Chapter 3: Solving Power Flow Equations

In the last chapter the Power flow equations (PFE’s) were introduced. In this chapter the methods of solving these PFE’s will be discussed.

The problem of solving PFE’s boils down to solving non-linear equations. So the set of equations which have to be solved numerically are

\[ F(X) = 0 \]

\( X \) has the dimension 2n for a n bus system

i.e \[ f_1(X) = 0, f_2(X) = 0, \ldots, f_{2n}(X) = 0. \]

For a PV bus \( Q \) & \( \delta \) are unknown, for a PQ bus \( V \) & \( \delta \) are unknown while for the slack bus \( P \) & \( Q \) are to be found out.

Since the set of 2n equations are inconsistent as has already been discussed in the last chapter, the method of solving PFE’s is to first solve a set of (2n-2) set of real and consistent equations and at the end find out \( P \) & \( Q \) for the slack bus using the power balance equations.

The question for us to answer now is how to solve in general \( f(X) = 0 \), a set of nonlinear equations.

3.1 Review of Gauss related Methods:

**Idea:**

Solve \( f(X) = 0 \)

Rewrite the equations into an equivalent set of equations of the form

\[ X = g(X) \]

Thus if one solves the equations \( X = g(X) \), effectively one has solved the original set of equations, i.e. \( f(X) = 0 \)

Gauss Method is iterative in nature and can be expressed as follows

\[ X_{k+1} = g(X_k) \]

given an initial guess \( X(0) \)

If \( X_{k+1} \) converges i.e. \( X_{k+1} \) is approximately equal to \( g(X_k) \), then we get a solution.

\[ X_k = g(X_k) \Rightarrow f(X_{k+1}) = f(X_k) = 0 \]

**Example:**

Let \[ 2X_1 + X_1 X_2 - 1 = 0 \]
\[ 2X_2 - X_1 X_2 + 1 = 0 \]

Rearranging

\[ X_1 = 0.5 - X_1 X_2 / 2 \]
\[ X_2 = -0.5 + X_1 X_2 / 2 \]

Let initial guess be \( X_1(0) = 0 \), \( X_2(0) = 0 \), which is called as the FLAT START.

Iteration \( K = 1 \)

\[ X_1^{(1)} = 0.5, \quad X_2^{(1)} = -0.5 \]

Iteration \( K = 2 \)

\[ X_1^{(2)} = 0.5 + 0.125 = 0.625 \]
\[ X_2^{(2)} = -0.5 - 0.125 = -0.625 \]

& so on until the solution converges.

However, two points are to be noted. Firstly, that there are several ways to rewrite \( f(X) = 0 \) into \( X = g(X) \). Thus selection of function \( g(X) \) is not unique.

For the above example another possible function for rearranging \( f(X) \) could have been as follows

\[
X_1 = (1 - 2X_1)/X_2 \quad \text{&} \quad X_2 = (2X_2 + 1)/X_1
\]

In this case flat start is not possible as the possible solution diverges. This brings out the second point that it is not necessary that all the initial conditions work which implies a sort of a convergence region. Thus the initial condition chosen must be in the convergence region.

For a function \( f(X) = X^2 - 5X + 4 = 0 \) the solution is +4 and +1.

Let us rearrange this equation to the form,

\[
X = X^2/5 + 4/5.
\]

It is found that there is convergence region \((-a, 4)\) around the solution \( X = 1 \). However there is no convergence around the solution \( X = 4 \), if we try to solve this iteratively.

### 3.2 Gauss-Seidal Method.

The idea is that once a variable is calculated corresponding to the gauss method, we use this calculated value of the variable immediately in the iterative equation of the next variable.

\[
i.e \quad X_{1}^{(k+1)} = g_1(X_{1}^{(k)}, \ldots, X_{n}^{(k)})
\]

\[
X_{2}^{(k+1)} = g_2(X_{1}^{(k+1)}, X_{2}^{(k)}, \ldots, X_{n}^{(k)})
\]

\[
X_{i}^{(k+1)} = g_i(X_{i}(k+1), \ldots, X_{i-1}^{(k+1)}, \ldots, X_{i}^{(k)}, \ldots, X_{n}^{(k)})
\]

The motivation is to save the memory space. It is found that gauss-seidal method converges faster than the Gauss method.

**Example:**

Solve

\[
2X_1 + X_1X_2 - 1 = 0
\]

\[
2X_2 - X_1X_2 + 1 = 0
\]

Rearranging,

\[
X_1 = 0.5 - (X_1X_2 / 2)
\]

\[
X_2 = -0.5 + (X_1X_2 / 2)
\]

Let the Initial Guess be \( X_1^{(0)} = 0, X_2^{(0)} = 0 \)

Applying Gauss-seidal method

\[
X_1^{(1)} = 0.5, \quad X_2^{(1)} = -0.5
\]

\[
X_1^{(2)} = 0.625, \quad X_2^{(2)} = -0.656
\]

\[
X_1^{(3)} = 0.705, \quad X_2^{(3)} = -0.731
\]
Comparing the corresponding values of the states with the Gauss method, it can be seen distinctly that Gauss-seidal method converges faster to the final solution. This holds true for linear equations that are diagonally dominant.

**BONUS QUESTION**
For Linear equations, is Gauss Elimination Method better than iterative methods? Why? Give suitable examples to demonstrate.

**Remarks:**

1) Jacobi Method is parallelizable for parallel processing

\[ X_i^{(k+1)} = g_i(X_1^{(k)}, \ldots, X_n^{(k)}) \]

\[ X_1^{(k+1)} = g_1(X_1^{(k)}, \ldots, X_n^{(k)}) \]

All equations in the above case can be computed concurrently. The speeding up of the computation is by a factor of \( n \), where \( n \) is the number of equations, provided there are no other overheads.

**BONUS QUESTION**
Is Gauss-Seidel Parallelizable?

### 3.3 Solving PFE's using Gauss method and Gauss seidel method.

To simplify our formulation in the beginning let us consider only PQ (Load) buses only. Later on we shall introduce the PV (Generator) buses. Of course the slack bus concept, where magnitude and angle are known, would exist in all cases of PFE solution.

We know,

\[ I_{bus} = Y_{bus} \times E_{bus} \]

Considering the \( p^{th} \) bus,

\[ I_p = (P_p - jQ_p) / E_p^* \] for all \( p=1,2,\ldots,n \) excepting slack bus i.e \( p \neq s \)

\[ I_p = \sum_{q=1}^{n} Y_{pq} E_q \]

\[ = Y_{pp} E_p + \sum_{q=1}^{n} \sum_{q \neq p} Y_{pq} E_q \]
Thus \[ E_p = \left[ I_p - \sum_{q=1}^{n} Y_{pq} E_q \right] / Y_{pp} \]

The iterative algorithm to solve the above using Gauss Method would be,

\[
E_p^{(k+1)} = \frac{1}{Y_{pp}} \left[ \frac{P_p - jQ_p}{E_p^{(k)}} - \sum_{q=1}^{n} Y_{pq} E_q^{(k+1)} - \sum_{q=p+1}^{n} Y_{pq} E_q^{k} \right]
\]

Here \( p = 1, 2, \ldots, n \) and \( p \neq s \)

Using the Gauss-Seidel approach the algorithm can be expressed as

\[
E_p^{(k+1)} = \frac{1}{Y_{pp}} \left[ \frac{P_p - jQ_p}{E_p^{(k)}} - \sum_{q=1}^{p-1} Y_{pq} E_q^{(k+1)} - \sum_{q=p+1}^{n} Y_{pq} E_q^{k} \right]
\]

Here \( p = 1, 2, \ldots, n \) and \( p \neq s \)

**Remark:**

1) Note that \( Y_{pp}, P_p, Q_p \) are known constants. Hence the term \( \left[ \frac{P_p - jQ_p}{Y_{pp}} \right] \) need not be computed repeatedly. Hence one may substitute \( K_{l_p} = \left[ \frac{P_p - jQ_p}{Y_{pp}} \right] \).

2) \( Y_{pq}/Y_{pp} \) can be computed once and substituted in every iteration. Take \( Y_{Lpq} = Y_{pq}/Y_{pp} \)

Now let us consider the generation bus

Here \( P \) and \(|E|\) are known, while \( Q \) and \( \delta \) are the unknowns.

Let

\[
E_p = e_p + j f_p \\
E_q = e_q + j f_q \\
Y_{pq} = g_{pq} + j b_{pq}
\]

\[
P_p + jQ_p = (e_p + jf_p) \sum_{q=1}^{n} (g_{pq} - jb_{pq})(e_q - jf_q)
\]

Solving this expression we end up with the following

\[
Q_p = - \left( e_p^2 + f_p^2 \right) b_{pp} + \sum_{q=1}^{n} -e_p(b_{pq}e_q + g_{pq}f_q) + f_p(g_{pq}e_q - b_{pq}f_q)
\]

Here \( e_p^2 + f_p^2 = [E_{schedule}]^2 \)
For the Gauss Method of solution the iterative equation would be
\[ Q_p^{(k+1)} = -(e_p^{2(k+1)} + f_p^{2(k+1)})b_p + \sum_{q=1}^{n} \left[ -e_p^{(k+1)}(b_{pq}e_q^{(k+1)} + g_{pq}f_q^{(k+1)}) + f_p^{(k+1)}(g_{pq}e_q^{(k+1)} - b_{pq}f_q^{(k+1)}) \right] \]

**Question:** How shall we put this expression into an iterative form?

Let \( e_p^{(k+1)} \) and \( f_p^{(k+1)} \) be the current estimates of \( e_p \) and \( f_p \) using as PQ bus iteration.

The procedure of adjusting \( e_p^{(k+1)} \) and \( f_p^{(k+1)} \) to give the desired magnitude is as follows

\[ \delta_p^{k+1} = \tan^{-1} \left[ \frac{f_p^{k+1}}{e_p^{k+1}} \right] \]

Assume that this angle is correct. Therefore,

\[ e_{p_{new}}^{k+1} = |E_p| \cos(\delta_p^{k+1}) \]
\[ = |E_p| \frac{e_p^{k+1}}{\sqrt{(e_p^{2k+1})^2 + (f_p^{2k+1})^2}} \]

Similarly

\[ f_{p_{new}}^{k+1} = |E_p| \sin(\delta_p^{k+1}) \]
\[ = |E_p| \frac{f_p^{k+1}}{\sqrt{(e_p^{2k+1})^2 + (f_p^{2k+1})^2}} \]

\[ E_{p_{new}}^{k+1} = e_{p_{new}}^{k+1} + jf_{p_{new}}^{k+1} \]

The flowchart for the Gauss-Seidel Method is given at the end of the chapter.

### 3.4 Newton-Raphson Method

**Idea:**

If we have to solve \( f(X) = 0 \)

Let the initial guess be such that \( f(X^{(0)}) \neq 0 \)

Let us find \( \Delta X \), such that \( f(X^{(0)} + \Delta X) \approx 0 \)

So that we can approximate \( f(X^{(0)} + \Delta X) \) by Taylor's theorem

\[ f(X + \Delta X) = f(X^{(0)}) + \frac{\partial f}{\partial X} \bigg|_{X_0} \Delta X \approx 0 \]

Rearranging the equation, we get

\[ \Delta X^{(0)} = -\left( \frac{\partial f}{\partial X} \right)^{-1}_{X_0} f(X_0) \]
The new value is given by  \[ X^{(1)} = X^{(0)} + \Delta X^{(0)} \]

The iterative expression is given by
\[
\Delta X^{(k)} = \left( \frac{\partial f}{\partial X} \right)^{-1}(X^k) f(X^k)
\]

\[ X^{(k+1)} = X^{(k)} + \Delta X^{(k)} \]

The stopping condition of the iterative loop can be either of the below two:

1. \[ \|\Delta X^{(k)}\| < \varepsilon \] This is normally used.
2. However \[ f(X) \leq \varepsilon \] can also be used.

**Example**

\[ 2X_1 + X_1X_2 - 1 = 0 \quad \text{----------} \quad \text{Let this be function } f_1 \]
\[ 2X_2 - X_1X_2 + 1 = 0 \quad \text{----------} \quad \text{And this be function } f_2 \]

Let us find the Jacobian Matrix

\[
J = \left[ \begin{array}{cc}
\frac{\partial f_1}{\partial X_1} & \frac{\partial f_1}{\partial X_2} \\
\frac{\partial f_2}{\partial X_1} & \frac{\partial f_2}{\partial X_2}
\end{array} \right]
\]

\[
J = \left[ \begin{array}{cc}
2 + X_2 & X_1 \\
-X_2 & 2 - X_1
\end{array} \right]
\]

**Ist Iteration**

\[
J^{(0)} = \left[ \begin{array}{cc}
2 & 0 \\
0 & 2
\end{array} \right] \quad f(X^0) = \left[ \begin{array}{c}
-1 \\
1
\end{array} \right] \quad \Delta X^0 = \left[ \begin{array}{c}
0.5 \\
-0.5
\end{array} \right] \quad X^1 = \left[ \begin{array}{c}
0.5 \\
-0.5
\end{array} \right]
\]

**IInd Iteration**

\[
J^{(1)} = \left[ \begin{array}{cc}
1.5 & 0.5 \\
0.5 & 1.5
\end{array} \right] \quad f(X^1) = \left[ \begin{array}{c}
-0.25 \\
0.25
\end{array} \right] \quad \Delta X^1 = \left[ \begin{array}{c}
0.25 \\
-0.25
\end{array} \right] \quad X^2 = \left[ \begin{array}{c}
0.75 \\
-0.75
\end{array} \right]
\]

**Remark:**
1. Newton-Raphson's method seems "faster" in the sense that it converges to the solution faster in lesser number of iterations.
2) However, it is to be noted that for each iteration it involves matrix inversion which might be time consuming.

3) Mathematically it is proven that under certain conditions N-R has smaller iteration complexity. However, our real concern is Total complexity which is nothing but the product of iteration complexity and per iteration complexity.

### 3.5 Newton-Raphson in Rectangular Form

Recall the following equations:

\[ P_p + jQ_p = E_p \hat{I}_p \]

\[ = E_p \left( \sum_{q=1}^{n} Y_{pq} E_q^* \right) \]

Let \( E_p = e_p + j f_p \)
\[ Y_{pq} = g_{pq} + j b_{pq} \]

Thus,

\[ P_p + jQ_p = (e_p + j f_p) \sum_{q=1}^{n} (g_{pq} - j b_{pq}) (e_q - j f_q) \]

\[ P_p = \sum_{q=1}^{n} \left\{ e_p (e_q g_{pq} - f_q b_{pq}) + f_p (f_q g_{pq} + e_q b_{pq}) \right\} \]

\[ \cong \Phi(e, f) \]

\[ Q_p = \sum_{q=1}^{n} \left\{ f_p (e_q g_{pq} - f_q b_{pq}) - e_p (f_q g_{pq} + e_q b_{pq}) \right\} \]

\[ \cong \Psi(e, f) \]

At every load bus we can linearize around an operating point given by \( e^{(0)}, f^{(0)} \)

Expanding by Taylor’s expansion

\[ P_i = \Phi_i(e_1^{(0)} + \Delta e_1, \ldots, e_n^{(0)} + \Delta e_n, \ldots) \]

\[ = \Phi_i(e + \Delta e, f + \Delta f) \]

Assume bus \( n \) to be the slack bus. Thus \( \Delta e_n = 0 \) and \( \Delta f_n = 0 \)

In general we can write

\[ P_i = \Phi_i(e^{(0)} + \Delta e_1, \ldots, e^{(0)} + \Delta e_n, \ldots) \]

\[ = \Phi_i(e + \Delta e, f + \Delta f) \]
\[ \Delta P_i = P_i - \Phi_i(e^{(0)}, \bar{f}^{(0)}) \]
\[ = \sum_{q=1}^{n-1} \frac{\partial \Phi_i}{\partial e_q} \Delta e_q + \sum_{q=1}^{n-1} \frac{\partial \Phi_i}{\partial f_q} \Delta f_q \]

Similarly,
\[ \Delta Q_i = \sum_{q=1}^{n-1} \frac{\partial \Psi_i}{\partial e_q} \Delta e_q + \sum_{q=1}^{n-1} \frac{\partial \Psi_i}{\partial f_q} \Delta f_q \]

Rewriting the above results in a matrix form,

\[
\begin{bmatrix}
\Delta P_1 \\
\vdots \\
\Delta P_{i} \\
\Delta P_{n-1}
\end{bmatrix} = \begin{bmatrix}
\cdots \\
\frac{\partial \Phi_i}{\partial e_q} & \frac{\partial \Phi_i}{\partial f_q} \\
\cdots \\
\frac{\partial \Psi_i}{\partial e_q} & \frac{\partial \Psi_i}{\partial f_q}
\end{bmatrix} \begin{bmatrix}
\Delta e_1 \\
\Delta e_i \\
\Delta e_{n-1}
\end{bmatrix} + \begin{bmatrix}
\Delta f_1 \\
\Delta f_i \\
\Delta f_{n-1}
\end{bmatrix}
\]

**Question:** What are these matrix entries and how to get them?

**Ans:** The entries in the above matrix are found as follows:
All **diagonal elements** of terms involving \( \Phi \) & \( e \) are defined as:
At \( i = p \)
\[
\frac{\partial \Phi_p}{\partial e_p} = 2e_p g_{pp} + \sum_{q=1 \atop q \neq p}^{n} (e_q g_{pq} - f_q b_{pq}) \quad \text{(1)}
\]

Now, some simplifications can be considered.

Let us define the term \( I_p = c_p + j d_p \)
The term \( I_p \) at each bus can be got from \([(P_p - j Q_p) / E_p^*] \)

Thus
\[
c_p + j d_p = \sum_{q=1}^{n} Y_{pq} * E_q
\]

\[
= (e_p + j f_p) (g_{pp} + j b_{pp}) + \sum_{q=1 \atop q \neq p}^{n} Y_{pq} * E_q
\]

Equating the real and imaginary quantities together, we get

\[
c_p = g_{pp} e_p - b_{pp} f_p + \sum_{q=1 \atop q \neq p}^{n} (g_{pq} e_q - b_{pq} f_q)
\]

\[
d_p = g_{pp} f_p + b_{pp} e_p + \sum_{q=1 \atop q \neq p}^{n} (g_{pq} f_q + b_{pq} e_q)
\]

Comparing equation (1) and (2) we get

\[
\frac{\partial \Phi_p}{\partial e_p} = 2e_p g_{pp} + c_p - g_{pp} e_p + b_{pp} f_p
\]

\[
= c_p + g_{pp} e_p + b_{pp} f_p
\]

By this approximations we can save some computations

All **off-diagonal** elements of terms involving \( \Phi & e \) are defined as:

\[
\frac{\partial \Phi_p}{\partial e_q} = e_p g_{pq} + b_{pq} f_p \quad \text{Here} \quad q \neq p
\]

All **diagonal** elements of terms involving \( \Phi & f \) are defined as:

\[
\frac{\partial \Phi_p}{\partial f_p} = 2f_p g_{pp} + \sum_{q=1 \atop q \neq p}^{n} (f_q g_{pq} + e_q b_{pq})
\]

Carrying out the similar exercise for approximation as in the previous case we get,
\[
\frac{\partial \Phi_p}{\partial f_p} = d_p + f_p g_{pp} - b_{pp} e_p
\]

The **off-diagonal** elements of terms involving \(\Phi\) & \(f\) are defined as:

\[
\frac{\partial \Phi_p}{\partial f_q} = -e_p b_{pq} + g_{pq} f_p \quad \text{Here} \quad q \neq p
\]

Similarly, for the terms involving \(\Psi\) the terms are follows. The reader may derive these terms on the similar lines as discussed above for \(\Phi\) terms.

\[
\frac{\partial \Psi_p}{\partial e_p} = -d_p + f_p g_{pp} - b_{pp} e_p
\]

\[
\frac{\partial \Psi_p}{\partial e_q} = -e_p b_{pq} + g_{pq} f_p \quad \text{Here} \quad q \neq p, s
\]

\[
\frac{\partial \Psi_p}{\partial f_p} = c_p - f_p b_{pp} - g_{pp} e_p
\]

\[
\frac{\partial \Psi_p}{\partial f_q} = -e_q g_{pq} - b_{pq} f_p \quad \text{Here} \quad q \neq p, s
\]

Let us now enlist a step by step solution procedure

1) Given a set of bus voltages

\[
(e^{(k)}, f^{(k)})
\]

2) Compute bus powers
$$P_p^{(k)} = \Phi_p(e^{(k)}, \bar{f}^{(k)})$$
$$Q_p^{(k)} = \Psi_p(e^{(k)}, \bar{f}^{(k)})$$

3) Compute the change in P’s and Q’s

$$\Delta P_p = P_p - P_p^{(k)}$$
$$\Delta Q_p = Q_p - Q_p^{(k)}$$

These conditions can be used for convergence checking also

4) Compute the bus currents

$$I_p^{(k)} = \frac{P_p - Q_p}{E^*_p} = c_p + j d_p$$

$$c_p = \text{Re}(I_p^{(k)})$$
$$d_p = \text{Im}ag(I_p^{(k)})$$

5) Update the Jacobian matrix elements

6) Solve for $\Delta \bar{e}$ & $\Delta \bar{f}$

7) Update the voltage estimates, that is

$$e^{(k+1)} = e^{(k)} + \Delta e^{(k)}$$
$$\bar{f}^{(k+1)} = \bar{f}^{(k)} + \Delta \bar{f}^{(k)}$$

8) GOTO step 2 until the solution converges.

Newton Raphson’s Method to deal with Generation bus:

$$P_p = \sum_{q=1}^{n} \left\{ e_p (e_q g_{pq} - f_q b_{pq}) + f_p (f_q g_{pq} + e_q b_{pq}) \right\}$$

$$\approx \Phi(e, f)$$

$$\left| E_p \right|_{\text{sched}}^2 = e_p^2 + f_p^2 \approx \Psi(e, f)$$
Expanding the above by Taylor’s series expression, we get

\[ \Delta P_p^{(k)} = P_p - \Phi(e_p^{(k)}, f_p^{(k)}) \]

\[ = \sum_{q=1}^{n-1} \frac{\partial \Phi}{\partial e_q} \Delta e_q^{(k)} + \sum_{q=1}^{n-1} \frac{\partial \Phi}{\partial f_q} \Delta f_q^{(k)} \]

\[ |E_p|^2_{\text{sched}} = (e_p^{(k)})^2 + (f_p^{(k)})^2 + \frac{\partial \Psi}{\partial e_p} \Delta f_p^{(k)} \]

\[ |E_p|^2_{\text{sched}} = (e_p^{(k)})^2 + (f_p^{(k)})^2 + 2e_p^{(k)} \Delta e_p^{(k)} + 2f_p^{(k)} \Delta f_p^{(k)} \]

Thus,

\[ \Delta |E_p|^2 \cong |E_p|^2_{\text{sched}} = (e_p^{(k)})^2 + (f_p^{(k)})^2 - 2e_p^{(k)} \Delta e_p^{(k)} - 2f_p^{(k)} \Delta f_p^{(k)} \]

This can be represented in the form of a matrix as follows:

\[
\begin{bmatrix}
\Delta P \\
\Delta Q \\
\Delta P \\
\Delta |E|^2
\end{bmatrix} = 
\begin{bmatrix}
\text{Load} \\
\text{Buses} \\
\text{Generator} \\
\text{Buses}
\end{bmatrix}
\begin{bmatrix}
\text{Jacobian Matrix} \\
\end{bmatrix}
\begin{bmatrix}
\Delta e \\
\Delta f
\end{bmatrix}
\]

Newton-raphson Method in Polar Form

\[ E_p = |E_p| e^{j\delta_p} \]

\[ Y_{pq} = |Y_{pq}| e^{j\theta_{pq}} \]

Now,
\[ P_p + jQ_p = E_p * Y_p^* \]
\[ = E_p \left( \sum_{q=1}^{n} Y_{pq}^* E_q^* \right) \]

\[ P_p = \sum_{q=1}^{n} |E_p||Y_{pq}||E_q| \text{ Cos(} \delta_p - \delta_q - \theta_{pq} \text{)} \quad \text{Let this be } \Phi(|E|, \delta) \]

\[ Q_p = \sum_{q=1}^{n} |E_p||Y_{pq}||E_q| \text{ Sin(} \delta_p - \delta_q - \theta_{pq} \text{)} \quad \text{Let this be } \Psi(|E|, \delta) \]

Expanding by Taylor’s expansion

\[ \Delta P_p^{(k)} = P_p - \Phi(|E|^{(k)}, \delta^{(k)}) \]
\[ = \sum_{q=1}^{n-1} \frac{\partial \Phi_p}{\partial |E_q|} \bigg|_k \Delta |E_q^{(k)}| + \sum_{q=1}^{n-1} \frac{\partial \Phi_p}{\partial \delta_q} \bigg|_k \Delta \delta_q^{(k)} \]

\[ \Delta Q_p^{(k)} = Q_p - \Psi(|E|^{(k)}, \delta^{(k)}) \]
\[ = \sum_{q=1}^{n-1} \frac{\partial \Psi_p}{\partial |E_q|} \bigg|_k \Delta |E_q^{(k)}| + \sum_{q=1}^{n-1} \frac{\partial \Psi_p}{\partial \delta_q} \bigg|_k \Delta \delta_q^{(k)} \]

Thus

\[ \Phi_p = |E_p|^2 Y_{pp} \text{ Cos(} - \theta_{pp} \text{)} + \sum_{q=1}^{n} Y_{pq} E_q |E_q| \text{ Cos(} \delta_p - \delta_q - \theta_{pq} \text{)} \]

The Jacobian terms can be got as follows

\[ \frac{\partial \Phi_p}{\partial |E_p|} = 2E_p Y_{pp} \text{ Cos(} \theta_{pp} \text{)} + \sum_{q=1}^{n} E_q Y_{pq} |E_q| \text{ Cos(} \delta_p - \delta_q - \theta_{pq} \text{)} \]
\[ \frac{\partial \phi_p}{\partial |E_q|} = |E_p| Y_{pq} \left| \cos(\delta_p - \delta_q - \theta_{pq}) \right| \]

Here \( q \neq s, p \)

\[ \frac{\partial \phi_p}{\partial \delta_p} = -\sum_{q=1}^{n} |E_p| E_q Y_{pq} \left| \sin(\delta_p - \delta_q - \theta_{pq}) \right| \]

\[ \frac{\partial \phi_q}{\partial \delta_q} = |E_p| E_q Y_{pq} \left| \sin(\delta_p - \delta_q - \theta_{pq}) \right| \]

Here \( q \neq s, p \)

Similarly, the second function is represented by,

\[ \Psi_p = |E_p|^2 Y_{pp} \left| \sin(-\theta_{pp}) + \sum_{q=1}^{n} |E_p| E_q Y_{pq} \left| \sin(\delta_p - \delta_q - \theta_{pq}) \right| \right| \]

The Jacobian terms for this function are

\[ \frac{\partial \Psi_p}{\partial |E_p|} = -2 |E_p| Y_{pp} \left| \sin(\theta_{pp}) + \sum_{q=1}^{n} E_q Y_{pq} \left| \sin(\delta_p - \delta_q - \theta_{pq}) \right| \right| \]

\[ \frac{\partial \Psi_p}{\partial |E_q|} = |E_p| \left| Y_{pq} \right| \left| \sin(\delta_p - \delta_q - \theta_{pq}) \right| \]

Here \( q \neq s, p \)

\[ \frac{\partial \Psi_p}{\partial \delta_p} = -\sum_{q=1}^{n} E_p E_q Y_{pq} \left| \cos(\delta_p - \delta_q - \theta_{pq}) \right| \]

\[ \frac{\partial \Psi_p}{\partial \delta_q} = -|E_p| E_q Y_{pq} \left| \cos(\delta_p - \delta_q - \theta_{pq}) \right| \]

Here \( q \neq s, p \)

3.5 Stott’s Fast Decoupled Method:

Let us begin writing the power balance equations starting from

\[ E_p = |E_p| e^{j\delta_p} \]

\[ Y_{pq} = g_{pq} + j b_{pq} \]
Therefore,
\[ P_p + jQ_p = E_p \ast I_p^* = E_p \left( \sum_{q=1}^{n} Y_{pq} E_q^* \right) \]

\[ P_p + jQ_p = |E_p| \sum_{q=1}^{n} (g_{pq} - jb_{pq}) |E_q| \left[ \cos(\delta_p - \delta_q) + j\sin(\delta_p - \delta_q) \right] \]

Equating real and imaginary parts separately we get

\[ P_p = \Phi_p(\|E_p\|, \delta_p) = |E_p| \sum_{q=1}^{n} \left[ g_{pq} \cos(\delta_p - \delta_q) + b_{pq} \sin(\delta_p - \delta_q) \right] \]

\[ Q_p = \Psi_p(\|E_p\|, \delta_p) = |E_p| \sum_{q=1}^{n} \left[ -b_{pq} \cos(\delta_p - \delta_q) + g_{pq} \sin(\delta_p - \delta_q) \right] \]

Applying Taylor’s expansion,

\[ \Delta P_p^{(k)} = P_p - \Phi(\|E_p^{(k)}\|, \delta_p^{(k)}) \]

\[ = \sum_{q=1}^{n} \frac{\partial \Phi_p}{\partial |E_q|} \Delta |E_q^{(k)}| + \sum_{q=1}^{n} \frac{\partial \Phi_p}{\partial \delta_q} \Delta \delta_q^{(k)} \]

\[ \Delta Q_p^{(k)} = Q_p - \Psi(\|E_p^{(k)}\|, \delta_p^{(k)}) \]

\[ = \sum_{q=1}^{n} \frac{\partial \Psi_p}{\partial |E_q|} \Delta |E_q^{(k)}| + \sum_{q=1}^{n} \frac{\partial \Psi_p}{\partial \delta_q} \Delta \delta_q^{(k)} \]

We can rewrite this as follows

\[ \Delta P_p^{(k)} = \sum_{q=1}^{n} \left| E_q \right| x \frac{\partial \Phi_p}{\partial |E_q|} \left| \Delta |E_q^{(k)}| \right| + \sum_{q=1}^{n} \frac{\partial \Phi_p}{\partial \delta_q} \left| \Delta \delta_q^{(k)} \right| \]

\[ \Delta Q_p^{(k)} = \sum_{q=1}^{n} \left| E_q \right| x \frac{\partial \Psi_p}{\partial |E_q|} \left| \Delta |E_q^{(k)}| \right| + \sum_{q=1}^{n} \frac{\partial \Psi_p}{\partial \delta_q} \left| \Delta \delta_q^{(k)} \right| \]

In the form of a matrix this can be represented as
It has been practically found that
(1) Active power component \( P \) has limited influence on the magnitude of voltage \( |E| \).
(2) Similarly, reactive power \( Q \) has limited influence on the angle \( \delta \).
Hence ignoring the coupling term from the matrix representation we are left with

\[
\begin{bmatrix}
\Delta P \\
\Delta Q \\
\end{bmatrix} = \begin{bmatrix}
\frac{\partial \Phi}{\partial \delta} | E | & \frac{\partial \Phi}{\partial | E |} \\
\frac{\partial \Psi}{\partial \delta} | E | & \frac{\partial \Psi}{\partial | E |} \\
\end{bmatrix} \Delta \delta \\
\frac{\Delta E}{| E |} \\
\end{bmatrix} \Delta \delta
\]

Thus, we have reduced the dimensions of the matrices to be solved. Hence we can save on computation time.

**Motivation:** Can the Jacobian be approximated by a constant matrix so that repeated inversion of it can be avoided?

**Approximation 1:**

The diagonal terms of \( \vartheta(\Phi) / \vartheta(\delta) \) is given by

\[
\frac{\partial \Phi}{\partial \delta_p} = \left| E_p \right| \sum_{q=1}^{n} E_q \left[ -g_{pq} \text{Sin}(\delta_p - \delta_q) - b_{pq} \text{Cos}(\delta_p - \delta_q) \right] \\
= -Q_p - \left| E_p \right|^2 b_{pp}
\]

The off-diagonal term of \( \vartheta(\Phi) / \vartheta(\delta) \) is given by

\[
\frac{\partial \Phi}{\partial \delta_q} = \left| E_p \right| \left| E_q \right| \left[ g_{pq} \text{Sin}(\delta_p - \delta_q) - b_{pq} \text{Cos}(\delta_p - \delta_q) \right]
\]

The diagonal terms of \( \vartheta(\Psi) / \vartheta(|E|) \) is given by
The off-diagonal terms of $\partial \Psi / \partial |E|$ is given by

$$|E_p| \frac{\partial \Psi_p}{\partial |E_p|} = |E_p| \left[ -2 |E_p| b_{pp} + \sum_{q=1}^{n} E_q \left[ g_{pq} \sin(\delta_p - \delta_q) - b_{pq} \cos(\delta_p - \delta_q) \right] \right]$$

$$= Q_p - |E_p|^2 b_{pp}$$

**Approximation 2:**

Considering,

$$\cos(\delta_p - \delta_q) = 1 \quad \text{as} \ (\delta_p - \delta_q) \text{ is very nearly zero.}$$

Hence we can consider

$$\sin(\delta_p - \delta_q) = \delta_p - \delta_q$$

Now $g_{pq}$ has a small value. Hence the term $g_{pq} \sin(\delta_p - \delta_q) \ll b_{pq}$ can also be ignored.

Because of the above approximations, the consideration of them in the equation of:

$$Q_p = |E_p| \sum_{q=1}^{n} E_q \left[ -b_{pq} \cos(\delta_p - \delta_q) + g_{pq} \sin(\delta_p - \delta_q) \right]$$

leads to $Q_p$ becoming very less compared to $|E_p|^2 b_{pp}$ and hence can be ignored.

This leads to the Jacobian expression being as follows:

$$\frac{\partial \Phi_p}{\partial \delta_p} = - |E_p|^2 b_{pp}$$

$$\frac{\partial \Phi_p}{\partial \delta_q} = |E_p| |E_q| b_{pq}$$

$$|E_p| \frac{\partial \Psi_p}{\partial |E_p|} = - |E_p|^2 b_{pp}$$

$$|E_p| \frac{\partial \Psi_p}{\partial |E_p|} = - b_{pq} |E_p| |E_q|$$
**Approximation 3:**

$|E_q|$ is approximately 1

Because of approximation upto 3 we get,

$$\Delta P_p = -\sum_{q=1}^{n} |E_p| \cdot |E_q| \cdot b_{pq} \Delta \delta_q$$

$$\frac{\Delta P_p}{|E_p|} = \sum_{q=1}^{n} |E_q| \cdot b_{pq} \Delta \delta_q$$

Now if we substitute $|E_q| = 1$

Then we get the following simplified expression

$$\frac{\Delta P}{|E|} = b' \Delta \delta$$

*Here* $b' = [-b_{pq}]$

Similarly for reactive power equations

$$\Delta Q_p = -\sum_{q=1}^{n} |E_p| \cdot |E_q| \cdot b_{pq} \frac{\Delta |E_q|}{|E_q|}$$

$$\frac{\Delta Q_p}{|E_p|} = -\sum_{q=1}^{n} b_{pq} \Delta |E_q|$$

$$\frac{\Delta Q}{|E|} = b'' \Delta E$$

*Here* $b'' = [-b_{pq}]$

Note that $b'$ is not the same as $b''$. This is because of the fact that $\frac{\Delta Q_p}{|E_p|}$ exists only for PV buses while the term $\frac{\Delta P_p}{|E_p|}$ exists for all the buses.

The dimension of $b'$ is always $(n-1) \times (n-1)$. However the effect of (a) tap changing component and (b) shunt capacitance is ignored in Fast Decoupled method.

**Motivation:**
To apply the VAR limits for using it into the Fast Decoupled method, in case of the PV buses.
Solution:
The method to deal with this is similar to the one we attempted while using the N-R method excepting that here we use "sensitivity" and mismatch to correct the voltage. The basic idea to calculate the sensitivity number is given as follows:

When Q limit is violated for a PV bus the idea is to adjust the magnitude at each iteration to drive Q violation to zero.

Let $|E|^k$ be the voltage magnitude after the k th iteration and $Q^k$ be the injected VAR computed using the voltages after the k th iteration

$$\Delta Q^k = Q_{\text{limit}} - Q^{(k)}$$

Thus $\Delta |E|^k = S \frac{\Delta Q^k}{|E|^k}$

Here S is called as the sensitivity factor

The adjusted voltage is calculated by

$$E_{\text{new}}^k = |E|^k + \Delta |E|^k$$

The diagonal element of the inverse of the augmented $B''$ matrix corresponding to the generator bus under consideration

$$\frac{\Delta Q}{|E|} = B^* \Delta E$$

$$\Delta |E| = B^{-1} \frac{\Delta Q}{|E|}$$

Now let us discuss a way to update S without carrying out $B^{-1}$ frequently:

Let $B''$ be the augmented $B''$ matrix such that

$$B''^{-1} = \begin{bmatrix} B'' & C \\ C' & D \end{bmatrix} = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$$

Here $A_4$ is the sensitivity matrix i.e $A_4 = S$

$$\begin{bmatrix} B'' & C \\ C' & D \end{bmatrix} * \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & 1 \end{bmatrix}$$

Thus $B''A_1 + CA_3 = I$

$CA_1 + DA_3 = 0$

$B''A_2 + CS = 0$

$C'A_2 + DS = 1$
\[ C'[B^n]^{-1}CS + DS = 1 \]

\[ S = \frac{1}{D - C'[B^n]^{-1}C} \]

**Question:** Suppose that one line is removed/added for contingency/restorative studies. Can we simplify the updating process of \( B' \) and \( B'' \)?

**Answer:**
Let \( B \) be the initial matrix and \( B_{\text{mod}} \) be the modified one. Thus

\[ B_{\text{mod}} = B - m_{ij} (b_{ij}) m_{ij}^t \]

Here \( b_{ij} = \) series susceptance of the removed line and

\[ m_{ij} = [0 \ 0 \ldots \ 1 \ 0 \ldots \ 0 \ -1 \ 0 \ldots \ 0]^t \]

The +1 is at end \( i \) and -1 is at end \( j \).

This is because that only the \( i \)th and \( j \)th column or row element would be affected.

Now, \( B_{\text{mod}}^{-1} = (B - m_{ij} (b_{ij}) m_{ij}^t)^{-1} \)

This can be simplified as follows

\[ (B_{\text{mod}})^{-1} = B^{-1} - B^{-1} m_{ij} (m_{ij}^t B^{-1} m_{ij} - 1 / b_{ij})^{-1} m_{ij}^t B^{-1} \]

It can be seen that only the terms within the square bracket have to be calculated and both of the terms inside are scalar. Hence the computational complexity is reduced.

**Incorporating Transformers (TCUL and \( \phi \)-shifting) into the Load flow calculation:**

**Question:** How does the transformer tap \( t \) influence the \( Y \) matrix?

**Ans:**
It is evident that the entries \( Y_{pp}, Y_{pq}, Y_{qq}, Y_{qp} \) will change with a change in the tap \( t \).

The following relation exists:

\[ Y_{pp} = Y_{pp}^{(0)} + \frac{Y_{pq}'}{t^2} \]

\[ Y_{qq} = Y_{qq}^{(0)} + \frac{Y_{pq}'}{t} \]

\[ P_p + jQ_p = E_p * I_p^* \]

\[ = E_p \left( \sum_{i=1}^{n} Y_{pi}^* E_i^* \right) \]

\[ = E_p Y_{pp}^* E_p^* + E_p Y_{pq}^* E_q^* + E_p \left( \sum_{i=p,q}^{n} Y_{pi}^* E_i^* \right) \]
\[
P_p + jQ_p = |E_p|^2 \left[ Y_{pp}^{(0)*} + \frac{Y_{pq}'}{t^2} \right] + E_p E_q' \left[ -\frac{Y_{pq}'}{t} \right] + \text{other terms}
\]

Let \( Y_{ij} = |Y_{ij}| e^{j\theta_{ij}} \)
\( E_i = |E_i| e^{j\delta_i} \)
\( Y_{pq}' = Y_{pq} | e^{-j\frac{\pi}{2}} \)

Assume that the transformers are purely reactive. Therefore

\[
P_p + jQ_p = |E_p|^2 \left[ Y_{pp}^{(0)*} | e^{-j\theta_{pp}} + \frac{Y_{pq}'}{t^2} e^{-j\frac{\pi}{2}} \right] + |E_p| |E_q| e^{j\delta_p} \left[ \frac{Y_{pq}'}{t} e^{-j\frac{\pi}{2}} \right] + |E_p| e^{j\delta_p} \sum_{i=1}^{n} |Y_{ps} | e^{-j\theta_{ps}} | E_i | e^{-j\delta_i}
\]

\[
P_p = \Phi_p = -|E_p| |E_q| \left| \frac{Y_{pq}'}{t} \right| \cos(\delta_p - \delta_q + \frac{\pi}{2}) + \text{terms not involving } t
\]

\[
Q_p = \Psi_p = |E_p|^2 \left| \frac{Y_{pq}'}{t^2} \right| - |E_p| |E_q| \left| \frac{Y_{pq}'}{t} \right| \sin(\delta_p - \delta_q + \frac{\pi}{2}) + \text{terms not involving } t
\]

Here \(|E_p| = |E_p|_{\text{scheduled}}\)

\[
\frac{\partial \Phi_p}{\partial t} = -|E_p| |E_q| \left| \frac{Y_{pq}'}{t^2} \right| \cos(\delta_p - \delta_q + \frac{\pi}{2})
\]
\[
= -|E_p| |E_q| \left| \frac{Y_{pq}'}{t^2} \right| \sin(\delta_p - \delta_q)
\]

\[
\frac{\partial \Psi_p}{\partial t} = -2 |E_p|^2 \left| \frac{Y_{pq}'}{t^2} \right| - |E_p| |E_q| \left| \frac{Y_{pq}'}{t^2} \right| \sin(\delta_p - \delta_q + \frac{\pi}{2})
\]
\[
= -2 |E_p|^2 \left| \frac{Y_{pq}'}{t^2} \right| + |E_p| |E_q| \left| \frac{Y_{pq}'}{t^2} \right| \cos(\delta_p - \delta_q)
\]

Similarly, for the bus \( q \)

\[
P_q = \Phi_q = -|E_p| |E_q| \left| \frac{Y_{pq}'}{t} \right| \cos(\delta_q - \delta_p + \frac{\pi}{2}) + \text{terms not involving } t
\]
\[ Q_q = \Psi_q = |E_p|^2 Y'_{pq} - |E_p||E_q| \frac{|Y'_{pq}|}{t} \sin(\delta_q - \delta_p + \frac{\pi}{2}) + \text{terms not involving } t \]

Hence,

\[
\frac{\partial \Phi_q}{\partial t} = -|E_p||E_q| \frac{|Y'_{pq}|}{t^2} \sin(\delta_q - \delta_p)
\]

\[
\frac{\partial \Psi_q}{\partial t} = |E_p||E_q| \frac{|Y'_{pq}|}{t^2} \cos(\delta_q - \delta_p)
\]

Next, we have to replace \(\Delta|E_p|\) by \(\Delta t\) in the N-R iterative algorithm. Moreover the limit of \(t_{\text{min}} < t < t_{\text{max}}\) has to be introduced into the algorithm. However, if by any case the limit is hit then \(t\) is fixed to the hit limit value and then \(\Delta|E_p|\) is re-introduced into the N-R algorithm.
Homework Set # 6
Q.1 Is $\omega$ accelerating the converging sequence to a solution to $f(x)=0$?
Q.2 Derive the expression for $Q_p$ for the PV bus.
Q.3 Check whether the flowchart of Gauss-Seidel method makes sense.
Q.4: Check the Jacobian components of rectangular N-R method.

Homework Set # 7
Q.1. Draw the flow-chart of Newton-Raphson Method for Load flow solution.
Q.2. Deal with the Generation bus in polar form for N-R method.
Q.3 Verify the load flow solution obtained by Fast-decoupled with a simple 2 bus system.
Q.4. Develop the data file format for Transaction Based Power Flow (TBPF) having:
   (a) What additional data should be supplied apart from the normal power flow data?
   (b) State whether any conventional data that is redundant and can be eliminated.
   (c) Design a data format which is compatible to the IEEE data format to include the transaction data

Homework set #8
Q 1. Suppose that we have a shunt capacitor that can be switched on and off on a bus. Investigate the $B^{-1}_{\text{mod}}$.
   (a) $m_{ij} = ?$
   (b) $B^{-1}_{\text{mod}}$ in terms of $B^{-1}$.
Form the YBUS

Form \( Klp \) vector & \( YLpq \) matrix where,
\[ p = 1 \ldots n, \quad p = s \]
\[ q = 1 \ldots n, \quad q = p \]

Set \( k = 0 \)
Set Initial Guess

Set Max(\( \Delta E \)) = 0,
Set bus count \( p = 1 \)

Read i/p data
Print i/p data

Is this Slack Bus?

Is this Gen. Bus?

Replace \( |E_p^k| \) by \( |E_p^{spec}| \)
& calculate
\[ e_{p_{new}} = |E_p^k| \cos(\delta_p) \]
\[ f_{p_{new}} = |E_p^k| \sin(\delta_p) \]

Calculate \( Q_p^k \)

X
Evaluate \( Q_g = Q_p^k + Q_{bp} \)

- If \( Q_g \geq Q_{\min} \):
  - \( Q_g = Q_{\min} \)
  - UPDATE \( Q_p^k \)

- If \( Q_g \leq Q_{\max} \):
  - \( Q_g = Q_{\max} \)
  - UPDATE \( Q_p^k \)

Recalculate \( K_{lp} \) vector

Solve for \( E_p^{k+1} = \left( \frac{K_{lp}}{E_p^k} \right) \):

- \( \sum Y_{pq} V_q^{k+1} \) for \( q = 1 \ldots (p-1) \)
- \( \sum Y_{pq} V_q^k \) for \( q = (p+1) \ldots n \)

Replace \( E_p^k \) by \( E_p^k \) new = \( e_p^k \) new + \( j f_p^k \) new
Calculate $\Delta E_p^k = E_p^{k+1} - E_p^k$

Is $|\Delta E_p^k| > \text{Max}(\Delta E)$?

Max($\Delta E$) = $|\Delta E_p^k|$

$E_p^{k+1} = E_p^k + \alpha \ast (\Delta E_p^k)$

$E_p^k = E_p^{k+1}$

CALCULATE line flows, P & Q for slack Bus, Print results & STOP
Steps are as follows:

1. Assume initial guess for all voltages except the slack bus. If flat start is assumed then choose $1 + j0.0$. Assume a suitable value of $\varepsilon$, the convergence criteria i.e if the absolute value of maximum change in voltage between two consecutive iterations for all buses is less than a pre-specified tolerance $\varepsilon$, the convergence is achieved and the iterative procedure is terminated.

1.(a) Set iteration count $K = 0$
1.(b) Set bus count $p = 1$
1.(c) Initialize $dEmax = 0$
1.(d) Check for slack bus. If it is slack bus go to step 4(a), otherwise go to next step

2. Check which of the buses are voltage controlled and which are load buses. For voltage controlled buses go to the next step otherwise go to Step 4.

3. Replace the value of voltage magnitude of PV bus to the specified value. Keep the phase angle same as that in that iteration. Calculate $Q$ for the bus. If $Q$ lies within
limit of maximum and minimum, then calculate \( K_{lp} \). Calculate the new value of the bus voltage for the PV bus. If not within limit i.e. \( Q < Q_{\text{min}} \) replace \( Q = Q_{\text{min}} \) or if \( Q > Q_{\text{max}} \) then \( Q = Q_{\text{max}} \). Now go to step 4.

4. For bus \( p \) evaluate \( E_{p}^{k+1} \) and also change in voltage i.e \( dE_{p}^{k} \). If \( \text{abs}(dE_{p}^{k}) \) is greater than \( dE_{\text{max}} \) then replace \( dE_{\text{max}} \) with \( \text{abs}(dE_{p}^{k}) \). Then calculate accelerated \( E_{p}^{k+1} \) and replace \( E_{p}^{k} \) with this new accelerated value after which go to step 4(a). If \( \text{abs}(dE_{p}^{k}) \) is not greater than \( dE_{\text{max}} \) just replace \( E_{p}^{k} \) with \( E_{p}^{k+1} \) and continue to step 4(a).

4(a) Advance bus count by 1 and check if all the buses have been taken into account. If yes go to next step or else goto step 1(d).

5. Find out if the \( dE_{\text{max}} \) is less than tolerance \( \varepsilon \). If yes, then goto step 6. If not, increment iteration count & check if it is less than maximum limit. If yes then go to step 2 for the next iteration. Otherwise print that no solution with tolerance specified for the maximum limit of iterations.

6. Calculate all the injected powers, slack bus power & print results.