Chapter 2
Load Flow Analysis

2.1 Introduction

Load flow analysis is the most important and essential approach to investigating problems in power system operating and planning. Based on a specified generating state and transmission network structure, load flow analysis solves the steady operation state with node voltages and branch power flow in the power system. Load flow analysis can provide a balanced steady operation state of the power system, without considering system transient processes. Hence, the mathematic model of load flow problem is a nonlinear algebraic equation system without differential equations. Power system dynamic analysis (see Chaps. 5 and 6) investigates system stability under some given disturbances. Its mathematic model includes differential equations. It should be pointed out that dynamic analysis is based on load flow analysis and the algorithm of load flow analysis is also the base for dynamic analysis methods. Therefore, familiarity with the theory and algorithms of load flow analysis is essential to understanding the methodology of modern power system analysis.

Using digital computers to calculate load flow started from the middle of the 1950s. Since then, a variety of methods has been used in load flow calculation. The development of these methods is mainly led by the basic requirements of load flow calculation, which can be summed up as:

1. The convergence properties
2. The computing efficiency and memory requirements
3. The convenience and flexibility of the implementation

Mathematically, the load flow problem is a problem of solving a system of nonlinear algebraic equations. Its solution usually cannot avoid some iteration process. Thus reliable convergence becomes the prime criterion for a load flow calculation method. With the scale of power system continually expanding, the dimension of load flow equations now becomes very high (several thousands to tens of thousands). For the equations with such high dimensions, we cannot ensure that any mathematical method can converge to a correct solution. This situation requires the researchers and scholars in the power system analysis field to seek more reliable methods.

X.-F. Wang et al., *Modern Power Systems Analysis*.
doi: 10.1007/978-0-387-72853-7,

In the early stages of using digital computers to solve power system load flow problems, the widely used method was the Gauss–Seidel iterative method based on a nodal admittance matrix (it will be simply called the admittance method below) [4]. The principle of this method is rather simple and its memory requirement is relatively small. These properties made it suit the level of computer and power system theory at that time. However, its convergence is not satisfactory. When the system scale becomes larger, the number of iteration increases sharply, and sometimes the iteration process cannot converge. This problem led to the use of the sequential substitution method based on the nodal impedance matrix (also called the impedance method).

At the beginning of the 1960s, the digital computer had developed to the second generation. The memory and computing speed of computers were improved significantly, providing suitable conditions for the application of the impedance method. As mentioned in Chap. 1, the impedance matrix is a full matrix. The impedance method requires the computer to store the impedance matrix that represents the topology and parameters of the power network. Thus it needs a great amount of computer memory. Furthermore, in each iteration, every element in the impedance matrix must be operated with, so the computing burden is very heavy.

The impedance method improved convergence and solved some load flow problems that the admittance method could not solve. Therefore, the impedance method was widely applied from then on and made a great contribution to power system design, operation, and research.

The main disadvantage of the impedance method is its high memory requirement and computing burden. The larger the system is, the more serious these defects are. To overcome the disadvantage, the piecewise solution method based on impedance matrix was developed [5]. This method divides a large system up into several small local systems and only the impedance matrixes of local systems and the impedances of tie lines between these local systems are to be stored in the computer. In this way, the memory requirement and computing burden are greatly alleviated.

The other approach to overcoming the disadvantages of the impedance method is to apply the Newton–Raphson method (also called the Newton method) [6]. The Newton method is a typical method used to solve nonlinear equations in mathematics with very favorable convergence. As long as the sparsity of the Jacobean matrix is utilized in the iterative process, the computing efficiency of the Newton method can be greatly improved. Since the optimal order eliminating method [7] began to be employed in the middle of the 1960s, the Newton method has surpassed the impedance method in the aspects of convergence, memory demand, and computing speed. It is still the favored method, and is widely used in load flow calculation today.

Since the 1970s, the load flow calculating method continues to develop in various ways. Among them the most successful is the fast decoupled method, also called the $P - Q$ decoupled method [8]. Comparing with the Newton method, this method is much simpler and more efficient algorithmically, and therefore more popular in many applications.

In the recent 20 years, research on load flow calculation is still very active. Many contributions seek to improve the convergence characteristics of the Newton method and the $P-Q$ decoupled method [9–15]. Along with the development of artificial intelligent theory, the genetic algorithm, artificial neural network algorithm, and fuzzy algorithm have also been introduced to load flow analysis [16–19]. However, until now these new models and new algorithms still cannot replace the Newton method and $P-Q$ decoupled method. Because the scales of power systems continue to expand and the requirements for online calculation become more and more urgent, the parallel computing algorithms are also studied intensively now and may become an important research field [20].

This chapter mainly discusses the currently widely used Newton method and $P-Q$ decoupled method.

The degree of flexibility and convenience of load flow calculation are also very important to computer application. In practice, load flow analysis is usually part of an interactive environment, rather than a pure calculation problem. Therefore, the human–computer interface should be friendly, allowing users to monitor and control the calculation process. To obtain an ideal operation scheme, it is usually necessary to modify the original data according to the computing results. Thus, the computing method should be flexible, permitting users to readily modify and adjust their operation scheme. Input and output processes should also receive careful attention.

Power system steady state analysis includes load flow analysis and static security analysis. Load flow analysis is mainly used in analyzing the normal operation state, while static security analysis is used when some elements are out of service. Its purpose is to check whether the system can operate safely, i.e., if there are equipment overloads, or some node voltages are too low or too high. In principle, static security analysis can be replaced by a series of load flow analyses. However, usually there are very many contingency states to be checked and the computation burden is quite large if a rigorous load flow calculation method is used. Hence special methods have to be developed to meet the requirement of efficient calculation. In the first part of this chapter, the models and algorithms of load flow calculation are introduced. In the second part, the problems related to static security analysis are discussed.

2.2 Formulation of Load Flow Problem

2.2.1 Classification of Node Types

An electric power system is composed of generators, transformers, transmission lines and loads, etc. A simple power system is illustrated in Fig. 2.1. In the process of power system analysis, the static components, such as transformers, transmission lines, shunt capacitors and reactors, are represented by their equivalent circuits

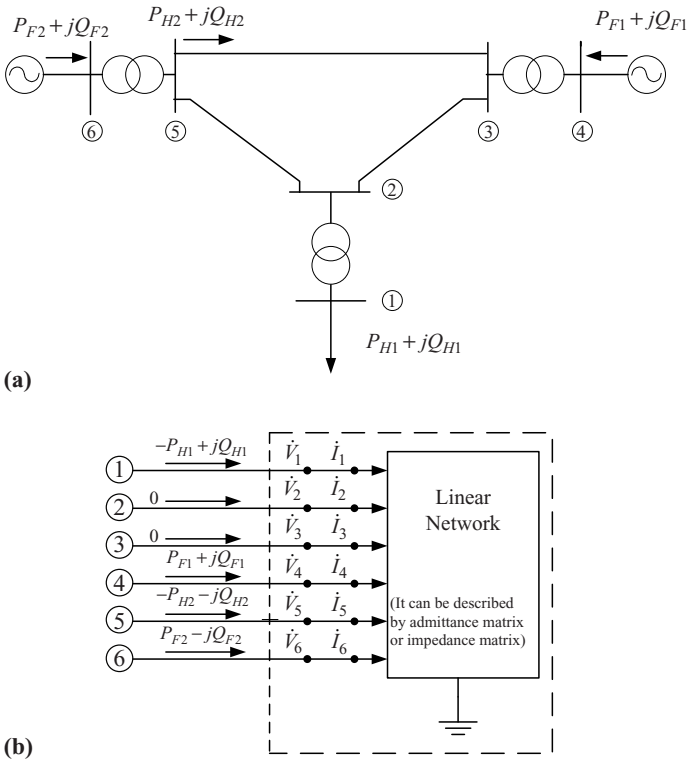

Fig. 2.1 Simple power system

consisting of R, L, C elements. Therefore, the network formed by these static components can be considered as a linear network and represented by the corresponding admittance matrix or impedance matrix. In load flow calculation, the generators and loads are treated as nonlinear components. They cannot be embodied in the linear network, see Fig. 2.1b. The connecting nodes with zero injected power also represent boundary conditions on the network.

In Fig. 2.1b, the relationship between node current and voltage in the linear network can be described by the following node equation:

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

or

$$\dot{I}_i = \sum_{j=1}^{n} Y_{ij}\dot{V}_j \quad (i = 1, 2, \ldots, n) \tag{2.2}$$

where $\dot{I}_i$ and $\dot{V}_j$ are the injected current at bus i and voltage at bus j, respectively, Y_{ij} is an element of the admittance matrix, n is the total number of nodes in the system.

To solve the load flow equation, the relation of node power with current should be used

$$\dot{I}_i = \frac{P_i - jQ_i}{\hat{V}_i} \quad (i = 1, 2, \ldots, n) \tag{2.3}$$

where P_i, Q_i are the injected active and reactive power at node i, respectively. If node i is a load node, then P_i and Q_i should take negative values. In (2.3), $\hat{V}_i$ is the conjugate of the voltage vector at node i. Substituting (2.3) to (2.2), we have,

$$\frac{P_i - jQ_i}{\hat{V}_i} = \sum_{j=1}^{n} Y_{ij}\dot{V}_j \quad (i = 1, 2, \ldots, n)$$

or

$$\frac{P_i + jQ_i}{\dot{V}_i} = \sum_{j=1}^{n} \hat{Y}_{ij}\hat{V}_j \quad (i = 1, 2, \ldots, n) \tag{2.4}$$

There are n nonlinear complex equations in (2.4). They are the principal equations in load flow calculation. Based on different methods to solve (2.4), various load flow algorithms can be formed.

In the power system load flow problem, the variables are nodal complex voltages and complex powers: V, θ, P, Q. If there are n nodes in a power system, then the total number of variables is $4n$.

As mentioned above, there are n complex equations or $2n$ real equations defined in principal by (2.4), thus only $2n$ variables can be solved from these equations, while the other $2n$ variables should be specified as original data.

Usually, two variables at each node are assumed known, while the other two variables are treated as state variables to be resolved. According to the original data, the nodes in power systems can be classified into three types:

1. ***PQ* Nodes:** For PQ nodes, the active and reactive power (P, Q) are specified as known parameters, and the complex voltage (V, θ) is to be resolved. Usually, substation nodes are taken as PQ nodes where the load powers are given constants. When output P and Q are fixed in some power plants, these nodes can also be taken as PQ node. Most nodes in power systems belong to the PQ type in load flow calculation.
2. ***PV* Nodes:** For PV nodes, active power P and voltage magnitude V are specified as known variables, while reactive power Q and voltage angle θ are to be resolved. Usually, PV nodes should have some controllable reactive power resources and can thus maintain node voltage magnitude at a desirable value. Generally speaking, the buses of power plants can be taken as PV nodes, because voltages at these buses can be controlled with reactive power capacity of their generators. Some substations can also be considered as PV nodes when they have enough reactive power compensation devices to control the voltage.

3. **Slack Node:** In load flow studies, there should be one and only one slack node specified in the power system, which is specified by a voltage, constant in magnitude and phase angle. Therefore, V and θ are given as known variables at the slack node, while the active power P and reactive power Q are the variables to be solved. The effective generator at this node supplies the losses to the network. This is necessary because the magnitude of losses will not be known until the calculation of currents is complete, and this cannot be achieved unless one node has no power constraint and can feed the required losses into the system. The location of the slack node can influence the complexity of the calculations; the node most closely approaching a large AGC power station should be used.

We will employ different methods to treat the above three kinds of nodes in power flow calculations.

2.2.2 Node Power Equations

As described above, power system load flow calculations can be roughly considered as the problem of solving the node voltage phasor for each node when the injecting complex power is specified. If the complex power can be represented by equations of complex voltages, then a nonlinear equation solving method, such as the Newton–Raphson method, can be used to solve the node voltage phasors. In this section, node power equations are deduced first.

The complex node voltage has two representation forms – the polar form and the rectangular form. Accordingly, the node power equations also have two forms.

From (2.4), the node power equations can be expressed as

$$P_i + jQ_i = \dot{V}_i \sum_{j \in i} \hat{Y}_{ij} \hat{V}_j \quad (i = 1, 2, \ldots, n) \tag{2.5}$$

where $j \in i$ means the node j should be directly connected with node i, including $j = i$. As discussed in Chap.1, the admittance matrix is a sparse matrix, and the terms in Σ are correspondingly few. If the voltage vector of (2.5) adopts polar form,

$$\dot{V}_i = V_i e^{j\theta_i} \tag{2.6}$$

where V_i,θ_i are the magnitude and phase angle of voltage at node i. The elements of admittance matrix can be expressed as

$$Y_{ij} = G_{ij} + jB_{ij}$$

Hence (2.5) can be rewritten as

$$P_i + jQ_i = V_i e^{j\theta_i} \sum_{j \in i} (G_{ij} - jB_{ij}) V_j e^{-j\theta_j} \quad (i = 1, 2, \ldots, n) \tag{2.7}$$

Combining the exponential items of above equation and using the relationship

$$e^{j\theta} = \cos\theta + j \sin\theta$$

we have,

$$P_i + jQ_i = V_i \sum_{j \in i} V_j (G_{ij} - jB_{ij})(\cos\theta_{ij} + j \sin\theta_{ij}) \quad (i = 1, 2, \ldots, n) \tag{2.8}$$

where $\theta_{ij} = \theta_i - \theta_j$, is the voltage phase angle difference between node i and j. Dividing above equations into real and imaginary parts,

$$\left.\begin{aligned} P_i &= V_i \sum_{j \in i} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \\ Q_i &= V_i \sum_{j \in i} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \end{aligned}\right\} \quad (i = 1, 2, \cdots, n) \tag{2.9}$$

This is the polar form of the nodal power equations. It is not only very important in the Newton–Raphson calculation process, but also essential to establish the fast decoupled method.

When the voltage vector is expressed in rectangular form,

$$\dot{V}_i = e_i + jf_i$$

where

$$e_i = V_i \cos\theta_i \quad f_i = V_i \sin\theta_i$$

We can obtain from (2.5),

$$\left.\begin{aligned} P_i &= e_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) + f_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) \\ Q_i &= f_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) - e_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) \end{aligned}\right\} \quad (i = 1, 2, \ldots, n) \tag{2.10}$$

Let

$$\left.\begin{aligned}\sum_{j\in i}(G_{ij}e_j - B_{ij}f_j) &= a_i\\ \sum_{j\in i}(G_{ij}f_j + B_{ij}e_j) &= b_i\end{aligned}\right\} \tag{2.11}$$

Obviously, a_i and b_i are the real and imaginary parts of injected current at node i and (2.10) can be simplified as,

$$\left.\begin{aligned}P_i &= e_i a_i + f_i b_i\\ Q_i &= f_i a_i - e_i b_i\end{aligned}\right\} \quad (i = 1, 2, \ldots, n) \tag{2.12}$$

This is the rectangular form of the nodal power equations.

Both (2.9) and (2.10) are the simultaneous nonlinear equations of node voltage phasors. They are usually expressed as the following forms as mathematical models of the load flow problem:

$$\left.\begin{aligned}\Delta P_i &= P_{is} - V_i\sum_{j\in i}V_j(G_{ij}\cos\theta_{ij} + B_{ij}\sin\theta_{ij}) = 0\\ \Delta Q_i &= Q_{is} - V_i\sum_{j\in i}V_j(G_{ij}\sin\theta_{ij} - B_{ij}\cos\theta_{ij}) = 0\end{aligned}\right\} \quad (i = 1, 2, \ldots, n) \tag{2.13}$$

and

$$\left.\begin{aligned}\Delta P_i &= P_{is} - e_i\sum_{j\in i}(G_{ij}e_j - B_{ij}f_j) - f_i\sum_{j\in i}(G_{ij}f_j + B_{ij}e_j) = 0\\ \Delta Q_i &= Q_{is} - f_i\sum_{j\in i}(G_{ij}e_j - B_{ij}f_j) + e_i\sum_{j\in i}(G_{ij}f_j + B_{ij}e_j) = 0\end{aligned}\right\} \tag{2.14}$$
$$(i = 1, 2, \ldots, n)$$

where P_{is}, Q_{is} are the specified active and reactive powers at node i. Based on the above two simultaneous equations, the load flow problem can be roughly summarized as: for specified P_{is},Q_{is} $(i = 1, 2, \ldots, n)$, find voltage vector V_i, θ_i or e_i,f_i $(i = 1, 2, \ldots, n)$, such that the magnitudes of the power errors $\Delta P_i, \Delta Q_i$, $(i = 1, 2, \ldots, n)$ of (2.13) or (2.14) are less then an acceptable tolerance.

2.3 Load Flow Solution by Newton Method

2.3.1 Basic Concept of Newton Method

The Newton–Raphson method is an efficient algorithm to solve nonlinear equations. It transforms the procedure of solving nonlinear equations into the procedure

of repeatedly solving linear equations. This sequential linearization process is the core of the Newton–Raphson method. We now introduce the Newton–Raphson method by the following nonlinear equation example,

$$f(x) = 0 \tag{2.15}$$

Let $x^{(0)}$ be the initial guess value of the above equation solution. Assume the real solution x is close to $x^{(0)}$,

$$x = x^{(0)} - \Delta x^{(0)} \tag{2.16}$$

where $\Delta x^{(0)}$ is a modification value of $x^{(0)}$. The following equation holds,

$$f(x^{(0)} - \Delta x^{(0)}) = 0 \tag{2.17}$$

When $\Delta x^{(0)}$ is known, the solution x can be calculated by (2.16). Expanding this function in a Taylor series expansion about point $x^{(0)}$yields:

$$\begin{aligned} f(x^{(0)} - \Delta x^{(0)}) = & f(x^{(0)}) - f'(x^{(0)})\Delta x^{(0)} + f''(x^{(0)})\frac{(\Delta x^{(0)})^2}{2!} - \\ & \cdots + (-1)^n f^{(n)}(x^{(0)})\frac{(\Delta x^{(0)})^n}{n!} + \cdots = 0 \end{aligned} \tag{2.18}$$

where $f'(x^{(0)}),\ldots, f^{(n)}(x^{(0)})$ are the different order partial derivatives of $f(x)$ at $x^{(0)}$. If the initial guess is sufficiently close to the actual solution, the higher order terms of the Taylor series expansion could be neglected. Equation (2.18) becomes,

$$f(x^{(0)}) - f'(x^{(0)})\Delta x^{(0)} = 0 \tag{2.19}$$

This is a linear equation in $\Delta x^{(0)}$ and can be easily solved.

Using $\Delta x^{(0)}$ to modify $x^{(0)}$, we can get $x^{(1)}$:

$$x^{(1)} = x^{(0)} - \Delta x^{(0)} \tag{2.20}$$

$x^{(1)}$ may be more close to the actual solution. Then using $x^{(1)}$ as the new guess value, we solve the following equation similar to (2.19),

$$f(x^{(1)}) - f'(x^{(1)})\Delta x^{(1)} = 0$$

Thus $x^{(2)}$is obtained:

$$x^{(2)} = x^{(1)} - \Delta x^{(1)} \tag{2.21}$$

Repeating this procedure, we establish the correction equation in the tth iteration:

$$f(x^{(t)}) - f'(x^{(t)})\Delta x^{(t)} = 0 \tag{2.22}$$

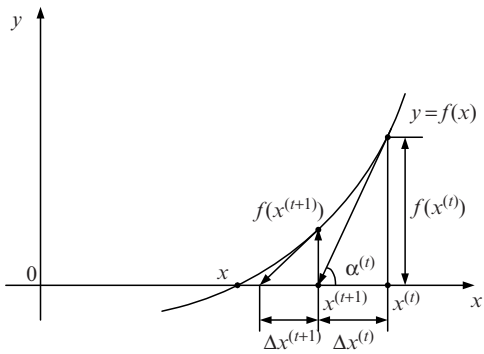

Fig. 2.2 Geometric interpretation of Newton method

or

$$f(x^{(t)}) = f'(x^{(t)})\Delta x^{(t)} \tag{2.23}$$

The left hand of the above equation can be considered as the error produced by approximate solution $x^{(t)}$. When $f(x^{(t)}) \to 0$, (2.15) is satisfied, so $x^{(t)}$ is the solution of the equation. In (2.22), $f'(x^{(t)})$ is the first-order partial derivative of function $f(x)$ at point $x^{(t)}$. It is also the slope of the curve at point $x^{(t)}$, as shown in Fig. 2.2,

$$\tan\alpha^{(t)} = f'(x^{(t)}) \tag{2.24}$$

The correction value $\Delta x^{(t)}$ is determined by the intersection of the tangent line at $x^{(t)}$ with the abscissa. We can comprehend the iterative process more intuitively from Fig. 2.2.

Now we will extend the Newton method to simultaneous nonlinear equations. Assume the nonlinear equations with variables $x_1, x_2, \ldots, x_n$;

$$\left.\begin{aligned} &f_1(x_1, x_2, \ldots, x_n) = 0 \\ &f_2(x_1, x_2, \ldots, x_n) = 0 \\ &\vdots \\ &f_n(x_1, x_2, \ldots, x_n) = 0 \end{aligned}\right\} \tag{2.25}$$

Specify the initial guess values of all variables $x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}$, let $\Delta x_1^{(0)}$, $\Delta x_2^{(0)}, \ldots, \Delta x_n^{(0)}$ be the correction values to satisfy the following equations,

$$\left.\begin{aligned} &f_1(x_1^{(0)} - \Delta x_1^{(0)}, x_2^{(0)} - \Delta x_2^{(0)}, \ldots, x_n^{(0)} - \Delta x_n^{(0)}) = 0 \\ &f_2(x_1^{(0)} - \Delta x_1^{(0)}, x_2^{(0)} - \Delta x_2^{(0)}, \ldots, x_n^{(0)} - \Delta x_n^{(0)}) = 0 \\ &\vdots \\ &f_n(x_1^{(0)} - \Delta x_1^{(0)}, x_2^{(0)} - \Delta x_2^{(0)}, \ldots, x_n^{(0)} - \Delta x_n^{(0)}) = 0 \end{aligned}\right\} \tag{2.26}$$

Expanding the above n equations via the multivariate Taylor series and neglecting the higher order terms, we have the following equations,

$$\left.\begin{aligned}
&f_1(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) - \left[\frac{\partial f_1}{\partial x_1}\Big|_0 \Delta x_1^{(0)} + \frac{\partial f_1}{\partial x_2}\Big|_0 \Delta x_2^{(0)} + , \ldots, + \frac{\partial f_1}{\partial x_n}\Big|_0 \Delta x_n^{(0)}\right] = 0 \\
&f_2(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) - \left[\frac{\partial f_2}{\partial x_1}\Big|_0 \Delta x_1^{(0)} + \frac{\partial f_2}{\partial x_2}\Big|_0 \Delta x_2^{(0)} + , \ldots, + \frac{\partial f_2}{\partial x_n}\Big|_0 \Delta x_n^{(0)}\right] = 0 \\
&\vdots \\
&f_n(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) - \left[\frac{\partial f_n}{\partial x_1}\Big|_0 \Delta x_1^{(0)} + \frac{\partial f_n}{\partial x_2}\Big|_0 \Delta x_2^{(0)} + , \ldots, + \frac{\partial f_n}{\partial x_n}\Big|_0 \Delta x_n^{(0)}\right] = 0
\end{aligned}\right\} \quad (2.27)$$

here $\frac{\partial f_i}{\partial x_j}|_0$ is the partial derivative of function $f_i(x_1, x_2, \ldots, x_n)$ over independent variable x_j at point $(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)})$. Rewrite the above equation in matrix form,

$$\begin{bmatrix} f_1(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) \\ f_2(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) \\ \vdots \\ f_n(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}\Big|_0 \frac{\partial f_1}{\partial x_2}\Big|_0 \cdots \frac{\partial f_1}{\partial x_n}\Big|_0 \\ \frac{\partial f_2}{\partial x_1}\Big|_0 \frac{\partial f_2}{\partial x_2}\Big|_0 \cdots \frac{\partial f_2}{\partial x_n}\Big|_0 \\ \vdots \\ \frac{\partial f_n}{\partial x_1}\Big|_0 \frac{\partial f_n}{\partial x_2}\Big|_0 \cdots \frac{\partial f_n}{\partial x_n}\Big|_0 \end{bmatrix} \begin{bmatrix} \Delta x_1^{(0)} \\ \Delta x_2^{(0)} \\ \vdots \\ \Delta x_n^{(0)} \end{bmatrix} \quad (2.28)$$

This is a set of simultaneous linear equations in the variables $\Delta x_1^{(0)}, \Delta x_2^{(0)}, \ldots, \Delta x_n^{(0)}$, usually called the correction equations of the Newton–Raphson method. After solving $\Delta x_1^{(0)}, \Delta x_2^{(0)}, \ldots, \Delta x_n^{(0)}$, we can get,

$$\left.\begin{aligned}
x_1^{(1)} &= x_1^{(0)} - \Delta x_1^{(0)} \\
x_2^{(1)} &= x_2^{(0)} - \Delta x_2^{(0)} \\
\vdots \quad &\quad \vdots \quad \quad \vdots \\
x_n^{(1)} &= x_n^{(0)} - \Delta x_n^{(0)}
\end{aligned}\right\} \quad (2.29)$$

$x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}$ will approach the actual solution more closely. The updated values are used as the new guess to solve the correction equation (2.28) and to further correct the variables. In this way the iterative process of the Newton–Raphson method is formed.

Generally, the correction equation in the tth iteration can be written as,

$$\begin{bmatrix} f_1(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \\ f_2(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \\ \vdots \quad \vdots \quad \vdots \\ f_n(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}\Big|_t \frac{\partial f_1}{\partial x_2}\Big|_t \cdots \frac{\partial f_1}{\partial x_n}\Big|_t \\ \frac{\partial f_2}{\partial x_1}\Big|_t \frac{\partial f_2}{\partial x_2}\Big|_t \cdots \frac{\partial f_2}{\partial x_n}\Big|_t \\ \vdots \quad \vdots \quad \vdots \\ \frac{\partial f_n}{\partial x_1}\Big|_t \frac{\partial f_n}{\partial x_2}\Big|_t \cdots \frac{\partial f_n}{\partial x_n}\Big|_t \end{bmatrix} \begin{bmatrix} \Delta x_1^{(t)} \\ \Delta x_2^{(t)} \\ \vdots \\ \Delta x_n^{(t)} \end{bmatrix} \quad (2.30)$$

or expressed in matrix form,

$$\mathbf{F}(\mathbf{X}^{(t)}) = \mathbf{J}^{(t)}\Delta\mathbf{X}^{(t)} \tag{2.31}$$

where

$$\mathbf{F}(\mathbf{X}^{(t)}) = \begin{bmatrix} f_1(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \\ f_2(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \\ \vdots \\ f_n(x_1^{(t)}, x_2^{(t)}, \ldots, x_n^{(t)}) \end{bmatrix} \tag{2.32}$$

is the error vector in the tth iteration;

$$\mathbf{J}^{(t)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}|_t \frac{\partial f_1}{\partial x_2}|_t \cdots \frac{\partial f_1}{\partial x_n}|_t \\ \frac{\partial f_2}{\partial x_1}|_t \frac{\partial f_2}{\partial x_2}|_t \cdots \frac{\partial f_2}{\partial x_n}|_t \\ \vdots \\ \frac{\partial f_n}{\partial x_1}|_t \frac{\partial f_n}{\partial x_2}|_t \cdots \frac{\partial f_n}{\partial x_n}|_t \end{bmatrix} \tag{2.33}$$

is the Jacobian matrix of tth iteration;

$$\Delta\mathbf{X}^{(t)} = \begin{bmatrix} \Delta x_1^{(t)} \\ \Delta x_2^{(t)} \\ \vdots \\ \Delta x_n^{(t)} \end{bmatrix} \tag{2.34}$$

is the correction value vector in the tth iteration.

We also have the equation similar to (2.29),

$$\mathbf{X}^{(t+1)} = \mathbf{X}^{(t)} - \Delta\mathbf{X}^{(t)} \tag{2.35}$$

With (2.31) and (2.35) solved alternately in each iteration, $X^{(t+1)}$ gradually approaches the actual solution. Convergence can be evaluated by the norm of the correction value,

$$\left\|\Delta\mathbf{X}^{(t)}\right\| < \varepsilon_1 \tag{2.36}$$

or by the norm of the function,

$$\left\|\mathbf{F}(\mathbf{X}^{(t)})\right\| < \varepsilon_2 \tag{2.37}$$

Here ε_1 and ε_2 are very small positive numbers specified beforehand.

2.3.2 Correction Equations

In Section 2.3.1, we derived two forms of the nodal power equations. Either can be applied in the load flow calculation model.

When the polar form (2.13) is used, the node voltage magnitudes and angles V_i,θ_i $(i = 1, 2, \ldots, n)$ are the variables to be solved. For a PV node, the magnitude of the voltage is specified. At the same time, its reactive power Q_{is} cannot be fixed beforehand as a constraint. Therefore, the reactive equations relative to PV nodes should not be considered in the iterative process. These equations will be used only to calculate the reactive power of each PV node after the iterative process is over and all node voltages have been calculated. Similarly, the voltage magnitude and angle of the slack node are specified, hence the related power equations do not appear in the iterative process. When the iteration has converged, the active and reactive power of the slack node can be calculated by using these power equations.

Assume that total number of system nodes is n, the number of PV nodes is r. For convenience, let the slack bus be the last node, i.e., node n.Therefore, we have $n-1$ active power equations,

$$\left.\begin{aligned}
&\Delta P_1 = P_{1s} - V_1 \sum_{j\in 1} V_j(G_{1j}\cos\theta_{1j} + B_{1j}\sin\theta_{1j}) = 0\\
&\Delta P_2 = P_{2s} - V_2 \sum_{j\in 2} V_j(G_{2j}\cos\theta_{2j} + B_{2j}\sin\theta_{2j}) = 0\\
&\vdots\\
&\Delta P_{n-1} = P_{n-1,s} - V_{n-1} \sum_{j\in(n-1)} V_j(G_{n-1,j}\cos\theta_{n-1,j} + B_{n-1,j}\sin\theta_{n-1,j}) = 0
\end{aligned}\right\} \tag{2.38}$$

and $n-r-1$ reactive power equations.

$$\left.\begin{aligned}
&\Delta Q_1 = Q_{1s} - V_1 \sum_{j\in 1} V_j(G_{1j}\sin\theta_{1j} - B_{1j}\cos\theta_{1j}) = 0\\
&\Delta Q_2 = Q_{2s} - V_2 \sum_{j\in 2} V_j(G_{2j}\sin\theta_{2j} - B_{2j}\cos\theta_{2j}) = 0\\
&\vdots\\
&\Delta Q_{n-1} = Q_{n-1,s} - V_{n-1} \sum_{j\in(n-1)} V_j(G_{n-1,j}\sin\theta_{n-1,j} - B_{n-1,j}\cos\theta_{n-1,j}) = 0
\end{aligned}\right\} \tag{2.39}$$

In the above equations, node voltage angle θ_i and magnitude V_i are the variables to be resolved. Here the number of θ_i is $n-1$ and the number of V_i is $n-r-1$. There

are $2n-r-2$ unknown variables in total and they can be solved by the above $2n-r-2$ equations.

Expanding (2.38) and (2.39) in a Taylor series, neglecting the high-order terms, the correction equation can be written as,

$$\begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \vdots \\ \Delta P_{n-1} \\ \cdots\cdots \\ \Delta Q_1 \\ \Delta Q_2 \\ \vdots \\ \Delta Q_{n-1} \end{bmatrix} = \left[\begin{array}{cccc:cccc} H_{11} & H_{12} & \cdots & H_{1,n-1} & N_{11} & N_{12} & \cdots & N_{1,n-1} \\ H_{21} & H_{22} & \cdots & H_{2,n-1} & N_{21} & N_{22} & \cdots & N_{2,n-1} \\ & & \cdots & & & & \cdots & \\ H_{n-1,1} & H_{n-1,2} & \cdots & H_{n-1,n-1} & N_{n-1,1} & N_{n-1,2} & \cdots & N_{n-1,n-1} \\ \cdots\cdots & \cdots\cdots & \cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots & \cdots\cdots \\ J_{11} & J_{12} & \cdots & J_{1,n-1} & L_{11} & L_{12} & \cdots & L_{1,n-1} \\ J_{21} & J_{22} & \cdots & J_{2,n-1} & L_{21} & L_{22} & \cdots & L_{2,n-1} \\ & & \cdots & & & & \cdots & \\ J_{n-1,1} & J_{n-1,2} & \cdots & J_{n-1,n-1} & J_{n-1,1} & J_{n-1,2} & \cdots & J_{n-1,n-1} \end{array}\right] \times \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \vdots \\ \Delta\theta_{n-1} \\ \cdots\cdots \\ \Delta V_1/V_1 \\ \Delta V_2/V_2 \\ \vdots \\ \Delta V_{n-1}/V_{n-1} \end{bmatrix} \tag{2.40}$$

The form of the voltage magnitude correction values represented here, $\Delta V_1/V_1$, $\Delta V_2/V_2, \ldots, \Delta V_{n-1}/V_{n-1}$, allow the elements in the Jacobian matrix to have similar expressions.

Taking partial derivations of (2.38), or (2.39), and noting that both P_{is}, Q_{is} are constants, we can obtain the elements of the Jacobian matrix as,

$$H_{ij} = \frac{\partial \Delta P_i}{\partial \theta_j} = -V_i V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \quad j \neq i \tag{2.41}$$

$$H_{ii} = \frac{\partial \Delta P_i}{\partial \theta_i} = V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \tag{2.42}$$

or

$$H_{ii} = V_i^2 B_{ii} + Q_i \tag{2.43}$$

$$N_{ij} = \frac{\partial \Delta P_i}{\partial V_j} V_j = -V_i V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \quad j \neq i \tag{2.44}$$

$$N_{ii} = \frac{\partial \Delta P_i}{\partial V_i} V_i = -V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) - 2V_i^2 G_{ii} = -V_i^2 G_{ii} - P_i \tag{2.45}$$

$$J_{ij} = \frac{\partial \Delta P_i}{\partial \theta_j} = V_i V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \quad j \neq i \tag{2.46}$$

$$J_{ii} = \frac{\partial \Delta P_i}{\partial \theta_j} = -V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) = V_i^2 G_{ii} - P_i \tag{2.47}$$

$$L_{ij} = \frac{\partial \Delta Q_i}{\partial V_j} V_j = -V_i V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) \quad j \neq i \tag{2.48}$$

$$L_{ii} = \frac{\partial \Delta Q_i}{\partial V_i} V_i = -V_i \sum_{\substack{j \in i \\ j \neq i}} V_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}) + 2V_i^2 B_{ii} = V_i^2 B_{ii} - Q_i \tag{2.49}$$

The concise form of (2.40) is

$$\begin{bmatrix} \Delta \mathbf{P} \\ \Delta \mathbf{Q} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & \mathbf{N} \\ \mathbf{J} & \mathbf{L} \end{bmatrix} \begin{bmatrix} \Delta\theta \\ \Delta \mathbf{V}/\mathbf{V} \end{bmatrix} \tag{2.50}$$

Comparing (2.50) with (2.40), the meaning of elements is obvious. The correction equation can be rearranged into the following form for convenience,

$$\begin{bmatrix} \Delta P_1 \\ \Delta Q_1 \\ \Delta P_2 \\ \Delta Q_2 \\ \vdots \\ \Delta P_{n-1} \\ \Delta Q_{n-1} \end{bmatrix} = \begin{bmatrix} H_{11} & N_{11} & H_{12} & N_{12} & \cdots & H_{1,n-1} & N_{1,n-1} \\ J_{11} & L_{11} & J_{12} & L_{12} & \cdots & J_{1,n-1} & L_{1,n-1} \\ H_{21} & N_{21} & H_{22} & N_{22} & \cdots & H_{2,n-1} & N_{2,n-1} \\ J_{21} & L_{21} & J_{22} & L_{22} & \cdots & J_{2,n-1} & L_{2,n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ H_{n-1,1} & N_{n-1,1} & H_{n-1,2} & N_{n-1,2} & \cdots & H_{n-1,n-1} & N_{n-1,n-1} \\ J_{n-1,1} & L_{n-1,1} & J_{n-1,2} & L_{n-1,2} & \cdots & J_{n-1,n-1} & L_{n-1,n-1} \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \Delta V_1/V_1 \\ \Delta\theta_2 \\ \Delta V_2/V_2 \\ \vdots \\ \Delta\theta_{n-1} \\ \Delta V_{n-1}/V_{n-1} \end{bmatrix} \tag{2.51}$$

When the rectangular form is adopted in the load flow model, the state variables to be solved are the real and imaginary parts of voltages, i.e., $e_1, f_1, e_2, f_2, \ldots, e_n, f_n$. Since the voltage phasor of the slack node is specified, the number of state variables is $2(n-1)$. We need $2(n-1)$ equations to solve these variables. In fact, every node has two equations except the slack bus. For PQ nodes, P_{is}, Q_{is} are given, so the equations are

$$\left.\begin{aligned} \Delta P_i &= P_{is} - e_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) - f_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) = 0 \\ \Delta Q_i &= Q_{is} - f_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) + e_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) = 0 \end{aligned}\right\} \tag{2.52}$$

For PV nodes, P_{is}, V_{is} are given, so the equations are

$$\left.\begin{aligned} \Delta P_i &= P_{is} - e_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) - f_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) = 0 \\ \Delta V_i^2 &= V_{is}^2 - (e_i^2 + f_i^2) = 0 \end{aligned}\right\} \tag{2.53}$$

There are $2(n-1)$ equations included in (2.52) and (2.53). Expanding them in a Taylor series expansion, neglecting the higher order terms, we can obtain the correction equation as follows,

$$\begin{bmatrix} \Delta P_1 \\ \Delta Q_1 \\ \Delta P_2 \\ \Delta Q_2 \\ \vdots \\ \Delta P_i \\ \Delta V_i^2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \frac{\partial \Delta P_1}{\partial e_1} & \frac{\partial \Delta P_1}{\partial f_1} & \frac{\partial \Delta P_1}{\partial e_2} & \frac{\partial \Delta P_1}{\partial f_2} & \cdots & \frac{\partial \Delta P_1}{\partial e_i} & \frac{\partial \Delta P_1}{\partial f_i} & \cdots \\ \frac{\partial \Delta Q_1}{\partial e_1} & \frac{\partial \Delta Q_1}{\partial f_1} & \frac{\partial \Delta Q_1}{\partial e_2} & \frac{\partial \Delta Q_1}{\partial f_2} & \cdots & \frac{\partial \Delta Q_1}{\partial e_i} & \frac{\partial \Delta Q_1}{\partial f_i} & \cdots \\ \frac{\partial \Delta P_2}{\partial e_1} & \frac{\partial \Delta P_2}{\partial f_1} & \frac{\partial \Delta P_2}{\partial e_2} & \frac{\partial \Delta P_2}{\partial f_2} & \cdots & \frac{\partial \Delta P_2}{\partial e_i} & \frac{\partial \Delta P_2}{\partial f_i} & \cdots \\ \frac{\partial \Delta Q_2}{\partial e_1} & \frac{\partial \Delta Q_2}{\partial f_1} & \frac{\partial \Delta Q_2}{\partial e_2} & \frac{\partial \Delta Q_2}{\partial f_2} & \cdots & \frac{\partial \Delta Q_2}{\partial e_i} & \frac{\partial \Delta Q_2}{\partial f_i} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \Delta P_i}{\partial e_1} & \frac{\partial \Delta P_i}{\partial f_1} & \frac{\partial \Delta P_i}{\partial e_2} & \frac{\partial \Delta P_i}{\partial f_2} & \cdots & \frac{\partial \Delta P_i}{\partial e_i} & \frac{\partial \Delta P_i}{\partial f_i} & \cdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{\partial \Delta V_i^2}{\partial e_i} & \frac{\partial \Delta V_i^2}{\partial f_i} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} \Delta e_1 \\ \Delta f_1 \\ \Delta e_2 \\ \Delta f_2 \\ \vdots \\ \Delta e_i \\ \Delta f_i \\ \vdots \end{bmatrix} \tag{2.54}$$

By differentiating (2.52) and (2.53), we can obtain elements of the Jacobian matrix. The off-diagonal elements of the Jacobian matrix for $j \neq i$ can be expressed as,

$$\left.\begin{aligned} \frac{\partial \Delta P_i}{\partial e_j} &= -\frac{\partial \Delta Q_i}{\partial f_j} = -(G_{ij}e_i + B_{ij}f_i) \\ \frac{\partial \Delta P_i}{\partial f_j} &= \frac{\partial \Delta Q_i}{\partial e_j} = B_{ij}e_i - G_{ij}f_i \\ \frac{\partial \Delta V_i^2}{\partial e_j} &= -\frac{\partial \Delta V_i^2}{\partial f_j} = 0 \end{aligned}\right\} \tag{2.55}$$

The diagonal elements of the Jacobian matrix for $j = i$,

$$\frac{\partial \Delta P_i}{\partial e_i} = -\sum_{j \in i} (G_{ij}e_j - B_{ij}f_j) - G_{ii}e_i - B_{ii}f_i$$

Using (2.11), we can rewrite the above expression as

$$\frac{\partial \Delta P_i}{\partial e_i} = -a_i - G_{ii}e_i - B_{ii}f_i$$

and can obtain the following elements similarly,

$$\left.\begin{aligned}
\frac{\partial \Delta Q_i}{\partial f_i} &= -\sum_{j\in i}(G_{ij}e_j - B_{ij}f_j) + G_{ii}e_i + B_{ii}f_i = -a_i + G_{ii}e_i + B_{ii}f_i \\
\frac{\partial \Delta P_i}{\partial f_i} &= -\sum_{j\in i}(G_{ij}f_j + B_{ij}e_j) + B_{ii}e_i - G_{ii}f_i = -b_i + B_{ii}e_i - G_{ii}f_i \\
\frac{\partial \Delta Q_i}{\partial e_i} &= \sum_{j\in i}(G_{ij}f_j + B_{ij}e_j) + B_{ii}e_i - G_{ii}f_i = b_i + B_{ii}e_i - G_{ii}f_i \\
\frac{\partial \Delta V_i^2}{\partial e_i} &= -2e_i \\
\frac{\partial \Delta V_i^2}{\partial f_i} &= -2f_i
\end{aligned}\right\} \quad (2.56)$$

The correction equations, in either polar form or rectangular form, are the basic equations that need repeatedly solving in Newton–Raphson load flow calculation. Investigating these equations, we can observe the following properties:

1. Equations (2.54) and (2.40) include $2(n-1)$ and $2(n-1)-r$ equations respectively.
2. From the expression of the off-diagonal elements of the Jacobian matrix either in polar form or in rectangular form, i.e., (2.41), (2.44), (2.46), (2.48), and (2.55), we can see that each of them is related to only one element of the admittance matrix. Therefore, if the element Y_{ij} in the admittance matrix is zero, the corresponding element in the Jacobian matrix of the correction equation is also zero. It means the Jacobian matrix is a sparse matrix, and has the same structure as the admittance matrix.
3. From the expression of the elements of the Jacobian matrix we can see that the Jacobian matrix is not symmetrical in either coordinate form. For example,

$$\frac{\partial \Delta P_i}{\partial \theta_j} \neq \frac{\partial \Delta P_j}{\partial \theta_i}, \quad \frac{\partial \Delta Q_i}{\partial V_j} \neq \frac{\partial \Delta Q_j}{\partial V_i}$$

$$\frac{\partial \Delta P_i}{\partial e_j} \neq \frac{\partial \Delta P_j}{\partial e_i}, \quad \frac{\partial \Delta Q_i}{\partial f_j} \neq \frac{\partial \Delta Q_j}{\partial f_i}, \text{etc.}$$

4. The elements in the Jacobian matrix are a function of node voltage phasors. Therefore, they will vary with node voltages during the iterative process. The Jacobian matrix must not only be updated but also be triangularized in each iteration. This has a major effect on the calculation efficiency of the Newton–Raphson method.

Many improvements of the Newton–Raphson method have focused on this problem.

For instance, when the rectangular coordinate is adopted and the injected current (see (2.4)) is used to form the load flow equations [12], the off-diagonal elements of

the Jacobian matrix become constant. This property can certainly be used to improve the solution efficiency. Semlyen and de Leon [13] suggest that the Jacobian matrix elements can be updated partially to alleviate the computing burden.

Both the above two forms of coordinate system are widely used in Newton–Raphson load flow algorithms. When the polar form is used, PV nodes can be conveniently treated. When the rectangular form is used, the calculation of trigonometric functions is avoided. Generally speaking, the difference is not very significant. A comparison between the two coordinate systems is carried out in [14].

The fast decoupled method is derived from the Newton–Raphson method in polar form. It will be discussed in Sect. 2.4. In the next section, we mainly introduce the Newton–Raphson method based on the correction equation of (2.54) in rectangular form.

2.3.3 Solution Process of Newton Method

In the Newton–Raphson method, the electric network is described by its admittance matrix. From (2.52), (2.53), (2.55), and (2.56) we know that all operations are relative to the admittance matrix. Therefore, forming the admittance matrix is the first step in the algorithm.

The solving process of the Newton method roughly consists of the following steps.

1. Specify the initial guess values of node voltage, $e^{(0)}, f^{(0)}$;
2. Substituting $e^{(0)}$, $f^{(0)}$ into (2.52) and (2.53), obtain the left-hand term of the correction equation, $\Delta P^{(0)}$, $\Delta Q^{(0)}$, and $(\Delta V^2)^{(0)}$;
3. Substituting $e^{(0)}, f^{(0)}$ into (2.55) and (2.56), obtain the coefficient matrix (Jacobian matrix) of the correction equation;
4. Solving (2.54), obtain the correction variables, $\Delta e^{(0)}$ and $\Delta f^{(0)}$;
5. Modify voltages;

$$\left.\begin{aligned} e^{(1)} &= e^{(0)} - \Delta e^{(0)} \\ f^{(1)} &= f^{(0)} - \Delta f^{(0)} \end{aligned}\right\} \tag{2.57}$$

6. Substituting $e^{(1)}$ and $f^{(1)}$ into (2.52) and (2.53), obtain $\Delta P^{(1)}$, $\Delta Q^{(1)}$, and $(\Delta V^2)^{(1)}$;
7. Check whether the iteration has converged. When it has converged, calculate branch load flow and output the results; otherwise take $e^{(1)}$ and $f^{(1)}$ as the new guess value, return to step (3) and start the next iteration.

The main flowchart of the Newton–Raphson method is shown in Fig. 2.3. The above steps introduce the main principles of the solution process. There are still many details to be clarified. As mentioned above, the solution procedure of the

Newton–Raphson method is essentially the process of iteratively forming and solving the correction equations. Dealing with the correction equation has a crucial influence over the memory requirement and computing burden. This problem will be presented in the next section. First, we discuss some other important issues.

The convergence characteristic of the Newton–Raphson method is excellent. Generally, it can converge in 6–7 iterations, and the number of iteration does not depend on the scale of the power system. Theoretically speaking, the Newton–Raphson method has a quadratic convergence characteristic if the initial guess values are close to the solution. If the initial guess values are not good enough, the iterative process may not converge or may converge to a solution at which the power system cannot operate. This property stems from the Newton method itself. As described above, the substance of the Newton method is sequential linearization of nonlinear equations. It is established on the assumption that Δe and Δf are very small so that their high-order terms can be neglected. Therefore, a good initial guess value is crucial because the Newton method is very sensitive to it.

Under normal operation states of power systems, the node voltage magnitudes are usually close to their nominal voltages, and the phase angle differences between the nodes of a branch are not very large. Therefore, a "flat start" initial guess value, i.e.,

$$e_i^{(0)} = 1.0 \quad f_i^{(0)} = 0.0 \quad (i = 1, 2, \ldots, n) \tag{2.58}$$

can give satisfactory results. In Fig. 2.3, the convergence condition is

$$\left\| \Delta P^{(t)}, \Delta Q^{(t)} \right\| < \varepsilon \tag{2.59}$$

where $\left\| \Delta P^{(t)}, \Delta Q^{(t)} \right\|$ is a norm representing the maximal modulus elements in vectors $\Delta P^{(t)}, \Delta Q^{(t)}$. This convergence criterion is very intuitive, and can be used to directly control the power errors. When the calculation is based on the per unit system, we can set $\varepsilon = 10^{-4}$ or 10^{-3}. If the base value is 100 MVA, the maximum error corresponds to 0.01 MVA or 0.1 MVA.

From Fig. 2.3 we know that in the Newton–Raphson load flow calculation, the Jacobian matrix must be formed and triangularized in each iteration. Hence the computing burden in each iteration is quite heavy. From the expressions of Jacobian elements one can see that in the iteration procedure, especially when it is near convergence, the change of the elements caused by voltage variation is not significant (see Example 2.1). Therefore, to decrease the computing effort, once a Jacobian matrix is formed, it could be used in several successive iterations.

2.3.4 Solution of Correction Equations

The Newton–Raphson method, with Gauss elimination solving the correction equation, has been used in load flow calculation since the 1950s.

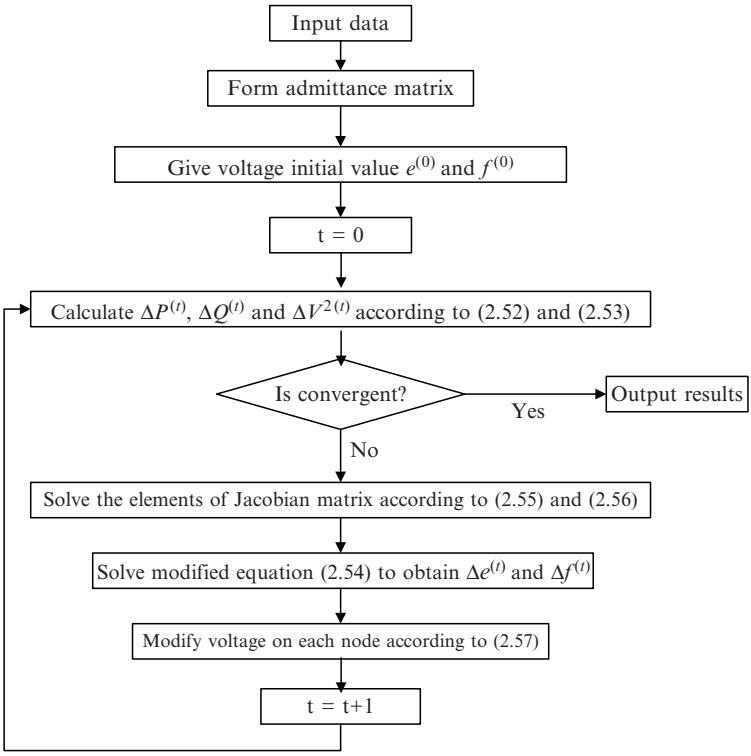

Fig. 2.3 Flowchart of Newton method

In the 1960s, the sparsity of the correction equation was fully investigated and employed in the iteration procedure. In this way, the storage and operation for zero elements in the Jacobian are avoided. When the technology of optimal node ordering is adopted, it can minimize the number of the fill-in nonzero elements in factorizing the Jacobian of the correction equation. This greatly reduces memory and computing requirements to almost proportional to the node number of the power system. Based on this sparsity technology, the Newton–Raphson method has become one of the most popular methods in power system load flow calculation [7].

With a simple system as shown in Fig. 2.4, we now illustrate some algorithmic tricks in solving the correction equation of the Newton–Raphson method. In Fig. 2.4, both node 3 and node 6 are generator nodes. We set node 3 as a *PV* node while node 6 the slack node; other nodes are all *PQ* nodes. The structure of the network admittance matrix is shown in Fig. 2.5.

The correction equation is given as (2.60). It does not include the equation related to node 6, the slack node.

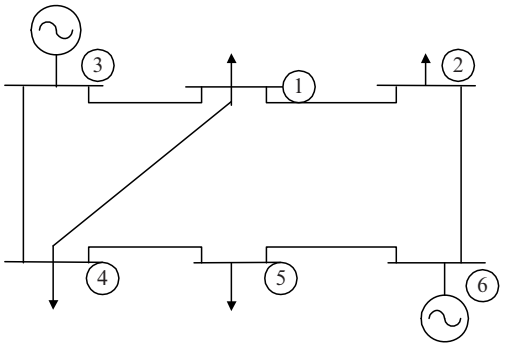

Fig. 2.4 Example of simple system

Fig. 2.5 Structure of admittance matrix

$$\begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} & & \\ Y_{21} & Y_{22} & & & & Y_{26} \\ Y_{31} & & Y_{33} & Y_{34} & & \\ Y_{41} & & Y_{43} & Y_{44} & Y_{45} & \\ & & & Y_{54} & Y_{55} & Y_{56} \\ & Y_{62} & & & Y_{65} & Y_{66} \end{bmatrix}$$

$$\begin{bmatrix} \Delta P_1 \\ \Delta Q_1 \\ \Delta P_2 \\ \Delta Q_2 \\ \Delta P_3 \\ \Delta V_3^2 \\ \Delta P_4 \\ \Delta Q_4 \\ \Delta P_5 \\ \Delta Q_5 \end{bmatrix} = \begin{bmatrix} H_{11} & N_{11} & H_{12} & N_{12} & H_{13} & N_{13} & H_{14} & N_{14} & & \\ J_{11} & L_{11} & J_{12} & L_{12} & J_{13} & L_{13} & J_{14} & L_{14} & & \\ H_{21} & N_{21} & H_{22} & N_{22} & & & & & & \\ J_{21} & L_{21} & J_{22} & L_{22} & & & & & & \\ H_{31} & N_{31} & & & H_{33} & N_{33} & H_{34} & N_{34} & & \\ 0 & 0 & & & R_{33} & S_{33} & 0 & 0 & & \\ H_{41} & N_{41} & & & H_{43} & N_{43} & H_{44} & N_{44} & H_{45} & N_{45} \\ J_{41} & L_{41} & & & J_{43} & L_{43} & J_{44} & L_{44} & J_{45} & L_{45} \\ & & & & & & H_{54} & N_{54} & H_{55} & N_{55} \\ & & & & & & J_{54} & L_{54} & J_{55} & L_{55} \end{bmatrix} \begin{bmatrix} \Delta e_1 \\ \Delta f_1 \\ \Delta e_2 \\ \Delta f_2 \\ \Delta e_3 \\ \Delta f_3 \\ \Delta e_4 \\ \Delta f_4 \\ \Delta e_5 \\ \Delta f_5 \end{bmatrix} \tag{2.60}$$

where the constant terms ΔP_i, ΔQ_i can be obtained by (2.52),

$$\left.\begin{aligned} \Delta P_i &= P_{is} - e_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) - f_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) \\ \Delta Q_i &= Q_{is} - f_i \sum_{j \in i} (G_{ij} e_j - B_{ij} f_j) + e_i \sum_{j \in i} (G_{ij} f_j + B_{ij} e_j) \end{aligned}\right\}$$

or can be written as

$$\left.\begin{aligned} \Delta P_i &= P_{is} - (e_i a_i + f_i b_i) \\ \Delta Q_i &= Q_{is} - (f_i a_i - e_i b_i) \end{aligned}\right\} \tag{2.61}$$

From (2.56) we know the diagonal elements of the Jacobian are

$$\left.\begin{aligned} H_{ii} &= \frac{\partial \Delta P_i}{\partial e_i} = -a_i - (G_{ii}e_i + B_{ii}f_i) \\ N_{ii} &= \frac{\partial \Delta P_i}{\partial f_i} = -b_i + (B_{ii}e_i - G_{ii}f_i) \\ J_{ii} &= \frac{\partial \Delta Q_i}{\partial e_i} = b_i + (B_{ii}e_i - G_{ii}f_i) \\ L_{ii} &= \frac{\partial \Delta Q_i}{\partial f_i} = -a_i + (G_{ii}e_i + B_{ii}f_i) \end{aligned}\right\} \tag{2.62}$$

Both (2.61) and (2.62) include components of the injected current at node i, a_i and b_i. To calculate ΔP_i, ΔQ_i, and the diagonal elements of Jacobian H_{ii}, N_{ii}, J_{ii}, L_{ii}, we must first compute a_i and b_i. From (2.11) we can see, the injected current components a_i and b_i at node i only depends on the i th row elements of the admittance matrix and voltage components of corresponding nodes. Therefore, a_i and b_i can be accumulated by sequentially taking the two terms and performing multiplication plus operation.

After a_i, b_i are known, ΔP_i and ΔQ_i can be easily obtained according to (2.61).

The nondiagonal elements of the Jacobian in (2.60) can be expressed by:

$$\left.\begin{aligned} H_{ij} &= \frac{\partial \Delta P_i}{\partial e_j} = -(G_{ij}e_i + B_{ij}f_i) \\ N_{ij} &= \frac{\partial \Delta P_i}{\partial f_j} = B_{ij}e_i - G_{ij}f_i \\ J_{ij} &= \frac{\partial \Delta Q_i}{\partial e_j} = B_{ij}e_i - G_{ij}f_i = N_{ij} \\ L_{ij} &= \frac{\partial \Delta Q_i}{\partial f_j} = G_{ij}e_i + B_{ij}f_i = -H_{ij} \end{aligned}\right\} \tag{2.63}$$

Obviously, the off-diagonal elements are only related to the corresponding admittance elements and voltage components. From (2.62), the ith diagonal element consists of, besides the injecting current components at node i(a_i and b_i), only the arithmetic operation results of the diagonal elements of admittance matrix $G_{ii} + jB_{ii}$ and voltage components $e_i + jf_i$.

In brief, the whole correction equation can be formed by sequentially taking and arithmetically operating the elements of the admittance matrix and corresponding voltage components.

If node i is PV node, the equation of ΔQ_i should be replaced by the equation of ΔV_i^2. The constant term ΔV_i^2 on the left hand and elements R_{ii} and S_{ii} of the Jacobian can be easily obtained from (2.53) and (2.56),

$$\left.\begin{aligned} R_{ii} &= \frac{\partial \Delta V_i^2}{\partial e_i} = -2e_i \\ S_{ii} &= \frac{\partial \Delta V_i^2}{\partial f_i} = -2f_i \end{aligned}\right\} \tag{2.64}$$

Forming the correction equation is a very important step in the Newton–Raphson method which remarkably affects the efficiency of the whole algorithm. Therefore, we should investigate the above equations carefully in coding the program.

When Gauss elimination is used to solve the correction equation, we usually eliminate the correction equation row by row. The augmented matrix corresponding to (2.60) is

$$\begin{bmatrix}
H_{11} & N_{11} & H_{12} & N_{12} & H_{13} & N_{13} & H_{14} & N_{14} & & & \Delta P_1 \\
J_{11} & L_{11} & J_{12} & L_{12} & J_{13} & L_{13} & J_{14} & L_{14} & & & \Delta Q_1 \\
H_{21} & N_{21} & H_{22} & N_{22} & & & & & & & \Delta P_2 \\
J_{21} & L_{21} & J_{22} & L_{22} & & & & & & & \Delta Q_2 \\
H_{31} & N_{31} & & & H_{33} & N_{33} & H_{34} & N_{34} & & & \Delta P_3 \\
0 & 0 & & & R_{33} & S_{33} & 0 & 0 & & & \Delta V_3^2 \\
H_{41} & N_{41} & & & H_{43} & N_{43} & H_{44} & N_{44} & H_{45} & N_{45} & \Delta P_4 \\
J_{41} & L_{41} & & & J_{43} & L_{43} & J_{44} & L_{44} & J_{45} & L_{45} & \Delta Q_4 \\
 & & & & & & H_{54} & N_{54} & H_{55} & N_{55} & \Delta P_5 \\
 & & & & & & J_{54} & L_{54} & J_{55} & L_{55} & \Delta Q_5
\end{bmatrix}$$

After the equations related to node 1 and 2 are eliminated, the augmented matrix is converted as shown in Fig. 2.6. This figure tell us when the equations related to node 2 are eliminated (row 3 and row 4), all operations are independent of equations related to node 3, 4, …, N. Therefore, in the eliminating procedure, we can eliminate the rows related to a node immediately after forming them.

In Fig. 2.6, elements such as $H''_{23}, N''_{23}, \ldots, L''_{24}$, etc. are fill-in nonzero elements created in the elimination process. To decrease the number of injected elements, we should optimize the node number ordering before load flow calculation (see Section 1.3.5). The element with superscript ($''$) represents that it has been manipulated. We need not save memory for the fill-in element in advance using this elimination procedure and thus the algorithm is simplified.

When the whole elimination procedure finished, the augmented matrix of correction equation becomes,

$$\left[\begin{array}{cccccccccc|c}
1 & N'_{11} & H'_{12} & N'_{12} & H'_{13} & N'_{13} & H'_{14} & N'_{14} & & & \Delta P'_1 \\
 & 1 & J'_{12} & L'_{12} & J'_{13} & L'_{13} & J'_{14} & L'_{14} & & & \Delta Q'_1 \\
 & & 1 & N'_{22} & H''_{23} & N''_{23} & H''_{24} & N''_{24} & & & \Delta P'_2 \\
 & & & 1 & J''_{23} & L''_{23} & J''_{24} & L''_{24} & & & \Delta Q'_2 \\
\hline
H_{31} & N_{31} & & & H_{33} & N_{33} & H_{34} & N_{34} & & & \Delta P_3 \\
 & & & & R_{33} & S_{33} & & & & & \Delta V_3^2 \\
H_{41} & N_{41} & & & H_{43} & N_{43} & H_{44} & N_{44} & H_{45} & N_{45} & \Delta P_4 \\
J_{41} & L_{41} & & & J_{43} & L_{43} & J_{44} & L_{44} & J_{45} & L_{45} & \Delta Q_4 \\
 & & & & & & H_{54} & N_{54} & H_{55} & N_{55} & \Delta P_5 \\
 & & & & & & J_{54} & L_{54} & J_{55} & L_{55} & \Delta Q_5
\end{array}\right]$$

Fig. 2.6 Diagram of eliminating row by row

$$\begin{bmatrix}
1 & N'_{11} & H'_{12} & N'_{12} & H'_{13} & N'_{13} & H'_{14} & N'_{14} & & & \Delta P'_1 \\
 & 1 & J'_{12} & L'_{12} & J'_{13} & L'_{13} & J'_{14} & L'_{14} & & & \Delta Q'_1 \\
 & & 1 & N'_{22} & H''_{23} & N''_{23} & H''_{24} & N''_{24} & & & \Delta P'_2 \\
 & & & 1 & J''_{23} & L''_{23} & J''_{24} & L''_{24} & & & \Delta Q'_2 \\
 & & & & 1 & N'_{33} & H'_{34} & N'_{34} & & & \Delta P'_3 \\
 & & & & & 1 & J''_{34} & L''_{34} & & & \Delta V_3^2 \\
 & & & & & & 1 & N'_{44} & H'_{45} & N'_{45} & \Delta P'_4 \\
 & & & & & & & 1 & J'_{45} & L'_{45} & \Delta Q'_4 \\
 & & & & & & & & 1 & N'_{55} & \Delta P'_5 \\
 & & & & & & & & & 1 & \Delta Q'_5
\end{bmatrix}$$

Finally, using a normal backward substitution, one can get $\Delta e_1, \Delta f_1, \ldots, \Delta e_5, \Delta f_5$ from $\Delta P'_1, \Delta Q'_1, \ldots, \Delta Q'_5$.

Following to the above discussion, we can summarize the algorithm via flowchart shown in Fig. 2.7, where R represents the slack node. The correction equation can be solved by the common Gauss elimination method. The above procedure adopts the strategy of eliminating the rows related to a node immediately after forming them. At the same time, the corresponding constant terms of the correction equation are also accumulated and eliminated. Thus the operation count per iteration is significantly reduced.

[Example 2.1] Calculate the load flow of the power system shown in Fig. 2.8.

[Solution] The load flow is calculated according to the procedures of the flowchart. The first step includes forming the admittance matrix and specifying the initial voltage values.

From Example 1.1 we know the admittance matrix of this system is

$$\mathbf{Y} = \begin{bmatrix}
1.37874 & -0.62402 & -0.75471 & & \\
-j6.29166 & +j3.90015 & +j2.64150 & & \\
-0.62402 & 1.45390 & -0.82987 & 0.00000 & \\
+j3.90015 & -j66.98082 & +j3.11203 & +j63.49206 & \\
-0.75471 & -0.82987 & 1.58459 & & 0.00000 \\
+j2.64150 & +j3.11203 & -j35.73786 & & +j31.74603 \\
 & 0.00000 & & 0.00000 & \\
 & +j63.49206 & & -j66.66667 & \\
 & & 0.00000 & & 0.00000 \\
 & & +j31.74603 & & -j33.33333
\end{bmatrix}$$

The initial values of node voltages are given in Table 2.1.

According to (2.52) and (2.53), we can establish the expression of the constant terms (mismatch terms) of the correction equations as

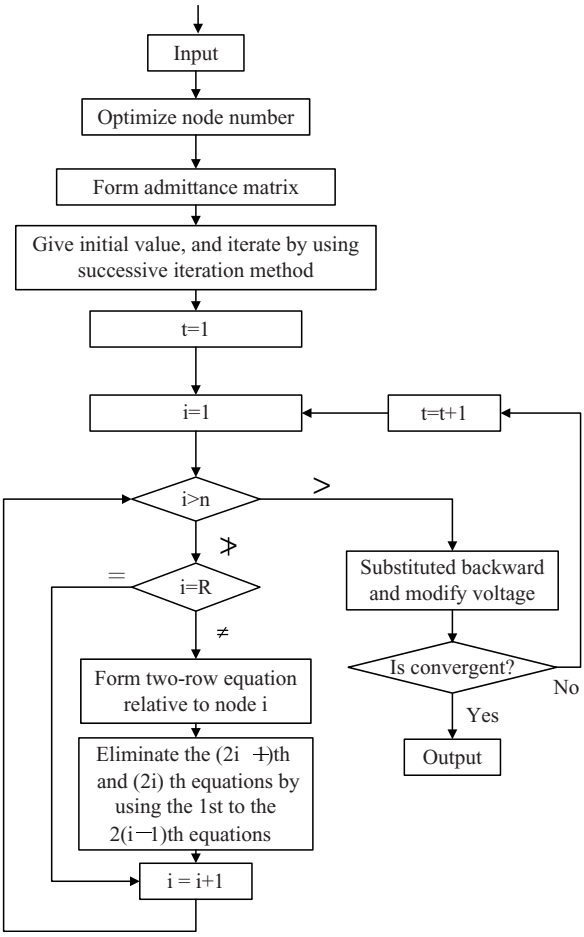

Fig. 2.7 Flowchart of Newton Method

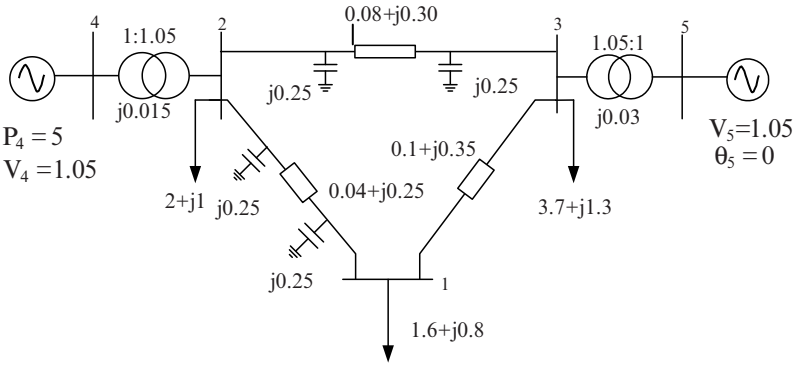

Fig. 2.8 Simple power system

Table 2.1 Voltage initial values

Node	1	2	3	4	5
$e^{(0)}$	1.00000	1.00000	1.00000	1.05000	1.05000
$f^{(0)}$	0.00000	0.00000	0.00000	0.00000	0.00000

$$\Delta P_1 = P_{1s} - e_1[(G_{11}e_1 - B_{11}f_1) + (G_{12}e_2 - B_{12}f_2) + (G_{13}e_3 - B_{13}f_3)] \\ -f_1[(G_{11}f_1 + B_{11}e_1) + (G_{12}f_2 + B_{12}e_2) + (G_{13}f_3 + B_{13}e_3)]$$

$$\Delta Q_1 = Q_{1s} - f_1[(G_{11}e_1 - B_{11}f_1) + (G_{12}e_2 - B_{12}f_2) + (G_{13}e_3 - B_{13}f_3)] + \\ e_1[(G_{11}f_1 + B_{11}e_1) + (G_{12}f_2 + B_{12}e_2) + (G_{13}f_3 + B_{13}e_3)]$$

$$\Delta P_4 = P_{4s} - e_4[(G_{42}e_2 - B_{42}f_2) + (G_{44}e_4 - B_{44}f_4)] - f_4[(G_{42}f_2 + B_{42}e_2) + \\ (G_{44}f_4 + B_{44}e_4)]$$

$$\Delta V_4^2 = V_{4s}^2 - (e_4^2 + f_4^2)$$

Using (2.55) and (2.56), we can obtain the expressions of Jacobian matrix elements:

$$\frac{\partial \Delta P_1}{\partial e_1} = -[(G_{11}e_1 - B_{11}f_1) + (G_{12}e_2 - B_{12}f_2) + (G_{13}e_3 - B_{13}f_3)] - G_{11}e_1 - B_{11}f_1$$

$$\frac{\partial \Delta P_1}{\partial f_1} = -[(G_{11}f_1 + B_{11}e_1) + (G_{12}f_2 + B_{12}e_2) + (G_{13}f_3 + B_{13}e_3)] + B_{11}e_1 - G_{11}f_1$$

$$\frac{\partial \Delta P_1}{\partial e_2} = -(G_{12}e_1 + B_{12}f_1), \quad \frac{\partial \Delta P_1}{\partial f_2} = B_{12}e_1 - G_{12}f_1$$

$$\frac{\partial \Delta P_1}{\partial e_3} = -(G_{13}e_1 + B_{13}f_1), \quad \frac{\partial \Delta P_1}{\partial f_3} = B_{13}e_1 - G_{13}f_1$$

$$\frac{\partial \Delta Q_1}{\partial e_1} = [(G_{11}f_1 + B_{11}e_1) + (G_{12}f_2 + B_{12}e_2) + (G_{13}f_3 + B_{13}e_3)] + B_{11}e_1 - G_{11}f_1$$

$$\frac{\partial \Delta Q_1}{\partial f_1} = -[(G_{11}e_1 - B_{11}f_1) + (G_{12}e_2 - B_{12}f_2) + (G_{13}e_3 - B_{13}f_3)] + G_{11}e_1 + B_{11}f_1$$

$$\frac{\partial \Delta Q_1}{\partial e_2} = \frac{\partial \Delta P_1}{\partial f_2}, \quad \frac{\partial \Delta Q_1}{\partial f_2} = -\frac{\partial \Delta P_1}{\partial e_2}$$

$$\frac{\partial \Delta Q_1}{\partial e_3} = \frac{\partial \Delta P_1}{\partial f_3}, \quad \frac{\partial \Delta Q_1}{\partial f_3} = -\frac{\partial \Delta P_1}{\partial e_3}$$

$$\frac{\partial \Delta P_4}{\partial e_4} = -[(G_{42}e_2 - B_{42}f_2) + (G_{44}e_4 - B_{44}f_4)] - G_{44}e_4 - B_{44}f_4$$

$$\frac{\partial \Delta P_4}{\partial f_4} = -[(G_{42}f_2 + B_{42}e_2) + (G_{44}f_4 + B_{44}e_4)] + B_{44}e_4 - G_{44}f_4$$

$$\frac{\partial \Delta V_4^2}{\partial e_4} = -2e_4$$
$$\frac{\partial \Delta V_4^2}{\partial f_4} = -2f_4$$

Thus according to (2.60), the correction equation of the first iteration can be written as

$$\begin{bmatrix} -1.37874 & -6.54166 & 0.62402 & 3.90015 & 0.75471 & 2.64150 & & \\ -6.04166 & 1.37874 & 3.90015 & -0.62402 & 2.64150 & -0.75471 & & \\ 0.62402 & 3.90015 & -1.45390 & -73.67881 & 0.82897 & 3.11203 & 0.00000 & 63.49206 \\ 3.90015 & 0.62402 & -60.28283 & 1.45390 & 3.11203 & -0.82897 & 63.49206 & 0.00000 \\ -0.75471 & 2.64150 & 0.82897 & 3.11203 & -1.58459 & -39.98688 & & \\ 2.64150 & -0.75471 & 3.11203 & -0.82897 & -32.38884 & 1.58459 & & \\ & & 0.00000 & 66.66666 & & & 0.00000 & -63.49206 \\ & & 0.00000 & 0.00000 & & & -2.10000 & 0.00000 \end{bmatrix} \begin{bmatrix} \Delta e_1 \\ \Delta f_1 \\ \Delta e_2 \\ \Delta f_2 \\ \Delta e_3 \\ \Delta f_3 \\ \Delta e_4 \\ \Delta f_4 \end{bmatrix}$$
$$= \begin{bmatrix} -1.60000 \\ -0.55000 \\ -2.00000 \\ 5.69799 \\ -3.70000 \\ 2.04901 \\ 5.00000 \\ 0.00000 \end{bmatrix}$$

the above equation, the elements in italic have maximal absolute value in each row of the Jacobian matrix. Obviously, if elements are arranged this way, the maximal elements do not appear at the diagonal positions.

It should be noted that this situation is not accidental. From the above equation we can conclude that the maximal element of each row is $\frac{\partial \Delta P_i}{\partial f_i}$ or $\frac{\partial \Delta Q_i}{\partial e_i}$. This is because the active power is mainly related to the vertical component of voltage while the reactive power is mainly related to the horizontal component of voltage in high voltage power systems.

To reduce the rounding error of the calculations, the maximal elements should be located in diagonal positions. There are two methods to satisfy this requirement: the first is to exchange positions of the equations relative to ΔQ and ΔP, i.e., to exchange odd numbered rows with even numbered rows; the second method is to exchange the variables Δe and Δf, i.e., to exchange odd numbered columns with even numbered columns of the Jacobian matrix.

We now introduce the first approach. Thus the above equation will be rearranged as,

$$\begin{bmatrix} \boldsymbol{-6.04166} & 1.37874 & 3.90015 & -0.62402 & 2.64150 & -0.75471 & & \\ -1.37874 & \boldsymbol{-6.54166} & 0.62402 & 3.90015 & 0.75471 & 2.64150 & & \\ 3.90015 & 0.62402 & \boldsymbol{-60.28283} & 1.45390 & 3.11203 & -0.82897 & 63.49206 & 0.00000 \\ 0.62402 & 3.90015 & -1.45390 & \boldsymbol{-73.67881} & 0.82897 & 3.11203 & 0.00000 & 63.49206 \\ 2.64150 & -0.75471 & 3.11203 & -0.82897 & \boldsymbol{-32.38884} & 1.58459 & & \\ -0.75471 & 2.64150 & 0.82897 & 3.11203 & -1.58459 & \boldsymbol{-39.98688} & & \\ & & & & & & \boldsymbol{-2.10000} & 0.00000 \\ & & 0.00000 & 66.66666 & & & 0.00000 & -63.49206 \end{bmatrix} \begin{bmatrix} \Delta e_1 \\ \Delta f_1 \\ \Delta e_2 \\ \Delta f_2 \\ \Delta e_3 \\ \Delta f_3 \\ \Delta e_4 \\ \Delta f_4 \end{bmatrix}$$

$$= \begin{bmatrix} -0.55000 \\ -1.60000 \\ 5.69799 \\ -2.00000 \\ 2.04901 \\ -3.70000 \\ 0.00000 \\ 5.00000 \end{bmatrix}$$

We can see the maximal element of each row appears in the diagonal position except for row 8.

As described in Section 2.3.4, the iteration procedure adopts the strategy of immediately eliminating the rows related to a node after forming them (see Fig. 2.7). The equations related to node 1 are formed as

$$\left[\begin{array}{cccccccc:c} -6.04166 & 1.37874 & 3.90015 & -0.62402 & 2.64150 & -0.75471 & 0 & 0 & -0.55000 \\ -1.37874 & -6.54166 & 0.62402 & 3.90015 & 0.75471 & 2.64150 & 0 & 0 & -1.60000 \end{array}\right]$$

After the elimination operation is executed, the first and second row of the upper triangular matrix can be obtained:

$$\left[\begin{array}{cccccccc:c} 1.00000 & -0.22820 & -0.64554 & 0.10328 & -0.43721 & 0.12491 & 0 & 0 & 0.09103 \\ & 1.00000 & 0.03879 & -0.58961 & -0.02215 & -0.41038 & 0 & 0 & 0.21505 \end{array}\right]$$

Then we establish the equations related to node 2, the corresponding augmented matrix is

$$\left[\begin{array}{cccccccc:c} 3.90015 & -0.62402 & -60.28283 & 1.45390 & 3.11203 & -0.82987 & 63.49206 & 0.0 & 5.69799 \\ 0.62402 & 3.90015 & -1.45390 & -73.67881 & 0.82987 & 3.11203 & 0.0 & 63.49206 & -2.0 \end{array}\right]$$

Executing the elimination operation, the third and forth rows of the upper triangular matrix become:

$$\left[\begin{array}{cccccc:c} 1.00000 & -0.02090 & -0.08348 & 0.02090 & -1.09894 & 0.00000 & -0.09184 \\ & 1.00000 & -0.01528 & -0.06609 & 0.01859 & -0.88943 & 0.04253 \end{array}\right]$$

Continuing this procedure until the eliminating operation procedure is finished, we have the upper triangular matrix:

$$\left[\begin{array}{cccccccc:c} 1.00000 & -0.22820 & -0.64554 & 0.10328 & -0.43721 & 0.12491 & & & 0.09103 \\ & 1.00000 & 0.03879 & -0.58961 & -0.02215 & -0.41038 & & & 0.21505 \\ & & 1.00000 & -0.02090 & -0.08348 & 0.02090 & -1.09894 & 0.00000 & -0.09148 \\ & & & 1.00000 & -0.01528 & -0.06609 & 0.01850 & -0.88943 & 0.04253 \\ & & & & 1.00000 & -0.03303 & -0.17246 & 0.03146 & -0.07548 \\ & & & & & 1.00000 & -0.02816 & -0.11194 & 0.12021 \\ & & & & & & 1.00000 & 0.00000 & 0.00000 \\ & & & & & & & 1.00000 & -0.45748 \end{array}\right]$$

After the backward substitution operation, the correcting increments of node voltages can be obtained,

$$\begin{bmatrix} \Delta e_1 \\ \Delta f_1 \\ \Delta e_2 \\ \Delta f_2 \\ \Delta e_3 \\ \Delta f_3 \\ \Delta e_4 \\ \Delta f_4 \end{bmatrix} = \begin{bmatrix} 0.03356 \\ 0.03348 \\ -0.10538 \\ -0.36070 \\ -0.05881 \\ 0.06900 \\ 0.00000 \\ -0.45748 \end{bmatrix}$$

Modifying the node voltage, the voltage vector becomes:

$$\begin{bmatrix} e_1 \\ f_1 \\ e_2 \\ f_2 \\ e_3 \\ f_3 \\ e_4 \\ f_4 \end{bmatrix} = \begin{bmatrix} 0.96643 \\ -0.33481 \\ 1.10533 \\ 0.36070 \\ 1.05881 \\ -0.66900 \\ 1.05000 \\ 0.45748 \end{bmatrix}$$

Using this voltage vector as the initial voltage value, we can repeat above operations. If the tolerance is set to $\varepsilon = 10^{-6}$, the calculation converges after five iterations. The evolution process of node voltages and power mismatches is shown in Tables 2.2 and 2.3.

Table 2.2 Node voltages in iterative process

Iterating No.	e_1	f_1	e_2	f_2	e_3	f_3	e_4	f_4
1	0.96643	−0.33481	1.10538	0.36074	1.05881	−0.06900	1.05000	0.45748
2	0.87365	−0.07006	1.03350	0.32886	1.03564	−0.07694	0.97694	0.38919
3	0.85947	−0.07176	1.02608	0.33047	1.03355	−0.07737	0.97464	0.39061
4	0.85915	−0.07182	1.02600	0.33047	1.03351	−0.07738	0.97461	0.39067
5	0.85915	−0.07182	1.02600	0.33047	1.03351	−0.07738	0.97461	0.39067

Table 2.3 Node power mismatches in iterative process

Iterating No.	ΔQ_1	ΔP_1	ΔQ_2	ΔP_2	ΔQ_3	ΔP_3	ΔP_4
1	−0.55000	−1.60000	5.69799#	−2.00000	2.04901	−3.70000	5.00000
2	−0.07263	−0.03473	−6.00881#	2.10426	−0.37144	0.04904	−2.39001
3	−0.02569	−0.06011	−0.41159#	0.15764	−0.00924	0.00329	−0.16193
4	−0.00078	−0.00032	−0.0030#	−0.00054	−0.00002	0.00000	0.00069
5	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

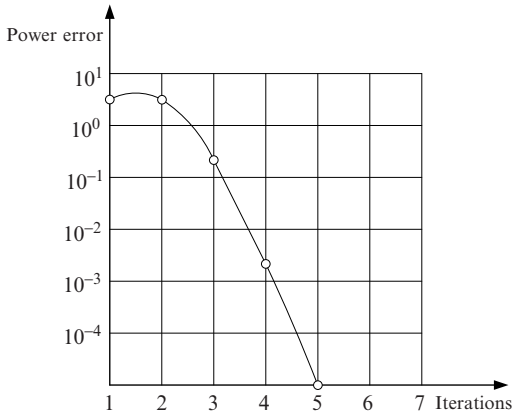

Fig. 2.9 Convergence property of Newton–Raphson method

To reveal the convergence property, the maximal power mismatches (with # in Table 2.3) in the iterative process are shown in Fig. 2.9.

In the iteration process, especially when it approaches convergence, the changes of the diagonal elements in the Jacobian are not very significant. To illustrate this point, the changes of the diagonal elements are given in Table 2.4.

The calculation results of node voltages are shown in Table 2.5.

Table 2.4 Diagonal elements of Jacobian matrix in iterative process

Iterating no.	$\frac{\partial \Delta Q_1}{\partial e_1}$	$\frac{\partial \Delta P_1}{\partial f_1}$	$\frac{\partial \Delta Q_2}{\partial e_2}$	$\frac{\partial \Delta P_2}{\partial f_2}$	$\frac{\partial \Delta Q_3}{\partial e_3}$	$\frac{\partial \Delta P_3}{\partial f_3}$	$\frac{\partial \Delta V_4^2}{\partial e_4}$	$\frac{\partial \Delta P_4}{\partial f_4}$
1	6.04166	6.54166	60.28283	73.67881	32.38884	39.08688	1.05000	63.49206
2	5.22590	6.84268	79.81886	69.30868	36.62734	38.83341	0.96259	70.18293
3	4.37415	6.42613	69.78933	69.61682	35.38612	38.39351	0.97528	65.61929
4	4.23077	6.38634	68.89682	69.52026	35.29706	38.33158	0.97463	65.14834
5	4.22720	6.38577	68.88900	69.51747	35.29572	38.33048	0.97461	65.14332

Table 2.5 Node voltage vectors

Node	Magnitude	Angle (°)
1	0.86215	−4.77851
2	1.07791	17.85353
3	1.03641	−4.28193
4	1.05000	21.84332
5	1.05000	0.00000

2.4 Fast Decoupled Method

2.4.1 Introduction to Fast Decoupled Method

The basic idea of the fast decoupled method is expressing the nodal power as a function of voltages in polar form; separately solving the active and reactive power equations [9] by using active power mismatch to modify voltage angle and using reactive power mismatch to modify voltage magnitude. In this way, the computing burden of load flow calculation is alleviated significantly. In the following, the derivation of the fast decoupled method from the Newton method is discussed.

As described previously, the core of the Newton load flow approach is to solve the correction equation. When the nodal power equation is expressed in polar form, the correction equation is (see (2.50)),

$$\begin{bmatrix} \Delta \mathbf{P} \\ \Delta \mathbf{Q} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & \mathbf{N} \\ \mathbf{J} & \mathbf{L} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{\theta} \\ \Delta \mathbf{V}/\mathbf{V} \end{bmatrix} \tag{2.65}$$

or can be written as,

$$\begin{aligned} \Delta \mathbf{P} &= \mathbf{H}\Delta \boldsymbol{\theta} + \mathbf{N}\Delta \mathbf{V}/\mathbf{V} \\ \Delta \mathbf{Q} &= \mathbf{J}\Delta \boldsymbol{\theta} + \mathbf{L}\Delta \mathbf{V}/\mathbf{V} \end{aligned} \tag{2.66}$$

This equation is derived strictly from the mathematical viewpoint. It does not take the characteristics of power systems into consideration.

We know that in high voltage power system the active power flow is mainly related to the angle of the nodal voltage phasor while reactive power flow is mainly

related to its magnitude. The experiences of many load flow calculations tell us that the element values of matrix **N** and **J** in (2.66) are usually relatively small. Therefore, the first step to simplify the Newton method is to neglect **N** and **J**, and (2.66) is simplified to

$$\left.\begin{aligned}\Delta\mathbf{P} &= \mathbf{H}\Delta\boldsymbol{\theta}\\ \Delta\mathbf{Q} &= \mathbf{L}\Delta\mathbf{V}/\mathbf{V}\end{aligned}\right\} \tag{2.67}$$

Thus a simultaneous linear equation of dimension $2n$ is simplified to two simultaneous linear equations of dimension n.

The second important step to simplify the Newton method is to approximate the coefficient matrices of (2.67) as constant and symmetric matrices.

As the phase angle difference across a transmission line usually is not very large (does not exceed 10°∼20°), so the following relations hold,

$$\left.\begin{aligned}&\cos\theta_{ij} \approx 1\\ &G_{ij}\sin\theta_{ij} \ll B_{ij}\end{aligned}\right\} \tag{2.68}$$

Furthermore, the admittance B_{Li} corresponding to the node reactive power is certainly far smaller than the imaginary part of the node self-admittance, i.e.,

$$B_{Li} = \frac{Q_i}{V_i^2} \ll B_{ii}$$

Accordingly,

$$Q_i \ll V_i^2 B_{ii} \tag{2.69}$$

Based on the above relationships, the element expressions of coefficient matrix in (2.67) can be represented as (see (2.41), (2.42), (2.48), and (2.49)):

$$\left.\begin{aligned}H_{ii} &= V_i^2 B_{ii}\\ H_{ij} &= V_i V_j B_{ij}\\ L_{ii} &= V_i^2 B_{ii}\\ L_{ij} &= V_i V_j B_{ij}\end{aligned}\right\} \tag{2.70}$$

Therefore, the coefficient matrix in (2.67) can be written as

$$\mathbf{H} = \mathbf{L} = \begin{bmatrix} V_1^2 B_{11} & V_1 V_2 B_{12} & \cdots & V_1 V_n B_{1n}\\ V_2 V_1 B_{21} & V_2^2 B_{22} & \cdots & V_2 V_n B_{2n}\\ & \vdots & & \\ V_n V_1 B_{n1} & V_n V_2 B_{n2} & \cdots & V_n^2 B_{nn}\end{bmatrix} \tag{2.71}$$

It can be further represented as the product of the following matrices:

$$\mathbf{H} = \mathbf{L} = \begin{bmatrix} V_1 & & & \\ & V_2 & 0 & \\ & 0 & \ddots & \\ & & & V_n \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ & \vdots & \vdots & \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} \begin{bmatrix} V_1 & & & \\ & V_2 & 0 & \\ & 0 & \ddots & \\ & & & V_n \end{bmatrix} \tag{2.72}$$

Substituting (2.72) into (2.67), we can rewrite the correction equations as follows:

$$\begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \vdots \\ \Delta P_n \end{bmatrix} = \begin{bmatrix} V_1 & & & \\ & V_2 & 0 & \\ & 0 & \ddots & \\ & & & V_n \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ & \vdots & \vdots & \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} \begin{bmatrix} V_1\Delta\theta_1 \\ V_2\Delta\theta_2 \\ \vdots \\ V_n\Delta\theta_n \end{bmatrix} \tag{2.73}$$

and

$$\begin{bmatrix} \Delta Q_1 \\ \Delta Q_2 \\ \vdots \\ \Delta Q_n \end{bmatrix} = \begin{bmatrix} V_1 & & & \\ & V_2 & 0 & \\ & 0 & \ddots & \\ & & & V_n \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ & \vdots & \vdots & \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} \begin{bmatrix} \Delta V_1 \\ \Delta V_2 \\ \vdots \\ \Delta V_n \end{bmatrix} \tag{2.74}$$

Multiplying both sides of the above equation with matrix,

$$\begin{bmatrix} V_1 & & & \\ & V_2 & & \\ & & \ddots & \\ & & & V_n \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{V_1} & & & \\ & \frac{1}{V_2} & & \\ & & \ddots & \\ & & & \frac{1}{V_n} \end{bmatrix}$$

one can obtain

$$\begin{bmatrix} \Delta P_1/V_1 \\ \Delta P_2/V_2 \\ \vdots \\ \Delta P_n/V_n \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ & \vdots & \vdots & \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} \begin{bmatrix} V_1\Delta\theta_1 \\ V_2\Delta\theta_2 \\ \vdots \\ V_n\Delta\theta_n \end{bmatrix} \tag{2.75}$$

and

$$\begin{bmatrix} \Delta Q_1/V_1 \\ \Delta Q_2/V_2 \\ \vdots \\ \Delta Q_n/V_n \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ & \vdots & \vdots & \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} \begin{bmatrix} \Delta V_1 \\ \Delta V_2 \\ \vdots \\ \Delta V_n \end{bmatrix} \tag{2.76}$$

The above two equations are the correction equations of the fast decoupled load flow method. The coefficient matrix is merely the imaginary part of the nodal admittance matrix of the system, and is thus a symmetric, constant matrix. Combining with the power mismatch equation (2.13), we obtain the basic equations of the fast decoupled load flow model

$$\Delta P_i = P_{is} - V_i \sum_{j \in i} V_j (G_{ij} \cos \theta_{ij} + B_{ij} \sin \theta_{ij}) \quad (i = 1, 2, \ldots, n) \tag{2.77}$$

$$\Delta Q_i = Q_{is} - V_i \sum_{j \in i} V_j (G_{ij} \sin \theta_{ij} - B_{ij} \cos \theta_{ij}) \quad (i = 1, 2, \ldots, n) \tag{2.78}$$

The iterative process can be briefly summarized in the following steps:

1. Specify node voltage vector initial value $\theta_i^{(0)}$, $V_i^{(0)}$
2. Calculate the node active power mismatch ΔP_i according to (2.77), and then calculate $\Delta P_i / V_i$
3. Solving correction equation (2.75), calculate the node voltage angle correction $\Delta \theta_i$
4. Modify the node voltage angle θ_i

$$\theta_i^{(t)} = \theta_i^{(t-1)} \Delta \theta_i^{(t-1)} \tag{2.79}$$

5. Calculate node reactive power mismatch ΔQ_i according to (2.78), and then calculate $\Delta Q_i / V_i$
6. Solving correction equation (2.76), calculate the node voltage magnitude correction ΔV_i,
7. Modify the node voltage magnitude V_i;

$$V_i^{(t)} = V_i^{(t-1)} - \Delta V_i^{(t-1)} \tag{2.80}$$

8. Back to step (2) to continue the iterative process, until all node power mismatches ΔP_i and ΔQ_{i} satisfy convergence conditions.

2.4.2 Correction Equations of Fast Decoupled Method

The main difference between the fast decoupled method and the Newton method stems from their correction equations. Comparing with correction (2.40) or (2.54) of the Newton method, the two correction equations of the fast decoupled method have the following features:

1. Equations (2.75) and (2.76) are two simultaneous linear equations of dimension n instead of a simultaneous linear equation of dimension $2n$

2. In (2.75) and (2.76), all elements of the coefficient matrix remain constant during the iterative process
3. In (2.75) and (2.76), the coefficient matrix is symmetric.

The benefit of the first feature for computing speed and storage is obvious. The second feature alleviates the computing burden in forming and eliminating the Jacobian within the iterative process. We can first form the factor table for the coefficient matrix of the correction equation (see (2.76)) by triangularization. Then we can carry out elimination and backward substitution operations for different constant terms $\Delta P/V$ and $\Delta Q/V$ through repeatedly using the factor table. In this way, the correction equation can be solved very quickly. The third feature can further improve efficiency in forming and storing the factor table.

All the simplifications adopted by the fast decoupled method only affect the structure of the correction equation. In other words, they only affect the iteration process, but do not affect the final results. The fast decoupled method and the Newton method use the same mathematical model of (2.13), if adopting the same convergence criteria we should expect the same accuracy of results.

It seems that (2.75) and (2.76) derived above have the same coefficient matrix, but in practice the coefficient matrixes of the two correction equations in the fast decoupled algorithms are different. We can simply write them as

$$\Delta \mathbf{P}/\mathbf{V} = \mathbf{B}'\mathbf{V}\Delta\boldsymbol{\theta} \tag{2.81}$$

$$\Delta \mathbf{Q}/\mathbf{V} = \mathbf{B}''\Delta\mathbf{V} \tag{2.82}$$

Here $\mathbf{V}$ is a diagonal matrix with the diagonal elements being the node voltage magnitudes.

First, we should point out that the dimensions of $\mathbf{B}'$ and $\mathbf{B}''$ are different. The dimension of $\mathbf{B}'$ is $n-1$ while the dimension of $\mathbf{B}''$ is lower than $n-1$. This is because (2.82) dose not include the equations related to PV nodes. Hence if the system has r PV nodes, then the dimension of $\mathbf{B}''$ should be $n-r-1$.

To improve the convergence, we use different methods to treat $\mathbf{B}'$ and $\mathbf{B}''$, and how we treat $\mathbf{B}'$ and $\mathbf{B}''$ will result in different fast decoupled methods, are not merely the imaginary part of the admittance matrix.

As described above, (2.81) and (2.82) are the correction equations based on a series of simplifications. Equation (2.81) modifies the voltage phase angles according to the active power mismatch; (2.82) modifies the voltage magnitudes according to the reactive power mismatch. To speed up convergence, the factors that have no or less effect on the voltage angle should be removed from $\mathbf{B}'$. Therefore, we use the imaginary part of admittance to form $\mathbf{B}'$ without considering the effects of shunt capacitor and transformer's off-nominal taps. To be specific, the off-diagonal and diagonal elements of $\mathbf{B}'$ can be calculated according to following equations:

$$B'_{ij} = -\frac{x_{ij}}{r_{ij}^2 + x_{ij}^2}, \quad B'_{ii} = \sum_{j\in i} \frac{x_{ij}}{r_{ij}^2 + x_{ij}^2} = \sum_{j\in i} B'_{ij} \tag{2.83}$$

where r_{ij} and x_{ij} is the resistance and reactance of branch ij, respectively.

Theoretically, the factors that have less effect on voltage magnitude should be removed from $\mathbf{B}''$. For example, the effect of line resistance to $\mathbf{B}''$ should be removed. Therefore, the off-diagonal and diagonal elements of $\mathbf{B}''$ can be calculated according to the following equations:

$$B''_{ij} = -\frac{1}{x_{ij}},\ B''_{ii} = \sum_{j\in i}\frac{1}{x_{ij}} - b_{io}\ B''_{ii} = \sum_{j\in i}\frac{1}{x_{ij}} - b_{io} \tag{2.84}$$

where b_{io} is the shunt admittance of the grounding branch of node i.

If $\mathbf{B}'$ and $\mathbf{B}''$ are formed according to (2.83) and (2.84), the fast decoupled method is usually called the *BX* algorithm. Another algorithm opposite to *BX* method is called the *XB* algorithm in which $\mathbf{B}'$ used in the $\Delta P \sim \Delta\theta$ iteration is formed according to (2.84), while $\mathbf{B}''$ used in the $\Delta Q \sim \Delta V$ iteration is formed according to (2.83). Although these two algorithms have different correction equations, their convergence rates are almost the same. Several IEEE standard test systems have been calculated to compare the convergence of these algorithms. Table 2.6 shows the number of iterations needed to converge for these test systems.

Many load flow calculations indicate that *BX* and *XB* methods can converge for most load flow problems for which the Newton method can converge. The authors of [9, 10] explain the implications of the simplifications made in the fast decoupled method. Wong et al. [19] propose a robust fast decoupled algorithm to especially treat the possible convergence problem caused by high r/x networks. Bacher and Tinney [26] adopt the sparse vector technique to improve the efficiency of the fast decoupled method.

From the above discussion we know that the fast decoupled method uses different correction equations to the Newton method, hence the convergence properties are also different. Mathematically speaking, the iteration method based on a fixed coefficient matrix to solve a nonlinear equation belongs to "the constant slope method." Its convergence process has the characteristic of the geometric series. If the iteration procedure is plotted on a logarithmic coordinate, the convergence characteristic is nearly a straight line. In contrast, convergence of the Newton method has a quadratic property and is quite similar to a parabola. Fig. 2.10 shows the typical convergence properties of the two methods.

Figure 2.10 illustrates that the Newton method converges slower at the early stages, but once converged to some degree its convergence speed becomes very fast. The fast decoupled method converges almost at the same speed throughout the iteration procedure. If the specified convergence criterion is smaller than the errors

Table 2.6 Convergence comparison of ***BX*** method and ***XB*** method

Systems	Newton	*BX*	*XB*
IEEE-5 bus	4	10	10
IEEE-30 bus	3	5	5
IEEE-57 bus	3	6	6
IEEE-118 bus	3	6	7

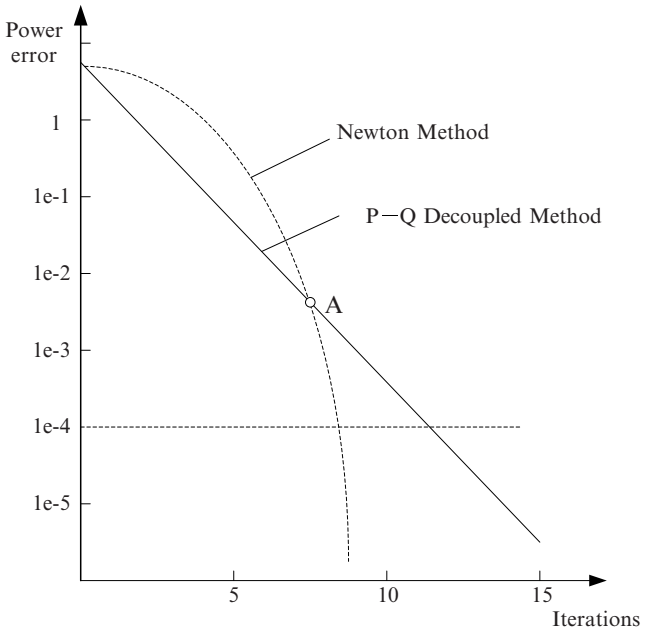

Fig. 2.10 Convergence properties of fast decoupled method and Newton method

at point A in Fig. 2.10, the iteration number of the fast decoupled method is larger than that of the Newton method. It can be roughly considered that a linear relation exists between the iteration number and the required precision when using the fast decoupled method.

Although the iteration number of the fast decoupled method is larger, its computing requirement in each iteration is far less than that of the Newton method. So the computing speed of the fast decoupled method is much higher than the Newton method.

2.4.3 Flowchart of Fast Decoupled Method

The principle flowchart of the fast decoupled method is shown in Fig. 2.11 which illustrates the main procedure and logical structure of the load flow calculation.

The symbols used in Fig. 2.11 are first introduced below:

t: counter for the iteration number

*K*01 a flag with "0" and "1" states, "0" indicates the active power iteration; while "1" the reactive power iteration. A whole iteration includes an active power iteration and a reactive power iteration.

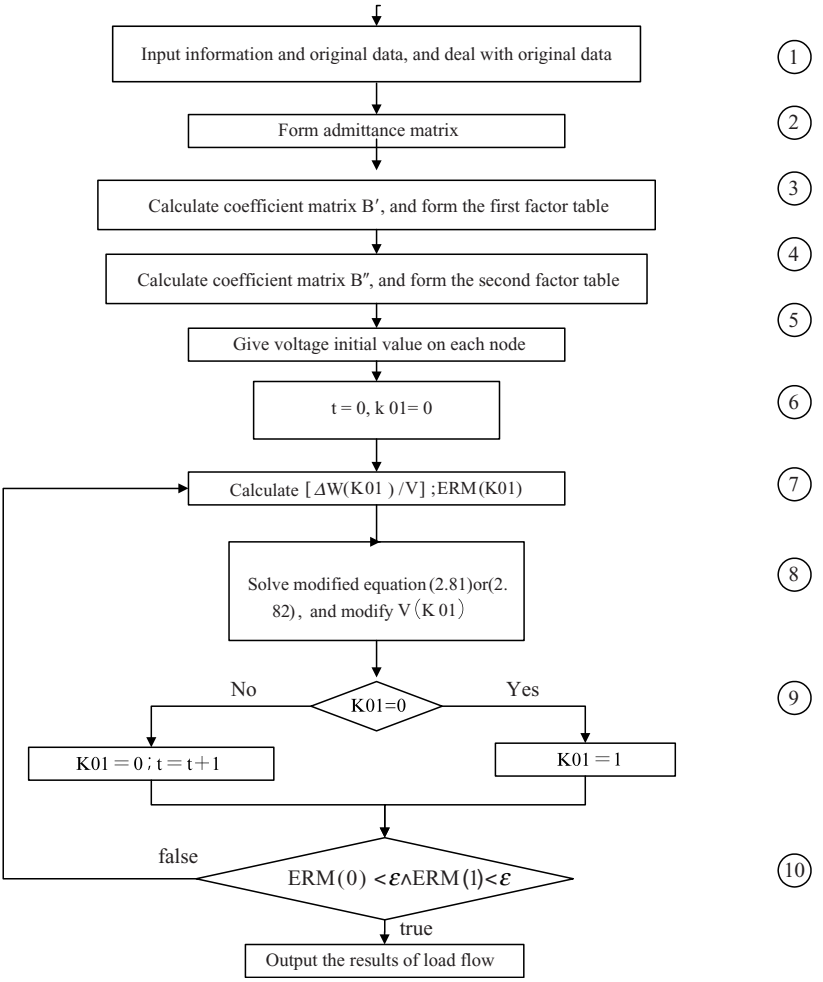

Fig. 2.11 Principle flowchart of $P-Q$ decoupled load flow program

$\Delta\mathbf{W}$: power mismatch vector: when $K01 = 0$, $\Delta\mathbf{W}(K01)$ is the mismatch of active power; when $K01 = 1$, $\Delta W(K01)$ is the mismatch of reactive power.

$\mathbf{V}$: Voltage vector: when $K01 = 0$, $\mathbf{V}(K01)$ represents voltage angle; when $K01 = 1$, $\mathbf{V}(K01)$ represents voltage magnitude.

ERM: Store the maximal power mismatch in an iteration: when $K01 = 0$, ERM ($K01$) stores the maximal active power mismatch; when $K01 = 1$, ERM($K01$) stores the maximal reactive power mismatch;

ε: Convergence criterion.

From the figure one can see, after inputting the problem data, the admittance matrix is formed. Then according to (2.83) the matrix $\mathbf{B}'$ is obtained, and triangularized to form the first factor table (block in Fig. 2.11).

After considering shunt capacitances of transmission lines and grounding branches of off-nominal taps of transformer, matrix $\mathbf{B}''$ can be formed according to (2.84), and then triangularized to form the second factor table (block in Fig. 2.11).

It should be point out that $\mathbf{B}'$ and $\mathbf{B}''$ can be formed at the same time when forming the admittance matrix. Meanwhile, the admittance matrix (block) should be stored for calculating the power mismatches according to (2.77) and (2.78).

In the flowchart, the iteration procedure is composed of blocks.

In block the initial voltage values are set accordingly for PQ nodes and PV nodes. For PQ node, the voltage magnitude can be set as the average voltage of the system; for a PV node, the voltage magnitude is set to the specified value. The voltage angle can be set to 0 as initial value for all nodes.

Block establishes the original state for iteration. The iteration procedure starts with a $P \sim \theta$ iteration, thus $K01$ is set to "0."

The iteration procedure in Fig. 2.11 follows 1θ and $1V$ mode. That is to say the iteration procedure is carries out by alternately solving $P \sim \theta$ and $Q \sim V$ correction equations.

Block calculates the node power mismatch according to (2.77) or (2.78) and records the maximal mismatch in ERM(K01) for checking the convergence condition.

Block solves correction equations, and further modifies the voltage magnitude and angle. Block establishes the state for the next iteration and counts the iteration number.

Block checks whether the iteration procedure converges. When both the $P \sim \theta$ and $Q \sim V$ iterations converge, the iteration procedure comes to an end, otherwise the process continues to the next iteration.

[Example 2.2] Using the fast decoupled method to calculate the load flow of the system shown in Fig. 2.8.

[Solution] The calculating procedure follows the flow chart of Fig. 2.11.
The admittance matrix of the system can be found in Example 1.l. The factor table used in $P \sim \theta$ iteration is

$$\begin{bmatrix} -0.15286 & -0.59620 & -0.40379 & \\ & -0.01466 & -0.06874 & -0.93125 \\ & & -0.02769 & -0.12087 \\ & & & -0.26061 \end{bmatrix}$$

It should be pointed out that $\mathbf{B}'$ used in forming the above factor table should be calculated according to (2.83). The factor table in $Q \sim V$ iteration is

$$\begin{bmatrix} -0.15135 & -0.60541 & -0.43243 \\ & -0.01541 & -0.07804 \\ & & -0.02895 \end{bmatrix}$$

Matrix $\mathbf{B}''$ used in forming the above factor table is calculated by (2.84). Because matrix $\mathbf{B}''$ does not include the elements related to PV nodes, it is a three-dimensional matrix,

$$\mathbf{B}'' = \begin{bmatrix} -6.60714 & 4.0000 & 2.85714 \\ 4.0000 & -67.30197 & 3.33333 \\ 2.85714 & 3.33333 & -36.17480 \end{bmatrix}$$

It is easy to establish the above factor table by an elimination operation on $\mathbf{B}''$.

The initial values of node voltages are similar to example 2.1 except the polar form is used here. The average operation voltage of system is:

$$V_0 = 1.00000$$

Then the initial value of node voltage vector is:

$$V_1^{(0)} = V_2^{(0)} = V_3^{(0)} = 1.00000$$

$$V_4^{(0)} = V_5^{(0)} = 1.05000$$

$$\theta_1^{(0)} = \theta_2^{(0)} = \theta_3^{(0)} = \theta_4^{(0)} = \theta_5^{(0)} = 0$$

According to (2.77) and (2.78), the functions of node power mismatches are given as follows:

$$\begin{aligned} \Delta P_1 &= P_{1s} - V_1[V_1 G_{11} + V_2(G_{12}\cos\theta_{12} + B_{12}\sin\theta_{12}) + \\ &\quad + V_3(G_{13}\cos\theta_{13} + B_{13}\sin\theta_{13})] \\ \Delta Q_1 &= Q_{1s} - V_1[-V_1 B_{11} + V_2(G_{12}\sin\theta_{12} - B_{12}\cos\theta_{12}) + \\ &\quad + V_3(G_{13}\sin\theta_{13} - B_{13}\cos\theta_{13})] \\ &\cdots\cdots \\ \Delta P_4 &= P_{4s} - V_4[V_2(G_{42}\cos\theta_{42} + B_{42}\sin\theta_{42}) + V_4 G_{44}] \end{aligned}$$

For the first $P \sim \theta$ iteration the node power mismatch can be calculated as

$$\Delta P^{(0)} = \begin{bmatrix} -1.60000 \\ -2.00000 \\ -3.70000 \\ 5.00000 \end{bmatrix}$$

Thus we have the right-hand term of the correction equation,

$$\left(\frac{\Delta P}{V}\right)^{(0)} = \begin{bmatrix} -1.60000 \\ -2.00000 \\ -3.70000 \\ 4.76190 \end{bmatrix}$$

Using the first factor table to execute elimination and backward substitution operations, we obtain the correcting value of node θ as

$$\Delta\theta^{(0)} = \begin{bmatrix} 0.09455 \\ -0.30580 \\ 0.07994 \\ -0.38081 \end{bmatrix}$$

Note that in the $P \sim \theta$ iteration, after solving the correction equation, we should obtain $\mathbf{V}_0\Delta\boldsymbol{\theta}$ (see (2.81)). But in this example, the calculation is based on per unit and $\mathbf{V}_0 = \mathbf{I}$, hence,

$$\mathbf{V}_0\Delta\boldsymbol{\theta}^{(0)} = \Delta\boldsymbol{\theta}^{(0)}$$

After modifying the node voltage angle, we get $\boldsymbol{\theta}^{(1)}$ as

$$\boldsymbol{\theta}^{(1)} = \boldsymbol{\theta}^{(0)} - \Delta\boldsymbol{\theta}^{(0)} = \begin{bmatrix} -0.09455 \\ 0.30580 \\ -0.07994 \\ 0.38080 \end{bmatrix}$$

The $Q \sim V$ iteration is carried out next. The node reactive power mismatches are

$$\Delta\mathbf{Q}^{(0)} = \begin{bmatrix} -1.11284 \\ 5.52890 \\ 1.41242 \end{bmatrix}$$

The right-hand term of the correction equation is

$$\left(\frac{\Delta\mathbf{Q}}{\mathbf{V}}\right)^{(0)} = \begin{bmatrix} -1.11284 \\ 5.52890 \\ 1.41242 \end{bmatrix}$$

Solving this equation, we obtain the voltage correct vector for PQ nodes:

$$\Delta\mathbf{V}^{(0)} = \begin{bmatrix} 0.10493 \\ -0.07779 \\ -0.03793 \end{bmatrix}$$

The modified node voltages can be calculated (see (2.80)):

$$V^{(1)} = V^{(0)} - \Delta V^{(0)} = \begin{bmatrix} 0.89057 \\ 1.07779 \\ 1.03793 \end{bmatrix}$$

Thus the first iteration is complete.

The iteration procedure repeats the above steps until the convergence condition is satisfied. When $\varepsilon = 10^{-5}$, the iteration procedure converges after ten iterations. The evolution of the node voltages is demonstrated in Table 2.7.

Table 2.8 shows the evolution of the maximal errors of the node powers and voltages in the iteration procedure.

The convergence property of the fast decoupled method used in this example is displayed in Fig. 2.12. From this figure we can see that the convergence characteristic of the fast decoupled method on a logarithmic coordinate is nearly a straight line. At the beginning, its convergence speed is faster than that of the Newton method.

The result of load flow calculation is shown in Fig. 2.13.

Table 2.7 Node voltage changes in the iteration process

Iterating No.	θ_1	V_1	θ_2	V_2	θ_3	V_3	θ_4
1	−0.09455	0.89507	0.30580	1.07779	−0.07995	1.03793	0.38080
2	−0.08227	0.87158	0.30728	1.07857	−0.07405	1.03743	0.37652
3	−0.08239	0.86512	0.31048	1.07813	−0.07448	1.03673	0.38010
4	−0.08316	0.86309	0.31117	1.07798	−0.07468	1.03652	0.38079
5	−0.08332	0.86244	0.31152	1.07794	−0.07471	1.03644	0.38115
6	−0.08339	0.86222	0.31162	1.07792	−0.07473	1.03642	0.38126
7	−0.08341	0.86215	0.31166	1.07791	−0.07473	1.03641	0.38129
8	−0.08342	0.86213	0.31167	1.07791	−0.07474	1.03640	0.38131
9	−0.08342	0.86212	0.31167	1.07791	−0.07474	1.03640	0.38131
10	−0.08342	0.86212	0.31168	1.07791	−0.07474	1.03641	0.38131

Note: the angles in the table are in rad

Table 2.8 Changes of maximal node power and voltage errors

Iterating No.	ΔP_M	ΔQ_M	$\Delta\theta_M$	ΔV_M
1	5.00000	5.52890	0.38080	0.10493
2	0.38391	0.15916	0.01228	0.02348
3	0.02660	0.03398	0.00358	0.00647
4	0.00898	0.01054	0.00077	0.00202
5	0.00279	0.00339	0.00036	0.00066
6	0.00095	0.00111	0.00011	0.00022
7	0.00031	0.00037	0.00004	0.00007
8	0.00010	0.00012	0.00001	0.00002
9	0.00003	0.00004	0.00000	0.00001
10	0.00001	0.00001	0.00000	0.00000

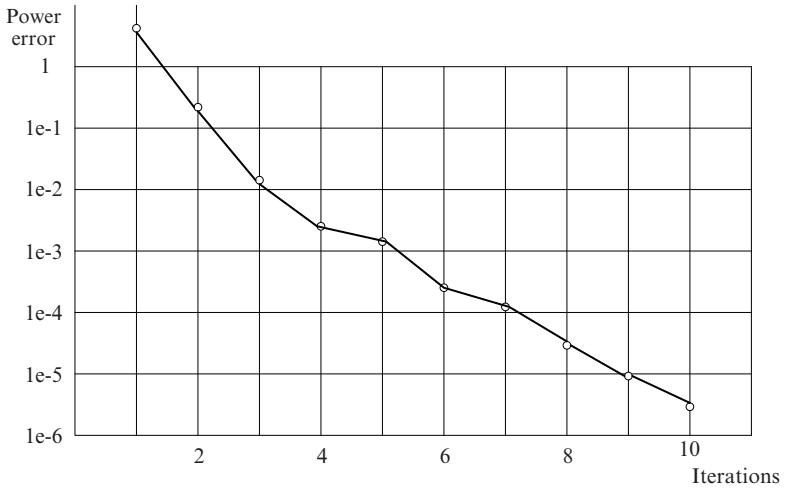

Fig. 2.12 Convergence property of $P - Q$ decoupled method

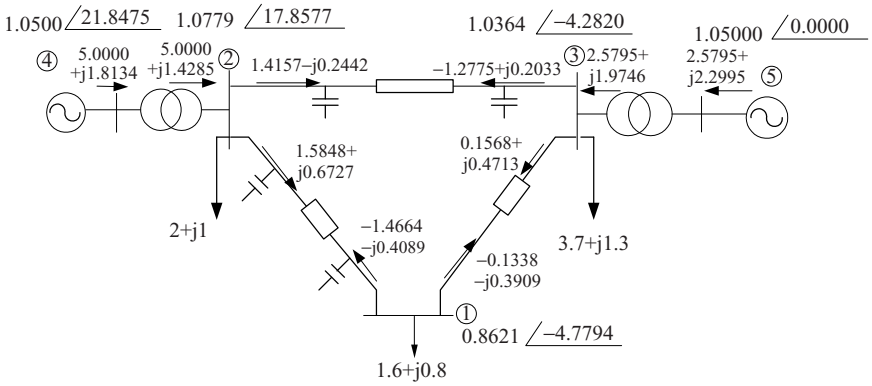

Fig. 2.13 Load flow calculation results

2.5 Static Security Analysis and Compensation Method

2.5.1 Survey of Static Security Analysis

Static security analysis is widely used in power system planning and dispatching to check the operation states when some system equipment sustains forced outages. It will answer the questions such as "what will happen if a 500 kV line is disconnected." When the results show that the power flows and voltages all are in the acceptable range, the system is static secure. When the results show that some transmission equipments are overloaded or the bus voltages of some nodes are beyond the constraints, the system is not static secure. Therefore static security analysis is a

very important part in power system security analysis and is discussed in this section. The dynamic performance analysis of power systems will be presented in the last two chapters of this book.

The static security analysis can be used in evaluating the enduring capability of a planning scheme, or an operating schedule of the power system. The static security analysis usually checks the typical forced outages of generator units or transmission equipments, onefold or two-fold. Sometimes it also inspects multi-fold outages, or common mode failures, e.g., those caused by relay system failures.

In power system planning, all credible outage cases should be considered in the static security analysis. According to the result of the static security analysis the system planner usually needs to add some redundant devices or to adjust the network scheme.

In power system operation, to avoid equipment damage and large area blackouts, the static security analysis, both online and off-line, is essential [21, 22]. In particular, the power market evolution introduces many uncertain factors to system operation, and increasing demands on the security monitor and control system.

Since the dynamic performance of the power system is not involved, the static security analysis is substantially a steady analysis problem. Through load flow calculations for all possible contingencies, we can judge whether the system is secure or not. Unfortunately, since the number of possible contingencies in static security analysis is very large, it is almost impossible to complete the task by the conventional load flow analysis method in a reasonable period of time for on-line or real-time use. Therefore, many special methods for static security analysis have been developed, such as the compensation method, DC load flow model and the sensitivity method, etc. These methods will be presented below.

2.5.2 Compensation Method

When a minor change of the network topology occurs in a power system, we can still use the original admittance matrix, even the original factor table to calculate the load flow after such a change. To accomplish this we usually use the compensation method.

The compensation method is a very useful tool in power system analysis, not only used in the static security evaluation but also widely applied in the dynamic performance study and short circuit current calculation.

We first introduce the basic principles of the compensation method.

Assume the admittance matrix and the factor table of network N shown in Fig. 2.14 have been formed, and the currents injected into the nodes are known,

$$I = \begin{bmatrix} \dot{I}_1 & \dots & \dot{I}_i & \dots & \dot{I}_j & \dots & \dot{I}_n \end{bmatrix}^{\mathrm{T}}$$

The problem in question is: when an impedance Z_{ij} is added across nodes i and j, how to solve the voltage $\boldsymbol{V}$ under the new condition by using the factor table of the original network N:

$$\dot{V} = \begin{bmatrix} \dot{V}_1 & \dot{V}_2 & \dots & \dots & \dot{V}_n \end{bmatrix}^{\mathrm{T}}$$

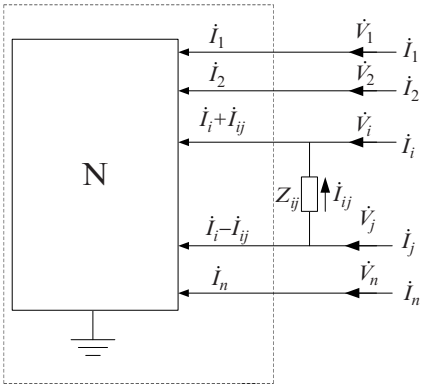

Fig. 2.14 Equivalent circuit for network branch changing

If we can get the current injecting into network N,

$$\boldsymbol{I} = \begin{bmatrix} \dot{I}_1 \\ \dot{I}_2 \\ \vdots \\ \dot{I}_i + \dot{I}_{ij} \\ \vdots \\ \dot{I}_j - \dot{I}_{ij} \\ \vdots \\ \dot{I}_n \end{bmatrix} \tag{2.85}$$

Thus the node voltage vector $\mathbf{V}$ can be calculated by an elimination and substitution manipulation on $\mathbf{I}'$ employing the original factor table. But before the node voltage vector is obtained, the current $\dot{I}_{ij}$ flowing into branch Z_{ij} is unknown. Therefore, the node voltage cannot be calculated directly according to $\mathbf{I}'$.

On the basis of the superposition principle, we can decompose network N shown in Fig. 2.14 into two equivalent networks, as showing in Fig. 2.15a, b. The node voltage vector $\mathbf{V}$ can be decomposed as

$$\mathbf{V} = \mathbf{V}^{(0)} + \mathbf{V}^{(1)} \tag{2.86}$$

where $\mathbf{V}^{(0)}$ is related to the original network without the added line, see Fig. 2.15a. Since the node injecting current vector $\mathbf{I}$ is known, $\mathbf{V}^{(0)}$ can be easily calculated by using the factor table of original network N:

$$V^{(0)} = \left[\dot{V}_1^{(0)} \quad \dot{V}_2^{(0)} \quad \dots \quad \dot{V}_i^{(0)} \quad \dots \quad \dot{V}_j^{(0)} \quad \dots \quad \dot{V}_n^{(0)} \right]^{\mathrm{T}} \tag{2.87}$$

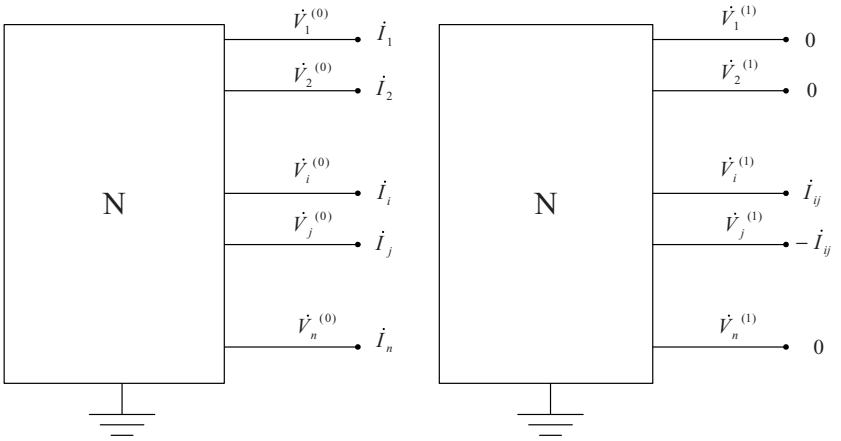

Fig. 2.15 Principle of compensation method

Now we discuss how to calculate $\mathbf{V}^{(1)}$ in Fig. 2.15b. In this figure, the current vector injected into the original network is

$$\mathbf{I}^{(1)} = \begin{bmatrix} 0 \\ \vdots \\ \dot{I}_{ij} \\ 0 \\ \vdots \\ -\dot{I}_{ij} \\ 0 \\ \vdots \end{bmatrix} = \dot{I}_{ij} \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ -1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \\ \\ \leftarrow i \\ \\ \\ \leftarrow j \\ \\ \end{matrix} \tag{2.88}$$

where $\dot{I}_{ij}$ is an unknown variable at this stage. But let $\dot{I}_{ij} = 1$, the node voltage can be calculated by using the original factor table:

$$\mathbf{V}^{(ij)} = \begin{bmatrix} \dot{V}_1^{(ij)} & \dot{V}_2^{(ij)} & \cdots & \dot{V}_i^{(ij)} & \cdots & \dot{V}_j^{(ij)} & \cdots & \dot{V}_n^{(ij)} \end{bmatrix}^{\mathrm{T}} \tag{2.89}$$

Because the network is linear, if the $\dot{I}_{ij}$ can be obtained, then the final voltage vector can be calculated by the following equation:

$$\mathbf{V} = \begin{bmatrix} \dot{V}_1^{(0)} \\ \dot{V}_2^{(0)} \\ \vdots \\ \vdots \\ \dot{V}_n^{(0)} \end{bmatrix} + \dot{I}_{ij} \begin{bmatrix} \dot{V}_1^{(ij)} \\ \dot{V}_2^{(ij)} \\ \vdots \\ \vdots \\ \dot{V}_n^{(ij)} \end{bmatrix} \tag{2.90}$$

Therefore, the problem we face now is to get $\dot{I}_{ij}$. Here we need utilize the equivalent generator principle.

As mentioned before, $\mathbf{V}^{(0)}$ is the node voltage when branch Z_{ij} is open. If we consider the whole system as the equivalent source of branch Z_{ij}, then the no-load voltage of this source is

$$\dot{E} = \dot{V}_i^{(0)} - \dot{V}_j^{(0)} \tag{2.91}$$

The equivalent internal impedance, Z_T, is

$$Z_{\mathrm{T}} = \dot{V}_i^{(ij)} - \dot{V}_j^{(ij)} \tag{2.92}$$

where $(\dot{V}_i^{(ij)} - \dot{V}_j^{(ij)})$ is the voltage drop between nodes i and j due to injecting positive and negative unit current into these nodes. Thus we have the equivalent circuit shown in Fig. 2.16, and can obtain $\dot{I}_{ij}$ directly:

$$\dot{I}_{ij} = -\frac{\dot{V}_i^{(0)} - \dot{V}_j^{(0)}}{Z'_{ij}} \tag{2.93}$$

where

$$Z'_{ij} = Z_{\mathrm{T}} + Z_{ij} \tag{2.94}$$

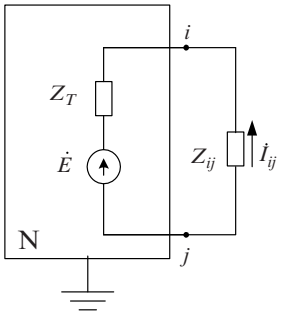

Fig. 2.16 Equivalent circuit to get $\boldsymbol{I}_{ij}$

Substituting $\dot{I}_{ij}$ into (2.90), we finally obtain node voltage vector $\mathbf{V}$.

The basic principle of compensation method has been introduced. In the practice, the compensation method can be used according to the following steps:

1. Find $\mathbf{V}^{(ij)}$ for the injecting unit current vector by using the factor table of the original normal network.

$$\mathbf{I}_{ij} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 1 \\ 0 \\ \vdots \\ -1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \\ \\ \\ \leftarrow i \\ \\ \\ \leftarrow j \\ \\ \end{matrix} \tag{2.95}$$

2. Calculate the internal impedance, Z_{T}, by (2.92), and then obtain Z'_{ij} by (2.94).
3. Calculate $\mathbf{V}^{(0)}$ by using the original factor table for the injected current vector $\mathbf{I}$ (see Fig. 2.15a).
4 Obtain the current $\dot{I}_{ij}$ flowing into branch Z_{ij} by (2.93).
5. Solving node voltage vector $\mathbf{V}$ according to (2.90).

In theory, the compensation method can also be used when more than one operation occurs simultaneously in the network. In this case, the above calculation steps should be used recursively.

Now, we will show how to use the compensation method to analysis the contingency state in the fast decoupled method.

The correction of (2.81) and (2.82) can be considered as the node equations of the network based on "admittance matrix" $\mathbf{B}'$ and $\mathbf{B}''$, and the injecting currents $\Delta\mathbf{P}/\mathbf{V}$ and $\Delta\mathbf{Q}/\mathbf{V}$, respectively. The node voltages $\mathbf{V_0}\Delta\boldsymbol{\theta}$ and $\Delta\mathbf{V}$ are the variables to be solved. In this way, the above calculation process can be followed directly. When branch ij trips, the branch impedances added between i and j for $\mathbf{B}'$ and $\mathbf{B}''$ should be (see Fig. 2.14):

$$Z'_{ij} = \frac{-1}{B_{ij}}, \quad Z''_{ij} = -x_{ij}, \tag{2.96}$$

If the tripped branch is a nonnominal tap transformer, the current representation in (2.95) should be changed as

$$\mathbf{I}^{(ij)} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ n_{\mathrm{T}} \\ 0 \\ \vdots \\ -1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \leftarrow i \\ \\ \\ \leftarrow j \\ \\ \\ \end{matrix} \tag{2.97}$$

where n_{T} is the nonnominal tap on the node j side of the transformer. In this situation, (2.91), (2.92), and (2.93) should be revised, respectively, as

$$\dot{E} = n_{\mathrm{T}}\dot{V}_i^{(0)} - \dot{V}_j^{(0)} \tag{2.98}$$

$$Z_{\mathrm{T}} = n_{\mathrm{T}}\dot{V}_i^{(ij)} - \dot{V}_j^{(ij)} \tag{2.99}$$

$$\dot{I}_{ij} = -\frac{n_{\mathrm{T}}\dot{V}_i^{(0)} - \dot{V}_j^{(0)}}{Z'_{ij}} \tag{2.100}$$

where $Z'_{ij} = Z_T + Z_{ij}$.

It should be noted, in above line outage operation, only the series branch of the opened line (or transformer) is considered in (2.96). Rigorously speaking, the shunt branches for line charging capacitance (and transformer ground branches) should also be tripped simultaneously. However, tripping three branches at the same time makes the calculation too complicated. Fortunately, practice indicates that the errors caused by neglecting grounding branches are not very significant. Therefore, the grounding branches can be neglected when the compensation method is used to analyze line outage states.

2.6 DC Load Flow Method

The DC load flow simplifies the AC load flow to a linear circuit problem. Consequently, it makes the steady state analysis of the power system very efficient. The main shortcoming of the DC load flow model is that it cannot be used in checking voltage limit violations. Because the DC load flow uses a linear model, it is not only suitable to efficiently treat the problem of line outages, but is also suitable to form linear optimization problems. Therefore, the DC load flow method has been widely used in power system planning and operating problems.

2.6.1 Model of DC Load Flow

The node active power equations of an AC load flow are given by (2.9),

$$P_i = V_i \sum_{j \in i} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \quad (i = 1, 2, \ldots, n) \tag{2.101}$$

Branch active power is

$$P_{ij} = V_i V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) - t_{ij} G_{ij} V_i^2 \tag{2.102}$$

where t_{ij} is the circuit transformer ratio per unit of branch ij, θ_{ij} is the phase angle difference across branch ij; G_{ij}, B_{ij} are the real and imaginary parts of corresponding elements of the node admittance matrix, respectively.

$$\theta_{ij} = \theta_i - \theta_j \tag{2.103}$$

$$G_{ij} + jB_{ij} = -\frac{1}{r_{ij} + jx_{ij}} = \frac{-r_{ij}}{r_{ij}^2 + x_{ij}^2} + j\frac{x_{ij}}{r_{ij}^2 + x_{ij}^2} \tag{2.104}$$

where, r_{ij}, x_{ij} are resistance and reactance of line ij. When $i = j$,

$$G_{ii} = -\sum_{\substack{j \in i \\ j \neq i}} G_{ij}$$

$$B_{ii} = -\sum_{\substack{j \in i \\ j \neq i}} B_{ij}$$

Under assumptions applied in the fast decoupled method, the above AC load flow equations can be simplified to the following equations.

$$P_i = \sum_{j \in i} B_{ij} \theta_{ij} \quad (i = 1, 2, \ldots, n)$$

which can be rewritten as,

$$P_i = \sum_{j \in i} B_{ij} \theta_i - \sum_{j \in i} B_{ij} \theta_j \quad (i = 1, 2, \ldots, n)$$

From (2.104), we know the first term in the right hand of the above equation is 0, thus we have,

$$P_i = -\sum_{j \in i} B_{ij} \theta_j \quad (i = 1, 2, \ldots, n) \tag{2.105}$$

The DC flow model usually has no negative sign, thus we redefine B_{ij} as,

$$B_{ij} = -\frac{1}{x_{ij}} \tag{2.106}$$

thus

$$B_{ii} = \sum_{\substack{j \in i \\ j \neq i}} \frac{1}{x_{ij}} \tag{2.107}$$

Finally, we establish the DC flow equation,

$$P_i = \sum_{j \in i} B_{ij} \theta_j \quad (i = 1, 2, \ldots, n) \tag{2.108}$$

or in matrix form,

$$\mathbf{P} = \mathbf{B}\boldsymbol{\theta} \tag{2.109}$$

where $\mathbf{P}$ is $\boldsymbol{\theta}$ the node injection power vector and its ith element is given by $P_i = P_{Gi} - P_{Di}$, here P_{Gi} and P_{Di} are the generator output and load at node i, respectively; is the phase angle vector and $\mathbf{B}$ is the matrix whose elements are defined by (2.106) and (2.107).

Equation (2.109) can also be expressed as follows

$$\boldsymbol{\theta} = \mathbf{X}\mathbf{P} \tag{2.110}$$

where $\mathbf{X}$ is the inverse of matrix $\mathbf{B}$,

$$\mathbf{X} = \mathbf{B}^{-1} \tag{2.111}$$

Similarly, substituting the simplifying conditions into (2.102), one obtains the active power flowing into branch ij,

$$P_{ij} = -B_{ij}\theta_{ij} = \frac{\theta_i - \theta_j}{x_{ij}} \tag{2.112}$$

or in matrix form,

$$\mathbf{P_l} = \mathbf{B}_l \boldsymbol{\Phi} \tag{2.113}$$

If the number of branches is l, $\mathbf{B}_l$ is an $l \times l$ diagonal matrix whose elements are branch admittance; $\mathbf{P_l}$ is the branch active power vector; $\boldsymbol{\Phi}$ is the end terminal phase angle difference vector.

Assuming that the network incidence matrix is **A**, then one arrives at

$$\mathbf{\Phi} = \mathbf{A}\boldsymbol{\theta} \tag{2.114}$$

Equations (2.109), (2.110), and (2.113) are basic DC load flow equations which are linear. Under given system operation conditions, the state variable θ may be obtained through triangularizition or matrix inversion from (2.110), then branch active power can be obtained from (2.113).

2.6.2 Outage Analysis by DC Load Flow Method

From the above discussion, it can be seen that it is very simple to solve system state and active power flow by DC load flow equations. It will also be shown that because these equations are linear, it is possible to carry out fast load flow computation after adding or tripping a line.

Assuming that the original network nodal impedance matrix is $\boldsymbol{X}$ and a branch k is connected between nodes i and j. When a line with reactance x_k is added in parallel with branch k, a new network is formed. We now demonstrate how to obtain the new network state vector in this situation from the original network impedance matrix and state vector.

Assuming the new network impedance matrix is $\boldsymbol{X}'$, it can be obtained according to the branch adding principle of section 1–4 (see (2.1–2.107)),

$$\mathbf{X}' = \mathbf{X} - \frac{\mathbf{X_L}\mathbf{X_L^T}}{\mathbf{X_{LL}}} \tag{2.115}$$

where $\mathbf{X_L} = \mathbf{X}\mathbf{e}_k$,

$$\mathbf{e}_k = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ -1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \\ \\ \leftarrow i \\ \\ \leftarrow j \\ \\ \\ \end{matrix} \tag{2.116}$$

$$\mathbf{X}' = \mathbf{X} - \frac{\mathbf{X}\mathbf{e}_k\mathbf{e}_k^T\mathbf{X}}{x_k + \mathbf{e}_k^T\mathbf{X}\mathbf{e}_k} \tag{2.117}$$

Equation (2.117) can be further reduced to,

$$\mathbf{X}' = \mathbf{X} + \beta_k\mathbf{X}\mathbf{e}_k\mathbf{e}_k^T\mathbf{X} \tag{2.118}$$

where

$$\beta_k = \frac{-1}{x_k + \chi_k} \tag{2.119}$$

$$\chi_k = \mathbf{e}_k^{\mathrm{T}} \mathbf{X} \mathbf{e}_k = X_{ii} + X_{jj} - 2X_{ij} \tag{2.120}$$

where X_{ii}, X_{jj}, X_{ij} are elements of $\mathbf{X}$.

From (2.118), the incremental change of the nodal impedance matrix is given by:

$$\Delta \mathbf{X} = \mathbf{X}' - \mathbf{X} = \beta_k \mathbf{X} \mathbf{e}_k \mathbf{e}_k^{\mathrm{T}} \mathbf{X} \tag{2.121}$$

According to (2.121) and (2.110), under constant nodal injection power conditions, the change in original state vector after adding line k is

$$\Delta \boldsymbol{\theta} = \Delta \mathbf{X} \mathbf{P} = \beta_k \mathbf{X} \mathbf{e}_k \phi_k \tag{2.122}$$

where $\phi_k = \mathbf{e}_k^{\mathrm{T}} \boldsymbol{\theta}$, is the terminal phase angle difference of branch k before the change. The new network state vector is given by

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \Delta \boldsymbol{\theta} = \boldsymbol{\theta} + \beta_k \mathbf{X} \mathbf{e}_k \phi_k \tag{2.123}$$

Thus after adding line k, the new network nodal impedance matrix and the new state vector can be obtained by (2.118) and (2.123) using the original network parameters. When line k trips, the above equations can still be applied with x_k being replaced by $-x_k$.

If the outage of branch k causes system disconnection, the new impedance matrix $\mathbf{X}'$ does not exist and β_k of (2.119) becomes infinite, i.e., $-x_k + \chi_k = 0$. Therefore, it is very easy to check whether the outage of a branch will cause system disconnection by using the DC load flow equation. However, it is impossible to carry out line outage analysis directly.

2.6.3 N-1 Checking and Contingency Ranking Method

A network design has to satisfy certain operational security requirements. As discussed earlier, a common network operational security requirement is to satisfy N-l checking, i.e., when one of N branches fails, the system operation criteria remain within given requirements. At the initial stage of forming a network configuration, the principle is to ensure that there is no overloading in the network; i.e., the network satisfies the requirements for securely transmitting power. To this end, one has to carry out the overload check for successive line outages. If the outage of a line causes overloading or system disconnection, then the network does not satisfy N-1 checking.

The strict N-l checking on all branches needs N line outage analyses, resulting in a large amount of computing. In practice, some line outages do not cause system overloading. Therefore, a contingency ranking is carried out according to the probability of system overload being caused by a line outage, then the checking is first performed on the lines with higher probability. If the checking of a line indicates that its outage does not cause overloading, the lines with lower rank are not subjected to any further checking, which significantly reduces the amount of computing. Such a process is also called contingency selection. A number of contingency ranking methods are available in the literature [23, 24], each having a different criterion for assessing the system contingency. This section describes a contingency ranking method based on the criterion of system overloading.

To reflect the overall system overloading, a system performance index (PI) is defined as follows:

$$PI = \sum_{l=1}^{L} \alpha_l w_l \left(\frac{P_l}{\bar{P}_l} \right)^2 \tag{2.124}$$

where P_l, active power of line l
$\bar{P}_l$, transmission capacity of branch l
α_l, number of parallel lines for branch l
w_l, weighting factor of line l, which reflects the influence of a fault
L, number of branches in the network

It can be seen from (2.124) that when there is no overloading, $P_l/\bar{P}_l$ is not greater than 1, the PI is small. When there is overloading in the system, $P_l/\bar{P}_l$ for the overloaded line is greater than 1 and the positive exponential element makes the PI relatively large. Therefore, this index generally reflects the system security. It may also be possible to use a higher order exponential instead of a square element in the equation to further obviate the overloading problem.

A sensitivity analysis of the PI with respect to the change of a line admittance will reveal the impact of an outage on the system security. When line k fails, the change in the PI is given by

$$\Delta PI_k = \frac{\partial PI}{\partial B_k} \Delta B_k \tag{2.125}$$

where $\Delta B_k = B_k$, is the admittance of line k. The bigger ΔPI_k is, the larger the increase in the PI will be, which indicates that the probability of a faulted line k causing system overloading becomes higher.

ΔPI_k may be calculated from Telegen's theorem and the adjoint network method. In the following study, we will derive a formula to calculate ΔPI_k directly using nominal load flow results.

Assuming that after line k fails other line flows become $P'(l = 1, 2, \ldots, L; l \neq k)$, the system performance index becomes,

$$PI' = \sum_{l=1}^{L} \alpha_l w_l \left(\frac{P'_l}{\bar{P}_l} \right)^2 \tag{2.126}$$

Hence

$$\Delta PI_k = PI' - PI \tag{2.127}$$

For the sake of simplicity, we change the system index to a function of voltage angles and express it in the matrix form. From (2.113),

$$\mathbf{P_l} = \mathbf{B}_l \phi_l \tag{2.128}$$

Substituting the above equation into (2.124) and defining

$$PI_\phi = PI = \sum_{l=1}^{L} \alpha_l w_l \frac{(B_l \phi_l)^2}{\bar{P}_l^2} = \phi^{\mathrm{T}} w_d \phi \tag{2.129}$$

where

$$\phi^{\mathrm{T}} = [\phi_1, \ldots, \phi_k, \ldots, \phi_L]$$

and

$$\mathbf{w}_d = \begin{bmatrix} \frac{\alpha_1 w_1 B_1^2}{\bar{P}_1^2} & & & 0 \\ & \cdots & & \\ & & \frac{\alpha_k w_k B_k^2}{\bar{P}_k^2} & \\ & & \cdots & \\ 0 & & & \frac{\alpha_L w_L B_L^2}{\bar{P}_L^2} \end{bmatrix}$$

Substituting (2.114) into (2.129), one obtains

$$PI_\phi = \boldsymbol{\theta}^{\mathrm{T}} \mathbf{A}^{\mathrm{T}} \mathbf{w}_d \mathbf{A} \boldsymbol{\theta} = \boldsymbol{\theta}^{\mathrm{T}} \omega \boldsymbol{\theta} \tag{2.130}$$

where

$$\mathbf{w} = \mathbf{A}^{\mathrm{T}} \mathbf{w}_d \mathbf{A} \tag{2.131}$$

is a symmetric matrix. Matrix $\mathbf{w}$ has the same structure as matrix $\mathbf{B}$. Thus its formation is equivalent to directly forming the admittance matrix using element $\alpha_l w_l B_l^2 / \bar{P}_l^2$ to replace B_l. Similarly, PI'_Φ can be expressed as

$$PI'_\phi = \boldsymbol{\theta}'^{\mathbf{T}} \mathbf{w} \boldsymbol{\theta}' \tag{2.132}$$

where $\boldsymbol{\theta}'$ is the voltage angle vector after the line k fails.

Equation (2.132) contains all elements relevant to line k which should not appear in the new system performance index PI'. Thus

$$PI' = PI'_{\phi} - w_k \frac{(B_k \phi'_k)^2}{\bar{P}_k^2} \tag{2.133}$$

Substituting (2.130) and (2.133) into (2.127), one obtains

$$\Delta PI_k = PI'_{\phi} - PI_{\phi} - \frac{w_k B_k^2}{\bar{P}_k^2}(\phi'_k)^2 = \boldsymbol{\theta}'^{\mathbf{T}}\mathbf{w}\boldsymbol{\theta}' - \boldsymbol{\theta}^{\mathbf{T}}\mathbf{w}\boldsymbol{\theta} - \frac{w_k B_k^2}{\bar{P}_k^2}(\phi'_k)^2 \tag{2.134}$$

From (2.123), we know

$$\begin{aligned} \boldsymbol{\theta}' &= \boldsymbol{\theta} + \beta_k \mathbf{X}\mathbf{e}_k \phi_k \\ \phi'_k &= \mathbf{e}_k \theta' = (1 + \beta_k \chi_k)\phi_k \end{aligned}$$

Substituting the above two equations into (2.134), we have

$$\begin{aligned} \Delta PI_k &= (\boldsymbol{\theta} + \beta_k \mathbf{X}\mathbf{e}_k \phi_k)^{\mathbf{T}}\mathbf{w}(\boldsymbol{\theta} + \beta_k \mathbf{X}\mathbf{e}_k \phi_k) - \boldsymbol{\theta}^{\mathbf{T}}\mathbf{w}\boldsymbol{\theta} - \frac{w_k B_k^2}{\bar{P}_k^2}(1 + \beta_k \chi_k)^2 \phi_k^2 \\ &= \beta_k \phi_k (\boldsymbol{\theta}^{\mathrm{T}}\mathbf{w}\mathbf{X}\mathbf{e}_k + \mathbf{e}_k^{\mathrm{T}}\mathbf{X}\mathbf{w}\boldsymbol{\theta}) + \beta_k^2 \phi_k^2 \mathbf{e}_k^{\mathrm{T}}\mathbf{X}\mathbf{w}\mathbf{X}\mathbf{e}_k - \frac{w_k B_k^2}{\bar{P}_k^2}(1 + \beta_k \chi_k)^2 \phi_k^2 \end{aligned} \tag{2.135}$$

Taking into account the symmetry of matrices $\mathbf{X}$ and $\mathbf{w}$, let

$$\begin{aligned} \gamma_k &= \boldsymbol{\theta}^{\mathrm{T}}\mathbf{w}\mathbf{X}\mathbf{e}_k = \mathbf{e}_k^{\mathrm{T}}\mathbf{X}\mathbf{w}\boldsymbol{\theta} = \mathbf{e}_k^{\mathrm{T}}\mathbf{R} \\ \tau_k &= \mathbf{e}_{\mathbf{k}}^{\mathbf{T}}\mathbf{X}\mathbf{w}\mathbf{X}\mathbf{e}_k = \mathbf{e}_{\mathbf{k}}^{\mathbf{T}}\mathbf{T}\mathbf{e}_k \end{aligned} \tag{2.136}$$

where

$$\begin{aligned} \mathbf{R} &= \mathbf{X}\mathbf{w}\boldsymbol{\theta} \\ \mathbf{T} &= \mathbf{X}\mathbf{w}\mathbf{X} \end{aligned} \tag{2.137}$$

Substituting (2.136) into (2.135), one obtains

$$\Delta PI_k = 2\beta_k \phi_k \gamma_k + \beta_k^2 \phi_k^2 \tau_k - \frac{w_k B_k^2}{\bar{P}_k^2}(1 + \beta_k \chi_k)^2 \phi_k^2 \tag{2.138}$$

When line k fails, β_k in the above equations becomes

$$\beta_k = \frac{-1}{-x_k + \chi_k} = \frac{B_k}{1 - B_k\chi_k}$$

Substituting the above equation into (2.138) gives,

$$\Delta PI_k = \frac{2B_k\phi_k\gamma_k}{1 - B_k\chi_k} + \frac{B_k^2\phi_k^2\tau_k}{(1 - B_k\chi_k)^2} - \frac{w_kB_k^2\phi_k^2}{(1 - B_k\chi_k)P_k^2} \tag{2.139}$$

Because $P_k = B_k\phi_k$,

$$\Delta PI_k = \frac{2P_k\gamma_k}{1 - B_k\chi_k} + \frac{P_k^2\tau_k}{(1 - B_k\chi_k)^2} - \frac{w_kP_k^2}{(1 - B_k\chi_k)P_k^2} \tag{2.140}$$

Equations (2.138), (2.139), and (2.140) have no essential difference except for different expressions. Variables in these equations are obtained from the normal load flow calculation. Under the condition that matrices **X**, **w**, **R**, **T** have been formed, it is very convenient to compute ΔPI after a line outage.

The process of contingency ranking is essential to compute the values of ΔPI from (2.138) [or (2.139) and (2.140)] for all lines and arrange them in descending order of magnitude of ΔPI. During the line outage analysis, load flow calculation and overload checking are first carried out on the line with the largest value of ΔPI, and then the procedure is continued until there is no overload caused by the outage of certain lines. The lines with smaller values of ΔPI are not subjected to further analysis because the probability of overload caused by other outages is very small. However, the use of this system performance index may cause a "screening" effect. For example, the value of ΔPI for the situation where there is overloading in some lines and the flow in the other lines is very small may be smaller than that for the situation where there is no overloading but line flows are large. Therefore, the contingency ranking by this index may introduce some error. In practice, one may decide that the line outage analysis is terminated only after a number of consecutive line outages do not cause system overloading.

Thinking and Problem Solving

1. What functions do the swing bus and *PV* buses in load flow calculations have? How should they be selected?
2. Compare the advantages and disadvantages of nodal power equations with polar coordinates and rectangular coordinates.
3. Give the characteristics of Newton method based modified equations in load flow calculation.
4. Give the physical meaning of Jacobian elements of the modified equation.

5. Give the flowchart of the Newton method based load flow calculation by polar coordinate equations.
6. Design the storage modes of the Jacobian matrix elements of the Newton method based load flow calculation with two kinds of coordinates.
7. How should node conversion, such as changing a PV node into a PQ node, or changing a PQ node into a PV node, be implemented in the design of a load flow program?
8. What simplified suppositions are considered in the $P-Q$ decomposition method? Why is it that the $P-Q$ decomposition method can obtain the same calculation accuracy as the Newton method after so many suppositions?
9. How can we improve the convergence of the $P-Q$ decomposition method when the ratio R/X is very big?
10. How can the compensation method be applied to the case with two branches out of service?
11. Prove that the DC load flow model has the same solution as that of the following optimizing problem:
    a. obj $\min \sum_{ij \in B} P_{ij} X_{ij}^2 P_{ij}$ and X_{ij} are the active load flow and reactance of branch ij, B is the branch set.
    b. $s.t.$ $\sum_{ij \in i} P_k = 0$ $ij \in i$ denotes all branches that connect to node i.
12. Discuss the issues raised by the N-1 checking method being used as static security analysis tool for electrical power systems.