Optimal Power Flow Analysis for Power Systems

Nannuri Pranay Kumar Reddy** , 21085052 , Btech, Part 4**(Research Paper 1)
Macha Venkata Vasishta** , 21085043 , Btech, Part 4**(Research Paper 2)
Jarupula Anitha** , 21085033 , Btech, Part 4**(Research Paper 3)

WEEK - 1

Research Paper 1: OPTIMUM POWER FLOW ANALYSIS BY NEWTON RAPHSON METHOD, A CASE STUDY

Description:

This research paper tells about optimal power flow(OPF) analysis in power system and how can this be done with newton raphson method. It tells about the objective of OPF and how to attain it using newton raphson method . And also some research is done to know how traditional load flow analysis differs from optimal power flow.

Literature:

Some key components we learnt though this paper are:

Load Flow Analysis
Optimal Power Flow Analysis
Newton Raphson Method
Objective functions for OPF

Summary:

Load Flow and OPF

Due to increasing load demands ,Power system grids get destabilise and may cause blackout etc. So frequently power systems should be analysed and should be modified according to the load demands. Load Flow (LF) and Optimal Power Flow (OPF) analyses are crucial for the efficient operation and analysing of power systems. Understanding the differences between these two methods is important.

Load Flow Analysis:
LF analysis is used to calculate the voltage magnitudes and angles at different buses, as well as the real and reactive power flows in the network under steady-state conditions. It helps in ensuring that the system operates within its limits under various load conditions.

Example:

Considering a simple power system with three buses (A, B, and C) connected by transmission lines. LF analysis can determine:
- The voltage at each bus (e.g., 1.0 p.u. at A, 0.98 p.u. at B, and 0.97 p.u. at C).
- The power flow on each line (e.g., 50 MW from A to B, 30 MW from B to C).

Optimal Power Flow Analysis:

OPF is same as LF but additionally it optimizes a specific objective function, such as minimizing generation cost, reducing power losses, or improving voltage stability. OPF takes into account additional constraints like generator limits, line capacities, and voltage limits(LF).

Example:

In the same three-bus system, OPF might aim to minimize the total generation cost while ensuring:
- The voltage at each bus remains within acceptable limits (e.g., 0.95 to 1.05 p.u.).
- The power flow on each line does not exceed its capacity (e.g., no more than 100 MW).

Key Differences Between LF and OPF:

- *Objective:* LF aims to determine the power system's steady-state conditions, whereas OPF aims to optimize system performance.

- *Complexity:* OPF is more complex due to the inclusion of optimization objectives and additional constraints.

- *Applications:* LF is used for routine analysis and planning, while OPF is used for operational optimization and advanced planning.

Newton-Raphson Method in LF and OPF:

The Newton-Raphson method is a iterative technique used for solving non-linear algebraic equations in both LF and OPF analysis. It is preferred due to its **quadratic convergence and robustness.**

Steps in Newton-Raphson Method:

1. *Initial Guess:* Start with an initial guess for the voltages and angles at each bus.
2. *Jacobian Matrix Calculation:* Compute the Jacobian matrix, which contains partial derivatives of the power flow equations with respect to the voltage magnitudes and angles.

3. *Update Solution:* Solving the linearized system of equations to update the voltage magnitudes and angles.
4. *Iteration:* Repeat the process until the changes in voltage magnitudes and angles are within acceptable limits.

Example:
For a bus with an initial voltage guess of 1.0 p.u., the Newton-Raphson method iteratively adjusts this value to converge to the true voltage value that satisfies the power flow equations.

Benefits of the Newton-Raphson Method:

- *Accuracy:* Provides highly accurate results for LF and OPF problems.
- *Convergence:* Converges quickly, especially when the initial guess is close to the true solution.
- *Robustness:* Handles large and complex power systems effectively.

Steps in OPF Analysis:

1. *Data Collection:* we should gather data on the power system, including bus voltages, line impedances, and generator characteristics.

2. *Model Formulation:* Develop mathematical models representing the power system and the optimization objectives.(For example we can simulate in Simulink).

3. *Solution Techniques*: Use the Newton-Raphson(or any iterative method) method and other optimization algorithms to solve the OPF problem.

Benefits of OPF:
- *Cost Efficiency:* Reduces operational costs by optimizing the dispatch of generation units.
- *Reliability:* Enhances system reliability by ensuring operation within safe limits.
- *Efficiency:* Improves the overall efficiency of the power system by minimizing losses.

Conclusion

In conclusion, LF analysis is essential for understanding the current state of a power system, while OPF analysis provides a powerful tool for optimizing system performance. The Newton-Raphson method plays a critical role in solving both LF and OPF problems due to its accuracy and quickness . Implementing OPF with the Newton-Raphson method can lead to significant improvements in efficiency and reliability, making it a valuable technique for power system engineers.

Research Paper 2: Optical Power Flow Methods : A Comprehensive Survey

Description: In this research paper we are going to see just the methods used to solve Optimal Power Flow(OPF) Analysis.

Literature: Some key components we learnt in though this paper are:

Optimal Load Flow
Artificial Intelligence

Summary:

The optimal power flow solution methods are mainly classifies into two methods
Traditional Methods and Artificial Intelligence(AI) Methods
The Traditional Methods are further further subdivided, these are:
a)Linear Programming(LP)
In this method we use linear equations and inequalities to solve the optimal power flow problems. The main advantage of using LP in OPF are its computational efficiency and its availability of robust solvers, making it effective for large-scale Power systems.However its only applicable to problems where the scenario could be approximated to linear models.

b)Gradient Method

As the name suggests, in this method we use the gradient(or derivative) of the objective function with respect to the decision variables such as generator outputs and voltage levels step by step i.e by iterations.This method is reliable, easy to implement, and converges for well-behaved functions, but they may take more time to converge or may get stuck in local minima(ex: Non-complex problems).

c)Newton-Raphson Method

Newton method is a second order method for unconstrained optimization based on the application of a 2nd order Taylor series expansion about the current candidate solution. It is a iterative technique. In this method we calculate the Jacobian matrix and updating the variables using inverse jacobian matrix and function's derivative. This method is known for its fast convergence, making it suitable for large-scale power systems.It can be computationally expensive due to matrix calculations.

d)Quadratic Programming

Quadratic Programming (QP) is a special form of non-linear programming whose objective function is quadratic and constraints are linear. We solve optimal power flow problems using QP solvers.This method can handle more complex and realistic cost functions compared to linear programming.

e)NonLinear Programming

Nonlinear programming (NLP) in Optimal Power Flow (OPF) is an advanced optimization technique used to determine the optimal operating conditions of a power system by minimizing a non-linear objective function subject to non-linear constraints. In this method we use iterative algorithms to find optimal solution.This method is ideal for realistic and detailed modeling of power systems.

f)Interior-Point Method

The objective of the Interior-Point method is to minimize an objective function while satisfying both equality and inequality constraints. In this method we solve optimal power flow using barrier functions to transform the constrained problem into a series of easier problems, then iteratively solves these problems by moving through the interior of the feasible region.It's highly efficient for large scale problems and its faster in converging compared to other methods.

Now we are going to see the Artificial Intelligence Methods:

a)Genetic Algorithm

The objective of this method is to find the optimal operating conditions of a power system by minimizing an objective function while meeting system constraints. We approach the optimal power flow problem by using a population of potential solutions that evolve over generations through selection, crossover, and mutation processes. This method is capable of handling complex, non-linear, and non-convex problems without requiring gradient information. Advantages this method can offer is that it can escape local optima, providing good solutions for highly complex and multi-modal problems. But these may be computationally expensive.

b)Particle Swarm Optimization

The objective of this method is to minimize an objective function while satisfying system constraints. We approach this problem by using a swarm of particles, where each particle represents a potential solution. Particles adjust their positions based on their own experience and the experience of neighboring particles.This method is effective for handling complex, non-linear, and multi-modal optimization problems.It's simple to implement, can escape local optima, and does not require gradient information.

c)Artificial Neural Network

The objective of this method is to optimize power system operations by predicting optimal operating points that minimize an objective function while meeting constraints. We approach this method by training a neural network using historical data or simulation results to learn the complex relationships between system variables and optimal solutions.This method is capable of modeling highly non-linear and complex relationships in power systems.Tha advantages of this method, once trained, ANNs provide fast and efficient predictions, are robust to noise, and can handle real-time optimization.The downside of this method is it requires a large amount of training data, can be computationally intensive to train, and may need careful tuning of network architecture and parameters.

d)Bee Colony Optimization

Bee Colony Optimization (BCO) in Optimal Power Flow (OPF) is a heuristic optimization technique inspired by the foraging behavior of honeybees.The objective of this method is to optimize power system operations by minimizing an objective function while satisfying system constraints.We approach this method by mimicking the behavior of bees searching for nectar, with "scout bees" exploring new areas and "worker bees" refining promising solutions.The advantages of this method are it is capable of escaping local optima and providing high-quality solutions through collective search and exploration strategies.

e)Differential Evolution

Differential Evolution (DE) in Optimal Power Flow (OPF) is a heuristic optimization algorithm inspired by evolutionary biology, used to solve complex optimization problems. The objective of this method is to minimize an objective function (e.g., generation cost, losses) while satisfying power system constraints. We approach this method by using a population of candidate solutions and iteratively evolves them through mutation, crossover, and selection processes to improve the solution.The advantages of this method are it’s simple to implement, robust, and capable of finding global optima or near-optimal solutions.

f)Grey Wolf Optimizer

The Grey Wolf Optimizer (GWO) in Optimal Power Flow (OPF) is a nature-inspired heuristic optimization technique modeled after the hunting behavior of grey wolves. The objective of this method is to optimize power system operations by minimizing an objective function while adhering to system constraints. We approach this method by mimicking the social hierarchy and hunting strategy of grey wolves, including leadership and collaborative searching, to explore and exploit the solution space. Advantages of this method are it’s simple to implement and capable of providing high-quality solutions by balancing exploration and exploitation.

g)Shuffled Frog-Leaping

Shuffled Frog Leaping Algorithm (SFLA) in Optimal Power Flow (OPF) is a population-based optimization technique inspired by the behavior of frogs leaping and sharing information in a swamp.The objective of this function is to optimize power system operations by minimizing an objective function (e.g., cost, losses) while satisfying system constraints.We approach this method by dividing the population of solutions into groups (memeplexes) that evolve independently, with frogs in each group "leaping" to explore new solutions and sharing information between groups to enhance overall performance.The advantages are as described-Balances exploration and exploitation effectively, and can find high-quality solutions by leveraging collective intelligence and collaboration.

Conclusion:

This paper reviews various optimization methods for solving Optimal Power Flow (OPF) problems. Classical methods, despite significant advancements, have limitations such as the need for linearization and differentiability, potential to get stuck in local optima, poor convergence, and inefficiency with large numbers of variables. In contrast, Artificial Intelligence (AI) methods offer greater versatility, effectively handling qualitative constraints and finding multiple optimal solutions in a single run. They are well-suited for multi-objective optimization and often excel in locating global optimum solutions.

Research Paper 3: A Review on Optimal power flow problem and solution methodologies

---

Description: This research paper explores the essential role of Optimal Power Flow (OPF) in power systems.The paper provides a review of various optimization techniques used for solving OPF problems.The research covers the evolution of OPF methods, starting from the 1960s, and discusses how these techniques have been adapted to address modern issues such as cost minimization, power losses, and system stability.

Literature: Some key components we learnt in though this paper are:

Conventional methods for OPF
Artificial Intelligence method for OPF

Summary:

Gradient Method:
This method is used in solving the Optimal Power Flow (OPF) problem by focusing on state and control variables. It simplifies the problem by reducing the number of variables through power flow equations. The method involves optimizing generation costs and reducing active losses using penalty functions, while also incorporating techniques like the Lagrange Multiplier for boundary verification.

Newton Method:
The Newton method is an effective solution algorithm known for its fast convergence when approaching a solution. It is particularly useful in power system applications where system voltages and generator outputs are near their nominal values. The method utilizes the Jacobian Matrix and Lagrangian Multipliers to optimize control variables.

Linear Programming Method (LP) :
This method involves formulating the optimization problem with a linear objective function and linear constraints, using non-negative variables. The Simplex method is often employed to solve the LP problems by selecting and updating variables at different buses. However, LP methods may struggle with infeasible situations. Advanced techniques include using piece-wise differentiable penalty functions and mixed-integer LP to address specific constraints like contingency and reduce transmission losses.

This method requires linearization of both the objective function and constraints in each iteration to improve results, and it can handle discrete variables such as capacitors and shunt reactors.

Quadratic Programming (QP):
This method is a type of nonlinear programming where the objective function is quadratic and the constraints are linear. This method uses algorithms like Wolfe's algorithm and Quasi-Newton techniques to solve the optimization problem. It is effective for various bus systems and can handle initial impractical starting points with fast convergence. The QP method is particularly useful for solving optimal power flow (OPF) problems, including those involving FACTS devices and large systems. It also addresses the sensitivity of solutions to different starting points and aims to achieve accurate results for both loss and cost minimization.

Interior Point (IP) Method:
This method is designed for solving large-scale linear programming (LP) problems efficiently. It operates within the interior of the feasible space, distinguishing it from methods like the Simplex method. The IP method has proven to be significantly faster in solving problems with many variables and constraints. Variations of the IP method, such as the Extended Quadratic Interior Point (EQIP) method and Interior Point Branch and Cut Method (IPBCM), have been developed for specific optimization challenges, including mixed-integer nonlinear problems. The IP method is applicable to both LP and quadratic programming (QP) problems and is often used for large-scale power system optimizations.

Particle Swarm Optimization (PSO) inspired by the social behaviors of animals to solve optimization problems.It is used in power systems for tasks like reactive power control and optimal power flow (OPF). PSO involves particles that adjust their positions based on their own and their neighbors' experiences. Enhanced PSO methods improve speed and accuracy by refining particle movement and handling constraints.

Artificial Bee Colony (ABC) Optimization inspired by the behavior of honey bees to solve problems. It has been effectively used for optimal power flow (OPF) in power systems. The algorithm explores solutions efficiently, often converging faster than other methods. Variants include Chaotic ABC and hybrid methods combining ABC with Differential Evolution for better performance. ABC has shown success in various test systems, demonstrating its accuracy and effectiveness.

Conclusion:

This paper reviews methods for solving Optimal Power Flow (OPF) problems, highlighting that traditional methods often face limitations, such as getting stuck in local optima and struggling with complex constraints. In contrast, artificial intelligence techniques offer greater flexibility and better solutions for complex, multi-objective problems. As renewable energy integration increases, AI methods become crucial for effective decision-making and system optimization. This review provides a solid foundation for future research in OPF.

REFERENCES:

Paper 1: Eltamaly, A. M., Sayed, Y., El-Sayed, A. H. M., & Elghaffar, A. N. A. (2018). Optimum power flow analysis by Newton raphson method, a case study. *Ann. Fac. Eng. Hunedoara*, *16*(4), 51-58.

Paper 2: Khamees, A. K., Badra, N., & Abdelaziz, A. Y. (2016). Optimal power flow methods: A comprehensive survey. *International Electrical Engineering Journal (IEEJ)*, *7*(4), 2228-2239.

Paper 3: Maskar, M. B., Thorat, A. R., & Korachgaon, I. (2017, February). A review on optimal power flow problem and solution methodologies. In *2017 International Conference on Data Management, Analytics and Innovation (ICDMAI)* (pp. 64-70). IEEE.

WEEK - 2

Overview of Optimal Power flow

Economic load dispatch(ELD) calculates how much power is produced at each power generator ignoring the transmission line limits.

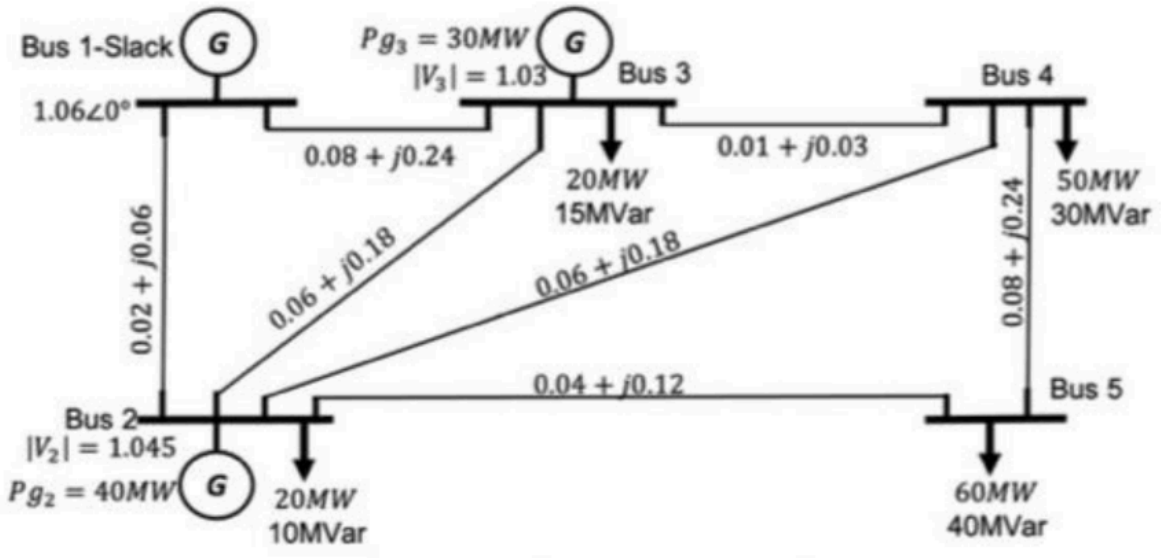
Optimal Power Flow(OPF) combines the load flow analysis and economic load dispatch.

Inshort ,the economic load dispatch is not considering the limits of real power losses due to which it may not provide the optimum economical scheduling of the power plants.

On the other hand, in optimal power flow, the losses are obtained first by performing load analysis in order to obtain the minimum losses so that cheapest overall generating cost is obtained.

Problem Statement

Taken a 5-bus power system with generator at buses 1,2, and 3 .Bus 1 , with its voltage is set at 1.06 and angle is 0 degree p.u ,it is taken as slack bus



Capacitive Shunt susceptance (pu)	
Line	B/2
1-2	0.03
1-3	0.025
2-3	0.02
2-4	0.02
2-5	0.015
3-4	0.01
4-5	0.025

INPUT DATA

Input data is taken from a problem in the book “Power System Analysis by Hadi Sadat, Chapter 7.

Some input values like reactive power limits ,real power limits, generation power limits are assumed.and cost functions are the following:

C1=200 + 7.0*P1 + 0.008*P1*P1;

C2=180+ 6.3*P2 + 0.009*P2*P2;

C3=140 + 6.8*P3 + 0.007*P3*P3;

P1,P2,P3 are in MW

10<=P1<=85

10<=P2<=80

10<=P3<=70

APPROACH

Here, we first calculated the Ybus matrix , and then solved the power flow analysis by Newton Raphson Method.

Active power Pi and reactive power Qi at bus i

$$P_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j)\right)$$

$$Q_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)\right)$$

1)Power Mismatch calculations

$$\Delta P_i = P_i^{\text{specified}} - P_i^{\text{calculated}}$$

$$\Delta Q_i = Q_i^{\text{specified}} - Q_i^{\text{calculated}}$$

2)Jacobian Matrix

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \Delta \theta \\ \Delta V \end{bmatrix}$$

3)Jacobian Elements

$$J_{11}(i,j) = \frac{\partial P_i}{\partial \theta_j} = -V_i V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j) \right)$$

$$J_{12}(i,j) = \frac{\partial P_i}{\partial V_j} = V_i \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j) \right)$$

$$J_{21}(i,j) = \frac{\partial Q_i}{\partial \theta_j} = -V_i V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j) \right)$$

$$J_{22}(i,j) = \frac{\partial Q_i}{\partial V_j} = V_i \left(G_{ii} - G_{ij} \cos(\theta_i - \theta_j) - B_{ij} \sin(\theta_i - \theta_j) \right)$$

4)Voltage and Magnitude Updates

$$\Delta \theta = J^{-1} \times \Delta P$$

$$\Delta V = J^{-1} \times \Delta Q$$

$$\theta^{\text{new}} = \theta^{\text{old}} + \Delta \theta$$

$$V^{\text{new}} = V^{\text{old}} + \Delta V$$

From Newton Raphson we noted the total transmission losses and power generated by slack bus..

$$P_{loss} = \sum_{i=1}^{n}\sum_{j=1}^{n} P_i B_{ij} P_j + \sum_{i=1}^{n} B_{i0} P_i + B_{00}$$ Ploss between 2 buses.

Now, Solved the system using Economic Load Dispatch(ELD) , and found out the amount of power to be generated by slack bus.

Now from the above values , we see the absolute difference between the power generated values from both methods , if the difference is less than 0.001 we will stop, else values are adjusted and the loops runs again.

CODE:

In the below code , it performs power flow analysis and optimizes power generation dispatch in an electrical grid. It calculates bus voltages and angles using the Newton-Raphson method, computes system losses, and adjusts generator outputs to minimize costs while meeting demand. The process iteratively refines the solution until the power system operates efficiently with minimal discrepancies(dpslack<0.001).

```
clear
basemva = 100;  accuracy = 0.0001;  maxiter = 10;

%        Bus Bus  Voltage Angle   ---Load---- -------Generator----- Static Mvar
%        No  code Mag.    Degree  MW    Mvar  MW  Mvar Qmin Qmax    +Qc/-Ql
busdata=[1   1    1.06    0.0      0     0     0   0    10   50      0
         2   2    1.045   0.0     20    10    40  30    10   50      0
         3   2    1.03    0.0     20    15    30  10    10   40      0
         4   0    1.00    0.0     50    30     0   0     0    0      0
         5   0    1.00    0.0     60    40     0   0     0    0      0];

%                                      Line code
%         Bus bus   R      X     1/2 B   = 1 for lines
%         nl  nr  p.u.   p.u.   p.u.     > 1 or < 1 tr. tap at bus nl
linedata=[1   2   0.02   0.06   0.030     1
          1   3   0.08   0.24   0.025     1
          2   3   0.06   0.18   0.020     1
          2   4   0.06   0.18   0.020     1
          2   5   0.04   0.12   0.015     1
          3   4   0.01   0.03   0.010     1
          4   5   0.08   0.24   0.025     1];

cost = [200  7.0    0.008
        180  6.3    0.009
        140  6.8    0.007];

mwlimits =[10  85
           10  80--
           10  70];
```

```
lfybus                    % form the bus admittance matrix
lfnewton     % Power flow solution by Newton-Raphson method
busout       % Prints the power flow solution on the screen
bloss              % Obtains the loss formula coefficients
gencost          % Computes the total generation cost $/h
dispatch          % Obtains optimum dispatch of generation
       % dpslack is the difference (absolute value) between
       % the scheduled slack generation determined from the
         % coordination equation, and the slack generation,
                  % obtained from the power flow solution.
--
while dpslack > 0.001               % Test for convergence...
busout              % Prints the final power flow solution
gencost   % Generation cost with optimum scheduling of gen.
```

This below code shows us the Newton-Raphson method in Optimal Load Flow method which is used to find the most optimal distribution of power generation in power systems.It iteratively solves the nonlinear equations of power flow and adjusts the generator outputs to meet the load demands while considering the system constraints.

```
%   Power flow solution by Newton-Raphson method
ns=0; ng=0; Vm=0; delta=0; yload=0; deltad=0;
nbus = length(busdata(:,1));
for k=1:nbus
n=busdata(k,1);
kb(n)=busdata(k,2); Vm(n)=busdata(k,3); delta(n)=busdata(k, 4);
Pd(n)=busdata(k,5); Qd(n)=busdata(k,6); Pg(n)=busdata(k,7); Qg(n) = busdata(k,8);
Qmin(n)=busdata(k, 9); Qmax(n)=busdata(k, 10);
Qsh(n)=busdata(k, 11);
    if Vm(n) <= 0  Vm(n) = 1.0; V(n) = 1 + j*0;
    else delta(n) = pi/180*delta(n);
         V(n) = Vm(n)*(cos(delta(n)) + j*sin(delta(n)));
         P(n)=(Pg(n)-Pd(n))/basemva;
         Q(n)=(Qg(n)-Qd(n)+ Qsh(n))/basemva;
         S(n) = P(n) + j*Q(n);
    end
end
for k=1:nbus
if kb(k) == 1, ns = ns+1; else, end
if kb(k) == 2 ng = ng+1; else, end
ngs(k) = ng;
nss(k) = ns;
end
Ym=abs(Ybus); t = angle(Ybus);
m=2*nbus-ng-2*ns;
maxerror = 1; converge=1;
iter = 0;
```

```
% Start of iterations
clear A  DC   J  DX
while maxerror >= accuracy & iter <= maxiter % Test for max. power mismatch
for i=1:m
for k=1:m
   A(i,k)=0;      %Initializing Jacobian matrix
end, end
iter = iter+1;
for n=1:nbus
nn=n-nss(n);
lm=nbus+n-ngs(n)-nss(n)-ns;
J11=0; J22=0; J33=0; J44=0;
   for i=1:nbr
     if nl(i) == n | nr(i) == n
        if nl(i) == n,  l = nr(i); end
        if nr(i) == n,  l = nl(i); end
        J11=J11+ Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
        J33=J33+ Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));
        if kb(n)~=1
        J22=J22+ Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));
        J44=J44+ Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
        else, end
        if kb(n) ~= 1  & kb(l) ~=1
        lk = nbus+l-ngs(l)-nss(l)-ns;
        ll = l -nss(l);
      % off diagonalelements of J1
        A(nn, ll) =-Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
              if kb(l) == 0  % off diagonal elements of J2
              A(nn, lk) =Vm(n)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));end
              if kb(n) == 0  % off diagonal elements of J3
              A(lm, ll) =-Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n)+delta(l)); end
              if kb(n) == 0 & kb(l) == 0  % off diagonal elements of  J4
              if kb(n) == 0 & kb(l) == 0  % off diagonal elements of  J4
              A(lm, lk) =-Vm(n)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));end
         else end
      else , end
   end
   Pk = Vm(n)^2*Ym(n,n)*cos(t(n,n))+J33;
   Qk = -Vm(n)^2*Ym(n,n)*sin(t(n,n))-J11;
   if kb(n) == 1 P(n)=Pk; Q(n) = Qk; end   % Swing bus P
     if kb(n) == 2  Q(n)=Qk;
         if Qmax(n) ~= 0
           Qgc = Q(n)*basemva + Qd(n) - Qsh(n);
           if iter <= 7                  % Between the 2th & 6th iterations
              if iter > 2                % the Mvar of generator buses are
                if Qgc  < Qmin(n),       % tested. If not within limits Vm(n)
                Vm(n) = Vm(n) + 0.01;    % is changed in steps of 0.01 pu to
                elseif Qgc  > Qmax(n),   % bring the generator Mvar within
                Vm(n) = Vm(n) - 0.01;end % the specified limits.
              else, end
          else,end
         else,end
     end
   if kb(n) ~= 1
     A(nn,nn) = J11;  %diagonal elements of J1
     DC(nn) = P(n)-Pk;
   end
   if kb(n) == 0
     A(nn,lm) = 2*Vm(n)*Ym(n,n)*cos(t(n,n))+J22;  %diagonal elements of J2
     A(lm,nn)= J33;        %diagonal elements of J3
     A(lm,lm) =-2*Vm(n)*Ym(n,n)*sin(t(n,n))-J44;  %diagonal of elements of J4
     DC(lm) = Q(n)-Qk;
   end
```

```
end
DX=A\DC';
for n=1:nbus
  nn=n-nss(n);
  lm=nbus+n-ngs(n)-nss(n)-ns;
    if kb(n) ~= 1
    delta(n) = delta(n)+DX(nn); end
    if kb(n) == 0
    Vm(n)=Vm(n)+DX(lm); end
 end
  maxerror=max(abs(DC));
     if iter == maxiter & maxerror > accuracy
   fprintf('\nWARNING: Iterative solution did not converged after ')
   fprintf('%g', iter), fprintf(' iterations.\n\n')
   fprintf('Press Enter to terminate the iterations and print the results \n')
   converge = 0; pause, else, end

end

if converge ~= 1
   tech= ('                    ITERATIVE SOLUTION DID NOT CONVERGE'); else,
   tech=('                  Power Flow Solution by Newton-Raphson Method');
end
V = Vm.*cos(delta)+j*Vm.*sin(delta);
deltad=180/pi*delta;
i=sqrt(-1);
k=0;
for n = 1:nbus
     if kb(n) == 1
```

```
      k=k+1;
      S(n)= P(n)+j*Q(n);
      Pg(n) = P(n)*basemva + Pd(n);
      Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n);
      Pgg(k)=Pg(n);
      Qgg(k)=Qg(n);      %june 97
      elseif  kb(n) ==2
      k=k+1;
      S(n)=P(n)+j*Q(n);
      Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n);
      Pgg(k)=Pg(n);
      Qgg(k)=Qg(n);  % June 1997
   end
yload(n) = (Pd(n)- j*Qd(n)+j*Qsh(n))/(basemva*Vm(n)^2);
end
busdata(:,3)=Vm'; busdata(:,4)=deltad';
Pgt = sum(Pg);  Qgt = sum(Qg); Pdt = sum(Pd); Qdt = sum(Qd); Qsht = sum(Qsh);

%clear A DC DX  J11 J22 J33 J44 Qk delta lk ll lm
%clear A DC DX  J11 J22 J33  Qk delta lk ll lm
```

The below code shows the method to calculate the YBUS matrix.To calculate the Y-bus matrix, sum the admittances of all branches connected to each bus for diagonal elements and take the negative of the admittance for off-diagonal elements connecting two buses.

```
%  This program obtains th Bus Admittance Matrix for power flow solution

j=sqrt(-1); i = sqrt(-1);
nl = linedata(:,1); nr = linedata(:,2); R = linedata(:,3);
X = linedata(:,4); Bc = j*linedata(:,5); a = linedata(:, 6);
nbr=length(linedata(:,1)); nbus = max(max(nl), max(nr));
Z = R + j*X; y= ones(nbr,1)./Z;        %branch admittance
for n = 1:nbr
if a(n) <= 0  a(n) = 1; else end
Ybus=zeros(nbus,nbus);     % initialize Ybus to zero
                % formation of the off diagonal elements
for k=1:nbr;
       Ybus(nl(k),nr(k))=Ybus(nl(k),nr(k))-y(k)/a(k);
       Ybus(nr(k),nl(k))=Ybus(nl(k),nr(k));
    end
end
                % formation of the diagonal elements
for  n=1:nbus
     for k=1:nbr
         if nl(k)==n
         Ybus(n,n) = Ybus(n,n)+y(k)/(a(k)^2) + Bc(k);
         elseif nr(k)==n
         Ybus(n,n) = Ybus(n,n)+y(k) +Bc(k);
         else, end
     end
end
clear Pgg

```

This program solves the coordination equation for optimizing power flow based on economic dispatch. To operate, it needs the total load demand (Pdt), the cost function matrix (cost), and the generator MW limits. If the MW limits are not specified, the powerflow will be carried out without considering these constraints. Additionally, if the base MVA and any loss coefficients (B, B0, and B00) are provided, the program will determine the optimal dispatch while incorporating system losses.

```
    clear Pgg
    if exist('Pdt')~=1
    Pdt = input('Enter total demand Pdt = ');
    else, end
    if exist('cost')~=1
    cost = input('Enter the cost matrix, cost = ');
    else, end
    ngg = length(cost(:,1));
    if exist('mwlimits')~=1
    mwlimits= [zeros(ngg, 1), inf*ones(ngg,1)];
    else, end
    if exist('B')~=1
    B = zeros(ngg, ngg);
    else, end
    if exist('B0')~=1
    B0=zeros(1, ngg);
    else, end
    if exist('B00')~=1
    B00=0;
    else, end
    if exist('basemva')~=1
    basemva=100;
    else, end
```

```
clear Pgg
Bu=B/basemva; B00u=basemva*B00;
alpha=cost(:,1); beta=cost(:,2); gama = cost(:,3);
Pmin=mwlimits(:,1); Pmax=mwlimits(:,2);
wgt=ones(1, ngg);
if Pdt > sum(Pmax)
Error1 = ['Total demand is greater than the total sum of maximum generation.'
          'No feasible solution.  Reduce demand or correct generator limits.'];
disp(Error1), return
elseif Pdt < sum(Pmin)
Error2 = ['Total demand is less than the total sum of minimum generation.    '
          'No feasible solution. Increase demand or correct generator limits.'];
disp(Error2), return
else, end
iterp = 0;                                  % Iteration counter
DelP = 10;              % Error in DelP is set to a high value

E=Bu;
if exist('lambda')~=1
lambda=max(beta);
end
while abs(DelP)  >= 0.0001  & iterp < 200     % Test for convergence
iterp = iterp + 1;                           % No. of iterations
for k=1:ngg
    if wgt(k) == 1
       E(k,k) = gama(k)/lambda + Bu(k,k);
       Dx(k) = 1/2*(1 - B0(k)- beta(k)/lambda);
       else, E(k,k)=1;  Dx(k) = 0;
            for m=1:ngg
               if m~=k
                  if m~=k
                  E(k,m)=0;
                  else,end
               end
        end
end
PP=E\Dx';
for k=1:ngg
if wgt(k)==1
   Pgg(k) = PP(k);
   else,end
end
Pgtt = sum(Pgg);
PL=Pgg*Bu*Pgg'+B0*Pgg'+B00u;
DelP =Pdt+PL -Pgtt ;                                    %Residual
for k = 1:ngg
   if Pgg(k) > Pmax(k) & abs(DelP) <=0.001,
   Pgg(k) = Pmax(k); wgt(k) = 0;
   elseif Pgg(k) < Pmin(k) & abs(DelP) <= 0.001
   Pgg(k) = Pmin(k); wgt(k) = 0;
   else, end
end
PL=Pgg*Bu*Pgg'+B0*Pgg'+B00u;
DelP =Pdt +PL - sum(Pgg);                                %Residual
  for k=1:ngg
  BP = 0;
     for m=1:ngg
         if m~=k
            BP = BP + Bu(k,m)*Pgg(m);
            else, end
     end
```

```
    end
  grad(k)=(gama(k)*(1-B0(k))+Bu(k,k)*beta(k)-2*gama(k)*BP)/(2*(gama(k)+lambda*Bu(k,k))^2);
  end
sumgrad=wgt*grad';
Delambda = DelP/sumgrad;                % Change in variable
lambda = lambda + Delambda;             % Successive solution
end
fprintf('Incremental cost of delivered power (system lambda) = %9.6f Rs/MWh \n', lambda*80);
fprintf('Optimal Dispatch of Generation:\n\n')
disp(Pgg')
%fprintf('Total system loss = %g MW \n\n', PL)
ng=length(Pgg);
n=0;
if exist('nbus')==1 | exist('busdata')==1
  for k=1:nbus
     if kb(k)~=0
        n=n+1;
        if n <= ng
        busdata(k,7)=Pgg(n); else, end
     else , end
  end
  if n == ng
     for k=1:nbus
         if kb(k)==1
         dpslack = abs(Pg(k)-busdata(k,7))/basemva;
         fprintf('Absolute value of the slack bus real power mismatch, dpslack = %8.4f pu \n', dpslack)
         else, end
     end
   else, end
 else, end
clear BP Dx DelP Delambda E PP grad sumgrad wgt Bu B00u B B0 B00
```

The below code contains how the generation cost is calculated.We do this by firstly obtaining the power generation values for each generator and then using the cost function, we calculate the generated cost for each generator, the sum of all cost values will give us the total cost generated.

```
%function [totalcost]=gencost(Pgg, cost)
if exist('Pgg')~=1
Pgg=input('Enter the scheduled real power gen. in row matrix ');
else,end
if exist('cost')~=1
cost = input('Enter the cost function matrix ');
else, end
ngg = length(cost(:,1));
Pmt = [ones(1,ngg); Pgg; Pgg.^2];
for i = 1:ngg
costv(i) = cost(i,:)*Pmt(:,i);
end
totalcost=sum(costv)*80;
fprintf('\nTotal generation cost = % 10.2f Rs/h \n', totalcost)
```

This code prints us the values that are calculated in steady state when doing Newton - Raphson method.

```
%clc
disp(tech)
fprintf('                      Maximum Power Mismatch = %g \n', maxerror)
fprintf('                            No. of Iterations = %g \n\n', iter)
head =['    Bus  Voltage  Angle    ------Load------    ---Generation---   Injected'
       '    No.  Mag.     Degree     MW       Mvar       MW       Mvar       Mvar '
       '                                                                          '];
disp(head)
for n=1:nbus
     fprintf(' %5g', n), fprintf(' %7.3f', Vm(n)),
     fprintf(' %8.3f', deltad(n)), fprintf(' %9.3f', Pd(n)),
     fprintf(' %9.3f', Qd(n)),  fprintf(' %9.3f', Pg(n)),
     fprintf(' %9.3f ', Qg(n)), fprintf(' %8.3f\n', Qsh(n))
end
    fprintf('      \n'), fprintf('    Total              ')
    fprintf(' %9.3f', Pdt), fprintf(' %9.3f', Qdt),
    fprintf(' %9.3f', Pgt), fprintf(' %9.3f', Qgt), fprintf(' %9.3f\n\n', Qsht)
```

This program calculates the B-coefficients of the loss formula as the function of real power generation. PL = P*B*P'+B0*P'+B00.

It requires the power flow solution..

```
12 -    clear B B0 B00
13 -    Zbus=inv(Ybus);
14 -    ngg=0;
15 -    I=-1/basemva*(Pd-j*Qd)./conj(V); %new
16 -    ID= sum(I);  %new
17
18 -    for k=1:nbus
19 -      if kb(k)== 0
20      %    I(k) = conj(S(k))/conj(V(k));
21      %   else, ngg=ngg+1; I(k)=0;  end
22 -       else, ngg=ngg+1;  end
23 -      if kb(k)==1 ks=k; else, end
24 -    end
25      %ID= sum(I);
26 -    d1=I/ID;
27 -    DD=sum(d1.*Zbus(ks,:));  %new
28 -    kg=0; kd=0;
29 -    for k=1:nbus
30 -       if kb(k)~=0
31 -       kg=kg+1;
32 -       t1(kg) = Zbus(ks,k)/DD;   %new
33 -       else, kd=kd+1;
34 -       d(kd)=I(k)/ID;
35 -       end
36 -    end
37 -    nd=nbus-ngg;
38 -    C1g=zeros(nbus, ngg);
39 -    kg=0;
```

```
  kg=0;
  for k=1:nbus
    if kb(k)~=0
    kg=kg+1;
      for m=1:ngg
         if kb(m)~=0
          C1g(k, kg)=1;
         else, end
      end
    else,end
  end
  C1gg=eye(ngg,ngg);
  C1D=zeros(ngg,1);
  C1=[C1g,conj(d1)'];
  C2gD=[C1gg; -t1];
  CnD=[C1D;-t1(1)];
  C2=[C2gD,CnD];
  C=C1*C2;
  kg=0;
  for k=1:nbus
    if kb(k)~=0
    kg=kg+1;
    al(kg)=(1-j*((Qg(k)+Qsh(k))/Pg(k)))/conj(V(k));  %new
    else,end
  end
  alp=[al, -V(ks)/Zbus(ks,ks)];
  for k=1:ngg+1
      for m=1:ngg+1
```

```
for k=1:nbus
  if kb(k)~=0
  kg=kg+1;
  al(kg)=(1-j*((Qg(k)+Qsh(k))/Pg(k)))/conj(V(k));  %new
  else,end
end
alp=[al, -V(ks)/Zbus(ks,ks)];
for k=1:ngg+1
    for m=1:ngg+1
    if k==m
    alph(k,k)=alp(k);
    else, alph(k,m)=0;end
end,end
T = alph*conj(C)'*real(Zbus)*conj(C)*conj(alph);
BB=0.5*(T+conj(T));
for k=1:ngg
    for m=1:ngg
    B(k,m)=BB(k,m);
    end
    B0(k)=2*BB(ngg+1,k);
end
B00=BB(ngg+1,ngg+1);
B, B0, B00
PL = Pgg*(B/basemva)*Pgg'+B0*Pgg'+B00*basemva;
fprintf('Total system loss = %g MW \n', PL)
clear I BB C C1 C1D C1g C1gg C2 C2gD CnD DD ID T al alp alph t1 d d1 kd kg ks nd ng
```

The below snippet will be the results that were calculated after doing power flow analysis and economic load dispatch. Here we obtained the results for each iteration during the Newton -

Raphson method, the total cost using this method, and the total cost after doing the economic load dispatch.

```
                  Power Flow Solution by Newton-Raphson Method
                     Maximum Power Mismatch = 1.43025e-05
                           No. of Iterations = 3

    Bus  Voltage  Angle    ------Load------    ---Generation---   Injected
    No.  Mag.     Degree     MW       Mvar       MW       Mvar       Mvar

     1   1.060    0.000     0.000     0.000    83.051     7.271     0.000
     2   1.045   -1.782    20.000    10.000    40.000    41.811     0.000
     3   1.030   -2.664    20.000    15.000    30.000    24.148     0.000
     4   1.019   -3.243    50.000    30.000     0.000     0.000     0.000
     5   0.990   -4.405    60.000    40.000     0.000     0.000     0.000

    Total                150.000    95.000   153.051    73.230     0.000


B =

    0.0218    0.0093    0.0028
    0.0093    0.0228    0.0017
    0.0028    0.0017    0.0179


B0 =

    0.0003    0.0031    0.0015
```

```
B00 =

   3.0523e-04

Total system loss = 3.05248 MW

Total generation cost =  130659.07 Rs/h
Incremental cost of delivered power (system lambda) = 621.408637 Rs/MWh
Optimal Dispatch of Generation:

   33.4558
   64.1101
   55.1005

Absolute value of the slack bus real power mismatch, dpslack =   0.4960 pu

B =

    0.0319    0.0118    0.0034
    0.0118    0.0137    0.0010
    0.0034    0.0010    0.0117


B0 =

    0.0030    0.0015    0.0005
```

```
B00 =

   3.0516e-04

Total system loss = 2.21219 MW
Incremental cost of delivered power (system lambda) = 618.933354 Rs/MWh
Optimal Dispatch of Generation:

   26.7427
   67.9440
   57.4127

Absolute value of the slack bus real power mismatch, dpslack =   0.0626 pu

B =

    0.0400    0.0125    0.0035
    0.0125    0.0132    0.0010
    0.0035    0.0010    0.0116


B0 =

    0.0040    0.0013    0.0005


B00 =

   3.0516e-04



Total system loss = 2.17295 MW
Incremental cost of delivered power (system lambda) = 619.903648 Rs/MWh
Optimal Dispatch of Generation:

   24.9190
   68.9023
   58.3061

Absolute value of the slack bus real power mismatch, dpslack =   0.0189 pu

B =

    0.0439    0.0127    0.0036
    0.0127    0.0131    0.0010
    0.0036    0.0010    0.0115


B0 =

    0.0044    0.0013    0.0004


B00 =

   3.0516e-04

Total system loss = 2.1632 MW
Incremental cost of delivered power (system lambda) = 620.351596 Rs/MWh
Optimal Dispatch of Generation:
```

```
-         -

   24.1668
   69.2685
   58.7069

Absolute value of the slack bus real power mismatch, dpslack =   0.0079 pu

B =

    0.0458    0.0129    0.0036
    0.0129    0.0131    0.0010
    0.0036    0.0010    0.0115


B0 =

    0.0046    0.0013    0.0004


B00 =

   3.0516e-04

Total system loss = 2.15942 MW
Incremental cost of delivered power (system lambda) = 620.565276 Rs/MWh
Optimal Dispatch of Generation:
```

```
Total system loss = 2.15942 MW
Incremental cost of delivered power (system lambda) = 620.565276 Rs/MWh
Optimal Dispatch of Generation:

   23.8165
   69.4362
   58.8966

Absolute value of the slack bus real power mismatch, dpslack =   0.0037 pu

B =

    0.0467    0.0129    0.0036
    0.0129    0.0131    0.0010
    0.0036    0.0010    0.0115


B0 =

    0.0047    0.0012    0.0004


B00 =

   3.0516e-04

Total system loss = 2.15772 MW
Incremental cost of delivered power (system lambda) = 620.670878 Rs/MWh
Optimal Dispatch of Generation:
```

```
Optimal Dispatch of Generation:

   23.6446
   69.5182
   58.9899

Absolute value of the slack bus real power mismatch, dpslack =   0.0018 pu

B =

    0.0472    0.0130    0.0036
    0.0130    0.0130    0.0010
    0.0036    0.0010    0.0115


B0 =

    0.0047    0.0012    0.0004


B00 =

   3.0516e-04

Total system loss = 2.15691 MW
Incremental cost of delivered power (system lambda) = 620.724115 Rs/MWh
Optimal Dispatch of Generation:
```

...

...

```
Total system loss = 2.15691 MW
Incremental cost of delivered power (system lambda) = 620.724115 Rs/MWh
Optimal Dispatch of Generation:

   23.5581
   69.5593
   59.0368

Absolute value of the slack bus real power mismatch, dpslack =   0.0009 pu
                  Power Flow Solution by Newton-Raphson Method
                     Maximum Power Mismatch = 1.90285e-08
                           No. of Iterations = 2

    Bus  Voltage  Angle    ------Load------    ---Generation---   Injected
    No.  Mag.     Degree     MW       Mvar       MW       Mvar       Mvar

     1   1.060    0.000     0.000     0.000    23.649    25.727     0.000
     2   1.045   -0.282    20.000    10.000    69.518    30.767     0.000
     3   1.030   -0.495    20.000    15.000    58.990    14.052     0.000
     4   1.019   -1.208    50.000    30.000     0.000     0.000     0.000
     5   0.990   -2.729    60.000    40.000     0.000     0.000     0.000

    Total                 150.000    95.000   152.157    70.545     0.000

Total generation cost =  127757.17 Rs/h
```

CONCLUSION

Cost calculated after Newton - Raphson Power flow method is - 1,30,659.07 Rs/hr

Cost calculated after Optimal Power Flow(loadflow+ELD) - 1,27,757.17 Rs/hr

So for 1hr - 2901.9 Rs are saved

For 1 day - 69,645.6 Rs are saved

For 1 month - 20,89,368 Rs are saved

For 1 year - 2,50,72,416 Rs are saved.

This is for a small grid (5 Bus system) this much of amount is saved, In larger scale power grids much more money is saved.

The optimal Power flow solution is calculated successfully.