OPGrid: Software Tool Development for Transmission Networks Analysis

by

Bassam Mohamed

Submitted to the Department of Electrical Engineering, Electronics, Computers and Systems
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Energy Conversion and Power Systems

at the

UNIVERSIDAD DE OVIEDO

July 2014

Author . . . .
Bassam Mohamed

Certified by . . . .
Pablo Arboleya Arboleya
Associate Professor
Thesis Supervisor

OPGrid: Software Tool Development for Transmission Networks Analysis

by

Bassam Mohamed

Submitted to the Department of Electrical Engineering, Electronics, Computers and Systems
on July 18, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Energy Conversion and Power Systems

Abstract

This thesis is focused on developing power flow solver based on Matlab tools. The solver is used to investigate the steady state of transmission network under different cases of demand variation. The solver is designed to be flexible and able to handle standard data format for exchanging input and output with other software tools. The active power dispatch can be solved based on three mode Conventional , Optimal and Droop Control. The solver setting and case parameters are organized in single structure to be flexible for configuration and easy for exchange. New data file format is proposed to encapsulate the all case information in single file. The solver is implemented to solve multiple case of demand variation by single call which to automate the analysis process and minimize to iteration time. The mathematical equation model can be selected to be based on conventional admittance matrix or incident matrix methods. The solver results of IEEE test cases have been verified with alternative commercial and scientific tools and they satisfied the required accuracy and performance. The solver is released as open source project to support additional functionality and full customization for scientific research.

Thesis Supervisor: Pablo Arboleya Arboleya
Title: Associate Professor

Acknowledgments

May words are not enough to thank my professors who guide me through all of my study like stars in sky. It was great honor for me to follow their light until now and forever. I would like to thank my wife and my kids for supporting me in hard times and for every person who try to help me specially my family and with great respect to my mother and my father. I would like to thanks also , Vattenfall Vindkraft company for supporting this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Symbol Notation

The following symbols are common be used to represent the power flow problem

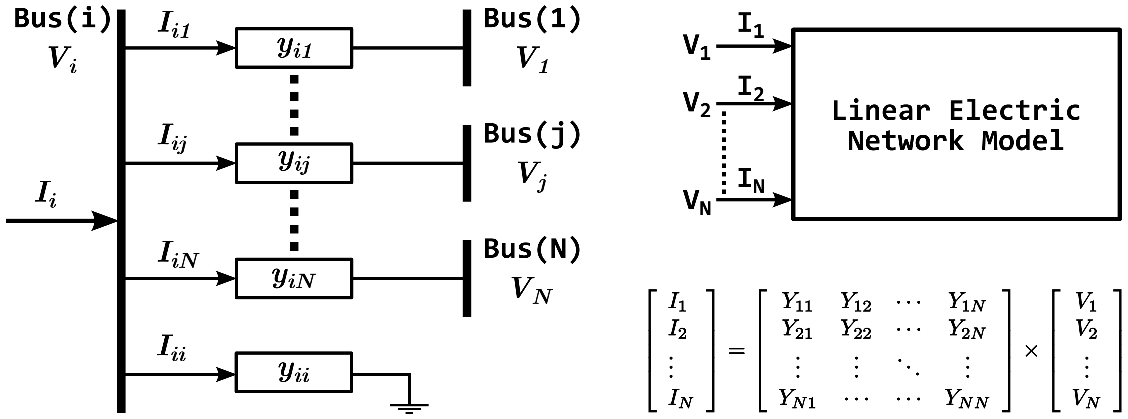
Symbol	Description
x	Dependent or state variables
$f(x)$	Objective function
$h(x)$	Function of equality constraints
$g(x)$	Function of inequality constraints
N	Total number of system buses
M	Total number of system lines
i	Source bus index
j	Destination bus index
k	Line index
J	$\sqrt{-1}$
S_i	Apparent power injected at bus i (generation - load)
P_i	Real power injected at bus i (generation - load)
Q_i	Reactive power injected at bus i (generation - load)
V_i	The complex voltage at bus i
δ_i	Voltage angle at bus i
E_i	Real component of complex voltage at bus i
F_i	Imaginary component of complex voltage at bus i
y_k	The complex admittance of line k
Y_{ij}	The complex admittance between bus i and j
B_{ij}	The susceptance (imaginary component of Y_{ij})
G_{ij}	The conductance (real component of Y_{ij})
θ_{ij}	The angle of Y_{ij}
$(\ )^*$	Complex conjugate
$(\ )^{*T}$	Complex conjugate and transpose of matrix
$(\ )^T$	Matrix transpose without complex conjugate
$[\ ]$	Diagonal matrix of column vector

Table 1.1: Symbol notation list

1.2 Admittance Matrix Model of Electric Network

The transmission lines connecting electric networks are linear elements. They can be modeled as equivalent circuit based on impedance matrix (Z) or admittance matrix (Y) . Each element in impedance matrix (Z_{ij}) defines the Thevenin's equivalent impedance between the two buses (i) and (j). The diagonal elements of impedance matrix equals to the Thevenin's equivalent impedance between the bus and ground. The impedance matrix is widely applied in the fault analysis of power systems. The admittance matrix is the inverse of the impedance matrix. The admittance matrix is common be used to model the electric network for solving the power flow [34]. The diagonal elements in admittance matrix (Y_{ii}) is called self admittance and it includes the sum of all admittance connected to the bus. The off diagonal elements (Y_{ij}) are called mutual admittance and it includes the negative value of admittance between buses (i) and (j). The following equations demonstrate the relation between injected current (I) and the bus voltage (V) based on admittance matrix method shown in Figure 1-1:

$$I = Y \times V \quad , \quad Y_{ij} = \begin{cases} -y_{ij} & j \neq i \\ \\ \sum_{j=1}^{N} y_{ij} & j = i \end{cases} \tag{1.1}$$



Bus(i)
V_i
I_{i1}
y_{i1}
Bus(1)
V_1
I_{ij}
y_{ij}
Bus(j)
V_j
I_i
I_{iN}
y_{iN}
Bus(N)
V_N
I_{ii}
y_{ii}
V_1
I_1
V_2
I_2
V_N
I_N
Linear Electric Network Model
$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N} \\ Y_{21} & Y_{22} & \cdots & Y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N1} & \cdots & \cdots & Y_{NN} \end{bmatrix} \times \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix}$

Figure 1-1: Linear network model based on admittance matrix method

1.2.1 Characteristics of Admittance Matrix

- Complex :

  All off diagonal elements has positive imaginary part. The diagonal elements may have positive or negative imaginary part based on number of inductive elements (lines and shunt reactors) and relative to capacitive elements (capacitor banks) connected to the bus.

- Sparse:

  The size of admittance matrix equals to the square of buses count ($N \times N$) and it does not depend on the real number of lines (M). But most of the elements are zeros. The following equation defines the real size required to store the admittance matrix : The count of non zero elements of $(Y) = N + 2 \times M$ [1]

- Symmetry :

  The admittance matrix has symmetry around the diagonal line. For linear networks without phase shift transformers, The upper triangle elements are equal to the lower triangle elements $Y_{ij} = Y_{ji}$. This feature can be used to reduce the calculation time and save the memory locations.

- Abstract :

  The admittance matrix hide the real parameter of the network such as the parallel lines and the shunt admittance at each bus. Different electric networks can have the same admittance matrix. So that , there is no way to recover back the network parameters from the admittance matrix.

- Constant :

  For conventional power flow the admittance matrix is created once and used as constant along all solver iteration processes. It depends only on the network parameters which does not change relative to any variable.

- Unique:

  For any linear electric network, there is a unique admittance matrix because the mathematical model of the matrix has single representation based on buses order .

[1]M : Lines count assuming parallel lines are joined as single line

1.3 Basic Power Flow Formulation

The conventional power flow problem represents the power balance between the net injected power at each bus and the total power exchanged through the lines connected to this bus. The net injected power (S_i) at each bus is defined as the net result of generation (S_{Gi}) and load power (S_{Li}) as shown in Equation 1.2. The following equations define the complex power based on the admittance matrix [35]:

$$S_i = S_{Gi} - S_{Li} \tag{1.2}$$

$$S = V \cdot I^* = V \cdot (Y \times V)^* \tag{1.3}$$

Equation 1.3 is the complex form of the essential power flow formulation. This equation can be represented in scalar form based on polar or rectangular coordinate of voltage and admittance as shown in the following forms [89]:

- Polar voltage and admittance : $V_i = |V_i| \cdot e^{J \cdot \delta_i} \quad , \quad Y_{ij} = |Y_{ij}| \cdot e^{J \cdot \theta_{ij}}$

$$P_i = |V_i| \cdot \sum_{j=1}^{N} |V_j| \cdot |Y_{ij}| \cdot \cos(\delta_i - \delta_j - \theta_{ij}) \tag{1.4}$$

$$Q_i = |V_i| \cdot \sum_{j=1}^{N} |V_j| \cdot |Y_{ij}| \cdot \sin(\delta_i - \delta_j - \theta_{ij}) \tag{1.5}$$

- Polar voltage and rectangular admittance : $V_i = |V_i| \cdot e^{J \cdot \delta_i} \quad , \quad Y_{ij} = G_{ij} + J \cdot B_{ij}$

$$P_i = |V_i| \cdot \sum_{j=1}^{N} |V_j| \cdot (G_{ij} \cdot \cos(\delta_i - \delta_j) + B_{ij} \cdot \sin(\delta_i - \delta_j)) \tag{1.6}$$

$$Q_i = |V_i| \cdot \sum_{j=1}^{N} |V_j| \cdot (G_{ij} \cdot \sin(\delta_i - \delta_j) + B_{ij} \cdot \cos(\delta_i - \delta_j)) \tag{1.7}$$

- Rectangular voltage and admittance : $V_i = Vd_i + J \cdot Vq_i \quad , \quad Y_{ij} = G_{ij} + J \cdot B_{ij}$

$$P_i = Vd_i \cdot \sum_{j=1}^{N} (Vd_j \cdot G_{ij} - Vq_j \cdot B_{ij}) + Vq_i \cdot \sum_{j=1}^{N} (Vq_j \cdot G_{ij} + Vd_j \cdot B_{ij}) \tag{1.8}$$

$$Q_i = Vq_i \cdot \sum_{j=1}^{N} (Vd_j \cdot G_{ij} - Vq_j \cdot B_{ij}) + Vd_i \cdot \sum_{j=1}^{N} (Vq_j \cdot G_{ij} + Vd_j \cdot B_{ij}) \tag{1.9}$$

1.4 Bus Classification

The buses of the electric network can be classified in general as generation buses and load buses. The generation buses are responsible of injecting active and reactive power to supply the network demand and regulate the bus voltage. The load buses consume active and reactive power and they can not regulate the bus voltage. The following bus classification is common be used in mathematical models for solving the power flow problem [98]:

- PQ Bus :

  PQ bus has fixed injected active power and reactive power. In normal case this type of buses is used to represent loads of the network but it can be used also to model small generation which has fixed active and reactive power and can not regulate the bus voltage. Most of the network buses are classified as PQ buses.

- PV Bus :

  PV bus is generation bus which defines the active power injection. It is also able to regulate the bus voltage to a given set point by injecting the required reactive power. PV bus is the normal model of any generation power plant. Some substations can be classified as PV buses if they have enough reactive power to regulate the bus voltage.

- Slack Bus :

  The slack bus is generation bus which can regulate its bus voltage and inject the required active power to balance the load and losses relative to other active power generation. Each electric network area must include only one slack bus. The slack bus should be selected as largest generation power plant. The voltage angle of slack bus is defined and it is used as reference for all other buses.

Bus Type	Given Variables	Unknown Variables
PQ	P_i , Q_i	V_i , δ_i
PV	P_i , V_i	Q_i , δ_i
Slack	V_i , δ_i	P_i , Q_i

Table 1.2: Power flow variables at each bus type

1.5 Solving the Basic Power Flow

The basic model of power flow demonstrated in section 1.3 can be defined by non linear system of equations. Each equation $h_i\left(X\right)$ represents the mismatch in active power $\Delta P_i(X)$ or reactive power $\Delta Q_i(X)$ between the injected power and the total exchange power through the lines connected to each bus. The vector (X) includes all unknown variables as defined in Table 1.2 for each bus based on its classification. The solver try to find the value of the unknown vector (X) which satisfy the mismatch functions $h_i\left(X\right)$ to be zero. The following equations define the mismatch function $h\left(X\right)$ as two groups of equations one for active power and the other one for reactive power.

$$h(X) = [\Delta P_1 \quad \Delta P_2 \cdots \Delta P_{Np} \quad \Delta Q_1 \quad \Delta Q_2 \cdots \Delta Q_{Nq}] \tag{1.10}$$

$$\Delta P_i = P_i(X) + P_{Li} - P_{Gi} \quad , \quad \Delta Q_j = Q_j(X) + Q_{Lj} - Q_{Gj} \tag{1.11}$$

This system of equations can be reduced by selecting only the mismatch equations of buses which have known injected power. Each PQ bus will have two equations and only one equation for each PV bus representing its active power. The unknown vector will include the bus voltage only. Other unknowns (P_G and Q_G) can be calculated after solving the reduced system using the following equations :

$$P_{Gi} = P_i(X) + P_{Li} \quad , \quad Q_{Gi} = Q_i(X) + Q_{Li} \tag{1.12}$$

Table 1.3 shows the main difference between the general system of equations and reduced system. The number of equation N_{eq} is reduced by $(N_{PV} + 1)$ which can speed up the solver iterations and reduce the matrix size.

Variable	General System	Reduced System
X	$[\delta \quad V \quad P_G \quad Q_G]$	$[\delta \quad V]$
Np	N	$N_{PQ} + N_{PV}$
Nq	N	N_{PQ}
N_{eq}	$2N$	$2N_{PQ} + N_{PV}$

Table 1.3: General and reduced system of equations for power flow

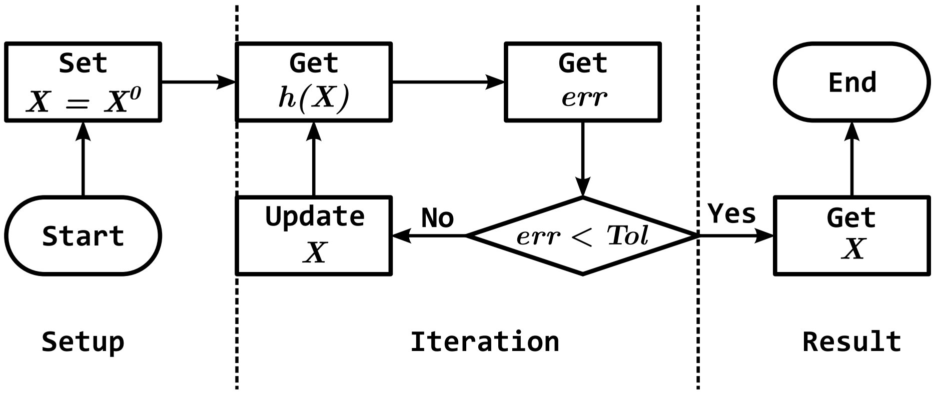


Figure 1-2: Iteration process for solving nonlinear system of equation

Solving nonlinear system of equation is iterative process as shown in Figure 1-2. It passes through three stages as following :

- Setup :

  Initialize the value of unknown vector by starting point (X^0). Flat start point is common to be used by setting all buses voltages to unity and others unknowns to zeros. In some cases the solver can not reach solution because the start point is selected far away from the solution point.

- Iteration :

  At each iteration, unknown vector is updated based on the solver algorithm. The iteration process stops when the error defined in Equation 1.13 is less than given tolerance or the iteration count is out of maximum limit.

  $$err \quad = \quad \|h(X)\| = \sqrt{h_1^2 + h_2^2 + \cdots + h_n^2} \tag{1.13}$$

- Result :

  Other unknowns can be calculated based on the solution vector (X). Finally the result is sorted in the required order and format.

The following sections explain three main methods for solving the basic power flow:

- Gauss Seidel Method
- Newton Raphson Method
- Fast Decoupled Method

1.6 Gauss Seidel Method

Gauss seidal method is based on rearrange the power flow equation to estimate the bus voltage (V_i^{k+1}) based on the last iteration value (V_i^k). The complex injection current at each bus (I_i) defined in Equation 1.14 is expressed by two relations. The first one is based on admittance matrix and the other one is based on injected power (S_i). Finally the bus voltage (V_i) can be estimated based on the injected power and its last value as shown in Equation 1.15. The equations are normally processed from top to bottom. Each equation will use the last updated value from the other equation because it will be more accurate [72].

$$I_i = Y_{ii} \cdot V_i + \sum_{j=1\,, j\neq i}^{N} Y_{ij} \cdot V_j = \left(\frac{S_i}{V_i}\right)^* \tag{1.14}$$

$$V_i^{k+1} = \frac{1}{Y_{ii}} \left[\left(\frac{S_i}{V_i^k}\right)^* - \sum_{j=1}^{i-1} Y_{ij} \cdot V_j^{k+1} - \sum_{j=i+1}^{N} Y_{ij} \cdot V_j^k \right] \tag{1.15}$$

The injection power is known for PQ buses but PV bus defines the active power only. So that the reactive power (Q_i) of each PV bus must be estimated by Equation 1.16. The estimated value of (Q_i) is checked to be within given limits. If the reactive power is out the boundary limit, the solver set it to the limit value and convert the bus type to be PQ because the bus can not regulate its voltage. The slack bus is not included in the iteration because its voltage is defined. The slack injected power can be calculated based on the bus voltage after reaching the solution [34].

$$Q_i = |V_i| \cdot \sum_{j=1}^{N} |V_j| \cdot (G_{ij} \cdot \sin(\delta_i - \delta_j) + B_{ij} \cdot \cos(\delta_i - \delta_j)) \tag{1.16}$$

The iteration process stops when the variation error between successive results is less than given tolerance. Equation Equation 1.17 defines this error as function of the absolute value voltage difference. The Guass seidal method is based on simple calculation per iteration but it needs many iteration to reach the solution and it depends on the number of buses and admittance matrix [25]. The conversion is not guaranteed and it is highly depending on the start point [34].

$$err = \sum_{j=1}^{N} \left| V_j^{k+1} - V_j^k \right| \tag{1.17}$$

1.7 Newton Raphson Method

The Newton Raphson method is used to find the roots on nonlinear equation using linear approximation. Taylor series expansion defined in Equation 1.18 can be used to represent any function based on its derivatives terms. Newton Raphson method neglects the higher order derivative terms and uses the first derivative only as shown in Equation 1.19.

$$h(X+\Delta X) = h(X) + \frac{\partial h(X)}{\partial X}\Delta X + \cdots + \left(\frac{\partial^n h(X)}{\partial X^n}\right)\frac{\Delta X^n}{n!} \tag{1.18}$$

$$h(X+\Delta X) \approx h(X) + \frac{\partial h(X)}{\partial X}\Delta X \tag{1.19}$$

When the function $h(X+\Delta X) \approx 0$ near the solution point, Equation 1.19 can be rewritten as relation between the current value of the function $h(X)$ and its derivative as defined in Equation 1.20. The same concept can be applied on system of nonlinear functions by partial derivative matrix (D) which is call Jacobin matrix. Equation 1.24 is the core of Newton Raphson iteration which calculate the new point based on the last point and the first partial derivative of the functions at last point [6], [96].

$$h(X) = -\frac{\partial h(X)}{\partial X}\Delta X \tag{1.20}$$

$$\begin{bmatrix} h_1(x_1, x_2, \cdots, x_n) \\ h_2(x_1, x_2, \cdots, x_n) \\ \vdots \\ h_n(x_1, x_2, \cdots, x_n) \end{bmatrix} = -\begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots & \frac{\partial h_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_n}{\partial x_1} & \frac{\partial h_n}{\partial x_2} & \cdots & \frac{\partial h_n}{\partial x_n} \end{bmatrix} \times \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_4 \end{bmatrix} \tag{1.21}$$

$$H(X) = -D \times \Delta X \quad , \quad D_{ij} = \frac{\partial h_i}{\partial x_j} \tag{1.22}$$

$$\Delta X = -D^{-1} \times H(X) = X(k+1) - X(k) \tag{1.23}$$

$$X(k+1) = X(k) - D^{-1} \times H(X) \tag{1.24}$$

The main feature of Newton Raphson :

- Quadratic convergence (less than 10 iteration even for large cases)
- Extensive calculation per iteration
- Less sensitive to start point but it still may not converge in some cases.

1.7.1 Solving Power Flow with Newton Raphson Method

Newton Raphson method was used for solving power flow problems since the late 1960s [86] and until now it still be used by most of the solvers as default algorithm because it provides fast and accurate results. The following equation defines the Newton Raphson iteration expression for power mismatch based on reduced system of equations using polar form of bus voltage.

$$-\begin{bmatrix} \frac{\partial P}{\partial \delta} & \frac{\partial P}{\partial V} \\ \frac{\partial Q}{\partial \delta} & \frac{\partial Q}{\partial V} \end{bmatrix} \times \begin{bmatrix} \Delta\delta \\ \Delta V \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \tag{1.25}$$

Each iteration requires solving system of linear equation ($A \times X = b$) by calculating the inverse or factorizing the Jacobian matrix. Nowadays , the computer possessing power increases rapidly but solving this system still need extensive calculation for large networks. So that, the following iterative methods are proposed to reduce the memory and time requirements for each iteration [89]:

- Quasi Newton (Dishonest) :

  Jacobian matrix may be kept constant for some iteration.
- Partial Update of Jacobian Matrix [50]:

  Only small portion of Jacobian matrix need to be updated.
- Preconditioners :

  Preconditioners convert the original system to be easier to solve with an iterative solver and gives the same solution.
- Jacobian Free Newton : [22]

  Adaptive preconditioner without need of Jacobian matrix or its eigenvalues.
- Krylov Newton methods [46][76]:

  Krylov methods approximate the inverse of the matrix (A) by polynomial $p(A)$. The following methods are based on Krylov subspace :
  - GMRES (Generalized Minimal Residual) [92]:
  - FGMRES (Flexible GMRES) :
  - BCGS (Bi-Conjugate Gradient Stabilized) [32]:
  - CGS (Conjugate Gradient Squared Method)

1.7.2 Convergence of Newton Raphson

Newton Raphson has stable convergence rate for most of the cases but it may not reach the expected results because it depends on starting point and the function slope variation. The Newton Raphson results can be classified as following cases [63]:

1. Well Conditioned Case Figure 1-3.(a):

   The solver can reach the result from flat start ($|V| = 1$, $\delta = 0$).

   This case is the most common case for normal power flow .

2. Ill Conditioned Case Figure 1-3.(b):

   The normal solver can not get the solution from flat start because of the attraction region is narrow or far away from the start point. Robust methods solve this case by using optimal multiplier (μ) as defined in Equation 1.26.

$$X(k+1) = X(k) + \mu\, \Delta X(k) \tag{1.26}$$

3. Saddle Point of Bifurcation:

   The solution can not be reached by normal or robust solvers because the Jacobian matrix is singular near the maximum loading conditions.

4. Limited Point of Bifurcation :

   The solver can not get the solution even for well conditioned point because the solution is out the boundary limits of the system. For example, the generator can not provide reactive power to regulate the bus voltage.

5. Unsolvable Case Figure 1-3.(c):

   There is no solution of the equations due to over loading of the system.

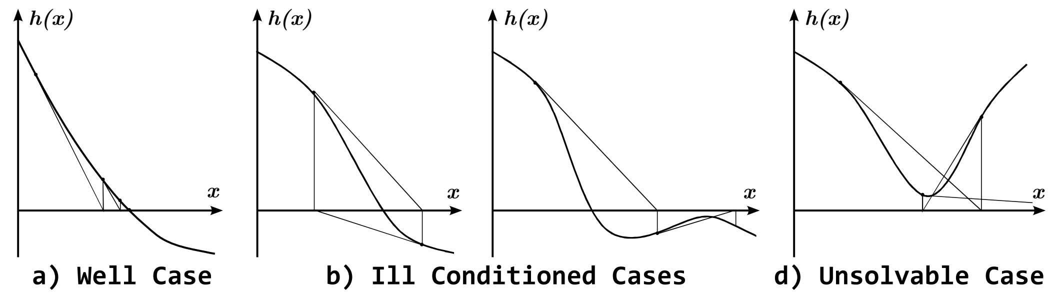


Figure 1-3: The example of convergence cases of Newton Raphson method [64]

1.8 Fast Decoupled Method

Fast decoupled method is based on Newton Raphson method after applying the following approximation conditions.

- The angle difference between adjacent buses is small (5 to 10 deg) :
  $\cos(\delta_i - \delta_j) \approx 1 \quad , \quad \sin(\delta_i - \delta_j) \approx 0$
- The impedance of transmission line is almost reactive : $G_{ij} \approx 0$
- The sensitivity of active power to bus voltage is almost zero : $\frac{\partial P_i}{\partial |V_j|} \approx 0$
- The sensitivity of reactive power to phase angle is almost zero : $\frac{\partial Q_i}{\partial \delta_j} \approx 0$

Finally the power flow equation can be decoupled into two systems of equations. The first one is $P\delta$ which defines the relation of active power injection with voltage angle at PQ and PV buses. The second system is QV which defines the relation between the reactive power at PQ buses and the voltage magnitude. The following equations demonstrate the essential Fast decouple system of active and reactive powers.

$$-\begin{bmatrix} \frac{\partial P}{\partial \delta} & 0 \\ 0 & \frac{\partial Q}{\partial V} \end{bmatrix} \times \begin{bmatrix} \Delta\delta \\ \Delta V \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \Longrightarrow \begin{array}{rcl} B^{\backprime} \times V \times \Delta\delta & = & \frac{\Delta P}{V} \\ B^{\backprime\backprime} \times \Delta V & = & \frac{\Delta Q}{V} \end{array} \tag{1.27}$$

Table 1.4 represents approximation methods used to calculate $B^{\backprime}$ and $B^{\backprime\backprime}$. Both $B^{\backprime}$ and $B^{\backprime\backprime}$ are constant. So that only one factorization is used for all iterations. The matrix size is reduced also which save the memory and reduce the iteration time even it requires more iterations than Newton Raphson method as shown in Table 1.5.

Method	$B^{\backprime}_{ij}$	$B^{\backprime}_{ii}$	$B^{\backprime\backprime}_{ij}$	$B^{\backprime\backprime}_{ii}$
BX	$(B_{ij}^2 + G_{ij}^2)/B_{ij}$	$-\sum B^{\backprime}_{ij}$	B_{ij}	$-2\,b_{i0} - \sum B^{\backprime\backprime}_{ij}$
XB	B_{ij}	$-\sum B^{\backprime}_{ij}$	$(B_{ij}^2 + G_{ij}^2)/B_{ij}$	$-2\,b_{i0} - \sum B^{\backprime\backprime}_{ij}$

Table 1.4: Fast decouple approximation methods [98]

IEEE Test Case	Newton Raphson	BX	XB
30	3	5	5
57	3	6	6
118	3	6	7

Table 1.5: Convergence of Fast decouple and Newton Raphson methods [89]

1.9 Optimal Power Flow Problem

The optimization problem searches for the minimum point of objective function $f(X)$ which satisfies the equality constrains $h(X)$ and it is bounded by limits of inequality constrains $g(X)$. OPF (Optimal Power Flow) is optimization problem which minimizes the cost function under the operation limits and network constrains. Equation 1.28 defines the mathematical model of OPF functions.

$$\begin{aligned} &min\{f(X)\} &&: \text{ Minimize the objective function} \\ &h_k(X) = 0 &&: \text{ Equality constrains functions } (k = 1 \cdots N_h) \\ &g_k(X) \leq 0 &&: \text{ Inequality constrains function } (k = 1 \cdots N_g) \end{aligned} \tag{1.28}$$

1.9.1 Karush-Kuhn-Tucker (KKT) Therom :

Karush-Kuhn-Tucker (KKT) is the first order necessary condition for a local minimum point as shown in the following equations. The (λ) is Lagrange multiplier and (μ) is KKT multiplier which must be zero for inactive constraints $(g(x) \leq 0)$.

$$\mu \geq 0 \quad \text{and} \quad \mu \cdot g(x) = 0 \tag{1.29}$$

$$\nabla f(x) + \lambda \cdot \nabla h(x) + \mu \cdot \nabla g(x) = 0 \tag{1.30}$$

1.9.2 Objective Cost Function

In many cases the OPF is used as economic dispatch to minimize the generation cost which may include the network losses. The cost function may be created by summation of many objective function with weights. The following factors are common to be included in the cost function:

• Network Losses [49]	• Generation Cost [27]
• Voltage Profile [26]	• Investment Cost [93]
• Control Shift [70]	• System Load ability [20]
• Voltage Stability [94]	• Load Shedding [7]

Table 1.6: The common factors of cost function for OPF

1.9.3 OPF Variables

OPF variables can be classified as state and control variables. The state variables represent the essential parameters of the system such as voltage angle and magnitude and they are normally continuous. The polar representation of voltage still be used since late 60's [27] until now [80]. However, voltage in rectangular coordinates with current injection instead of power has been used also as state variables [51]. The control variable is used to represent parameters of control device such as tap ratio of transformer or input as injected reactive power.

1.9.4 Equality Constrains

Equality constrains are used to represent the power mismatch equations at each bus. The full AC power flow is modeled by nonlinear constrains as defined in section 1.3. DC power flow uses linear approximation model to represent active power flow assuming unity voltage profile and no power losses as defined in Equation 1.31. Even this approximation is fast but it may produce unaccepted errors for large system.

$$\begin{gathered} P_i = \sum B_{ij} \cdot (\delta_i - \delta_j) \\ V \approx 1 \quad , \quad \sin(\delta_i - \delta_j) \approx (\delta_i - \delta_j) \end{gathered} \tag{1.31}$$

1.9.5 Inequality Constrains

Inequality Constrains are used to define the boundary limits of the power flow variables. The inequality constrain can be linear as bus voltage and generation power limits. The power limit of transmission line is an example of nonlinear constrain. Inequality constrain can be used also to define the boundary of static stability [94].

$$\begin{array}{ccccc} P_{min} & \leq & P_{gi} & \leq & P_{max} & \\ Q_{min} & \leq & Q_{gi} & \leq & Q_{max} & \text{Linear Constrains} \\ V_{min} & \leq & |V_i| & \leq & V_{max} & \\ \hline S_{min} & \leq & |V_i - V_j|^2 \cdot Y_{ij} & \leq & S_{max} & \text{Nonlinear Constrains} \end{array} \tag{1.32}$$

1.10 OPF Formulation

OPF formulation is based on approximation model used to represent the objective function and the constrains. The following methods are common be used to represent the OPF formulation :

1.10.1 Nonlinear Programming (NLP)

The first OPF formulation was defined by Carpentier (1962) based on nonlinear system of equations with continuous variables [17]. NLP gives accurate model of system characteristics but it requires extensive calculation and time. Discrete variables as transformer taps need to be approximated to continuous variables [79]. The variables can be classified into decision variables and state variables which are related to power flow equations. This method reduces the variables of OPF and may improve the convexity [91]. Computational improvements was achieved by redefining NLP formulation to reduce the degree of nonlinearity [48, 59, 8]. Lavaei and coworkers prove that when the matrix has a zero eigenvalue of multiplicity two, globally optimal solution of OPF can be obtained by solving dual problem. Each problem is convex. This case is applied on IEEE benchmark systems by adding small resistance to all transformers with zero resistance [56].

1.10.2 Linear Programming (LP)

Linear model of OPF is based on approximation of the original model which is nonlinear and non convex system of equations. The linear model is widely used in industry [74] because it has the following features :

- Convex system
- Well developed solver such as the Simplex Solver.
- Efficient handling of inequality.
- Easy to detect infeasibility.
- Fast Convergence.
- Guarantee a global optimal solution.

The basic model of Linear OPF is DC power flow with linear objective function. DC OPF can be solved with single step and it is extremely fast and robust compared with the real model. The OPF system can be also linearized at operating point [83]. This method is used as basic method for of SLP (Sequential Linear Programming). Although the LP is suitable for many cases, the simplification may cause unaccepted errors in the results. The global optimal point of LP many not a optimal point for real NLP and may not even feasible solution. So that LP many not be suitable for some applications such reactive power dispatch with minimizing the losses [95]. However, special algorithms are developed and implemented to obtain accepted solution and improve the LP accuracy.

1.10.3 Quadratic Programming (QP)

Quadratic programming is special case of NLP with quadratic objective function as define in Equation 1.33.

$$f(X) = \frac{1}{2}X^T \times Q \times X + q^T \tag{1.33}$$

If (Q) matrix is positive semidefinite [2], the problem become convex and its global optimal point is completely characterized by Karush-Kuhn-Tucker (KKT) conditions. Otherwise, if (Q) matrix is not positive semidefinite, there will not be guarantee for global optimal point and may be many local optimal points. QP improves the accuracy and convergence relative to LP specially when the objective function is quadratic such as generation cost. Glavitsch and Spoerry [33] (1983) represented incremental power flow using the rectangular coordinates based on non-sparse QP OPF formulation. Burchett (1982) implemented large scale sparse power flow model which can be used with sequential quadratic programming (SQP)[12]. Contaxis (1986) proposed an implementation for decoupled QP formulation suitable for SQP [24].

[2]Positive Semidefinite : All eigenvalues ≥ 0 and $X^T \times Q \times X \geq 0$

1.10.4 Mixed Integer Linear Programming (MILP)

Capitanescu and Wehenkel (2010) represents some heuristic techniques to model discrete variables but these methods are not entirely satisfactory. For example, rounding continuous variables to the nearest discrete value may lead to suboptimal or even infeasible solutions [15]. MILP is used with linearized system to represent discrete elements by integer variables. MILP formulations are often solved with sequential approach similar to SLP. MILP still has inaccuracy due to linear approximation of power flow equations. Lobato (2001) used MILP formulation to model capacitor control action [60]. Lima (2003) implemented optimal phase shifter placement with MILP formulation based on the DC power flow equations [58].

1.10.5 Mixed Integer Nonlinear Programming (MINLP)

MINLP is the most accurate formulation of OPF because it able to represent the nonlinear characteristics of the power flow equation and the uses integer variables to represent the actual state of discrete devices. However , it is complex mathematical model as tradeoff between accuracy and speed. Aminifar proposed OPF formulation based on MINLP for placement of UPFC [5]. Aouss Gabash proposed battery management system in distribution network based on OPF with MINLP formulation which optimizes the charge and discharge periods [31].

1.11 Deterministic Methods for Solving OPF

OPF solver classified as shown in Figure 1-4 into deterministic methods and intelligent methods. The intelligent are suitable for non deterministic or large cases when the conventional methods are not applicable. This section will focus on deterministic methods only because the they cover all required study cases.

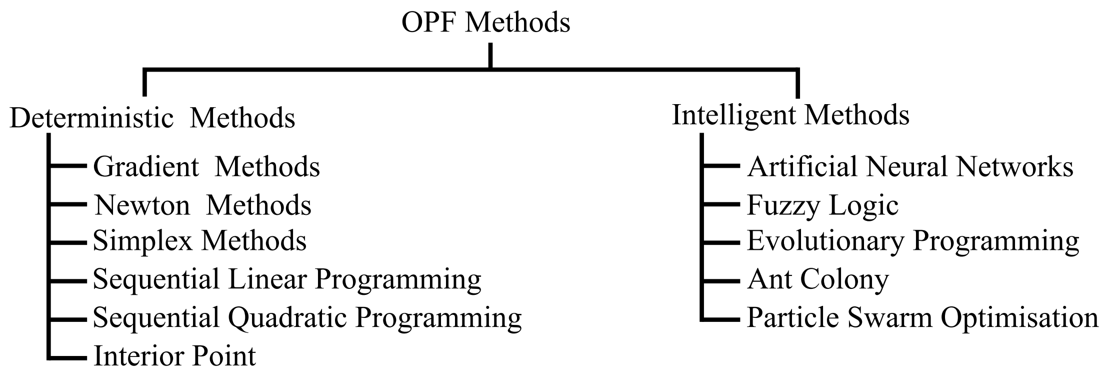


Figure 1-4: Methods for solving OPF problem

1.11.1 Newton Methods

Newton methods uses Hessian matrix $H(X)$ (second order derivative of objective function) to update the search direction (dx) as defined in Equation 1.34. The step size in search direction is selected to achieve the maximum improvement of objective function. The Newton methods have quadratic convergence curve but local optimum is not guaranteed unless Hessian matrix is positive semidefinite in optimal point. Newton methods use Lagrangian function with penalty terms for the constraints as define in Equation 1.34.

$$L(X) = f(X) + \sum \lambda_n \cdot h_n(X) + \sum \mu_m \cdot g_m(X) \tag{1.34}$$

$$dx = -H(X)^{-1} \times \nabla L(X) \tag{1.35}$$

$$X(k+1) = X(k) + \alpha \cdot dx \tag{1.36}$$

First Newton based OPF was presented by Sasson (1973) with heuristically computed penalty factors without Lagrangian function [77]. The active inequality is a major problem for Newton based algorithms because it is not known prior to the solution. Active set and penalty (ASP) method was proposed by Sun (1984) for relaxation and enforcement inequality constraint based on Lagrangian with heuristic scheme [84]. Quasi Newton methods approximate the Hessian matrix to reduce the calculation for each iteration. Housos and Irisarri (1982) reviewed the two important methods for OPF with Quasi Newton Broyden Fletcher Goldfarb Shanno (BFGS) method and Davidon Fletcher Powell (DFP) method [45].

1.11.2 Gradient Methods

Gradient methods use the first derivative of the objective function as defined in Equation 1.37. The gradient improves the search direction but it is slower than other methods based on second order derivative. The solution may not be local optimal point because the gradient methods search for stationary point ($\nabla f(X) = 0$) .The global optimization is not applicable for non convex problems which the case of OPF.

$$\nabla f(X) = \begin{bmatrix} \frac{\partial f(X)}{\partial x_1} & \frac{\partial f(X)}{\partial x_2} & \cdots & \frac{\partial f(X)}{\partial x_n} \end{bmatrix} \tag{1.37}$$

- Reduced Gradient (RG) method :

  The RG method was proposed by Wolf (1967) to solve NLP problems with linear constraints [90]. Dommel and Tinney (1968) used RG method to solve OPF by adding penalty terms to objective function to enforce the constraint [27]. The RG method has Zig-Zag search characteristic near solution point.

- Conjugate Gradient (CG) method :

  The CG method is the most well known iterative methods for solving NLP problems with sparse systems of linear equations. CG method update the search direction based on vector of scaled linear combination of current gradient with previous search directions. This method reduces the number of iteration and helps to avoid Zig-Zag searching because it provides nonzero search direction and linearly independent of all previous directions vectors. Burchett (1982) demonstrated the advantages of CG for OPF relative to RG [13].

- Generalized Reduced Gradient (GRG) method :

  Abadie and Carpentier (1969) propose GRG method to enable direct treatment of inequality and nonlinear constraint [2]. Nonlinear constraints can be linearized at operating point to create sub problems with linear constraints which can solved by RG or CG method. Feasibility recovery is done at end of each iteration to correct the approximation error introduced by linearization. Peschon (1972) demonstrated the first OPF with GRG method and its benefits of avoiding of penalty terms.

1.11.3 Simplex Method

The simplex method is the most robust and efficient method for LP cases such as DC OPF formulation. It can find optimal point systematic even it takes exponential time. Incremental linear model is proposed to OPF via small change around a based point. Stott and Hobson (1978) compared DC-OPF vs. OPF using incremental linear models and implemented SCED (Security Constrained Economic Dispatch) [82]. The simplex solver is also used for many SLP algorithms for OPF.

1.11.4 Sequential Linear Programming

SLP is used to optimize nonlinear function by series of linear approximations at each solution point until convergence [39]. Trust Region is used to restrict the search region because SLP can not get optimum if the linearization leads to unconstrained direction [9]. OPF with SLP is iterative processes between conventional power flow and linearized LP sub problems [96]. SLP provides the speed of LP and the accuracy of NLP [4]. However , SLP may have oscillation near optimum due to linearization [75]. It may cause divergence for highly nonlinear function [43].

1.11.5 Sequential Quadratic Programming

SQP approximates NLP problem to series of QP problems for every solution point until convergence [9]. SQP is more efficient than NLP but it may have oscillations near the optimal solution [75]. The onventional power flow is normally used to linearize the constraints then the QP is solved by deterministic optimization method. Burchett (1982) used gradient method combined with Simplex-like method to solve the QP sub problems [13, 11]. Chang introduced many heuristic techniques on SQP using Newton's method to improve algorithm performance [18]. Grudinin formulated SCED based on extended SQP with quadratic separable algorithm for reactive power optimization [42, 43]. Lehmkoster (2002) included FACTS device models to minimize the cost under constraints based on SQP [57].

1.11.6 Interior Point Methods (IPMs)

IPMs are used to solve LP and NLP by adding barrier terms to the objective function to get optimal solution following a central trajectory inside the feasible region. The IPM was proposed by Karmarkar as alternative method for Simplex[53]. IPMs uses variant of Newton's method for NLP problems to satisfy the KKT conditions by estimating the appropriate value of Lagrange multipliers, decision variables, slack variables [69]. New IPMs are improved to be easy to handle nonlinear constrains directly , fast convergence and able to find the solution even with starting out of the feasible region [16]. The following methods are based on IPMs :

- Primal-Dual Interior Point Methods (PDIPM) :
  PDIPM is based on solving LP or NLP for primal , dual and slack variables using barrier terms which represent KKT conditions [69]. PDIPM was used by Granville (1994) to solve NLP OPF of reactive power dispatch problem [37].
- Predictor Corrector (PDIPM) :
  Predictor corrector method was proposed by Mehrotra to improve search direction of PDIPMs [61]. First, prediction find the greatest improvement to optimality using Newton direction with neglecting the bounds. Then, the correction restores centrality by estimating the proper value for the barrier parameters.
- Multiple Centrality Corrections (MCC-PDIPM) :
  Gondzio proposed heuristic method to apply repeated correction which reduce the difference in complementarity product [36]. Torres and Quintana (2001) extended this method to NLP OPF [87]. MCC method accelerates the convergence by improving the step lengths at each iteration.
- Trust Region Interior Point Methods (TRIPM) :
  TRIPM limits the search at each iteration to be within region where the approximation is valid. This method can be use to solve problems with nonlinearity, nonconvexity, and ill-conditioning but it required extreme processing. Min and Shengsong (2005) implemented a trust region method with MCC-PDIPM based on SLP to solve OPF [67].

1.11.7 Comparison of Deterministic Methods for OPF

Method	Ref.	Conv.[1]	Description
RG	[27]	Global	Inequality constraints are satisfied by penalty functions. Slow convergence due to ”Zig-Zag” search.
CG	[12]	Global	Overcomes ”Zig-Zag” search problem of RG. But it still need penalty terms.
GRG	[71]	Local	Avoid penalty terms and simplify sensitivity analysis by using. successive linearization and slack variables.
Newton	[84]	Local	It has quadratic convergence but requires penalty terms. Used as local solver in many other methods.
Quasi-Newton	[45]	Local	Keep Jacobin matrix constant for some iteration. Reduce the calculation but it is not widely used for OPF
Simplex	[82]	Global	Used for DC OPF or as local solver in SLP. Best option for speed in many applications. Low accuracy due to linearization of OPF equations.
SLP	[4]	Local	Widely used by commercially OPF for economic dispatch. Certain cases may have instability and oscillation. Not good option for reactive power dispatch.
SQP	[12, 19]	Local	Faster than SLP for many OPF formulations. Used as alternative to IPMs in many commercial OPF packages.
IPM	[88]	Local	The fastest and most efficient deterministic algorithms. Many researches to select parameters and assure convergence.
TRIPM	[81]	Global	Assure convergence for highly nonlinear and unstable cases. Works when IPMs fails but at lower speed.

Table 1.7: Comparison of Deterministic Methods for OPF [30]
[1] Conv. : Convergence Domain

1.12 Free Software for Power Flow Analysis

Open source tools helps to extend the functionality and fix problems by collaboration of researcher to share their ideas and model. Table 1.8 lists the important open source tools for power flow analysis which can be classified as the following :

- Native Language : C , Fortran ,...

  Native languages provides the best option for performance but it needs different compiler for each platform and additional libraries for mathematical operation.
- Scripting Languages : Python , Java ,...

  This option supports many platforms but at it slower because it is based on interpreters. Compiled libraries and JIT (Just in Time Compilation) are used to speed up the execution. Additional libraries are still required but the integration is much easier than native languages.
- Scientific Scripting Languages : MATLAB , Octave , Modelica , ...

  Complete frame work of libraries for different platforms with huge documentation and supported by wide scientific research communities. Acceleration option can be used to compile the script to machine language for high performance. This is the best option for portability , functionally and performance.

MATPOWER is MATLAB functions for power flow analysis. It has been used by many researchers because it is implemented to be easy for modification at best performance and it also includes many test cases which can be used for verification.

Software	Year	Language	Functions	Cite	Ref.
UWPFLOW	1966	C	Power flow and Voltage stability	40	[14]
MATPOWER	1997	MATLAB	Power flow and OPF	516	[99]
PSAT	2002	MATLAB	Power system analysis toolbox	392	[62]
InterPSS	2006	Java	Power Flow	20	[97]
DCOPFJ	2007	Java	DC optimal power flow	131	[85]
OpenDSS	2009	Delphi	Power flow in distribution network	5	[28]
Minpower	2011	Python	OPF	2	[38]
Dome	2011	Python	Power system analysis	4	[65]
PowerSystem	2012	Modelica	Power analysis library	1	[23]
GridLAB-D	2012	C++	Power flow for distribution network	65	[21]

Table 1.8: Open source software for power flow analysis [66]

Chapter 2

Mathematical Model

Contents

2.1 Symbol Notation

The following symbols notation are used to represent the mathematical model

Symbol	Description
N	Total number of system buses
M	Total number of system lines
i	Source bus index
j	Destination bus index
k	Line index
J	$\sqrt{-1}$
S_i	Apparent power injected at bus i (generation - load)
P_i	Real power injected at bus i (generation - load)
Q_i	Reactive power injected at bus i (generation - load)
V_i	The complex voltage at bus i
δ_i	Voltage angle at bus i
CS	Connection matrix for sending bus
CR	Connection matrix for receiving bus
a	Complex transformer ratio
Γ	Modified incident matrix $M \times N$
G	The conductance
B	The susceptance
y	The admittance of line
y_S	The admittance at sending bus
y_R	The admittance at receiving bus
V_S	The complex voltage at sending bus
V_R	The complex voltage at receiving bus
I_{SR}	The complex line current injected from sending bus
I_{RS}	The complex line current injected from receiving bus
I_S	The complex current injected from sending bus
I_R	The complex current injected from receiving bus
I_{Sg}	The complex current injected from sending bus to ground
I_{Rg}	The complex current injected from receiving bus to ground
$(\ )^*$	Complex conjugate
$(\ )^{*T}$	Complex conjugate and transpose of matrix
$(\ )^T$	Matrix transpose without complex conjugate
$[\ ]$	Diagonal matrix of column vector

Table 2.1: Symbol notation list

2.2 PI Circuit Model of Transmission Line

The AC transmission line mathematical model is based on telegraph equations which are set of partial differential equation. Lumped PI circuit shown in Figure 2-1 is used as approximated model for the steady state solution of telegraph equations. The parameters of PI model can be calculated based on line configuration which include the conductors parameters (length , cross section , resistance,...) and tower geometry (height , separation distances,...). The series impedance represents the total line resistance and inductance. The shunt admittance is used to model the leakage current between the line and ground and the corona effect. If the line is symmetry the shunt admittance at both sending and receiving buses will be equal. In many cases , the shunt admittance is simplified to be the half of the total charging susceptance ($y_S = y_R = B$) of the line. The following equation demonstrate the relation between the line currents based on the voltages at sending bus and receiving bus [34].

$y = 1/(R + J \cdot X)$	: Line admittance
$y_S = (B_S + J \cdot G_S)$	: Admittance to ground at sending bus
$y_R = (B_R + J \cdot G_R)$	: Admittance to ground at receiving bus
$I_S = I_{SR} + I_{Sg} = y \cdot (V_S - V_R) + y_S \cdot V_S$	: Line current at sending bus
$I_R = I_{RS} + I_{Rg} = y \cdot (V_R - V_S) + y_R \cdot V_R$	: Line current at receiving bus

$$\begin{bmatrix} I_S \\ I_R \end{bmatrix} = \begin{bmatrix} (y + y_S) & -y \\ -y & (y + y_R) \end{bmatrix} \times \begin{bmatrix} V_S \\ V_R \end{bmatrix} \tag{2.1}$$

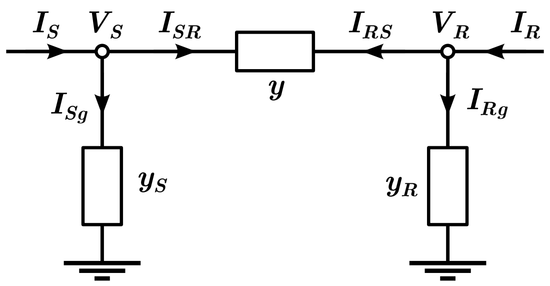


Figure 2-1: PI Circuit model of transmission line

2.3 Transformer Model

The transformers are used to connect networks with different voltage level. Per unit system uses different based voltage for each level and normalize the buses voltages to unity. So that, transformers with fixed ratio are modeled by per unit system normalization. The variable transformers can be classified in two main types [6]:

- Load Tap Changer Transformer (LTC):
  LTC transformer is used to compensate the voltage droop due to over load on the network by regulating the voltage or the reactive power through the line.
- Phase Shift Transformer (PST) :
  PST introduces a phase shift between the two buses to control the active power injection through the line.

The ideal transformer with complex ratio shown in Figure 2-2 can be used to model LTC and PST or both of them at same time. The following equations represent this mathematical model based on input and output voltage and current:

$S_1 = V_1 \cdot I_1^*$	: Input apparent Power
$S_2 = V_2 \cdot I_2^*$	: Output apparent Power
$S_1 = S_2$	: Power balance between input and output power
$a = \|a\|\, e^{J\varphi}$	: Complex transformer ratio
$\|a\|$	: Tap ratio
φ	: Phase shift
$V_2 = a \cdot V_1$	: Output voltage
$I_2 = I_1^*/a$	: Output current

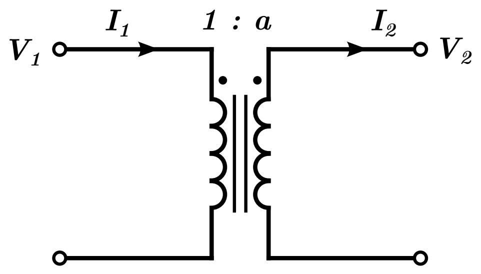


Figure 2-2: Ideal transformer model

2.4 Transmission Line with Transformer Model

Figure 2-3 shows the general model of transmission line including variable transformer at sending bus. This circuit model can be simplified by modeling the (y_S) at the sending bus as shown in figure Figure 2-3.b. Other simplified model may be used by assuming that $(\,|a|^2 \approx 1)$. The following equations defines the line current relative to sending and receiving buses [6] [40].

$$\begin{aligned}
V &= a \cdot V_S - V_R && : \text{ Voltage droop across the line} \\
I &= y \cdot V = y\,(a \cdot V_S - V_R) && : \text{ The line current} \\
I_{SR} &= a^* \cdot I = y \cdot \left(|a|^2 \cdot V_S - a^* \cdot V_R\right) && : \text{ The line current relative to sending bus} \\
I_{RS} &= -I = y\,(V_R - a \cdot V_S) && : \text{ The line current relative to receiving bus}
\end{aligned}$$

$$\begin{bmatrix} I_{SR} \\ \\ I_{RS} \end{bmatrix} = y \cdot \begin{bmatrix} |a|^2 & -a^* \\ \\ -a & -1 \end{bmatrix} \times \begin{bmatrix} V_S \\ \\ V_R \end{bmatrix} \tag{2.2}$$

Equation 2.2 is single line model of admittance matrix which has three variable elements depends on the transformer ratio. This required to update admittance matrix in three location for each iteration. The full admittance matrix will integrate all lines parameters to be used for calculating the injected current at each bus. So that there is no way to extract the single line current because the admittance matrix aggregate all parallel line to be as single line. The following section will explain another method to replace the admittance matrix.

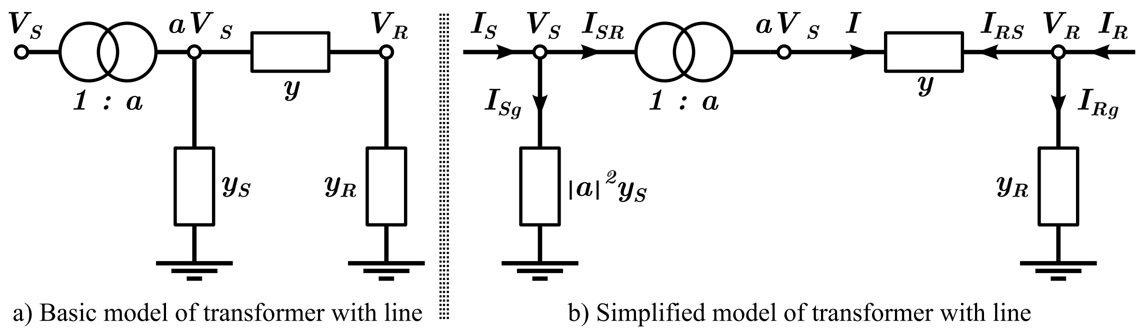


Figure 2-3: General model of transmission line with transformer

2.5 Incident Matrix Model of Transmission Line

The electric network can be represented as directed graph. The incident matrix is used to define any directed graph as following. Each row of incident matrix represents single line which include only 1 for sending bus and -1 for receiving bus. Each column includes nonzero elements which represent the direction of each line connected to the bus of this column as defined in Equation 2.3 [89]. The transmission line model can be fully defined by two equations. The first one is KVL (Kirchoff Voltage Law) at each line as shown in Equation 2.4. The second one is KCL (Kirchoff Current Law) at each bus as shown in Equation 2.5.

$$m(k,i) = \begin{cases} 1 & \text{bus } i \text{ is sending bus of line } k \\ 0 & \text{bus } i \text{ is not connected to line } k \\ -1 & \text{bus } i \text{ is receiving bus of line } k \end{cases} \tag{2.3}$$

$$I = y \cdot \begin{bmatrix} 1 & -1 \end{bmatrix} \times \begin{bmatrix} V_S & V_R \end{bmatrix}^T \qquad \text{KVL of the line} \tag{2.4}$$

$$\begin{bmatrix} I_{SR} & I_{RS} \end{bmatrix}^T = \begin{bmatrix} 1 & -1 \end{bmatrix}^{*T} \times I \qquad \text{KCL at each bus} \tag{2.5}$$

This model can be extended to represent transmission line with transformer at sending bus. The incident matrix need to be modified to include the transformer ratio as shown in Equation 2.6. The (T) operator in KVL indicates the normal transpose without any modifications. The $(*T)$ operator in KCL indicates the complex conjugate transpose which is essential to represent the phase shift transformer.

$$\Gamma(k,i) = \begin{cases} a & \text{bus } i \text{ is sending bus of line } k \\ 0 & \text{bus } i \text{ is not connected to line } k \\ -1 & \text{bus } i \text{ is receiving bus of line } k \end{cases} \tag{2.6}$$

$$I = y \cdot \begin{bmatrix} a & -1 \end{bmatrix} \times \begin{bmatrix} V_S & V_R \end{bmatrix}^T \qquad \text{KVL of the line} \tag{2.7}$$

$$\begin{bmatrix} I_{SR} & I_{RS} \end{bmatrix}^T = \begin{bmatrix} a & -1 \end{bmatrix}^{*T} \times I \qquad \text{KCL at each bus} \tag{2.8}$$

2.6 Connection Matrix

The connection matrix are use to relate two variables defined in different objects. For example , the voltage at both end of the line can be defined relative the bus voltage matrix using connection matrices (CS) and (CR). Each row of connection matrix represents a single line and only the selected bus column is set to 1 while other element are cleared to zero as defined in Equation 2.9 and Equation 2.10. The connection matrices of the line can be used also to build the modified incident matrix as defined in Equation 2.13 [99]. Figure 2-4 shows a simple example of connection matrix to represent a network with four nodes and five lines

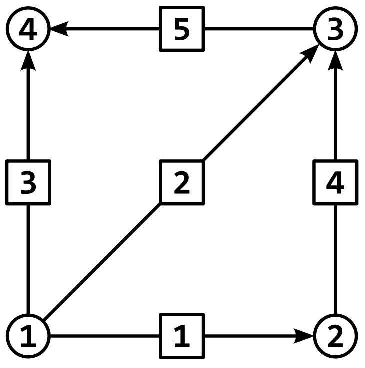
$$CS(k,i) = \begin{cases} 1 & \text{bus } i \text{ is the sending bus of line } k \\ 0 & \text{Otherwise} \end{cases} \tag{2.9}$$

$$CR(k,i) = \begin{cases} 1 & \text{bus } i \text{ is the receiving bus of line } k \\ 0 & \text{Otherwise} \end{cases} \tag{2.10}$$

$$V_S = CS \times V \qquad \text{The voltage at sending bus} \tag{2.11}$$

$$V_R = CR \times V \qquad \text{The voltage at receiving bus} \tag{2.12}$$

$$\Gamma = [a] \times CS - CR \qquad \text{The modified incident matrix} \tag{2.13}$$



$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{CS} - \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{CR} = \underbrace{\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}}_{\Gamma}$$

Figure 2-4: Example of creating incident matrix based on connection matrix

2.7 The Mathematical Model of Electric Load

The load can be defined by ZIP (Impedance , Current , Power) as following [64] [89]:

- Constant Power Load : Figure 2-5.a

  This type consumes fixed amount of power regardless of its voltage. The load power is included in the injection power equation.

- Constant Current Load : Figure 2-5.b

  The current of this load is independent of load voltage. This current can be added to the injection current of the bus.

- Constant Impedance Load : Figure 2-5.c

  The load current is linear proportional to its voltage. This type of load can be model as shunt admittance to the ground.

The following equations define general load model based on polynomial function [29]:

$$P = P_0 \left(\frac{V}{V_0} \right)^m \tag{2.14}$$

$$Q = Q_0 \left(\frac{V}{V_0} \right)^m \tag{2.15}$$

$$m = \begin{cases} 0 & \text{Constant Power Load} \\ 1 & \text{Constant Current Load} \\ 2 & \text{Constant Impedance Load} \end{cases} \tag{2.16}$$

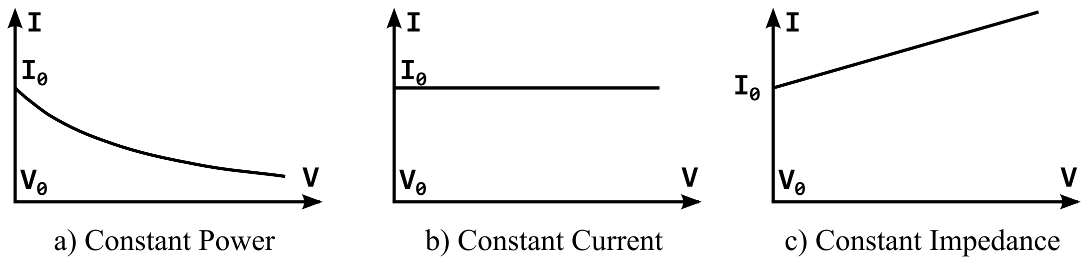


Figure 2-5: The electrical load types

2.8 Droop Model of Active Power Generation

The droop characteristic of generator represents the frequency deviation relative to variation in active power demand. Droop control is emulated in new inverter to achieve load sharing among parallel connected inverters. The following equation demonstrate the droop characteristic as shown in Figure 2-6 [64].

$$P_g = P_0 - d \cdot (f - f_0) \qquad \text{Active power} \tag{2.17}$$

$$d = \Delta P / \Delta f \qquad \text{Droop slope (MW/Hz)} \tag{2.18}$$

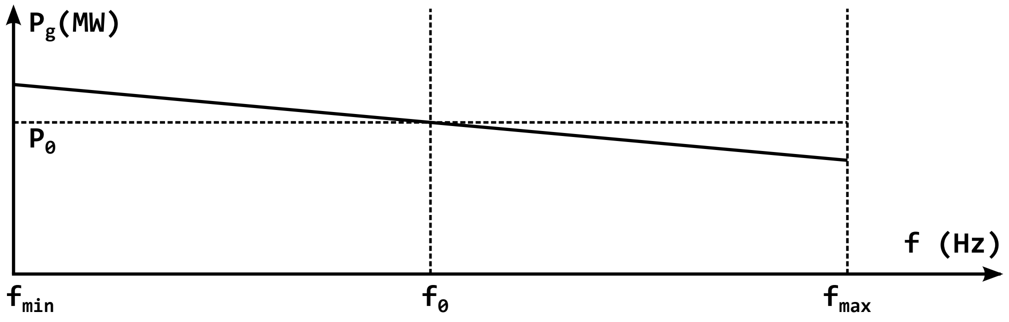


Figure 2-6: The generated active power relative to frequency deviation

2.9 Objective Function

The objective function represents the cost of the system which depends on the investment and running cost. The running cost includes the generation , maintenance and losses costs. Each generator has cost function for active and reactive power. This cost function can be approximated as polynomial form as defined in Equation 2.19 and Equation 2.20. The losses can be defined as linear function of generation and load as shown in Equation 2.21 and Equation 2.22 [98].

$$f_{Pg} = \sum_{k=0}^{n} a_k \cdot P_g^k \qquad \text{Cost function of active power} \tag{2.19}$$

$$f_{Qg} = \sum_{k=0}^{n} b_k \cdot Q_g^k \qquad \text{Cost function of reactive power} \tag{2.20}$$

$$f_{PL} = c \cdot (P_g - P_L) \qquad \text{Cost function of active power losses} \tag{2.21}$$

$$f_{QL} = c \cdot (Q_g - Q_L) \qquad \text{Cost function of reactive power losses} \tag{2.22}$$

2.10 FACTS Device Modeling

FACTS (Flexible AC Transmission Devices) are used to improve the overall system performance and capabilities. Old FACTS devices are based on switching reactors and capacitors to emulate variable impedance. New FACTS devices use voltage source inverter based on PWM switching with current control to achieve smooth and fast response [3] . The FACTS devices can be classified as following :

- Shunt Devices :

  This device is connected to the bus to regulate the bus voltage or add reactive power support. The shunt device can be defined by the bus number and the reactive power injection limits. The mathematical model of the shunt device can be simplified as generator with zero active power and limited reactive power. The shunt device extends the reactive power limits of PV bus and convert PQ bus to be as PV bus because it can regulate the bus voltage [73] [54]. The shunt device can be used to model the following devices :

  - STATCOM (Static Synchronous Compensator) : Voltage source type
  - SVC (Static Var Compensator) : Variable impedance type

- Series Devices :

  The series device is used to improve the power transmission capabilities by compensating the line impedance. The series device can be defined with line index and the compensation factor. Equation 2.23 defines the line impedance base on the compensation factor [35]. The series device can be used to model the following devices:

  - SSSC (Static Synchronous Series Compensator) : Voltage source type
  - TCSC (Thyrister Controlled Series Capacitor) : Variable impedance type

$$z = R + JX \cdot (1 - k) \qquad : \text{Line impedance wiht compansation} \tag{2.23}$$

$$0 \leqslant k \leqslant 0.3 \qquad : \text{Compansation factor} \tag{2.24}$$

Other types of FACTS devices can be modeled with combination of series and shunt devices or with variable transformer model.

2.11 Power Flow Equations with Incident Matrix

The following equations summarize the power flow formulation based on the explained mathematical model. The operator [ ] is used to convert single column vector to be diagonal matrix. The dot operator is used to represent element by element multiplication [99].

$$\Gamma = [a] \times CS - CR \qquad \text{Modified incident matrix } (M \times N) \tag{2.25}$$

$$I_L = [y] \times (\Gamma \times V) \qquad \text{Line current } (M \times 1) \tag{2.26}$$

$$S_L = [y] \times |\Gamma \times V|^2 \qquad \text{Line losses } (M \times 1) \tag{2.27}$$

$$S_{SR} = a \cdot V_S \cdot I_L^* \qquad \text{Power injected from sending bus } (M \times 1) \tag{2.28}$$

$$S_{RS} = -V_R \cdot I_L^* \qquad \text{Power injected from receiving bus } (M \times 1) \tag{2.29}$$

$$Y_S = CS^T \times \left(|a|^2 \cdot y_S\right) \qquad \text{Admittance at sending bus } (N \times 1) \tag{2.30}$$

$$Y_R = CR^T \times y_R \qquad \text{Admittance at receiving bus } (N \times 1) \tag{2.31}$$

$$Y_n = Y_S + Y_R + Y_g \qquad \text{The total admittance to ground } (N \times 1) \tag{2.32}$$

$$Y = \Gamma^{*T} \times ([y] \times \Gamma) + [Y_n] \qquad \text{Admittance matrix } (N \times N) \tag{2.33}$$

$$I_n = I_L \times \Gamma^* + Y_n \cdot V \qquad \text{Injection current at each Bus } (N \times 1) \tag{2.34}$$

$$S_n = V \cdot I_n^* \qquad \text{Injection Power at each bus } (N \times 1) \tag{2.35}$$

Chapter 3

Solver Implementation

Contents

3.1 Introduction

`PF_Solver` is power flow solver based on Matlab functions. The solver is implemented as single Matlab function file which includes the main function and other sub functions. All functions uses shared data storage using nesting function method to reduce the access time and improve the performance. Data variables are organized in structures to be encapsulated and protected from corruption due to name conflict. Variables of the main function are shared with other nested functions but variables of each nested function are private. Private variable can be used by its function only. Figure 3-1 shows data shearing based on nested function organization.

The solver functions are classified in layers. The outer layer functions interact with external world to load data , run the solver and save the results. The inner layer functions represents the mathematical model of the system and internal variables handling. In many cases, the function has no inputs or outputs variables. The function reads input data from shared structure and then it modifies the results in other shared structure. This way reduces the time for passing input and output data between functions. Some function has single output variable (`errFlg`) to indicate if there is an error or not.

The solver configuration and other global data are based on setting parameters stored in shared structure which can be saved and loaded from file. This makes the solver easy to be customized and used for different cases without changing the software instructions.

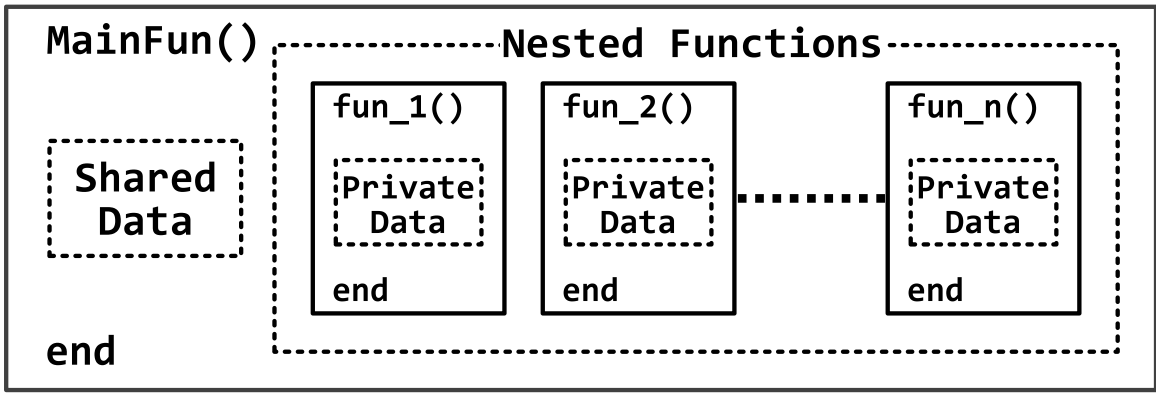


Figure 3-1: Shared data by nested function organization

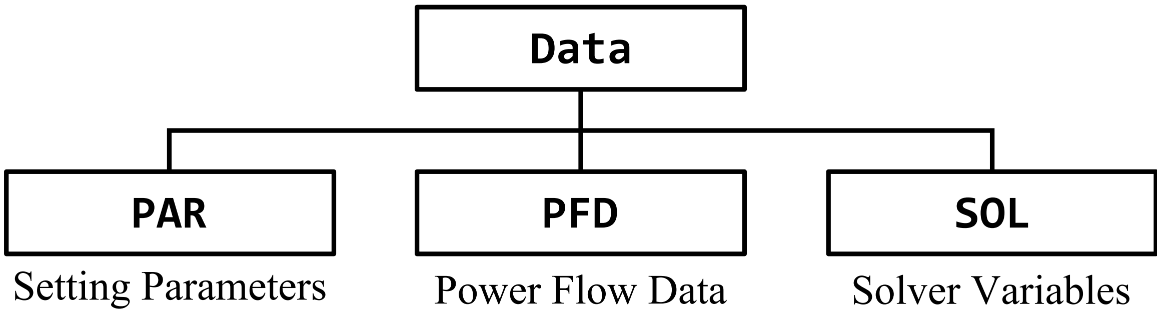


Figure 3-2: Solver data organization

3.2 Solver Data Organization

Solver data are classified into three different structures shown in Figure 3-2. The following sections will explain each data structure.

- **PAR** : The setting parameters.
- **PFD** : The grid elements (bus , generator, line , transformer) data.
- **SOL** : The solver variables and result.

3.3 Setting Parameters : PAR

The setting parameters include solver configuration parameters and global data limits. Each parameter represents single value stored in shared structure named `PAR`. This structure is created by the main function `PF_Solver()` and initialized by `PF_InitPar()` function. The setting parameters can be saved as text format in (***.ini**) file with `PF_SavePar()` function and loaded agin by `PF_LoadPar()` function. The parameter file is defined by string variable stored in `PAR.FileName`. The parameter file format is defined based on the following rules :

- Single parameter per line : `ParName = ParValue`
- Parameter name (`ParName`) is case sensitive
- Parameter value (`ParValue`) must be numeric
- No restriction on parameter order
- No restriction on white spaces
- Comment line must starts with % or //

The parameter file can be edit by any text editor to change the parameters values as required. Single parameter file can be used with different cases or individual parameter file for each case. Parameter file may include only few parameters which are different from the default values. The parameter value can be double , integer, or Boolean. All parameters are stored as double because it can support all other types. For Boolean parameters, zero value represents the False condition and any other value will represent True condition. Some integer parameters must be positive or zero like `DevCaseIndex`. Other integer parameters can be positive or negative. The parameter type and boundaries should be taken into account while editing the parameters file. The setting parameter can be classified based of the functionality as following :

- **Solver Configuration :** `SolverType` **,** `UseYbus` **, ...**
- **Case Parameters :** `BaseFreq` **,** `BasePower` **,** `DevCaseIndex` , **...**
- **Constrains Parameters :** `Limit_Pg` **,** `Pg_Min` **,** `Pg_Max` **, ...**
- **OPF Configuration :** `Optimal_Pg` **,** `Optimal_Tmag` **, ...**

Table 3.1 shows the description of solver configuration and case parameters. The global parameters are used when this data is not defined in input file. For example, the `BaseFreq` is not defined in most of input file format.

Parameter	Description
`BasePower`	Base Power (MVAR)
`BaseFreq`	Base Frequency (Hz)
`SolverType`	0 : Normal , 1 : Optimal
`UseYbus`	0 : Incident matrix , 1 : Admittance matrix
`UseDroop`	1 : Use droop control for Active power dispatch
`StartPoint`	0 : Flat start , 1 : Last solution
`CostFun`	0 : Active Losses , 1 : Generation Cost , 2 : Voltage profile
`Tol_X`	Maximum tolerance of X
`Tol_Con`	Maximum tolerance of constraint
`Tol_Fun`	Maximum tolerance of objective function
`Max_Iter`	Maximum of iteration count
`DevCaseIndex`	Selected deviation case index for solving single case mode
`FirstCaseIndex`	First index of deviation case for sequential solve mode
`LastCaseIndex`	Last index of deviation case for sequential solve mode
`Limit_OFF`	1 : Disable all limits

Table 3.1: Solver configuration and case parameters names and descriptions

Basic Power Flow	Optimal Power Flow	Power Flow with Droop
`SolverType` = 0	`SolverType` = 1	`SolverType` = 0
`UseDroop` = 0	`UseDroop` = 0	`UseDroop` = 1

Table 3.2: Solver Modes

The solver configuration parameters are critical because they control the whole program execution process. The system equations will be built based on the solver configuration parameters and OPF parameters. Some parameter are not included for a given mode. For example , `UseDroop` will not be used during optimal power flow. Table 3.2 shows the basic solver modes defined based on configuration parameters.

The constraints parameters define the maximum and minimum limits of variables used with OPF. The input file should include limits of voltage and power but in some case this data is not included. So that , the configuration parameter can be used select which limit is applied or disabled. OPF settings define if the variable is optimized or not. Table 3.3 lists short abbreviation parts which is used to build the setting parameter names. Table 3.4 shows full list of all available constraints parameters. The frequency limits are used only to detect the frequency violation during active power dispatch based on droop control.

Abbreviation	Description
`Min`	Minimum
`Max`	Maximum
`Pg`	Generated active power (MW)
`Qg`	Generated reactive power (MVAR)
`S`	The line apparent power (MVA)
`V`	Bus voltage magnitude (pu)
`Kc`	Compensation factor of series device (pu)
`Qsh`	Reactive power of shunt device (MVAR)
`Tmag`	Tap ratio of transformer (pu)
`Tang`	Phase shift of tap transformer (deg)
`Limit`	0 : No Limit , 1 : Use input limits , 2 : Use global limits
`Optimal`	0 : Use fixed value , 1 : Find optimal value for minimal cost

Table 3.3: Setting parameters abbreviation

Par.	Description
Optimal_Pg	Optimize the generation power
Optimal_Vmag	Optimize the voltage of PV and slack bus
Optimal_Tmag	Optimize the Tap Ratio
Optimal_Tang	Optimize the Tap Angle
Optimal_Qsh	Optimize the shunt device
Optimal_Kc	Optimize the series device
Limit_Freq	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_V	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_S	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Pg	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Qg	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Tmag	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Tang	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Kc	0 : No Limit , 1 : Input limits , 2 : Global limits
Limit_Qsh	0 : No Limit , 1 : Input limits , 2 : Global limits
FreqErr_Min	Minimum percent of error in frequency (%)
FreqErr_Max	Maximum percent of error in frequency (%)
V_Min	Mimimum voltage of the bus (pu)
V_Max	Maximum voltage of the bus (pu)
S_Max	Maximum power of the Line (MVA)
Pg_Min	Mimimum active power of the bus (MW)
Pg_Max	Maximum active power of the bus (MW)
Qg_Min	Mimimum reactive power of the bus (MVAR)
Qg_Max	Maximum reactive power of the bus (MVAR)
Tmag_Min	Mimimum tap transformer ratio (pu)
Tmag_Max	Maximum tap transformer ratio (pu)
Tang_Min	Mimimum tap transformer angle (Deg)
Tang_Max	Maximum tap transformer angle (Deg)
Qsh_Min	Shunt mimimum Q (MVAR)
Qsh_Max	Shunt maximum Q (MVAR)
Kc_Min	Series mimimum Kc (pu) 0.0
Kc_Max	Series maximum Kc (pu) 0.4

Table 3.4: Constrains and OPF parameters

3.4 Power Flow Data : PFD

PFD is used as common interface between different data files format and the solver engine. The input data files are loaded and converted to PFD format. Then the solver uses the PFD to generate the results. PFD can be updated by results of solver. New results can be stored in PFD and can be saved as any supported data format. So that, PFD can be used to convert between different data files formats even without running the solver. The PFD include the following data:

- `BasePower` : Base power used for pu system (MVA)
- `BaseFreq` : Base frequency (Hz)
- `DevPar` : Deviation parameters
- `Bus` : Bus data
- `Line` : Transmission lines data
- `Gen` : Generators data
- `Trans` : Transformers data
- `Shunt` : Shunt device
- `Series` : Series device

The following section will explain each data structure. The solver depends on bus data and line data as essential input data to create the system of equations which represent the network. The input data can be reduced and rearranged to be suitable for certain data file format. This conversion may not be reversible and some data may be removed. The data can be classified in the following groups :

- Basic Power Flow Data Type:

  It is required for all modes of the solver and supported by all data files format.
- Optimal Power Flow Data Type:

  It is required for OPF and it may not be supported by other data files format.
- Active Power Droop Control Data Type:

  It is used only by the solver and it is not supported by any data file format.
- Deviation Case Data :

  It is used only by solver but it can be converted to other data files format.

3.4.1 Bus Data : `PFD.Bus`

Table 3.5 shows the field names of bus data. It is selected to support different data file formats . Some variables can be calculated from another one. For example, (`GL`) can be calculated from (`G`). The load data functions will be responsible to calculate any missing variables or load them from global parameters. Some variables are not essential for solver but they are required for data representation like `PFD.Bus.Name`. Each variable represents single column vector with `PFD.Bus.Count` elements. The bus number is the primer key of the data. The bus types must include only one slack bus. The last group of parameters are used by solver for deviation cases.

Field Name	Type	Description
`Num`	integer	Bus number
`Type`	integer	1 : PQ , 2 : PV , 3 : Slack
`PL`	double	Active power of load (MW)
`QL`	double	Reactive power of load (MVAR)
`GL`	double	Active power of shunt conductance (MW) at V = 1 pu
`BL`	double	Reactive power of shunt susceptance (MVAR) at V = 1 pu
`Area`	integer	Area number
`Vmag`	double	Voltage magnitude (pu)
`Vang`	double	Voltage angle (deg)
`Name`	string	Bus name string
`KV`	double	Nominal voltage (KV)
`Zone`	integer	Zone number
`MinV`	double	Minimum voltage limit (pu)
`MaxV`	double	Maximum voltage limit (pu)
`G`	double	Shunt conductance (pu)
`B`	double	Shunt susceptance (pu)
`Pg`	double	Active power of generation (MW)
`Qg`	double	Reactive power of generation (MVAR)
`MinPg`	double	Minimum active power limit (MW)
`MaxPg`	double	Maximum active power limit (MW)
`MinQg`	double	Minimum reactive power limit (MVAR)
`MaxQg`	double	Maximum reactive power limit (MVAR)
`PL_DevIndex`	integer	The index of deviation parameter for PL
`QL_DevIndex`	integer	The index of deviation parameter for QL
`PL_Offset`	double	The deviation offset of PL (MW)
`QL_Offset`	double	The deviation offset of QL (MVAR)
`P_Slope`	double	Active power droop control (MW/Hz)

Table 3.5: Bus data field

3.4.2 Generator Data : `PFD.Gen`

Generator data is used to define the generators characteristics at PV and slack buses. Table 3.6 shows the data field of `PFD.Gen`. Each field represents single column with `PFD.Gen.Count` elements. The variable `PFD.Gen.Bus` represents the bus number where the generator is connected. The `PFD.Gen_Bus` with `PFD.Gen.ID` are used to identify each generator. The load function can create the `PFD.Gen` based on bus data to support other file format. The solver allows more than one generator at the same bus and uses aggregation as following :

```
for k = 1 : PFD.Gen.Count
  n = PFD.Gen.BusIndex(k);
  PFD.Bus.Pg(n) = PFD.Bus.Pg(n) + PFD.Gen.Pg(k);
  PFD.Bus.Qg(n) = PFD.Bus.Qg(n) + PFD.Gen.Qg(k);
  PFD.Bus.MinPg(n) = PFD.Bus.MinPg(n) + PFD.Gen.MinPg(k);
  PFD.Bus.MinQg(n) = PFD.Bus.MinQg(n) + PFD.Gen.MinQg(k);
  PFD.Bus.MaxPg(n) = PFD.Bus.MaxPg(n) + PFD.Gen.MaxPg(k);
  PFD.Bus.MaxQg(n) = PFD.Bus.MaxQg(n) + PFD.Gen.MaxQg(k);
end
```

Field Name	Type	Description
`Bus`	integer	Bus number
`ID`	integer	The order of generator at the bus
`Pg`	double	Active power of generation (MW)
`Qg`	double	Reactive power of generation (MVAR)
`MaxQg`	double	Maximum reactive power limit (MVAR)
`MinQg`	double	Minimum reactive power limit (MVAR)
`RegVmag`	double	Regulation reference voltage
`RegBus`	double	Bus number of voltage regulation
`Active`	Boolean	0: Deactivated , 1 : Activated
`Var`	double	Reactive power for voltage regulation (MVAR / V)
`MaxPg`	double	Maximum active power limit (MW)
`MinPg`	double	Minimum active power limit (MW)
`P_Slope`	double	Active power droop control (MW/Hz)

Table 3.6: Generator data field

3.4.3 Transmission Line Data : `PFD.Line`

Table 3.7 shows the field which used to define each transmission line. Each line is identified by three fields (`SrcBus` , `DstBus` , `CKT`). The line data defines the network topology which can be represented by incident matrix. The following script is used to create modified incident matrix `PFD.Gamma` from line data :

```
PFD.Gamma = zeros(PFD.Line.Count,PFD.Bus.Count);
for k = 1 : PFD.Line.Count
   Src = PFD.Line.SrcIndex(k);
   Dst = PFD.Line.DstIndex(k);
   a = PFD.Line.Tap(k) .* exp(1j .* PFD.Line.Tang(k) .* pi/180);
   PFD.Gamma(k,Src) = a;
   PFD.Gamma(k,Dst) = -1;
end
```

Field Name	Type	Description
`SrcBus`	integer	Source bus number
`DstBus`	integer	Destination bus number
`CKT`	integer	The circuit number
`R`	double	Line resistance (pu)
`X`	double	Line reactance (pu)
`B`	double	Line susceptance (pu)
`MaxS_1,2,3`	double	Maximum apparent power (MVA)
`Tmag`	double	Tap transformer ratio
`Tang`	double	Tap transformer angle (deg)
`G1`	double	Src. bus shunt conductance (pu)
`B1`	double	Src. bus shunt susceptance (pu)
`G2`	double	Dst. bus shunt conductance (pu)
`B2`	double	Dst. bus shunt susceptance (pu)
`Active`	Boolean	0: Deactivated , 1 : Activated
`MinTmag`	double	Minimum ratio of tap transformer
`MaxTmag`	double	Maximum ratio of tap transformer
`MinTang`	double	Minimum shift angle of tap transformer (deg)
`MaxTang`	double	Maximum shift angle of tap transformer (deg)

Table 3.7: Transmission line data field

3.4.4 Transformer Data : `PFD.Trans`

The transformer data is used to define variable tap changer transformers. Tap changer transformer is used to control the bus voltage by changing the tap ratio. Phase shift transformer is used also to improve the power transfer capacity of the line. Table 3.8 shows the transformer data fields. The solver assumes zero error of voltage regulation and estimate the tap ratio as continuous variable. Transformer data is used to update the data of corresponding lines then the solver will depend on the line data only. The following code is used to update the line parameters of transformer based on three field (`SrcBus` , `DstBus` , `CKT`) :

```
for n = 1 : PFD.Trans.Count
   for k = 1 : PFD.Line.Count
      if ((PFD.Line.SrcBus(k) == PFD.Trans.SrcBus(n)) && ...
           (PFD.Line.SrcBus(k) == PFD.Trans.SrcBus(n)) && ...
           (PFD.Line.CKT(k) == PFD.Trans.CKT(n)))
           PFD.Line.TmagMin(k) = PFD.Trans.TmagMin(n);
           PFD.Line.TmagMax(k) = PFD.Trans.TmagMax(n);
           break;
      end
   end
end
```

Field Name	Type	Description
`SrcBus`	integer	Source bus number
`DstBus`	integer	Destination bus number
`CKT`	integer	The circuit number
`RegBus`	double	Bus index of voltage regulation
`TmagMax`	double	Maximum ratio of tap transformer
`TmagMin`	double	Minimum ratio of tap transformer
`MinV`	double	Minimum voltage limit (pu)
`MaxV`	double	Maximum voltage limit (pu)
`Step`	double	Voltage Step(pu)
`TableZ`	integer	Impedance table index

Table 3.8: Transformer line data field

3.4.5 Shunt Device Data `PFD.Shunt`

The shunt device can be used to model the following :

- STATCOM (Static Synchronous Compensator)
- TCSC (Thyrister Controlled Series Capacitor)
- Shunt Reactor or Capacitor Banks

The shunt device model is defined by its injected reactive power and the active power is zero. The model can be used with basic power flow mode to estimate the required reactive power at any PQ bus to set the bus voltage. It also can be used with OPF solver to estimate the optimal capacity of the shunt device. This model is based on the same power flow constraints for generator but with zero active power limits. The shunt device is normally connected to PQ buses and converts it to PV bus if it is set on voltage regulation mode as shown in the following script. Table 3.9 lists the data fields of the shunt device. The shunt device data is supported only with XLS data format but it can be defined as generator to be compatible with other formats.

```
for k = 1 : PFD.Shunt.Count
  n = PFD.Shunt.BusIndex(k);
  BusType = PFD.Bus.Type(n);
  if ((BusType == 1) & (PFD.Shunt.RegMode(k) > 0))
    PFD.Bus.Type(n) = 2; % Convert PQ bus to PV bus
  end
end
```

Field Name	Type	Description
`Bus`	integer	Bus number
`Qsh`	double	Reactive power (MVAR)
`MinQsh`	double	Minimum reactive power (MVAR)
`MaxQsh`	double	Maximum reactive power (MVAR)
`RegRef`	double	Voltage regulation set point (pu)
`RegMode`	double	0 : Voltage regulation , 1 : Fixed reactive power

Table 3.9: Shunt device data field

3.4.6 Series Device Data : `PFD.Series`

The series device can be used to model the following :

- Series Capacitor
- SSSC (Static Synchronous Series Compensator)
- TCSC (Thyrister Controlled Series Capacitor)

The series device model is based on compensation factor (`Kc`) which represents the reduction ratio in the line reactance. compensation factor can be constant for basic power flow or variable in OPF. This model is full suitable for both methods of admittance matrix and incident matrix and support lines with transformer of any type. The equivalent capacitance of the compensation factor can be calculated using Equation 3.1 .The compensation factor is used to update the line impedance during each iteration as shown in the following script.

$$C = \frac{1}{\omega \times XL \times Kc} \tag{3.1}$$

```
for n = 1 : PFD.Series.Count
  m = PFD.Series.LineIndex(n);
  SOL.Kc(m) = PFD.Series.Kc(n);
end
SOL.Line_Z = complex(PFD.Line.R, PFD.Line.X * (1 - SOL.Kc));
SOL.Line_Y = SOL.Line_Z .^(-1);
```

Field Name	Type	Description
`SrcBus`	integer	Source bus number
`DstBus`	integer	Destination bus number
`CKT`	integer	The circuit number
`Kc`	double	Compensation factor (pu)
`MinKc`	double	Minimum Compensation factor (pu)
`MaxKc`	double	Maximum Compensation factor (pu)
`RegRef`	double	Set point of regulation (MW or deg)
`RegMode`	double	0 : Fixed , 1 : Active power , 2 : Angle

Table 3.10: Series device data field

3.4.7 Deviation Parameters : PFD.DevPar

Deviation parameter is variable used to represent deviation ratio in (pu) relative to reference value. Deviation case is a set of deviation parameters used to modify the base case. The PFD.DevPar is matrix used to store deviation parameters. Each column represents single deviation parameter and each row defines a deviation case as shown in Figure 3-3. The deviation case is selected by PAR.DevCaseIndex and deviation parameters are selected by PFD.Bus.PL_DevIndex and PFD.Bus.QL_DevIndex as demonstrated in the following code :

```
PL = PFD.Bus.PL + PFD.PL_Offset;
QL = PFD.Bus.QL + PFD.QL_Offset;
CaseIndex = PAR.DevCaseIndex;
if (CaseIndex > 0)
  for k = 1 : PFD.Bus.Count
    DevIndex = PFD.Bus.PL_DevIndex(k);
    if (DevIndex > 0)
      PL(k) = PL(k) * ( 1 + PFD.DevPar(CaseIndex, DevIndex));
    end
    DevIndex = PFD.Bus.QL_DevIndex(k);
    if (DevIndex > 0)
      QL(k) = QL(k) * ( 1 + PFD.DevPar(CaseIndex, DevIndex));
    end
  end
end
```

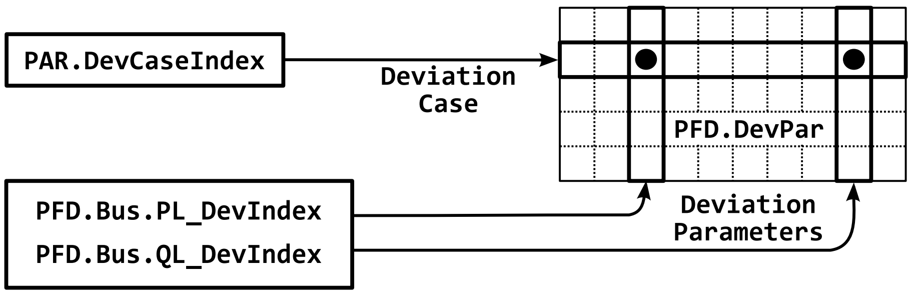


Figure 3-3: Deviation parameter

3.5 Model Variables : SOL

SOL structure holds the solution result in addition to other intermediate variables used during solver process. The solver build the system model based on state variables which include predefined constant inputs and unknown variables. The constant parameters include any variable which remain constant during the solver iteration process. The demand at each bus is an example of constant parameter. The unknown variable has initial value and boundary limits. The solver try to find the unknown variables which satisfy the system equation and minimizing the object function for optimal power flow. The solver uses the state variables listed in Table 3.11 to represent the system model and the final solution. Other results like lines currents are calculate in the post processing based on the buses voltages. The injected power at each bus is calculated by subtracting demand power from the generation power. The injected current can be calculated from the bus voltage and injected power. Table 3.12 lists the unknown variables at each bus based on bus type and solver mode. The optimal power flow mode supports optimization of tap transformer , series and shunt device based on setting parameters. The solver detect if any variable is fixed when the maximum and minimum limits are equal. Frequency is add to the unknown variables vector in active power droop mode only.

Element	State Variables
Bus	`Vmag` `Vang` `Pg` `Qg`
Tap transformer	`Tmag` `Tang`
Grid Frequency	`Freq`
Series Device	`Kc`
Shunt Device	`Qg`

Table 3.11: State variables types for each element

Bus Type	Normal Power Flow	Optimal Power Flow	Power Flow with Droop
PQ	`Vmag` , `Vang`	`Vmag` , `Vang`	`Vmag` , `Vang`
PV	`Qg` , `Vang`	`Pg` , `Qg` , `Vang`	`Qg` , `Vang`
Slack	`Pg` , `Qg`	`Pg` , `Qg`	`Qg`

Table 3.12: Unknown variables at each bus type based on solver mode

3.6 Unknown Variables Indexing

The solver selects unknowns vector from 8 types of state variables as listed in Table 3.11. Each unknown variable type has two indexes used to update the original state variables from the unknown vector at each iteration as shown in this script :

SOL.Vmag(SubIndex.Vmag) = X(VarIndex.Vmag)

- `SubIndex` $\begin{cases} \text{Bus Index for } \texttt{Vang} , \texttt{Vmag} , \texttt{Pg} , \texttt{Qg} \\ \text{Line Index for } \texttt{Tang} , \texttt{Tmag} , \texttt{Kc} \end{cases}$
- `VarIndex` : Variable order in unknowns vector

`PF_Setup()` function selects unknowns vector and creating the `SubIndex` structure based on the `PFD` and `PAR`. Then `PF_InitVarIndex()` function is used to create and initialize the `VarIndex` structure based on `SubIndex`. The Table 3.13 shows the variable indexing for simple system shown in Figure 3-4. The optimal power flow includes the tap transformer variables and active power generated at the PV bus.

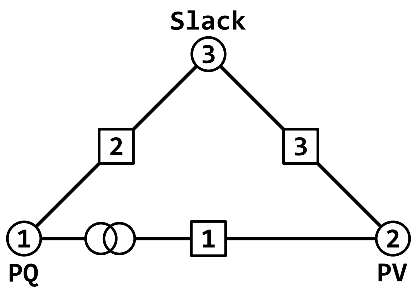


Figure 3-4: Simple network as an example for unknown vector indexing

	Conventional Power Flow			Optimal Power Flow		
`VarType`	`Count`	`VarIndex`	`SubIndex`	`Count`	`VarIndex`	`SubIndex`
`Vang`	2	`[1,2]`	`[1,2]`	2	`[1,2]`	`[1,2]`
`Vmag`	1	`[3]`	`[1]`	1	`[3]`	`[1]`
`Pg`	1	`[4]`	`[3]`	2	`[4,5]`	`[2,3]`
`Qg`	2	`[5,6]`	`[2,3]`	2	`[6,7]`	`[2,3]`
`Tmag`	0	`[ ]`	`[ ]`	1	`[8]`	`[1]`
`Tang`	0	`[ ]`	`[ ]`	1	`[9]`	`[1]`
`Freq`	0	`[ ]`	`[ ]`	0	`[ ]`	`[ ]`
`Kc`	0	`[ ]`	`[ ]`	0	`[ ]`	`[ ]`

Table 3.13: Variables indexing example

3.7 Solve Function

`PF_Solve()` function is used to create the system model and solve it based of input data and setting parameters. First , `PF_Setup()` created the system equations and variables. Then the solver is selected as following :

`(PAR.SolverType = 1)` : Optimal Power Flow with `fmincon()` function

`(PAR.SolverType = 0)` : Basic or Droop based Power Flow with `fsolve()` function

`fsolve()` function is used to solve system of nonlinear equations as following :

```
[X, EqValue, ExitFlag] = fsolve(StartValue, @PF_EqFun, options)
```

- Start Value :

  Start values can be selected be zeros for all variables and unity for bus voltages. When (`PAR.StartPoint = 1`) , the start values will be loaded from `SOL` .

- Equations Function Address : `@PF_EqFun`

  The `PF_EqFun()` takes unknown vector (`X`) and calculate the value of each equation then it returns them as single vector (`h`).

  ```
  function h = PF_EqFun(X)
  ```

- Options :

  Options define the stop criteria as maximum iterations, maximum error of (`h`), maximum difference of successive values of (`X`). `optimset()` function is used to create option structure as shown in the following code :

  ```
  options = optimset('Display','off','MaxFunEvals', 1e6,...
  'MaxIter',MaxTryCount,'TolFun',MaxErr_Fun,'TolX',MaxErr_X);
  ```

- Variables Vector : `X`

  This is the final value of the unknown vector.

- The equations value : `EqValue`

  This is the last value returned from `PF_EqFun()`.

  It should be near zero if the result is correct.

- Exit Flag : `ExitFlag`

  This variable indicate the exit conditions of the function.

  It should be positive if the result is correct.

`fmincon()` function is used to find the minimum value of `PF_MinFun()` function under the system limits represented by `PF_LimitFun()`. The following additional parameters are required by `fmincon()` :

```
[X, MinValue, ExitCode] = fmincon(@PF_MinFun, StartValue,...
          [],[],[],[],Min, Max, @PF_LimitFun, options)
```

- Object Function Address : `@PF_MinFun`

  This function is used to represent the system cost function which can include system losses or generation cost.
- Min,Max : The variables limits
- Non Linear Constrains Function Address : `@PF_LimitFun`

  The first output from this function represents the inequality constrained and the second one represents the equality constrained.
- Options :

  Options define the stop criteria as the maximum iterations, the maximum tolerance of (`f`) and the maximum tolerance of limits. `optimset()` function is used to create the option structure as shown in the following code :

  ```
  options=optimset('MaxFunEvals',10e5,'MaxIter',MaxTryCount,...
  'Display','off','TolFun',MaxErr_Fun,'TolCon',MaxErr_Con);
  ```

The final results is created by `PF_GetSol()` based on the unknowns vector.
Figure 3-5 shows the flow chart of `PF_Solve()` function as explained.

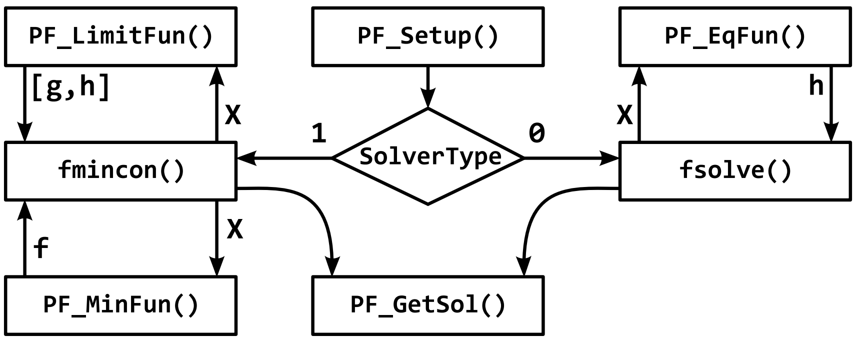


Figure 3-5: Flow chart of Solve function

3.8 File Formats

PF_Solver() support three different data files formats as listed in Table 3.14. The PF_LoadData() and PF_SaveData() can detect the data file format based on its extension and call the correct function automatically. The data conversion between different format may not be reversible because the extra data can be lost and there is no way to return it back. The solver defines its own format of PFD using Excel XLS file. This format is recommended for future use because it supports all other formats and it is easy to edit. Single XLS file can include all deviation cases and results in addition to the setting parameters. So that it is the best option to exchange the data with the solver. Figure 3-6 shows the data flow inside the solver. The data file should be loaded first to PFD. Then the PF_Solve() function will get the result and store it in SOL. The final solver result can be saved after updating the data of PFD with PF_UpdateData() function. The setting parameters can be loaded and saved to parameter file using functions PF_LoadPar() and PF_SavePar().

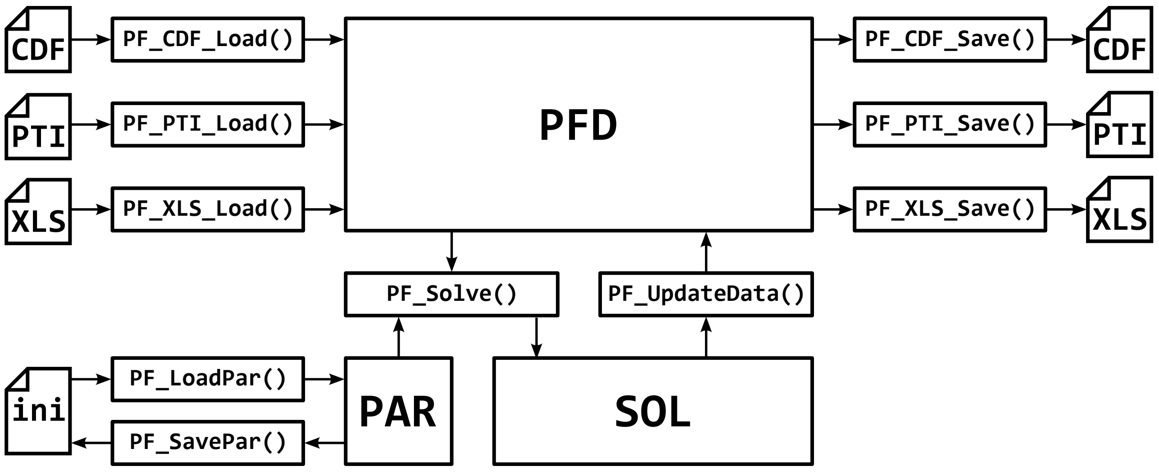


Figure 3-6: Data flow of solver

File Type	Description	Ref.
CDF	IEEE Common Data File	[41]
PTI	Power Technologies Incorporated format	[78]
XLS	Excel file based on PTI format	PFD

Table 3.14: The supported file formats

3.9 Functions Reference

Table 3.15 shows the outer layer function with short description.

Function Name	Description
`PF_CDF_Load()`	Load PFD data from CDF file
`PF_CDF_Save()`	Save PFD data to CDF file
`PF_PTI_Load()`	Load PFD data from PTI file
`PF_PTI_Save()`	Save PFD data to PTI file
`PF_XLS_Load()`	Load PFD data from XLS file
`PF_XLS_Save()`	Save PFD data to XLS file
`PF_LoadData()`	Load PFD data from `PFD.InputFileName`
`PF_SaveData()`	Save PFD data to `PFD.OuputFileName`
`PF_LoadPar()`	Load setting parameters from `PAR.FileName`
`PF_SavePar()`	Save setting parameters to `PAR.FileName`
`PF_Load()`	Load parameters and data
`PF_Save()`	Save parameters and data
`PF_Setup()`	Create and initialize the variables
`PF_Solve()`	Run the solver and show the results
`PF_ShowSol()`	Show results of SOL on screen
`PF_UpdateData()`	Update data of PFD by results from SOL

Table 3.15: Function reference

Chapter 4

Test Results Verification and Conclusion

Contents

4.1 Introduction

The proposed solver has been test and verified for the following modes:

- Basic Power Flow :
- Optimal Power Flow :
- Power Flow with Active Power Droop :
- Optimal voltage profile using shunt device:
- Active power flow of transmission line using series device :

The test is based on IEEE test cases which can be downloaded from University of Washington archive in CDF format [47]. The test case file are converted to PSS/E 26 format to be suitable for importing it to PowerFactory. IEEE test case are defined also in many articles [68] and included as an examples in may power analysis software [99].Table 4.1 lists the properties of each test case.

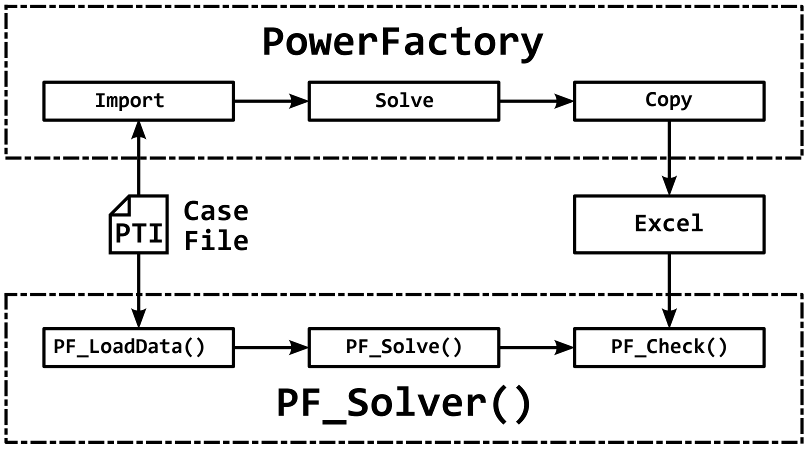


Figure 4-1: Result verification of solver with PowerFactory

Test Case	Buses	N_{PQ}	N_{PV}	Transformers	Lines	File Name
IEEE 14	14	9	4	3	20	C0014
IEEE 30	30	24	5	4	41	C0030
IEEE 57	57	50	6	17	80	C0057
IEEE 118	118	63	54	9	186	C0118
IEEE 300	300	230	69	107	411	C0300

Table 4.1: Test Cases

4.2 Basic Power Flow Results

The results of the conventional power is represented by bus voltage because other results can be calculate based on it. PowerFactory from DIgSILENT is used to verify the results of each case. Each IEEE test case has been converted to PTI file format and imported to PowerFactory. The result of each case is exported from PowerFactory to Excel and saved in text file to be used as reference case for the solver. The IEEE 300 case has been modified to remove the phase shift of transformer because the import function does not support variable phase shift. The following shows the error between the solver results and PowerFactory results. The error in voltage magnitude is about 10^{-5} and the error in the voltage angle is less than 10^{-3} which is accepted tolerance of power flow analysis. The PowerFactory results are represented in only 6 digits after the decimal point which explain this error. The solver result also matches many published research results [1],[52],[68].

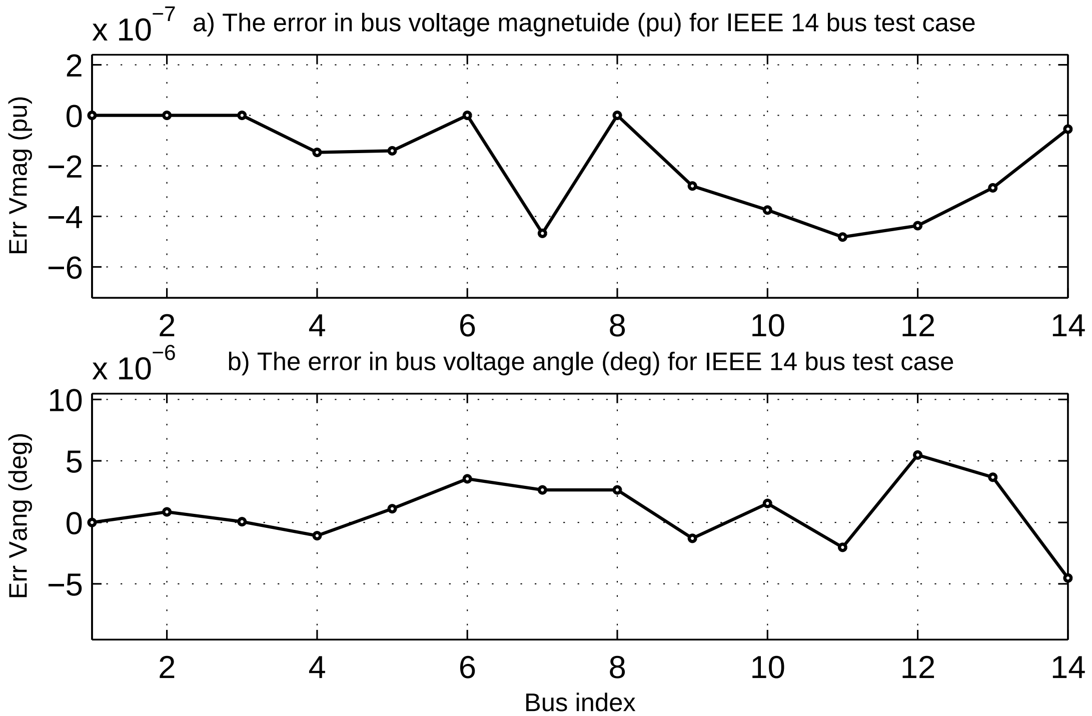


Figure 4-2: The voltage error of basic power flow results for IEEE 14 bus test case based on PowerFactory as reference

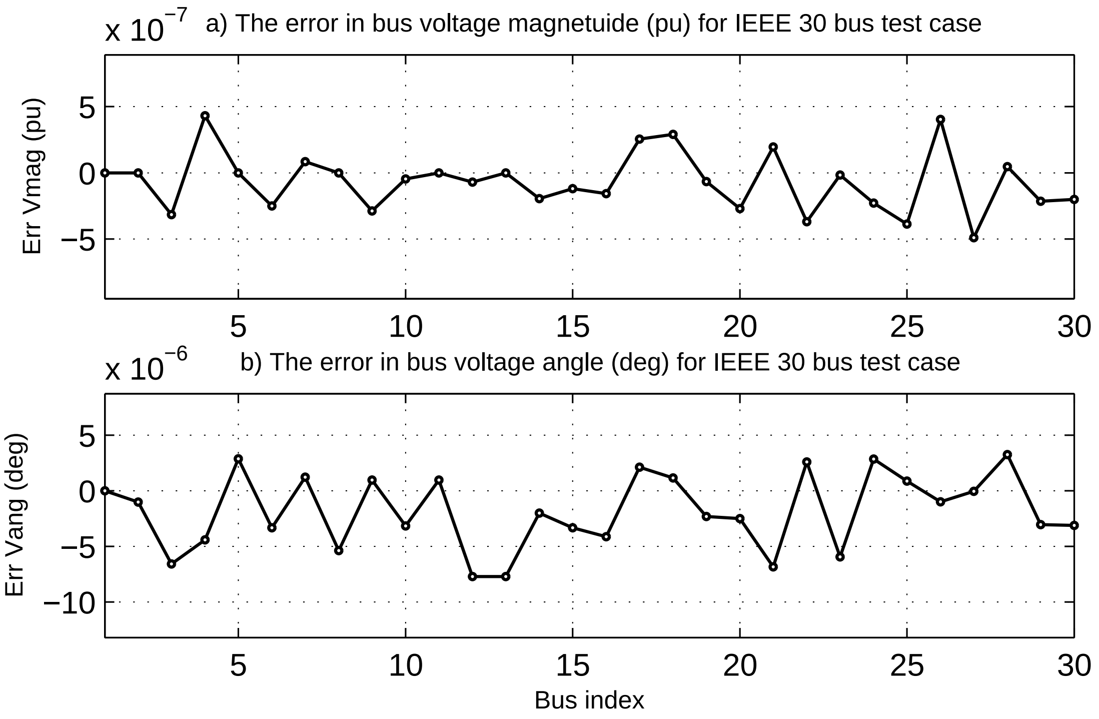


Figure 4-3: The voltage error of basic power flow results for IEEE 30 bus test case based on PowerFactory as reference

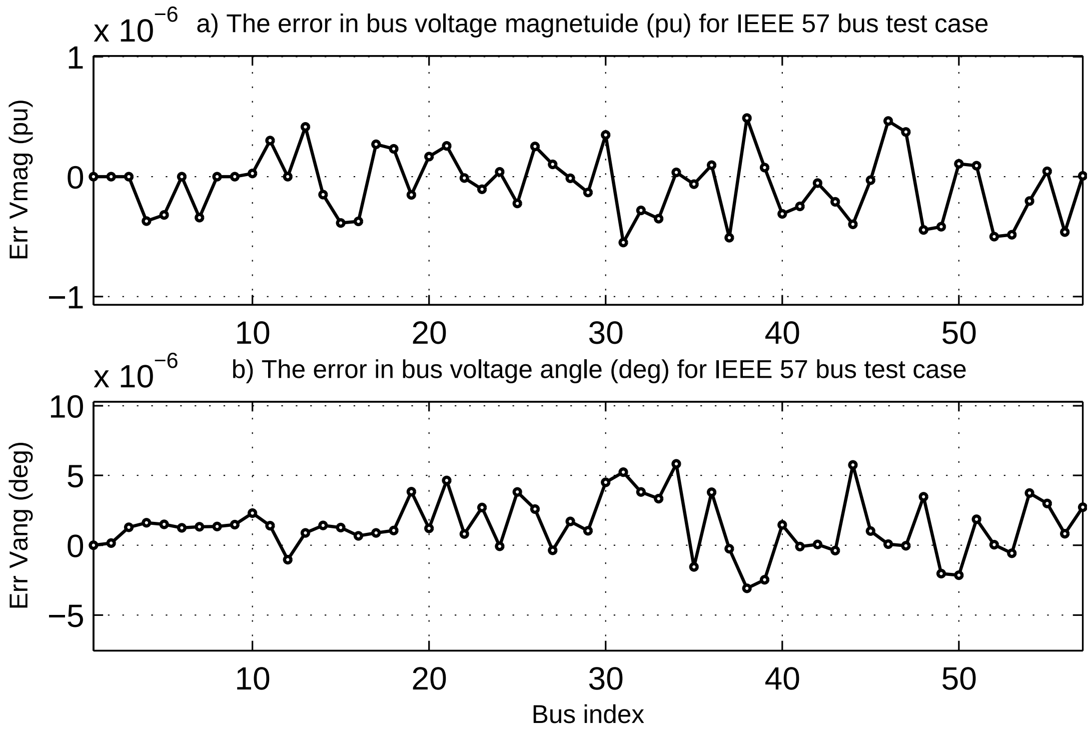


Figure 4-4: The voltage error of basic power flow results for IEEE 57 bus test case based on PowerFactory as reference

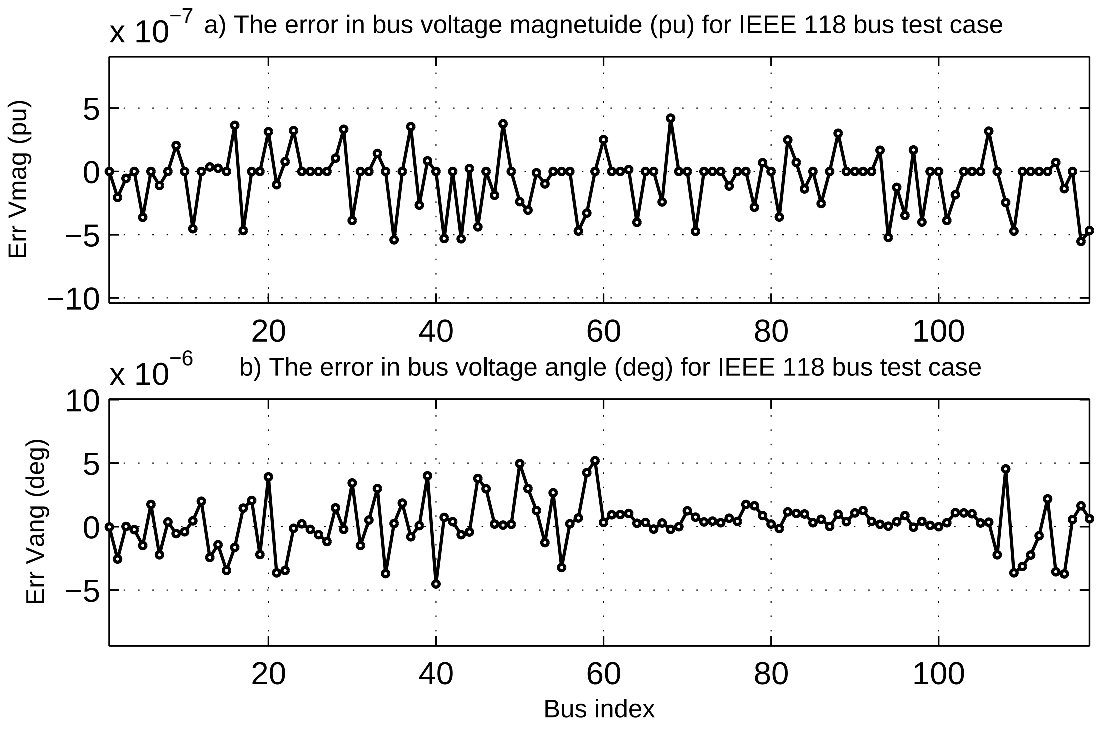


Figure 4-5: The voltage error of basic power flow results for IEEE 118 bus test case based on PowerFactory as reference

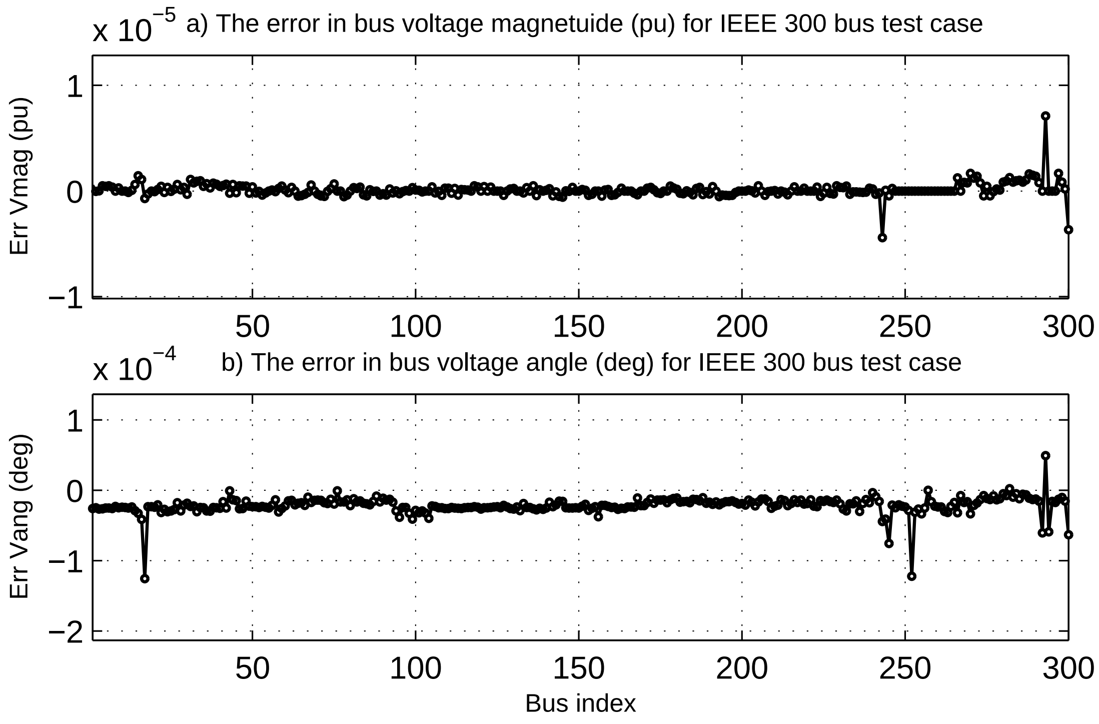


Figure 4-6: The voltage error of basic power flow results for IEEE 300 bus test case based on PowerFactory as reference

4.3 Power Flow with Active Power Droop Results

This test verify the basic power flow with considering the active power of generator is set based on droop characteristic. The reference power is calculated from basic power flow at normal loading condition. Then the solver apply the deviation parameters on the load at each bus and solve the power flow again to get the new frequency and the generation power. The reactive load is kept constant during this test. Table 4.2 lists the setting parameters for droop and load deviation.

Parameter	Value	Description
`BaseFreq`	60 Hz	Base frequency
`P_Slope`	50 MW/Hz	Droop slope used the same for all generators
`PL_Offset`	0	Deviation offset
`PL_DevPar`	1	Deviation parameter index
`DevCaseIndex`	1	Deviation case index
`DevPar(1)`	-2 % .. 2%	Deviation parameter for active power load

Table 4.2: Droop test parameters

DevPar / Case	IEEE 14	IEEE 30	IEEE 57	IEEE 118	IEEE 300
-2.00 %	60.056257	60.062531	60.129379	60.079709	60.166826
-1.50 %	60.042202	60.046910	60.097046	60.059792	60.125164
-1.00 %	60.028140	60.031282	60.064705	60.039868	60.083472
-0.50 %	60.014073	60.015645	60.032356	60.019937	60.041751
0.50 %	59.985921	59.984347	59.967636	59.980056	59.958220
1.00 %	59.971836	59.968687	59.935264	59.960106	59.916411
1.50 %	59.957745	59.953018	59.902884	59.940150	59.874571
2.00 %	59.943649	59.937342	59.870496	59.920187	59.832703

Table 4.3: Frequency (Hz) relative to load variation for IEEE test cases

4.4 Optimal Power Flow Results

The solver is tested in OPF mode with IEEE test cases to minimize the losses. The bus voltage is free within constraints boundary. Generators which work as motor ($P_g < 0$) or Synchronous Condenser (motor runs without load and can vary its reactive power only) ($P_g = 0$) are not included in active power dispatch because they can not vary their active power. The tap changer is kept constant as fixed network state. Each case has been test with solver and the result is verified with MATPOWER. The test use the setting shown in Table 4.4.

Parameter	Value	Description
`MinVmag`	0.94	Minimum level of bus voltage (pu)
`MaxVmag`	1.06	Maximum level of bus voltage (pu)
Power Limits		The same as MATPOWER 4.1 test cases

Table 4.4: Droop test parameters

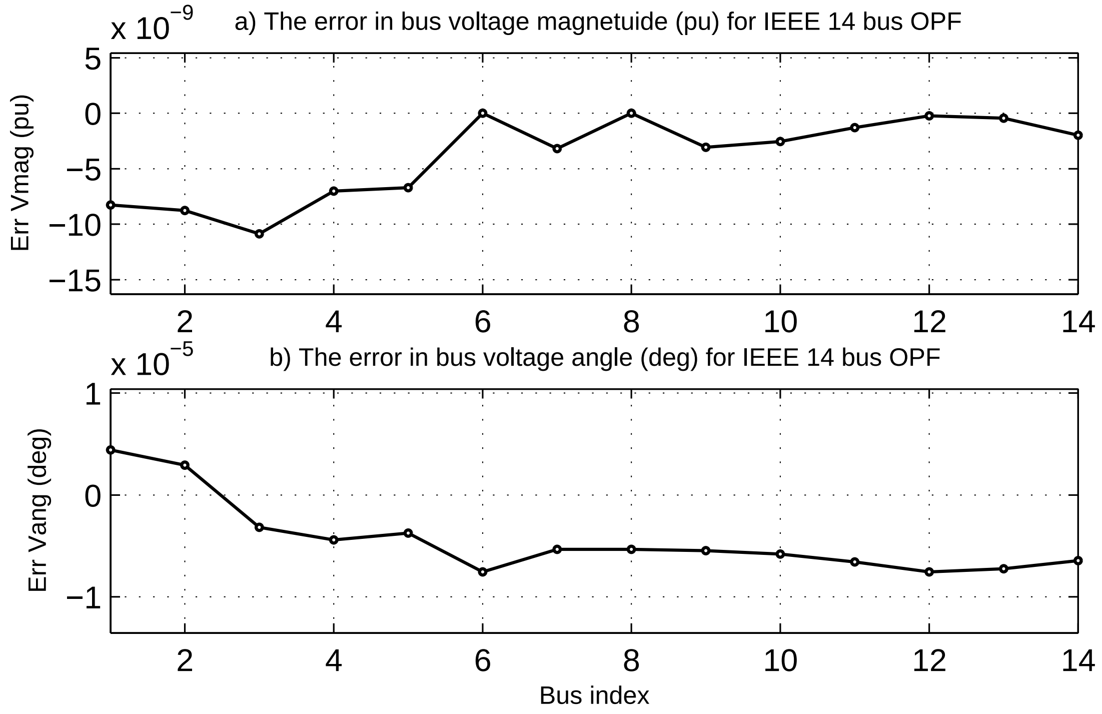


Figure 4-7: Voltage error of OPF for IEEE 14 bus test case
based on MATPOWER as reference case

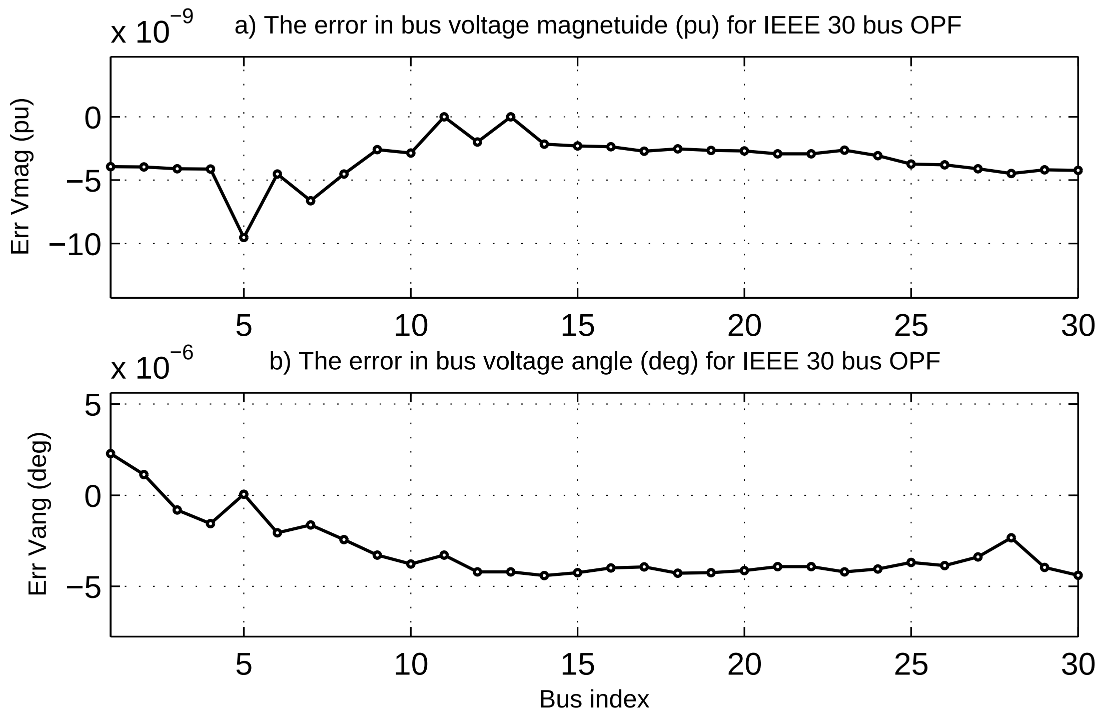


Figure 4-8: Voltage error of OPF for IEEE 30 bus test case
based on MATPOWER as reference case

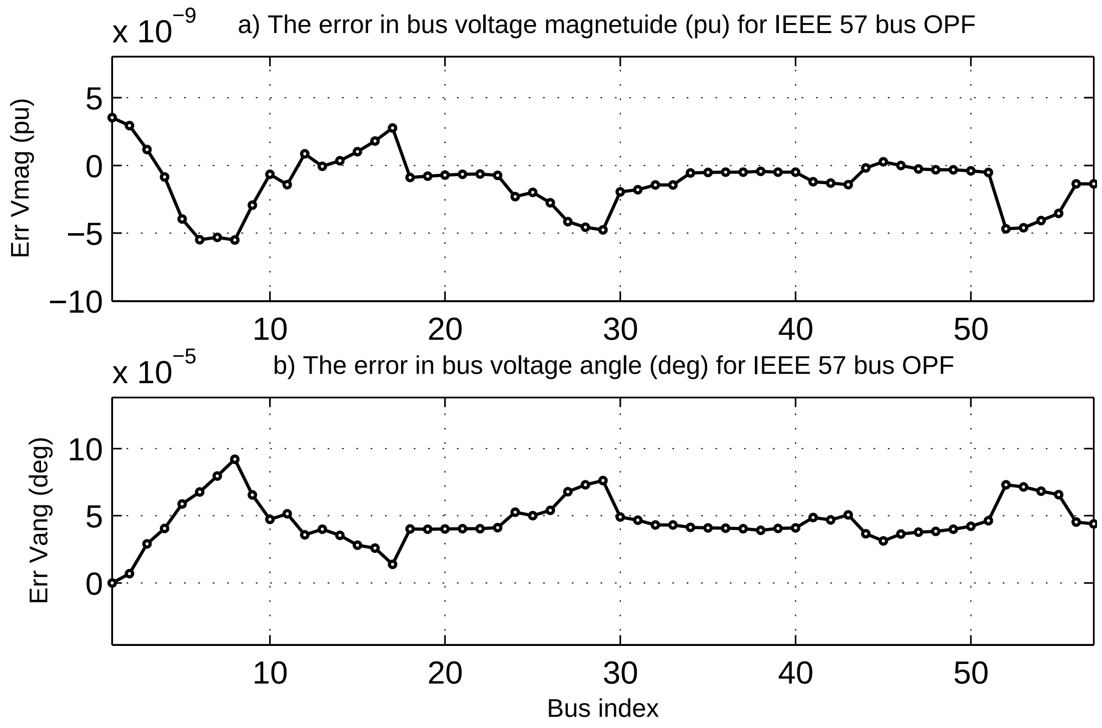


Figure 4-9: Voltage error of OPF for IEEE 57 bus test case
based on MATPOWER as reference case

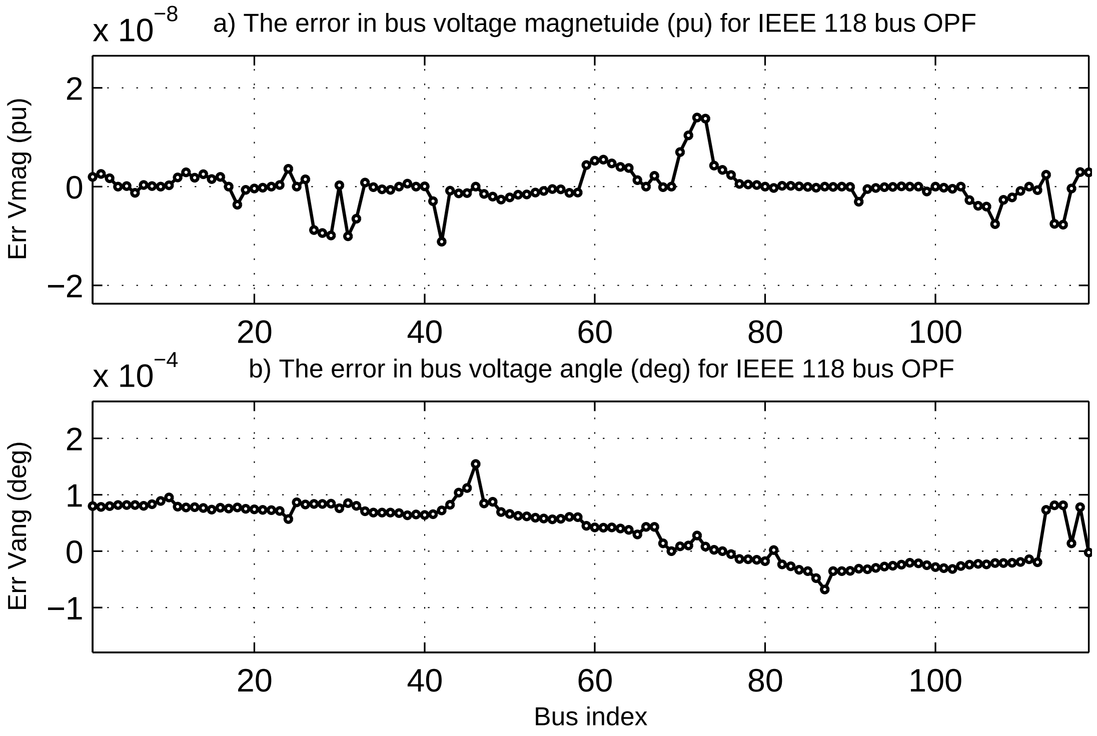


Figure 4-10: Voltage error of OPF for IEEE 118 bus test case based on MATPOWER as reference case

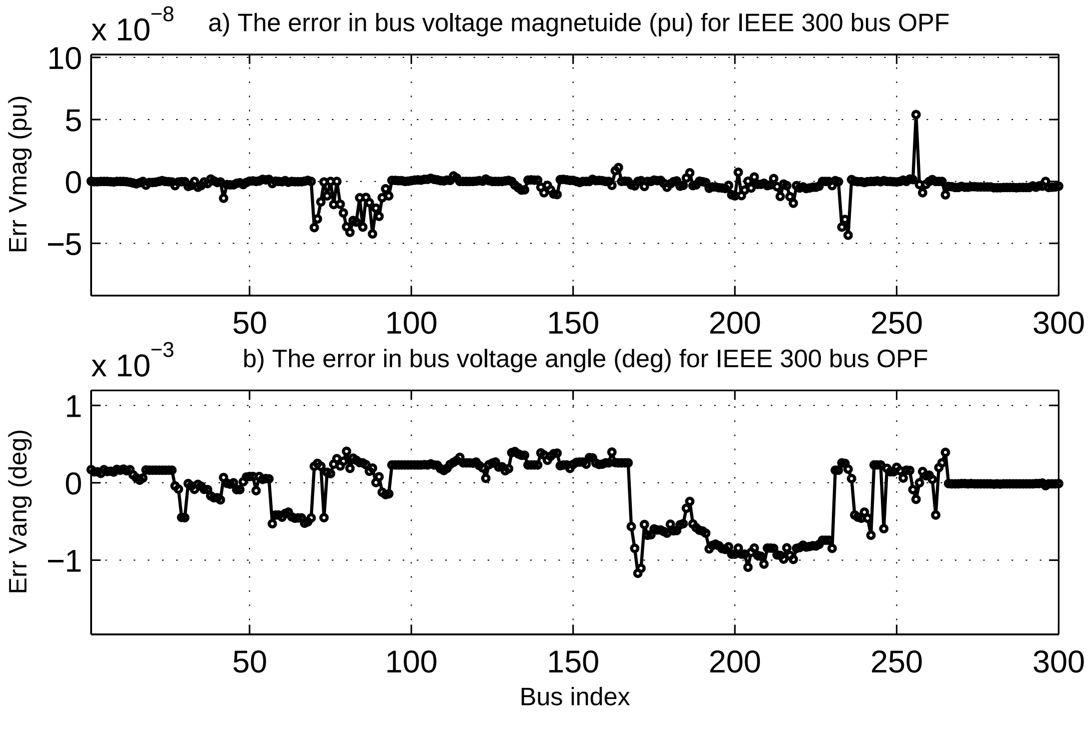


Figure 4-11: Voltage error of OPF for IEEE 300 bus test case based on MATPOWER as reference case

4.5 Optimal Voltage Profile using Shunt Device

This test selects the best location of shunt device which has the optimal voltage profile. The IEEE 14 bus case is used as base case for this test. Equation 4.1 defines the objective function which used to minimized the bus voltage $V(k)$ deviation from its nominal value which is selected to be (1 pu) for this test. The active power and voltage of PV buses are kept as base case. Table 4.5 shows the test result of each location for the shunt device. The minimum deviation of the voltage profile can be achieved at bus number 10 as shown in Figure 4-12.

$$f = \sqrt{\frac{1}{N_{PQ}} \sum \left(V\left(k\right) - 1\right)^2} \tag{4.1}$$

Case	Bus Num.	Vmag	Qsh (MVAR)	Voltage Deviation (%)
1	4	1.010929	-16.6252	4.5574
2	5	1.012809	-16.1752	4.6134
3	7	1.051215	-13.1528	4.3297
4	9	1.035442	-18.7538	3.8724
5	**10**	**1.021451**	**-20.4549**	**3.7336**
6	11	1.037953	-14.4058	4.2080
7	12	1.039077	-11.5239	4.4409
8	13	1.039915	-11.9852	4.4463
9	14	0.999864	-16.4461	4.0019

Table 4.5: The result of selecting shunt device location for optimal voltage profile

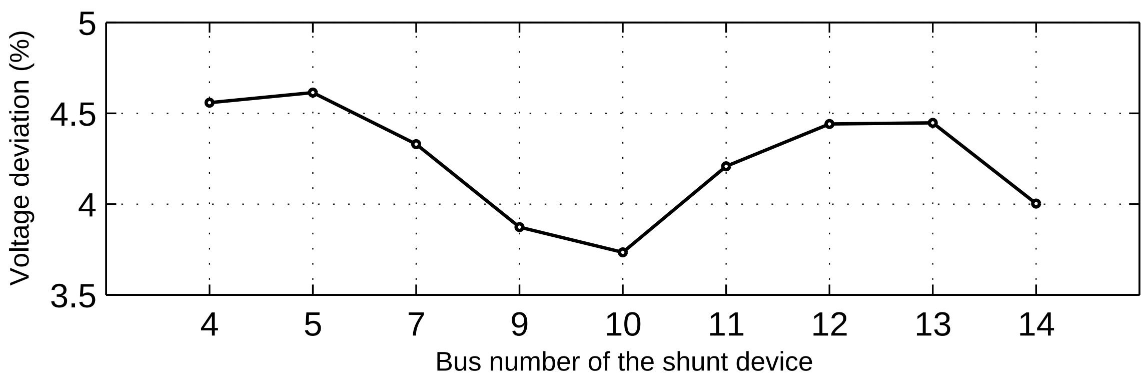


Figure 4-12: The voltage deviation relative to the shunt device location

4.6 Series Device to Control the Active Power Flow

This test add series device at line (1,5) of IEEE 14 bus test case to control the active power flow through the line. The solver estimate the compensation factor required to regulate the active power flow to be 90 MW as shown in Table 4.7 and Table 4.8. The compensation factor improves the line capacity and reduce the network over load.

Field Name	Value	Description
`Kc`	0.2	Compensation factor (pu)
`MinKc`	0	Minimum Compensation factor (pu)
`MaxKc`	0.4	Maximum Compensation factor (pu)
`RegRef`	90	Set point of regulation (MW)
`RegMode`	1	0 : Fixed , 1 : Active power regulation

Table 4.6: Series device data field

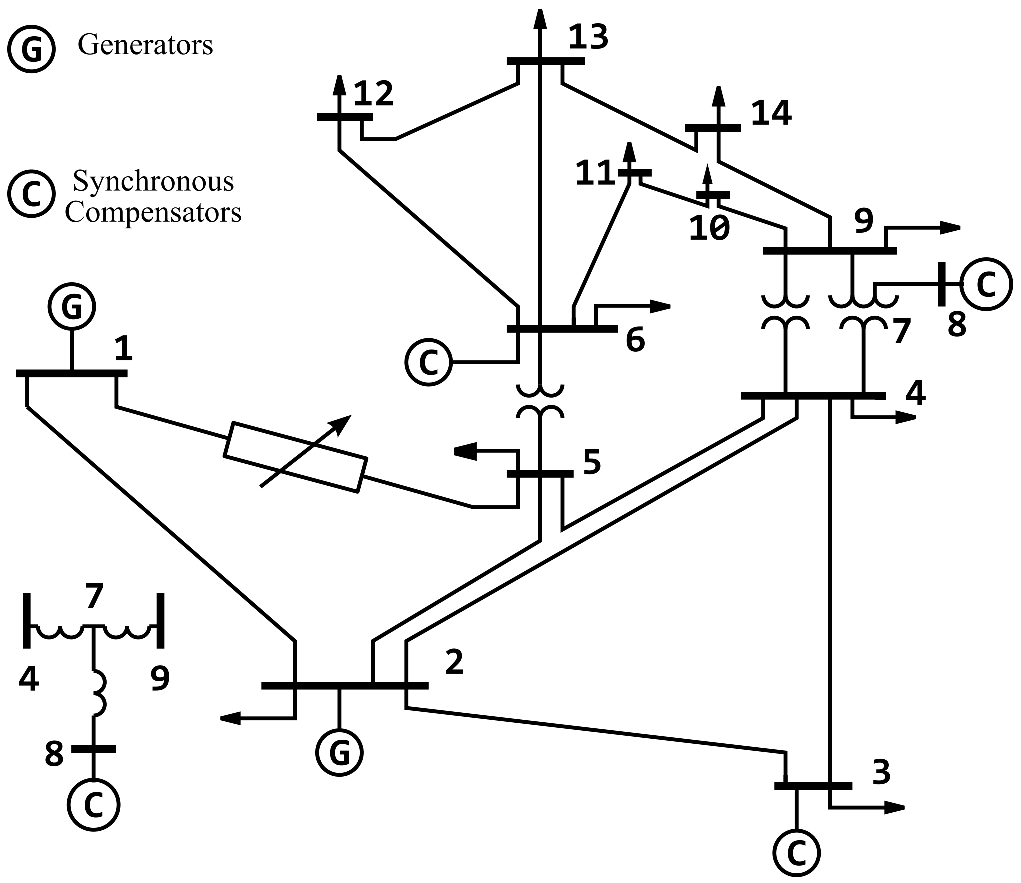


Figure 4-13: The digram of IEEE 14 bus case with series device at line (1,5) [55]

	Base Case		With Series Device	
Bus Num.	Vmag (pu)	Vang (deg)	Vmag (pu)	Vang (deg)
1	1.060000	0.000	1.060000	0.000
2	1.045000	-4.983	1.045000	-4.359
3	1.010000	-12.725	1.010000	-11.767
4	1.017671	-10.313	1.017757	-9.041
5	1.019514	-8.774	1.019407	-7.282
6	1.070000	-14.221	1.070000	-12.799
7	1.061520	-13.360	1.061476	-12.050
8	1.090000	-13.360	1.090000	-12.050
9	1.055932	-14.939	1.055763	-13.609
10	1.050985	-15.097	1.050825	-13.752
11	1.056907	-14.791	1.056804	-13.407
12	1.055189	-15.076	1.055190	-13.660
13	1.050382	-15.156	1.050338	-13.747
14	1.035530	-16.034	1.035408	-14.670

Table 4.7: The bus voltage of base case and with compensation of line (1,5)

Line			Base Case		With Series Device		
N.	Src.	Dst.	P (MW)	Q (MVAR)	P (MW)	Q (MVAR)	Kc (pu)
1	1	2	-152.5853	30.5592	-134.7376	23.0545	0.000000
2	**1**	**5**	**-72.7475**	**4.7863**	**-90.0000**	**10.9628**	**0.347157**
3	2	3	-70.9143	3.8363	-68.0884	2.7556	0.000000
4	2	4	-54.4548	4.7813	-48.4853	2.2665	0.000000
5	2	5	-40.6125	-0.3009	-32.4201	-3.5886	0.000000
6	3	4	23.6591	-4.1728	26.5858	-5.1201	0.000000
7	4	5	61.6727	-14.2010	70.1801	-17.0931	0.000000
8	4	7	-28.0742	11.3843	-27.7281	11.2988	0.000000
9	4	9	-16.0798	1.7323	-15.8813	1.6674	0.000000
10	5	6	-44.0873	-8.0495	-44.6400	-7.9477	0.000000
11	6	11	-7.2979	-3.4445	-7.6326	-3.3346	0.000000
12	6	12	-7.7143	-2.3540	-7.7549	-2.3330	0.000000
13	6	13	-17.5359	-6.7989	-17.7058	-6.7439	0.000000
14	7	8	0.0000	17.6235	-0.0000	17.6505	0.000000
15	7	9	-28.0742	-4.9766	-27.7281	-5.1053	0.000000
16	9	10	-5.2147	-4.1849	-4.8817	-4.2988	0.000000
17	9	14	-9.3102	-3.3629	-9.1032	-3.4459	0.000000
18	10	11	3.7979	1.6445	4.1326	1.5346	0.000000
19	12	13	-1.6080	-0.7483	-1.6484	-0.7271	0.000000
20	13	14	-5.5898	-1.6371	-5.7968	-1.5541	0.000000

Table 4.8: The line results of base case and compensation of line (1,5)

4.7 Conclusion

The proposed power flow solver has been implemented and verified relative to IEEE test cases result generated by PowerFactory and MATPOWER. The solver satisfies the following requirements:

- Support different data file format for input and output : CDF , PTI , XLS
- Solve active power dispatch with three modes :
  - Basic power flow : Calculate the active power of slack bus only
  - Active power droop control: Dispatch active power based on frequency
  - Optimal Power Flow : Optimize the active power for minimal cost function
- All configuration can be customized externally via setting parameters.
- Detect the input data error or missing parameters and gives clear error message.
- Support result verification with other tools.
- Use variable indexing to organize the vector of unknowns
- Extend the mathematical model with minimal changes
- Implement the mathematical model by admittance and incident matrices.
- Apply partial update of matrix during each iteration to speed up the solver.
- Support variable tap changer transformer and phase shift transformer.
- Estimate the reactive power of shunt device to regulate the bus voltage.
- Estimate the compensation factor of the line to achieve the required power flow.
- Automated analysis of deviation cases.
- Encapsulate all result and data in single file.
- The maximum error of voltage magnitude is about 10^{-5} (pu).
- The maximum error of voltage angle is about 10^{-3} (deg).

4.8 Future work

The following improvements are scheduled to be implemented in the future release:

- Extend the testing system to cover large networks.
- Improve the solver speed by using compiled library.
- Replace fmincon function with specialized algorithm for power flow.
- Detect infeasible cases and overcome the ill conditions.
- Support reactive power droop with voltage control.
- Implement UPFC (Unified Power Flow Control) model .
- Support discrete control for tap transformer.
- Support ON/OFF control of shunt capacitors and inductors.
- Compare results of different deviation cases.
- Creating report and figures for result
- Support additional data file formats

Appendix A

Test Cases Data

This appendix includes the diagrams of IEEE test cases for transmission network. The input file of each case is provide from University of Washington archive in CDF format [47]. IEEE test case are defined also in many articles [68] and included as an examples in many power analysis software [99].

Test Case	Buses	PQ Buses	PV Buses	Transformers	Lines
IEEE 14	14	9	4	3	20
IEEE 30	30	24	5	4	41
IEEE 57	57	50	6	17	80
IEEE 118	118	64	53	9	186
IEEE 300	300	231	68	107	411

Table A.1: Test Cases [51]

A.1 IEEE 14 bus Test Case

The IEEE 14 Bus Test Case represents a portion of the American Electric Power System which is located in the Midwestern US as of February, 1962. Basically this 14 bus system has 14 buses, 5 generators and 11 loads. The 14 bus test case does NOT have line limits. Compared to 1990's power systems, it has low base voltages and an overabundance of voltage control capability.

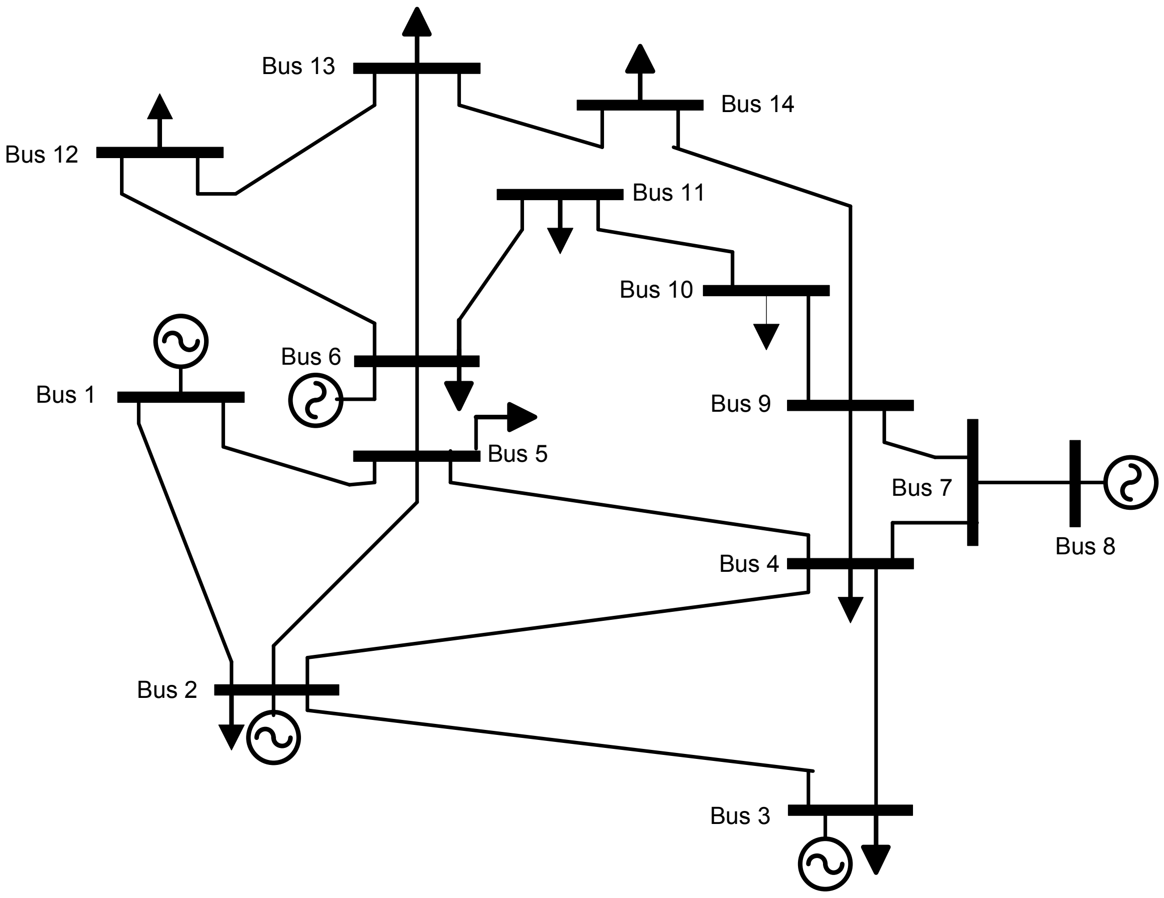


Figure A-1: One line diagram of IEEE 14 bus test case [68]

A.2 IEEE 30 bus Test Case

The IEEE 30 Bus Test Case represents a portion of the American Electric Power System (in the Midwestern US) as of December, 1961. Basically, the bus has 15 buses, 2 generators and 3 synchronous condensers. The 11 kV and 1.0 kV base voltages are the guess, which may not be the actual data. The model actually has these buses at either 132 or 33 kV.

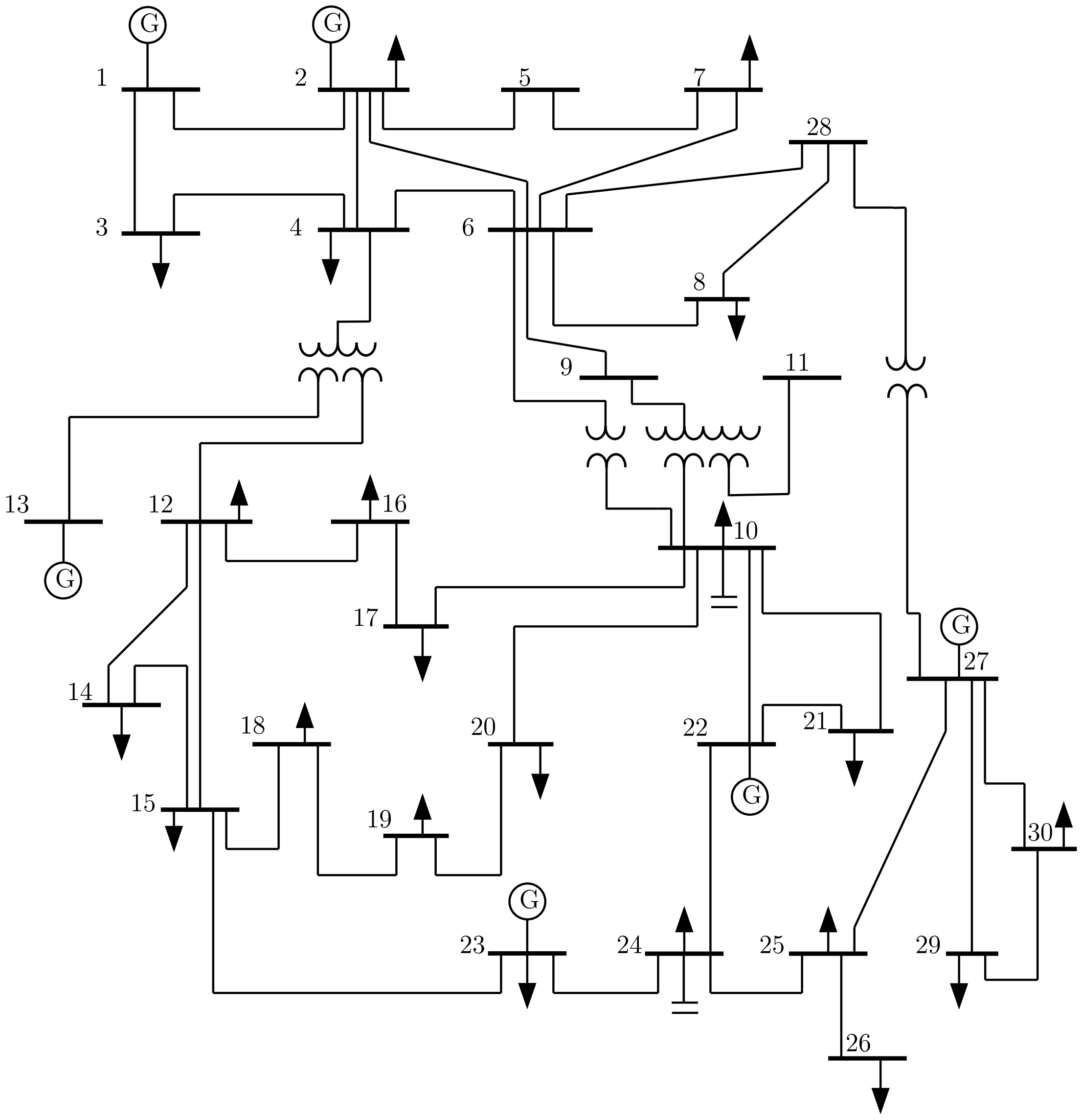


Figure A-2: One line diagram of IEEE 30 bus test case [44]

A.3 IEEE 57 bus Test Case

The IEEE 57 Bus Test Case represents a portion of the American Electric Power System (in the Midwestern US) as it was in the early 1960's. From the graph, we can see that this 57 bus system has 57 buses, 7 generators and 42 loads.

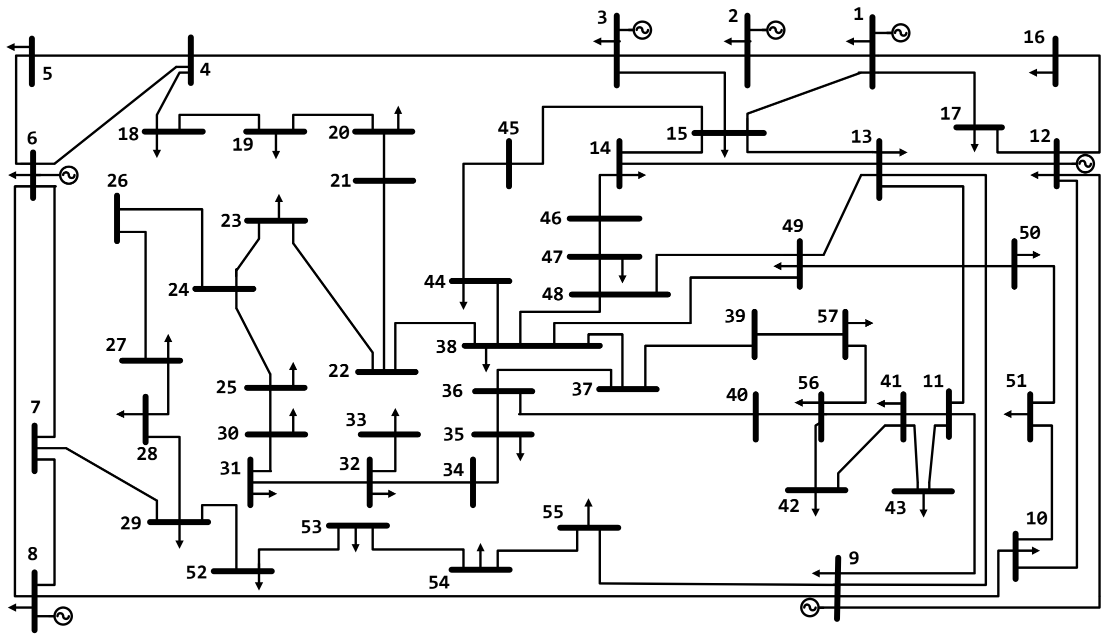


Figure A-3: One line diagram of IEEE 57 bus test case [68]

A.4 IEEE 118 bus Test Case

This IEEE 118 Bus Test Case represents a portion of the American Electric Power System (in the Midwestern US) as of December, 1962. Basically, this IEEE 118 bus system contains 19 generators, 35 synchronous condensers, 177 lines, 9 transformers and 91 loads.

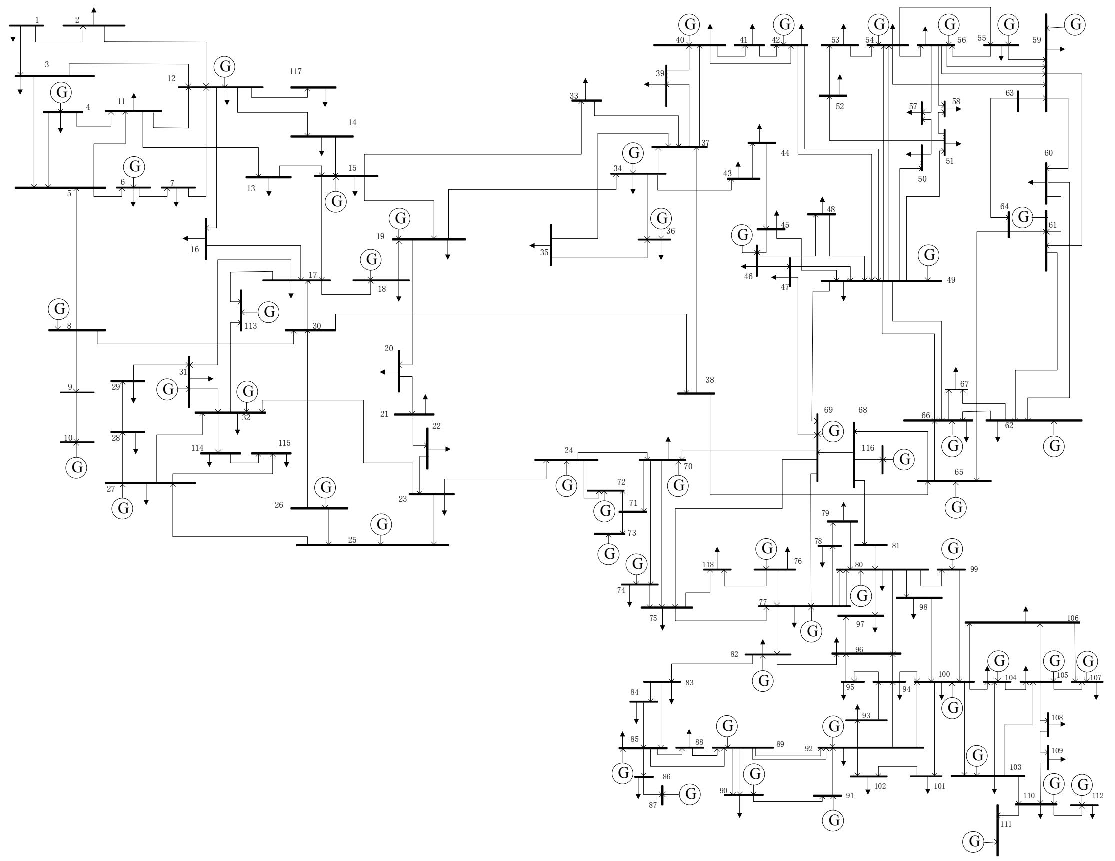


Figure A-4: One line diagram of IEEE 118 bus test case [10]

A.5 IEEE 300 bus Test Case

IEEE 300 bus system contains 69 generators, 60 LTCs, 304 transmission lines and 195 loads. The generator spinning reserves for the power is approximately 667 GW for positive and 713 GW for negative.

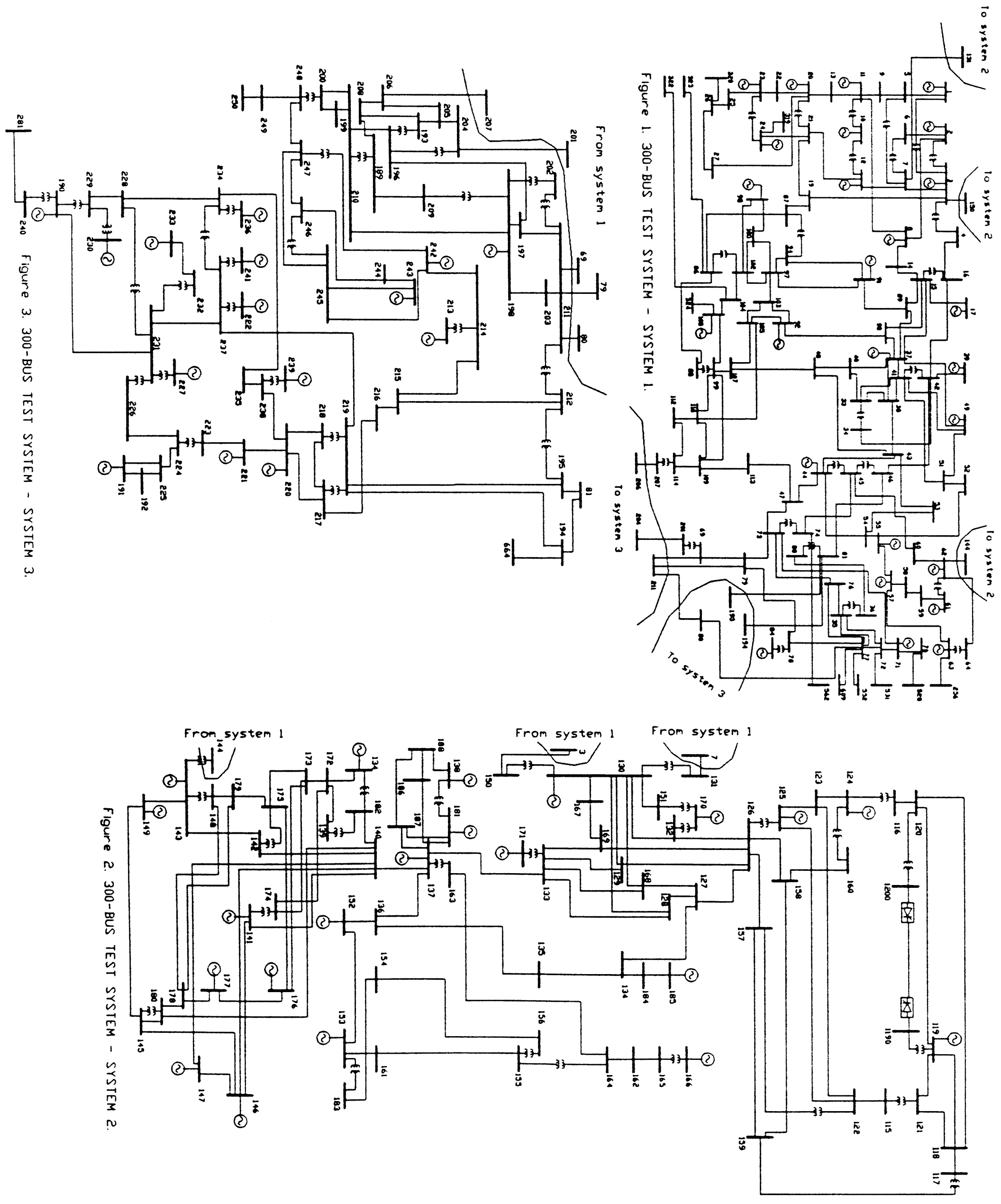


Figure 1. 300-BUS TEST SYSTEM - SYSTEM 1.
Figure 2. 300-BUS TEST SYSTEM - SYSTEM 2.
Figure 3. 300-BUS TEST SYSTEM - SYSTEM 3.
From system 1
To system 2
To system 3

Figure A-5: One line diagram of IEEE 118 bus test case [47]

Bibliography

[1] *PowerApps Validation Document Load Flow*, 2008.

[2] Carpentier J Abadie J. Generalization of the wolfe reduced gradient method to the case of nonlinear constraints. *Fletcher R (ed) Optimization*, pages 37–47, 1969.

[3] E. Acha, C.R. Fuerte-Esquivel, H. Ambriz-Pérez, and C. Angeles-Camacho. *FACTS: Modelling and Simulation in Power Networks.* Wiley, 2004.

[4] O. Alsac, J. Bright, M. Prais, and B. Stott. Further developments in lp-based optimal power flow. *Power Systems, IEEE Transactions on*, 5(3):697–711, Aug 1990.

[5] F. Aminifar, M. Fotuhi-Firuzabad, A. Khodaei, and S.O. Faried. Optimal placement of unified power flow controllers (upfcs) using mixed-integer non-linear programming (minlp) method. In *Power Energy Society General Meeting, 2009. PES '09. IEEE*, pages 1–7, July 2009.

[6] Goran Andersson. *Modelling and Analysis of Electric Power Systems.* ETH Zurich, 2008.

[7] A.S.A. Awad, T.H.M. EL-Fouly, and M.M.A. Salama. Optimal ess allocation and load shedding for improving distribution system reliability, 2014.

[8] Anibal T Azevedo, Aurelio RL Oliveira, Marcos J Rider, and Secundino Soares. How to efficiently incorporate facts devices in optimal active power flow model. *MANAGEMENT*, 6(2):315–331, 2010.

[9] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms.* Wiley, 2013.

[10] Ellery A Blood. From static to dynamic electric power network state estimation: The role of bus component dynamics. 2011.

[11] R.C. Burchett, H.H. Happ, and D. R. Vierath. Quadratically convergent optimal power flow. *Power Apparatus and Systems, IEEE Transactions on*, PAS-103(11):3267–3275, Nov 1984.

[12] R.C. Burchett, H.H. Happ, D. R. Vierath, and K.A. Wirgau. Developments in optimal power flow. *Power Apparatus and Systems, IEEE Transactions on*, PAS-101(2):406–414, Feb 1982.

[13] R.C. Burchett, H.H. Happ, and K.A. Wirgau. Large scale optimal power flow. *Power Apparatus and Systems, IEEE Transactions on*, PAS-101(10):3722–3732, Oct 1982.

[14] CA Canizares and F Alvarado. Uwpflow: continuation and direct methods to locate fold bifurcations in ac/dc/facts power systems. *University of Waterloo*, 1999.

[15] F. Capitanescu and L. Wehenkel. Sensitivity-based approaches for handling discrete variables in optimal power flow computations. *Power Systems, IEEE Transactions on*, 25(4):1780–1789, Nov 2010.

[16] Florin Capitanescu, Mevludin Glavic, Damien Ernst, and Louis Wehenkel. Interior-point based algorithms for the solution of optimal power flow problems. *Electric Power Systems Research*, 77:508 – 517, 2007.

[17] J Carpentier. Contribution to the economic dispatch problem. *Bulletin de la Societe Francoise des Electriciens*, 3(8):431–447, 1962.

[18] S.-K. Chang, F. Albuyeh, M.L. Gilles, G.E. Marks, and K. Kato. Optimal real-time voltage control. *Power Systems, IEEE Transactions on*, 5(3):750–758, Aug 1990.

[19] S.-K. Chang, F. Albuyeh, M.L. Gilles, G.E. Marks, and K. Kato. Optimal real-time voltage control. *Power Systems, IEEE Transactions on*, 5(3):750–758, Aug 1990.

[20] Y.-C. Chang. Multi-objective optimal thyristor controlled series compensator installation strategy for transmission system loadability enhancement. *Generation, Transmission Distribution, IET*, 8(3):552–562, March 2014.

[21] DP Chassin, K Schneider, and C Gerkensmeyer. Gridlab-d: An open-source power systems modeling and simulation environment. In *Transmission and Distribution Conference and Exposition, 2008. D. IEEE/PES*, pages 1–5. IEEE, 2008.

[22] Ying Chen and Chen Shen. A jacobian-free newton-gmres(m) method with adaptive preconditioner and its application for power flow calculations. *Power Systems, IEEE Transactions on*, 21(3):1096–1103, Aug 2006.

[23] A.S. Chieh, P. Panciatici, and J. Picard. Power system modeling in modelica for time-domain simulation. In *PowerTech, 2011 IEEE Trondheim*, pages 1–8, June 2011.

[24] G.C. Contaxis, C. Delkis, and G. Korres. Decoupled optimal load flow using linear or quadratic programming. *Power Systems, IEEE Transactions on*, 1(2):1–7, May 1986.

[25] J.C. Das. *Power System Analysis: Short-Circuit Load Flow and Harmonics*. Power Engineering (Willis). Taylor & Francis, 2002.

[26] A.T. Davda, B. Azzopardi, B.R. Parekh, and M.D. Desai. Dispersed generation enable loss reduction and voltage profile improvement in distribution network. *Power Systems, IEEE Transactions on*, 29(3):1242–1249, May 2014.

[27] H.W. Dommel and W.F. Tinney. Optimal power flow solutions. *Power Apparatus and Systems, IEEE Transactions on*, PAS-87(10):1866–1876, Oct 1968.

[28] RC Dugan. The open distribution system simulator (opendss), 2009.

[29] IEEE Task Force. Load representation for dynamic performance analysis of power systems. *Power Systems, IEEE Transactions on*, 8(2):472–482, May 1993.

[30] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. Optimal power flow: a bibliographic survey i. *Energy systems*, 3(3):259–289, 2012.

[31] A. Gabash and Pu Li. Flexible optimal operation of battery storage systems for energy supply networks. *Power Systems, IEEE Transactions on*, 28(3):2788–2797, Aug 2013.

[32] N. Garcia. Parallel power flow solutions using a biconjugate gradient algorithm and a newton method: A gpu-based approach. In *Power and Energy Society General Meeting, 2010 IEEE*, pages 1–4, July 2010.

[33] H. Glavitsch and M. Spoerry. Quadratic loss formula for reactive dispatch. *Power Apparatus and Systems, IEEE Transactions on*, PAS-102(12):3850–3858, Dec 1983.

[34] J.D. Glover, M. Sarma, and T. Overbye. *Power System Analysis & Design, SI Version*. Cengage Learning, 2011.

[35] A. Gomez-Exposito, A.J. Conejo, and C. Canizares. *Electric Energy Systems: Analysis and Operation*. Electric Power Engineering Series. Taylor & Francis, 2008.

[36] Jacek Gondzio. Multiple centrality corrections in a primal-dual method for linear programming. *Computational Optimization and Applications*, 6(2):137–156, 1996.

[37] S. Granville. Optimal reactive dispatch through interior point methods. *Power Systems, IEEE Transactions on*, 9(1):136–146, Feb 1994.

[38] A. Greenhall, R. Christie, and J.-P. Watson. Minpower: A power systems optimization toolkit. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–6, July 2012.

[39] Ro E Griffith and RA Stewart. A nonlinear programming technique for the optimization of continuous processing systems. *Management science*, 7(4):379–392, 1961.

[40] L.L. Grigsby. *Power Systems, Third Edition*. Number v. 4 in The electric power engineering handbook. Taylor & Francis, 2012.

[41] W. Group. Common format for exchange of solved load flow data. *Power Apparatus and Systems, IEEE Transactions on*, PAS-92(6):1916–1925, Nov 1973.

[42] N. Grudinin. Combined quadratic-separable programming opf algorithm for economic dispatch and security control. *Power Systems, IEEE Transactions on*, 12(4):1682–1688, Nov 1997.

[43] N. Grudinin. Reactive power optimization using successive quadratic programming method. *Power Systems, IEEE Transactions on*, 13(4):1219–1225, Nov 1998.

[44] Arnim Herbig. *On Load Flow Control in Electric Power Systems*. PhD thesis, Royal Institute of Technology Stockholm, 2000.

[45] E.C. Housos and G.D. Irisarri. A sparse variable metric optimization method applied to the solution of power system problems. *Power Apparatus and Systems, IEEE Transactions on*, PAS-101(1):195–202, Jan 1982.

[46] R. Idema, D.J.P. Lahaye, C. Vuik, and L. Van der Sluis. Scalable newton-krylov solver for very large power flow problems. *Power Systems, IEEE Transactions on*, 27(1):390–396, Feb 2012.

[47] IEEE. Test cases for power flow. http://www.ee.washington.edu/research/pstca/.

[48] R.A. Jabr. Optimal power flow using an extended conic quadratic formulation. *Power Systems, IEEE Transactions on*, 23(3):1000–1008, Aug 2008.

[49] R.A. Jabr. Minimum loss operation of distribution networks with photovoltaic generation. *Renewable Power Generation, IET*, 8(1):33–44, January 2014.

[50] R. Jegatheesan, N. Mohd Nor, and M.F. Romlie. Newton-raphson power flow solution employing systematically constructed jacobian matrix. In *Power and Energy Conference, 2008. PECon 2008. IEEE 2nd International*, pages 180–185, Dec 2008.

[51] Q.Y. Jiang, H.-D. Chiang, C.X. Guo, and Y.J. Cao. Power-current hybrid rectangular formulation for interior-point optimal power flow. *Generation, Transmission Distribution, IET*, 3(8):748–756, August 2009.

[52] Ricardo B. Ramkhelawan K. S. Sastry Musti. *Power System Load Flow Analysis using Microsoft Excel.* University of West Indies, 2012.

[53] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[54] B. Kazemtabrizi and E. Acha. A comparison study between mathematical models of static var compensators aimed at optimal power flow solutions. In *Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean*, pages 1125–1128, March 2012.

[55] SK Mena Kodsi and Claudio A Canizares. Modeling and simulation of ieee 14 bus system with facts controllers. *University of Waterloo, Canada, Tech. Rep*, 2003.

[56] J. Lavaei and S.H. Low. Zero duality gap in optimal power flow problem. *Power Systems, IEEE Transactions on*, 27(1):92–107, Feb 2012.

[57] C. Lehmkoster. Security constrained optimal power flow for an economical operation of facts-devices in liberalized energy markets. *Power Delivery, IEEE Transactions on*, 17(2):603–608, Apr 2002.

[58] F. G M Lima, F.D. Galiana, I. Kockar, and J. Munoz. Phase shifter placement in large-scale systems via mixed integer linear programming. *Power Systems, IEEE Transactions on*, 18(3):1029–1034, Aug 2003.

[59] Whei-Min Lin, Cong-Hui Huang, and Tung-Sheng Zhan. A hybrid current-power optimal power flow technique. *Power Systems, IEEE Transactions on*, 23(1):177–185, Feb 2008.

[60] E. Lobato, R. Rouco, M. I. Navarrete, R. Casanova, and G. Lopez. An lp-based optimal power flow for transmission losses and generator reactive margins minimization. In *Power Tech Proceedings, 2001 IEEE Porto*, volume 3, pages 5 pp. vol.3–, 2001.

[61] Sanjay Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on optimization*, 2(4):575–601, 1992.

[62] F. Milano. An open source power system analysis toolbox. *Power Systems, IEEE Transactions on*, 20(3):1199–1206, Aug 2005.

[63] F. Milano. Continuous newton's method for power flow analysis. *Power Systems, IEEE Transactions on*, 24(1):50–57, Feb 2009.

[64] F. Milano. *Power System Modelling and Scripting.* Power Systems. Springer, 2010.

[65] F. Milano. A python-based software tool for power system analysis. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pages 1–5, July 2013.

[66] F. Milano and L. Vanfretti. State of the art and future of oss for power systems. In *Power Energy Society General Meeting, 2009. PES '09. IEEE*, pages 1–7, July 2009.

[67] W. Min and L. Shengsong. A trust region interior point algorithm for optimal power flow problems. *International Journal of Electrical Power & Energy Systems*, 27(4):293–300, May 2005.

[68] Gustavo Adolfo Valverde Mora. *Uncertainty and state estimation of power systems.* PhD thesis, School of Electrical and Electronic Engineering University of Manchester, 2012.

[69] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[70] A. Ouammi, H. Dagdougui, and R. Sacile. Optimal control of power flows and energy local storages in a network of microgrids modeled as a system of systems, 2014.

[71] J. Peschon, D.W. Bree, and L.P. Hajdu. Optimal power-flow solutions for power system planning. *Proceedings of the IEEE*, 60(1):64–70, Jan 1972.

[72] N.V. Ramana and R.N. V. *Power System Analysis.* Pearson, 2011.

[73] B.S. Rao and K. Vaisakh. Application of acsa to solve single/multi objective opf problem with facts devices. In *Advanced Computing Technologies (ICACT), 2013 15th International Conference on*, pages 1–6, Sept 2013.

[74] N.S. Rau. Issues in the path toward an rto and standard markets. *Power Systems, IEEE Transactions on*, 18(2):435–443, May 2003.

[75] W.D. Rosehart, C.A. Canizares, and A. Vannelli. Sequential methods in solving economic power flow problems. In *Electrical and Computer Engineering, 1997. Engineering Innovation: Voyage of Discovery. IEEE 1997 Canadian Conference on*, volume 1, pages 281–284 vol.1, May 1997.

[76] D. Saha and S. Singha. A new approach to load flow analysis using krylov subspace methods for well conditioned systems. In *India Conference (INDICON), 2011 Annual IEEE*, pages 1–6, Dec 2011.

[77] A.M. Sasson, F. Viloria, and F. Aboytes. Optimal load flow solution using the hessian matrix. *Power Apparatus and Systems, IEEE Transactions on*, PAS-92(1):31–41, Jan 1973.

[78] Siemens Power Transmission and Distribution. *PSS/E-31.0 Users Manual Managing Power Flow Data.*

[79] E.M. Soler, E.N. Asada, and G.R.M. da Costa. Penalty-based nonlinear solver for optimal reactive power dispatch with discrete controls. *Power Systems, IEEE Transactions on*, 28(3):2174–2182, Aug 2013.

[80] M. Soroush and J.D. Fuller. Accuracies of optimal transmission switching heuristics based on dcopf and acopf. *Power Systems, IEEE Transactions on*, 29(2):924–932, March 2014.

[81] A.A. Sousa, G.L. Torres, and C.A. Canizares. Robust optimal power flow solution using trust region and interior-point methods. *Power Systems, IEEE Transactions on*, 26(2):487–499, May 2011.

[82] B. Stott and Eric Hobson. Power system security control calculations using linear programming, part i. *Power Apparatus and Systems, IEEE Transactions on*, PAS-97(5):1713–1720, Sept 1978.

[83] B. Stott, J. Jardim, and O. Alsac. Dc power flow revisited. *Power Systems, IEEE Transactions on*, 24(3):1290–1300, Aug 2009.

[84] D.I. Sun, Bruce Ashley, Brian Brewer, Art Hughes, and William F. Tinney. Optimal power flow by newton approach. *Power Apparatus and Systems, IEEE Transactions on*, PAS-103(10):2864–2880, Oct 1984.

[85] Junjie Sun and Leigh Tesfatsion. Dynamic testing of wholesale power market designs: An open-source agent-based framework. *Computational Economics*, 30(3):291–327, 2007.

[86] William F. Tinney and C.E. Hart. Power flow solution by newton's method. *Power Apparatus and Systems, IEEE Transactions on*, PAS-86(11):1449–1460, Nov 1967.

[87] G.L. Torres and V.H. Quintana. On a nonlinear multiple-centrality-corrections interior-point method for optimal power flow. *Power Systems, IEEE Transactions on*, 16(2):222–228, May 2001.

[88] G.L. Torres and V.H. Quintana. On a nonlinear multiple-centrality-corrections interior-point method for optimal power flow. *Power Systems, IEEE Transactions on*, 16(2):222–228, May 2001.

[89] X.F. Wang, Y. Song, and M. Irving. *Modern Power Systems Analysis*. Power electronics and power systems. Springer, 2010.

[90] Wolf. *Nonlinear Programming*. Wiley and Sons, 1967.

[91] A.J. Wood, B.F. Wollenberg, and G.B. Sheblé. *Power Generation, Operation and Control*. Wiley, 1996.

[92] Wei Xu and Wei Li. Efficient preconditioners for newton-gmres method with application to power flow study. *Power Systems, IEEE Transactions on*, 28(4):4173–4180, Nov 2013.

[93] W. Yao, J. Zhao, F. Wen, Z. Dong, Y. Xue, Y. Xu, and K. Meng. A multi-objective collaborative planning strategy for integrated power distribution and electric vehicle charging systems, 2014.

[94] T. Zabaiou, L.-A. Dessaint, and I. Kamwa. Preventive control approach for voltage stability improvement using voltage stability constrained optimal power flow based on static line voltage stability indices. *Generation, Transmission Distribution, IET*, 8(5):924–934, May 2014.

[95] Wenjuan Zhang and L.M. Tolbert. Survey of reactive power planning methods. In *Power Engineering Society General Meeting, 2005. IEEE*, pages 1430–1440 Vol. 2, June 2005.

[96] X.P. Zhang, C. Rehtanz, and B. Pal. *Flexible AC Transmission Systems: Modelling and Control: Modelling and Control*. Power systems. Physica-Verlag, 2006.

[97] M. Zhou and Shizhao Zhou. Internet, open-source and power system simulation. In *Power Engineering Society General Meeting, 2007. IEEE*, pages 1–5, June 2007.

[98] Jizhong Zhu. *Optimization of power system operation*, volume 49. John Wiley & Sons, 2009.

[99] R.D. Zimmerman, C.E. Murillo-Sanchez, and R.J. Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *Power Systems, IEEE Transactions on*, 26(1):12–19, Feb 2011.