follows

$$\left[\boldsymbol{V}_i^{k+1}\right]^{acc} = \boldsymbol{V}_i^k + \alpha(\boldsymbol{V}_i^{k+1} - \boldsymbol{V}_i^k) \tag{24}$$

,whose value should not exceed 2.0 to avoid divergence. Empirically α has been determined to have optimal values between 1.4 and 1.6.

However, Equation (22) cannot be directly applied to PV buses for two reasons: (1) Q_i^{sp} is unknown for those buses and (2) the resulting voltage magnitude after each iteration will differ from the specified value. The first problem is avoided by replacing Q_i^{sp} by the value computed with the best available voltages. The second problem is avoided by scaling the estimated voltage so that the phase angle is updated, but the specified voltage magnitude is retained:

$$\left[\boldsymbol{V}_i^{k+1}\right]^{corr} = V_i^{sp}\boldsymbol{V}_i^{k+1}/V_i^k \tag{25}$$

Nowadays, the only practical application of the Gauss–Seidel method is to use it as a starter of the Newton–Raphson method, and only in the cases for which the Newton-Raphson method does not converge from the flat voltage profile start.

2.4.2 Impedance Matrix Methods

The inverse of the bus admittance matrix

$$Z = Y^{-1} \tag{26}$$

, known as the bus impedance matrix, is usually used in fault analysis. When the network is weakly connected to ground (very small shunt admittances), matrix $\boldsymbol{Y}$ is almost singular, and $\boldsymbol{Z}$ is numerically ill-defined. This problem is prevented by eliminating the slack bus and working with the resulting reduced matrix.

Let $\boldsymbol{V}_r$ and $\boldsymbol{I}_r$ be the vectors obtained by removing the slack bus variables. Then, Equation (3) can be rewritten as

$$\boldsymbol{I}_r = \boldsymbol{Y}_r\boldsymbol{V}_r + \boldsymbol{Y}_s\boldsymbol{V}_s \tag{27}$$

where $\boldsymbol{Y}_r$ is the admittance matrix obtained by removing the row and column of the slack bus, $\boldsymbol{Y}_s$ is the eliminated column, and $\boldsymbol{V}_s$ is the slack bus voltage. Reordering the terms leads to

$$\boldsymbol{V}_r = \boldsymbol{Z}_r[\boldsymbol{I}_r - \boldsymbol{Y}_s\boldsymbol{V}_s] \tag{28}$$

, where $\boldsymbol{Z}_r = \boldsymbol{Y}_r^{-1}$ is the reduced bus impedance matrix. Starting from a set of initial voltages $\boldsymbol{V}_r^0$, bus currents are obtained from

$$\boldsymbol{I}_i = (P_i^{sp} - jQ_i^{sp})/\boldsymbol{V}_i^* \qquad i = 1,2,\dots,n-1 \tag{29}$$

The bus currents resulting from (29) are substituted back into Equation (28), and the process is repeated until convergence is obtained.

This procedure allows several improvements to be implemented, all of them leading to better convergence than the Gauss–Seidel method. The price paid for the improved convergence comes from matrix $\boldsymbol{Z}_r$ being full, which means that the cost per iteration grows with n^2. If sparsity techniques [9, 28] are adopted, this burden is significantly reduced by using a triangular factorization of Y_r, instead of using its explicit inverse.

2.4.3 Newton–Raphson Method

This method successively updates unknown values through first-order approximations of the nonlinear functions. Using the first two terms in the Taylor series expansion of Equation (19) around x^k yields

$$f(x) \cong f(x^k) + F(x^k)(x^{k+1} - x^k) = 0 \tag{30}$$

, where $F = \partial f / \partial x$ is the Jacobian matrix of $f(x)$. Then, starting from the initial value x^0, corrections $\Delta x^k$are obtained by solving the linear equation system:

$$-F(x^k)\,\Delta x^k = f(x^k) \tag{31}$$

and updated values x^{k+1} from

$$x^{k+1} = x^k + \Delta x^k \tag{32}$$

The iterative process is stopped when

$$\max_i |f_i(x^k)| \leq \varepsilon \tag{33}$$

, for a sufficiently small ε. For values of x^0 close to the solution, the Newton–Raphson method converges quadratically. However, when it diverges, it does so quadratically as well. Irrespective of the network size, starting from the flat voltage profile, it takes from three to five iterations to attain convergence [29], but the number of iterations can significantly increase when adjustments are considered, such as changing transformer taps for some control purpose.

Unlike the simpler methods described before, which can be implemented in complex form, the need to carry out derivatives in the presence of the conjugate operator requires that the system of equations be separated into its real components.

The polar formulation of the Newton-Raphson equations is the most popular formulation, and is provided as follows. Vector x consists of the following $2n_L + n_G$ elements:

$$x = [\theta|V]^T = [\theta_1, \theta_2, \ldots, \theta_n - 1|V_1, V_2, \ldots, V_{nL}]^T \tag{34}$$

, and the nonlinear functions can be expressed, for every bus, as the mismatch (error) between the specified power and the power computed with the most recent *x* value. That is

$$f(x) = [\Delta P|\Delta Q]^T = [\Delta P_1, \Delta P_2, \ldots, \Delta P_n - 1|\Delta Q_1, \Delta Q_2, \ldots, \Delta Q_{nL}]^T \tag{35}$$

where

$$\Delta P_i = P_i^{sp} - V_i \sum_{j=1}^{n} V_j \left(G_{ij} cos\theta_{ij} - B_{ij} sin\theta_{ij} \right) \qquad i = 1, 2, \ldots, n_L - 1 \tag{36}$$

$$\Delta Q_i = Q_i^{sp} - V_i \sum_{j=1}^{n} V_j \left(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij} \right) \qquad i = 1, 2, \ldots, n_L \tag{37}$$

Using the above notation and dividing the Jacobian into blocks, Equation (31) applied to the load flow problem becomes [1, 30]:

$$\begin{bmatrix} H & N \\ M & L \end{bmatrix}^k \begin{bmatrix} \Delta\theta \\ \Delta V/V \end{bmatrix}^k = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix}^k \tag{38}$$

and Equation (32) becomes

$$\begin{bmatrix} \theta \\ V \end{bmatrix}^{k+1} = \begin{bmatrix} \theta \\ V \end{bmatrix}^k + \begin{bmatrix} \Delta\theta \\ \Delta V \end{bmatrix}^k \tag{39}$$

The use of $\Delta V/V$ instead of ΔV makes the Jacobian matrix more symmetrical (note that structurally the Jacobian is fully symmetric, but not numerically).

Using the derivative formula,

$$-\partial(f_i^{sp} - f_i)/\partial x_j = \partial f_i/\partial x_j \tag{40}$$

,where *f* is either *P* or *Q*, and *x* refers to *V* or θ, the Jacobian block elements are obtained according to their definitions as follows:

$$\begin{aligned} H_{ij} &= \partial P_i/\partial\theta_j; \qquad N_{ij} = V_j\, \partial P_i/\partial V_j \\ M_{ij} &= \partial Q_i/\partial\theta_j; \qquad L_{ij} = V_j\, \partial Q_i/\partial V_j \end{aligned} \tag{41}$$

The resulting expressions are as follows:

$$
\begin{aligned}
&For\ i \neq j && H_{ij} = L_{ij} = V_i V_j\left(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij}\right) \\
& && N_{ij} = -M_{ij} = V_i V_j\left(G_{ij} cos\theta_{ij} + B_{ij} sin\theta_{ij}\right) \\
&For\ i = j && H_{ii} = -Q_i - V_i^2 B_{ii} \quad N_{ii} = P_i + V_i^2 G_{ii} \\
& && M_{ii} = P_i - V_i^2 G_{ii} \quad L_{ii} = Q_i - V_i^2 B_{ii}
\end{aligned}
\tag{42}
$$

There are many common terms among the Jacobian expressions, and those used in calculating the mismatch vectors ΔP and ΔQ, which should be taken into account to save computational effort. This way, the Jacobian is nearly a by-product of the computation of the power mismatch vectors.

Solving the load flow problem using the Newton–Raphson method consists of the following steps:

1. Initialize the state vector with the flat voltage profile or with the solution of a previous case.

2. Compute $[\Delta P|\Delta Q]$ and the Jacobian elements. If all components of the mismatch vector are in absolute value lower than ε, then stop. Otherwise, continue.

3. Obtain $[\Delta\theta|\Delta V/V]$ by solving the system Equation (38).

4. Update $[\theta|V]$ using Equation (39) and go back to step 2.

For each PV bus, an equation is removed from the above system, which is one of the advantages of the polar formulation.

In many cases, convergence is improved if ΔQ is replaced by $\Delta Q/V$, which is simply achieved by dividing each row (or equation) by its correponding V_i. This way the only nonlinear term with respect to V_i in $\Delta Q_i/V_i$ is Q_i^{sp}/V_i, which is relatively small compared with the other terms.

2.4.4 Fast Decoupled Load Flow

Regardless of the effective computational techniques used in the Newton-Raphson implementation, execution times can be unacceptable for certain online applications, especially those dealing with multiple cases and very large networks. Sometimes speed of solution is as important as accuracy, which justifies the efforts devoted in the 1970s to developing fast versions of the Newton–Raphson methodology.

The first simplification is to ignore the Jacobian dependence on the current state. However, since the Jacobian is essentially a by-product of the power mismatch computation, the major computational saving that comes from the use of a constant Jacobian is from its triangular factorization. Furthermore, as the convergence rate slightly decreases, the extra iterations required are compensated by the computational savings.

The second and most important simplification comes from considering the weak coupling between active powers and voltage magnitudes, and the weak coupling between reactive power and phase angles [31, 32]. This translates into numerical values of matrices *N* and *M* in Equation (38) being significantly smaller than those of the diagonal blocks *H* and *L*. Inspecting Equation (42), this is mainly due to two reasons: (a) phase angle differences between adjacent buses are rather small, implying that $cos\theta_{ij} \approx 1$ and $sin\ \theta_{ij} \approx 0$ and (b) for high-voltage transmission networks the ratio $r/x = g/b \ll 1$. Therefore, it is expected that the performance of decoupled models is less satisfactory when solving heavily loaded systems and lower voltage levels.

Among several decoupled Newton–Raphson formulations proposed in the literature, the most successful is the *fast decoupled load flow* (FDLF), published in 1974 [2]. In addition to zeroing matrices *N* and *M*, the following simplifications are made:

1. The scaled mismatch vectors $\Delta P/V$, $\Delta Q/V$ are used instead of ΔP, ΔQ.

2. Because $Q_i$is usually lower than 1 pu and B_{ii} typically ranges between 20 and 50 pu, it is assumed that:

$cos\theta_{ij} \approx 1; G_{ij}sin\,\theta_{ij} \ll B_{ij}\ ; Q_i \ll B_{ii}V_i^2$

3. For active subproblem, voltage magnitudes are set to 1 pu, shunt reactors and capacitors are ignored in matrix *H*, including π models, and voltage regulators of transformers are ignored.

4. Regarding the reactive subproblem, phase shifting transformers are ignored in matrix *L*.

Using the previously mentioned assumptions, Equation (38) is reduced to the following decoupled systems:

$$B'\Delta\theta = \Delta P/V \tag{43}$$

$$B''\Delta V = \Delta Q/V \tag{44}$$

, where matrices B' and B'', being constant, need to be built and factorized only once. Moreover, experiments showed that ignoring line resistances in matrix B' benefits the convergence significantly. This way, the elements of the matrices B' and B'' become

$$B'_{ij} = -\frac{1}{x_{ij}}; \qquad B'_{ii} = \sum_{j\in i} 1/x_{ij}$$

$$B''_{ij} = -B_{ij}; \qquad B''_{ii} = -B_{ii}$$

, where x_{ij} is the series reactance of the element connecting buses i and j, B_{ij} is the (i, j) imaginary component of the bus admittance matrix, and $j \in i$ signifies the set of buses j adjacent to bus i.

To obtain a solution using this method, an iterative process consisting of repeatedly solving Equations (43) and (44) sequentially is used, where at each iteration the most recent values of θ and V are used, until both ΔP and ΔQ satisfy the convergence criterion. The convergence rate of the FDLF is almost the same as the coupled version during the first iterations, but it slows down as the solution is approached. In any case, the extra iterations required are well compensated since the computational cost per iteration can be between four and five times less than that of the standard Newton–Raphson method. This makes the FDLF the perfect tool in those applications involving a large number of load flow solutions (e.g. optimal power flow and contingency analysis).

2.4.5 DC Power Flow

Even though both P and Q are nonlinear functions of V and θ, some reasonable linear approximations between P and θ can be found, resulting in the so-called DC load flow. This method is based on the assumption that $V_i = 1$ at all buses. Therefore, sacrificing the ability to track reactive power flows or any other voltage related data. With this assumption, the flow of active power, given by Equation (17), is simplified to

$$P_{ij} = G_{ij}\left(cos\theta_{ij} - 1\right) + B_{ij}sin\theta_{ij} \tag{45}$$

Considering that phase angle differences between adjacent buses are small ($cos\theta_{ij} \approx 1$ and $sin\ \theta_{ij} \approx \theta_i - \theta_j$) leads to

$$P_{ij} = B_{ij}(\theta_i - \theta_j) \tag{46}$$

where B_{ij} is

$$B_{ij} = \frac{x_{ij}}{r_{ij}^2 + x_{ij}^2} = \frac{1/x_{ij}}{1+(r_{ij}/x_{ij})^2} \tag{47}$$

where r_{ij} and x_{ij} are the series resistance and reactance, respectively. For values of $r/x < 3$, which is typical in transmission networks, the error introduced when B_{ij} is replaced by $1/x_{ij}$ is lower than 1%. Therefore, Equation (46) becomes

$$P_{ij} = \frac{1}{x_{ij}}\left(\theta_i - \theta_j\right); \quad \left(\theta_i - \theta_j\right) = x_{ij}P_{ij} \tag{48}$$

Let A represent the branch-to-node incidence matrix, θ the vector of bus phase angles, both of them reduced by removing the slack row, X a diagonal matrix of branch reactances, and P_f the vector of branch active power flows. Then Equation (48) can be written in matrix form as follows:

$$A^T\theta = XP_f \tag{49}$$

$$P_f = [X^{-1}A^T]\theta \tag{50}$$

Since all branch resistances are ignored, the sum of all active powers is zero. This means that the slack bus power is a linear combination of the remaining buses. Conservation of Power can be applied to the power flows to find the net injected powers, P, using

$$P = AP_f \tag{51}$$

Finally, eliminating branch power flows by substituting Equation (50) into Equation (51), the desired linear relationship between power injections and phase angles is obtained as

$$P = [AX^{-1}A^T]\theta = B\theta \tag{52}$$

, where matrix B has the same structure (sparse and symmetric) as that of the bus admittance matrix, and its values are computed using only branch reactances.

Phase angles can be removed from the equation, leading to a linear relationship between power flows and power injections:

$$P_f = [X^{-1}A^TB^{-1}]P = S_fP \tag{53}$$

Each element of matrix S_f provides the sensitivity between the respective power flow and power injection. This expression is useful for analyzing power flow changes due to branch or generator outages. Also, the so-called power transfer distribution factors (PTDF), which are used to

determine power transfer capabilities in competitive electricity markets, can be obtained from matrix S_f.

Although the DC model is lossless, actual power losses can be estimated in terms of active power flows by using the form $R_{ij}P_{ij}^2$.

2.4.6 Radial Distribution Networks

A majority of distribution networks are designed and built so that at least an alternative path is provided in case a feeder (note, feeder is a term used to describe an electrical distribution circuit) section fails for any reason, which leads to meshed configurations. The vast majority of distribution feeders are operated in a radial configuration. This is done to reduce both the short-circuit power and the complexity of the switching and protection systems. This means that, at the distribution level, reliability is sacrificed to reduce cost.

Feeders are characterized by high r/x ratios. This results in a very poor performance of the FDLF, and the Newton–Raphson method also suffers when applied to heavily loaded feeders.

On the other hand, the configuration of the most common feeders (radial topology, a single source of power) is suitable for the development of other iterative numerical methods, which are simpler than and which also can outperform Jacobian-based algorithms. Here these methods will be referred to as sweep methods. Sweep methods are an adaptation of applying the Impedance Matrix Method to radial networks [17]. Only the single-phase version will be discussed here.

Consider a radial network with n buses, numbered from the source in such a way that each bus precedes its successors downstream. Starting with the flat start profile $\boldsymbol{V}^0$, the solution process consists of three steps, which are repeated until the complex voltages at two consecutive iterations satisfying a convergence criterion

1. Starting with the estimated voltages, calculate the net current drawn from each bus

$$\boldsymbol{I}_i^k = (\boldsymbol{S}_i^{sp}/\boldsymbol{V}_i^k)^* + y_{si}\boldsymbol{V}_i^k \qquad i = n, n-1, \dots, 2 \tag{54}$$

, where $\boldsymbol{S}_i^{sp}$ is the complex power absorbed by the local load at bus i and y_{si} is the shunt admittance connected to the bus.

2. Sweeping all tree branches in a backward direction (upstream), calculate branch currents $\boldsymbol{I}_{ij}$ by using Kirchhoff's current law

$$\boldsymbol{I}_{ij}^k = \boldsymbol{I}_j^k + \sum_{m \in i, m \neq i} \boldsymbol{I}_{jm} \qquad j = n, n-1, \dots, 2 \tag{55}$$

, where i and j are the sending and receiving buses, respectively. In practice, steps 1 and 2 can be merged as a single step.

3. Sweeping the tree in the opposite direction (downstream), bus voltages are updated from the head by considering the branch voltage drops:

$$\boldsymbol{V}_j^{k+1} = \boldsymbol{V}_i^{k+1} - z_{ij}\boldsymbol{I}_{ij}^k \qquad j = 2,3, \dots, n \tag{56}$$

where z_{ij} is the series impedance of branch $i - j$, and the complex voltage of the feeding bus, V_1, is specified (slack bus).

Because of the radial topology of the system, there is no need to explicitly build and handle any impedance or admittance matrix. A linked list is sufficient to perform the forward and backward sweeps.

A more robust version is introduced by working with branch power flows, instead of branch currents [16]. For this version, Equation (54) is replaced by

$$\boldsymbol{S}_i^k = \boldsymbol{S}_i^{sp} + y_{si}{}^* \left(\boldsymbol{V}_i^k\right)^2 \qquad i = n, n-1, \ldots, 2 \tag{57}$$

Then, the forward sweep updates voltages from

$$\boldsymbol{V}_j^{k+1} = \boldsymbol{V}_i^{k+1} - z_{ij} \left[\frac{S_{ij}^k}{\boldsymbol{V}_j^k}\right]^* \qquad j = 2,3, \ldots, n \tag{58}$$

In subsequent chapters, the performance of a number of power flow programs based on the methods described in this chapter will be compared among themselves and also with a new power flow approach based upon Graph Trace Analysis. It should be noted that upon convergence none of the methods considered here go back and check that KVL, KCL, and power balances are satisfied.

CHAPTER 3 – GRAPH TRACE ANALYSIS

In this chapter a review of Graph Trace Analysis (GTA) is provided. How topology iterators are used to implement traces is discussed. These traces are used to apply basic circuit laws to solve the power flow problem.

3.1 REVIEW OF GRAPH TRACE ANALYSIS

Graph Trace Analysis (GTA) is derived from the Generic Programming concept of Computer Science, where containers, that store objects, offer iterators to algorithms [33]. Algorithms use the iterators to process through the objects stored in the container. The Generic Programming paradigm is illustrated graphically in Figure 2.a.

With GTA, the Computer Science iterator is specialized to a *topology iterator*. The components of the physical system model are stored in a container, and the components offer topology iterators to analysis algorithms. Topology iterators are used to implement *traces* through the components, and to locate components that are physically connected together.

There is a unique edge in the graph (the graph that describes the topology of the model) associated with each component in the physical system model. Here it is assumed that the component and its associated edge have the same unique identifier. It is also assumed that the topology iterators can be used to process either through model components or through edges of the graph associated with the model. Here the terms "component" and "edge" are used interchangeability. The topology iterators of GTA are illustrated graphically in Figure 2.b.

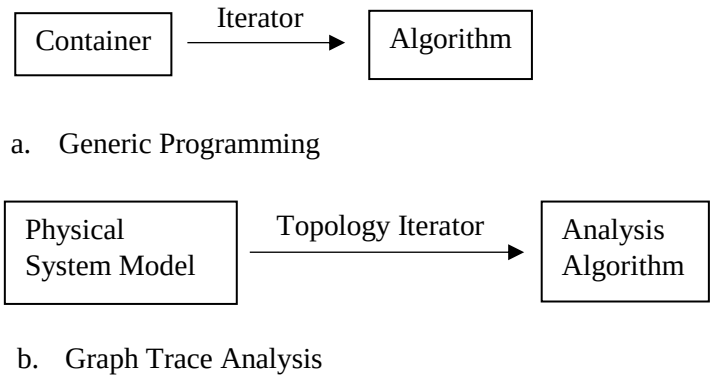


Figure 2: Graph Trace Analysis Extension of the Generic Programming Modeling Paradigm of Computer Science

In traditional graph theory both nodes and edges are used in describing a graph [34]. In GTA the graph is only considered to have edges, where the topology iterators associated with an edge contain information that may be used to locate neighboring edges. Figure 3 provides a graphical illustration of a topology iterator owned by edge A being used to locate edge B. There are two fundamental topology iterators, feeder path and cotree feeder path (or simply, cotree) iterators. A cotree identifies an edge in a graph, which, if removed, breaks an independent loop.

In traditional graph theory each edge has two associated nodes, and if the graph is directed, there is a starting node and an ending node. In GTA the start of an edge and end of an edge are defined with respect to the feeder path trace. The feeder path trace for an edge traces back to the one-and-only-one starting edge for the forward trace. The starting edge here is always associated with a component that is a source, or more exactly, a reference source. A system may have any number of reference sources, but each component will have one-and-only-one reference source.

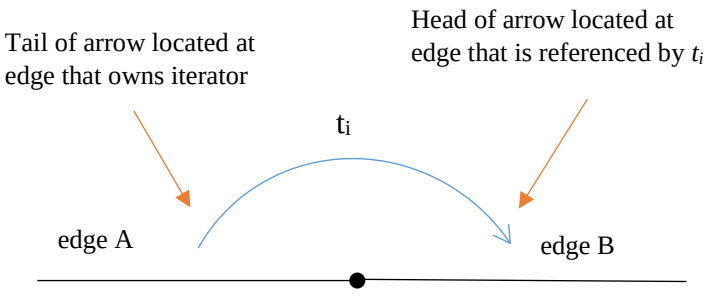


Figure 3: Topology Iterator (ti) Arrow Representation: Edge A uses topology iterator ti to retrieve edge B

Topology iterators may be used to implement KVL and KCL. GTA may be viewed as using topology iterators to trace through a graph, creating sets of components (e.g., components in an independent loop), and then applying operators to the "trace created sets," where the applied operators may result in the creation of new sets [35] .

GTA can be used to implement numerical analysis algorithms for solving systems of equations. An object-oriented feature that is implemented with GTA is the separation of system equations (e.g., KVL and KCL) from component equations. That is, system equations can only obtain information that can be measured at the connection points (or terminals) of the components, and the system equations can have no knowledge about the internal behavior of the component. It will be demonstrated in later chapters that a single GTA based power flow algorithm can solve all major topologies associated with electric power systems.

3.2 TOPOLOGY ITERATORS

In GTA, each component p has one, and only one, reference source. Each component p has a measurement reference, which is defined away from its reference source. Thus, current flow i is positive if it is flowing away from its reference source s. This is illustrated in Figure 4.

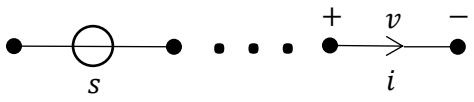


Figure 4: Voltage and Current conventions for GTA model, where S is the reference source f

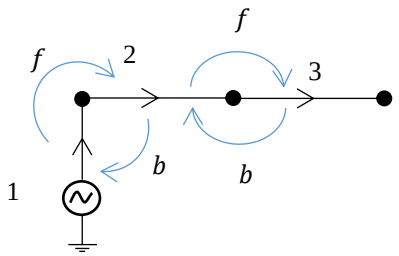
In describing the traces, each edge is identified by an integer that is unique within the model. These integer identifiers, as shown in Figure 5, are placed by the edges on the model and are used to describe the edges included in the traces. Table 1 defines key trace sets. The forward and backward traces are used to trace through every edge referenced to the same source, where each edge referenced to the source is only included in a forward or backward trace once.

Table 1: Definition of some GTA trace sets

Trace Set	Definition
FT_i	Ordered set created with recursive application of forward iterator f starting at component i; if i is not specified, then the forward trace starts at the reference source under consideration
BT_i	Ordered set created with recursive application of backward iterator b starting at component i; if i is not specified, then the backward trace starts at the ending component associated with the reference source under consideration
FP_i	Ordered set created with recursive application of feeder path iterator fp starting at component i
LT_{ct}	Ordered set created by the union of recursive application of fp, starting at cotree edge ct, with recursive application of fp, starting at adjacent edge for ct

The forward and backward traces for the circuit in Figure 5.a can be defined as $FT = \{1,2,3\}$ and $BT = \{3, 2, 1\}$, respectively. For Figure 5.b the feeder path traces for components 3, 4, and 5 are $FP_3 = \{3, 2, 1\}$, $FP_4 = \{4, 2, 1\}$, and $FP_5 = \{5, 6, 7\}$, respectively. The loop trace for the cotree element of Figure 5.b is created by the union of the feeder path traces of the components connected at the cotree, and is given by $LT_3 = \{3, 2, 1, 5, 6, 7\}$.

(a)



(b)

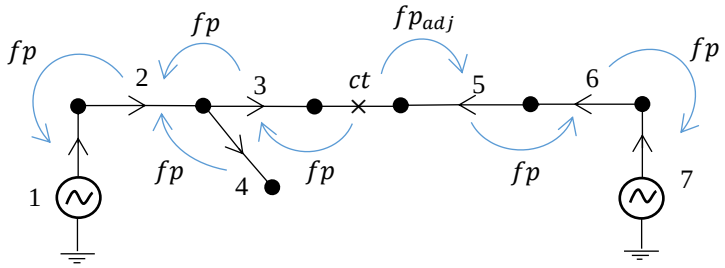


Figure 5: Basic Circuits. Part (a) shows a radial circuit, and (b) shows a looped circuit.

A topology iterator for a component p returns a component that is topologically related to component p. Four commonly used iterators are:

- $p[f]$: forward topology iterator for component p that returns the component in the forward trace direction, where [] is an overloaded operator for component p, and f is the forward topology iterator argument (for example array index or pointer)
- $p[b]$: backward topology iterator for component p that returns the component in the backward trace direction

- $p[fp]$: feeder path topology iterator notation for component p that returns the feeder path component for p, a component that is physically connected to p and which supplies power to p from the reference source for p
- $p[ct]$: cotree topology iterator notation for component p that returns the cotree component associated with p, a component that is physically connected to p and which, if removed from the graph, breaks an independent loop. Either the feeder path trace from $p[ct]$ or the feeder path trace for the adjacent component of $p[ct]$ terminates on the feeder path source for p.

There are many methods that may be used to implement topology iterators, including arrays, maps, and pointers. The four topology iterators discussed above may be considered as a quadruply linked list that contains a doubly linked list as follows:

$$quadruply\ linked\ list \begin{cases} \left.\begin{matrix} p[f] \\ p[b] \end{matrix}\right\} doubly\ linked\ list \\ p[fp] \\ p[ct] \end{cases} \tag{59}$$

Thus, $p[f]$ and $p[b]$ represent the doubly linked list of computer science. $p[fp]$ and $p[ct]$ provide physical connectivity information for p and are used in writing equations. The forward and backward traces may be used to process through all the components in the model, and when calculations are performed at a component p, the feeder path and/or cotree topology iterators are used to implement the calculations.

To obtain a source object, an array of systems with index i: $sys[i]$ is used along with an array of sources for each system with index n: $sys[i].sour[n]$. Thus a source object p may be obtained by

$$s = p = sys[i].sour[n] \tag{60}$$

where s and p are both assigned to the same source component.

Using topology iterators, algorithms can navigate through the system using forward, backward, feeder path, and cotree traces. Other traces can also be defined. It should be noted that the feeder path iterator and the cotree iterator are fundamental in that the forward and backward iterators can be derived from knowledge of the feeder path and cotree iterators.

The forward trace component for component p, $p1$, is given by

$$p1 = p[f] \tag{61}$$

The last component in the forward trace from source s, is given by

$$p_{end} = s[e] \tag{62}$$

, which provides the last component in the forward trace from source s, or the component from which a backward trace maybe be performed. Topology iterators are illustrated in Figure 6.

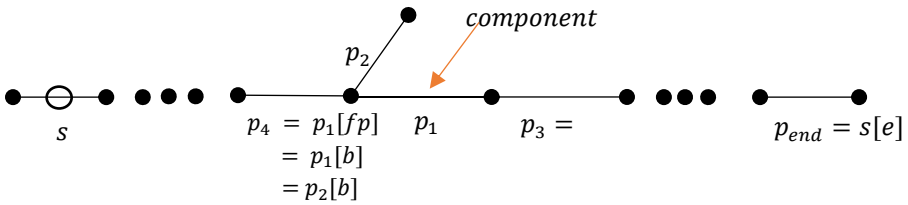


Figure 6: Illustration of Topology Iterators

Building on the previous definitions, a notation will now be introduced for employing topology iterators in algorithms, or applications. Variable $var[x]$ for component p for application APP is given by

$$APP[p].var[x] \tag{63}$$

where: $APP(p)$ is an application manipulating or calculating values for component p.

$var[x]$ is a specific variable defined for the application with argument x.

Some examples using this notation are as follows:

- $PF[p].v[3]$ = three-phase voltage array, v, for component p calculated by the power flow application PF.
- $PF[p[f]].v[3]$= three-phase voltage array, belonging to the power flow application PF, for the forward trace component of component p.

The voltage across a given component p for phase ph may be calculated using

$$PF[p[fp]].v[ph] - PF[p].v[ph] \tag{64}$$

- $RA[p].downtime$ = downtime for component p calculated by the Reliability Analysis application RA.

Reference [36] illustrates how traces can be used to develop sets for reliability analysis. The notation makes it easy for different applications to be brought together to work as a team.

3.3 CIRCUIT ANALYSIS WITH GTA

GTA syntax for basic circuit analysis is now illustrated, where to simplify the discussion notations for the application (e.g., PF or RA) and three phases are dropped. The voltage drop across component p in the circuit of Figure 7 is given by

$$v[p] = v[p[fp]] - i[p] \cdot z[p] \tag{65}$$

and component p current, as a result of implementing KCL for components, is given by

$$i[p] = i[p_1] + i[p_2] + i_L[p] + i[p[ct]] \tag{66}$$

Currents for all components in the system can be calculated with one reverse trace by

$$BT \rightarrow \{\, i[p] = 0 | i[p] \mathrel{+}= i_L[p] + i[p[ct]], i[p[fp]]] \mathrel{+}= i[p]\} \tag{67}$$

where '+=' is an operator that takes the value of the left-hand side, sums it to the right-hand side, and then re-assigns the result to the left-hand side.

'→ ' is the set element operator that for $S \rightarrow B$ applies operations in B to every element p of set S

'|' is used to define the end of the initialization section.

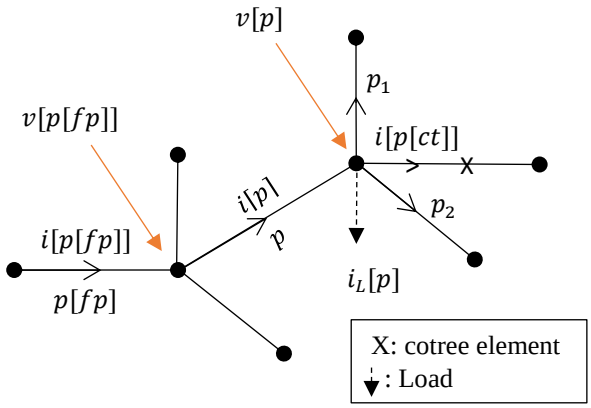


Figure 7: Implementation of KVL and KCL with GTA

KVL for cotree ct in the circuit of Figure 7 is given by

$$LT_{ct} \rightarrow \{\, v_{sum} = 0 \mid v_{sum} \mathrel{+}= v[p[fp]] - v[p] \,\} \tag{68}$$

where LT_{ct} is the set of components in the loop associated with the cotree component ct.

The backward trace set BT_S associated with source s is given by

$$BT_s = \{ p = s[e] \mid p = p(b) \} \tag{69}$$

where: $p = s[e]$ is used to initialize the trace to the last component.

$p()$ is a topology iterator for component p that is recursive and continues to return components until none exist.

GTA notation will be used to describe the power flow algorithm discussed in Chapter 4.

CHAPTER 4 – ROBUST GTA ALGORITHM

In this chapter we introduce a new GTA power flow algorithm using a modified Gauss-Seidel that is coupled with a continuation method. This is done to increase both computational speed and robustness. The GTA power flow algorithm is described using the notation introduced in Chapter 3.

4.1 GENERATING COTREE EQUATIONS WITH LOOP TRACES

The GS-GTA power flow solution presented here uses topology iterators and the GTA notation discussed previously in [37]. While tracing loops a Gauss-Seidel equation can be built and solved for the cotree current of each loop. Consider the cotree shown in Figure 8. KVL for the loop in Figure 8 is given by (70) and (71)

$$+\Delta V_x + \Delta V_{ct} - \Delta V_y = 0 \tag{70}$$

$$\Delta V_{ct} = -\Delta V_x + \Delta V_y \tag{71}$$

,where ct is a cotree edge and where $\Delta V_x, \Delta V_y, \Delta V_{ct}$ are the voltage drops across components x, y, and ct, respectively, and I_x, I_y, I_{ct} are the currents flowing through the components, respectively.

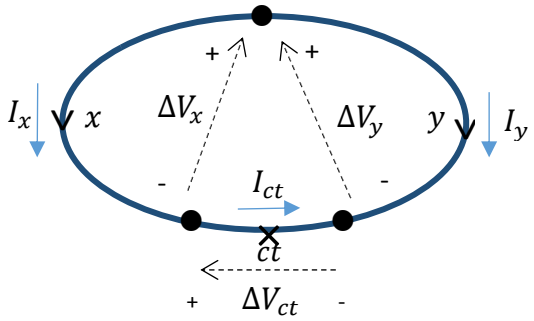


Figure 8:Voltage changes and current flows for loop consisting of three edges – x, y, and ct, where ct is the cotree edge

Assume that the cotree edge ct is associated with a zero impedance component (components that are used to create cotrees in a model, such as sectionalizing devices, in reality often have a very small impedance that is generally neglected in power flow solutions). Thus, here the voltage drop across cotree edges is assumed to be zero. Setting the cotree voltage in (71) to zero, and writing the equation in terms of impedances and currents we have

$$0 = -Z_x I_x + Z_y I_y \tag{72}$$

$$0 = -Z_x(I_x - I_{ct}) - Z_x I_{ct} + Z_y\left(I_y + I_{ct}\right) - Z_y I_{ct} \tag{73}$$

$$I_{ct}(Z_x + Z_y) = -Z_x(I_x - I_{ct}) + Z_y\left(I_y + I_{ct}\right) \tag{74}$$

Solving for the cotree current on the left hand side of (74) as the cotree current at iteration $k+1$, and taking the right hand side quantities to be at iteration k, we have an equation that can be used in a Gauss-Seidel iteration [38]

$$I_{ct}^{k+1} = \frac{-Z_x\left(I_x^k - I_{ct}^k\right) + Z_y(I_y^k + I_{ct}^k)}{Z_x + Z_y} \tag{75}$$

Equation (75) can be implemented as a loop trace is performed, and thus does not have to be explicitly written. Thus, as the power flow convergence progresses, parameter changes can occur to components in a loop, and those changes will be taken into account during the next trace iteration.

4.2 CONTINUATION OF GAUSS-SEIDEL

When circuits become heavily meshed, this means loops will have larger overlapping impedance. Poor convergence rates are associated with high overlapping loop impedances. As overlapping impedances increase, the off-diagonal entries in the impedance matrix become larger with respect to the diagonal entries. A measure of convergence of the Gauss-Seidel algorithm relies on the matrix being diagonally dominant ($\sum_{j\neq i}|a_{ij}| < |a_{ii}|$ for each row i). However, with the ratio between off-diagonals and diagonal entry for each row gets larger and closer to one, the convergence rate becomes slower.

$$\rho \leq \left\| \frac{-a_{ij}}{a_{ii}} \right\|_{\infty} = \max_{i} \frac{1}{|a_{ii}|} \sum_{j\neq i} |a_{ij}| < 1 \tag{76}$$

This can be avoided by scaling the off-diagonal entries of the Z matrix gradually from 0% to 100%. This can be viewed as solving the loops as if they are completely independent in the first iteration of the algorithm, and then gradually scaling the overlapping loop impedances as the iterations progress, until the scaling factors on the impedances become unity.

$$x_i^{k+1} = \frac{1}{a_{ii}} \left(b_i - \rho \sum_{j=1}^{i-1} a_{ij}\, x_j^{k+1} \right) \tag{77}$$

$$i = 1,2, \dots, n \; and \; \rho = 0, 0.2, \dots, 0.8, 1$$

4.3 GAUSS-SEIDEL GTA POWER FLOW ALGORITHM

The GS-GTA power flow algorithm is presented here in terms of GTA notation [37]. The algorithm employs three application objects, which are

- The power flow object [39] itself, PF, whose responsibility is to calculate voltages, currents, and power flows for every component in the system.
- An immittance object, Z, whose responsibility is to calculate impedances and admittances for every component in the system as a function of temperature, where either sequence impedances, phase impedances, or harmonic impedances may be requested. Here fundamental phase self and mutual impedances are used.
- A load object, L, whose responsibility is to calculate load values for every load bus in the system, where the load itself may have a voltage dependency and/or time dependency.

In the GS-GTA power flow algorithm it is assumed that all cotrees are implemented with switches that have zero impedance, and for the first iteration (i.e., k = 1), cotree flows are assumed (typically assumed to be zero unless a time series power flow is being performed, in which case initial conditions come from the solution of the previous iteration). The algorithm is given by

$$S \rightarrow FTs \rightarrow \{\, PF[p].v^k = PF\big[p[fp]\big].v^k - h_k\, Z[p].z \times PF[p].i^k\} \tag{78}$$

$$S \rightarrow BTs \rightarrow \{\, PF[p].i^k = 0, p = s[e] \;\big|\; PF[p].i^k \mathrel{+}= u_k L[p].{i_L}^k$$

$$+PF[p].{i_y}^k + PF\big[p[ct]\big].i^k \;, \qquad PF\big[p[fp]\big].i^k \mathrel{+}= PF[p].i^k \,\} \tag{79}$$

$$CT \rightarrow LT_{ct}^k \rightarrow \{\, z^k[ct] = 0 \;\big|\; z^k[ct] \mathrel{+}= h_k\, Z[p].z\} \tag{80}$$

$$CT \rightarrow LT_{ct}^{k} \rightarrow \{ VF^{k}[ct] = 0 \mid VF^{k}[ct] += h_k \, Z[p].z \times (PF[p].i^{k} - PF[p].i^{k}[ct]) \} \quad (81)$$

$$CT \rightarrow \left\{ i^{k+1}[ct] = \frac{VF^{k}[ct]}{z^{k}[ct]} \right\} \quad (82)$$

$$Error_{ct} = CT \rightarrow \left\{ e^{k}[ct] = \left| PF\left[ct[fp]\right].v^{k} \right| - \left| PF\left[ct[adj]\right].v^{k} \right| \right\} \quad (83)$$

$$Error_{ct} \rightarrow ForAll[e[ct] < \varepsilon_1] \quad (84)$$

$$S \rightarrow \{ \mathbb{S}_{in} = 0 | \mathbb{S}_{in} += PF[s].\mathbb{S}^{k} \} \quad (85)$$

$$S \rightarrow FTs \rightarrow \{ \mathbb{S}_{out} = 0 | \mathbb{S}_{out} += PF[p].\mathbb{S}^{k} \} \quad (86)$$

$$Error_{\mathbb{S}} = \mathbb{S}_{in} - \mathbb{S}_{out} < \varepsilon_2 \quad (87)$$

where ‘→ ’ is the set element operator, such that for $S \rightarrow B$ applies operations in B to every element p of set S

S is a set of sources, and s is an element of set S

$PF[p]$ = power flow object associated with component p

$Z[p]$ = immittance object associated with component p

$Z[p].z$= impedance matrix for component p

h_k = immittance continuation scaling factor as a function of iteration number

$L[p]$ = load estimation object associated with component p

$L[p].i_L$= voltage dependent load current vector for component p

u_k = load continuation scaling factor as a function of iteration number

k = iteration index

$PF[p].v^{k}$= phase voltage vector for component p at iteration k

$PF[p].i^{k}$= phase current vector for component p at iteration k

$PF[p].i_y{}^k$= shunt current vector for component p at iteration k

CT = set of cotrees

$z^k[ct]$= summation of impedance around a loop for cotree ct at iteration k

VF= voltage drop summation around a loop with cotree current contributions factored out

$Error_{ct}$= set of voltage errors across cotree components

$e^k[ct]$= voltage error across cotree component ct at iteration k

$\varepsilon_1, \varepsilon_2$ = convergence tolerance

$PF[p].\mathbb{S}^k$= complex power for component p at iteration k

$\mathbb{S}_{in}$= complex power produced by sources

$\mathbb{S}_{out}$= complex power consumed by components, including losses and loads.

$Error_{\mathbb{S}}$= | $\mathbb{S}_{in}$ - $\mathbb{S}_{out}$ |.

Let us now consider the GS-GTA power flow algorithm described by Equations (78) – (87). Using forward traces, the voltage drops associated with all tree edges are calculated by equation (78). This updates the voltages at the end of each component. When the continuation method [21] is used, line impedances and/or loads are scaled as a part of the solution. Using backward traces, the currents for all components are calculated using (79).

If the set of cotree components, CT, is empty, then equations (80) – (84) are skipped. If cotree components exist, then cotree currents are solved for in equations (80) – (81). Equation (80) calculates loop impedances, equation (81) calculates voltage drops around loops, where the cotree current has been factored out, and equation (84) updates the estimates for all cotree currents.

Equations (83) - (84) check to see if all voltage drops across cotree components are within the convergence criteria. Then, equations (85) – (87) check that the power balance is within the convergence criteria. The convergence criteria for both (84) and (87) have to be satisfied for the algorithm to converge.

In traditional power flow algorithms equation substitutions are used to derive a system of n equations in n unknowns, where n represents a set of independent variables (e.g., node voltages). In contrast, here the GS-GTA algorithm iterates using $2n$ variables (i.e. flows and voltages for each component). It should be noted that in the GS-GTA solution, KVL and KCL equations are explicitly solved every iteration via the forward, backward, and cotree traces.

4.4 COMPUTATIONAL COMPLEXITY

An important step in algorithm development is considering time complexity. Big O notation is the most common metric for calculating time complexity to evaluate an algorithm [40]. It describes the execution time of a task in relation to the number of steps required to complete it. Or in a more simple way, it describes how does the execution time of an algorithm increases with the increase in the size of the modeled system.

The complexity of GS-GTA algorithm has been assessed here in an empirical manner. A circuit with a certain number of nodes has been used to record the simulation time. Then duplicates of this circuit have used to predict the complexity growth of the algorithm with model size.

Figure 9 shows how the simulation time grows with the number of nodes in the model. The figure also contains two regression lines to fit the curve, linear and quadratic. The quadratic regression fits the curve to the equation $5 \cdot 10^{-11}x^2 + 2 \cdot 10^{-5}x + 0.0178$, and the linear regression fits the curve to $3 \cdot 10^{-5}x - 0.4735$. For the quadratic fit it may be noted that the coefficient of the quadratic term is six orders of magnitude smaller than the linear term. In addition, it may be noted that the linear line fits the curve well. This can also be explained in terms of the R^2 of the regression

model. While the nonlinear curve has $R^2 = 0.9999$ which is essentially 1, the linear curve has $R^2 = 0.99$ which is still close enough to be considered a good fit of the data. This is an indication that the computational complexity of the algorithm can be considered to be linear to a certain extent. The circuit used in this analysis is the same hybrid model discussed in section 5.1 later in the dissertation.

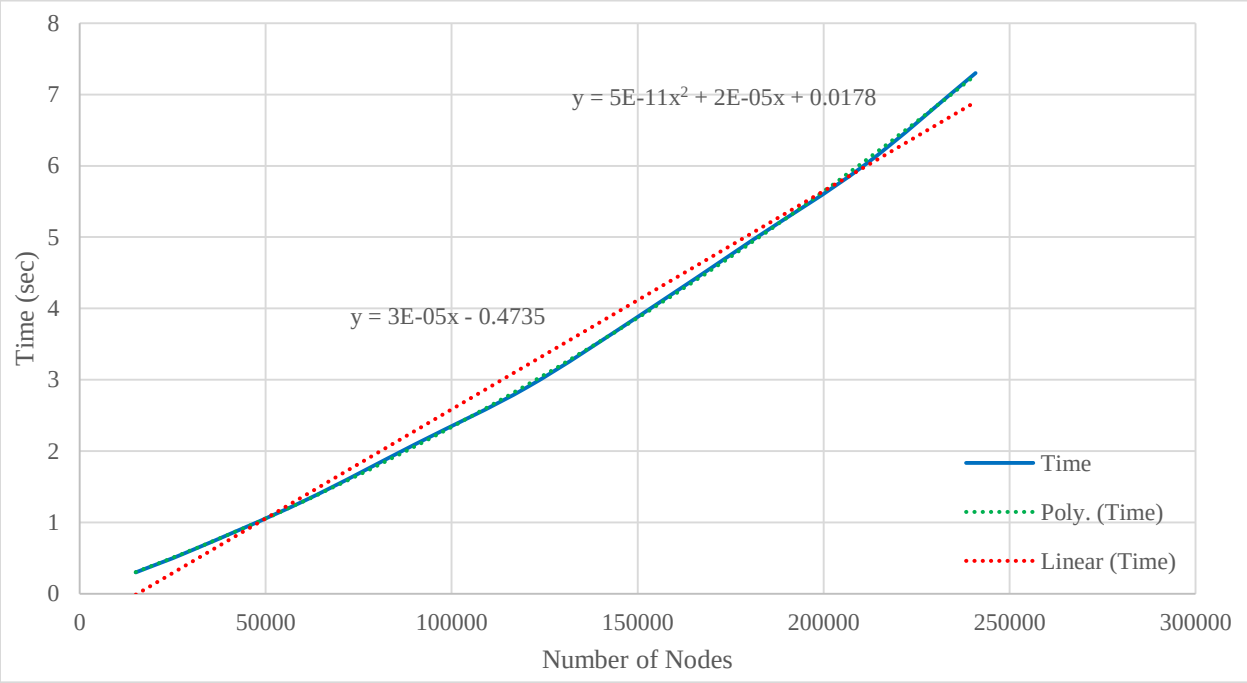


Figure 9: Computation complexity of GS-GTA algorithm as a function of number of nodes

CHAPTER 5 – CASE STUDIES

The GS-GTA based power flow described in the previous chapter will be verified against accepted solutions for both IEEE standard transmission and standard distribution networks, and will be used to solve models that include transmission, substations, and distribution, all in the same model (i.e., a hybrid model). One reason for solving hybrid models is to analyze the operation of the power system with high penetration of renewables at the distribution level.

This section is divided into four parts. First, the GS-GTA power flow solution is verified against IEEE standard models, and a set of hybrid models derived from IEEE standard models. Next, the GS-GTA power flow algorithm is used to solve a real-world integrated transmission and distribution system. Third, a set of robustness testing circuits is used to compare GS-GTA power solutions against seven other power flow solvers. Finally, the GS-GTA power flow is used to study the impact of certain additions (DSR and solar PV) on the steady-state voltage stability of the system.

5.1 IEEE STANDARD MODELS

It has been demonstrated that the GS-GTA power flow considered here can solve transmission systems, radial distribution systems, and hybrid systems [37]. Table 2 shows a list of IEEE standard transmission and distribution system models that have been used to verify the GS-GTA power flow, and a list of models created from standard IEEE models that have been solved. The table shows for each circuit the number of components, generators, solar PVs, and loops included in the model. Note that some of the modified and/or hybrid circuits have had solar PV added.

Table 2: IEEE standard models used to verify GS-GTA power flow solution along with modified and hybrid models obtained from IEEE standard models.

	Model name	Components	Generators	Solar PV	Loops/Cotrees
IEEE Standard Transmission Circuits	IEEE 14 transmission 3-ph	49	5	-	7
	IEEE 30 transmission 3-ph	89	6	-	12
	IEEE 39 transmission 3-ph	99	10	-	16
	IEEE 57 transmission 3-ph	18	7	-	24
	IEEE 118 transmission 3-ph	424	54	-	69
IEEE Standard Distribution Circuits	IEEE 37 distribution	73	1	-	-
	IEEE 13 distribution	32	1	-	-
	IEEE 34 distribution	61	1	-	-
	IEEE 123 distribution	224	1	-	-
	IEEE Composite	286	1	-	-
	IEEE 4 Grnd Ld	10	2	-	-
	IEEE 4 Open Del Ld	10	2	-	-
	IEEE 4 Ungrnd Ld	10	2	-	-
Modified and Hybrid Circuits From Standard IEEE Models	IEEE 39 transmission 3-ph unbalanced	187	10	-	8
	IEEE 39 transmission - 123 distribution with PV	22033	23	58	27
	IEEE ISM 39-14-123	23442	72	-	87
	IEEE 9 transmission – 123 distribution	3300	9	-	1

The results of the standard IEEE transmission and distribution test systems have been confirmed against published, accepted solutions. Figure 10 and Figure 11 show a comparison between the published and GS-GTA solutions for the voltage magnitudes for the *IEEE 118* bus transmission system modeled in three phases, and the *IEEE 123 distribution* system, respectively [41, 42].

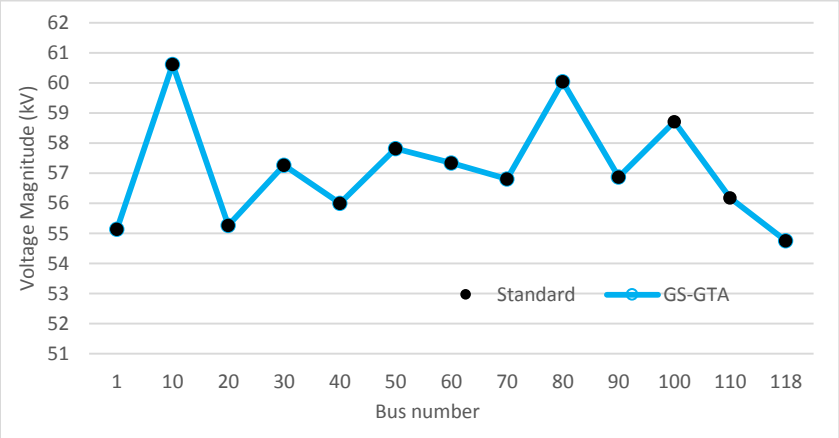


Figure 10: IEEE 118 transmission 3-ph result comparison between GS-GTA and published solution

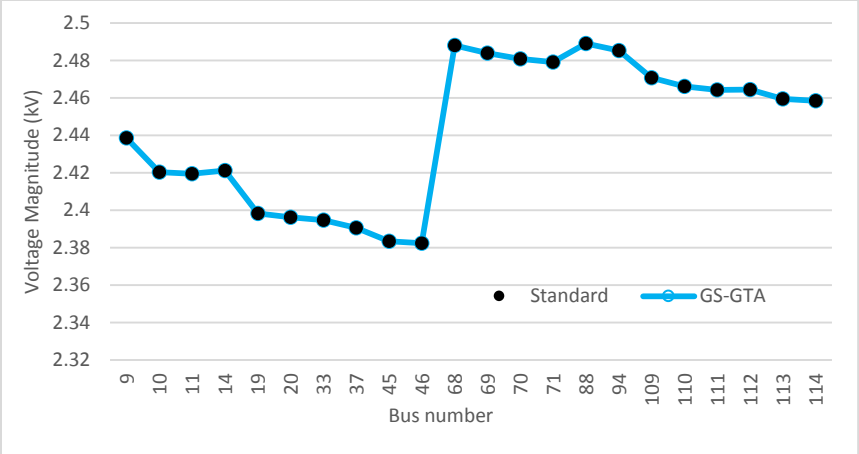


Figure 11: IEEE 123 distribution result comparison GS-GTA and published solution

Some of the hybrid models that have been solved were built by combining standard IEEE transmission and distribution system models into a single model, where substation models are used to connect the transmission system to the distribution system, and where the transmission system model has been converted to three-phase [43].

The modified and hybrid circuits shown in Table 2 were originally standard IEEE test systems that have been altered and/or combined to make more complex, larger systems. The *IEEE 39 transmission 3-ph unbalanced* model has unbalanced transmission lines [44]. The *IEEE 39 transmission–123 distribution with PV* model has the *IEEE 39 transmission 3-ph* model as the transmission circuit, where 12 of the 19 loads in the transmission system model have each been replaced with 8 distribution networks, each consisting of 12 unbalanced *IEEE 123 distribution* feeders. In addition, the model includes PV generation at the distribution level added to selected buses. More details about this model can be found in [45]. The *IEEE ISM 39-14-123* model includes the *IEEE 39 transmission 3-ph* system as the transmission network, ten *IEEE 14 transmission 3-ph* systems as sub-transmission networks, where each of the *IEEE 14 transmission*

3-ph models has ten *IEEE 123 distribution* feeders (thus there are 100 *IEEE 123 distribution* models in the *IEEE ISM 39-14-123* system).

In the *IEEE 39 transmission – 123 distribution with PV* system there are thirty-one MW's of PV generation, operating at unity power factor, that have been added to the distribution system connected to the transmission system at Bus 23. Also, 5 MW of PV generation, operating at unity power factor, have been added to the distribution system connected to Bus 12. The total PV generation represents about 5% of the total system generation. Table 3 compares the power flow results with and without the PV generation. As can be seen in Table 3, adding PVs to the distribution system at Bus 23 changed the power flows at the bus, and increased the imbalance from 12% to 35%.

Table 3 Transmission system flows before and after adding PVs to IEEE 39

	Bus 23 of the IEEE 39 Bus System			
	Before Adding PV		**After Adding PV**	
	kW	**kVAR**	**kW**	**kVAR**
Phase A	9,526	4,487	-865	4,243
Phase B	9,128	2,054	-1,446	2,062
Phase C	11,449	2,895	742	2,652

With the PVs in service, the voltage phase angles on Bus 23 advance by about 0.16 degrees on all three phases. This results in pushing power, previously flowing into Bus 23 from other sources, away from Bus 23. The addition of the PV generation at the distribution level results in flows throughout the transmission system being redistributed. As shown in Table 4, the flows on line 21-16 decreased by 7%, while the flows increased by 7% on line 19-16.

Table 4 Flows on lines 21-16 and 19-16 before and after adding PV

	Line 21-16			Line 19-16		
Phase	**w/o PV (kW)**	**with PV (kW)**	**% Change**	**w/o PV (kW)**	**with PV (kW)**	**% Change**
A	12690.2	13526.4	6.59%	16826.2	15628.9	-7.12%
B	12596.5	13449.6	6.77%	16121.2	14903.7	-7.55%
C	12979.9	13846.2	6.67%	19601.3	18369.8	-6.28%

A comparison on size of model and speed of solution between the GS-GTA and ATP [46] power flow algorithms is now provided. ATP is limited to a maximum model size of 30,000 buses, where each phase in a connection is a bus (which means a three phase terminal counts as three buses). The GS-GTA power flow algorithm is only limited by the size of computer memory available.

Power flow convergence times for the GS-GTA and ATP power flows are compared using a hybrid model constructed from standard IEEE transmission and distribution system models. The hybrid model used in the comparison consists of the WECC-9 bus transmission system [47] model supplying 21 distribution systems, each of which is created from the IEEE 123 bus standard distribution system. In this hybrid model the 21 distribution systems replace approximately 20% of the original load on the WECC-9 bus system. The model contains 3300 components, and approximately 2592 nodes. A schematic of the model in DEW is shown in Figure 12.

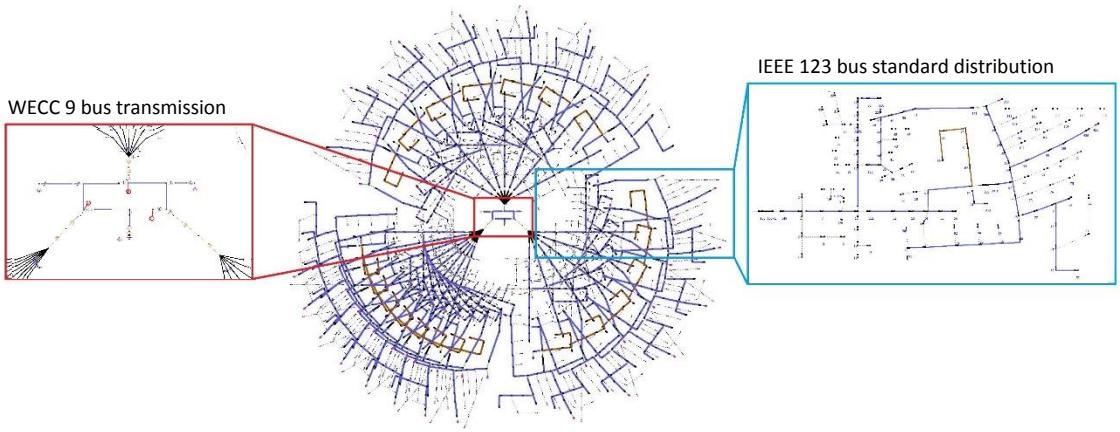


Figure 12: Hybrid model consisting of WECC-9 bus transmission system and 21- IEEE 123 bus standard distribution system models.

Solving the power flow for this hybrid model took on average 0.85 seconds using GS-GTA, compared to 2.12 seconds for ATP (excluding the time it takes ATP to compile the circuit, which on average is around 20 seconds). Thus, the GS-GTA algorithm solves approximately 2.5 (25, if compile time is included) times faster than the ATP algorithm. Both solutions agree to within 2% across all bus voltage magnitudes and power flows.

5.1.1 Voltage Stability for IEEE 39 Bus

In this study, the standard IEEE 39 bus transmission system model is used to investigate convergence of heavily loaded transmission systems. Figure 13 shows the 39-bus system, which contains 10 generators [48]. Steady-state voltage stability curve calculations are performed and voltage results are recorded for Bus 26 shown in Figure 13.

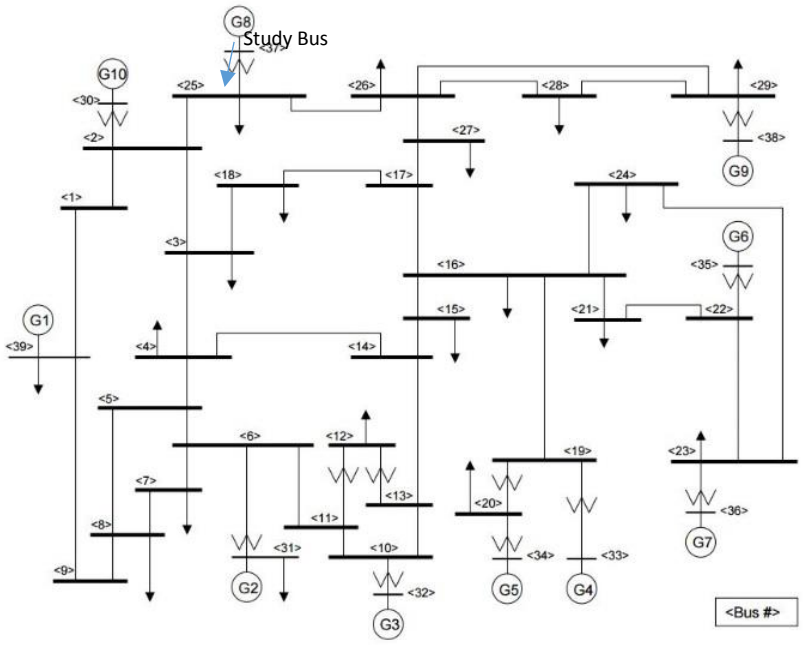


Figure 13: Standard IEEE 39 Bus Transmission System Model used for steady-state, voltage stability curve calculations

Figure 14 shows the steady-state voltage stability curve calculations from five power flow solvers - GS-GTA, NR, and Vendors 1-3. The figure shows that only GS-GTA was able to solve all loading conditions up to the tip of the nose (λ=1.7). Vendor 2, Vendor 3, NR and Vendor 1 did not converge after reaching load levels of λ=0.18, λ=0.36, λ=0.43, and λ=1.0, respectively. The value of λ here is the load scaling with respect to the nominal load. The nominal load for the 39 bus system is 6150 MW.

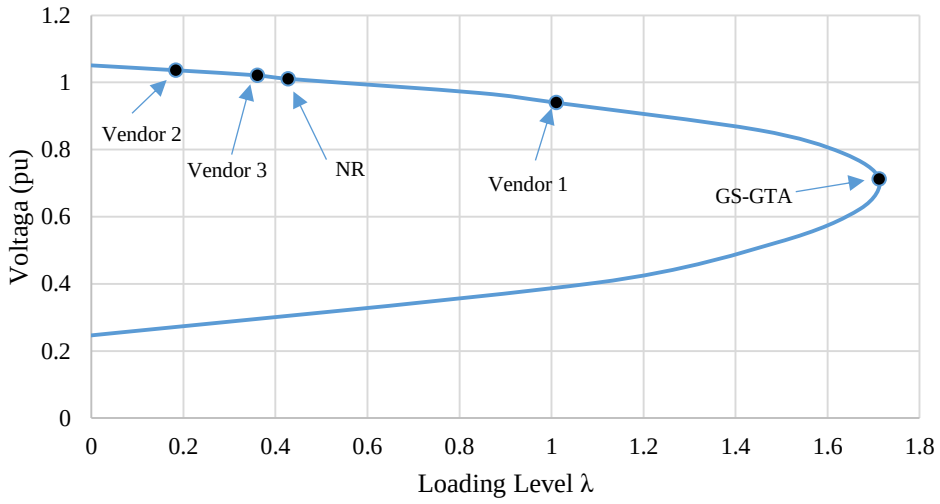


Figure 14: Voltage stability curve for IEEE 39 Bus circuit, comparing GS-GTA, NR, Vendor 1, Vendor 2 and Vendor 3

5.2 Real-World Integrated Transmission and Distribution System

This case study uses a model of real system, which includes transmission, radial distribution, lightly meshed distribution, and heavily meshed distribution networks. The model is shown in Figure 15. This model is multi-phase with 117,727 components, of which 66,132 are three-phase, 1080 are two-phase, and 50,515 single-phase. The model has 250,501 nodes, 268 feeders, 11863 load buses, and 11,957 transformers, 57 of which are LTCs, most of which are interconnected in lightly meshed distribution. It contains 6 independent loops at the transmission voltage level and 184 independent loops at the distribution voltage level, of which 115 loops occur in heavily meshed networks and 69 loops occur in lightly meshed distribution. All loads are modeled with an 8760 load model.

Figure 15: Model of real transmission and distribution networks combined in a single model, where distribution includes radial, lightly meshed, and heavily meshed distribution. Arrow and cross indicate substation location where time varying power flow plot is shown in Figure 16

Figure 16 shows a plot of the time varying power flow for a weekday in July occurring at the substation marked with a cross in Figure 15, where the power flow at the substation has a good balance. For this case study GS-GTA solutions are compared with another GTA solver [49], referred to here as DEW-GTA, power flow solutions. None of the other power flow solvers considered previously can handle transmission and distribution circuits together in the same model.

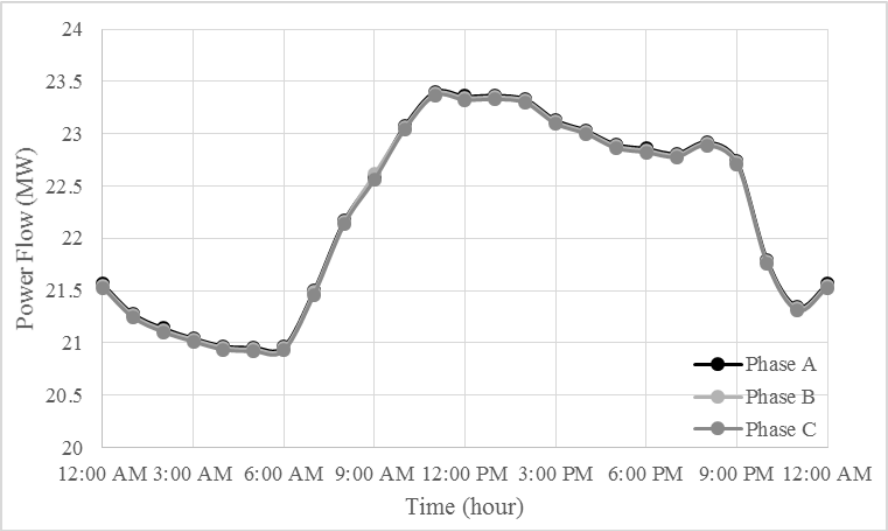


Figure 16: Time varying three-phase power flow for a weekday in July occurring at the marked substation location in Figure 15.

A special feature of the system of Figure 15 is that it contains 203 network protectors, which allow power flow in one direction only. Thus, the network topology may change during power flow iterations. When the GS-GTA algorithm detects power flow in the reverse direction for a network protector, the algorithm opens the network protector. The GS-GTA algorithm handles the changing topology of the network by updating the local topology iterators [37] at the network protectors affected.

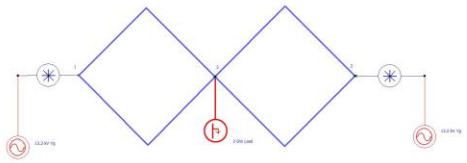
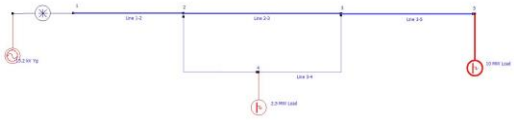
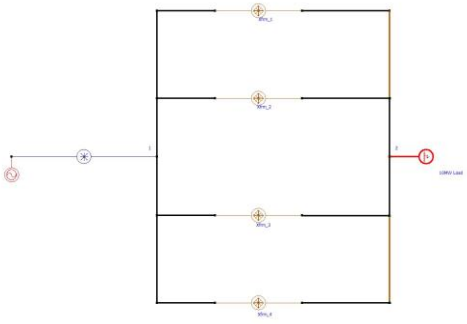
In time series power flow calculations over a 24-hour period, GS-GTA was able to solve the circuit in Figure 15 in 29.8 seconds on average, compared to 28.8 seconds on average for the sensitivity-matrix based DEW-GTA algorithm. It may be noted that the model solves within 4.5 seconds when all network protectors and controllers (e.g. capacitor banks and transformer tap changers) are locked. The large difference in solution time is an indication of how computationally expensive the process of opening and closing the network protectors is, in addition to any controller action. The initial condition on all network protectors was set to closed, and when the time series analysis finished seven network protectors were open. The solutions from the two different power flow calculations were within 0.1% of one another. Even though the DEW-GTA calculation solved slightly faster, the GS-GTA calculation supports distributed computations much better than the DEW-GTA calculation.

This is because as discussed in Chapter 4, the GS-GTA completely relies on traces to solve meshed networks. The DEW-GTA power flow shows advantages in terms of speed of convergence when the circuit does not have a large number of loops [21]. However, when the number of loops grows large, constructing and solving the sensitivity matrix involves significant computations.

5.3 ROBUSTNESS TEST CIRCUITS

The GS-GTA power flow solution is now compared with solutions from a number of widely used, primarily matrix based, power flows. In this comparison a set of ten circuits that represent real-world connections and/or extreme loading, impedance or voltage conditions are used. These circuits are referred to here as robustness testing circuits. More details about the circuits and their solutions can be found in Appendix A – Robustness Testing Circuits Data and Appendix B – Robustness Testing Circuits Solutions. Table 5 provides descriptions and schematics for the ten robustness-testing circuits. Two of the robustness testing circuits, circuits 3 and 6, were employed in [21].

Table 5: Descriptions and schematics of robustness testing circuit models used in power flow comparisons

Stiff Circuit Number	Circuit Description	Circuit Schematic
1	Heavily loaded looped system	
2	Single phase loop inside of three-phase loop	
3	Four substation transformers connected in parallel	

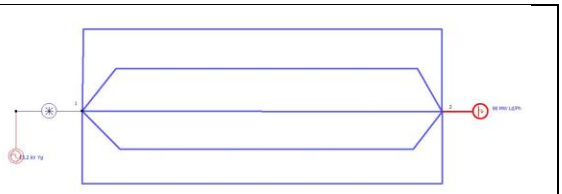
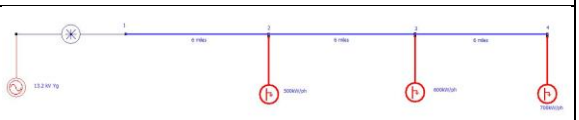
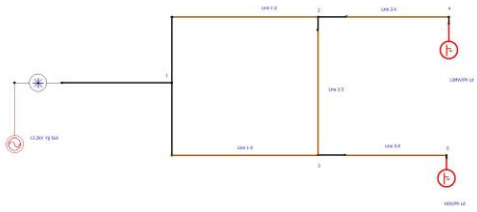
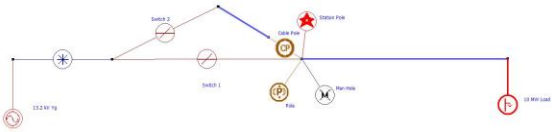
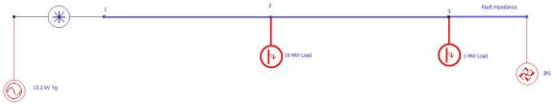
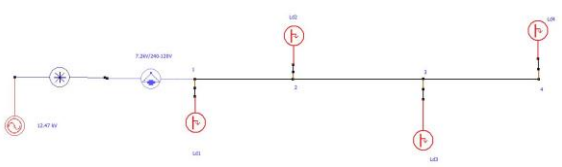
4	Low impedance loops, as might occur in a downtown network, where the cables have the same impedance	
5	Heavily loaded radial system	
6	Low impedance and high impedance lines running in parallel and serving loads differing in magnitude by a factor of 1000.	
7	Low impedance loops, as might occur in a downtown network, where the cables have different impedances	
8	Incorporating structural components – poles and manholes, such as occur in GIS systems - along with switches in parallel	
9	Three-phase voltage source with one phase set to zero volts (used to test the ability of a power flow algorithm to calculate a fault current)	
10	Three wire secondary system	

Table 6 and Table 7 show the power flow comparisons between GS-GTA and three open source power flow algorithms [46, 50, 51] and four transmission system power flow commercial products (referred to as Vendor 1 – Vendor 4 in Table 6 and Table 7), where the legend is shown at the bottom of the table.

Table 6: Robustness testing circuit model comparisons between the GS-GTA power flow and other transmission power flow algorithms

Stiff Circuit Number	GS-GTA Version 11	PowerWorld Version 18 Vendor 1	PSLF Version 16.1 Vendor 2	PSSE Version 32 Vendor 3
1	✓	××	0.5% Difference	0.3% Difference
2	✓	×	×	×
3	✓	×	✓	✓
4	✓	××	0.1% Difference	✓
5	✓	5% Difference	22% Difference	1.3% Difference
6	✓	10% Difference	0.8% Difference	12% Difference
7	✓	××	✓	✓
8	✓	×	×	×
9	✓	×	×	✓
10	✓	×	×	×
Legend	✓Converged with correct answer ×× Didn't converge		× Couldn't model % Converged with difference.	

Table 7: Robustness testing circuit model comparisons between the GS-GTA power flow and other distribution power flow algorithms

Stiff Circuit Number	GS-GTA Version 11	ATP Version 5.9	GridLAB-D Version 3.2	OpenDSS Version 7.6.5.18	CYME Version Vendor4
1	✓	✓	×	××	2% Difference
2	✓	✓	××	0.32% Difference	13.55% Difference
3	✓	✓	×	✓	✓
4	✓	✓	×	7.4% Difference	2.6% Difference
5	✓	✓	××	25% Difference	××
6	✓	✓	××	38% Difference	××
7	✓	0.17% Difference	×	15.6% Difference	××
8	✓	0.5% Difference	×	×	✓
9	✓	✓	×	××	2% Difference
10	✓	✓	××	✓	12.61% Difference
Legend	✓Converged with correct answer ×× Didn't converge		× Couldn't model % Converged with difference.		

From the tables above, all power flow solutions failed to solve all of the robustness testing circuits except for the GS-GTA and ATP power flows. When a power flow algorithm failed to solve a robustness testing circuit, it was due to one of the following three reasons:

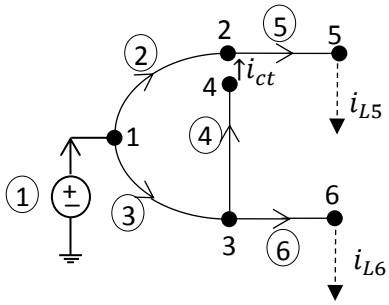
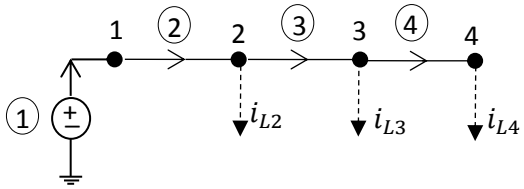
1. Not being able to model features included in the circuit, such as switches in parallel
2. Not converging to a solution
3. Converging to a solution that did not agree with the solution obtained from multiple other power flow algorithms.

The GS-GTA power flow algorithm is able to provide accurate solutions to all ten robustness-testing circuits. For every circuit at least one other algorithm is used to verify the GS-GTA algorithm solution. The ATP (Alternate Transients Program) power flow provides accurate solutions to eight of the circuits, and vendor 3 only provides accurate solutions to four of the circuits. OpenDSS and Vendor 2 can only solve two of the circuits, and GridLAB-D and Vendor 1 are not able to solve any of the circuits accurately.

Two robustness-testing circuits are used to compare the GS-GTA power flow against the Newton-Raphson (NR) power flow algorithm. Convergence properties of the algorithms are investigated along with the ability to calculate the steady state voltage stability curve.

The circuits shown in Table 8 are used to compare the convergence properties of the GS-GTA algorithm against other power flow algorithms [37]. For the first case, a Newton-Raphson algorithm programmed by the authors is used in the comparison. Programming of the Newton-Raphson algorithm was performed so that the behavior of the elements of the Jacobian matrix could be investigated. In the second case, the performance of different Newton-Raphson based power flows, including the one programmed by the authors, are compared with the GS-GTA algorithm.

Table 8: Robustness testing circuits 5 and 6

Case Number	Case Name	Circuit Model	Parameters
1	High impedance loop		Source: 13.2kV Yg Line 2: R = varies – see Table 9 Line 3: R = 1 Ohms Lines 4, 5, 6: R = 0.1 Ohms Load 5: varies – see Table 9 Load 6: 1kW/phase
2	Radial circuit		Source: 13.2kV Yg Each line impedance: R + jX = 2.5 + j6.0 Ohms Load 1: varies – see Table 10 Load 2: varies – see Table 10 Load 3: varies – see Table 10

5.3.1 Case 1: High Impedance Loop

In this case the Newton-Raphson algorithm used in the comparison was programmed by the authors, and is referred to as the NR algorithm. A parametric study is performed with the Case 1 circuit of Table 8, where the load at Bus 5 and the impedance of Line 2 are varied as shown in Table 9. Table 9 also shows the voltage magnitudes calculated at Bus 5 for both solutions, and the condition number [52] of the Jacobian matrix for the NR algorithm.

Table 9 shows that as the line impedance or loading level are increased, a point is reached where the NR algorithm does not converge to a solution. It may be noted from Table 9 that the condition number of the Jacobian increases as the line impedance or loading level increase, where increases in the condition number indicate that the matrix is becoming ill-conditioned. The GS-GTA algorithm converges for all parametric studies. In addition, Kirchhoff voltage and current laws, along with the power balance, are satisfied for all of the GS-GTA solutions.

Table 9: Case 1 voltage magnitude results at bus 5 and Jacobian condition number as a function of impedance of Line 2 and load at Bus 5

Case	Load (MW)	Impedance (ohms)	Condition Number of Jacobian	NR Solution Bus 5 Voltage Magnitude (pu)	GS-GTA Solution Bus 5 Voltage Magnitude (pu)	Difference
1.1	0.001	1	71	1	1	0
1.2	1	1	72	0.9891	0.9891	0
1.3	10	1	82	0.8776	0.8777	0
1.4	12	1	85	0.848	0.8481	0
1.5	12.22	1	88	0.8445	0.8446	0
1.6	0.001	10	130	1	1	0
1.7	1	10	132	0.9808	0.9808	0
1.8	10	10	192	0.7493	0.7493	0
1.9	12	10	265	0.6568	0.6566	0
1.10	12.22	10	476	0.6425	0.6424	0
1.11	0.001	100	142	1	1	0
1.12	1	100	145	0.9791	0.9793	0
1.13	10	100	233	0.7132	0.7131	0
1.14	12	100	581	0.5673	0.5673	0
1.15	12.22	100	NA	Did Not Converge	0.5049	NA

Let us now look at what the elements of the Jacobian matrix are doing as a function of the algorithm iteration step. It should be noted that all elements of the Jacobian show similar behavior. Therefore, a single element, the (8,8) element, was chosen for illustration. Three parameter variations are considered in this evaluation of the Jacobian matrix elements, where the Line 2 impedance is set at 100 ohms and the Bus 5 load varies from 10 to 12.22 MW, as in cases 1.13-1.15. Figure 17 shows how the value of the (8,8) element of the Jacobian varies as a function of the NR iteration step as parameterized by the load variation. At the last loading value, 12.22 MW, the NR does not converge.

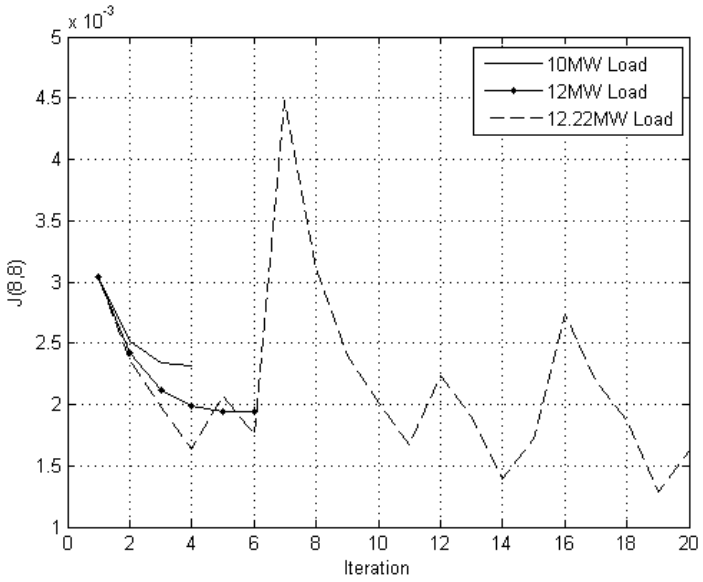


Figure 17: Jacobian (8,8) element variation with iteration for cases 1.13, 1.14, and 1.15 with loading levels of 10MW, 12MW, and 12.22MW, respectively.

Steady-state voltage stability is also used here as a measure of the robustness of power flow algorithms. The voltage stability (nose) curve for the circuit of Case 1 shown in Table 8 is shown in Figure 18, where λ is a load multiplier with respect to the base case load of 10MW. The nose curve was calculated using a continuation power flow algorithm, abbreviated here as Cont-PF, presented in [14]. The nose curve of Figure 18 was also calculated using GS-GTA, and the results

agreed with those of the Cont-PF algorithm. The NR algorithm was not able to converge beyond λ=0.925, whereas GS-GTA was able to solve to the tip of the nose curve, λ=1.11.

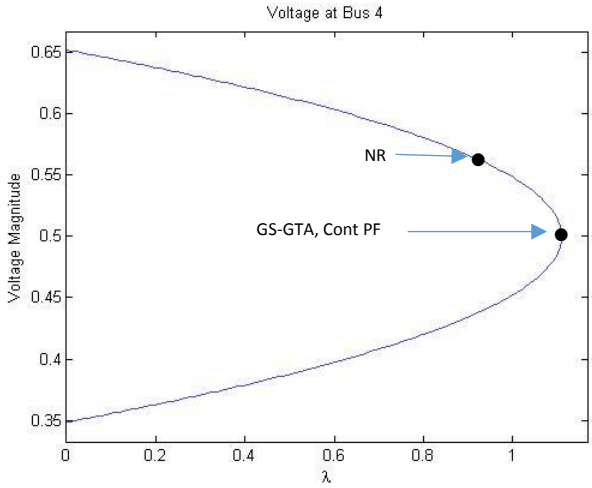


Figure 18: Voltage stability curve with 100 Ohm impedance for NR, Cont-PF, and GS-GTA algorithms, where the NR algorithm failed at λ=0.925, and GS-GTA and Cont-PF converged all the way to the curve nose where λ=1.11

5.3.2 Case 2: Radial Circuit

The Case 2 circuit in Table 8 is used to compare GS-GTA against widely used implementations of the Newton-Raphson algorithm, in addition to the NR power flow algorithm developed by the authors and considered in Case 1. The solutions are also compared to an analytical solution, where the following equations are used as the basis for the analytical solution:

$$V_2 = V_1 - I_2 Z_2 = 1.0 - I_2 Z_2 \tag{88}$$

$$V_3 = V_2 - I_3 Z_3 \tag{89}$$

$$V_4 = V_3 - I_4 Z_4 \tag{90}$$

$$I_4 = I_{L4} \tag{91}$$

$$I_3 = I_4 + I_{L3} = I_{L4} + I_{L3} \tag{92}$$

$$I_2 = I_3 + I_{L2} = I_{L4} + I_{L3} + I_{L2} \tag{93}$$

For the analytical solution, equations (91)-(93) are substituted into equations (88)-(90), and then (88)-(90) are solved for V_2, V_3, and V_4. Whether the system of equations is linear or nonlinear depends on the load type. Here a nonlinear, constant power load is used.

The analytical solution is compared in Table 10 with a number of widely used power flow solvers [50, 53-55], including DEW-GTA. Table 10 shows the results for the voltage at node 4 of the Case Study 2 circuit of Table 8. Note that two of the solutions, the solution from the ATP algorithm and the GS-GTA solution, agree exactly with the analytical solution, and thus these solutions are shown in the same column. From Table 10 it may be noted that many of the algorithms converge to a solution that does not agree with the analytical solution. Values shown in bold in Table 10 represent converged values that differ from the analytical solution by more than 1%. The largest error in a converged value occurs in subcase 2.7, where the error is 7 %.

Plots of node 4 voltage errors for each subcase of Table 10 are shown in Figure 19. When the loading conditions are low, all of the power flow solvers have little to no error. However, once the loading increases beyond the loading of case 2.4, some of the algorithm errors become significant. All power flow algorithms, except GS-GTA, ATP, and DEW-GTA had large errors, or were not able to converge to a solution, beyond case 2.5.

Table 10: Case 2 circuit node 4 per unit voltage solutions from different power flow solvers for varying loading levels. Node 4 voltage values shown in bold differ from the verified solution by more than 1%.

Sub-Case	Load Level (kW)	Analytical, ATP, and GS-GTA (pu)	DEW-GTA Version 10.74 (pu)	NR (pu)	GridLAB-D Version 3.2 (pu)	Vendor 1 (pu)	Vendor 2 (pu)	Vendor 3 (pu)
2.1	300/400/500	0.8282	0.8282	0.8282	0.8282	0.8288	0.8283	0.8282
2.2	350/450/550	0.7936	0.7938	0.7938	0.7937	0.7961	0.794	0.7937
2.3	400/500/600	0.7505	0.7505	0.7505	0.7505	0.7512	0.7505	0.7505
2.4	450/550/650	0.6893	0.6897	0.6893	0.6893	0.6912	0.6896	0.6897
2.5	460/560/660	0.6723	0.673	0.6723	0.6723	0.6746	0.6729	0.6748
2.6	470/570/670	0.6517	0.6528	0.6517	0.6517	0.6548	0.6533	**0.6594**
2.7	480/580/680	0.6239	0.625	0.6239	0.6239	**0.6306**	0.6291	**0.6436**
2.8	490/590/690	0.5808	0.5816	**Did Not Converge**	**Did Not Converge**	**0.6059**	**0.5969**	**0.6216**

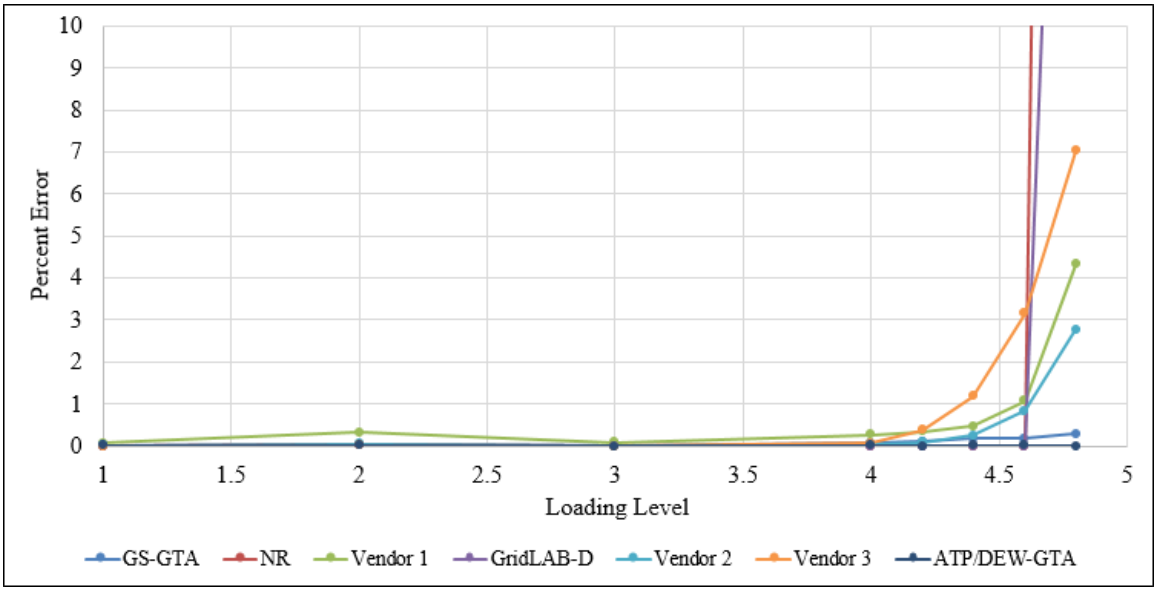


Figure 19: Percent errors for voltage at node 4 for Case Study 2 circuit as loading level is increased for different power flow solvers

It should be noted that when convergence becomes difficult, some of the power flow solvers apply approximations, such as switching constant power loads to impedance loads, in order to achieve convergence.

Figure 20 shows the voltage stability curve calculated with the Cont-PF algorithm for Case 2 circuit, where the GS-GTA algorithm obtained the same solution as the Cont-PF algorithm. Note that the value of λ in Figure 20 scales the subcase 2.1 load of Table 10. In Figure 20 the maximum loading point beyond which algorithms do not converge, or do not calculate accurate results, are shown. The Vendor 3, NR, and Vendor 2 algorithms were not able to converge beyond λ=0.795, λ=0.894, and λ=0.894, respectively. The GS-GTA was able to solve all loading conditions, including the maximum loading point at λ=0.944.

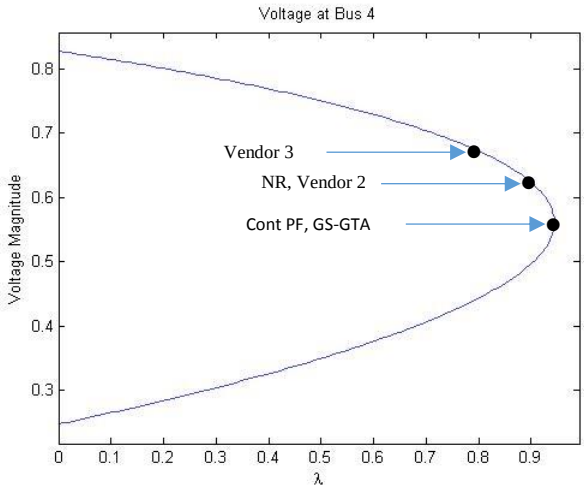


Figure 20: Voltage stability curve for Case Study 2 circuit comparing GS-GTA, Cont-PF, NR, Vendor 2, and Vendor 3

5.4 DSR AND PV IMPACT ON VOLTAGE STABILITY LIMIT

With the potential of plotting the steady-state voltage stability curve using the GS-GTA algorithm, it is possible to study the voltage stability limit on different topologies of the power system. This expands the previously known continuation of power flow algorithm [14], applied on transmission networks only, to include any integrated transmission and distribution model. Moreover, this allows us to study the impact of different additions, like DSRs and PVs, on the voltage stability limit of the system. In this section we discuss two cases: DSR application on a simple circuit, and integration of PVs at the distribution level in a hybrid model.

5.4.1 DSR Application

For this case study we will use the circuit shown in Figure 21. This circuit demonstrates a very common application for DSRs.

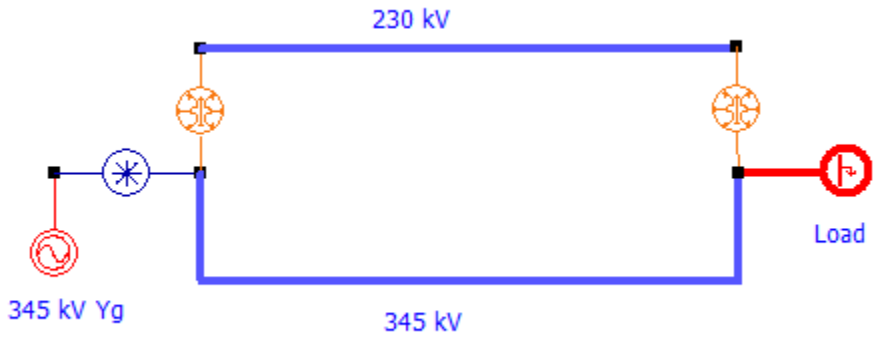


Figure 21: Circuit for DSR case study with 230kV and 345kV parallel lines

The circuit has a 345kV source, a nominal load of 400MW, and two parallel lines with different operating voltages, namely 230kV and 345kV, with thermal limits of 150 MW and 480 MW, respectively. With the existence of two parallel transmission lines, it is common for one of these

lines to reach its thermal limit with plenty of capacity on the second line. Therefore, it is desired to push some of the power through the second line, which allows more power transfer capability of the whole network. The 230 kV line reaches its thermal limit at a 465 MW load. However, the 345 kV still has some capacity. Therefore, a good solution would be to install DSRs at the 230 kV line to push more power through the 345 kV line and release the congestion in the 230 kV line. The installation of 140 DSR modules at each phase with an impedance of 1mH/module allows the circuit to transfer more power before reaching thermal limits. DSRs allow 60 MW to be transferred to the load before both lines hit their thermal limits at the same time.

It is of interest to evaluate how the addition of DSRs affects the voltage stability limit of the system. Figure 22 shows the voltage stability curves for the circuit with and without the addition of DSRs. It can be seen that the voltage stability limit of the system was reduced by almost 24MW ((1.87-1.81)*400MW = 24). However, this shift is less than half the amount of power DSRs allowed to be delivered by the system.

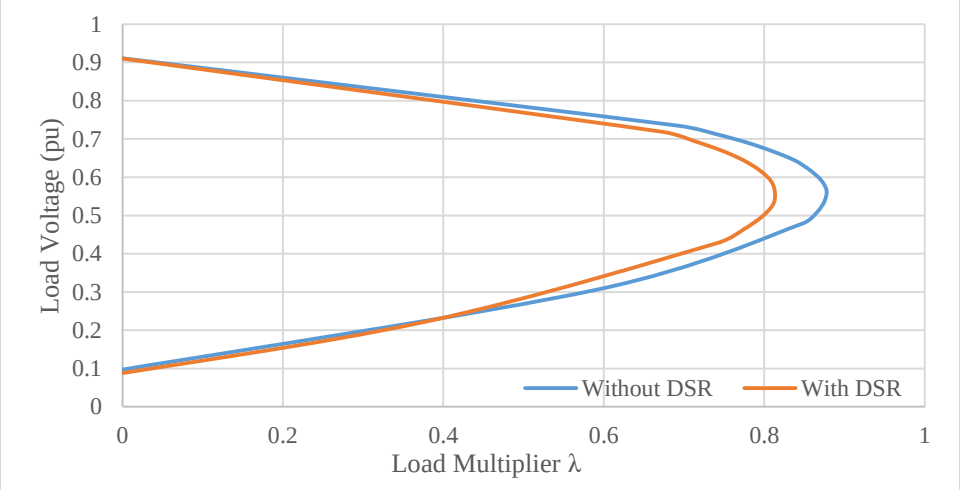


Figure 22: Voltage stability curves for circuit in Figure 21 with and without DSRs

5.4.2 Stability Curve for Hybrid Models with PV

Another interesting case to consider the impact on the voltage stability limit is a hybrid model with PV integration at the distribution level. The circuit is shown in Figure 23. The original IEEE 39 bus transmission network loads were scaled down to approximately 10% of their original nominal values, and each load was replaced with 8 IEEE 123 bus distribution feeders. The complete model is made up of 6,445 components, 223 are at the transmission level and the rest are distribution components. 331 PV units were added at the distribution level with an output of 680MW. The total load of the system is 689MW, which means PV at the distribution level is supplying 99% of the generation. The PV generation at the distribution level pushes 129 MW into the transmission system. In addition, the system is three phase and has an imbalance of 15%.

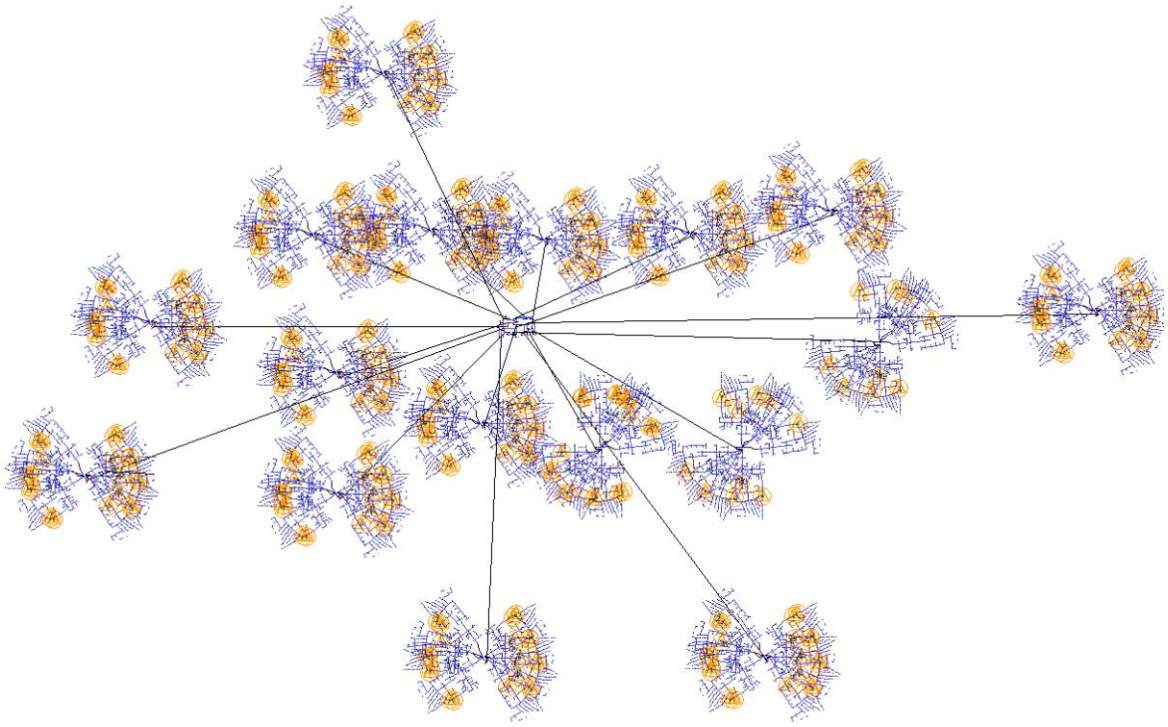


Figure 23: Hybrid model of IEEE 39 bus for transmission network and 17 distribution feeders made from IEEE 123 distribution circuits.

To study the voltage stability impact of PV in the distribution level, two cases are considered here, with and without PV generation. The voltage stability curves for both cases are shown in Figure 24.

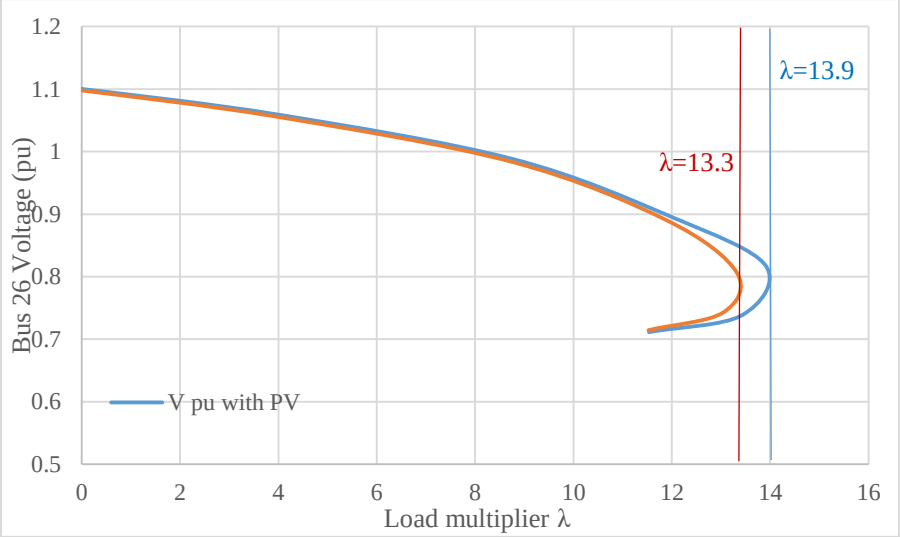


Figure 24: Voltage stability curves for hybrid circuit in Figure 23 with and without PV generation

We can notice that the voltage stability curves are almost identical until a load scaling factor of λ=11 is reached. Beyond this point, we notice that the system with PV generation has a better voltage profile, all the way to the tip of the curve. In addition, we can notice that the system with PV generation has a larger voltage stability margin. The maximum loading point for the system without PV is λ=13.3, while it is λ=13.9 for the system with PV generation. This difference means that the system can handle 413.4 MW more load with PV generation before voltage collapse, in comparison with the system without PV generation.

It should be taken into account that this does not merely show that we have more generation in the system with the addition of PV generation and, therefore, the system can handle more load. During our study no limits are placed on the 10 transmission generators and they are available to supply

as much power as the system requires. Thus, this is an indication that the presence of Distributed Energy Resources (DER) at the distribution level can have a positive effect on the voltage stability of the system by increasing the margin before voltage collapse.

CHAPTER 6 – CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

With the growth of generation resources at the distribution level, integrated transmission and distribution modeling is becoming more important. In addition, power flow solvers need to provide accurate answers for realistic circuit models under extreme loading and voltage conditions.

A new power flow algorithm, where Graph Trace Analysis (GTA) is used to implement a modified Gauss-Seidel algorithm coupled with a continuation method, is presented. The new algorithm is referred to as the GS-GTA power flow algorithm.

It has been shown that GS-GTA power flow is able to solve both transmission and distribution in the same model. It was also demonstrated that the GS-GTA based power flow could solve a variety of hybrid models, including models with reverse power flow from distribution to transmission [37].

The GS-GTA approach to solving the power flow problem is fundamentally very different from the Newton-Raphson approach. The overall system of equations to be solved consists of Component Equations (CE), Kirchhoff Voltage Law Equations (KVL), and Kirchhoff Current Law Equations (KCL). Assuming a system with *n* busses (not counting the reference), with the Newton-Raphson approach, the KVL (in terms of node voltages) are substituted into the CE, and that result is in turn substituted into the KCL to produce a system of *n* equations in *n* unknowns, the node voltages. After that, the first two terms of a Taylor series expansion are used to linearize the

nonlinear system of equations, and the linear model is iteratively solved to arrive at a solution, and the solution is not checked against KCL and KVL.

On the other hand, GS-GTA converges a system of *2m* equations in *2m* unknowns, where *m* is the number of components in the system, for which no equation substitutions are used. During each iteration of the GTA power flow, KVL and KCL are explicitly implemented and checked using traces. That is, unless both KVL and KCL are satisfied, the GS-GTA power flow does not converge. This results in a very robust power flow solution, as demonstrated by the examples presented.

The robustness of the GS-GTA power flow has been investigated and compared to the robustness of traditional power flow algorithms. Using a set of test circuits that represent features of real power system construction, a comparison between different types of traditional algorithms (Newton-Raphson and Forward/Backward Sweep) that solve transmission and distribution networks has been performed. It was demonstrated that the GS-GTA power flow was the only algorithm capable of solving the example circuits.

In comparing the GS-GTA power flow against the most accurate of the power flow algorithms tested, the GS-GTA power flow could solve models that are orders of magnitude larger, and on a specific example the GS-GTA power flow was demonstrated to solve 2.5 times faster.

The GS-GTA power algorithm was also shown to be able to solve a model of a real system, which included transmission, radial distribution, lightly meshed distribution, and heavily meshed distribution networks, all in one system. This model contained over 200 network protectors. These network protectors create an extra challenge for power flow algorithms as the network topology often changes between power flow iterations. However, such changing topology does not add any

additional topology computation burden to the GS-GTA power flow. This is opposed to matrix-based algorithms, which would require re-construction of matrices due to the topology changes.

As implemented, the only GS-GTA power flow limitation on model size is available computer memory, where a one million-component model requires approximately 2 GB of computer memory [10]. Because GS-GTA calculations work in terms of traces, GS-GTA based calculations can be distributed across processors by simply distributing the model. In addition, using the method described in this work, which does not require matrices, but completely relies on traces, the algorithm can be distributed across multiple processors without the need for a master or controlling processor.

The ability of the GS-GTA algorithm to solve for the steady-state voltage stability curve, or nose curve, was also tested. It was shown that the GS-GTA algorithm could be used to calculate the nose curves, and the answers produced by the GS-GTA algorithm were verified using a different continuation power flow algorithm.

The ability to solve for the steady-state voltage stability curve allows studying the behavior of more complicated systems, other than transmission networks. It is of interest to study the impact that the addition of PV or DSR, for example, have on the system voltage stability limits. Examples presented show that GS-GTA is able to perform this type of voltage stability analysis. No other power flow algorithm has been reported that

- Can analyze steady state voltage stability for systems with DSRs
- Can analyze steady state voltage stability for integrated transmission and distribution models, where high PV penetration exists in the distribution system.

6.2 FUTURE WORK

As part of the future work, we are interested in implementing the GS-GTA algorithm using distributed computation and record algorithm running speed differences. The fact that the GS-GTA algorithm does not rely on matrices makes distributing the algorithm by distributing the model on multiple processors possible [22]. For large systems distributing the model and calculations running on the model will decrease the convergence time. We are also interested in investigating GS-GTA based time series analysis and compare it with traditional methods. Time series analysis is more important now in power systems with distributed generation. The intermittency of renewable energy resources requires time series analysis in order to capture the changes that the power system experiences due to rapid fluctuations in generation.

Another part of proposed future work is to develop a theory behind the reason why GTA based algorithms are more robust than traditional algorithms. We would also like to investigate non-Jacobian Newton-Raphson algorithms. In this dissertation we show how Newton-Raphson algorithms struggle in many conditions, and the culprit in most of these cases is the Jacobian matrix. Therefore, we would like to compare the GS-GTA algorithm with a Krylov space based Newton-Raphson.

REFERENCES

[1] W.F. Tinney, C.E. Hart, Power flow solution by Newton's method, IEEE Transactions on Power Apparatus and systems, (1967) 1449-1460.

[2] B. Stott, O. Alsaç, Fast decoupled load flow, IEEE transactions on power apparatus and systems, (1974) 859-869.

[3] S. Tripathy, G.D. Prasad, O. Malik, G. Hope, Load-flow solutions for ill-conditioned power systems by a Newton-like method, IEEE Transactions on Power apparatus and Systems, (1982) 3648-3657.

[4] S. Iwamoto, Y. Tamura, A load flow calculation method for ill-conditioned power systems, IEEE Transactions on Power Apparatus and Systems, (1981) 1736-1743.

[5] W. Kersting, D. Mendive, An application of ladder network theory to the solution of three-phase radial load-flow problems, in: IEEE Conference Paper presented at the IEEE Winter Power Meeting, New York, 1976.

[6] PNNL, Integrated Transmission and Distribution Control, (2013).

[7] H. Jain, K. Rahimi, A. Tbaileh, R.P. Broadwater, A.K. Jain, M. Dilek, Integrated Transmission & Distribution System Modeling and Analysis: Need & Advantages, arXiv preprint arXiv:1606.01612, (2016).

[8] A. Tiranuchit, R. Thomas, A posturing strategy against voltage instabilities in electric power systems, IEEE Transactions on Power Systems, 3 (1988) 87-93.

[9] K.M. Sambarapu, S.M. Halpin, Sparse matrix techniques in power systems, in: System Theory, 2007. SSST'07. Thirty-Ninth Southeastern Symposium on, IEEE, 2007, pp. 194-198.

[10] P. Wiley, A parallel architecture comes of age at last, IEEE spectrum, 24 (1987) 46-50.

[11] S. Naka, T. Genji, Y. Fukuyama, Practical equipment models for fast distribution power flow considering interconnection of distributed generators, in: Power Engineering Society Summer Meeting, 2001, IEEE, 2001, pp. 1007-1012.

[12] V. Ramesh, On distributed computing for on-line power system applications, International Journal of Electrical Power & Energy Systems, 18 (1996) 527-533.

[13] D. Rajicic, A. Bose, A modification to the fast decoupled power flow for networks with high R/X ratios, IEEE Transactions on Power Systems, 3 (1988) 743-746.

[14] V. Ajjarapu, C. Christy, The continuation power flow: a tool for steady state voltage stability analysis, IEEE transactions on Power Systems, 7 (1992) 416-423.

[15] C.S. Cheng, D. Shirmohammadi, A three-phase power flow method for real-time distribution system analysis, IEEE Transactions on Power Systems, 10 (1995) 671-679.

[16] G.-X. Luo, A. Semlyen, Efficient load flow for large weakly meshed networks, IEEE Transactions on Power Systems, 5 (1990) 1309-1316.

[17] D. Shirmohammadi, H. Hong, A. Semlyen, G. Luo, A compensation-based power flow method for weakly meshed distribution and transmission networks, IEEE Transactions on Power Systems, 3 (1988) 753-762.

[18] R. Broadwater, S. Rahman, H. Shaalan, R. Lee, A distribution engineering workstation for undergraduate and graduate education, IEEE transactions on power systems, 8 (1993) 1385-1391.

[19] D. Cheng, D. Zhu, R.P. Broadwater, S. Lee, A graph trace based reliability analysis of electric power systems with time-varying loads and dependent failures, Electric power systems research, 79 (2009) 1321-1328.

[20] L.R. Feinauer, M.E. Ison, R.P. Broadwater, Generic algorithms for analysis of interdependent multidomain network systems, International Journal of Critical Infrastructures, 6 (2009) 81-95.

[21] M. Dilek, F. De Leon, R. Broadwater, S. Lee, A robust multiphase power flow for general distribution networks, IEEE Transactions on Power Systems, 25 (2010) 760-768.

[22] F. Li, R.P. Broadwater, Distributed algorithms with theoretic scalability analysis of radial and looped load flows for power distribution systems, Electric Power Systems Research, 65 (2003) 169-177.

[23] H. Hale, J. Ward, Digital computer solution of power flow problems, AIEE Transactions, pt. III (Power Apparatus and Systems), 75 (1956) 398-402.

[24] A. Glimn, G. Stagg, Automatic calculation of load flows, Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems, 76 (1957) 817-825.

[25] H. Hale, R. Goodrich, Digital Computation or Power Flow-Some New Aspects, Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems, 78 (1959) 919-923.

[26] P. Gupta, M.H. Davies, Digital computers in power system analysis, Proceedings of the IEE-Part A: Power Engineering, 108 (1961) 383-398.

[27] A. Brameller, J. Denmead, Some improved methods for digital network analysis, Proceedings of the IEE-Part A: Power Engineering, 109 (1962) 109-116.

[28] J.Q. Wu, A. Bose, Parallel solution of large sparse matrix equations and parallel power flow, IEEE Transactions on Power Systems, 10 (1995) 1343-1349.

[29] B. Stott, Effective starting process for Newton-Raphson load flows, in: Proceedings of the Institution of Electrical Engineers, IET, 1971, pp. 983-987.

[30] W.F. Tinney, J.W. Walker, Direct solutions of sparse network equations by optimally ordered triangular factorization, Proceedings of the IEEE, 55 (1967) 1801-1809.

[31] S.T. Despotovic, B.S. Babic, V.P. Mastilovic, A rapid and reliable method for solving load flow problems, IEEE Transactions on Power Apparatus and Systems, (1971) 123-130.

[32] B. Stott, Decoupled Newton load flow, IEEE Transactions on Power Apparatus and Systems, (1972) 1955-1959.

[33] D.R. Musser, A.A. Stepanov, Generic programming, in: International Symposium on Symbolic and Algebraic Computation, Springer, 1988, pp. 13-25.

[34] D.B. West, Introduction to graph theory, Prentice hall Upper Saddle River, 2001.

[35] D. Kleppinger, R. Broadwater, C. Scirbona, Generic reconfiguration for restoration, Electric Power Systems Research, 80 (2010) 287-295.

[36] R.P. Broadwater, H.E. Shaalan, A. Oka, R.E. Lee, Distribution system reliability and restoration analysis, Electric power systems research, 29 (1994) 203-211.

[37] A. Tbaileh, H. Jain, R. Broadwater, J. Cordova, R. Arghandeh, M. Dilek, Graph Trace Analysis: An Object-Oriented Power Flow, Verifications and Comparisons, Electric Power Systems Research, (2017).

[38] A. Greenbaum, Iterative methods for solving linear systems, Siam, 1997.

[39] T. Gaddis, Starting Out with C++ from Control Structures through Objects, Brief Version, Pearson, 2015.

[40] P. Bachmann, Die analytische zahlentheorie, Teubner, 1894.

[41] IEEE, Published Solutions - IEEE 118 Bus System, in, http://www2.ee.washington.edu/research/pstca/pf118/ieee118cdf.txt, 2016.

[42] IEEE, Published Solutions - IEEE 123 Bus System, in, http://ewh.ieee.org/soc/pes/dsacom/testfeeders/, 2016.

[43] H. Jain, A. Parchure, R.P. Broadwater, M. Dilek, J. Woyak, Three-Phase Dynamic Simulation of Power Systems Using Combined Transmission and Distribution System Models, (2015).

[44] M.N. Nazir, S. Omran, R. Broadwater, Feasibility of DSR applications in transmission grid operation—Control of power flow and imbalanced voltage, Electric Power Systems Research, 131 (2016) 187-194.

[45] H. Jain, Derivation of Current Control Expressions for D-PV and Dynamic Simulation Parameters for Synchronous Generators and D-PV, in, https://www.scribd.com/document/317305565/D-PV-Derivations-and-Hybrid-Model-Description, 2015.

[46] ATP, ATPDraw Download, in, http://www.atpdraw.net/downloads.php, 2016.

[47] IEEE, WECC 9 Bus System, in, http://icseg.iti.illinois.edu/wscc-9-bus-system/, 2016.

[48] A. Pai, Energy function analysis for power system stability, Springer Science & Business Media, 2012.

[49] EDD, DEW Software, in, http://www.edd-us.com/, 2016.

[50] GridLAB-D, GridLAB-D Download, in, http://www.gridlabd.org/, 2016.

[51] OpenDSS, OpenDSS Download, in, http://smartgrid.epri.com/SimulationTool.aspx, 2016.

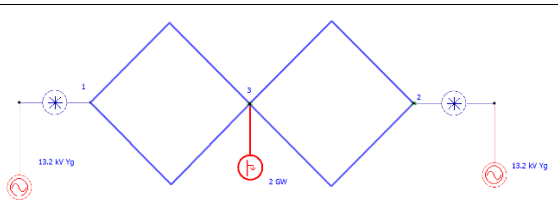
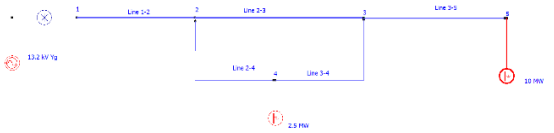
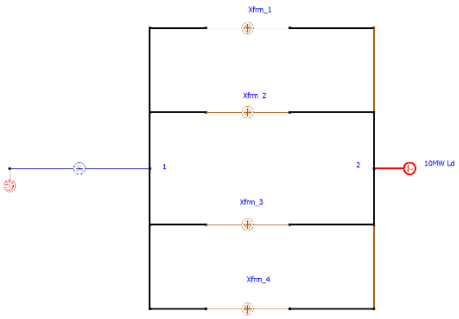
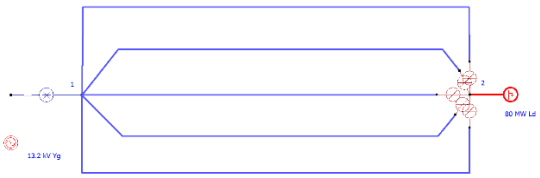
[52] G.H. Golub, C.F. Van Loan, Matrix computations, JHU Press, 2012.

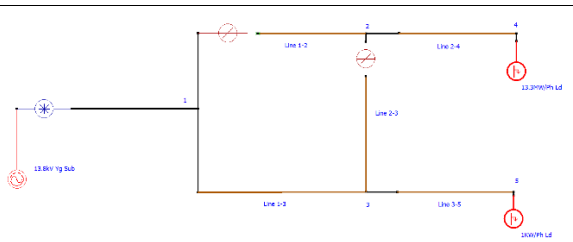
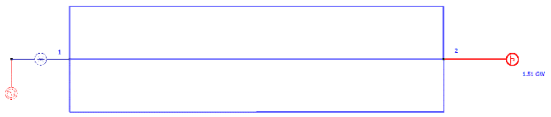
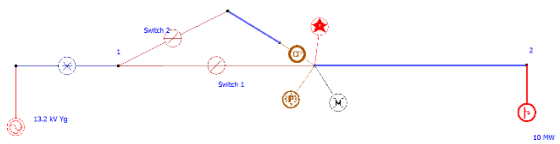
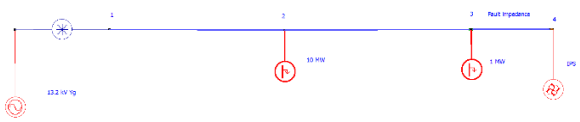
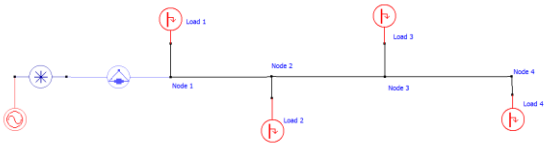
[53] PowerWorld, PowerWorld in, https://www.powerworld.com/, 2017.

[54] Siemens, PSS/E, in, http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/software-solutions/planning-data-management-software/planning-simulation/pages/pss-e.aspx, 2017.

[55] GE, PSLF, in, http://www.geenergyconsulting.com/practice-area/software-products/pslf, 2017.

APPENDIX A – ROBUSTNESS TESTING CIRCUITS DATA

Case number	Case Name	Circuit model	Parameters	Case description
1	Double circuit with single load		Source: 13.2kV Yg Each line impedance: X = 0.174 Ohms Load: 2 GW balanced	Heavily loaded looped system
2	Stiff 3-phase circuit with 1-phase loop		Source: 13.2kV Yg Each 3-ph Line: R+jX = 0.8+j1.9 Ohms 1-ph Line: R+jX = 0.4+j0.95 Ohms Load 4: 2.5MW phase A Load 5: 10MW	Single-phase loop inside of three-phase loop
3	Four parallel transformers		Source: 34.5kV Yg Each Transformer: 34.5/13.8 kV Each transformer impedance: X = 5.576 Ohms Load: 10MW balanced	Four substation transformers connected in parallel
4	Stiff low impedance loop		Source: 13.2kV Yg Each line impedance: X = 1.7 Ohms Load: 80MW per phase	Low impedance loops as might occur in a downtown network where cables have the same impedance
5	Radial circuit with three loads 6 miles apart		Source: 13.2kV Yg Each line impedance: R + jX = 2.5 + j6.0 Ohms Load 1: 500kW/phase Load 2: 600kW/phase Load 3: 700kW/phase	Heavily loaded radial system
6	High impedance loop		Source: 13.2kV Yg Line 1-2: R = 1000 Ohms Line 1-3:	Low impedance and high impedance lines running in

			R = 1 Ohms Line 2-3: R = 0.1 Ohms Line 2-4: R = 0.1 Ohms Line 3-5: R = 0.1 Ohms Load 4: 12 MW/phase Load 5: 1kW/phase	parallel and serving loads differing in magnitude by a factor of 1000.
7	Loops with small difference impedance		Source: 13.2kV Yg Line impedances: jX = j 0.17424 Ohms jX = j 0.17249 Ohms jX = j 0.17075 Ohms Load: 1.50GW balanced	Low impedance lines such as might occur in a downtown network where cables have different impedances
8	Switches in parallel with poles		Source: 13.2kV Yg Line impedance: jX = j 1.74 Ohms Load: 10MW balanced	Incorporating structural components, poles and manholes, and parallel switches into the analysis
9	Fault using IPS		Source: 13.2kV Yg Each Line impedance: jX = j 1.74 Ohms Load 1: 10MW balanced Load 2: 1MW balanced Fault impedance: X=0	Simulating faults by setting one phase of voltage source to zero
10	Three wire secondary system		Source: 12.47kV Yg Line impedance: Loads	Detailed circuit of a secondary system beyond distribution transform

APPENDIX B – ROBUSTNESS TESTING CIRCUITS SOLUTIONS

Table 11: Robust test circuit 1 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 3	ABC	6.60	6.60	6.60	-30.00	-150.00	90.00

Table 12: Robustness test circuit 1 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1	ABC	21868.54	21868.54	21868.54	-29.97	-149.97	90.03
line 2	ABC	21867.82	21867.82	21867.82	-29.95	-149.95	90.05
line 3	ABC	21867.82	21867.82	21867.82	150.05	30.05	-89.95
line 4	ABC	21867.82	21867.82	21867.82	150.05	30.05	-89.95

Table 13: Robustness test circuit 2 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	6.69	7.05	7.05	-10.17	-125.47	114.53
Bus 3	ABC	6.37	6.56	6.56	-14.86	-131.83	108.17
Bus 4	A	6.46			-13.83		
Bus 5	ABC	5.97	6.16	6.16	-22.36	-139.11	100.89

Table 14: Robustness test circuit 2 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
Line 1-2	ABC	763.26	437.29	437.29	-18.55	-138.95	101.05
Line 2-3	ABC	299.99	437.29	437.29	-19.88	-138.95	101.05
Line 2-4	A	463.40			-17.69		
Line 3-4	A	138.01			-27.08		
Line 3-5	ABC	437.26	437.29	437.29	-22.15	-138.95	101.05

Table 15: Robustness test circuit 3 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	19.92	19.92	19.92	0.00	-120.00	120.00
Bus 2	ABC	7.95	7.95	7.95	-4.20	-124.20	115.80

Table 16: Robustness test circuit 3 transformers currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
Xfmr 1	ABC	41.84	41.84	41.84	-4.13	-124.13	115.87
Xfmr 2	ABC	41.84	41.84	41.84	-4.16	-124.16	115.84
Xfmr 3	ABC	41.84	41.84	41.84	-4.13	-124.13	115.87
Xfmr 4	ABC	41.84	41.84	41.84	-4.13	-124.13	115.87

Table 17: Robustness test circuit 4 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	6.69	6.69	6.69	-28.68	-148.68	91.32

Table 18: Robustness test circuit 4 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1	ABC	2099.32	2099.32	2099.32	-28.65	-148.65	91.35
line 2	ABC	2099.32	2099.32	2099.32	-28.65	-148.65	91.35
line 3	ABC	2099.40	2099.40	2099.40	-28.67	-148.67	91.33
line 4	ABC	2099.32	2099.32	2099.32	-28.65	-148.65	91.35
line 5	ABC	2099.32	2099.32	2099.32	-28.65	-148.65	91.35

Table 19: Robustness test circuit 5 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	6.73	6.73	6.73	-9.95	-129.95	110.05
Bus 3	ABC	6.19	6.19	6.19	-18.53	-138.53	101.47
Bus 4	ABC	5.94	5.94	5.94	-23.65	-143.65	96.35

Table 20: Robustness test circuit 5 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1-2	ABC	235.10	235.10	235.10	-18.14	-138.14	101.86
line 2-3	ABC	170.41	170.41	170.41	-21.29	-141.29	98.71
line 3-4	ABC	91.85	91.85	91.85	-23.65	-143.65	96.35

Table 21: Robustness test circuit 6 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 3	ABC	6.05	6.05	6.05	0.00	-120.00	120.00
Bus 4	ABC	5.73	5.73	5.73	0.00	-120.00	120.00
Bus 5	ABC	6.05	6.05	6.05	0.00	-120.00	120.00

Table 22: Robustness test circuit 6 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
Line 1-2	ABC	1.73	1.73	1.73	180.00	60.00	-60.00
Line 1-3	ABC	1572.99	1572.99	1572.99	0.00	-120.00	120.00
Line 2-3	ABC	1572.86	1572.86	1572.86	0.00	-120.00	120.00
Line 2-4	ABC	1574.59	1574.59	1574.59	0.00	-120.00	120.00
Line 3-5	ABC	0.13	0.13	0.13	0.00	-120.00	120.00

Table 23: Robustness test circuit 7 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	6.61	6.61	6.61	-29.85	-149.85	90.15

Table 24: Robustness test circuit 7 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1	ABC	21768.02	21768.02	21768.02	-29.82	-149.82	90.18
line 2	ABC	21995.38	21995.38	21995.38	-29.83	-149.83	90.17
line 3	ABC	22212.26	22212.26	22212.26	-29.82	-149.82	90.18

Table 25: Robustness test circuit 8 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	7.59	7.59	7.59	-5.72	-125.72	114.28

Table 26: Robustness test circuit 8 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1-2	ABC	436.02	436.02	436.02	-5.49	-125.49	114.51

Table 27: Robustness test circuit 9 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (B)	VMagKv (C)	VAngDg (A)	VAngDg (B)	VAngDg (C)
Bus 1	ABC	7.62	7.62	7.62	0.00	-120.00	120.00
Bus 2	ABC	3.79	5.38	7.61	-5.30	-79.05	117.13
Bus 3	ABC	0.06	7.62	7.62	89.95	-30.00	120.00

Table 28: Robustness test circuit 9 line currents

Component Name	Phase	IMagA (A)	IMagA (B)	IMagA (C)	IAngDg (A)	IAngDg (B)	IAngDg (C)
line 1-2	ABC	2216.28	2874.69	218.69	-84.79	105.31	117.13
line 2-3	ABC	2179.34	3310.97	218.69	-96.17	104.73	-62.87

Table 29: Robustness test circuit 10 bus voltages

Component Name	Phase	VMagKv (A)	VMagKv (A')	VAngDg (A)	VAngDg (A')
Bus 1	AA'	0.118441	0.118441	0.061787	179.9382
Bus 1'	AA'	0.118354	0.118354	0.073149	-179.927
Bus 2	AA'	0.114035	0.114035	-0.01427	179.9857
Bus 2'	AA'	0.11367	0.11367	0.034748	-179.965
Bus 3	AA'	0.110211	0.110211	-0.16176	179.8382
Bus 3'	AA'	0.110032	0.110032	-0.13678	179.8632
Bus 4	AA'	0.092372	0.092372	-0.41842	179.5816
Bus 4'	AA'	0.090128	0.090128	-0.50092	179.4991

Table 30: Robustness test circuit 10 line and transformer currents

Component Name	Phase	VMagKv (A)	VMagKv (A')	VAngDg (A)	VAngDg (A')
Bus 1	AA'	0.12	0.12	0.06	179.94
Bus 1'	AA'	0.12	0.12	0.07	-179.93
Bus 2	AA'	0.11	0.11	-0.01	179.99
Bus 2'	AA'	0.11	0.11	0.03	-179.97
Bus 3	AA'	0.11	0.11	-0.16	179.84
Bus 3'	AA'	0.11	0.11	-0.14	179.86
Bus 4	AA'	0.09	0.09	-0.42	179.58
Bus 4'	AA'	0.09	0.09	-0.50	179.50