Unbalanced Power Systems Resolution Including Power and Frequency Regulation

by

Samuel AUBERT

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE
SUPÉRIEURE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
M.Sc.A.

MONTREAL, 29 SEPTEMBER 2016

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS:

Pierre-Jean Lagacé, Thesis Supervisor
Department of Electrical Engineering, École de technologie supérieure

Maarouf Saad, Thesis Co-Supervisor
Department of Electrical Engineering, École de technologie supérieure

Ambrish Chandra, President of the Board of Examiners
Department of Electrical Engineering, École de technologie supérieure

Laurent Lenoir, External Examiner
Institut de recherche d'Hydro-Québec

Serge Lefebvre, External Examiner
Institut de recherche d'Hydro-Québec

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON 16 SEPTEMBER 2016

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I would like to express my appreciation to Pierre-Jean Lagacé for his valuable advice and generosity throughout the project. I would also like to thank Maarouf Saad, Laurent Lenoir, Serge Lefebvre, and the GRÉPCI team for useful discussions and communications. This work was supported by Hydro-Québec, as part of the RDC on voltage control in view of renewable energies in power distribution systems, and by the École de technologie supérieure.

Last but not least, I am highly indebted to my wife for her constant support and patience.

RÉSOLUTION DE RÉSEAUX ÉLECTRIQUES DÉSÉQUILIBRÉS COMPRENANT LA RÉGULATION DE PUISSANCE ET DE FRÉQUENCE

Samuel AUBERT

RÉSUMÉ

L'analyse par écoulement de puissance constitue une approche efficace de résolution de systèmes de réseaux électriques en régime permanent. Elle joue un rôle fondamental dans les études et la conception de tels systèmes. Ce travail revisite un modèle de barre d'équilibre distribuée, l'adapte à la résolution de réseaux triphasés (déséquilibrés) par la méthode de Newton et finalement l'implémente dans un programme informatique. Le modèle tient compte de transactions économiques effectuées entre des réseaux interconnectés et de la variation de fréquence occasionnée par des perturbations de charge et de puissance générée. À titre d'étude préliminaire, un modèle de barre d'équilibre unique considérant la fréquence comme étant fixe est également décrit et implémenté. Plusieurs matrices d'admittance de composants de réseaux électriques sont développées à des fins de polyvalence. Des composants monophasés sont aussi mis en place de façon à faciliter le traitement de charges déséquilibrées et à permettre l'analyse de systèmes de transmission et de distribution électrique. Par ailleurs, la régulation de tension au moyen de transformateurs triphasés et monophasés à changement de prise est aussi possible. Le fonctionnement et la logique du programme informatique sont expliqués en détail. Grâce à des procédures vectorielles, entre autres mesures, une grande vitesse de calcul fut réalisée. Plusieurs exemples numériques sont présentés, parmi lesquels figure celui d'un grand réseau de 3000 barres triphasées (9000 noeuds). Comme il se doit, dans chacun des cas les conditions imposées sont remplies et le bilan des puissances est respecté à chacune des barres.

Mots clés: Écoulement de puissance triphasé, charge déséquilibrée, génération de puissance distribuée, régulation de puissance, régulation de fréquence, méthode de Newton, Newton-Raphson

UNBALANCED POWER SYSTEMS RESOLUTION INCLUDING POWER AND FREQUENCY REGULATION

Samuel AUBERT

ABSTRACT

Power flow analyses constitute an effective tool in determining the steady state solution of power systems. Hence they play a fundamental role in the investigation and design of such systems. The present work revisits a distributed slack bus model, adapts it to the resolution of three-phase (unbalanced) power systems by Newton's method, and implements it in a computer program. The model takes into consideration economic power transactions between areas of a network, as well as the variation in network frequency resulting from load versus generated power perturbations and ensuing control operations. As an initial investigation, a single slack bus model that assumes a constant frequency is also described and implemented. Various component admittance matrices are derived explicitly. Single-phase components are also implemented in order to facilitate the treatment of unbalanced loads, and to permit the study of systems that comprise both transmission and distribution elements. In addition, the voltages at specified buses can be regulated through the operation of tap changers installed on three-phase and single-phase transformers. The implementation code is discussed in detail. High computation speeds could be reached by devising several array-based procedures, among other measures. A number of numerical examples are presented, among which is a large network containing 3000 three-phase buses (9000 nodes). As it should, in every case the imposed conditions are met, and the balance in power is respected at every bus.

Keywords: Three-phase power flow, unbalanced load, distributed generation, power regulation, frequency regulation, Newton's method, Newton-Raphson

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

AGC	Automatic Generation Control
ch.	Chapter
div. tol.	The minimum allowed error for any element of the mismatch vector ε to assess divergence of the program's iterative process
excl.	Excluding
fig.	Figure
Gen1	Single-phase generator bus (power and frequency regulation model)
Gen3	Three-phase generator bus (power and frequency regulation model)
Ld	Load bus (power and frequency regulation model)
nr3r	The name of the computer program that implements the models of ch. 2 (*nr* stands for Newton-Raphson, *3* for three-phase circuits, and *r* for regulation of power and frequency)
P1	Designation of the set of all *PV1* (*Gen1*) and *PQ* (*Ld*) nodes
pu	Per unit
PV1	Single-phase *PV* bus (classic model)
PV3	Three-phase *PV* bus (classic model)
Ref.	Reference bus (power and frequency regulation model)
sec.	Section
tol.	The maximum allowed error for any element of the mismatch vector ε to assess convergence of the program's iterative process

LIST OF SYMBOLS

C_{adj}	Code that indicates which action is to be taken in response to the inspection of a tap changer and its associated controlled voltage
$diag(x)$	Diagonal matrix having as its diagonal elements the elements of the vector x
f_{sch}	Scheduled network frequency
N	Total number of nodes in the network
N_{br}	Total number of branches in the network
n	Total number of three-phase branches
N_{tb}	Total number of tie branches in the network
N_{zg}	Total number of zigzag earthing transformers in the network
n_{adj1}	Number of tap adjustments actually performed per iteration
n_{adj2}	Number of necessary tap adjustments per iteration
n_{mis}	Counter of the number of times that all necessary tap adjustments cannot be performed. A value of 3 or more leads to a condition violation status.
n_t	Total number of taps of a tap changer
n_x	Actual tap position of a tap changer
P	Active power
$P_{T\,sch}$	Scheduled power transfer
Q	Reactive power
S	Apparent power
θ_{ij}	Shorthand for $\theta_i - \theta_j$
$x[i]$	In the computing context, the i^{th} component of array x
Y_{br}	Branch admittance matrix
Y_{bus}	Network admittance matrix (or bus admittance matrix). Where no confusion may arise, Y_{bus} may be written as Y.
Y_m	Transformer magnetizing admittance (also represented without subscript where no confusion may arise)

Y_{tr}	Transformer admittance matrix
$\equiv$	Definition
∇_x	Gradient with respect to the vector x. For simplicity, the subscript x may be dropped.
$\in$	Is an element of
$\otimes$	Kronecker product

INTRODUCTION

Power flow studies represent a valuable tool in determining the steady state solutions of large and complex power systems. For given load conditions and network information, such as the topology and impedance characteristics, they yield the voltages, currents, powers, and in some cases other key variables of the system. As a result, power flow studies are of great use in the planning, designing, and maintaining of power networks.

The present thesis revisits a distributed slack bus model, adapts it to three-phase (unbalanced) systems, and discusses its implementation in a computer program. The model is referred to as the transferred power and frequency regulation model. As its name indicates, it takes into consideration the effects of economic power transactions between areas of a network, as well as variations in frequency resulting from network perturbations and ensuing control operations. Additionally, single-phase network components are implemented. Doing so facilitates the treatment of unbalanced loads, and it allows to solve systems akin to combined transmission and distribution power systems. Besides, the voltage regulation by tap changers installed on both three-phase and single-phase transformers is realized. For comparative purposes, a second model which assumes a single slack bus and no frequency variation is also examined and implemented in the same framework. Such model is referred to as the classic model.

The present project was motivated by the need of Hydro-Québec to perform multiple parallel analyses of large networks comprising both transmission and distribution systems. In particular, the development of smart grids, the growing involvement of homes in energy production, and their increasing capability for energy self-management and process automation call for detailed investigations of the impact of combining distribution networks with transmission networks. The resolution of such systems is a challenging task given the necessity to reach solutions in a short time; efficient algorithms and procedures must be devised to that effect. Moreover, the differing characteristics of those types of network present a difficulty; for

instance, their dissimilar R/X ratios (low in transmission systems versus high in distribution systems) can lead to ill-conditioned power flow problems.

The above-mentioned assignment was realized through the use of Newton's method (also known as the Newton-Raphson method). Though it exhibits a good convergence rate and robustness, it is still inferior in those regards to the Levenberg-Marquardt mehod, as discussed in Lagacé (2012). Nevertheless, Newton's method remains widely used in power flow analyses, and it is generally reliable given reasonable initial conditions. Hence it represents a good foundational tool to tackle the problem of solving unbalanced complex power systems. Furthermore, the program developed may be modified with relative ease to rely on the Levenberg-Marquardt method if its robustness is eventually shown to be insufficient in face of certain ill-conditioned problems. Another limitation of the work is its disregard for generator power limits. In other words, no change from a generator bus to a load bus is operated when a generator reaches its physical power limits. Such functionality adds complexity to the program, and it was judged unnecessary for the current needs of the project.

The thesis is divided into four main parts. The first chapter provides the backdrop for the investigation by reviewing relevant papers from the literature. The second chapter first presents theoretical concepts necessary to set up the power flow problem. Then it derives the admittance matrices for various network components, and describes how to combine them to obtain the bus admittance matrix. Next, the classic model and the power and frequency regulation model are formulated. The third chapter describes the procedures devised to implement the two power flow models in a computer program. A flow chart is also provided to better understand the interactions of those constituent parts of the program. The fourth chapter contains a set of numerical problems that illustrate the main functionalities of the program. Those problems also contribute to its validation process. Unless otherwise stated, derivations and computations

are made in the per-unit system. Finally, additional supporting information can be found in the appendices. In particular, a detailed program instruction manual is included in Appendix III.

CHAPTER 1

BACKGROUND

Power flow studies of large networks have long been performed by applying Newton's method to a system of mismatch equations at different nodes (or buses) categorized as generator nodes or load nodes, and where a so-called slack bus is defined additionally to compensate without constraint the difference between the scheduled power generation and the load throughout the network. This approach was put forward by van Ness and Griffin (1961), and adapted by Tinney and Hart (1967), among others, to extended systems through the use of optimally ordered Gaussian elimination combined with programming techniques to work around contemporary computer limitations.

In spite of its success at providing solutions to large power systems in a relatively robust way, the above-mentioned framework is limited in that it does not reflect significant physical aspects of electrical networks, such as the variation in frequency resulting from load and generation disturbances. Furthermore, it assumes unrealistically an ideal bus capable of making up supply versus demand unbalances at will. Thus Okamura *et al.* (1975) proposed a more general model, which also relies on Newton's method, but which differs from its predecessors in that it takes into account load and generator characteristics, as well as control schemes in effect to respond to changes in a network. Though it considers control factors, the model focuses on providing a steady-state solution based on a set of boundary conditions. As required, the model frequency is allowed to vary in such a way as to obtain an equilibrium between the power generated, the load, and the losses across the system. As opposed to using a single slack bus, the model allows for the distribution of the compensation of the unbalance between supply and demand among all generators. Such principles are referred to as automatic load frequency control. Furthermore, the load is allowed to vary in accordance with variations in frequency and voltage levels. Similarly, an expression for the generated power that incorporates the effect of system variables, in particular the network frequency, is introduced.

A paper published by Saadat (1979) revisits the load flow problem as formulated by Okamura *et al.* (1975), though by employing a power-perturbation technique to determine iteratively the solution. The proposed algorithm's convergence rate was observed to be similar to the one of the conventional Newton-Raphson method. Yet the approach is more or less flexible, in part due to its necessity to specify the real and reactive powers for all buses (except the reference bus).

In like manner, Ćalović and Strezoski (1981) developed a steady-state load flow model based on Newton's method, which takes into account the effect of frequency deviation, control operations of generators, as well as the load dependence on frequency and voltage. No slack bus is used. The corresponding bus is treated as a regular system bus with equations that express the balance in active and reactive power. As a result, two additional unknowns are contributed to the system, typically the magnitude and the phase of the voltage at the bus in question. Primary control (prime mover response) and secondary control (automatic generation control, or AGC) are conducted at distinct buses. If a bus i is involved with primary control (p), then the generated active power is expressed as $P_{Gi} = P_{G0i} + k_i^p \Delta f$, where k_i^p reflects the speed-droop setting on the turbine governor of the generator system, and Δf stands for the steady-state frequency deviation. Likewise, if it is involved with secondary control (s), then $P_{Gi} = P_{G0i} + k_i^s \Delta G$, where k_i^s is the participation factor of the generator in the overall control activity, and ΔG is the static area control error (a linear combination of the steady-state frequency deviation and the tie line power deviation). In both cases, P_{G0i} represents the base load on the generator at the bus. If a bus does participate in neither primary nor secondary control, then P_{Gi} simply reduces to P_{G0i}. Per iteration, either Δf or ΔG is treated as an additional unknown variable. Depending on the limitations of the power plants and the load conditions, the model thus requires a switch between the scheme taking into account the effect of AGC (with ΔG as an unknown) and the one that takes into account the primary control (with Δf as an unknown). Such switching mechanism complicates the implementation of the model into a program. With regard to voltage regulation, it is performed in two ways: by a change of reactive injection at the relevant bus(es), in which case the reactive injection Q_i takes the place of the voltage magnitude in the

set of unknowns, or by on-load tap changers whose action is reproduced by the introduction of a turn-ratio variable, which similarly replaces the respective voltage magnitude in the set of unknowns. In the latter case, a limitation is encountered as the turn-ratio variable takes part in the calculations as a continuous variable. Additional operations are thus needed to account for its actual discrete nature.

Thus far the determination of the power generation in the distributed slack bus methods was generally made in consideration of the characteristics of turbines at generator buses and their control scheme. A different view is taken by Zobian and Ilić (1997), who extend the work of Ćalović and Strezoski (1981), but who assume a constant network frequency, and perform an economic dispatch of the net power imbalance to generating units through participation factors determined on the basis of combined cost and reliability criteria. The paper seeks to clarify the impact of simultaneous economic transactions of power flows in interconnected power networks, and to calculate the contribution of the generating units that participate in balancing the system in response to transactions.

According to Tong and Miu (2005), the models discussed so far may not be suitable to describe distribution systems, for example due to the fact that generally the main source of a distribution system is the substation, which does not exhibit characteristics peculiar to turbines. Moreover, the high R/X ratios typical of distribution systems demand for a consideration of the allocation of loss among the generating units. Consequently, based on Kirschen *et al.* (1997) and Strbac *et al.* (1998), the authors use the concept of generator domains to compute the participation factors iteratively. More concretely, therein the participation factor K_i associated with generator i is obtained as the ratio of the loss associated with the generator and the total real power loss in the system ($K_i = P_{Gi}^{loss}/P_{Loss}$). In this way, the active power generation at bus i is defined by $P_{Gi} = P_{Gi}^{load} + K_i P_{Loss}$, where P_{Loss} is taken as an additional system unknown to be determined (along with the other unknowns) by Newton's method. Thus the definition of the generator domains takes into consideration the locations of generators, the network topology, as well as the load distribution. Interestingly, the model is defined in a three-phase framework, thus

allowing for the specification of unbalanced network conditions. However, no consideration of network frequency is explicitly made.

In heavily unbalanced network conditions, in particular as seen in distribution networks, neutral currents may be significant, and therefore may not be ignored in power flow analyses. Besides, neutral point voltages may also deviate notably from zero. In such circumstances, a reformulation of the power flow problem is necessary to include explicitly the effect of the neutral conductors and grounding in the system mismatch equations. Penido *et al.* (2008) propose such a methodology, where Newton's method is used to solve a set of nonlinear equations derived from the net current injections at all nodes of an n bus system. The system is then composed of $8n$ equations, where the unknowns are (basically) taken to be the real and imaginary parts of the voltages (in rectangular form) at all three phases and at the neutral point. The introduction of the neutral nodes adds complexity to the analysis, and requires modifications to the more common component models.

With the focus on large-scale distribution systems of arbitrary topologies, Kocar *et al.* (2014) examine three load-flow algorithms based on the modified augmented nodal analysis (MANA). The algorithms are characterized respectively by the use of fixed-point iterations, Newton's method, and the so-called dishonest Newton method, which does not require the calculation of the Jacobian matrix at every iteration. MANA permits the treatment of multi-phase and unbalanced networks, and it is flexible in accommodating load flow constraints. Regulator tap controls are also implemented. A relatively large network is used to compare the performance of the three approaches. The one using Newton's method is the most robust, and it requires the least number of iterations. However, in the particular case examined, its solution time is slightly longer than with the other methods. The fixed-point iterations procedure generally shows poorer convergence, and it requires more iterations to reach the solution. Yet in certain conditions it can still outperform Newton's method. As regards the Dishonest Newton method, it presents a compromise between the other two approaches in terms of calculation time, though its ability to converge depends on the chosen Jacobian matrix update frequency. On the whole,

the authors present avenues worth considering to improve a program's ability to solve complex distribution systems.

In a first stage, the present work consists in the implementation of a single slack bus model (referred to as the classic model) in a three-phase framework. The classic model is helpful in providing basic solutions to (unbalanced) power systems with moderate computing efforts. In a second stage, a distributed slack bus model (referred to as the transferred power and frequency regulation model) is implemented based on the ideas of Okamura *et al.* (1975), yet in the three-phase framework, to accept unbalanced network conditions. This second model incorporates important features of physical network behaviour, such as the effects of primary and secondary control. The program is designed in such a way as to accept further functionalities, e.g. specific to distribution networks, without the necessity of a full overhaul. Various algebraic expressions for admittance matrices of components have also been derived, and the tap changer functionality has been implemented.

CHAPTER 2

NETWORK MODELLING

2.1 Power-flow analysis based on Newton's method

The power-flow analysis is a well-known approach to compute the steady-state voltages at all buses of an electrical network, and indirectly to calculate the power through the components of the network. Moreover, it can be used to determine the value of other variables such as the network frequency. Wood and Wollenberg (1996) offer an informative presentation of the method.

The analysis consists in driving key quantities, referred to as the mismatches, to zero, by varying a set of variables through an iterative process. The mismatches are stored in a vector $\varepsilon(x)$, and the variables in the vector x. The notation $\varepsilon(x)$ implies that each entry of $\varepsilon(x)$ is dependent on the unknowns contained in x. The derivation of the mismatches relies on the fact that a balance must be reached in the active power and in the reactive power leaving every node i of the network, to wit

$$\varepsilon_{Pi}(x) \equiv P_{Yi}(x) + P_{Li}(x) - P_{Gi}(x) = 0 \tag{2.1}$$

$$\varepsilon_{Qi}(x) \equiv Q_{Yi}(x) + Q_{Li}(x) - Q_{Gi}(x) = 0 \tag{2.2}$$

where the subscript Y indicates the power that enters the Y_{bus} components, L the load, and G the power generation. Depending on the types of bus present in the network, a subset of the ε_{Pi} and ε_{Qi} quantities is kept to form the vector $\varepsilon(x)$, and node voltage moduli and phases are selected to build the vector of unknowns x.

The above procedure results in a nonsingular system, which may be solved by Newton's method. Referring to the Appendix I (where g is replaced by ε), a solution for x is found by carrying out iteratively the computations contained in algorithm 2.1. As its name indicates, the

parameter *max iteration* represents the maximum number of iterations allowed, and *tolerance* stands for the maximum permissible mismatch value to assess convergence.

Algorithm 2.1 Basic implementation of Newton's method

Input : x_0 (initial values of the unknowns)
Output : x (solved values of the unknowns), $\varepsilon(x)$ (mismatch values)

```
1 x = x_0                                          # initialization of the unknowns
2 ε_0 = ε(x)                                       # initial mismatch values
3 for i = 1 to max iteration do
4     if max{ε(x)} ≤ tolerance then break
5     ∇ε(x)                                        # Jacobian matrix
6     Δx = −[∇ε(x)]^{−1} ε(x)                      # adjustment terms
7     x = x + Δx                                   # adjustments of the unknowns
8     ε(x)                                         # updated mismatch values
9 end
```

The derivation of the power flow equations will be presented in greater details in the remainder of the chapter. In particular, two power flow models will be described, namely (i) the classic model and (ii) the power and frequency regulation model. In the latter, additional relations and variables will be introduced to take into consideration variations in the network frequency as well as constraints imposed on the power distribution. The specific programming procedures developed to implement Newton's method are explained in Chapter 3.

2.2 Power entering the bus

As seen in several texts on power systems[1], the voltages at the N nodes of a network can be related to the currents entering the network at those nodes via an admittance matrix Y:

$$I_i = \sum_{k=1}^{N} Y_{ik} V_k$$

[1] See, for instance, p. 100 of Wood and Wollenberg (1996).

where the subscripts i and k label the nodes. Thus the power entering every node i can be expressed as follows:

$$\begin{aligned} S_{Y\,i} &= P_{Y\,i} + jQ_{Y\,i} \\ &= V_i I_i^* \\ &= V_i \sum_{k=1}^{N} Y_{ik}^* V_k^* \end{aligned} \tag{2.3}$$

By breaking down Y_{ik} into its real and imaginary parts, and V_i and V_k into their respective moduli and phases, i.e.

$$\begin{aligned} Y_{ik} &= G_{ik} + jB_{ik} \\ V_i &= |V_i|\, \mathrm{e}^{j\theta_i} \\ V_k &= |V_k|\, \mathrm{e}^{j\theta_k} \end{aligned}$$

the active and reactive power in (2.3) can be rewritten as

$$P_{Y\,i} = \sum_{k=1}^{N} |V_i|\,|V_k|\,(G_{ik}\cos\theta_{ik} + B_{ik}\sin\theta_{ik}) \tag{2.4}$$

$$Q_{Y\,i} = \sum_{k=1}^{N} |V_i|\,|V_k|\,(G_{ik}\sin\theta_{ik} - B_{ik}\cos\theta_{ik}) \tag{2.5}$$

where θ_{ik} is a shorthand for $\theta_i - \theta_k$.

2.3 The bus admittance matrix

2.3.1 Construction of the bus admittance matrix

The admittance matrix introduced in sec. 2.2 is commonly known as the bus admittance matrix. As displayed in (2.4) and (2.5), it determines the power flowing through the buses of the network. Thus its knowledge is essential to perform power flow calculations. Herein is

described how the matrix (also denoted Y_{bus}) is obtained from a combination of the admittance matrices of its constituent branches (Y_{br}) and shunt components. Later in the chapter, the explicit Y_{br} matrices of several types of branches and the admittance matrix of the zigzag earthing transformer will be derived.

Given a circuit component (e.g. power line, transformer, shunt element), based on Kirchhoff's laws the vector of the voltages at its nodes (V_x) can be related to the vector of the currents injected at those nodes (I_x) by a matrix Y_x, i.e. $I_x = Y_x V_x$. For example, in the simplest case a shunt element $Y_{sh} = 0.5$ pu connected to a node at $V = 1$ pu implies a current injected in the element of $I = Y_{sh} V = 0.5$ pu (in this one-dimensional case, the admittance matrix is in fact a scalar).

The same applies when several circuit components are connected to a common set of nodes. Each component will be traversed by current as determined by its admittance matrix and node voltages. The total current injected in the network at a given node will merely be the sum of the currents entering the network components connected to that node. Hence, if the admittance matrices of the branches and shunt components of a network are known, then their combined effect can also be obtained simply by adding them systematically into a wider matrix that reflects the impedance characteristics of the network.

The combination of the admittance elements is achieved by associating each node with a unique index, corresponding to a row and a column of Y_{bus}. Admittance values of uncoupled shunt elements are added to the diagonal of Y_{bus}, because each such element is connected between a node and the ground. The zigzag earthing transformer, which also connects a bus to the ground, requires the addition of diagonal as well as off-diagonal admittance elements, because of its coupling among the phases of the bus. As to the branches, they link nodes in various ways that depend on the topology of the network. As a result, the row and column indices of Y_{br} associate with the ones of Y_{bus} in various ways. Nevertheless, by knowing the buses to which the branches connect, it is possible to add the elements of the Y_{br} matrices to the appropriate Y_{bus} entries. Note that a three-phase branch will require the addition of 36 elements (3 primary

nodes connected to 3 secondary nodes, which leads to a 6×6 matrix), whereas a single-phase branch will require the addition of 4 elements (1 primary node connected to 1 secondary node, which leads to a 2×2 matrix). Detailed information regarding the Y_{bus} construction procedure is provided in sec. 3.3.

2.3.2 Derivation method of network component admittance matrices

The derivation of the admittance matrices of individual network components consists in the following steps:

a. Specify an adequate circuit to model the component;

b. Apply Kirchhoff's laws to the nodes and meshes of the circuit specified in step a.;

c. If the component is a transformer, then determine the per unit winding ratio such that phase or line voltages have respectively the same per unit magnitude on both sides of the transformer;

d. Rearrange the equations resulting from steps b. (and c.) so as to obtain a system of the form $M_1 I = M_2 V$. Since $I = YV$, the admittance matrix can be evaluated simply, that is $Y = M_1^{-1} M_2$.

In the last step, I represents the vector of the currents entering the component, and V the vector of voltages at the corresponding junctions of the component.

Note that circuit quantities such as transformer coil impedances and the magnetizing admittance are treated in the per unit system.

2.3.3 Notation

Before delving into the derivation of the admittance matrices of the network components, it is worthwhile to lay basic notation so as to improve the manageability of three-phase equations.

The first two items below, namely the vector of propagation constants, and the vectors of characteristic impedances and admittances, are used to obtain the admittance matrix of the three-phase power line. The following items mainly help to simplify the transformer-related equations.

Vector of propagation constants The propagation constants corresponding to the zero, positive and negative sequences are gathered into a single vector

$$\gamma_{012} = \begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{bmatrix} = \begin{bmatrix} \sqrt{z_0 y_0} \\ \sqrt{z_1 y_1} \\ \sqrt{z_2 y_2} \end{bmatrix}$$

where y_i and z_i, $i \in \{0,1,2\}$, are the admittance and impedance per unit length in the three modes.

Vectors of characteristic impedances and admittances Likewise, the characteristic impedances corresponding to the zero, positive and negative sequences are gathered into the single vector

$$Z_{c012} = \begin{bmatrix} Z_{c0} \\ Z_{c1} \\ Z_{c2} \end{bmatrix} = \begin{bmatrix} \sqrt{z_0/y_0} \\ \sqrt{z_1/y_1} \\ \sqrt{z_2/y_2} \end{bmatrix}$$

The vector of characteristic admittances Y_{c012} is the element-wise inverse of Z_{c012}.

Phase subscripts Capital letters are used to indicate phases on the primary side of a component, and lower case letters to indicate phases on the secondary side. For example, I_A and I_a are respectively the phase A currents on the primary and secondary sides of a given component.

Voltage and current vectors V_{xyz} denotes the 3×1 vector containing the voltages at the nodes x, y and z of the same three-phase bus, e.g.

$$V_{ABC} = \begin{bmatrix} V_A \\ V_B \\ V_C \end{bmatrix}$$

For simplicity, the vector of voltages at phases labelled by a prime, e.g. x', y', and z', may be written as V'_{xyz}. The same notation applies to the current vector I_{xyz}.

Furthermore, the phase voltages and currents on the primary and secondary sides of a transformer may be combined into the single 6×1 vectors:

$$V = \begin{bmatrix} V_{ABC} \\ V_{abc} \end{bmatrix} \quad \text{and} \quad I = \begin{bmatrix} I_{ABC} \\ I_{abc} \end{bmatrix}$$

Identity matrices The $m \times m$ identity matrix is denoted by I_m.

Unit matrices The $m \times n$ unit matrix is denoted by $J_{m \times n}$, or by J_m if $m = n$.

Zero matrices The $m \times n$ zero matrix is denoted by $0_{m \times n}$, or by 0_m if $m = n$.

Permutation matrices $P_{(123)}$ denotes the third order matrix that permutes the elements of a 3×1 vector as follows:

$$\begin{bmatrix} x_3 \\ x_1 \\ x_2 \end{bmatrix} = P_{(123)} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

The inverse of $P_{(123)}$ is denoted by $P_{(132)}$, and takes the form

$$P_{(132)} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

2.3.4 Power lines

2.3.4.1 Single-phase line

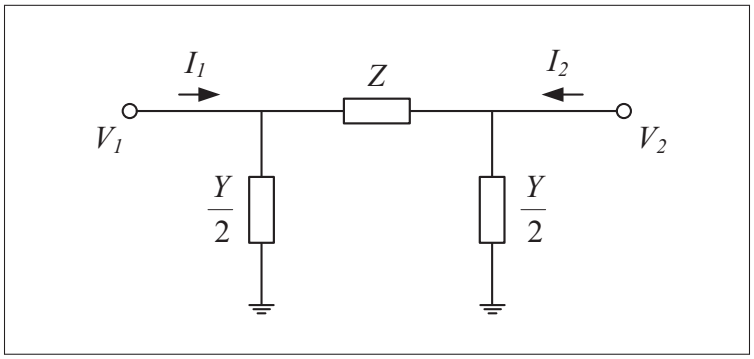


Figure 2.1 Equivalent π-circuit of a single-phase power line

By modelling the single-phase power line as a π circuit, as shown in fig. 2.1, and by applying Kirchhoff's laws, it is straightforward to determine the corresponding admittance matrix $Y_{\ell 1\phi}$:

$$Y_{\ell 1\phi} = \begin{bmatrix} \frac{Y}{2} + \frac{1}{Z} & -\frac{1}{Z} \\ -\frac{1}{Z} & \frac{Y}{2} + \frac{1}{Z} \end{bmatrix} \tag{2.6}$$

In the general case (e.g. including lines longer than 250 km), it can be shown that

$$\frac{Y}{2} = \frac{\tanh(\gamma\ell/2)}{Z_c} \tag{2.7}$$

$$Z = Z_c \sinh(\gamma \ell) \tag{2.8}$$

where $\gamma = \sqrt{zy}$, $Z_c = \sqrt{z/y}$, z is the impedance per unit length, y the admittance per unit length, and ℓ the length of the line. A detailed derivation of (2.7) and (2.8) can be found in Glover *et al.* (2012).

2.3.4.2 Three-phase line

As a first approximation, the three-phase line may be treated as three decoupled π circuits of the form shown in fig. 2.1. As a consequence, the admittance matrix is constructed as follows:

$$Y'_{\ell 3\phi} = \begin{bmatrix} \frac{Y}{2}+\frac{1}{Z} & 0 & 0 & -\frac{1}{Z} & 0 & 0 \\ 0 & \frac{Y}{2}+\frac{1}{Z} & 0 & 0 & -\frac{1}{Z} & 0 \\ 0 & 0 & \frac{Y}{2}+\frac{1}{Z} & 0 & 0 & -\frac{1}{Z} \\ -\frac{1}{Z} & 0 & 0 & \frac{Y}{2}+\frac{1}{Z} & 0 & 0 \\ 0 & -\frac{1}{Z} & 0 & 0 & \frac{Y}{2}+\frac{1}{Z} & 0 \\ 0 & 0 & -\frac{1}{Z} & 0 & 0 & \frac{Y}{2}+\frac{1}{Z} \end{bmatrix} \tag{2.9}$$

A more refined model should incorporate the effect of electromagnetic coupling between the phases. In addition, values for the impedance and shunt admittance could be obtained using a similar approach as in the case of the single-phase long line model. Thus, let us model the three-phase line as a combination of line segments such as the one shown in fig. 2.2, where the impedance and admittance values per unit length are gathered in the matrices

$$Z_{abc} = \begin{bmatrix} z_{aa} & z_{ab} & z_{ac} \\ z_{ab} & z_{bb} & z_{bc} \\ z_{ac} & z_{bc} & z_{cc} \end{bmatrix} \qquad Y_{abc} = \begin{bmatrix} y_{aa} & y_{ab} & y_{ac} \\ y_{ab} & y_{bb} & y_{bc} \\ y_{ac} & y_{bc} & y_{cc} \end{bmatrix} \tag{2.10}$$

Note that due to the bilateral symmetry of the phase-to-phase links, the matrices Z_{abc} and Y_{abc} are symmetric.

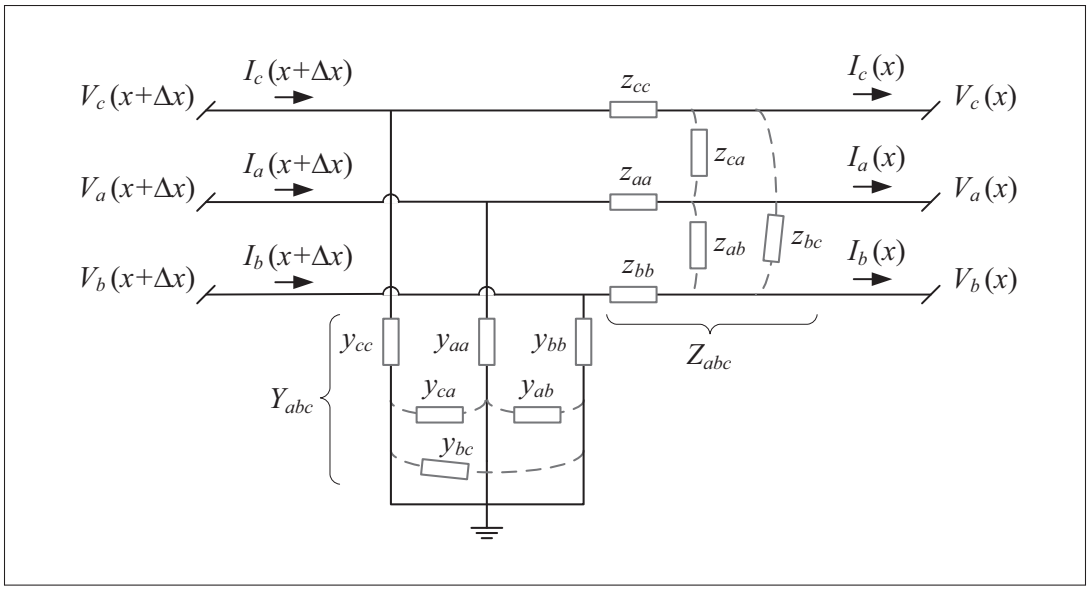


Figure 2.2 Equivalent circuit of a segment of a three-phase power line

Based on fig. 2.2,

$$V_{abc}(x+\Delta x) - V_{abc}(x) = Z_{abc}\,\Delta x\, I_{abc}(x)$$
$$I_{abc}(x+\Delta x) - I_{abc}(x) = Y_{abc}\,\Delta x\, V_{abc}(x+\Delta x)$$

Dividing both sides by Δx, and taking the limit as $\Delta x \to 0$ gives the differential equations

$$\frac{dV_{abc}(x)}{dx} = Z_{abc}\, I_{abc}(x)$$
$$\frac{dI_{abc}(x)}{dx} = Y_{abc}\, V_{abc}(x)$$

which may be differentiated and combined to yield

$$\frac{d^2V_{abc}(x)}{dx^2} = Z_{abc}\, Y_{abc}\, V_{abc}(x) \tag{2.11}$$
$$\frac{d^2I_{abc}(x)}{dx^2} = Y_{abc}\, Z_{abc}\, I_{abc}(x) \tag{2.12}$$

analogously to the single-phase case. A solution to the system formed by (2.11) and (2.12) can be found easily by diagonalizing $Z_{abc}Y_{abc}$ and $Y_{abc}Z_{abc}$.

In a way similar to Kersting (2002), for completely transposed lines, the matrices of impedance and admittance per unit length are expressed in the simple forms

$$Z_{abc} = \begin{bmatrix} z_s & z_m & z_m \\ z_m & z_s & z_m \\ z_m & z_m & z_s \end{bmatrix} \qquad Y_{abc} = \begin{bmatrix} y_s & y_m & y_m \\ y_m & y_s & y_m \\ y_m & y_m & y_s \end{bmatrix}$$

where the diagonal and off-diagonal elements of the impedance matrix in (2.10) are averaged to give the self and mutual impedances, i.e. $z_s = (z_{aa}+z_{bb}+z_{cc})/3$ and $z_m = (z_{ab}+z_{ac}+z_{bc})/3$, and similarly for the self and mutual admittances, which are given by $y_s = (y_{aa}+y_{bb}+y_{cc})/3$ and $y_m = (y_{ab}+y_{ac}+y_{bc})/3$. In such case, the diagonalizing process can be achieved by means of the Fortescue matrix ($a = e^{j2\pi/3}$)

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & a^2 & a \\ 1 & a & a^2 \end{bmatrix}$$

that is

$$\frac{d^2V_{012}(x)}{dx^2} = (A^{-1}Z_{abc}Y_{abc}A)\,V_{012}(x)$$
$$\frac{d^2I_{012}(x)}{dx^2} = (A^{-1}Y_{abc}Z_{abc}A)\,I_{012}(x)$$

Note that the diagonal matrices $A^{-1}Z_{abc}Y_{abc}A$ and $A^{-1}Y_{abc}Z_{abc}A$ are equal; in particular

$$Z_{012} = A^{-1}Z_{abc}A = \begin{bmatrix} z_0 & 0 & 0 \\ 0 & z_1 & 0 \\ 0 & 0 & z_1 \end{bmatrix}$$

$$Y_{012} = A^{-1} Y_{abc} A = \begin{bmatrix} y_0 & 0 & 0 \\ 0 & y_1 & 0 \\ 0 & 0 & y_1 \end{bmatrix}$$

where $z_0 = z_s + 2z_m$ and $z_1 = z_s - z_m$ (similarly for y_0 and y_1).

Hence, by expressing the voltage and current vectors in the basis of the symmetrical components, i.e. $V_{abc} = A V_{012}$ and $I_{abc} = A I_{012}$, the equations (2.11) and (2.12) become decoupled, and the solution for each mode can be solved as in the single-phase case. Taking $x = \ell$ and $x = 0$ at the left and right ends of the line respectively, in compact form it follows that

$$\begin{bmatrix} V_{012}(\ell) \\ I_{012}(\ell) \end{bmatrix} = \begin{bmatrix} diag(\cosh(\gamma_{012}\ell)) & diag(Z_{c012}\sinh(\gamma_{012}\ell)) \\ diag(Y_{c012}\sinh(\gamma_{012}\ell)) & diag(\cosh(\gamma_{012}\ell)) \end{bmatrix} \begin{bmatrix} V_{012}(0) \\ I_{012}(0) \end{bmatrix}$$

Converting back to the *abc* basis yields

$$\begin{bmatrix} V_{abc}(\ell) \\ I_{abc}(\ell) \end{bmatrix} = \begin{bmatrix} A_3 & B_3 \\ C_3 & A_3 \end{bmatrix} \begin{bmatrix} V_{abc}(0) \\ I_{abc}(0) \end{bmatrix} \tag{2.13}$$

where

$$\begin{aligned} A_3 &= A\ diag(\cosh(\gamma_{012}\ell))A^{-1} \\ B_3 &= A\ diag(Z_{c012}\sinh(\gamma_{012}\ell))A^{-1} \\ C_3 &= A\ diag(Y_{c012}\sinh(\gamma_{012}\ell))A^{-1} \end{aligned}$$

By rearranging (2.13), and using $I'_{abc} \equiv -I_{abc}$, the standard form $I = Y_{\ell 3\phi} V$ can be obtained:

$$\begin{bmatrix} I_{abc}(\ell) \\ I'_{abc}(0) \end{bmatrix} = \begin{bmatrix} A_3 B_3^{-1} & C_3 - A_3 B_3^{-1} A_3 \\ -B_3^{-1} & A_3 B_3^{-1} \end{bmatrix} \begin{bmatrix} V_{abc}(\ell) \\ V_{abc}(0) \end{bmatrix}$$

Using the hyperbolic identity $1-\coth^2 x = -\operatorname{csch}^2 x$, the off-diagonal block matrices $-B_3^{-1}$ and $C_3 - A_3 B_3^{-1} A_3$ can be shown to be equal. Therefore, the square matrix simplifies to give

$$Y_{\ell 3\phi} = \begin{bmatrix} Y_a & Y_b \\ Y_b & Y_a \end{bmatrix} \tag{2.14}$$

where

$$Y_a = A\ diag(Y_{c012}\coth(\gamma_{012}\ell))A^{-1}$$
$$Y_b = -A\ diag(Y_{c012}\operatorname{csch}(\gamma_{012}\ell))A^{-1}$$

By performing the matrix multiplications explicitly, and noting that $Y_{c1} = Y_{c2}$ and $\gamma_1 = \gamma_2$ (since $z_1 = z_2$ and $y_1 = y_2$), the expressions for the elements of Y_a and Y_b are obtained:

$$Y_a = \begin{bmatrix} Y_{a1} & Y_{a2} & Y_{a2} \\ Y_{a2} & Y_{a1} & Y_{a2} \\ Y_{a2} & Y_{a2} & Y_{a1} \end{bmatrix} \qquad Y_b = \begin{bmatrix} Y_{b1} & Y_{b2} & Y_{b2} \\ Y_{b2} & Y_{b1} & Y_{b2} \\ Y_{b2} & Y_{b2} & Y_{b1} \end{bmatrix}$$

$$Y_{a1} = \frac{1}{3}\Big(Y_{c0}\coth(\gamma_0\ell) + 2Y_{c1}\coth(\gamma_1\ell)\Big) \qquad Y_{b1} = -\frac{1}{3}\Big(Y_{c0}\operatorname{csch}(\gamma_0\ell) + 2Y_{c1}\operatorname{csch}(\gamma_1\ell)\Big)$$
$$Y_{a2} = \frac{1}{3}\Big(Y_{c0}\coth(\gamma_0\ell) - Y_{c1}\coth(\gamma_1\ell)\Big) \qquad Y_{b2} = -\frac{1}{3}\Big(Y_{c0}\operatorname{csch}(\gamma_0\ell) - Y_{c1}\operatorname{csch}(\gamma_1\ell)\Big)$$

An analogous derivation applying to the case where the matrices $Z_{abc}Y_{abc}$ and $Y_{abc}Z_{abc}$ cannot be diagonalized by the Fortescue matrix (e.g. the case of non-transposed lines) can be found in Acha and Usaola (2009).

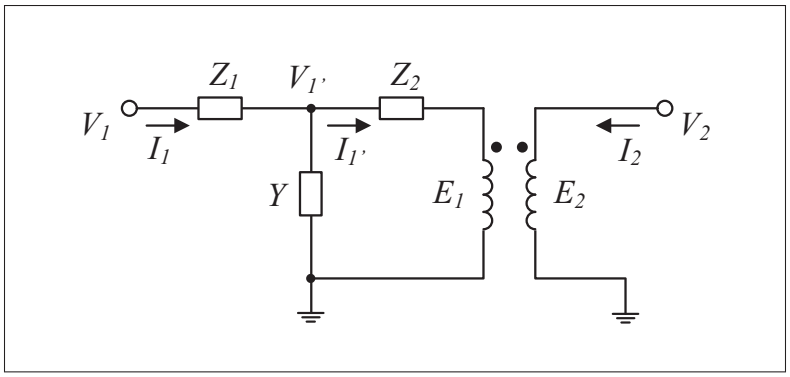


Figure 2.3 Equivalent circuit of the single-phase transformer

2.3.5 Power transformers

2.3.5.1 The single-phase transformer

The circuit used to model the single-phase transformer is shown in fig. 2.3. Application of Kirchhoff's laws results in the following equations:

$$V_1 = Z_1 I_1 + V_{1'} \tag{2.15}$$

$$V_{1'} = Z_2 I_{1'} + E_1 \tag{2.16}$$

$$I_1 = Y V_{1'} + I_{1'} \tag{2.17}$$

Furthermore, in the per unit system

$$E_1 = E_2 = V_2 \tag{2.18}$$

$$I_{1'} = -I_2 \tag{2.19}$$

Substituting I_1' in (2.17) by $-I_2$, and using the resulting expression to eliminate V_1' in equation (2.15) yields

$$V_1 = \left(Z_1 + \frac{1}{Y}\right) I_1 + \frac{I_2}{Y} \tag{2.20}$$

Similarly, E_1 and I_1' in (2.16) can be substituted by means of (2.18) and (2.19), and V_1' in (2.15) can be replaced by the resulting expression, thus giving

$$V_1 - V_2 = Z_1 I_1 - Z_2 I_2 \tag{2.21}$$

Finally, (2.20) and (2.21) can be arranged in the matrix form $M_1 I = M_2 V$, where

$$M_1 = \begin{bmatrix} Z_1 & -Z_2 \\ Z_1 + 1/Y & 1/Y \end{bmatrix}$$

$$M_2 = \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix}$$

such that

$$Y_{tr1\phi} = M_1^{-1} M_2 = \frac{1}{Z_1 + Z_2 + Y Z_1 Z_2} \begin{bmatrix} 1 + Y Z_2 & -1 \\ -1 & 1 + Y Z_1 \end{bmatrix}$$

2.3.5.2 Three-phase transformer classification

Three-phase transformers come in a variety of types, based on their coil connections. Therefore, in order to distinguish the ones that are modelled herein unambiguously, the classification system described below was adopted.

Transformer configurations are identified as in IEC (2011), with two exceptions: first, upper case letters denote winding connection types connected to the branch primary bus (first bus specified in the branch input data structure), and lower case letters denote winding connection types on the side of the secondary bus (second bus specified in the branch input data structure); second, the neutral point of a star or zigzag winding is always assumed to branch out, and thus for simplicity no symbol is used to make such distinction. Connections come in three varieties, namely star, delta, and zigzag, and are denoted respectively by the letters Y, D, and Z (or y,

d, and *z*). The displacement between corresponding phases (normally phase *a*) is expressed in terms of the numbers 0 to 11 as arranged on an analog clock, each corresponding to a 30° increment. The phase voltage on the primary side is placed at position 0 (the reference), and the secondary phase voltage lies at a position that is dependent on the configuration. By convention, phasors turn anticlockwise. Thus, numbers on the right hand side of the clock indicate that the secondary voltage phasor lags the corresponding primary voltage phasor (e.g. position 1 for a 30° lag), and vice versa for the numbers on the other half of the clock. The classification method is illustrated by the phasor diagrams of sec. 2.3.5.3 to 2.3.5.8.

2.3.5.3 *Yy0* transformer

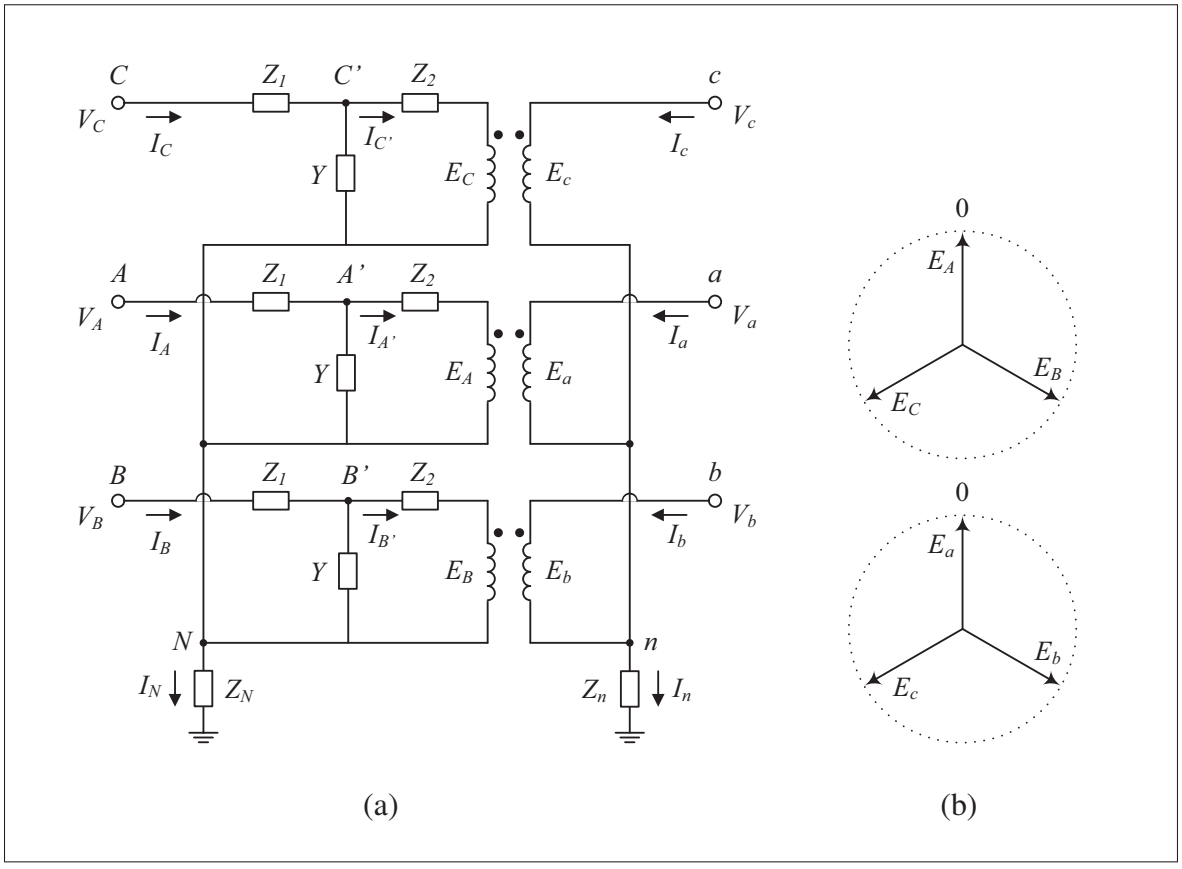


Figure 2.4 Equivalent circuit (a) and phasor representation (b) of the *Yy0* transformer

The circuit used to model the in-phase star-star transformer is shown in fig. 2.4 (a). Application of Kirchhoff's laws results in the following equations:

$$V_{ABC} = V'_{ABC} + Z_1 I_{ABC} \tag{2.22}$$

$$V'_{ABC} - Z_N I_N J_{3\times 1} = Z_2 I'_{ABC} + E_{ABC} \tag{2.23}$$

$$I_{ABC} = I'_{ABC} + Y(V'_{ABC} - Z_N I_N J_{3\times 1}) \tag{2.24}$$

$$I_N = I_A + I_B + I_C \tag{2.25}$$

$$V_{abc} = E_{abc} + Z_n I_n J_{3\times 1} \tag{2.26}$$

$$I_n = I_a + I_b + I_c \tag{2.27}$$

Furthermore, in the per unit system, matching voltages and currents on opposite sides of the transformer demands that $E_{ABC} = E_{abc}$ and $I'_{ABC} = -I_{abc}$; fig. 2.4 (b) particularly illustrates the correspondence between the voltages.

In order to obtain a system of the form $M_1 I = M_2 V$, equation (2.23) is combined with (2.24) so as to eliminate V'_{ABC}:

$$I_{ABC} = (1 + YZ_2) I'_{ABC} + Y E_{ABC}$$

or equivalently

$$I_{ABC} = -(1 + YZ_2) I_{abc} + Y E_{abc} \tag{2.28}$$

Further substitution of I_n in (2.26) by equation (2.27), then of E_{abc} in (2.28) by the resulting expression leads to the first desired matrix equation, written in terms of the transformer junction currents and voltages only:

$$I_{ABC} + \Big((1 + YZ_2) + YZ_n J_3\Big) I_{abc} = Y V_{abc} \tag{2.29}$$

In similar fashion, the primed components of (2.24) may be eliminated by means of equations (2.22) and the current correspondence relation $I'_{ABC} = -I_{abc}$:

$$\Big((1 + YZ_1) + YZ_N J_3\Big) I_{ABC} + I_{abc} = Y V_{ABC} \tag{2.30}$$

By combining (2.30) and (2.29) into the single matrix equation

$$\begin{bmatrix} (1+YZ_1)+YZ_NJ_3 & I_3 \\ I_3 & (1+YZ_2)+YZ_nJ_3 \end{bmatrix} \begin{bmatrix} I_{ABC} \\ I_{abc} \end{bmatrix} = \begin{bmatrix} YI_3 & 0_3 \\ 0_3 & YI_3 \end{bmatrix} \begin{bmatrix} V_{ABC} \\ V_{abc} \end{bmatrix}$$

the admittance matrix of the transformer can be obtained as described in sec. 2.3.2, that is

$$Y_{Yy0} = \begin{bmatrix} (1+YZ_1)+YZ_NJ_3 & I_3 \\ I_3 & (1+YZ_2)+YZ_nJ_3 \end{bmatrix}^{-1} \begin{bmatrix} YI_3 & 0_3 \\ 0_3 & YI_3 \end{bmatrix}$$

The explicit expressions of the admittance matrix can be worked out directly, and are listed below. For compactness, the assignment $\xi \equiv Z_1 + Z_2 + YZ_1Z_2$ is made.

$$Y_{Yy0} = \begin{bmatrix} a & b & b & c & d & d \\ b & a & b & d & c & d \\ b & b & a & d & d & c \\ c & d & d & e & f & f \\ d & c & d & f & e & f \\ d & d & c & f & f & e \end{bmatrix} \tag{2.31}$$

where

$$a = \frac{1+YZ_2+3YZ_n+2\Big(Z_N+Z_n+3YZ_NZ_n+YZ_2Z_N(2+YZ_2+3YZ_n)\Big)/\xi}{\xi+3\Big(Z_N+Z_n+3YZ_NZ_n+Y(Z_2Z_N+Z_1Z_n)\Big)}$$

$$b = -\frac{Z_N+Z_n+3YZ_NZ_n+YZ_2Z_N(2+YZ_2+3YZ_n)}{\xi\Big(\xi+3\Big(Z_N+Z_n+3YZ_NZ_n+Y(Z_2Z_N+Z_1Z_n)\Big)\Big)}$$

$$c = -\frac{\xi+2\Big(Z_N+Z_n+3YZ_NZ_n+Y(Z_2Z_N+Z_1Z_n)\Big)}{\xi\Big(\xi+3\Big(Z_N+Z_n+3YZ_NZ_n+Y(Z_2Z_N+Z_1Z_n)\Big)\Big)}$$

$$d = \frac{Z_N + Z_n + 3YZ_NZ_n + Y(Z_2Z_N + Z_1Z_n)}{\xi\left(\xi + 3\left(Z_N + Z_n + 3YZ_NZ_n + Y(Z_2Z_N + Z_1Z_n)\right)\right)}$$

$$e = \frac{(1+YZ_1)\left(\xi + 2(Z_N + Z_n + 3YZ_NZ_n + YZ_1Z_n)\right) + YZ_N(Z_1 + 3Z_2 + 3YZ_1Z_2)}{\xi\left(\xi + 3\left(Z_N + Z_n + 3YZ_NZ_n + Y(Z_2Z_N + Z_1Z_n)\right)\right)}$$

$$f = \frac{-(1+YZ_1)(Z_N + Z_n + 3YZ_NZ_n + YZ_1Z_n) + YZ_1Z_N}{\xi\left(\xi + 3\left(Z_N + Z_n + 3YZ_NZ_n + Y(Z_2Z_N + Z_1Z_n)\right)\right)}$$

2.3.5.4 *Yd1* transformer

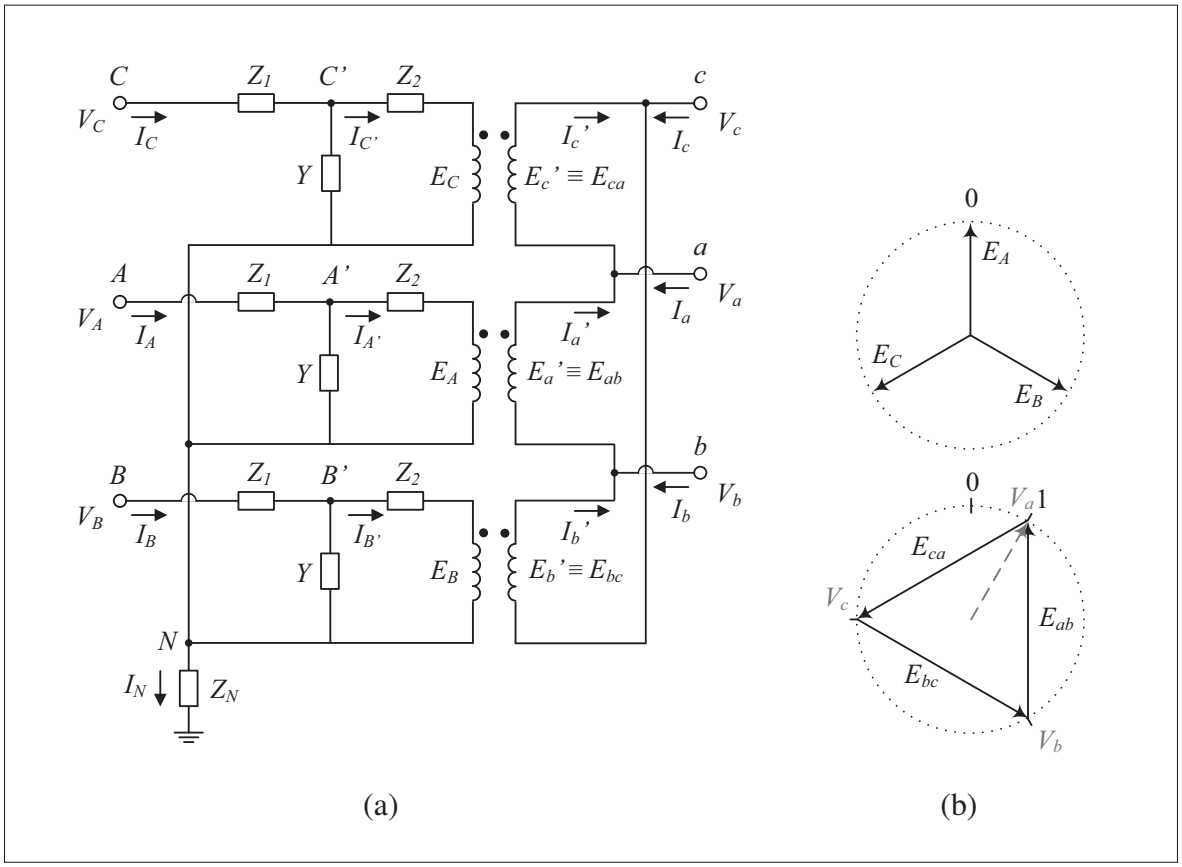


Figure 2.5 Equivalent circuit (a) and phasor representation (b) of the *Yd1* transformer

The circuit used to model the *Yd1* transformer is shown in fig. 2.5 (a). By applying Kirchhoff's laws, the following equations are obtained:

$$V_{ABC} = Z_1 I_{ABC} + V'_{ABC} \tag{2.32}$$

$$V_{ABC} = Z_1 I_{ABC} + Z_2 I'_{ABC} + E_{ABC} + Z_N I_N J_{3\times 1} \tag{2.33}$$

$$I_{ABC} = I'_{ABC} + Y(V'_{ABC} - Z_N I_N J_{3\times 1}) \tag{2.34}$$

$$I_N = I_A + I_B + I_C \tag{2.35}$$

$$(I_3 - P_{(132)})V_{abc} = E'_{abc} \tag{2.36}$$

$$I_{abc} = (P_{(123)} - I_3)I'_{abc} \tag{2.37}$$

In addition, the voltage and current correspondence equations take the forms

$$E_{ABC} = m(V_{abc} - V_{bca}) = m(I_3 - P_{(132)})V_{abc} \tag{2.38}$$

$$I'_{ABC} = \frac{I'_{abc}}{m} \tag{2.39}$$

where m is the per unit transformation ratio.

The value of m can be obtained by seeking a unit ratio between the phase voltage moduli on both sides of the transformer. By considering the first component relation of (2.38), and referring to fig. 2.5 (b), one can write

$$\frac{1}{m}\frac{E_A}{E_A} = \frac{V_a - V_b}{E_A}$$

$$\frac{1}{m} = 1\angle -30^\circ - 1\angle -150^\circ$$

$$m = \frac{1}{\sqrt{3}}$$

With the above equations at hand, a system of the form $M_1 I = M_2 V$ can be laid out. First, substituting V'_{ABC} in (2.34) by (2.32), and multiplying from the left by $m(P_{(123)} - I_3)$ result in

$$m(1+YZ_1)(P_{(123)} - I_3)I_{ABC} - m(P_{(123)} - I_3)I'_{ABC} = mY(P_{(123)} - I_3)V_{ABC}$$

which can be rewritten into

$$m(1+YZ_1)(P_{(123)} - I_3)I_{ABC} - I_{abc} = mY(P_{(123)} - I_3)V_{ABC} \tag{2.40}$$

using (2.37) and (2.39).

Second, starting with equation (2.33), the vector I'_{ABC} can be eliminated by means of the equation that results from the combination of (2.32) and (2.34), i.e.

$$I'_{ABC} = (1+YZ_1)I_{ABC} - YV_{ABC} + YZ_N I_N J_{3\times 1}$$

and E_{ABC} can be replaced by $m(I_3 - P_{(132)})V_{abc}$ according to (2.38). After rearranging the terms, it follows that

$$\Big((Z_1 + Z_2 + YZ_1Z_2) + (1+YZ_2)Z_N J_3\Big)I_{ABC} = (1+YZ_2)V_{ABC} - m(I_3 - P_{(132)})V_{abc} \tag{2.41}$$

Analogously as in the previous section, equations (2.41) and (2.40) can be combined into a single matrix equation of the form $M_1 I = M_2 V$, where

$$M_1 = \begin{bmatrix} \Big((Z_1 + Z_2 + YZ_1Z_2)I_3 + (1+YZ_2)Z_N J_3\Big) & 0_3 \\ m(1+YZ_1)(P_{(123)} - I_3) & -I_3 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} (1+YZ_2)I_3 & -m(I_3 - P_{(132)}) \\ mY(P_{(123)} - I_3) & 0_3 \end{bmatrix}$$

The explicit admittance matrix formulae result from the product $Y_{Yd1} = M_1^{-1} M_2$, and are given below (with m replaced by $1/\sqrt{3}$).

$$Y_{Yd1} = \begin{bmatrix} a & b & b & -c & c & 0 \\ b & a & b & 0 & -c & c \\ b & b & a & c & 0 & -c \\ -c & 0 & c & d & e & e \\ c & -c & 0 & e & d & e \\ 0 & c & -c & e & e & d \end{bmatrix} \tag{2.42}$$

where

$$a = \frac{(1+YZ_2)\Big(Z_1+Z_2+YZ_1Z_2+2Z_N(1+YZ_2)\Big)}{(Z_1+Z_2+YZ_1Z_2)\Big(Z_1+Z_2+YZ_1Z_2+3Z_N(1+YZ_2)\Big)}$$

$$b = \frac{-Z_N(1+YZ_2)^2}{(Z_1+Z_2+YZ_1Z_2)\Big(Z_1+Z_2+YZ_1Z_2+3Z_N(1+YZ_2)\Big)}$$

$$c = \frac{1}{\sqrt{3}(Z_1+Z_2+YZ_1Z_2)}$$

$$d = \frac{2(1+YZ_1)}{3(Z_1+Z_2+YZ_1Z_2)}$$

$$e = \frac{-(1+YZ_1)}{3(Z_1+Z_2+YZ_1Z_2)}$$

2.3.5.5 *Dd0* transformer

The circuit used to model the delta-delta transformer is shown in fig. 2.6 (a). By applying Kirchhoff's laws, the following equations can be obtained:

$$V_{ABC} - V'_{ABC} = Z_1 I''_{ABC} \tag{2.43}$$

$$V'_{ABC} - P_{(132)} V_{ABC} = Z_2 I'_{ABC} + E'_{ABC} \tag{2.44}$$

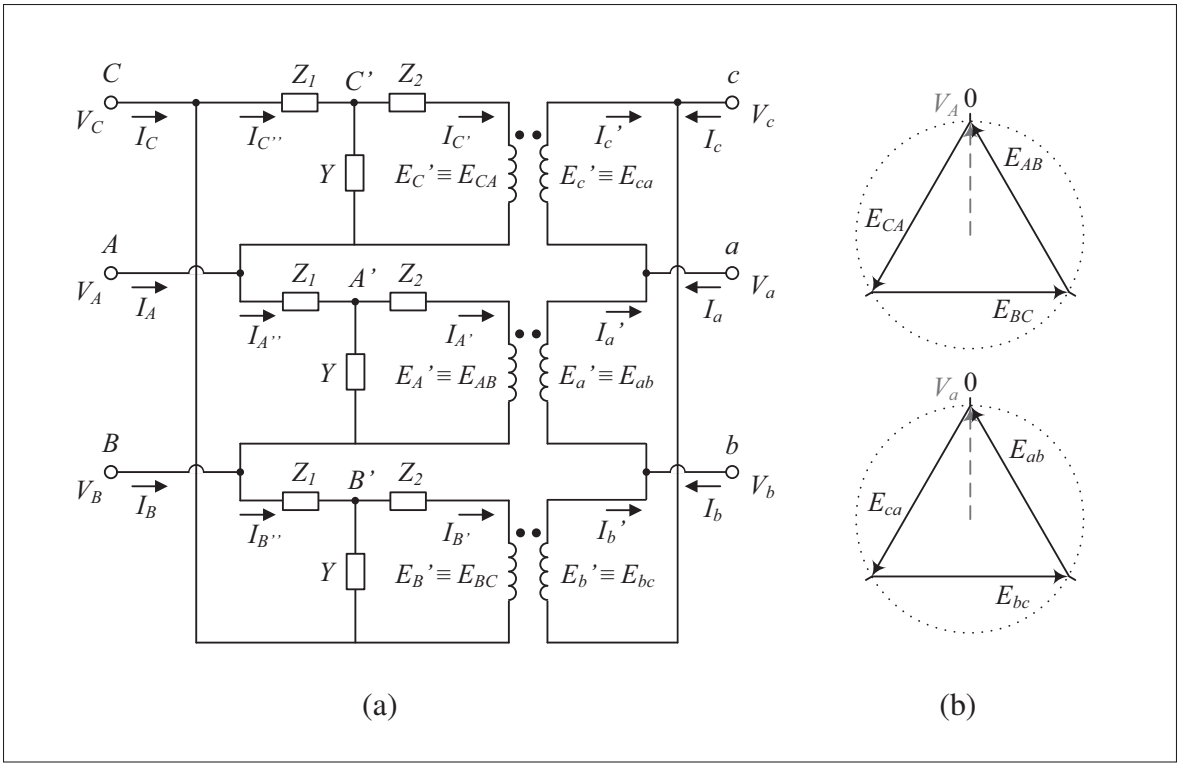


Figure 2.6 Equivalent circuit (a) and phasor representation (b) of the *Dd0* transformer

$$I_{ABC} = (I_3 - P_{(123)})I''_{ABC} \tag{2.45}$$

$$I''_{ABC} = I'_{ABC} + Y(V'_{ABC} - P_{(132)}V_{ABC}) \tag{2.46}$$

$$(I_3 - P_{(132)})V_{abc} = E'_{abc} \tag{2.47}$$

$$I_{abc} = (P_{(123)} - I_3)I'_{abc} \tag{2.48}$$

Using the same reasoning as in the previous sections, it is straightforward to show that $E'_{ABC} = E'_{abc}$ and $I'_{ABC} = I'_{abc}$ in the per unit system. In particular, fig. 2.6 (b) shows the relationships between the voltage phasors on both sides of the ideal transformer.

The first equation of the system of the form $M_1 I = M_2 V$ is determined in a few steps. Considering the equivalences $E'_{ABC} = E'_{abc}$ and $I'_{ABC} = I'_{abc}$, substitution of E'_{ABC} in (2.44) by (2.47),

followed by left multiplication of $(P_{(123)} - I_3)$ to get rid of I'_{abc} through (2.48), leads to

$$(P_{(123)} - I_3)V'_{ABC} - (I_3 - P_{(132)})V_{ABC} = Z_2 I_{abc} + (J_3 - 3I_3)V_{abc} \tag{2.49}$$

Moreover, (2.43) and (2.45) can be combined, yielding

$$(P_{(123)} - I_3)V'_{ABC} = Z_1 I_{ABC} + (P_{(123)} - I_3)V_{ABC} \tag{2.50}$$

which can be used to replace V'_{ABC} in (2.49) to give the first significant equation:

$$Z_1 I_{ABC} - Z_2 I_{abc} = (3I_3 - J_3)V_{ABC} + (J_3 - 3I_3)V_{abc} \tag{2.51}$$

Next, (2.46) is mutiplied from the left by $I_3 - P_{(123)}$, and (2.45), (2.48), and (2.50) are used to eliminate respectively the doubly and singly primed components. The result is the second significant equation:

$$(1 + YZ_1)I_{ABC} + I_{abc} = Y(3I_3 - J_3)V_{ABC} \tag{2.52}$$

The matrices M_1 and M_2 can be constructed directly through the combination of (2.51) and (2.52), that is

$$M_1 = \begin{bmatrix} Z_1 I_3 & -Z_2 I_3 \\ (1 + YZ_1)I_3 & I_3 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} 3I_3 - J_3 & J_3 - 3I_3 \\ Y(3I_3 - J_3) & 0_3 \end{bmatrix}$$

and the admittance matrix of the branch can be calculated as prescribed in sec. 2.3.2. Its explicit contents are given below:

$$Y_{Dd0} = \begin{bmatrix} a & b & b & c & d & d \\ b & a & b & d & c & d \\ b & b & a & d & d & c \\ c & d & d & e & f & f \\ d & c & d & f & e & f \\ d & d & c & f & f & e \end{bmatrix} \tag{2.53}$$

where

$$a = \frac{2(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad d = \frac{1}{Z_1+Z_2+YZ_1Z_2}$$
$$b = \frac{-(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad e = \frac{2(1+YZ_1)}{Z_1+Z_2+YZ_1Z_2}$$
$$c = \frac{-2}{Z_1+Z_2+YZ_1Z_2} \qquad f = \frac{-(1+YZ_1)}{Z_1+Z_2+YZ_1Z_2}$$

2.3.5.6 *Yz11* transformer

The circuit used to model the star-zigzag transformer of the type *Yz11* is shown in fig. 2.7 (a). By applying Kirchhoff's laws, the following equations can be obtained:

$$V_{ABC} = Z_1 I_{ABC} + V'_{ABC} \tag{2.54}$$
$$V'_{ABC} = Z_2 I'_{ABC} + E_{ABC} + Z_N I_N J_{3\times1} \tag{2.55}$$
$$I_{ABC} = I'_{ABC} + Y(V'_{ABC} - Z_N I_N J_{3\times1}) \tag{2.56}$$
$$I_N = I_A + I_B + I_C \tag{2.57}$$
$$V_{abc} = [E_{ax} - E_{nx}, E_{by} - E_{ny}, E_{cz} - E_{nz}]^T + Z_n I_n J_{3\times1} \tag{2.58}$$
$$I_n = I_a + I_b + I_c \tag{2.59}$$

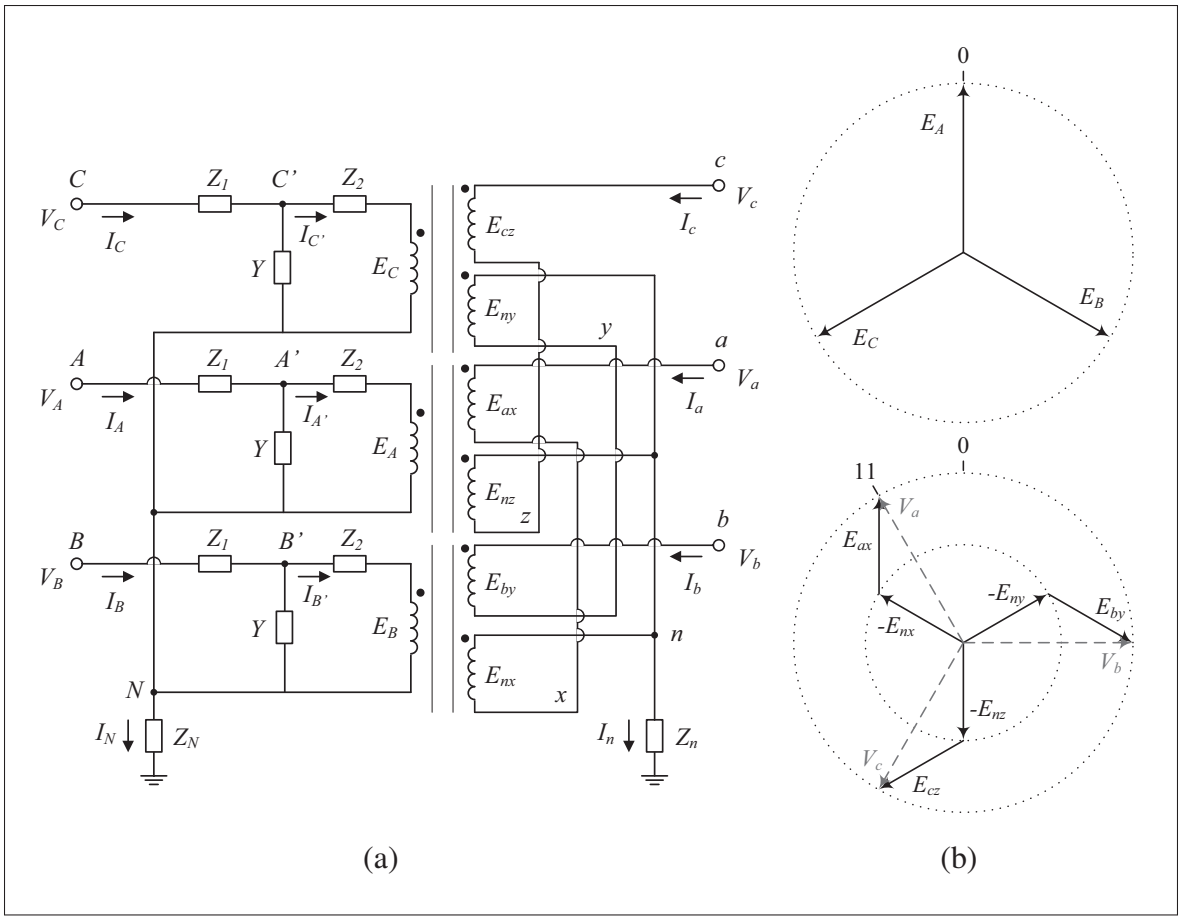


Figure 2.7 Equivalent circuit (a) and phasor representation (b) of the *Yz11* transformer

In addition, based on basic transformer principles[2], the correspondence relations between the voltages and currents on the primary and secondary sides can be written as follows:

$$E_{ABC} = m \begin{bmatrix} E_{ax} \\ E_{by} \\ E_{cz} \end{bmatrix} = m \begin{bmatrix} E_{nz} \\ E_{nx} \\ E_{ny} \end{bmatrix} \tag{2.60}$$

$$I'_{ABC} = \frac{1}{m}(I_{cab} - I_{abc}) = \frac{1}{m}(P_{(123)} - I_3)I_{abc} \tag{2.61}$$

[2] See, for instance, pp. 97–101 of Slemon and Straughen (1982).

where an equal number of turns in each of the secondary windings is assumed. The value of the per unit transformation ratio m can be obtained with the help of (2.58), and by imposing the requirement that phase voltages on both sides of the transformer have the same moduli. Thus,

$$\begin{aligned}\frac{V_a}{E_A} &= \frac{E_{ax}-E_{nx}}{E_A}\\ &= \frac{1}{m}(1-\frac{E_B}{E_A})\\ &= \frac{1}{m}(1-\angle -120^\circ) = \frac{\sqrt{3}}{m}\angle 30^\circ\end{aligned}$$

such that $m=\sqrt{3}$.

A sufficient number of equations have now been gathered to proceed to the derivation of the matrices M_1 and M_2. First, by substituting V'_{ABC} and I_N in (2.56) using (2.54) and (2.57) respectively, and by using (2.61) to eliminate I'_{ABC}, one obtains

$$\Big((1+YZ_1)+YZ_NJ_3\Big)I_{ABC}+\frac{1}{m}(I_3-P_{(123)})I_{abc}=YV_{ABC} \tag{2.62}$$

Second, equation (2.58) can be written in a more convenient way using (2.60) and (2.59) in the form $I_nJ_{3\times 1}=J_3I_{abc}$, that is

$$V_{abc}=\frac{1}{m}(I_3-P_{(132)})E_{ABC}+Z_nJ_3I_{abc} \tag{2.63}$$

Moreover, substitution of $V'_{ABC}-Z_NI_NJ_{3\times 1}$ in (2.55) by means of (2.56), followed by the use of expression (2.61) to get rid of I'_{ABC}, yields

$$I_{ABC}+\frac{1}{m}(1+YZ_2)(I_3-P_{(123)})I_{abc}=YE_{ABC} \tag{2.64}$$

Equation (2.64) can then be used to substitute E_{ABC} in (2.63) by an expression composed of I_{ABC} and I_{abc}, resulting in

$$\frac{1}{m}(I_3 - P_{(132)})I_{ABC} + \left(\frac{1}{m^2}(1+YZ_2)(3I_3 - J_3) + YZ_nJ_3\right)I_{abc} = YV_{abc} \tag{2.65}$$

Finally, by combining (2.62) and (2.65) into the single matrix equation $M_1 I = M_2 V$, it is straightforward to extract the matrices M_1 and M_2:

$$M_1 = \begin{bmatrix} (1+YZ_1)I_3 + YZ_NJ_3 & (I_3 - P_{(123)})/m \\ (I_3 - P_{(132)})/m & (1+YZ_2)(3I_3 - J_3)/m^2 + YZ_nJ_3 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} YI_3 & 0_3 \\ 0_3 & YI_3 \end{bmatrix}$$

Solving explicitly for $Y_{Yz11} = M_1^{-1}M_2$ with m set to $\sqrt{3}$ yields the following expressions:

$$Y_{Yz11} = \begin{bmatrix} a & b & b & -c & 0 & c \\ b & a & b & c & -c & 0 \\ b & b & a & 0 & c & -c \\ -c & c & 0 & d & e & e \\ 0 & -c & c & e & d & e \\ c & 0 & -c & e & e & d \end{bmatrix} \tag{2.66}$$

where

$$a = \frac{2 + 3Y(Z_1 + Z_2 + YZ_1Z_2) + 6YZ_N(1+YZ_2)}{3(Z_1 + Z_2 + YZ_1Z_2)\left(1 + Y(Z_1 + 3Z_N)\right)} \qquad d = \frac{1}{9Z_n} + \frac{2(1+YZ_1)}{3(Z_1 + Z_2 + YZ_1Z_2)}$$

$$b = -\frac{1 + 3YZ_N(1+YZ_2)}{3(Z_1 + Z_2 + YZ_1Z_2)\left(1 + Y(Z_1 + 3Z_N)\right)} \qquad e = \frac{1}{9Z_n} - \frac{1+YZ_1}{3(Z_1 + Z_2 + YZ_1Z_2)}$$

$$c = \frac{1}{\sqrt{3}(Z_1 + Z_2 + YZ_1Z_2)}$$

2.3.5.7 *Dz0* transformer

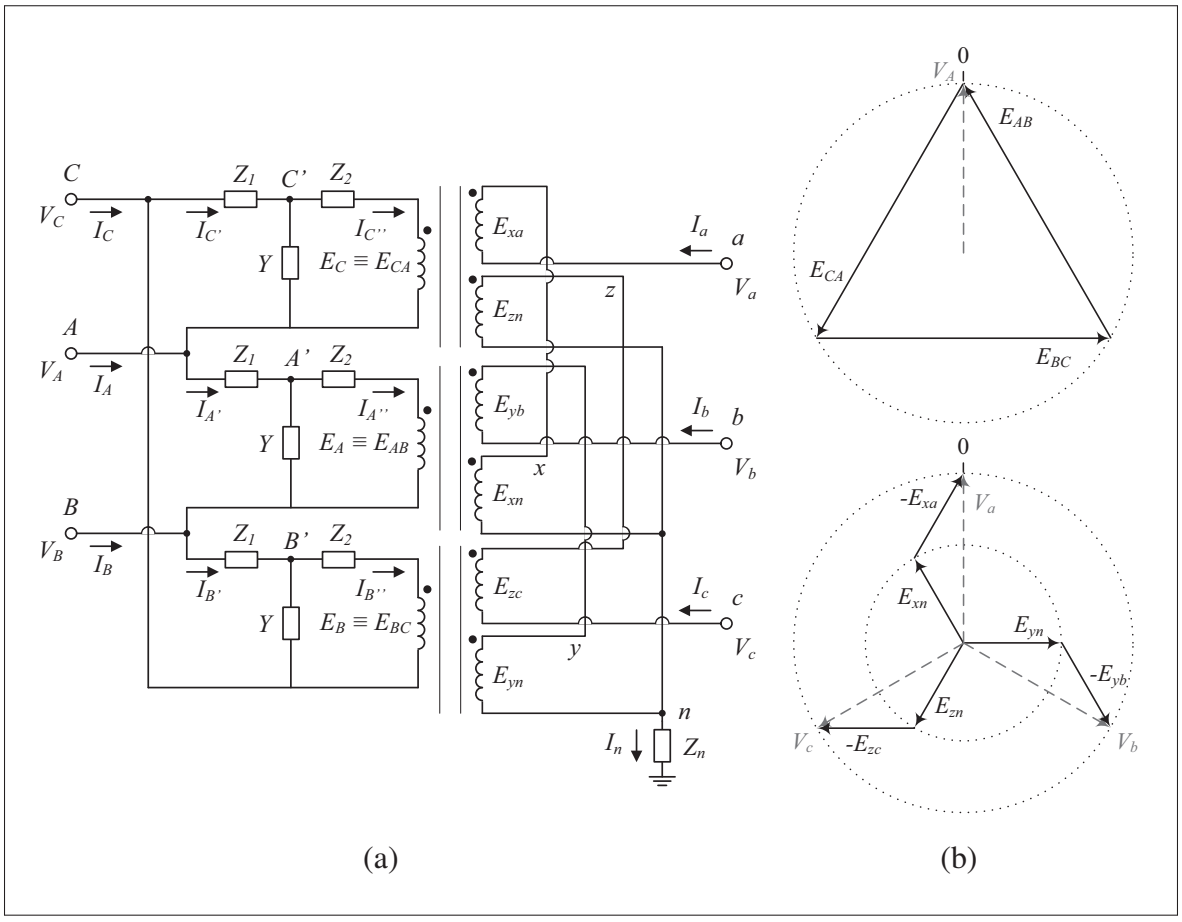


Figure 2.8 Equivalent circuit (a) and phasor representation (b) of the *Dz0* transformer

The circuit used to model the delta-zigzag transformer of the type *Dz0* is shown in fig. 2.8 (a). The following equations can be obtained through the use of Kirchhoff's laws:

$$V_{ABC} = Z_1 I'_{ABC} + V'_{ABC} \tag{2.67}$$

$$V'_{ABC} - P_{(132)} V_{ABC} = Z_2 I''_{ABC} + E_{ABC} \tag{2.68}$$

$$I_{ABC} = (I_3 - P_{(123)}) I'_{ABC} \tag{2.69}$$

$$I'_{ABC} = I''_{ABC} + Y(V'_{ABC} - P_{(132)} V_{ABC}) \tag{2.70}$$

$$V_{abc} = [-E_{xa} + E_{xn}, -E_{yb} + E_{yn}, -E_{zc} + E_{zn}]^T + Z_n I_n J_{3\times 1} \tag{2.71}$$

$$I_n = I_a + I_b + I_c \tag{2.72}$$

In addition, as mentioned in the previous section, the correspondence relations between the voltages and currents on the primary and secondary sides can be deduced from basic transformer principles:

$$E_{ABC} = m \begin{bmatrix} E_{yb} \\ E_{zc} \\ E_{xa} \end{bmatrix} = m \begin{bmatrix} E_{xn} \\ E_{yn} \\ E_{zn} \end{bmatrix} \tag{2.73}$$

$$I''_{ABC} = \frac{1}{m}(I_{bca} - I_{abc}) = \frac{1}{m}(P_{(132)} - I_3)I_{abc} \tag{2.74}$$

where an equal number of turns in the secondary windings is assumed. By using (2.71), and imposing the requirement that line voltages on both sides of the ideal transformer have the same moduli, the value of m is obtained:

$$\begin{aligned} V_a - V_b &= -E_{xa} + E_{xn} + E_{yb} - E_{yn} \\ &= \frac{1}{m}(2E_A - E_B - E_C) \\ \frac{V_a - V_b}{E_A} &= \frac{1}{m}(2 - \angle -120^\circ - \angle 120^\circ) = \frac{3}{m} \\ m &= 3 \end{aligned}$$

The threefold relationship expressed by (2.73) is illustrated in fig. 2.8 (b).

The next step is to determine the matrices M_1 and M_2. Starting with equation (2.70), I''_{ABC} is replaced by its equivalent expression in (2.74):

$$I'_{ABC} = \frac{1}{m}(P_{(132)} - I_3)I_{abc} + Y(V'_{ABC} - P_{(132)}V_{ABC})$$

Then, substitution of V'_{ABC} using (2.67) leads to

$$I'_{ABC} = \frac{1}{1+YZ_1}\left(\frac{1}{m}(P_{(132)} - I_3)I_{abc} + Y(I_3 - P_{(132)})V_{ABC}\right) \tag{2.75}$$

In order to obtain the first usable equation, there remains to multiply (2.75) from the left by $(I_3 - P_{(123)})$, and to substitute I'_{ABC} by means of (2.69). Consequently,

$$I_{ABC} + \frac{3I_3 - J_3}{m(1+YZ_1)}I_{abc} = \frac{Y}{1+YZ_1}(3I_3 - J_3)V_{ABC} \tag{2.76}$$

To complete the system $M_1 I = M_2 V$, (2.71) is simplified by means of the relations described in (2.73):

$$V_{abc} = \frac{1}{m}(I_3 - P_{(123)})E_{ABC} + Z_n I_n J_{3\times 1} \tag{2.77}$$

Moreover, substitution of I''_{ABC} in (2.68) by means of (2.74) gives

$$V'_{ABC} - P_{(132)}V_{ABC} = \frac{Z_2}{m}(P_{(132)} - I_3)I_{abc} + E_{ABC} \tag{2.78}$$

Equations (2.78) and (2.77) can be combined to eliminate E_{ABC}, that is

$$V_{abc} = \frac{1}{m}(I_3 - P_{(123)})\left(V'_{ABC} - P_{(132)}V_{ABC} + \frac{Z_2}{m}(I_3 - P_{(132)})I_{abc}\right) + Z_n I_n J_{3\times 1} \tag{2.79}$$

In addition, equations (2.67) and (2.69) can be combined to obtain the relation

$$(I_3 - P_{(123)})V'_{ABC} = (I_3 - P_{(123)})V_{ABC} - Z_1 I_{ABC}$$

which allows to replace V'_{ABC} in (2.79) with an expression written in terms of the desired vectors V_{ABC} and I_{ABC} only. The second usable equation thus follows:

$$Z_1 I_{ABC} + \left(\left(\frac{Z_2}{m} - mZ_n\right)J_3 - \frac{3Z_2}{m}I_3\right)I_{abc} = (3I_3 - J_3)V_{ABC} - mV_{abc} \tag{2.80}$$

Based on equations (2.76) and (2.80), the matrices M_1 and M_2 can be written directly, i.e.

$$M_1 = \begin{bmatrix} I_3 & (3I_3 - J_3)/(m(1+YZ_1)) \\ Z_1 & (Z_2/m - mZ_n)J_3 - 3Z_2I_3/m \end{bmatrix}$$

$$M_2 = \begin{bmatrix} Y(3I_3 - J_3)/(1+YZ_1) & 0_3 \\ 3I_3 - J_3 & -mI_3 \end{bmatrix}$$

and the explicit expressions of Y_{Dz0} can be derived as stated in sec. 2.3.2. By letting $m = 3$, Y_{Dz0} takes the following form:

$$Y_{Dz0} = \begin{bmatrix} a & b & b & c & d & d \\ b & a & b & d & c & d \\ b & b & a & d & d & c \\ c & d & d & e & f & f \\ d & c & d & f & e & f \\ d & d & c & f & f & e \end{bmatrix} \tag{2.81}$$

where

$$a = \frac{2(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad d = \frac{1}{Z_1+Z_2+YZ_1Z_2}$$

$$b = \frac{-(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad e = \frac{1}{9Z_n} + \frac{2(1+YZ_1)}{Z_1+Z_2+YZ_1Z_2}$$

$$c = \frac{-2}{Z_1+Z_2+YZ_1Z_2} \qquad f = \frac{1}{9Z_n} - \frac{1+YZ_1}{Z_1+Z_2+YZ_1Z_2}$$

2.3.5.8 *Dz10* transformer

The circuit used to model the delta-zigzag transformer of the type *Dz10* is shown in fig. 2.9. The equations resulting from Kirchhoff's laws are the same as with the *Dz0* transformer, due to

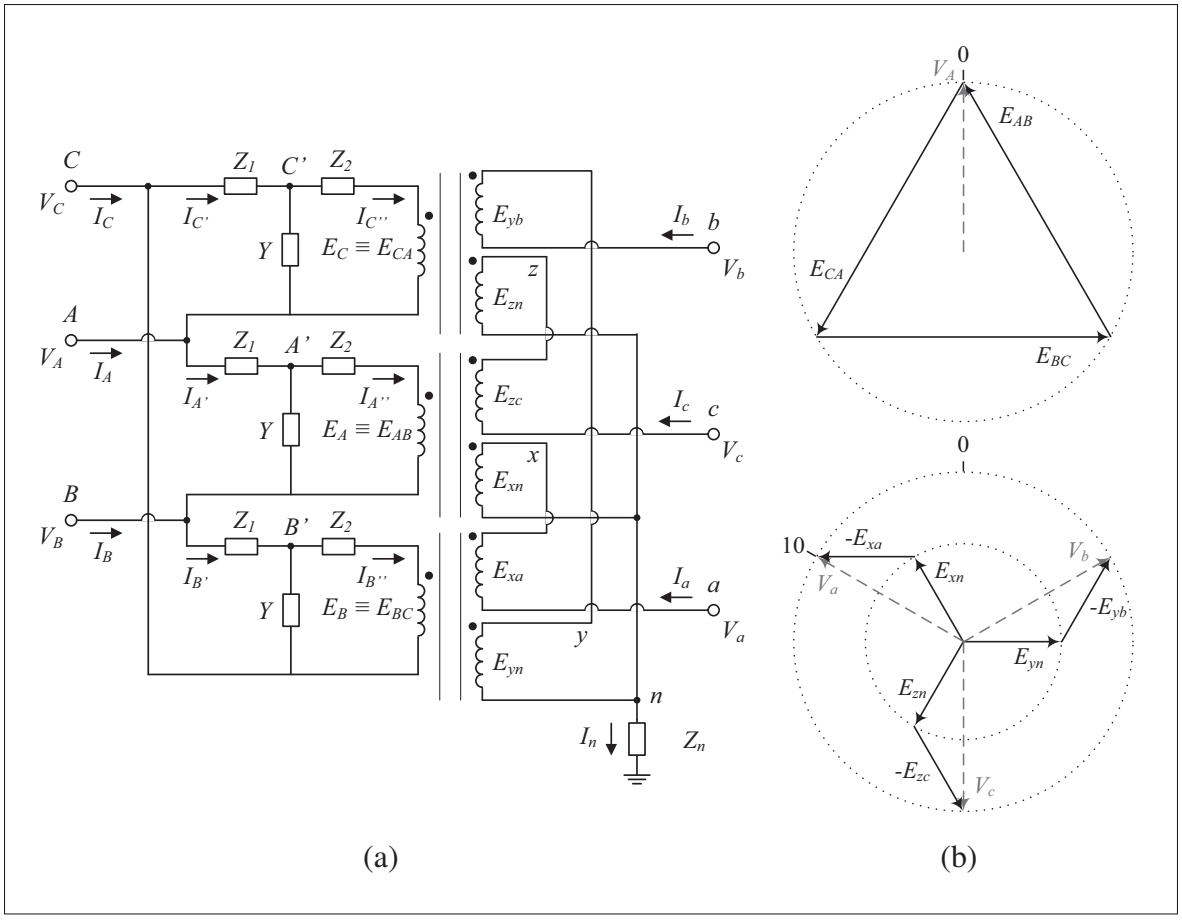


Figure 2.9 Equivalent circuit (a) and phasor representation (b) of the *Dz10* transformer

the unchanged topology on the primary side, and consistent phase labelling on the secondary side. Yet, the different coil connections lead to the distinct correspondence equations:

$$E_{ABC} = m \begin{bmatrix} E_{zc} \\ E_{xa} \\ E_{yb} \end{bmatrix} = m \begin{bmatrix} E_{xn} \\ E_{yn} \\ E_{zn} \end{bmatrix} \tag{2.82}$$

$$I''_{ABC} = \frac{1}{m}(I_{cab} - I_{abc}) = \frac{1}{m}(P_{(123)} - I_3)I_{abc} \tag{2.83}$$

The per unit transformation ratio can be determined similarly as with the *Dz0* transformer:

$$\begin{aligned} V_a - V_b &= -E_{xa} + E_{xn} + E_{yb} - E_{yn} \\ &= \frac{1}{m}(E_A - 2E_B + E_C) \\ \frac{V_a - V_b}{E_A} &= \frac{1}{m}(1 + 2\angle 60^\circ + \angle 120^\circ) = \frac{3}{m}\angle 60^\circ \end{aligned}$$

Hence it is straightforward to see that a unit ratio between the secondary and primary line voltages will be obtained only if $m = 3$. As in the previous section, fig. 2.9 (b) reflects this factor of 3 as well as the phase shifts between the various voltages defined in the transformer.

The first equation of the system $M_1 I = M_2 V$ can be obtained by following the steps described in sec. 2.3.5.7 together with the equations (2.82) and (2.83) derived above:

$$I_{ABC} + \frac{J_3 - 3P_{(123)}}{m(1 + YZ_1)} I_{abc} = \frac{Y}{1 + YZ_1}(3I_3 - J_3)V_{ABC} \tag{2.84}$$

Likewise, the second equation is obtained as prescribed in sec. 2.3.5.7, except that due to the different primary versus secondary voltage and current relations, the equation resulting from the combination of (2.67) and (2.69) needs to be multiplied from the left by $-P_{(132)}$ in order to proceed to the substitution of V'_{ABC}. Hence,

$$\frac{Z_1}{m} P_{(132)} I_{ABC} + \left(\frac{3Z_2}{m^2} I_3 + \left(Z_n - \frac{Z_2}{m^2} \right) J_3 \right) I_{abc} = \frac{1}{m}(3P_{(132)} - J_3)V_{ABC} + V_{abc} \tag{2.85}$$

From (2.84) and (2.85), the matrices M_1 and M_2 can be constructed directly, that is

$$M_1 = \begin{bmatrix} I_3 & (J_3 - 3P_{(123)})/(m(1 + YZ_1)) \\ Z_1 P_{(132)}/m & 3Z_2 I_3/m^2 + (Z_n - Z_2/m^2)J_3 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} Y(3I_3 - J_3)/(1 + YZ_1) & 0_3 \\ (3P_{(132)} - J_3)/m & I_3 \end{bmatrix}$$

and the explicit expressions of Y_{Dz10} can be derived as stated in sec. 2.3.2. By letting $m = 3$, Y_{Dz10} takes the following form:

$$Y_{Dz10} = \begin{bmatrix} a & b & b & c & c & d \\ b & a & b & d & c & c \\ b & b & a & c & d & c \\ c & d & c & e & f & f \\ c & c & d & f & e & f \\ d & c & c & f & f & e \end{bmatrix} \tag{2.86}$$

where

$$a = \frac{2(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad d = \frac{2}{Z_1+Z_2+YZ_1Z_2}$$

$$b = \frac{-(1+YZ_2)}{Z_1+Z_2+YZ_1Z_2} \qquad e = \frac{1}{9Z_n} + \frac{2(1+YZ_1)}{Z_1+Z_2+YZ_1Z_2}$$

$$c = \frac{-1}{Z_1+Z_2+YZ_1Z_2} \qquad f = \frac{1}{9Z_n} - \frac{1+YZ_1}{Z_1+Z_2+YZ_1Z_2}$$

2.3.6 The tap changer and other transformer generalisations

2.3.6.1 Single-phase transformer

By introducing the transformation ratio a_1, as shown in fig. 2.10, the applicability of the single-phase transformer model can be enlarged to include the effect of a tap changer. Variations in a_1 can thus be effected in order to meet a voltage constraint imposed on either side of the transformer. As regards the ratio a_2, it is fixed, and it is intended to facilitate the treatment of cases where the secondary voltage differs from its corresponding base voltage.

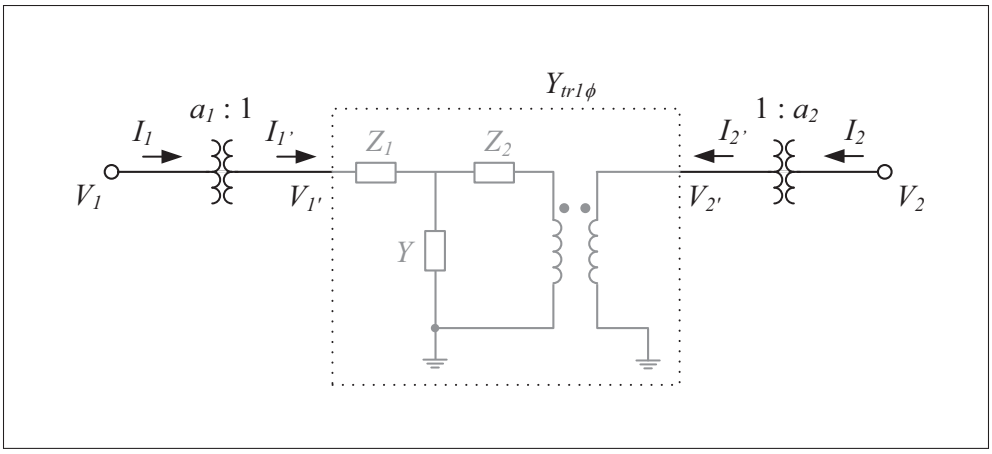


Figure 2.10 Approximate representation of a tap changer operating on a single-phase transformer described by the admittance matrix $Y_{tr1\phi}$

Assuming an actual tap position n_x, a total number of taps n_t, and limiting transformation ratios $a_{1\,min}$ and $a_{1\,max}$, the actual transformation ratio follows from the relation

$$a_1(n_x) = \frac{n_x - 1}{n_t - 1}(a_{1\,max} - a_{1\,min}) + a_{1\,min} \tag{2.87}$$

Moreover, the step in transformation ratio caused by a single tap increase is given by

$$\Delta a_1 = \frac{a_{1\,max} - a_{1\,min}}{n_t - 1} \tag{2.88}$$

The effect of introducing a_1 and a_2 on the branch admittance matrix can be investigated as done before, i.e. by seeking a system of equations of the form $I = YV$. The additional factors imply the following relations:

$$V_1 = a_1 V_{1'} \qquad V_2 = a_2 V_{2'}$$

$$I_1 = \frac{1}{a_1} I_{1'} \qquad I_2 = \frac{1}{a_2} I_{2'}$$

Furthermore, by definition

$$I' = Y_{tr1\phi} V'$$

$$\begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = Y_{tr1\phi} \begin{bmatrix} 1/a_1 & 0 \\ 0 & 1/a_2 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

such that

$$\begin{aligned} Y_{br} &= \begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix}^{-1} Y_{tr1\phi} \begin{bmatrix} 1/a_1 & 0 \\ 0 & 1/a_2 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{a_1^2}(Y_{tr1\phi})_{11} & \frac{1}{a_1 a_2}(Y_{tr1\phi})_{12} \\ \frac{1}{a_2 a_1}(Y_{tr1\phi})_{21} & \frac{1}{a_2^2}(Y_{tr1\phi})_{22} \end{bmatrix} \end{aligned} \tag{2.89}$$

The effects of a change in a_1 on V_1 and V_2 depend on the transformer characteristics and particular surrounding network components and their topology. Nevertheless, by neglecting the effect of the transformer winding impedances and magnetizing admittance, one can see that $V_2 \approx V_1/a_1$. Thus, given that V_2 has to be maintained at its current value, a decrease (increase) in V_1 would demand a corresponding decrease (increase) in a_1. Similarly, assuming that V_2 needs to be decreased (increased) for a given V_1, then a_1 would have to be increased (decreased) accordingly.

2.3.6.2 Three-phase transformer

As in the single-phase case, the three-phase transformer models are generalised through the introduction of the transformation ratios a_1 and a_2 (see fig. 2.11). The inner block represents a bare three-phase transformer. The tap changer mechanism, for instance implemented on the transformers of types *Yy0* and *Yd1*, is reflected by changes in a_1, where the formulae (2.87) and (2.88) remain applicable. Changes in a_1 also lead to changes in the primary and secondary voltages in a way similar to the single-phase case. The ratio a_2 is fixed, and is intended to

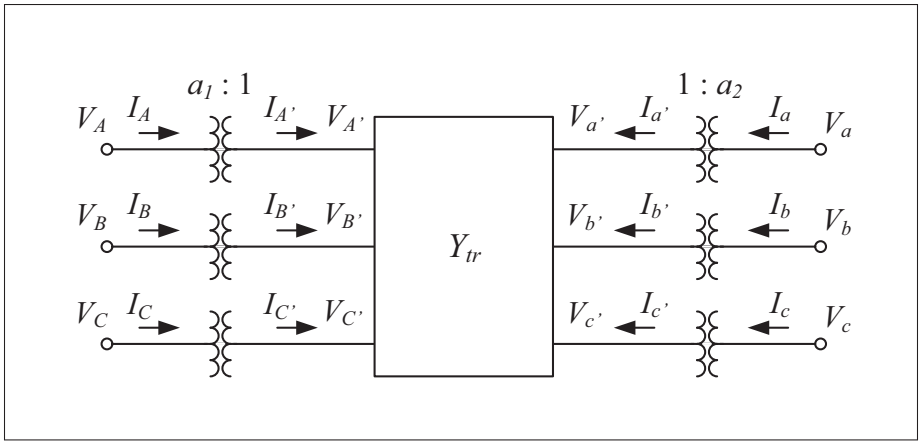


Figure 2.11 Approximate representation of a tap changer operating on a three-phase transformer described by the admittance matrix Y_{tr}

facilitate the treatment of the cases where the secondary voltage differs from its corresponding base voltage.

The effect of the factors a_1 and a_2 on Y_{tr} can be clarified by seeking a system of equations of the form $M_1 I = M_2 V$. First, a_1 and a_2 relate the voltages and currents as follows:

$$V_{ABC} = a_1 V'_{ABC} \qquad V_{abc} = a_2 V'_{abc}$$

$$I_{ABC} = \frac{1}{a_1} I'_{ABC} \qquad I_{abc} = \frac{1}{a_2} I'_{abc}$$

Second, Y_{tr} is defined in such a way that $I' = Y_{tr} V'$, where the vectors I' and V' are constructed in accordance with sec. 2.3.3. Therefore,

$$I' = Y_{tr} V'$$

$$\begin{bmatrix} a_1 I_3 & 0_3 \\ 0_3 & a_2 I_3 \end{bmatrix} I = Y_{tr} \begin{bmatrix} I_3/a_1 & 0_3 \\ 0_3 & I_3/a_2 \end{bmatrix} V$$

from which the overall admittance matrix can be deduced directly, that is

$$Y_{br} = \begin{bmatrix} a_1 I_3 & 0_3 \\ 0_3 & a_2 I_3 \end{bmatrix}^{-1} Y_{tr} \begin{bmatrix} I_3/a_1 & 0_3 \\ 0_3 & I_3/a_2 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{1}{(a_1)^2}(Y_{tr})_{KL} & \frac{1}{a_1 a_2}(Y_{tr})_{K\ell} \\ \frac{1}{a_2 a_1}(Y_{tr})_{kL} & \frac{1}{(a_2)^2}(Y_{tr})_{k\ell} \end{bmatrix} \tag{2.90}$$

where each element shown in the matrix is itself a 3×3 matrix with $K, L \in \{A, B, C\}$ and $k, \ell \in \{a, b, c\}$.

2.3.7 The zigzag earthing transformer

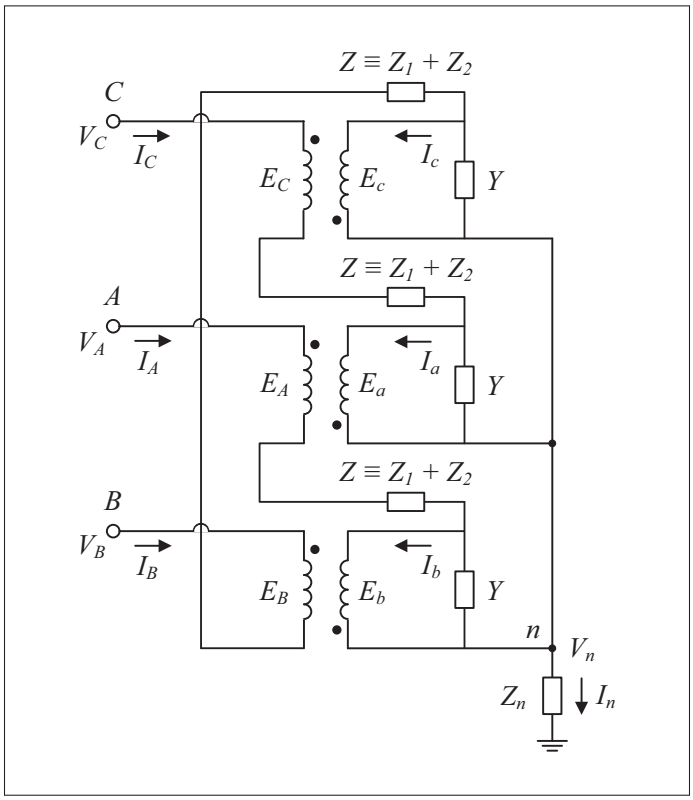


Figure 2.12 Equivalent circuit of the zigzag earthing transformer

The circuit used to model the zigzag earthing transformer is shown in fig. 2.12. The primary (Z_1) and secondary (Z_2) coil impedances are in series, and are merged into the single impedance $Z = Z_1 + Z_2$. By applying Kirchhoff's laws, using the notation of sec. 2.3.3, the following relations are obtained:

$$V_{ABC} = E_{ABC} + ZI_{ABC} - P_{(132)}E_{abc} + Z_n I_n J_{3\times 1} \tag{2.91}$$

$$I_{ABC} = P_{(132)}I_{abc} - YP_{(132)}E_{abc} \tag{2.92}$$

$$I_n = I_A + I_B + I_C \tag{2.93}$$

Furthermore, given an equal number of turns in the primary and secondary coils, and taking into consideration the indicated polarities in fig. 2.12,

$$E_{ABC} = E_{abc} \tag{2.94}$$

$$I_{ABC} = I_{abc} \tag{2.95}$$

The objective is to rearrange the above equations so as to have a system of the form $I_{ABC} = Y_{tr}V_{ABC}$, from which Y_{tr} can be extracted. First, equation (2.94) can be used to simplify (2.91), that is

$$V_{ABC} = (I_3 - P_{(132)})E_{abc} + (ZI_3 + Z_n J_3)I_{ABC} \tag{2.96}$$

Second, multiplying (2.92) from the left by $-P_{(123)}/Y$, and substituting I_{abc} according to (2.95) yield

$$E_{abc} = \frac{1}{Y}(I_3 - P_{(123)})I_{ABC} \tag{2.97}$$

Finally, replacing E_{abc} in (2.96) by the expression (2.97) leads to

$$V_{ABC} = \left(\left(Z + \frac{3}{Y}\right)I_3 + \left(Z_n - \frac{1}{Y}\right)J_3\right)I_{ABC}$$

such that

$$Y_{zn} = \left(\left(Z + \frac{3}{Y}\right)I_3 + \left(Z_n - \frac{1}{Y}\right)J_3\right)^{-1}$$

The explicit computation of the inverse thus results in the following matrix:

$$Y_{zn} = \begin{bmatrix} a & b & b \\ b & a & b \\ b & b & a \end{bmatrix} \tag{2.98}$$

where

$$a = \frac{1 + Y(Z + 2Z_n)}{(3 + YZ)(Z + 3Z_n)}$$
$$b = \frac{1 - YZ_n}{(3 + YZ)(Z + 3Z_n)}$$

2.4 The classic model

The classic model constitutes a basic approach to perform power flow studies of electrical networks. It consists in applying Newton's method to the power flow problem with a single swing bus, in line with Wood and Wollenberg (1996) among others. Its simplicity arises from the assumption that the network frequency keeps its nominal value, due the ability of one of its buses, namely the swing bus (also known as slack bus), to generate or absorb power at will so as to fulfill the load conditions and power constraints of the network.

2.4.1 Load model

In a way similar to Okamura *et al.* (1975), the active and reactive load expressions are defined such as to accommodate the various forms that a physical load may take in an electric power network:

$$P_{Li} = (P_{L0i} + P_{L1i}) + P_{L\text{-}Ii} |V_i| + P_{L\text{-}Zi} |V_i|^2 + P_{L\text{-}NPi} |V_i|^{NP_i} \tag{2.99}$$

$$Q_{Li} = (Q_{L0i} + Q_{L1i}) + Q_{L\text{-}Ii} |V_i| + Q_{L\text{-}Zi} |V_i|^2 + Q_{L\text{-}NQi} |V_i|^{NQ_i} \tag{2.100}$$

Focusing on the first expression, the term in round brackets represents the set value for the load. The reason why two constants are used (instead of one) is purely technical, that is to a have a common data structure with the power and frequency regulation model (described in sec. 2.5). The following two terms represent respectively loads of constant current and constant impedance. The last term, comprising the parameters $P_{L\text{-}NP\,i}$ and NP_i, allows to describe a load with a dependence in voltage raised to the arbitrary NP_i-th power. The same applies to the second expression.

2.4.2 Power generation model

In the classic framework, power generation is modelled by a constant, namely $\tilde{P}_{G\,i}$ in the three-phase case, or $P_{G\,i}$ in the single-phase case.

2.4.3 Types of bus

Since the model admits electrical networks composed of both three-phase and single-phase elements, from the outset it is helpful to define basic labelling rules for the nodes. Without loss of generality, it is assumed that all nodes contained in three-phase buses are labelled before (i.e. have smaller indices than) the ones corresponding to single-phase buses, and that the nodes within a three-phase bus are labelled by respecting the order *abc* of the phases. For example, if i is the index corresponding to the phase-a node of an arbitrary bus, then $i+1$ and $i+2$ correspond respectively to the phase-b and phase-c nodes of the same bus.

2.4.3.1 The swing bus

The swing bus is characterized by a fixed voltage modulus (typically of unit value in the per unit system) and a phase-a voltage phase of fixed arbitrary value (which is typically zero). The swing bus characteristic phase acts as a reference for the other voltage phases of the network, i.e. only their relative values with respect to the characteristic phase enter the calculations. Hence, in any network under study a bus must be designated as swing bus.

If the swing bus is three-phase, then its three voltage moduli are equal, and the three phases are arranged 120° apart, where the phase-a angle is set as the reference.

At every node of a swing bus, the apparent power generation is adjusted (without limit) to balance the power going into the bus admittance components and the load.

2.4.3.2 The *PV* bus

As its name suggests, the *PV* bus is characterized by a fixed level of active power generation and a fixed voltage modulus.

If the *PV* bus is three-phase (denoted by *PV3*), then the voltage moduli at its three nodes are all equal to the characteristic voltage modulus. The corresponding voltage phases are arranged uniformly 120° apart, yet with an unknown global orientation. Normally, the phase-a voltage phase, say θ_i, is selected as the unknown variable based on which the other phases are expressed, i.e. $\theta_{i+1} = \theta_i - 120^\circ$, $\theta_{i+2} = \theta_i + 120^\circ$.

Still in the three-phase case, the specified active power generation corresponds to the sum of the active power generated at every phase of the bus, without constraint on its distribution among the phases.

If the *PV* bus is single-phase (denoted by *PV1*), then its voltage modulus takes the characteristic value, and the lone voltage phase is treated as an unknown.

Lastly, at every node of the *PV* bus, the generated reactive power is adjusted (without limit) to balance the reactive power going into the bus admittance components and the load.

2.4.3.3 The *PQ* bus

At a *PQ* bus, the generated power is fixed, and the load takes the definite form presented in sec. 2.4.1.

The treatment of a *PQ* bus is the same irrespective of whether the bus is three-phase or single-phase, because in the former case the powers flowing through the different nodes are decoupled. Thus, for every node the complex voltages must be solved in order to achieve a balance between the generated power, the power going into the admittance components, and the load. As a result, two unknowns are introduced per *PQ* node: the phase and the modulus of the complex voltage.

2.4.3.4 Main characteristics of the different bus types

Table 2.1 summarises the differences between the various types of bus in terms of their fixed or conditioned quantities, the unknowns that they generate, and the quantities that are adjusted in the course of the iteration process. Of great importance is the number of unknown(s) generated as per bus type: zero for the swing bus, one for the *PV* bus, irrespective of whether it is three-phase or single-phase, six for the three-phase *PQ* bus, and two for the single-phase *PQ* bus. Note that the subscript G denotes the power generated, L the load, and Y the power going into the branches and shunt elements of the network. In the cases of the swing bus and the *PV3* bus, i represents a phase-a node and $i' \in \{i, i+1, i+2\}$. Otherwise, i can label any node.

Table 2.1 Bus types and characteristics in the classic model

Bus type	**Fixed quantities**	**Unknowns**	**Adjusted quantities**
Swing	$\lvert V_{i'}\rvert$, θ_i	-	$P_{Gi'} = P_{Yi'} + P_{Li'}$, $Q_{Gi'} = Q_{Yi'} + Q_{Li'}$
PV3	$\lvert V_{i'}\rvert$, $\tilde{P}_{Gi}$	θ_i	$Q_{Gi'} = Q_{Yi'} + Q_{Li'}$
PV1	$\lvert V_i\rvert$, P_{Gi}	θ_i	$Q_{Gi} = Q_{Yi} + Q_{Li}$
PQ	P_{Gi}, Q_{Gi}, P_{Li}, Q_{Li}	$\lvert V_i\rvert$, θ_i	-

2.4.4 The vector of mismatches

In view of table 2.1, given a network of n_{PV3} three-phase *PV* buses, n_{PV1} single-phase *PV* buses, and n_{PQ} *PQ* nodes, a number $n_{PV3} + n_{PV1} + 2n_{PQ}$ of unknowns are generated. They can

be arranged in a single vector x, expressed in the following compact form:

$$x = \begin{bmatrix} \theta_{PV3a} \\ \theta_{PV1} \\ \theta_{PQ} \\ |V_{PQ}| \end{bmatrix} \tag{2.101}$$

where θ_{PV3a} represents the voltage phases at the phase-a nodes of the *PV3* buses, θ_{PV1} the voltage phases at the *PV1* nodes, θ_{PQ} the voltage phases at the *PQ* nodes, and $|V_{PQ}|$ the voltage moduli at the *PQ* nodes.

The vector of mismatches is built based on the same bus ordering. Starting with the *PV3* buses, balance in power at every bus demands that the following quantity be brought to zero:

$$\tilde{\varepsilon}_{P(PV3)\,i}(x) = \sum_{p=i}^{i+2} \left\{ P_{Y\,p}(x) + P_{L\,p}(|V_p|) \right\} - \tilde{P}_{Gi} \tag{2.102}$$

where $\tilde{P}_{Gi}$ and $|V_i| = |V_{i+1}| = |V_{i+2}|$ are the set parameters per bus. The tilde is used to denote a three-phase quantity, which is labelled by the index of its corresponding phase-a node. Likewise, the mismatches

$$\varepsilon_{P(PV1)\,i}(x) = P_{Y\,i}(x) + P_{L\,i}(|V_i|) - P_{Gi} \tag{2.103}$$

at the *PV1* nodes need to be brought to zero, where P_{Gi} and $|V_i|$ are the set parameters, and i labels any *PV1* node. Lastly, based on the same principle the *PQ* mismatches

$$\varepsilon_{P(PQ)\,i}(x) = P_{Y\,i}(x) + P_{L\,i}(|V_i|) - P_{Gi} \tag{2.104}$$

$$\varepsilon_{Q(PQ)\,i}(x) = Q_{Y\,i}(x) + Q_{L\,i}(|V_i|) - Q_{Gi} \tag{2.105}$$

in which case i labels any PQ node, must be made to vanish. Gathering the expressions (2.102)–(2.105) into a single vector yields the vector of mismatches:

$$\varepsilon(x) = \begin{bmatrix} \tilde{\varepsilon}_{P(PV3)}(x) \\ \varepsilon_{P(PV1)}(x) \\ \varepsilon_{P(PQ)}(x) \\ \varepsilon_{Q(PQ)}(x) \end{bmatrix} \tag{2.106}$$

where each entry shown represents a set of mismatches of the indicated type. Obviously, the notation $f(x)$ is a shorthand to indicate that the function f is dependent on the elements of the vector x, as opposed to the vector x as a whole.

2.4.5 The Jacobian matrix

As pointed out in sec. 2.1, the computation of the Jacobian matrix is necessary to implement Newton's method. Symbolically, the classical Jacobian matrix can be represented as follows:

$$\nabla\varepsilon(x) = \begin{bmatrix} \frac{\partial\tilde{\varepsilon}_{P(PV3)}}{\partial\theta_{PV3a}} & \frac{\partial\tilde{\varepsilon}_{P(PV3)}}{\partial\theta_{PV1}} & \frac{\partial\tilde{\varepsilon}_{P(PV3)}}{\partial\theta_{PQ}} & \frac{\partial\tilde{\varepsilon}_{P(PV3)}}{\partial|V_{PQ}|} \\ \frac{\partial\varepsilon_{P(PV1)}}{\partial\theta_{PV3a}} & \frac{\partial\varepsilon_{P(PV1)}}{\partial\theta_{PV1}} & \frac{\partial\varepsilon_{P(PV1)}}{\partial\theta_{PQ}} & \frac{\partial\varepsilon_{P(PV1)}}{\partial|V_{PQ}|} \\ \frac{\partial\varepsilon_{P(PQ)}}{\partial\theta_{PV3a}} & \frac{\partial\varepsilon_{P(PQ)}}{\partial\theta_{PV1}} & \frac{\partial\varepsilon_{P(PQ)}}{\partial\theta_{PQ}} & \frac{\partial\varepsilon_{P(PQ)}}{\partial|V_{PQ}|} \\ \frac{\partial\varepsilon_{Q(PQ)}}{\partial\theta_{PV3a}} & \frac{\partial\varepsilon_{Q(PQ)}}{\partial\theta_{PV1}} & \frac{\partial\varepsilon_{Q(PQ)}}{\partial\theta_{PQ}} & \frac{\partial\varepsilon_{Q(PQ)}}{\partial|V_{PQ}|} \end{bmatrix}$$

From the mismatch expressions of the previous section, i.e. (2.4), (2.5), (2.99), (2.100) substituted in (2.102)–(2.105), the derivatives can be computed directly. To that effect, a couple of remarks are in order. First, considering that the admittance matrices of the network components are all symmetric (refer to sec. 2.3), Y_{bus} can be treated as symmetric. Second, recall that the voltage phases at the phase-b and phase-c nodes of a *PV3* bus depend on the one at the phase-a node, i.e. $\theta_{i+1} = \theta_i - 120^\circ$, $\theta_{i+2} = \theta_i + 120^\circ$. Therefore, the derivatives of mismatch relations with respect to θ_i must be properly handled through the use of the chain rule. For

example, assuming that i and j designate phase-a nodes contained in different *PV3* buses, the derivative of the mismatch $\tilde{\varepsilon}_{Pi}$ with respect to the phase θ_j includes the following terms:

$$\begin{aligned}\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} &= \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta'_j}\frac{\partial \theta'_j}{\partial \theta_j} + \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_{j+1}}\frac{\partial \theta_{j+1}}{\partial \theta_j} + \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_{j+2}}\frac{\partial \theta_{j+2}}{\partial \theta_j} \\ &= \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta'_j} + \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_{j+1}} + \frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_{j+2}}\end{aligned}$$

where the notation $\theta'_j = \theta_j$ is used to emphasise the double role played by θ_j, i.e. that of a function and that of a variable .

The resulting Jacobian matrix expressions are shown in table 2.2. The column i specifies the bus type associated with the mismatch, and column j the bus type associated with the derivative variable. For convenience, since the treatment of *PV1* and *PQ* nodes are the same with regards to the phase derivatives, together they are referred to as *P1* nodes (single-phase nodes with an associated phase unknown). Moreover, for simplicity summations over all nodes are written without bounds.

Table 2.2 Formulae of the Jacobian matrix elements in the classic model

Jacobian matrix elements	i	j
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_i} = \lvert V_i\rvert \sum_{p=i}^{i+2} \sum_k \lvert V_k\rvert (-G_{pk}\sin\theta_{pk} + B_{pk}\cos\theta_{pk}) + \lvert V_i\rvert^2 (B_{i,i+1} + B_{i,i+2} + B_{i+1,i+2} - B_{ii} - B_{i+1,i+1} - B_{i+2,i+2})$	*PV3a*	*PV3a* ($j = i$)
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2} \sum_{q=j}^{j+2} (G_{pq}\sin\theta_{pq} - B_{pq}\cos\theta_{pq})$	*PV3a*	*PV3a* ($j \neq i$)
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2} (G_{pj}\sin\theta_{pj} - B_{pj}\cos\theta_{pj})$	*PV3a*	*P1*
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{q=j}^{j+2} (G_{iq}\sin\theta_{iq} - B_{iq}\cos\theta_{iq})$	*P1*	*PV3a*
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_i} = \lvert V_i\rvert \sum_k \lvert V_k\rvert (-G_{ik}\sin\theta_{ik} + B_{ik}\cos\theta_{ik}) - B_{ii}\lvert V_i\rvert^2$	*P1*	*P1* ($j = i$)
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert (G_{ij}\sin\theta_{ij} - B_{ij}\cos\theta_{ij})$	*P1*	*P1* ($j \neq i$)
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{q=j}^{j+2} (-G_{iq}\cos\theta_{iq} - B_{iq}\sin\theta_{iq})$	*PQ*	*PV3a*
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert (-G_{ij}\cos\theta_{ij} - B_{ij}\sin\theta_{ij})$	*PQ*	*P1* ($j \neq i$)
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_i} = \lvert V_i\rvert \sum_k \lvert V_k\rvert (G_{ik}\cos\theta_{ik} + B_{ik}\sin\theta_{ik}) - G_{ii}\lvert V_i\rvert^2$	*PQ*	*PQ* ($j = i$)
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \lvert V_j\rvert} = \lvert V_i\rvert \sum_{p=i}^{i+2} (G_{pj}\cos\theta_{pj} + B_{pj}\sin\theta_{pj})$	*PV3a*	*PQ*
$\frac{\partial \varepsilon_{Pi}}{\partial \lvert V_i\rvert} = P_{L\text{-}Ii} + 2P_{L\text{-}Zi}\lvert V_i\rvert + NP_i\, P_{L\text{-}NPi}\lvert V_i\rvert^{NP_i-1} + \sum_k \lvert V_k\rvert (G_{ik}\cos\theta_{ik} + B_{ik}\sin\theta_{ik}) + G_{ii}\lvert V_i\rvert$	*PQ*	*PQ* ($j = i$)
$\frac{\partial \varepsilon_{Pi}}{\partial \lvert V_j\rvert} = \lvert V_i\rvert (G_{ij}\cos\theta_{ij} + B_{ij}\sin\theta_{ij})$	*P1*	*PQ* ($j \neq i$)
$\frac{\partial \varepsilon_{Qi}}{\partial \lvert V_i\rvert} = Q_{L\text{-}Ii} + 2Q_{L\text{-}Zi}\lvert V_i\rvert + NQ_i\, Q_{L\text{-}NQi}\lvert V_i\rvert^{NQ_i-1} + \sum_k \lvert V_k\rvert (G_{ik}\sin\theta_{ik} - B_{ik}\cos\theta_{ik}) - B_{ii}\lvert V_i\rvert$	*PQ*	*PQ* ($j = i$)
$\frac{\partial \varepsilon_{Qi}}{\partial \lvert V_j\rvert} = \lvert V_i\rvert (G_{ij}\sin\theta_{ij} - B_{ij}\cos\theta_{ij})$	*PQ*	*PQ* ($j \neq i$)

2.5 The power and frequency regulation model

In reality, no generator can produce power in an unconstrained fashion independently of its rotating speed (as is the case with the classic model's swing bus). Variations in load necessarily lead to variations in generator speed, however small such variations may be. In fact, a change in speed occurs when the torques acting on the shaft of a rotor do not add up to zero. For instance, if the mechanical torque that is driving the shaft of a generator is countered by a greater electrical torque produced by the load, then its rotational speed should decrease. Likewise, the power consumption by rotating machines is also affected by their rotational speeds. Though the generators and other rotating machines connected to a network may operate at various speeds, due to their differing number of poles, they all share the same frequency, i.e. the network frequency. The faster they rotate, the higher is the frequency, and vice versa. The present model thus goes one step further than the classic model in that it incorporates the effect of network frequency variation in its equations.

In addition, the model takes into consideration contracts that define the power transfer between different areas of a power grid, and the need to keep the network frequency close to its scheduled value. The contracts are generally made between balancing authorities that oversee the load and the power generation within their designated zone of the overall power grid. The transfer in power is done through one or more lines commonly referred to as tie lines. Maintaining the network frequency close to its scheduled value is also necessary to avoid a deterioration in the quality of the supplied electrical energy as well as equipment damage. As will be seen in sec. 2.5.4, those constraints are reflected in an additional expression of the vector of mismatches.

To be concise, in the remainder of this document the power and frequency regulation model may be referred to as the *regulation model*.

2.5.1 Load model

In agreement with Okamura *et al.* (1975), among others, the expressions for the active and reactive load, i.e. (2.99) and (2.100), are modified to include the effect of frequency variation:

$$P_{Li} = P_{L0\,i} + (1 + K_{P\,i}\Delta f)(P_{L1\,i} + P_{L\text{-}I\,i}\,|V_i| + P_{L\text{-}Z\,i}\,|V_i|^2 + P_{L\text{-}NP\,i}\,|V_i|^{NP_i}) \tag{2.107}$$

$$Q_{Li} = Q_{L0\,i} + (1 + K_{Q\,i}\Delta f)(Q_{L1\,i} + Q_{L\text{-}I\,i}\,|V_i| + Q_{L\text{-}Z\,i}\,|V_i|^2 + Q_{L\text{-}NQ\,i}\,|V_i|^{NQ_i}) \tag{2.108}$$

where $\Delta f = f - f_{sch}$ represents the frequency deviation from the scheduled frequency. As mentioned in sec. 2.4.1, the above equations are defined so as to accommodate various types of load.

2.5.2 Power generation model

Considering the interrelation between load changes and power generation, speed governing mechanisms are used to control the operation of prime movers, so as to stabilize generator output and speed. As explained in Wood and Wollenberg (1996) and Okamura *et al.* (1975), this results in the introduction of the term $-\Delta f/R_i$ in the expression of power generation, where R_i represents the specific speed-droop characteristics of the generator connected to node i. Furthermore, contractual agreements in power transfer between network areas or a working frequency range target may necessitate a correction P_r in the total power output (referred to as the power insufficiency of the generators) at a particular instance of operation. The individual output of the generators involved in providing such power correction may be set according to economic and reliability factors. A participation factor β_i is thus defined for each generator in accordance with a specific scheme (e.g. through economic dispatch calculations), such that the output contribution of the generator at node i is given by $\beta_i P_r$ and $\sum_i \beta_i = 1$. By combining the above-mentioned terms with the set power generation value, the total active power output of the generator at node i may be expressed as follows:

$$P_{Gi} = P_{G\text{-}set\,i} - \frac{\Delta f}{R_i} + \beta_i P_r \tag{2.109}$$

2.5.3 Types of bus

Three types of bus are also defined in the present model. They share several similarities with the classic model's swing bus, *PV* bus, and *PQ* bus. For distinguishing purposes, respectively they will be referred to as the reference bus, the generator bus, and the load bus.

As in the classic model (see sec. 2.4.3), nodes of three-phase buses are labelled before the ones of single-phase buses, and the labelling of the nodes of a three-phase bus respects the order *abc* of the phases.

2.5.3.1 The reference bus

The reference bus is characterized by a fixed voltage modulus (typically of unit value in the per unit system) and a phase-*a* voltage phase of fixed arbitrary value (which is typically zero). The characteristic phase of the reference bus marks the zero angle position for the other voltage phases of the network, i.e. only their relative values with respect to the characteristic phase enter the calculations. Hence in any network under consideration a bus must be designated as reference bus.

If the reference bus is three-phase, then its three voltage moduli are equal, and the three phases are arranged 120° apart, where the one corresponding to the phase-*a* is defined as the phase of reference.

The total active power generation at a reference bus is described by the expression (2.109), and the generated reactive power is adjusted (without limit) to match the reactive power going into the bus admittance components and the load. As will be seen in the next section, the reference bus is a special case of the generator bus.

2.5.3.2 The generator bus

The generator bus differs from the *PV* bus in its variable generation of active power. Otherwise, its treatment is essentially the same as the one of the *PV* bus.

Thus, at a generator bus the active power is specified according to (2.109). If the bus is three-phase, then the sum of the active power generated at its constituent nodes is specified, without constraint on how the power is distributed among the phases.

The generator bus is also characterized by a fixed voltage modulus. If the bus is three-phase, then the voltage moduli at its three nodes are all equal to the characteristic voltage modulus. The corresponding voltage phases are arranged uniformly 120° apart, yet with an unknown global orientation. Normally, the phase-*a* voltage phase is selected as the unknown variable based on which the other phases are evaluated. If the generator bus is single-phase, then its voltage modulus takes the characteristic value, and the lone voltage phase is treated as an unknown.

At every node of the generator bus, the generated reactive power is adjusted (without limit) to balance the reactive power going into the bus admittance components and the load.

The shorthands *Gen3* and *Gen1* are used to designate respectively a three-phase and a single-phase generator bus.

2.5.3.3 The load bus

The load bus is the power and frequency regulation model equivalent of the *PQ* bus.

Thus, at a load bus the generated power is fixed (represented by P'_{Gi} and Q'_{Gi}), and the load takes the definite form presented in sec. 2.5.1. The treatment of a load bus is the same irrespective of whether it is three-phase or single-phase, because in the former case the powers flowing through the different nodes are decoupled. For every node, the complex voltages must be solved in order to achieve a balance between the generated power, the power going into the admittance components, and the load. Therefore two unknowns are introduced per node: the phase and the modulus of the complex voltage.

The shorthand *Ld* is used to designate concisely a load bus.

2.5.3.4 Main characteristics of the different types of bus

Table 2.3 summarises the differences between the allowed bus types in terms of their fixed or conditioned quantities, the unknowns that they generate, and the quantities that are adjusted in the course of the iteration process. Of great importance is the number of unknowns generated as per bus type: zero for the reference bus, one for the generator bus, irrespective of whether it is three-phase or single-phase, six for the three-phase load bus, and two for the single-phase load bus. The subscript G denotes the power generated, L the load, and Y the power going into the branches and shunt elements of the network. The expressions $\tilde{P}_{Gi}$ and P_{Gi} follow from the power generation model (three-phase and single-phase generation respectively), and P_{Li} and Q_{Li} from the load model. The quantities P'_{Gi} and Q'_{Gi} denote constant power generation. In the cases of the reference bus and the three-phase generator bus, i represents a phase-a node and $i' \in \{i, i+1, i+2\}$. Otherwise, i labels any node.

Table 2.3 Bus types and characteristics in the power and frequency regulation model

Bus type	**Fixed / Conditioned quantities**	**Unknowns**	**Adjusted quantities**
Ref.	$\lvert V_{i'}\rvert$, θ_i, $\tilde{P}_{Gi}$	-	$Q_{Gi'} = Q_{Yi'} + Q_{Li'}$
Gen3	$\lvert V_{i'}\rvert$, $\tilde{P}_{Gi}$	θ_i	$Q_{Gi'} = Q_{Yi'} + Q_{Li'}$
Gen1	$\lvert V_i\rvert$, P_{Gi}	θ_i	$Q_{Gi} = Q_{Yi} + Q_{Li}$
Load	P'_{Gi}, Q'_{Gi}, P_{Li}, Q_{Li}	$\lvert V_i\rvert$, θ_i	-

2.5.4 The vector of mismatches

In view of table 2.3, given a network of n_{Gen3} three-phase generator buses, n_{Gen1} single-phase generator buses, and n_{Ld} load nodes, $n_{Gen3} + n_{Gen1} + 2n_{Ld}$ unknowns have to be determined. Moreover, two additional unknowns were introduced in the power generation and load models: the power insufficiency variable P_r and the network frequency f. All such variables can be

grouped into the single vector x:

$$x = \begin{bmatrix} \theta_{Gen3a} \\ \theta_{Gen1} \\ \theta_{Ld} \\ |V_{Ld}| \\ P_r \\ f \end{bmatrix} \tag{2.110}$$

where θ_{Gen3a} represents the voltage phases at the phase-a nodes of the three-phase generator buses, θ_{Gen1} the voltage phases at the single-phase generator buses, and θ_{Ld} and $|V_{Ld}|$ respectively the voltage phases and moduli at the load nodes.

The vector of mismatches is built in similar fashion as in the classic model, with the exception that two equations are added to allow for the resolution of the new variables P_r and f. The first equation is obtained easily by noting that (unlike the swing bus) the active power is not adjusted at a reference bus, such that a mismatch equation is set up in order to ensure the balance in power:

$$\tilde{\varepsilon}_{P(Ref)\,i}(x) = \sum_{p=i}^{i+2} \left\{ P_{Y\,p}(x) + P_{L\,p}(\left|V_p\right|, f) \right\} - \tilde{P}_{Gi}(P_r, f) \tag{2.111}$$

The term $\tilde{P}_{Gi}(P_r, f) = 3(P_{G\text{-}set\,i} - \Delta f / R_i + \beta_i P_r)$ is a three-phase quantity composed of the set power generation $3P_{G\text{-}set\,i}$ and the parameters R_i and β_i. The index i of the three-phase quantities corresponds to the phase-a of the bus.

The second equation reflects the operation of generators so as to fulfil a given power exchange agreement between areas of a network, and to maintain the network frequency at (or close to) its scheduled value. It is built as a linear combination of the gap between the active power transferred in designated tie line(s) with respect to its scheduled value, i.e. $P_T - P_{T\,sch}$, as well as the gap in frequency with respect to its scheduled value, i.e. $f - f_{sch}$:

$$\varepsilon_{pf} = a_p(P_T - P_{T\,sch}) + a_f(f - f_{sch}) \tag{2.112}$$

The active power transferred from the nodes m in an area M, to the nodes n in another area N, through N_{tb} tie branch(es) denoted by the index t, is obtained according to

$$P_T = \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} V_m \sum_{k\in\{m_t,n_t\}} (Y_t)^*_{mk} V^*_k \right\} \tag{2.113}$$

$$= \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} \sum_{k\in\{m_t,n_t\}} |V_m|\,|V_k|\,(G_t - jB_t)_{mk} e^{j\theta_{mk}} \right\} \tag{2.114}$$

$$= \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} \sum_{k\in\{m_t,n_t\}} |V_m|\,|V_k| \left\{ (G_t)_{mk}\cos\theta_{mk} + (B_t)_{mk}\sin\theta_{mk} \right\} \tag{2.115}$$

where $\{m_t\}$ stands for the set of primary nodes of tie branch t, and $\{m_t, n_t\}$ for the set of primary and secondary nodes of the same tie branch. Without loss of generality, a primary tie branch node is defined to be one where the conventional positive apparent power flows into its associated tie branch(es), and conversely a secondary tie branch node is one where the conventional positive apparent power flows out of its tie branch(es). The parameters a_p and a_f are determined so as to reproduce the control approach in effect. For example, flat tie line control, where the established power transfer is met, corresponds to the case $a_p \neq 0$ and $a_f = 0$. Similarly, flat frequency control, where the power output of a set of generators is regulated so as to maintain the network frequency at its scheduled value, corresponds to the case $a_p = 0$ and $a_f \neq 0$. If both a_p and a_f are nonzero, which is referred to as tieline bias control, then the extent to which ΔP_T approaches zero depends on the relative weight of the coefficient a_p with respect to a_f (and vice versa for Δf).

The remaining elements of the vector of mismatches closely parallel the ones in the classic model. Starting with the three-phase generator bus, balance in power demands that its corresponding mismatch (at the bus composed of the nodes i, $i+1$, and $i+2$)

$$\tilde{\varepsilon}_{P(Gen3)\,i}(x) = \sum_{p=i}^{i+2} \left\{ P_{Y\,p}(x) + P_{L\,p}(|V_p|, f) \right\} - \tilde{P}_{Gi}(P_r, f) \tag{2.116}$$

be brought to zero, where the active power generation is conditioned according to (2.109), and the voltage moduli are set at an arbitrary value. It should be apparent that the reference

bus is a special case of the generator bus, where the voltage moduli are typically set to unity, and the voltage phase acts as the reference relative to which the other phases of the circuit are evaluated. Likewise, the mismatch equation at a *Gen1* node takes the form

$$\varepsilon_{P(Gen1)\,i}(x) = P_{Y\,i}(x) + P_{Li}(|V_i|, f) - P_{Gi}(P_r, f) \tag{2.117}$$

where $P_{G\text{-}set\,i}$ and $|V_i|$ are set parameters, and i labels any *Gen1* node. Lastly, following the same reasoning, the load bus mismatches are given by

$$\varepsilon_{P(Ld)\,i}(x) = P_{Y\,i}(x) + P_{Li}(|V_i|, f) - P'_{Gi} \tag{2.118}$$

$$\varepsilon_{Q(Ld)\,i}(x) = Q_{Y\,i}(x) + Q_{Li}(|V_i|, f) - Q'_{Gi} \tag{2.119}$$

where i labels any load node. Recall that P'_{Gi} and Q'_{Gi} stand for constant power generation, and may very well be set to zero.

The vector of mismatches is finally obtained by gathering the expressions (2.111)–(2.119) as follows:

$$\varepsilon(x) = \begin{bmatrix} \tilde{\varepsilon}_{P(Ref)}(x) \\ \tilde{\varepsilon}_{P(Gen3)}(x) \\ \varepsilon_{P(Gen1)}(x) \\ \varepsilon_{P(Ld)}(x) \\ \varepsilon_{Q(Ld)}(x) \\ \varepsilon_{pf}(x) \end{bmatrix} \tag{2.120}$$

where each entry shown represents the set of mismatches of the type indicated.

2.5.5 The Jacobian matrix

Recall that the computation of the Jacobian matrix is necessary to implement Newton's method, and thus to solve for the voltages across the network. Based on the unknowns (2.110) and

mismatches (2.120), its contents can be represented symbolically as follows:

$$\nabla\boldsymbol{\varepsilon}(x) = \begin{bmatrix} \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial\theta_{Gen3a}} & \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial\theta_{Gen1}} & \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial\theta_{Ld}} & \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial|V_{Ld}|} & \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial P_r} & \frac{\partial\tilde{\varepsilon}_{P(Ref)}}{\partial f} \\ \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial\theta_{Gen3a}} & \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial\theta_{Gen1}} & \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial\theta_{Ld}} & \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial|V_{Ld}|} & \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial P_r} & \frac{\partial\tilde{\varepsilon}_{P(Gen3)}}{\partial f} \\ \frac{\partial\varepsilon_{P(Gen1)}}{\partial\theta_{Gen3a}} & \frac{\partial\varepsilon_{P(Gen1)}}{\partial\theta_{Gen1}} & \frac{\partial\varepsilon_{P(Gen1)}}{\partial\theta_{Ld}} & \frac{\partial\varepsilon_{P(Gen1)}}{\partial|V_{Ld}|} & \frac{\partial\varepsilon_{P(Gen1)}}{\partial P_r} & \frac{\partial\varepsilon_{P(Gen1)}}{\partial f} \\ \frac{\partial\varepsilon_{P(Ld)}}{\partial\theta_{Gen3a}} & \frac{\partial\varepsilon_{P(Ld)}}{\partial\theta_{Gen1}} & \frac{\partial\varepsilon_{P(Ld)}}{\partial\theta_{Ld}} & \frac{\partial\varepsilon_{P(Ld)}}{\partial|V_{Ld}|} & \frac{\partial\varepsilon_{P(Ld)}}{\partial P_r} & \frac{\partial\varepsilon_{P(Ld)}}{\partial f} \\ \frac{\partial\varepsilon_{Q(Ld)}}{\partial\theta_{Gen3a}} & \frac{\partial\varepsilon_{Q(Ld)}}{\partial\theta_{Gen1}} & \frac{\partial\varepsilon_{Q(Ld)}}{\partial\theta_{Ld}} & \frac{\partial\varepsilon_{Q(Ld)}}{\partial|V_{Ld}|} & \frac{\partial\varepsilon_{Q(Ld)}}{\partial P_r} & \frac{\partial\varepsilon_{Q(Ld)}}{\partial f} \\ \frac{\partial\varepsilon_{pf}}{\partial\theta_{Gen3a}} & \frac{\partial\varepsilon_{pf}}{\partial\theta_{Gen1}} & \frac{\partial\varepsilon_{pf}}{\partial\theta_{Ld}} & \frac{\partial\varepsilon_{pf}}{\partial|V_{Ld}|} & \frac{\partial\varepsilon_{pf}}{\partial P_r} & \frac{\partial\varepsilon_{pf}}{\partial f} \end{bmatrix}$$

Such a representation is helpful to visualize the different types of derivatives to be performed. Let us first examine the phase derivatives of the power mismatches ($\tilde{\varepsilon}_P$, ε_P, and ε_Q). As seen in sec. 2.5.3, close parallels can be drawn between the generator bus and the *PV* bus, and between the load bus and the *PQ* bus. In particular, their corresponding mismatches exhibit the same dependence on the phases, since they share the expressions for the power entering the bus (P_Y and Q_Y). Consequently, the results obtained in table 2.2 remain valid in the present model. Furthermore, because the reference bus is in fact a special case of the generator bus, its phase derivatives take the same form as $\partial\tilde{\varepsilon}_{P(PV3)\,i}/\partial\theta_j$ (or $\partial\varepsilon_{P(PV1)\,i}/\partial\theta_j$, if it is single-phase).

The next step is to determine the derivatives of the power mismatches with respect to the voltage moduli. Results similar to the classic case can be expected, because the expressions for the power entering the bus (P_Y and Q_Y) are unchanged, and the load model differs from the classic one merely by the multiplicative factors $(1+K_{P\,i}\Delta f)$ and $(1+K_{Q\,i}\Delta f)$. Hence only the derivatives of the mismatches at load buses with respect to their associated voltage moduli are modified.

The power mismatches exhibit a linear, or no dependence at all, on the power insufficiency or frequency. Consequently, the corresponding derivatives are straightforward to compute.

The expressions for all power mismatch derivatives can be found in tables 2.4–2.7, where the ones coincident with the classic model are repeated for clarity.

Table 2.4 Derivatives of the active power mismatches at the three-phase reference bus

Jacobian matrix elements, $i = i_{Ref}$	j
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2}\sum_{q=j}^{j+2}(G_{pq}\sin\theta_{pq} - B_{pq}\cos\theta_{pq})$	*Gen3a*
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2}(G_{pj}\sin\theta_{pj} - B_{pj}\cos\theta_{pj})$	*P1*
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \lvert V_j\rvert} = \lvert V_i\rvert \sum_{p=i}^{i+2}(G_{pj}\cos\theta_{pj} + B_{pj}\sin\theta_{pj})$	*Ld*
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial P_r} = -3\beta_i$	-
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial f} = \frac{3}{R_i} + 3K_{Pi}(P_{L1\,i} + P_{L\text{-}I\,i}\lvert V_i\rvert + P_{L\text{-}Z\,i}\lvert V_i\rvert^2 + P_{L\text{-}NP\,i}\lvert V_i\rvert^{NP_i})$	-

Table 2.5 Derivatives of the active power mismatches at the three-phase generator buses

Jacobian matrix elements, $i = i_{Gen3a}$	j
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_i} = \lvert V_i\rvert \sum_{p=i}^{i+2}\sum_{k}\lvert V_k\rvert(-G_{pk}\sin\theta_{pk} + B_{pk}\cos\theta_{pk}) + \lvert V_i\rvert^2\Big(\sum_{\substack{p,q=i\\p<q}}^{i+2} B_{pq} - \sum_{p=i}^{i+2} B_{pp}\Big)$	*Gen3a* ($j = i$)
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2}\sum_{q=j}^{j+2}(G_{pq}\sin\theta_{pq} - B_{pq}\cos\theta_{pq})$	*Gen3a* ($j \neq i$)
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \theta_j} = \lvert V_i\rvert \lvert V_j\rvert \sum_{p=i}^{i+2}(G_{pj}\sin\theta_{pj} - B_{pj}\cos\theta_{pj})$	*P1*
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial \lvert V_j\rvert} = \lvert V_i\rvert \sum_{p=i}^{i+2}(G_{pj}\cos\theta_{pj} + B_{pj}\sin\theta_{pj})$	*Ld*
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial P_r} = -3\beta_i$	-
$\frac{\partial \tilde{\varepsilon}_{Pi}}{\partial f} = \frac{3}{R_i} + 3K_{Pi}(P_{L1\,i} + P_{L\text{-}I\,i}\lvert V_i\rvert + P_{L\text{-}Z\,i}\lvert V_i\rvert^2 + P_{L\text{-}NP\,i}\lvert V_i\rvert^{NP_i})$	-

Table 2.6 Derivatives of the active power mismatches at the single-phase generator buses and load buses

Jacobian matrix elements, $i = i_{P1}$		j
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_j}$	$= \lvert V_i \rvert \lvert V_j \rvert \sum_{q=j}^{j+2} (G_{iq} \sin\theta_{iq} - B_{iq} \cos\theta_{iq})$	*Gen3a*
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_i}$	$= \lvert V_i \rvert \sum_k \lvert V_k \rvert (-G_{ik} \sin\theta_{ik} + B_{ik} \cos\theta_{ik}) - B_{ii} \lvert V_i \rvert^2$	*P1* ($j = i$)
$\frac{\partial \varepsilon_{Pi}}{\partial \theta_j}$	$= \lvert V_i \rvert \lvert V_j \rvert (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij})$	*P1* ($j \neq i$)
$\frac{\partial \varepsilon_{Pi}}{\partial \lvert V_j \rvert}$	$= \lvert V_i \rvert (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij})$	*Ld* ($j \neq i$)
	$i = i_{Gen1}$	**j**
$\frac{\partial \varepsilon_{Pi}}{\partial P_r}$	$= -\beta_i$	-
$\frac{\partial \varepsilon_{Pi}}{\partial f}$	$= \frac{1}{R_i} + K_{Pi}(P_{L1i} + P_{L\text{-}Ii} \lvert V_i \rvert + P_{L\text{-}Zi} \lvert V_i \rvert^2 + P_{L\text{-}NPi} \lvert V_i \rvert^{NP_i})$	-
	$i = i_{Ld}$	**j**
$\frac{\partial \varepsilon_{Pi}}{\partial \lvert V_i \rvert}$	$= (1 + K_{Pi}\Delta f)(P_{L\text{-}Ii} + 2P_{L\text{-}Zi} \lvert V_i \rvert + NP_i\, P_{L\text{-}NPi} \lvert V_i \rvert^{NP_i - 1}) + \sum_k \lvert V_k \rvert (G_{ik} \cos\theta_{ik} + B_{ik} \sin\theta_{ik}) + G_{ii} \lvert V_i \rvert$	*Ld* ($j = i$)
$\frac{\partial \varepsilon_{Pi}}{\partial P_r}$	$= 0$	-
$\frac{\partial \varepsilon_{Pi}}{\partial f}$	$= K_{Pi}(P_{L1i} + P_{L\text{-}Ii} \lvert V_i \rvert + P_{L\text{-}Zi} \lvert V_i \rvert^2 + P_{L\text{-}NPi} \lvert V_i \rvert^{NP_i})$	-

Table 2.7 Derivatives of the reactive power mismatches at load buses

Jacobian matrix elements, $i = i_{Ld}$	j
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_j} = \lvert V_i \rvert \lvert V_j \rvert \sum_{q=j}^{j+2} (-G_{iq} \cos\theta_{iq} - B_{iq} \sin\theta_{iq})$	*Gen3a*
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_j} = \lvert V_i \rvert \lvert V_j \rvert (-G_{ij} \cos\theta_{ij} - B_{ij} \sin\theta_{ij})$	*P1* $(j \neq i)$
$\frac{\partial \varepsilon_{Qi}}{\partial \theta_i} = \lvert V_i \rvert \sum_k \lvert V_k \rvert (G_{ik} \cos\theta_{ik} + B_{ik} \sin\theta_{ik}) - G_{ii} \lvert V_i \rvert^2$	*Ld* $(j = i)$
$\frac{\partial \varepsilon_{Qi}}{\partial \lvert V_i \rvert} = (1 + K_{Qi} \Delta f)(Q_{L\text{-}Ii} + 2Q_{L\text{-}Zi} \lvert V_i \rvert + NQ_i \, Q_{L\text{-}NQi} \lvert V_i \rvert^{NQ_i - 1}) + \sum_k \lvert V_k \rvert (G_{ik} \sin\theta_{ik} - B_{ik} \cos\theta_{ik}) - B_{ii} \lvert V_i \rvert$	*Ld* $(j = i)$
$\frac{\partial \varepsilon_{Qi}}{\partial \lvert V_j \rvert} = \lvert V_i \rvert (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij})$	*Ld* $(j \neq i)$
$\frac{\partial \varepsilon_{Qi}}{\partial P_r} = 0$	-
$\frac{\partial \varepsilon_{Qi}}{\partial f} = K_{Qi}(Q_{L1i} + Q_{L\text{-}Ii} \lvert V_i \rvert + Q_{L\text{-}Zi} \lvert V_i \rvert^2 + Q_{L\text{-}NQi} \lvert V_i \rvert^{NQ_i})$	-

To complete the picture, the derivatives of ε_{pf} are performed. Starting with the phase derivatives, two distinctions must be made: whether the phase involved in the derivative corresponds to a primary node or a secondary node of a tie branch, and whether the node is of the type *Gen3* or *P1* (the set of the *PV1* and *Ld* nodes).

Taking the case of $\partial\varepsilon_{pf}/\partial\theta_i$, where i represents a primary tie branch node categorised as *P1*, its expression is derived as follows:

$$\begin{aligned}
\frac{\partial\varepsilon_{pf}}{\partial\theta_i}\Big|_{i=i_{P1},\ \textit{tie prim.}} &= a_p \,\mathrm{Re}\Big\{ j\sum_{t=1}^{N_{tb}} \Big(\sum_{k\in\{m_t,n_t\}} V_i(Y_t)^*_{ik} V_k^* - \sum_{m\in\{m_t\}} V_m(Y_t)^*_{mi} V_i^* \Big) \Big\} \\
&= a_p \,\mathrm{Im}\Big\{ \sum_{t=1}^{N_{tb}} \Big(- \sum_{k\in\{m_t,n_t\}} V_i(Y_t)^*_{ik} V_k^* + \sum_{m\in\{m_t\}} V_m(Y_t)^*_{mi} V_i^* \Big) \Big\} \qquad (2.121) \\
&= a_p \sum_{t=1}^{N_{tb}} |V_i| \Big(\sum_{k\in\{m_t,n_t\}} |V_k| \left\{ -(G_t)_{ik}\sin\theta_{ik} + (B_t)_{ik}\cos\theta_{ik} \right\} \\
&\qquad + \sum_{m\in\{m_t\}} |V_m| \left\{ (G_t)_{mi}\sin\theta_{mi} - (B_t)_{mi}\cos\theta_{mi} \right\} \Big) \qquad (2.122)
\end{aligned}$$

Similarly, if i is a secondary tie branch node of the type *P1*, it follows that

$$\begin{aligned}
\frac{\partial\varepsilon_{pf}}{\partial\theta_i}\Big|_{i=i_{P1},\ \textit{tie sec.}} &= a_p \,\mathrm{Re}\Big\{ -j\sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} V_m(Y_t)^*_{mi} V_i^* \Big\} \\
&= a_p \,\mathrm{Im}\Big\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} V_m(Y_t)^*_{mi} V_i^* \Big\} \qquad (2.123) \\
&= a_p \sum_{t=1}^{N_{tb}} |V_i| \sum_{m\in\{m_t\}} |V_m| \left\{ (G_t)_{mi}\sin\theta_{mi} - (B_t)_{mi}\cos\theta_{mi} \right\} \qquad (2.124)
\end{aligned}$$

The derivatives of ε_{pf} with respect to the *Gen3a* phases are performed likewise, yet by taking into consideration the relationship between the three phases of the bus through the application of the chain rule (refer to sec. 2.4.5).

As to the voltage moduli derivatives, they also take two different forms, depending on whether they are done with respect to a voltage modulus on the primary side or the secondary side of a

tie branch. Their expressions can be obtained as follows:

$$\left.\frac{\partial \varepsilon_{pf}}{\partial |V_i|}\right|_{i=i_{Ld},\ \textit{tie prim.}} = a_p \frac{\partial}{\partial |V_i|} \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} \sum_{k\in\{m_t,n_t\}} V_m (Y_t)^*_{mk} V^*_k \right\}$$

$$= a_p \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \left(\sum_{k\in\{m_t,n_t\}} e^{j\theta_i} (Y_t)^*_{ik} V^*_k + \sum_{m\in\{m_t\}} V_m (Y_t)^*_{mi} e^{-j\theta_i} \right) \right\}$$

$$= a_p \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \left(\sum_{k\in\{m_t,n_t\}} |V_k| e^{j\theta_{ik}} (Y_t)^*_{ik} + \sum_{m\in\{m_t\}} |V_m| (Y_t)^*_{mi} e^{j\theta_{mi}} \right) \right\} \quad (2.125)$$

$$= a_p \sum_{t=1}^{N_{tb}} \left(\sum_{k\in\{m_t,n_t\}} |V_k| \{(G_t)_{ik} \cos\theta_{ik} + (B_t)_{ik} \sin\theta_{ik}\} \right.$$
$$\left. + \sum_{m\in\{m_t\}} |V_m| \{(G_t)_{mi} \cos\theta_{mi} + (B_t)_{mi} \sin\theta_{mi}\} \right) \quad (2.126)$$

$$\left.\frac{\partial \varepsilon_{pf}}{\partial |V_i|}\right|_{i=i_{Ld},\ \textit{tie sec.}} = a_p \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} V_m (Y_t)^*_{mi} e^{-j\theta_i} \right\}$$

$$= a_p \mathrm{Re}\left\{ \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} |V_m| (Y_t)^*_{mi} e^{j\theta_{mi}} \right\} \quad (2.127)$$

$$= a_p \sum_{t=1}^{N_{tb}} \sum_{m\in\{m_t\}} |V_m| \{(G_t)_{mi} \cos\theta_{mi} + (B_t)_{mi} \sin\theta_{mi}\} \quad (2.128)$$

Finally, the derivative with respect to the power insufficiency vanishes, because of the independence of ε_{pf} on P_r, and the derivative with respect to the frequency is a_f, which can be obtained by inspection.

Table 2.8 contains the expressions of all possible Jacobian derivatives of ε_{pf}, where the column on the right gives the type of the node associated with the derivative variable, as well as its location with respect to the tie branch.

Table 2.8 Derivatives of the regulation constraint ε_{pf}

Jacobian matrix elements	*i* / tie branch end
$\frac{\partial \varepsilon_{pf}}{\partial \theta_i} = a_p \sum_{t=1}^{N_{tb}} \sum_{r=i}^{i+2} \lvert V_r \rvert \Big(\sum_{k \in \{m_t, n_t\}} \lvert V_k \rvert \{-(G_t)_{rk} \sin\theta_{rk} + (B_t)_{rk} \cos\theta_{rk}\} + \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mr} \sin\theta_{mr} - (B_t)_{mr} \cos\theta_{mr}\} \Big)$	*Gen3a* / primary
$\frac{\partial \varepsilon_{pf}}{\partial \theta_i} = a_p \sum_{t=1}^{N_{tb}} \sum_{r=i}^{i+2} \lvert V_r \rvert \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mr} \sin\theta_{mr} - (B_t)_{mr} \cos\theta_{mr}\}$	*Gen3a* / secondary
$\frac{\partial \varepsilon_{pf}}{\partial \theta_i} = a_p \sum_{t=1}^{N_{tb}} \lvert V_i \rvert \Big(\sum_{k \in \{m_t, n_t\}} \lvert V_k \rvert \{-(G_t)_{ik} \sin\theta_{ik} + (B_t)_{ik} \cos\theta_{ik}\} + \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mi} \sin\theta_{mi} - (B_t)_{mi} \cos\theta_{mi}\} \Big)$	*P1* / primary
$\frac{\partial \varepsilon_{pf}}{\partial \theta_i} = a_p \sum_{t=1}^{N_{tb}} \lvert V_i \rvert \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mi} \sin\theta_{mi} - (B_t)_{mi} \cos\theta_{mi}\}$	*P1* / secondary
$\frac{\partial \varepsilon_{pf}}{\partial \lvert V_i \rvert} = a_p \sum_{t=1}^{N_{tb}} \Big(\sum_{k \in \{m_t, n_t\}} \lvert V_k \rvert \{(G_t)_{ik} \cos\theta_{ik} + (B_t)_{ik} \sin\theta_{ik}\} + \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mi} \cos\theta_{mi} + (B_t)_{mi} \sin\theta_{mi}\} \Big)$	*Ld* / primary
$\frac{\partial \varepsilon_{pf}}{\partial \lvert V_i \rvert} = a_p \sum_{t=1}^{N_{tb}} \sum_{m \in \{m_t\}} \lvert V_m \rvert \{(G_t)_{mi} \cos\theta_{mi} + (B_t)_{mi} \sin\theta_{mi}\}$	*Ld* / secondary
$\frac{\partial \varepsilon_{pf}}{\partial P_r} = 0$	-
$\frac{\partial \varepsilon_{pf}}{\partial f} = a_f$	-

CHAPTER 3

MODEL IMPLEMENTATION

The present chapter describes the methodology developed to implement the models of ch. 2 into the computer program *nr3r*. In every routine a strong emphasis is placed on reaching high computation speed. Information regarding the used programming language and computing tools can be found in Appendix II.

3.1 Program flowchart

The program flowchart is shown in fig. 3.1. It is applicable to both the classic model and the power and frequency regulation model. The differences between the two approaches are clarified in the following sections, which examine the routines in greater details.

To give a few explanatory remarks, tests are contained in boxes with rounded corners. The term $tol.$ represents the maximum allowed error for each ε_i element to assess convergence of the iterative process. Likewise, $div.\,tol.$ represents the minimum error for the ε_i elements to assess divergence of the iterative process. Moreover, k is the iteration variable, and k_{max} its maximum allowed value. The contents bounded by braces or in dotted boxes apply only when the tap changing functionality is activated. In that case, the parameters n_{adj1}, n_{adj2} and n_{mis} represent respectively the number of performed tap adjustments per iteration, the number of necessary tap adjustments per iteration, and the number of passed iteration cycles with unfulfilled tap adjustments.

3.2 Read and organize the input data

The input data elements and their structures are described in Appendix III. The data are divided into four main categories: the bus data, the branch data, the zigzag earthing transformer data, and the transferred power and frequency regulation data. The read process consists in dividing

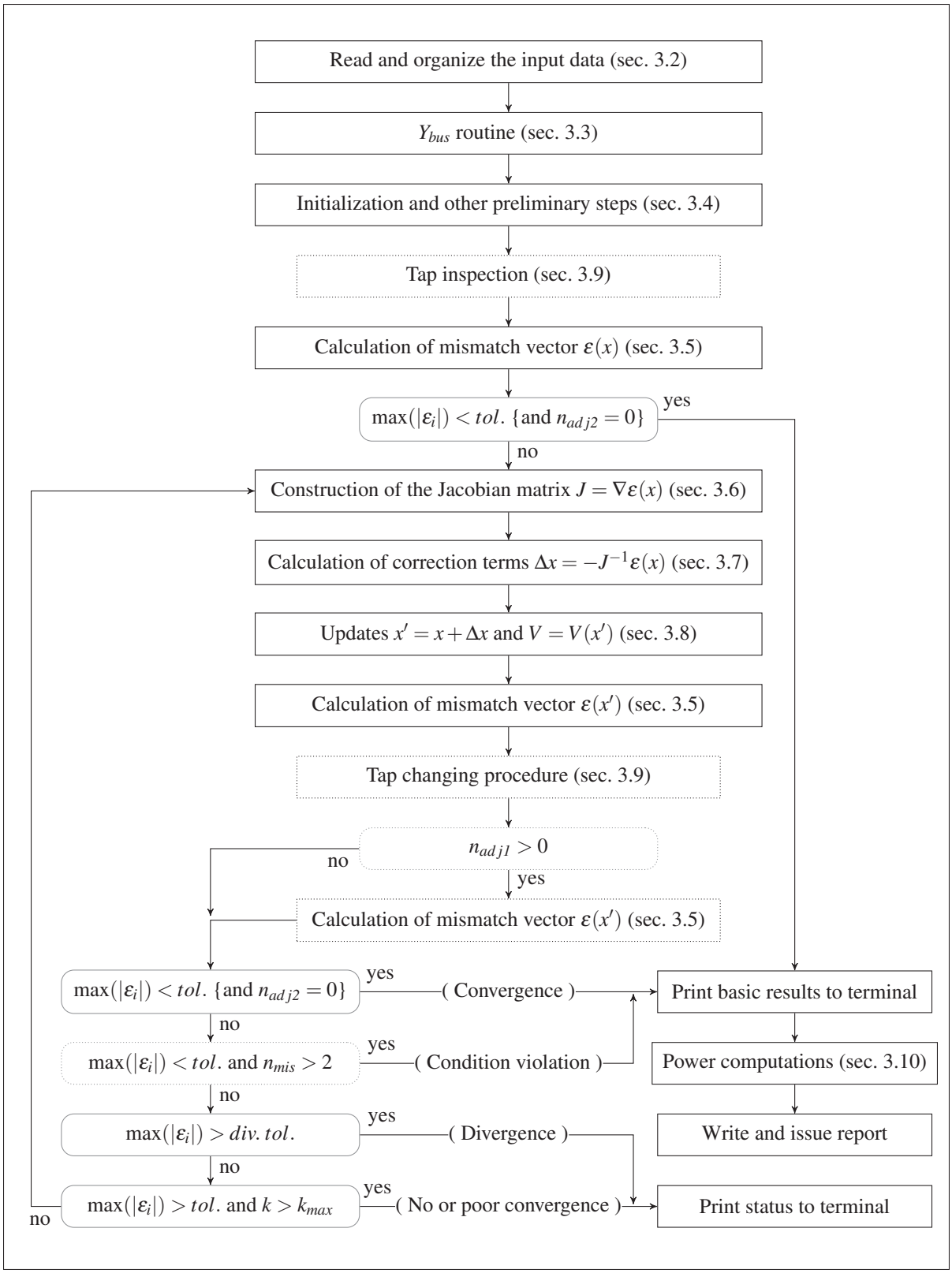


Figure 3.1 Flowchart showing the main routines and procedures of the program *nr3r*

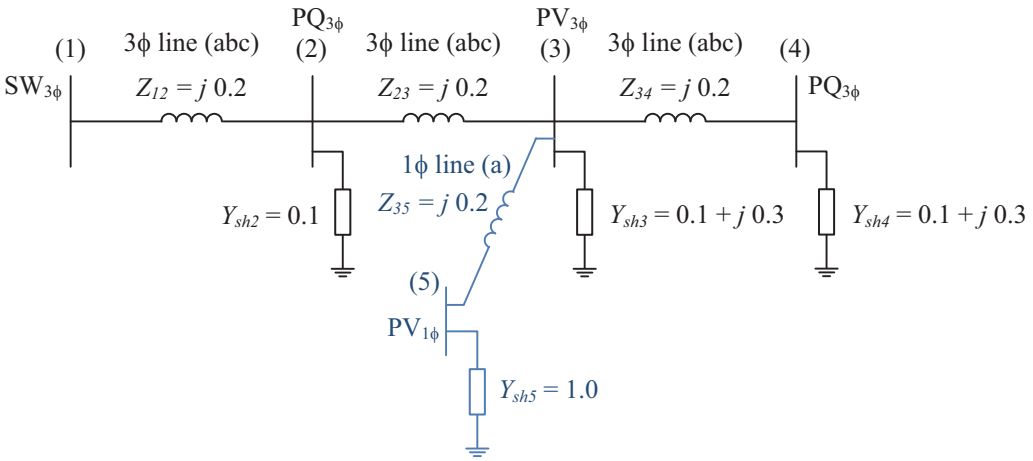
and storing those data into arrays of like entries (for example, the array of node numbers from the bus data) and into program variables so as to perform calculations in later stages efficiently.

The program first reads the bus data. The number of nodes N is obtained from the size n_{B3} of the list of three-phase buses and the size n_{B1} of the list of single-phase buses, i.e. $N = 3n_{B3} + n_{B1}$. The arrays used to store the bus data are then initialized based on N, and are filled according to the order in which the buses appear in the input file, beginning with the elements of *B3*, followed by the ones of *B1*. In the former case, the program duplicates the provided bus data, corresponding to the a phases, to also account for the phases b and c. Exceptionally the voltage phases at the b and c nodes are shifted respectively by -120° and $+120^\circ$ with respect to the provided phase a value, and the node numbers are generated by adding $-j$ and $+j$ to their associated bus number. The procedure results in several node based arrays, the coordinates of which are referred to as the node indices. It is worth emphasizing that such node indices differ from the node numbers, which can be fetched by means of node indices, but which are derived directly from the information provided by the program user. Taking the circuit of fig. 3.2 as an example, where bus #5 is a phase a single-phase node, the array of node numbers (*node_no*) annotated by the node indices reads as follows:

$$node\_no = [1,\ 1-1j,\ 1+1j,\ 2,\ 2-1j,\ 2+1j,\ 3,\ 3-1j,\ 3+1j,\ 4,\ 4-1j,\ 4+1j,\ 5]$$

$$\text{node index:}\quad 0,\quad 1,\quad 2,3,\quad 4,\quad 5,6,\quad 7,\quad 8,9,\quad 10,\quad 11,12$$

Branch data are stored similarly in the following step, where every branch is associated with a branch index (as opposed to a node index). Additionally, a tool (named *ibran_ptr*) is created to identify a branch index from a triplet composed of the branch's primary and secondary bus numbers, and circuit number *ckt*. Such tool takes the form of a dictionary, where the keys are given by the triplets, and the values by the branch indices. If an unknown triplet is provided, then the value -1 is returned. With *ibran_ptr*, it will then be possible to retrieve the admittance matrix of any branch associated with the provided triplet, and then to calculate the power going into that branch.



(a)

```
# Filename: nr3c_test_14a.dat

B3=[
# No,  name, type, Eb, mod, ph, Gsh, Bsh, PG, QG, PL0, QL0, ...
[  1, 'SW1',    1,  1,   1,  0,   0,   0,  0,  0,   0,   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[  2, 'PQ2',    3,  1,   1,  0, 0.1,   0,  0,  0,   0,   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[  3, 'PV3',    2,  1,   1,  0, 0.1, 0.3,  0,  0,   0,   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[  4, 'PQ4',    3,  1,   1,  0, 0.1, 0.3,  0,  0,   0,   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
]

L3=[
# index_I, index_J, ckt,   name, R,    X, G, B
[       1,       2,   1, 'L_12', 0, 0.20, 0, 0],
[       2,       3,   1, 'L_23', 0, 0.20, 0, 0],
[       3,       4,   1, 'L_34', 0, 0.20, 0, 0]
]

B1=[
# No,  name, type, Eb, mod, ph, Gsh, Bsh, PG, QG, PL0, QL0, ...
[  5, 'PV5',    2,  1,   1,  0, 1.0,   0,  0,  0,   0,   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
]

L1=[
# index_I, index_J, ckt,   name, R,    X, G, B
[       3,       5,   1, 'La35', 0, 0.20, 0, 0]
]
```

(b)

Figure 3.2 (a) Sample network and (b) corresponding input data used to illustrate basic program concepts

Due to their simplicity, zigzag earthing transformer data (contained in the list $TrZg$) are extracted directly into separate arrays. As in the cases of the bus and branch data, the values corresponding to a specific zigzag earthing transformer (e.g. bus number, coil impedance) can be fetched in those arrays by means of a dedicated index.

As to the transferred power and frequency regulation data (contained in the list $regfPT$), they are extracted at once from the input file. Problems with regards to the arbitrariness of the number of tie branches are circumvented by requiring the tie branch data to be specified together in a regular manner at the end of the list, and by extracting likewise data in steps of four. Furthermore, since the coefficient a_f and the scheduled transferred power $P_{T\text{-}sch}$ are specified in three-phase pu_{MW}/Hz and pu_{MW}, their specified values are multiplied by three in the program. In fact, the calculations of the unknowns are performed at the single-phase level.

3.3 Y_{bus} routine

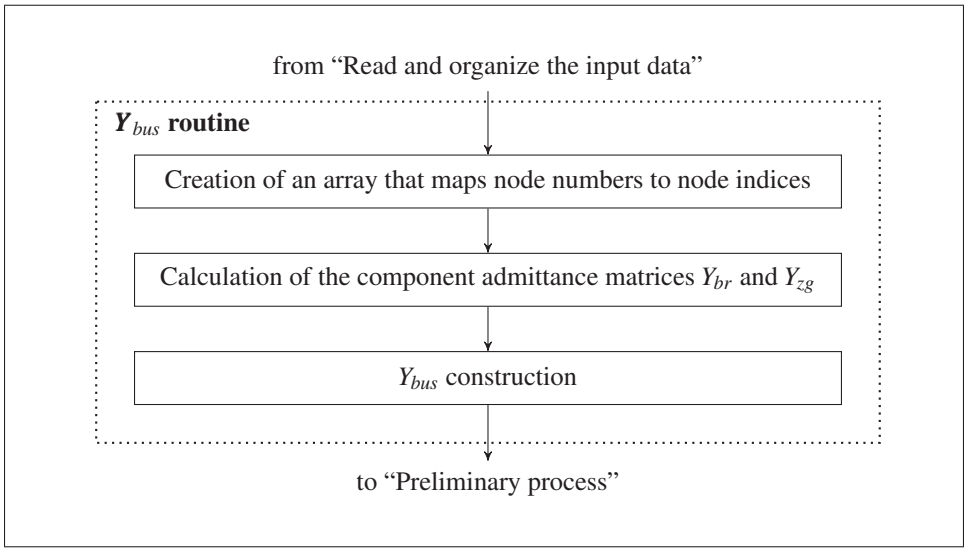


Figure 3.3 Steps performed within the Y_{bus} procedure

Recall from sec. 2.3.1 that the bus admittance matrix Y_{bus} is developed by adding the shunt admittance values of every node, the admittance matrix elements of the branches and zigzag earthing transformers in such a way that only elements with common node indices are added

together. The present section aims to describe in greater details how this approach is realised. The main steps involved in the process are shown in fig. 3.3.

For convenience, let us jump directly to the discussion of the second preliminary step, namely the calculation of the component admittance matrices Y_{br} and Y_{zg}; the creation of the mapping array from node numbers to node indices will be discussed afterwards. Branch data are scanned, and one by one their admittance matrices are calculated according to the formulae of sec. 2.3. Assuming a number N_{br} of branches, the results are stored in an array of dimension $N_{br} \times 36$ (denoted by *bran_ymat*), where each row corresponds to a branch, and the 36 columns contain the elements of Y_{br} taken in the row-major order (note that single-phase Y_{br} elements only fill the first four columns).

If zigzag earthing transformers are present in the network, their admittance matrix elements are stored similarly in a $N_{zg} \times 9$ array (denoted by *trzg_ymat*), where N_{zg} represents the number of zigzag earthing transformers, and the 9 columns contain the elements of the matrix (2.98) taken in the row-major order.

In power networks, only a small subset of all branches connect to any given bus. As a consequence, bus admittance matrices are generally filled with zeros.[1] In order to save memory space and increase calculation speed, Y_{bus} is created in the sparse matrix form. This is done most efficiently by means of three arrays containing respectively row indices, column indices, and the corresponding values to be added in the matrix. If several values are given identical row and column entries, then they are summed by default. In compact form, if I and J are the arrays of row and column indices, and M is the sparse matrix to be created, then the array of values D satisfies $M_{I(i)J(i)} = \sum_{\{k|I(k)=I(i),J(k)=J(i)\}} D(k)$. The objective is thus to list all values to be entered in Y_{bus} in an array D, and to carefully store their respective row and column numbers in the arrays I and J.

[1] Refer to p. 97 of Wood and Wollenberg (1996), among other sources, for a brief description of the rules to follow to construct the network Y matrix.

Decoupled shunt admittance elements are readily available from the bus data, and are stored in an array during the read process of the program. Such array can be used easily to form the part D_{sh} of the overall array of values D used to construct Y_{bus}. Because the decoupled shunt components each connect between a node and the ground, their admittance values add to the diagonal of Y_{bus}. In addition, since those values are specified in the bus data, their array is ordered in agreement with the bus admittance matrix by construction. As a result, the row and column values associated with D_{sh} are given by $I_{sh} = J_{sh} = [0, 1, 2, \ldots, N-1]$, where N represents the total number of nodes of the network.

Branch admittance values are stored in the two-dimensional array *bran_ymat*, which can be ravelled into a one-dimensional array D' in the row-major order. The corresponding arrays of row and column coordinates I' and J' can be created by examining the index pattern that results from enumerating the elements of the Y_{br} matrices in the row-major order. Since Y_{bus} is defined in terms of node indices, the primary and secondary node numbers specified in all branch data need to be converted into node indices. This is done by devising a two-dimensional mapping array such that for any node number (or array of node numbers) $x + jy$, the corresponding node index can be retrieved in row x and column $y+1$ of the array.[2] In the program, such array is referred to as *map_nn2i*. In the case of three-phase branches, the resulting primary and secondary node indices correspond by construction to phase a nodes, and their phase b and c counterparts are identified easily based on the fact that they follow each other in the bus data. In other words, if i_a represents the array of node indices (phase a) obtained from the primary bus numbers of all three-phase branches, then $i_b = i_a + 1$ and $i_c = i_a + 2$ contain respectively the phase b and phase c primary node indices of those branches. Moreover, the same reasoning applies to the arrays of secondary node indices, denoted by j_a, $j_b = j_a + 1$, and $j_c = j_a + 2$.

Focusing on the admittance matrix Y_{br}^k of an arbitrary three-phase branch indexed by k, for which the primary and secondary node indices are known from the specified bus numbers,

[2] In reality, the mapping array is constructed in such a way that the node index (or array of node indices) in question is retrieved at position $(x - x_{min}, y+1)$, where x_{min} is the smallest node number of the network, since the smallest specified node number may be large, as it is arbitrarily entered by the user, and memory space should be spared.

the enumeration of the matrix elements in the row-major order is described by the following ordered node index coordinates (the Y_{bus} coordinates):

$$I_k = [i_a[k], \ldots, i_a[k], i_b[k], \ldots, i_b[k], i_c[k], \ldots, i_c[k], j_a[k], \ldots, j_a[k], j_b[k], \ldots, j_b[k], j_c[k], \ldots, j_c[k]]$$
$$J_k = [i_a[k], i_b[k], i_c[k], j_a[k], j_b[k], j_c[k], \ldots, i_a[k], i_b[k], i_c[k], j_a[k], j_b[k], j_c[k]]$$

where each distinct element in I_k, and the series of indices $i_a[k]$, $i_b[k]$, $i_c[k]$, $j_a[k]$, $j_b[k]$, $j_c[k]$ in J_k have sixfold frequencies. On this basis, the I and J indices of all branches can be laid out in one go, thus allowing for high computation speed.

In fact, given a network of n three-phase branches, the node indices can be arranged in the following two-dimensional array:

$$ij_{3\phi} = \begin{matrix} [[& i_a[1], & i_a[2], & \ldots & i_a[n] &], \\ [& i_b[1], & i_b[2], & \ldots & i_b[n] &], \\ [& i_c[1], & i_c[2], & \ldots & i_c[n] &], \\ [& j_a[1], & j_a[2], & \ldots & j_a[n] &], \\ [& j_b[1], & j_b[2], & \ldots & j_b[n] &], \\ [& j_c[1], & j_c[2], & \ldots & j_c[n] &]] \end{matrix} \tag{3.1}$$

Then, the array of row indices can be constructed directly by performing the Kronecker product of $ij_{3\phi}$ ravelled in the column-major order and the unit array of size six. Symbolically, this translates into

$$I_{3\phi} = [i_a[1], i_b[1], \ldots, j_c[1], i_a[2], i_b[2], \ldots, j_c[2], \ldots, i_a[n], i_b[n] \ldots, j_c[n]] \otimes [1,1,1,1,1,1]$$

Likewise, the array of column indices $J_{3\phi}$ can be constructed by performing the Kronecker product of the unit array of dimension 6×1 and $ij_{3\phi}$, and by ravelling the resulting array in the column-major order.

The above method can also be applied to the simpler case of the single phase branches, thus yielding $D_{1\phi}$, $I_{1\phi}$ and $J_{1\phi}$. The row and column indices are determined based on the array $ij_{1\phi}$ of the primary and secondary node indices that correspond to the node numbers specified for the single phase branches, i.e.

$$ij_{1\phi} = \begin{matrix}[[& i_p[1], & i_p[2], & \dots & i_p[n'] &], \\ [& j_p[1], & j_p[2], & \dots & j_p[n'] &]]\end{matrix} \tag{3.2}$$

where n' is the number of single phase branches, and p is a dummy index representing the different phases to which the branches connect.

The case of the zigzag earthing transformers is handled in similar fashion, with the difference that only one three phase bus is specified per transformer. Thus D_{zg} is obtained by ravelling the two-dimensional array *trzg_ymat* in the row-major order, and I_{zg} and J_{zg} are inferred as above based on the array

$$ij_{zg} = \begin{matrix}[[& i_a[1], & i_a[2], & \dots & i_a[N_{zg}] &], \\ [& i_b[1], & i_b[2], & \dots & i_b[N_{zg}] &], \\ [& i_c[1], & i_c[2], & \dots & i_c[N_{zg}] &]]\end{matrix} \tag{3.3}$$

At this point, all the necessary information is available to build the Y_{bus} sparse matrix (of dimension $N \times N$). The required arrays of matrix values D, of row indices I, and of column indices J, can be generated through the concatenations of the previously obtained arrays:

$$D = [D_{sh}, D_{3\phi}, D_{1\phi}, D_{zg}]$$
$$I = [I_{sh}, I_{3\phi}, I_{1\phi}, I_{zg}]$$
$$J = [J_{sh}, J_{3\phi}, J_{1\phi}, J_{zg}]$$

Note that no explicit *for* loop is necessary, which greatly improves the calculation speed.

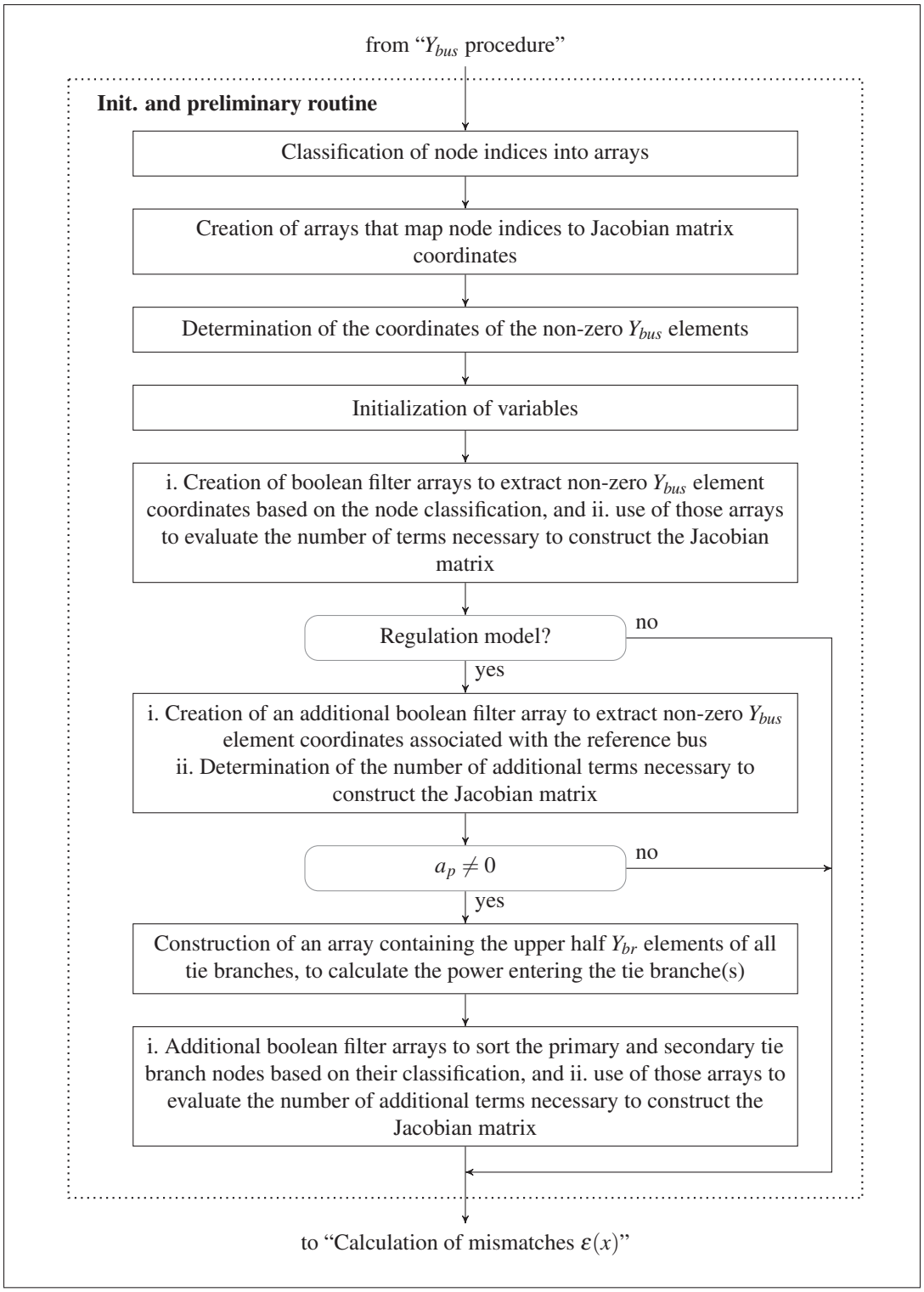


Figure 3.4 Main steps that enter the initialization and preliminary routine

3.4 Initialization and other preliminary steps

The initialization and preliminary routine mainly consists in constructing various arrays needed to perform the calculations of the subsequent routines efficiently. Its main steps are given by the flow chart of fig. 3.4, and are commented in the remainder of the section. Note that the activation of the tap changing functionality only effects the initialization of variables.

3.4.1 Classification of node indices

Node indices are classified into distinct arrays based on the categories defined in sec. 2.4 (e.g. *PV3a*, *P1*, *PQ*) and sec. 2.5 (e.g. *Gen3a*, *P1*, *Ld*). Doing so allows to construct the vector of mismatches, in particular to perform the adjustments of table 2.1 and 2.3. Morever, it permits to identify which Jacobian matrix expression applies to a given pair of mismatch and unknown (each defined in terms of such node indices). Obviously, at the end of every iteration, when the unknowns have been calculated, the identification of a node is also necessary to perform the correct updates, e.g. in the classic model, a *PV* node requires a voltage phase update only, whereas a *PQ* node demands an update of both its voltage modulus and phase. In the case of the circuit of fig. 3.2 (treated in the classic model), for example, the following arrays of node indices are constructed:

$$
\begin{aligned}
nodei\_sw &= [0,1,2] \\
nodei\_pv &= [6,7,8,12] \\
nodei\_pv3 &= [6,7,8] \\
nodei\_p1 &= [12,3,4,5,9,10,11] \\
nodei\_pq &= [3,4,5,9,10,11]
\end{aligned}
$$

They correspond respectively to the *swing*, all *PV*, *PV3*, *P1*, and *PQ* nodes. In particular, the last three are necessary to construct the Jacobian matrix.

3.4.2 Mapping arrays from node indices to Jacobian matrix coordinates

Assuming that the Jacobian matrix derivatives have been computed (based on the node index classification), the results have to be inserted in the Jacobian matrix. In order to do so, one needs to link the node indices previously classified into arrays with their associated positions in the vector of mismatches ε and in the vector of unknowns x, i.e. the Jacobian matrix coordinates. By construction, knowing where a node index maps into x is sufficient to also deduce its associated position in ε. For each set of classified node indices, a mapping tool is created in the form of an array of size N containing the desired coordinates in x at the positions defined by those indices, while assigning a value of -1 to all the other entries.[3] To have a more concrete understanding of the method, consider once more the case of fig. 3.2. The array that maps the *P1* node indices into the positions of their corresponding θ variables in x is

$$jacoi\_p1 = [-1,-1,-1,2,3,4,-1,-1,-1,5,6,7,1]$$

Thus, as an example, the position in x of the phase at the node indexed by 4 (*PQ* node) can be fetched as follows: $jacoi\_p1[4] = 3$. Similarly, the array that maps the *PQ* node indices into their corresponding $|V|$ variables is given by

$$jacoi\_q = [-1,-1,-1,8,9,10,-1,-1,-1,11,12,13,-1]$$

The case of the *PV3* nodes is slightly different, in that all three node indices of a given *PV3* bus are related to one coordinate in x, that is the position of the associated phase a voltage phase. Thus, to complete the present example:

$$jacoi\_pv3a = [-1,-1,-1,-1,-1,-1,\quad 0,-1,-1,-1,-1,-1,-1]$$
$$jacoi\_pv3b = [-1,-1,-1,-1,-1,-1,-1,\quad 0,-1,-1,-1,-1,-1]$$
$$jacoi\_pv3c = [-1,-1,-1,-1,-1,-1,-1,-1,\quad 0,-1,-1,-1,-1]$$

[3] In the present context, a *mapping* essentially implies a correspondence relation, not necessarily one-to-one, defined by means of an array, between two sets of numbers. As such it differs from its strict mathematical meaning.

$$jacoi\_pv3 = [-1,-1,-1,-1,-1,-1,\quad 0,\quad 0,\quad 0,-1,-1,-1,-1]$$

The construction of the Jacobian matrix in the regulation model follows more or less the same logic, except that special care must be taken to offset by one unit the coordinates of the standard mismatch elements, due to the insertion of the reference bus mismatch $\tilde{\varepsilon}_{P(Ref)}$ in the first position of ε. The row associated with mismatch ε_{pf} is also easily identifiable as it is the last element of ε. Analogously, the columns associated with P_r and f are easily identifiable as those elements are the penultimate and last ones of x.

3.4.3 Coordinates of the non-zero Y_{bus} elements

Considering the sparse nature of the bus admittance matrix, an important gain in calculation speed can be reached by restricting the evaluation of the Jacobian matrix expressions only to those terms that contain non-zero Y_{bus} elements. Consequently the row and column indices of such non-zero elements are preliminarily extracted in two arrays, referred to as *nzi* and *nzj* respectively.

3.4.4 Initialization of variables

Without going into excessive details, the array of complex voltages at all nodes (*node_v*) is constructed from the initial node voltage moduli and phases, and the loop counter *i* (shown in algorithm 2.1) is set to zero. In addition, provided that the tap changing function is activated, various variables used to monitor the tap changing activity are initialized.

3.4.5 Creation of boolean filter arrays based on the node classification

As shown in the tables of sec. 2.4.5 and 2.5.5, the computation of the elements of the Jacobian matrix requires knowning the category of the node associated with the mismatch being differentiated as well as the category of the node associated with the differentiation variable. An array that extracts all indices in *nzi* or *nzj* of a given category can be created by mapping

all elements of any of those arrays by means of the arrays introduced in the preceding section, and then by imposing the condition that the resulting elements be greater than -1. The result is a boolean array that can be applied to the original array *nzi* or *nzj* in order to keep only the items falling in the desired category.

As an example, consider the array of row indices of all non-zero Y_{bus} elements for the network shown in fig. 3.2:

$$nzi = [0,0,1,1,2,2,3,3,3,4,4,4,5,5,5,6,6,6,6,7,7,7,8,8,8,9,9,10,10,11,11,12,12]$$

The boolean array used to extract all *P1* nodes is created as follows:

$$\begin{aligned} f\_p1 &= jacoi\_p1[nzi] > -1 \\ &= [F,F,F,F,F,F,T,T,T,T,T,T,T,T,T,F,F,F,F,F,F,F,F,F,F,T,T,T,T,T,T,T,T] \end{aligned}$$

where 'F' stands for false, and 'T' for true. Finally, by applying *f_p1* back on *nzi*, one can verify that only the *nzi* elements of the type *P1* are kept:

$$nzi[f\_p1] = [3,3,3,4,4,4,5,5,5,9,9,10,10,11,11,12,12]$$

As a reference, a convention is used to name the boolean filter arrays: the part preceding the underscore indicates whether the filter is defined based on *nzi* — in which case f is used — or on *nzj* — in which case h is used; the part following the underscore indicates which set of node indices is kept.

The boolean filter arrays also provide the number of instances where nodes of two specific categories are linked through the Y_{bus} elements. In fact, the element-wise multiplication of a boolean array applied to *nzi* with another applied to *nzj* yields a third boolean array with true entries only at those positions that satisfy both node criteria. Such true entries can then be easily counted. For example, in order to count the number of cases where a *PV3* node is connected

to a *P1* node, the boolean array that identifies the *PV3* entries of *nzi* is multiplied (element-wise) by the one that identifies the *P1* entries of *nzj*, and the elements of the resulting array are summed (knowing that only the true results contribute a unit), i.e. $(f\_pv3 * h\_p1).sum()$. The ability to count the number of cases associated with each combination of node index categories (defined by the Jacobian matrix formulae) is useful to initialize arrays necessary to build the Jacobian matrix in sparse form.

In the power and frequency regulation model framework, in addition to the above results, the boolean filter array approach is used to identify the reference node indices in *nzi*, and the node indices of the other categories to which they are linked. With regard to the number of elements resulting from the differentiation of the mismatches with respect to f and P_r, they can be deduced directly from the known numbers of *Ref*, *Gen3*, *P1*, and *Ld* nodes in the network.

Given that a_p is non-zero, a few preliminary tasks need to be done to calculate ε_{pf} and its derivatives. First, the branch index of every tie branch is obtained from *ibran_ptr* and the triplet composed of the primary bus number, the secondary bus number, and the *ckt* parameter, which then allows to retrieve the corresponding Y_{br} elements from *bran_ymat*, and to store them into array *ytbr* (dimension $N_{tb} \times 36$). Yet as seen in (2.113), only the upper half tie branch Y_{br} matrix elements are necessary to calculate the transferred power. Thus, a useful array containing only those elements is created from *ytbr*:

$$ytbr1 = ytbr[:, 0:18].reshape(3 * nTB, 6) \tag{3.4}$$

where the colon selects all rows of *ytbr*, and $0:18$ the first 18 columns, i.e. the upper half Y_{br} elements. The resulting array is also reshaped into a $3N_{tb} \times 6$ array, by respecting the row-major order of its elements. Second, the primary and secondary node indices of the tie branche(s) (*nodei_tb* and *nodej_tb*) are obtained from the provided primary and secondary bus numbers and the mapping array *map_nn2i*. Lastly, the boolean array technique discussed above is applied to categorize those tie branch nodes, so as to perform the derivatives of table 2.8. Note

that the program accepts only three-phase power transfers, or equivalently power transferred between three-phase buses only.

3.5 Computation of mismatches

3.5.1 Mismatches in the classic model

Key arrays containing among others the shunt admittance values, the load parameters, and the power generation parameters at every node were previously created (sec. 3.2). Here they can be utilized directly to compute an N-dimensional array *eps_n* containing the balance in complex power $S_{Y\,i} + S_{L\,i} - S_{G\,i}$ at every node i, and from which the array of mismatches can be constructed.

Starting with the power into Y_{bus} (S_Y), it is computed at once from the array of complex voltages (*node_v*) and the bus admittance matrix, based on formula (2.3).

Based on equations (2.99) and (2.100), the arrays of the active load and the reactive load ($S_L = P_L + jQ_L$) are then computed from the element-wise absolute value of *node_v* and their respective node-based arrays of load parameters (see sec. 3.2).

The treatment of the generated power depends on the node type. In particular, the swing bus definition demands that generated power at every swing node be set equal to the sum of the Y_{bus} power and the load at those nodes (see table 2.1). Using the array of swing node indices *nodei_sw*, the swing bus adjustments are done as follows:

$$bus\_SG[nodei\_sw, 0] = eps\_n[nodei\_sw].real$$
$$bus\_SG[nodei\_sw, 1] = eps\_n[nodei\_sw].imag$$

where *bus_SG* is the $N \times 2$ array containing the generated active power in its first column, and the generated reactive power in its second column. It is one of the node based arrays created in routine *read and organize the input data*. Similarly, the *PV* bus definition demands that

the generated reactive power at every *PV* node be set equal to the sum of the reactive Y_{bus} power and the reactive load at those nodes. Using the array of *PV* node indices *nodei_pv*, the adjustments are done as follows:

$$bus\_SG[nodei\_pv, 1] = eps\_n[nodei\_pv].imag$$

Once the adjustments have been completed, the active and reactive parts of the generated power (respectively the first and second columns of *bus_SG*) can be subtracted from the sum of the previously obtained power arrays to obtain *eps_n* (with a factor of j in front of the reactive components). Thus, *eps_n* is created incrementally, first containing only the S_Y portion, then $S_Y + S_L$, and finally $S_Y + S_L - S_G$.

A subset of the elements of *eps_n* can be selected to build the vector of mismatches (2.106). The *PV3* bus mismatches are obtained by taking the real part of the sum of the complex mismatches at the three nodes of the buses, that is

$$eps\_pv3 = (eps\_n[nodei\_pva] + eps\_n[nodei\_pvb] + eps\_n[nodei\_pvc]).real$$

where the arrays of *PV3* node indices were used to extract the mismatches at the phases a, b, and c of the *PV3* buses. The other elements of the vector of mismatches are also fetched from *eps_n* by means of their corresponding arrays of node indices (the real part is taken to select the active mismatches, and the imaginary part to select the reactive mismatches). Their concatenation with *eps_pv3* then yields the vector of mismatches:

$$eps = concatenate([eps\_pv3,\ eps\_n[nodei\_p1].real,\ eps\_n[nodei\_pq].imag])$$

Node based arrays thus allow to compute the vector of mismatches simply, without the use of loops or other expensive computing methods.

3.5.2 Mismatches in the power and frequency regulation model

The part of the vector of mismatches involved with the *Gen3* buses, the *P1* nodes, and the *Ld* nodes is determined as in the classic case, yet with additional node based arrays to account for the specifics of the power and frequency regulation model, i.e. different load and power generation models. The main differences in the procedure lie in the determination of $\tilde{\varepsilon}_{P(Ref)}$ and ε_{pf}.

The reference bus possesses essentially the same characteristics as a generator bus. Therefore, as far as the mismatches are concerned, its treatment is the same as the one of a *Gen3* bus (or *Gen1* bus if it is single-phase). As a consequence, only the adjustment in reactive power is made at the reference nodes:

$$bus\_SG[nodei\_sw, 1] = eps\_n[nodei\_sw].imag$$

For practical purposes, common naming is used for arrays that parallel ones in the classic model, e.g. *nodei_sw* is used here to store the node indices of the reference bus. Once the array *eps_n* has been calculated, the active power mismatch at the reference bus is obtained simply through a summation function acting on the reference bus element(s):

$$eps\_ref = eps\_n[nodei\_sw].sum().real$$

which is equally applicable whether the reference bus is three-phase or single-phase.

The evaluation of ε_{pf} is done in a few steps, by carefully arranging the arrays representing the elements of P_T in (2.113), and taking advantage of the distributive and element-wise multiplicative properties of array objects (in the *NumPy* package). First, the array of voltages at the primary tie branch nodes, involved in the left-multiplication of (2.113), is set up as a $3N_{tb} \times 1$ array:

$$nu1 = (brw * node\_v[nodei\_tb]).reshape(3 * nTB, 1) \tag{3.5}$$

The array *brw* is introduced only to give additional flexibility to the program.[4] For example, it allows to define the transferred power as measured on one side (with the corresponding nodes specified as primary, each with a *brw* value of +1) or the other (with the corresponding nodes specified as secondary, each with a *brw* value of -1) of a tie branch. It is built once as a $3N_{tb}$-dimensional array, by repeating three times each *w* value specified per branch.

Second, the right multiplication by the voltages at the primary and secondary tie branch nodes is effected by using an array consisting of a stack of stacks of three identical arrays of the voltages at the nodes of every tie branch. Given *tbr_ij*, the array of tie branch node indices where each row is associated with a distinct tie branch, the first three columns with the primary node indices, and the last three columns with the secondary indices, the voltage array is obtained as follows:

$$nu2 = tile(node\_v[tbr\_ij], (1,3)).reshape(3 * nTB, 6) \tag{3.6}$$

where the tile function repeats an array according to a specified two-dimensional pattern, and the reshape operation is done so as to match the array with *ytbr1* (array of admittance matrix elements constructed in the preliminary routine of sec. 3.4).

At this point, all necessary elements to compute ε_{pf} are available. The active power transferred can be computed from the arrays (3.4)–(3.6) as follows:

$$PT = sum((nu1 * conj(ytbr1 * nu2)).real)$$

where the summation function sums all elements of its input array, and *conj* performs the complex conjugate of its argument. The quantity ε_{pf} thus follows from

$$eps\_pf = ap * (PT - PTsch) + af * (f - fsch)$$

[4] Refer to sec. 2.4 of the appendix III for additional information, especially regarding how the reference direction indices should be specified in the input file.

where the frequency f is calculated by Newton's method (or in the case of the first iteration, it is assigned an initial value), and the other parameters were stored in the initial routine (sec. 3.2).

As in the classic model framework, the vector of mismatches is obtained by concatenation:[5]

$$eps = concatenate([eps\_ref,\ eps\_pv3,\ eps\_n[nodei\_p1].real,\ eps\_n[nodei\_pq].imag,\ eps\_pf])$$

As already mentioned, for simplicity the program uses common array naming with the classic model. Nevertheless, due to the similarities between the bus types in both models, it should be clear that the array of *Gen3* mismatches is given by *eps_pv3*, and that of the reactive *Ld* mismatches is given by $eps\_n[nodei\_pq].imag$.

3.6 Construction of the Jacobian matrix

3.6.1 Insertion of values in the Jacobian matrix

The Jacobian matrix is created by using the same tool as for the bus admittance matrix, that is the class *csr_matrix* (from the package *SciPy*), which builds a sparse matrix based on three input arrays, namely one (D) containing the results to be added in the matrix, and the others (M and N) containing respectively their associated row and column indices. As mentioned in sec. 3.3, values with the same row and column indices are added in the matrix by default. Thus, the general method to insert elements in the Jacobian matrix consists of a few steps:

a. Identify the node indices i associated with the differentiated mismatches ε_i by means of the relevant filter array, e.g. $nzi[filtre]$, convert them into their respective row coordinates m of the Jacobian matrix by using one of the mapping arrays built in the preliminary routine (sec. 3.4.2), and insert them in the array M;

[5] In practice, the first and last elements, which are scalars, must be transformed into one-dimensional arrays before being concatenated, i.e. $array([eps\_ref])$ and $array([eps\_pf])$.

b. Identify the node indices j associated with the differentiation variables x_j by means of the filter array used in the previous step, e.g. $nzj[filtre]$, convert them into their respective column coordinates n of the Jacobian matrix by using one of the mapping arrays built in the preliminary routine (sec. 3.4.2), and insert them in the array N;

c. Insert the associated results, i.e. constituting $\frac{\partial \varepsilon_m}{\partial x_n}$, in the array D.

Prior to the application of the method, the arrays M, N, and D have been initialized by means of the boolean filter arrays, as pointed out in sec. 3.4.5. A simple subroutine (referred to as *store_results*) is used in order to systematically insert values into those arrays.

In the power and frequency regulation model, special attention must be paid to take into account the additional mismatch contributed by the reference node, which results in shifting the other mismatches by one unit with respect to the classic model's Jacobian matrix.

3.6.2 Classic model

The objective is to determine the various expressions of table 2.2 in a way that minimizes computing time. In particular, loops should be avoided, as they prove to be quite resource intensive.

The Jacobian matrix expressions generally consist of a linear combination of trigonometric functions, the coefficients of which contain bus admittance matrix elements. Due to the sparse nature of that matrix, the extent of the calculations can be significantly reduced by focusing only on the Y_{bus} elements that are non-zero. This is done by calculating once the arrays of all basic quantities that form the Jacobian matrix terms, based on the previously determined arrays *nzi* and *nzj* (containing respectively the row and column indices of all non-zero admittance matrix elements). The calculation of a set of matrix elements can then be carried out merely by picking the needed elements from those arrays. In more concrete terms, the conductances and susceptances of the non-zero Y_{bus} elements are first organized in two arrays according to

nzi and *nzj*:

$$nzg = yb[nzi, nzj].A1.real \tag{3.7}$$

$$nzb = yb[nzi, nzj].A1.imag \tag{3.8}$$

The $A1$ functionality simply returns the matrix elements in a flattened array. Next, the one-time calculations of the basic quantities used to obtain the Jacobian matrix terms are done as follows:

$$VI = absolute(node\_v[nzi]) \tag{3.9}$$

$$VJ = absolute(node\_v[nzj]) \tag{3.10}$$

$$ThIJ = angle(node\_v[nzi]) - angle(node\_v[nzj]) \tag{3.11}$$

$$cThIJ = cos(ThIJ) \tag{3.12}$$

$$sThIJ = sin(ThIJ) \tag{3.13}$$

where each operator on the right-hand side acts on an element-wise basis.

A subroutine *select_dat* is created in order to select elements of those arrays. The selection is realized by means of a combined filter array, given by the element-wise multiplication of two of the boolean filter arrays prepared in the preliminary routine (sec. 3.4.5). Which boolean arrays are to be used depends on the case under consideration (see table 2.2). For example, let us consider the case of $\frac{\partial \varepsilon_{Pi}}{\partial \theta_j}$, where i is a *P1* node index, and j a *PV3a* node index. The array that selects all combinations of *P1* and *PV3* node indices with corresponding non-zero Y_{bus} elements is given by

$$filtre = f\_p1 * h\_pv3$$

With *filtre* it is then straightforward to pick the quantities needed from (3.7)–(3.10) and (3.12)–(3.13) to calculate the Jacobian matrix elements of interest. For example, all relevant $|V_i|$ values are obtained as $vi = VI[filtre]$. The subroutine *select_dat* uses this principle to return

all required G_{ij}, B_{ij}, $|V_i|$, $|V_j|$, $\cos\theta_{ij}$, and $\sin\theta_{ij}$ basic quantities, given the complete arrays (3.7)–(3.10), (3.12)–(3.13), and a boolean array of the type *filtre* as its input.

Once the basic quantities are available, there remains to multiply them as required by table 2.2. In each case, the result is an array of elementary Jacobian matrix terms. Depending on the form of the Jacobian matrix expression under consideration, subsets of such elementary terms may or may not need to be added together. The case of $\frac{\partial \varepsilon_{Pi}}{\partial \theta_j}$, where i and j label different *P1* nodes, for example, does not require a summation, such that the corresponding elementary terms can be directly inserted in the Jacobian matrix. If i and j label the same *P1* nodes, however, a summation is required.

One can identify two different types of summations: the summations over all node indices k when the mismatch and the differentiation variable have the same node index; the summations resulting from the three-phase nature of the *PV3* bus mismatches, and the fact that such buses are each associated with a single voltage phase unknown.

The latter case can be treated in a fairly simple way, owing to the nature of the *csr* sparse matrix function. In fact, in the process of inserting results into the arrays D, M, and N, a summation over the three phases of a *PV3* bus can be realized by mapping all three node indices to the same Jacobian matrix coordinate, since the *csr_matrix* tool sums values (in D) with identical coordinates (in M and N). This is exactly what the array *jacoi_pv3* does.

The former case is a little more elaborate due to the arbitrariness of network topologies. In fact, those summations involve an arbitrary number of non-zero terms, depending on how many corresponding Y_{bus} elements are non-zero. Nevertheless, an examination of the array *nzi* proves to be quite informative, allowing for a systematic solution of the problem. Note that *nzi* lists all node indices, each having a specific number of occurrences (also referred to as frequency here). The frequencies of all elements are obtained and stored in an array *ifreq* by means of a function called *bincount* (from the *NumPy* package), such that the number of occurrences of the index i in *nzi* is given by $ifreq[i]$. When constructing a set of Jacobian matrix elements (associated with a specific category of nodes), an array of elementary terms is obtained, and

named *eldat* (for elementary data). Such array is also associated with repeated node indices, the frequencies of which can be obtained from *ifreq* by looking at the positions described by those indices; that sub-array will be referred to as *ifreqx*. For example, the frequencies of all *PQ* node indices are obtained from $ifreqx = ifreq[nodei\_pq]$. The frequencies indicate which subsets of the array of elementary terms should be added together. In order to be computationally efficient, the summation results sought will be derived from the array of cumulative sums of *eldat* (computed with the function *cumsum*, from the *NumPy* package) through the application of the following procedure:

a. Compute the array of cumulative sums of *eldat*, i.e. $eld\_cs = cumsum(eldat)$;

b. Compute the array of cumulative sums of *ifreqx* subtracted by 1, i.e.

$$p1 = cumsum(ifreqx) - 1$$

which corresponds to the end positions of every subset of *eldat* to be summed together (defined by a repeated node index *i* in *nzi*);

c. Create the array of the begin positions of every subset of *eldat* to be summed together, by shifting *p1* by one position, i.e.

$$p0 = concatenate((array([0]),\ p1[:-1]))$$

where $[:-1]$ points to all elements except the last one;

d. Derive the summations of elementary terms to be inserted in the Jacobian matrix from *eld_cs* using *p0* and *p1*, and store them in an array *depsdx*, i.e.

$$depsdx = eld\_cs[p1] - eld\_cs[p0]$$
$$depsdx[0] = eld\_cs[p1[0]]$$

In the last step, for every set of summed *eldat* elements, the subtration is performed to nullify the contribution to the cumulative sum coming from previous elements. The second assignment is needed since there are no prior elements to the first subset of summed *eldat* elements. Since the above steps are applied repeatedly to compute the various Jacobian matrix elements, they are integrated in the subroutine *sorted_sum* (taking *eldat* and *ifreqx* as input, and returning *depsdx* as its output).

Additional remarks need to be made regarding the computation of the extra terms present in the derivative expressions with $i = j$. The filter arrays used to extract elements of the same node category but with $i \neq j$ (i.e. $f\_pv3 * h\_pv3$, $f\_p1 * h\_p1$, $f\_pq * h\_p1$) do not in fact exclude the cases $i = j$. Nevertheless, this turns out to be favourable, because such cases actually correspond to the extra terms sought in the derivative expressions with $i = j$. The mapping of their node indices to Jacobian matrix coordinates then naturally assigns the results to the correct entries of the Jacobian matrix, which are then added to their trigonometric complement by the *csr* matrix building tool.

Finally, the terms originating from the load expressions in the voltage derivatives can be computed by applying the array of *PQ* node indices *nodei_PQ* to the node based arrays built in the initial routine that reads the input data. Regarding the array of voltages $|V_i|$, it is obtained from *VI* as follows:

$$vi = VI[p2[nodei\_pq]]$$

where $p2 = cumsum(ifreq) - ifreq$ is the array that points to the first elements of all subsets of *nzi* defined by a distinct index value (recall that array elements are labelled from 0).

To illustrate the methodology, fig. 3.5 shows the function that yields all derivatives of the *PV3* bus mismatches. The parameter k is used to successively enter elements in the arrays M, N, and D. The parameters *nP1* and *nPQ* give respectively the number of *P1* and *PQ* nodes.[6]

[6] In practice, some of the arrays are used globally, and are attached to a class q. In such case, their name is preceded by the prefix 'q.', which for simplicity is omitted here.

```
def deps_PV3dx(q, M, N, D, k, nzg, nzb, VI, VJ, cThIJ, sThIJ):
    """Derivatives of eps_PV3 w.r.t. the voltage phases and moduli."""
    # i,j: node indices; m,n: Jacobian matrix indices

    # 1.1.1 ELEMENT deps_PV3i/dth_i (i = i_pv3)
    filtre = f_pv3
    gik, bik, vi, vk, cthik, sthik = select_dat(
                                        nzg, nzb, VI, VJ, cThIJ, sThIJ, filtre)
    eldat = vi*vk*(-gik*sthik + bik*cthik)
    depsdx = sorted_sum(eldat, ifreq[nodei_pv3])
    m = jacoi_pv3[nodei_pv3]
    M, N, D, k = store_results(M,N,D,k, row=m, col=m, value=depsdx)
    # NB Terms (-Bii - Bi+1,i+1 - Bi+2,i+2 + Bi,i+1 + Bi,i+2 + Bi+1,i+2)|Vi|^2
    # are introduced in 1.1.2

    # 1.1.2 ELEMENTS deps_PV3i/dth_j (i = i_pv3, j = j_pv3)
    filtre = f_pv3*h_pv3
    gxy, bxy, vx, vy, cthxy, sthxy = select_dat(
                                        nzg, nzb, VI, VJ, cThIJ, sThIJ, filtre)
    depsdx = vx*vy*(gxy*sthxy - bxy*cthxy)
    m = jacoi_pv3[nzi[filtre]]
    n = jacoi_pv3[nzj[filtre]]
    M, N, D, k = store_results(M,N,D,k, row=m, col=n, value=depsdx)

    # 1.2 ELEMENTS deps_PV3_i/dth_j (i = i_pv3, j = j_pv1 or j_pq)
    if nP1 > 0:
        filtre = f_pv3*h_p1
        if filtre.any():
            gij, bij, vi, vj, cthij, sthij = select_dat(
                                        nzg, nzb, VI, VJ, cThIJ, sThIJ, filtre)
            depsdx = vi*vj*(gij*sthij - bij*cthij)
            m = jacoi_pv3[nzi[filtre]]
            n = jacoi_p1[nzj[filtre]]
            M, N, D, k = store_results(M,N,D,k, row=m, col=n, value=depsdx)

    # 1.3 ELEMENTS deps_PV3_i/dVj (i = i_pv3, j = j_pq)
    if nPQ > 0:
        filtre = f_pv3*h_pq
        if filtre.any():
            gpj, bpj, vi, vj, cthpj, sthpj = select_dat(
                                       nzg, nzb, VI, None, cThIJ, sThIJ, filtre)
            depsdx = vi*(gpj*cthpj + bpj*sthpj)
            m = jacoi_pv3[nzi[filtre]]
            n = jacoi_q[nzj[filtre]]
            M, N, D, k = store_results(M,N,D,k, row=m, col=n, value=depsdx)
return M, N, D, k
```

Figure 3.5 Function that computes all $\tilde{\varepsilon}_{P(PV3)}$ derivatives (in the classic model)

3.6.3 Power and frequency regulation model

As far as the mismatch derivatives with respect to voltage phases and moduli are concerned, a comparison of table 2.2 and tables 2.5–2.7 shows that the regulation model has most of its Jacobian matrix expressions in common with the classic model. Therefore, the techniques presented in the previous section can also be applied to construct the Jacobian matrix in the current framework. One exception relates to $\frac{\partial \varepsilon_{Pi}}{\partial |V_i|}$ and $\frac{\partial \varepsilon_{Qi}}{\partial |V_i|}$, which exhibit slightly different load related terms. Nevertheless, the computation approach is unchanged, where the factors K_{Pi} and K_{Qi} are both available as node based arrays. Another exception is the presence of the derivatives of the reference bus mismatch. Their treatment presents no particular hurdle, however, since the reference bus mismatch has the same characteristics has a *Gen3* bus mismatch.

The derivatives with respect to P_r are directly obtainable from the node based array *bus_beta*, by applying the appropriate array of classified node indices, i.e. *nodei_sw*, *nodei_pva*, or *nodei_pv1* (containing respectively the *Ref*, the *Gen3* phase-*a*, and the *Gen1* node indices). For technical reasons, in the case of the *Gen3* buses the factor of -3 is added directly in the matrix terms, whereas in the case of the reference bus it is derived from the number of elements in *nodei_sw*.

The mismatch derivatives with respect to f share the same structure, constructed in terms of node based arrays. The expressions can thus be computed at once at the beginning of the Jacobian matrix procedure, together with the arrays of basic quantities (3.7)–(3.13). Depending on the derivatives under consideration, the values sought may be selected from the resulting array by means of the appropriate array of node indices (e.g. *nodei_pv1*). Regarding the array of voltages $|V_i|$ at all nodes, it is obtained from *VI* as follows:

$$vi = VI[p2]$$

where as seen before $p2 = cumsum(ifreq) - ifreq$ is the array that points to the first elements of all subsets of *nzi* defined by a distinct index value.

The main difficulty lies in computing the last row of the Jacobian matrix, namely the derivatives of ε_{pf} (assuming that $a_p \neq 0$). Fortunately, part of the methodology developed to compute the transferred power can be reused. In particular, the $3N_{tb} \times 6$ dimensional array

$$vyv = nu1 * conj(ytbr1 * nu2)$$

composed of elements of the form $V_i(Y_t)^*_{ik}V^*_k$ plays a central role in the computations (see sec. 3.5.2 for details on its construction). In fact, considering that the tie branches are generally not numerous, the derivatives of the power transferred are implemented in their complex forms, structured as shown in (2.121), (2.123), (2.125), and (2.127).

Starting with the voltage phase derivatives, the plan is to construct two arrays containing all basic results $\frac{\partial \varepsilon_{pf}}{\partial \theta_i}$, where i is a primary tie branch node in a first instance, and a secondary tie branch node in a second instance. Once those arrays are available, desired values can be selected from them, and inserted into the Jacobian matrix by means of boolean arrays that retain tie branch nodes of a specific category, i.e. *Gen3*, *P1*, or *Ld*. In essence, the approach is very similar to what is done to obtain the other phase derivatives of the Jacobian.

In order to obtain the derivatives of the form (2.121), reorganise *vyv* so as to line up horizontally all blocks associated with a distinct tie branch:

$$tau1 = concatenate(vsplit(vyv[:, 0:3], nTB), axis = 1) \tag{3.14}$$

where $[:, 0:3]$ picks the first three columns, and *vsplit* breaks *vyv* along the vertical axis into N_{tb} blocks. The axis along which the arrays are joined is determined through the *axis* parameter: '0' for vertical concatenation, and '1' for horizontal concatenation. The array of the $\frac{\partial \varepsilon_{pf}}{\partial \theta_i}$ results (i a primary tie branch node) is thus obtained as follows:

$$dPdthi = ap * (sum(tau1, axis = 0) - sum(vyv, axis = 1)).imag \tag{3.15}$$

In a way similar to the concatenation tool, summations on an array can be made along the '0' axis (vertical) or the '1' axis (horizontal).

Next, construct the counterpart of (3.14), which keeps the columns of *vyv* associated with the secondary tie branch nodes:

$$tau2 = concatenate(vsplit(vyv[:,3:],nTB),axis = 1) \tag{3.16}$$

As suggested by (2.123), summing the elements along the columns yields the array of $\frac{\partial \varepsilon_{pf}}{\partial \theta_i}$ results, where i is a secondary tie branch node:[7]

$$dPdthj = ap * sum(tau2,axis = 0).imag \tag{3.17}$$

In order to illustrate the process of inserting results into the Jacobian matrix, let us consider the $\frac{\partial \varepsilon_{pf}}{\partial \theta_i}$ elements, where i denotes primary tie branch nodes falling into the category *P1*. The row index m (equal to the number of θ and $|V|$ variables $+1$) is easily determined as the derivatives of ε_{pf} lie in the last row of the matrix. As to the corresponding column numbers, they can be obtained by converting the primary tie branch node indices of the type *P1* by means of the appropriate mapping array:

$$n = jacoi\_p1[nodei\_tb[f\_tbp1]]$$

The boolean array *f_tbp1* is used to extract the *P1* node indices from the array of all primary tie branch node indices *nodei_tb*. It also serves to get the corresponding derivatives out of *dPdthi*:

$$dPdthi[f\_tbp1]$$

which can directly be input into the Jacobian matrix through the function *store_results*. As a remark, the process is very similar in the case of the *Gen3* tie branch nodes, except that the

[7] In the program, *j* in *dPdthj* is used to indicate a secondary tie branch node index.

contributions from all three phases (in *dPdthi*) need to be added together due to the chain rule. Yet, the Jacobian column indices n are obtained by converting only the associated indices of phase a.

Continuing with the voltage modulus derivatives $\frac{\partial \varepsilon_{pf}}{\partial |V_i|}$, the results can be derived from *vyv* as well, though voltage moduli must be cancelled out to account for the differentiation process. Assuming that i labels primary tie branch nodes of the type *Ld*, as shown in (2.125) the product rule requires the cancellation of the left-hand side voltage moduli in one summand, and of the right-hand side voltage moduli in the other summand. The left-hand side cancellation can be reflected in a fairly simple way as follows:

$$lyv = (1/absolute(node\_v[nodei\_tb])).reshape(3*nTB,1)*vyv$$

where the elements of the left-hand side array are broadcasted across the columns of *vyv*. The right-hand side cancellation is operated by means of the modulus version of (3.6), that is

$$vyl = vyv/tile(absolute(node\_v[tbr\_ij]),(1,3)).reshape(3*nTB,6)$$

where the division is performed on an element-wise basis (like the operator *absolute*). In a way that parallels the use of (3.14) and (3.16) to compute the phase derivatives, the following remodelled sub-arrays of *vyl* are also necessary to obtain the voltage moduli derivatives:

$$tau3 = concatenate(vsplit(vyl[:,0:3],nTB),axis=1)$$
$$tau4 = concatenate(vsplit(vyl[:,3:],nTB),axis=1)$$

Referring to (2.125), given *lyv* and *tau3*, it is possible to obtain the array of $\frac{\partial \varepsilon_{pf}}{\partial |V_i|}$ results, where i is a primary tie branch node:

$$dPdVi = ap*(sum(lyv,axis=1)+sum(tau3,axis=0)).real \tag{3.18}$$

Similarly, referring to (2.127), from *tau4* it is possible to obtain the array of $\frac{\partial \varepsilon_{pf}}{\partial |V_i|}$ results, where i is a secondary tie branch node:

$$dPdVj = ap * sum(tau4, axis = 0).real \tag{3.19}$$

The insertion of the elements of (3.18) and (3.19) into the Jacobian matrix, by making use of boolean arrays that keep only *Ld* nodes out of the tie branch nodes, follows the same logic as explained above. Therefore, such details are omitted here.

In a final step, the lower-right corner element of the Jacobian matrix, i.e. a_f, can be inserted with ease, since the dimension of the matrix is easily obtainable from the program input data.

3.7 Computation of the correction terms of the unknowns

Following Newton's method (refer to sec. 1 of the appendix I), corrections to the unknowns in x can be obtained by solving the sytem of linear equations:

$$-\varepsilon = J\Delta x \tag{3.20}$$

where $J \equiv \nabla\varepsilon$ is the Jacobian matrix. Equivalently, the correction terms Δx follow from

$$\Delta x = -J^{-1}\varepsilon$$

The solver *scipy.sparse.spsolve* (from the *SciPy* library), which takes J and $-\varepsilon$ as its input, is used to solve (3.20) for Δx. It is efficient even with large sparse matrices. For example, the computing time to solve (3.20) for a test network composed of 9000 nodes was verified to be negligible compared to the time required to complete other steps of the program (e.g. reading and organising the input data, construction of the Jacobian matrix).

3.8 Variable updates

Once the array for Δx has been computed, it is used to correct its respective voltage phases and moduli. In the regulation model, additional corrections are also made to the variables P_r and f: adding the penultimate element of x to P_r, and the last element of x to f.

In order to respect the order of the phase variables in (2.101) or (2.110), and to locate the corresponding voltages in *node_v*, the array

$$nodei\_p = concatenate([nodei\_pva, nodei\_pv1, nodei\_pq]) \tag{3.21}$$

is used. It consists in the concatenation of the ordered sets of node indices for the *PV3* phase-*a*, *PV1*, and *PQ* nodes (equivalently *Gen3* phase-*a*, *Gen1*, and *Ld* nodes in the regulation model), as those nodes each have an associated voltage phase variable. The voltage moduli are similarly identified with the array of *PQ* (*Ld*) node indices, i.e. *nodei_pq*.

Hence, the phase updates $V_i = |V_i|\, e^{j(\theta_i + \Delta\theta_i)}$, where i points to either a *PV* (*Gen*) or *PQ* (*Ld*) node, are implemented as follows:

$$\begin{aligned}
node_v[nodei_p] &= absolute(node_v[nodei_p]) * \\
&\qquad exp(1j * (angle(node_v[nodei_p]) + dx[:nTh])) \\
node_v[nodei_pvb] &= absolute(node_v[nodei_pva]) * \\
&\qquad exp(1j * (angle(node_v[nodei_pva]) - 120 * pi/180)) \\
node_v[nodei_pvc] &= absolute(node_v[nodei_pva]) * \\
&\qquad exp(1j * (angle(node_v[nodei_pva]) + 120 * pi/180))
\end{aligned}$$

where *nTh* is the total number of phase variables, and $dx[:nTh]$ is the sub-array of *dx* that contains only the phase corrections. The indices of the *PV3* (*Gen3*) phase *b* and *c* nodes are simply shifted from the ones of the phase *a* nodes, that is $nodei\_pvb = nodei\_pva + 1$ and $nodei\_pvc = nodei\_pva + 2$.

Similarly, the moduli updates $V_i = (|V_i| + \Delta|V_i|)\, e^{j\theta_i}$, where i points to a PQ (Ld) node, are implemented in the following way:

$$node\_v[nodei\_pq] = (absolute(node\_v[nodei\_pq]) + dx[nTh:nThV]) * \\ exp(1j * angle(node\_v[nodei\_pq]))$$

where $nThV$ is the total number of voltage phase and moduli variables, and $dx[nTh:nThV]$ is the sub-array of dx that contains only the moduli corrections.

3.9 The tap changing procedure

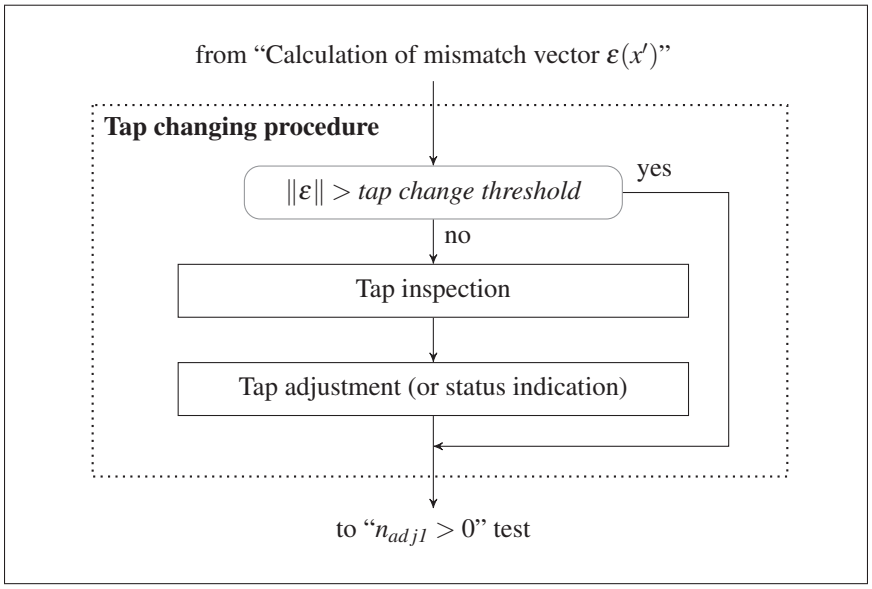


Figure 3.6 Subroutines that make up the tap changing procedure

As shown in fig. 3.6, the tap changing procedure consists of two sets of actions, carried out on the condition that the elements of the mismatch vector are sufficiently small. In fact, a maximum threshold (10^{-3}) for the norm of the vector of mismatches is imposed so as to operate the tap changer analysis based on a sufficiently reliable network system solution. In the *tap inspection* subroutine, the voltages at the controlled buses and the statuses of the respective

tap changers are examined, following which action codes are generated. In the *tap adjustment* subroutine, actions are taken in response to the action codes.

3.9.1 Tap inspection

The objective of the tap inspection process is to examine and to report the status of each tap changer, based on which actions can be taken in a subsequent step. The status is expressed through an action code, constructed by combining bit-like elements, each based on a specific criterion. The resulting combination is then converted to a number in the decimal system in a way similar to the conversion of a binary number into a decimal number. This allows to avoid the use of multiply imbedded *if* statements in the program.

The action code C_{adj} is defined as follows:

$$C_{adj} = \begin{cases} 0 & \text{if conditions are satisfied under the status quo} \\ 1 + \psi_0\, 2^0 + \psi_1\, 2^1 + \psi_2\, 2^2 + \psi_3\, 2^3 & \text{otherwise} \end{cases}$$

where each ψ coefficient is associated with a criterion. First, ψ_0 expresses the relative value of the controlled bus voltage V_c with respect to its allowed bounds:

$$\psi_0 = \begin{cases} -1 & \text{if } V_{min} \leq V_c \leq V_{max} \\ 0 & \text{if } V_c < V_{min} \\ 1 & \text{if } V_c > V_{max} \end{cases}$$

Second, ψ_1 reflects the location of the controlled bus with respect to the transformer branch:

$$\psi_1 = \begin{cases} 0 & \text{if it is the primary bus} \\ 1 & \text{if it is the secondary bus} \end{cases}$$

Note that the case where the controlled bus is neither the primary nor the secondary bus is verified and excluded at the start of the program, i.e. when reading and organizing the input data. Third, based on sec. 2.3.6.1, the necessary tap increment can be written as $dn = (-1)^{\psi_0+\psi_1}$, which is used along with the actual tap position n_x to determine whether a tap change is possible or not. The result is expressed by ψ_2:

$$\psi_2 = \begin{cases} 1 & \text{if } (dn > 0 \text{ and } n_x \geq n_t) \text{ or } (dn < 0 \text{ and } n_x \leq 1) \text{ (not possible)} \\ 0 & \text{otherwise (possible)} \end{cases}$$

where n_t is the total number of taps. Lastly, given that two consecutive tap changes of different directions lead to a unit increment of the oscillation variable Θ, the *bit* ψ_3 indicates whether a tap changer is in an oscillation state or not:

$$\psi_3 = \begin{cases} 1 & \text{if } \Theta > 2 \text{ (oscillation)} \\ 0 & \text{if } 0 \leq \Theta \leq 2 \text{ (no oscillation)} \end{cases}$$

The threshold of two oscillations is set out of pure choice. Besides, to track the number of necessary tap changes per iteration, whenever $C_{adj} \neq 0$ the value of n_{adj2} is incremented by one.

Algorithm 3.1 shows how the above method is implemented in the program. A description of the array variables is given in Table 3.1. The top entries are branch based arrays constructed directly from the input data; their contents are thus fetched by branch indices. The bottom entries are based on the sub-array of the branch indices associated with a tap changing functionality (i.e. *ibran_tc*). In particular, the array *tadj_case* is composed of all computed codes C_{adj} of the transformers equipped with a tap changer. Table 3.2 contains the scalar variables.

A few remarks are then in order. To begin with, the inspection process is repeated with every tap changer (correspondingly every branch index of *ibran_tc*), and the iteration variable k is used to fetch elements in arrays structured as *ibran_tc*. For a three-phase controlled bus, *vstat*

is determined with the help of array *tau*:

$$\tau = \begin{matrix}[[V_{c(i)} - V_{min}, & V_{c(i+1)} - V_{min}, & V_{c(i+2)} - V_{min}\,],\\ [\,V_{max} - V_{c(i)}, & V_{max} - V_{c(i+1)}, & V_{max} - V_{c(i+2)}\,]]\end{matrix}$$

where $V_{c(i)}$, $V_{c(i+1)}$, and $V_{c(i+2)}$ are the voltage moduli at the controlled bus (grouped in the array *vc3* in the program), and V_{min} and V_{max} are the specified minimum and maximum voltage moduli for the controlled bus. The presence of a negative τ element is thus indicative of a voltage bound violation. The τ construction allows to identify the worst case, by seeking the smallest (most negative) value. This is achieved through the function *unravel_index*, which returns the row (η_0) and column (η_1) indices of the minimum value. The remainder of the algorithm is self-explanatory.

3.9.2 Tap adjustment

Algorithm 3.2 shows how the results of the tap inspection process are used to implement or not tap adjustments. The subroutine's input consists of the array of C_{adj} values for all tap changer branches, the bus admittance matrix, as well as a set of other variables mainly related to the changing of tap positions, represented by s_{in} for simplicity. Essentially, the subroutine operates in reaction to the C_{adj} values, and generates a correction matrix for every required tap change. Its main output variable is the (modified) Y_{bus} matrix. Yet it also returns other (modified) parameters denoted by s_{out}, as well as the parameter n_{adj1} that keeps track of the number of adjustments actually performed.

The subroutine starts by initializing n_{adj1} to zero. The correction matrix Y_c is also initialized as an empty sparse matrix of the same size as Y_{bus}. In fact, as shown in line 19, the correction matrix should be constructed in such a way that its addition to the bus admittance matrix reflects the effect of all operated tap changes, that is

$$Y'_{bus} = Y_{bus} + Y_c \tag{3.22}$$

Here and in the remainder of the section, the prime symbol ($'$) is used to denote a new version of a variable.

Next, the subroutine iterates over the branches with a tap changing functionality, in every instance deriving its action from the value of C_{adj}. The various cases that may occur are commented in the algorithm. Unless the voltage conditions are fulfilled without the need for a tap adjustment (case $C_{adj} = 0$), the subroutine relies on the function *adj_routine* to perform the tasks associated with a change (or not) in tap position. Several input elements are thus required: in order t_{in}, which stands for a set of global variables not listed (for simplicity), *br*, *k*, *yc*, the required tap increment, and a remark to be printed in the report of results. Upon the completion of its instructions, *adj_routine* returns a set of (modified) global variables (denoted by t_{out}) as well as a new correction matrix that incorporates, as applicable, the effect of a tap change.

In every iteration, if *dn* is non-zero, then *adj_routine* adjusts the tap position ($n'_x = n_x + dn$), and it updates the transformation ratio a_1 based on (2.87). As will be seen below, the corresponding row element of *bran_ymat* is also updated. Furthermore, the coherence of the tap changing activity is verified by incrementing up the oscillation variable if the current tap change is done in a direction opposite to the corresponding tap change of the previous iteration, i.e. $\Theta' = \Theta + 1$, if $dn \times dn_0 = -1$, where dn_0 represents the preceding tap change.

A variation in the transformation ratio requires a local modification of Y_{bus}, in accordance with equations (2.89) and (2.90). In particular, the elements of the corresponding branch admittance matrix, stored in *bran_ymat*, need to be modified, and reinserted in Y_{bus}. Taking the case of the three-phase transformer equipped with a tap changer (the case of the single-phase transformer is handled similarly), the branch admittance matrix can be divided into four 3×3 block matrices:

$$Y_{br} = \begin{bmatrix} Y_{br}^{I} & Y_{br}^{II} \\ Y_{br}^{III} & Y_{br}^{IV} \end{bmatrix}$$

where, referring to (2.90),

$$Y_{br}^{I} = \frac{1}{(a_1)^2}(Y_{tr})_{KL}$$

$$Y_{br}^{II} = \frac{1}{a_1\, a_2}(Y_{tr})_{K\ell}$$

$$Y_{br}^{III} = \frac{1}{a_2\, a_1}(Y_{tr})_{kL}$$

$$Y_{br}^{IV} = \frac{1}{(a_2)^2}(Y_{tr})_{k\ell}$$

Using the above definition, the correction due to the new transformation ratio a_1' is realized through the matrix

$$dY_{br} = \begin{bmatrix} (a_1^2/a_1'^2 - 1)\, Y_{br}^{I} & (a_1/a_1' - 1)\, Y_{br}^{II} \\ (a_1/a_1' - 1)\, Y_{br}^{III} & 0 \end{bmatrix}$$

which when added to Y_{bus} will result in cancelling the contributions of the old Y_{br} elements, and in replacing them with their corrected versions. There remains to locate the elements of dY_{br} into dY_c, a sparse matrix of the same dimension as Y_{bus}, such that

$$Y_c' = Y_c + dY_c \tag{3.23}$$

Since the primary and secondary node numbers of the branch in question are known, this is done easily by identifying their corresponding node indices, and arranging them into row and column index arrays, in such a way as to match a third array containing the elements of dY_{br}.[8] In other words, the methodology used to construct the bus admittance matrix can be reused here to obtain dY_c out of dY_{br}.

The process of adding incremental matrices dY_c to an intermediate version of Y_c is repeated for every branch so as to obtain in the end the full correction matrix Y_c of (3.22). Note that such incremental process is possible due to the linear nature of the construction of Y_{bus} from the Y_{br} elements.

[8] The sparse matrix type *csr_matrix* is also used to construct dY_c.

3.10 Power computations

At this point, the system of equations (3.20) has been solved, and the voltages at all nodes are known. Furthermore, the bus admittance matrix has been determined, and perhaps adjusted to account for tap changes. Consequently, all necessary information is available to compute the various powers in the system.

The main challenge is to determine efficiently the power that goes into the various branches or the zigzag earthing transformers of the network. In every case, the computations are based on an equation of the form (2.3). Furthermore, in order to achieve optimal computation speed, a vectorial approach is utilized in every case, using a set of common basic principles. In particular, the method reuses some of the work carried out to obtain the bus admittance matrix. Due to the similarity of the computations to determine the powers going into the different types of branches and zigzag earthing transformers, only the approach to obtain the powers into the three-phase branches will be described explicitly here. From the information provided, it should be straightforward to obtain the power values entering the other components of the network.

The array of left-hand side voltages in (2.3), call it *vl*, is obtained by unravelling the array of indices $ij_{3\phi}$ (3.1) in the column-major order, and by applying the resulting one-dimensional array of indices to the array of voltages *node_v*. The branch admittance matrix elements are readily available from *bran_ymat*, and can be reshaped into a $6n \times 6$ dimensional array y, equivalent to stacking vertically the 6×6 admittance matrices of the branches. The symbol n represents the number of three-phase branches. As regards the right-hand side voltages, they are repeated and arranged into another $6n \times 6$ array *vr*, so as to be element-wise multipliable with the array of branch admittance matrix elements. The array of right-hand side voltages is thus created in a few simple steps:

a. Tile $ij_{3\phi}$ vertically six times;

b. Unravel the resulting array in the column-major order;

c. Reshape the resulting array to obtain a $6n \times 6$ dimensional array of node indices;

d. Apply such array to *node_v*.

The part $\sum_{k=1}^{n} Y_{ik}^{*} V_k^{*}$ is thus obtained by applying the complex conjugate to the element-wise product $y * vr$, and then by summing along the horizontal axis (axis 1). The result is a one-dimensional array that is mutiplied element-wise with *vl* to obtain the part of (2.3) that applies to the three-phase branches.

The powers going into the single-phase branches and zigzag earhting transformers can be obtained in a similar manner by using the arrays (3.2) and (3.3) respectively.

In order to compute the power into the uncoupled shunt elements, the following formula is used:

$$\begin{aligned}(S_{sh})_i &= |V_i|^2 \, (Y_{sh})_i^{*} \\ &= |V_i|^2 \, \{(G_{sh})_i - j(B_{sh})_i\}\end{aligned}$$

where i is a node index. The computation presents no particular difficulty, because the voltages are known at all nodes, and the shunt conductances and susceptances are readily available from the input data.

The loads can also be computed from the known voltages (and network frequency), based on the load model formulae (2.99) and (2.100) in the classic model, and (2.107) and (2.108) in the power and frequency regulation model. The other parameters characterising the load are also at hand since the performance of the routine that reads and organizes the input data.

Analogously, the generated active power at the *Ref* and *Gen* buses (in the power and frequency regulation model) is computed from the network frequency and the power insufficiency of the generators, based on the generator model formula (2.109). The parameters $P_{G\text{-}set\,i}$, R_i, and β_i are then readily available from the input data. In the classic model, the generated active power at a *PV* bus consists only of the specified parameter $P_{G\text{-}set\,i}$.

Finally, it should be recalled from sec. 2.4.3.4 and 2.5.3.4 that the generated reactive power at the *PV* and *Gen* buses, as well as the generated apparent power at the swing bus, are adjusted quantities which are readily available at this stage of the computation.

Algorithm 3.1 Tap inspection

Input : Elements of table 3.1 (excl. *dn0* and *tadj_case*), *node_v*, *oscillation*
Output : *nadj2*, *tadj_case*

nadj2 = 0
tadj_case = *zeros(ibran_tc.size)* # Default 0: no adj. needed
$k = -1$
for *br* **in** *ibran_tc* **do**
    $k = k + 1$
    # 0. Determination of vstat (Vctrl vs Vmin, Vmax)
    $vstat = 0$
    if $bran\_type[br] == 1010$ **:** # Single-phase transformer
        $vc = abs(node\_v[ctrlj\_ptr[k, 0]])$
        if $vc < br\_vlim[br, 0]$ **:**
            $vstat = 1$ # vstat = 1: Vctrl < Vmin
            $vctrl[k] = [vc, ctrlj\_ptr[k, 0]]$
        elif $vc > br\_vlim[br, 1]$ **:**
            $vstat = 2$ # vstat = 2: Vctrl > Vmax
            $vctrl[k] = [vc, ctrlj\_ptr[k, 0]]$
    else :
        $vc3 = absolute(node\_v[ctrlj\_ptr[k]])$
        $tau = vstack((vc3 - br\_vlim[br, 0],\ br\_vlim[br, 1] - vc3))$
        $eta0,\ eta1 = unravel\_index(tau.argmin(), tau.shape)$
        if $(tau < 0).any()$ **:**
            $vstat = 1 + eta0$
            $vctrl[k] = [vc3[eta1], ctrlj\_ptr[k, eta1]]$
    $psi0 = vstat - 1$ # psi0 = -1 when vstat = 0 is inconsequential
    # 1. Ctrl bus junction: psi1 = 0, primary; 1, secondary
    $psi1 = 0$ **if** $(br\_ctrlj[br] == bran\_I[br])$ **else** 1
    # 2. Tap change availability: psi2 = 0,available; 1,unavailable
    $dn = (-1) * *(psi0 + psi1)$ # Necessary tap change (dn = +1 or -1)
    $psi2 = 1$ **if** $(dn > 0$ **and** $nx[k] \geq br\_ntaps[br])$ **or** $(dn < 0$ **and** $nx[k] \leq 1)$ **else** 0
    # 3. Tap oscillation
    $psi3 = 1$ **if** $oscillation[k] > 2$ **else** 0
    if $vstat == 0$ **and** $(1 \leq nx[k] \leq br\_ntaps[br])$ **and** $oscillation \leq 2$ **:**
        $tadj\_case[k] = 0$ # No adjustment necessary
    else :
        $tadj\_case[k] = 1 + psi0 + 2 * psi1 + 4 * psi2 + 8 * psi3$
        $nadj2 = nadj2 + 1$
end

Table 3.1 Description of the main array variables used in the tap changing procedure

Array	Symbol	Definition
bran_I		Branch primary node indices (phase *a* node indices for three-phase buses)
bran_type		Branch type identification codes
br_ctrlj		Controlled bus numbers ('0' for no voltage control functionality)
br_ntaps	n_t	Numbers of taps (the symbol designates any element of the array)
br_vlim		Minimum (first column) and maximum (second column) allowed voltage moduli at the controlled buses
ctrlj_ptr		Node indices corresponding to the controlled bus numbers (phase *a*, *b*, and *c* node indices in the first, second, and third columns respectively). The node index for a single-phase controlled bus is stored in the first column, irrespective of its associated phase.
dn0		Tap changes ($+1$, -1, or 0) performed during a previous iteration
ibran_tc		Branch indices of transformers with a tap changer
nx	n_x	Tap positions (the symbol designates any element of the array)
tadj_case	C_{adj}	Action codes (the symbol designates any element of the array)
vctrl		Voltage moduli that fail to satisfy the voltage condition (first column) (value of '0' otherwise), and the corresponding node indices (second column)

Table 3.2 Definitions of the main scalar variables used in the tap changing procedure

Variable	Symbol	Definition
misadj	n_{mis}	Counter to track the number of times that all necessary tap adjustments cannot be performed. A value of 3 or more leads to a status of condition violation.
nadj1	n_{adj1}	Counter of the adjustments actually performed (per iteration)
nadj2	n_{adj2}	Counter of the necesssary tap adjustments (per iteration)
nnode	N	Total number of nodes in the network
oscillation	Θ	Counter of successive tap changes of opposite signs. Once the value has reached a threshold (e.g. 3), the tap changing process is halted.

Algorithm 3.2 Tap adjustment and status

Input : s_{in}, yb, $tadj\_case$
Output : s_{out}, yb, $nadj1$

$nadj1 = 0$
$yc = csr\_matrix((nnode, nnode))$ # Ybus correction matrix
$k = -1$
for br **in** $ibran\_tc$ **do**
    $k = k + 1$
    if $tadj\_case[k] == 0$ **:**
        continue # 0: No adjustment necessary for this branch
    if $tadj\_case[k]$ **in** $(1, 4)$ **:**
        # 1: Vc < Vmin, Vc on primary side, no oscillation, dn = +1
        # 4: Vc > Vmax, Vc on secondary side, no oscillation, dn = +1
        $t_{out}, yc = adj\_routine(t_{in}, br, k, yc, 1,$ 'dn = +1'$)$
    if $tadj\_case[k]$ **in** $(2, 3)$ **:**
        # 2: Vc > Vmax, Vc on primary side, no oscillation, dn = -1
        # 3: Vc < Vmax, Vc on secondary side, no oscillation, dn = -1
        $t_{out}, yc = adj\_routine(t_{in}, br, k, yc, -1,$ 'dn = -1'$)$
    if $tadj\_case[k]$ **in** $(5, 8)$ **:**
        # 5: Vc < Vmin, Vc on primary side, no oscillation, dn = 0 (nx = n_max)
        # 8: Vc > Vmax, Vc on secondary side, no oscillation, dn = 0 (nx = n_max)
        $t_{out}, yc = adj\_routine(t_{in}, br, k, yc, 0,$ 'dn = 0, max. tap position reached'$)$
    if $tadj\_case[k]$ **in** $(6, 7)$ **:**
        # 6: Vc > Vmax, Vc on primary side, no oscillation, dn = 0 (nx = n_min)
        # 7: Vc < Vmin, Vc on secondary side, no oscillation, dn = 0 (nx = n_min)
        $t_{out}, yc = adj\_routine(t_{in}, br, k, yc, 0,$ 'dn = 0, min. tap position reached'$)$
    if $tadj\_case[k] > 8$ **:**
        $t_{out}, yc = adj\_routine(t_{in}, br, k, yc, 0,$ 'Oscillating tap, adjustment halted'$)$
end
$yb = yb + yc$

CHAPTER 4

SAMPLE CASES OF VALIDATION AND DISCUSSION

A large number of test cases were carried out to validate the program *nr3r*. In this chapter, some of them are presented to illustrate the primary features of the program, that is the ability to solve unbalanced network systems comprising both three-phase and single-phase components, to solve systems that include transformer tap changers, and to perform the regulation of transferred power and network frequency. In each case, the balance in power is met at every node, and the imposed conditions are fulfilled. In addition, a large system composed of 3000 three-phase buses is used to evaluate the computation efficiency of the program. Calculations are everywhere performed in the per unit system.

4.1 Basic unbalanced network

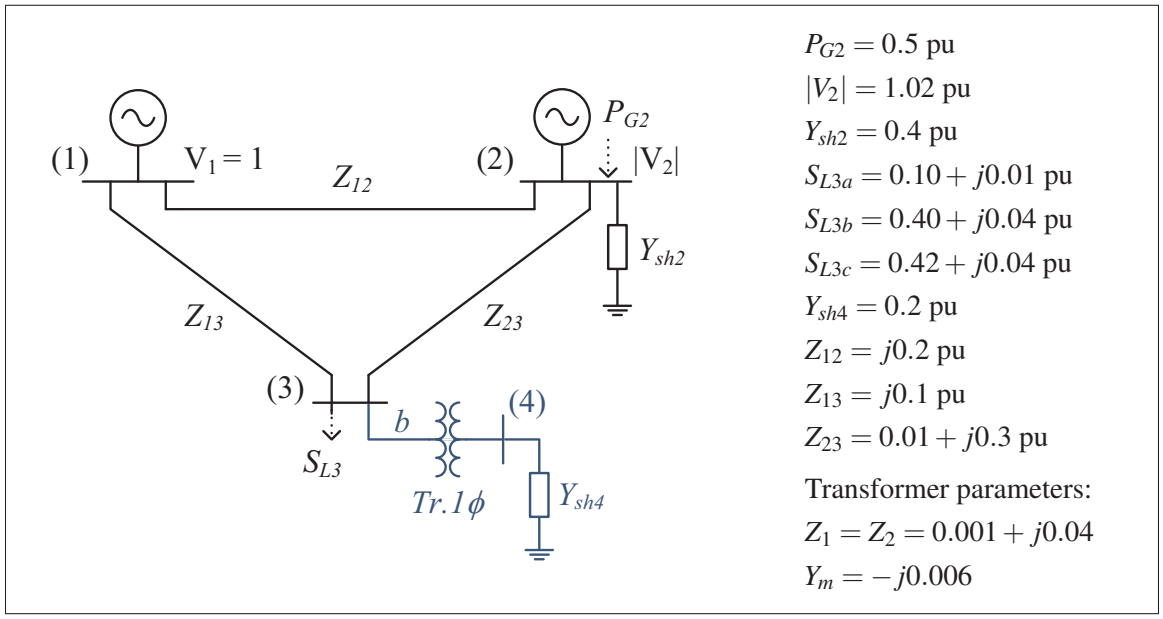


Figure 4.1 Unbalanced network including a single-phase transformer

Figure 4.1 shows an unbalanced network composed of three three-phase buses and one single-phase bus. A single-phase transformer links node #3*b* to the single-phase bus #4. The analysis

```
_____________________________ NR3R Ver. 1.1 (01 February 2016) _____________________________
Time: 12.06.2016 13:27
Path: /home/saubert2/Documents/Python/nr3r

1. INPUT SUMMARY
Input file: ../test_files/ex_msca/nr3c_test_94.dat
Number of nodes:                          10
Number of buses:                           6 (2 three-ph. and 4 single-ph.)
Number of branches:                        4 (3 three-ph. and 1 single-ph.)
Number of zigzag earth. transformers:      0

2. CONVERGENCE BEHAVIOUR
Model: Classic (ideal swing)
Calculation method: Newton-Raphson

It.        eps_Pmax         (Node)         eps_Qmax         (Node)         ||eps||          Adj.
  0  4.1778024417314186e-01   (3c)  -5.6592674805772132e-02   (3a)  6.6835435560208878e-01  -/-
  1 -6.1164068997304755e-04   (3c)   1.4997704901081564e-02   (3b)  1.6495764916039685e-02  -/-
  2  1.0807142792890190e-06   (3b)   1.8673305720105782e-05   (3b)  1.9135824686629867e-05  -/-
  3  4.3297587737356480e-12   (3b)   2.9454182148835883e-11   (3b)  3.0190907085944791e-11  -/-

Calculation time until reaching tolerance threshold for eps_max: 0.0076s
```

Figure 4.2 Report of results (part one) for the network of fig. 4.1

is conducted in the classic model by taking bus #1 as the swing bus, bus #2 as a *PV* bus, and buses #3 and #4 as *PQ* buses. Furthermore, as will be the case in the remainder of the chapter, the short and decoupled line approximation is used. By initializing all voltage moduli to one, the phase *a* voltage phases to 0°, and the phase *b* voltage phase at bus #4 to -120° (a detailed description of the input data structure is given in appendix III), the results shown in fig. 4.2 and fig. 4.3 are obtained.

The second section of the report displays a rapid convergence of the computations, where the tolerance threshold of 10^{-10} for the maximum mismatch amplitude is reached in three iterations.

The third section shows the voltages across the network; the columns *|V0|* and *th0* contain the voltage moduli and phases used to initialize the iterative process, whereas the columns *|V|* and *th.* contain the final values. As expected, due to the unbalanced nature of the problem, the voltages at bus #3 have different magnitudes, and their phases are not evenly distributed. As regards the voltages at buses #1 and #2, they respect the imposed conditions.

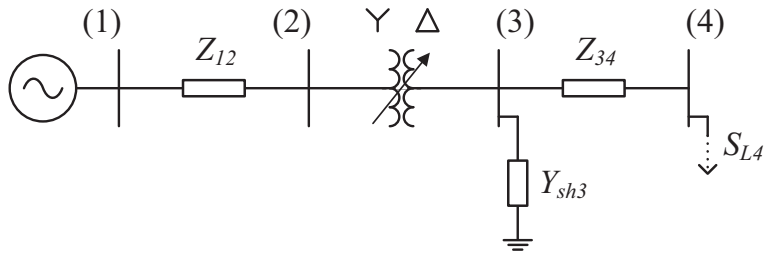
```
3. VOLTAGES
Bus#   Name   Type Ph. |V0| (pu) th0 (deg)    |V| (pu)       |V| (kV)       th. (deg)    Eb (kV)
---- -------- ---- --- --------- --------- -------------- -------------- --------------- -------
   1   SW1    REF   a   1.00000     0.0000   1.0000000000   1.0000000000    0.0000000000  1.000
                    b   1.00000  -120.0000   1.0000000000   1.0000000000 -120.0000000000  1.000
                    c   1.00000   120.0000   1.0000000000   1.0000000000  120.0000000000  1.000
   2   PV2    GEN   a   1.02000     0.0000   1.0200000000   1.0200000000   -0.0930247877  1.000
                    b   1.02000  -120.0000   1.0200000000   1.0200000000 -120.0930247877  1.000
                    c   1.02000   120.0000   1.0200000000   1.0200000000  119.9069752123  1.000
   3   PQ3a   LOAD  a   1.00000     0.0000   1.0041699652   1.0041699652   -0.4420068973  1.000
   3   PQ3b   LOAD  b   1.00000  -120.0000   0.9999357774   0.9999357774 -122.5806549706  1.000
   3   PQ3c   LOAD  c   1.00000   120.0000   1.0012560993   1.0012560993  118.1907170006  1.000
   4   PQ4b   LOAD  b   1.00000  -120.0000   0.9991684753   0.9991684753 -123.4964893294  1.000

4. POWERS AND THEIR BALANCE AT EVERY NODE
Node    1a: S_br (Line 1-2           )  =     0.0082802994 + i(   -0.0999932781) pu
            S_br (Line 1-3           )  =     0.0774656793 + i(   -0.0414008471) pu
            S_G                         =     0.0857459787 + i(   -0.1413941252) pu
            eps_S                       =       0.0000e+00 + i(      8.8818e-16) pu
Node    1b: S_br (Line 1-2           )  =     0.0082802994 + i(   -0.0999932781) pu
            S_br (Line 1-3           )  =     0.4502280700 + i(    0.0107832847) pu
            S_G                         =     0.4585083694 + i(   -0.0892099934) pu
            eps_S                       =      -7.7716e-16 + i(      1.4433e-15) pu
Node    1c: S_br (Line 1-2           )  =     0.0082802994 + i(   -0.0999932781) pu
            S_br (Line 1-3           )  =     0.3161235607 + i(   -0.0075693120) pu
            S_G                         =     0.3244038601 + i(   -0.1075625901) pu
            eps_S                       =       8.3267e-16 + i(      4.7184e-16) pu
Node    2a: S_br (Line 1-2           )  =    -0.0082802994 + i(    0.1020067219) pu
            S_br (Line 2-3           )  =     0.0225663505 + i(    0.0531332375) pu
            S_sh                        =     0.4161600000 + i(    0.0000000000) pu
            S_G                         =     0.5000000000 + i(    0.1551399594) pu
            eps_P (sum abc) + i eps_Q   =      -7.8604e-14 + i(     -6.9389e-16) pu
Node    2b: S_br (Line 1-2           )  =    -0.0082802994 + i(    0.1020067219) pu
            S_br (Line 2-3           )  =     0.1497774176 + i(    0.0664296778) pu
            S_sh                        =     0.4161600000 + i(    0.0000000000) pu
            S_G                         =     0.5000000000 + i(    0.1684363997) pu
            eps_P (sum abc) + i eps_Q   =      -7.8604e-14 + i(     -6.9389e-16) pu
Node    2c: S_br (Line 1-2           )  =    -0.0082802994 + i(    0.1020067219) pu
            S_br (Line 2-3           )  =     0.1040171301 + i(    0.0617891744) pu
            S_sh                        =     0.4161600000 + i(    0.0000000000) pu
            S_G                         =     0.5000000000 + i(    0.1637958963) pu
            eps_P (sum abc) + i eps_Q   =      -7.8604e-14 + i(     -5.2736e-16) pu
Node    3a: S_br (Line 2-3           )  =    -0.0225343207 + i(   -0.0521723433) pu
            S_br (Line 1-3           )  =    -0.0774656793 + i(    0.0421723433) pu
            S_L                         =     0.1000000000 + i(    0.0100000000) pu
            eps_S                       =      -1.3878e-17 + i(      4.6265e-15) pu
Node    3b: S_br (Line 1-3           )  =    -0.4502280700 + i(    0.0094988747) pu
            S_br (Tr1p 3-4           )  =     0.1997474505 + i(    0.0091896909) pu
            S_br (Line 2-3           )  =    -0.1495193805 + i(   -0.0586885656) pu
            S_L                         =     0.4000000000 + i(    0.0400000000) pu
            eps_S                       =       4.3293e-12 + i(      2.9453e-11) pu
Node    3c: S_br (Line 2-3           )  =    -0.1038764393 + i(   -0.0575684520) pu
            S_br (Line 1-3           )  =    -0.3161235607 + i(    0.0175684520) pu
            S_L                         =     0.4200000000 + i(    0.0400000000) pu
            eps_S                       =      -2.3037e-14 + i(      7.9463e-13) pu
Node    4b: S_br (Tr1p 3-4           )  =    -0.1996675284 + i(    0.0000000000) pu
            S_sh                        =     0.1996675284 + i(    0.0000000000) pu
            eps_S                       =      -2.9600e-12 + i(      3.9741e-12) pu

_____________________________________ END OF REPORT _____________________________________
```

Figure 4.3 Report of results (part two) for the network of fig. 4.1

The fourth section gives for every node the complex power (in single-phase pu) that enters every branch connected to the node. The conditions on the active power generation at bus #2 and the load conditions at bus #3 are confirmed to be fulfilled. In addition, the balance in

power at every node is provided. All values are seen to vanish, up to residuals that depend on the imposed convergence threshold and the computing precision. As a final remark, the generated active power at each of the nodes of bus #2 may differ, and thus does not necessarily correspond to the printed 0.5 pu values. In fact, for practical reasons, the generated active power at the node level of a *PV* bus is intentionally printed to be the third of the total generated active power at the bus (in single-phase pu). Nevertheless, the sum of the printed generated active power at the three nodes (here 1.5 single-phase pu, or equivalently 0.5 three-phase pu) always corresponds to the generated active power at the bus.

4.2 Tap changing operation



$V_1 = 1$ pu	Transformer and tap changer parameters:
$Y_{sh3} = 0.7 + j0.1$ pu	$Z_1 = Z_2 = j0.05$ pu, $Y_m = 0$
$S_{L4} = 0.9 + j0.1$ pu	$\lvert V_{3\,min}\rvert = 0.93$ pu, $\lvert V_{3\,max}\rvert = 1.07$ pu
$Z_{12} = 0.01 + j0.1$ pu	$a_{1\,min} = 0.9$, $a_{1\,init} = 1$, $a_{1\,max} = 1.1$
$Z_{34} = 0.01 + j0.1$ pu	$n_t = 11$

Figure 4.4 Network including a tap changing *Yd1* transformer controlled based on the voltage at bus #3

The system of fig. 4.4 is used to demonstrate the implementation of the tap changer mechanism into *nr3r*. The network consists of four three-phase buses, connected by two three-phase lines and one transformer having a *Yd1* winding connection. A tap changer is installed on the star configuration of the transformer. The load, the lines, and the shunt elements are balanced. The analysis is conducted in the classic model by taking bus #1 as the swing bus, and the other

```
2.1 CONVERGENCE BEHAVIOUR
Model: Classic (ideal swing)
Calculation method: Newton-Raphson

It.        eps_Pmax         (Node)         eps_Qmax         (Node)         ||eps||          Adj.
  0  9.0000000000000002e-01   (4a)  -1.0000000000000679e-01   (3c)  1.9899748742132399e+00  0/0
  1  5.0667954621725908e-02   (4b)   2.4028298274428597e-01   (2b)  5.1222065826196705e-01  0/0
  2  2.3183741832572657e-02   (4b)   1.2755227596694434e-02   (3c)  5.4979014771539055e-02  0/0
  3  7.1336872839267151e-04   (4c)   2.2995124448302564e-04   (3b)  1.5268247584690397e-03  0/0
  4 -3.0172287390029168e-02   (3c)   1.8792450421647428e-01   (2c)  4.4290514467767117e-01  1/1
  5  1.8885248779433317e-03   (4a)   4.5822035339218173e-03   (3c)  8.8517559415566271e-03  0/0
  6 -3.1364304534949608e-02   (3c)   1.9948987433350510e-01   (2a)  4.7113173222500954e-01  1/1
  7  1.8989322983594681e-03   (4b)   4.9057489010305363e-03   (3a)  9.3645286135976905e-03  0/0
  8  1.2477085022055689e-05   (4a)   8.6354252398125986e-06   (3b)  2.9730407853979747e-05  0/0
  9  1.3712664337361957e-10   (4a)   5.4902239610061295e-11   (3c)  3.0254076873779522e-10  0/0
 10  2.5262456020752189e-15   (3c)  -3.1232288128858716e-15   (2c)  7.6452066630898633e-15  0/0

Calculation time until reaching tolerance threshold for eps_max: 0.0195s

2.2 TAP CHANGING ACTIVITY
It  Adj.   Branch   Ctrl# |Vmin|  |Vmax|  |Vctrl0| (ph) n0 n1              Remarks
-- ----- ---------- ----- ------- ------- ------------ -- -- ---------------------------------
 4  1/1  TrYd1 TC       3 0.93000 1.07000  0.900532 (b)  6  5 dn = -1
 6  1/1  TrYd1 TC       3 0.93000 1.07000  0.921943 (b)  5  4 dn = -1

3. VOLTAGES
Bus#   Name   Type Ph. |V0| (pu) th0 (deg)    |V| (pu)       |V| (kV)       th. (deg)    Eb (kV)
---- -------- ---- --- --------- --------- -------------- -------------- --------------- -------
   1   SW1    REF   a   1.00000     0.0000   1.0000000000   1.0000000000    0.0000000000  1.000
                    b   1.00000  -120.0000   1.0000000000   1.0000000000 -120.0000000000  1.000
                    c   1.00000   120.0000   1.0000000000   1.0000000000  120.0000000000  1.000
   2   PQ2    LOAD  a   1.00000     0.0000   0.9303742314   0.9303742314   -9.2540005935  1.000
                    b   1.00000  -120.0000   0.9303742314   0.9303742314 -129.2540005935  1.000
                    c   1.00000   120.0000   0.9303742314   0.9303742314  110.7459994065  1.000
   3   PQ3    LOAD  a   1.00000   -30.0000   0.9439725674   0.9439725674  -48.9036347542  1.000
                    b   1.00000  -150.0000   0.9439725674   0.9439725674 -168.9036347542  1.000
                    c   1.00000    90.0000   0.9439725674   0.9439725674   71.0963652458  1.000
   4   PQ4    LOAD  a   1.00000   -30.0000   0.9182934354   0.9182934354  -54.7966527085  1.000
                    b   1.00000  -150.0000   0.9182934354   0.9182934354 -174.7966527085  1.000
                    c   1.00000    90.0000   0.9182934354   0.9182934354   65.2033472915  1.000
```

Figure 4.5 Sections 2 and 3 of the report of results for the network of fig. 4.4

buses as *PQ* buses. Furthermore, the computations are initialized by setting all phase *a* voltage phasors to unity. The initial transformation ratio is also set to one (corresponding to the initial tap position $n_x = 5$).

Figure 4.5 shows part of the results of the computations. Convergence is reached after ten iterations (based on the threshold 10^{-10} for the maximum mismatch amplitude), where tap changes are performed at iterations 4 and 6. As explained in sec. 3.9, a precondition for a tap change is to have a sufficiently small norm of the mismatch vector (here 10^{-3} is imposed as the maximum acceptable value), which is why the first tap change is carried out only at the fourth iteration, and one iteration must elapse before the second one is performed. Referring to the

```
4. POWERS AND THEIR BALANCE AT EVERY NODE
Node    1a: S_br (L_12               )  =     1.5622604325 + i(    0.6611185413) pu
            S_G                        =     1.5622604325 + i(    0.6611185413) pu
            eps_S                      =       0.0000e+00 + i(      0.0000e+00) pu
Node    1b: S_br (L_12               )  =     1.5622604325 + i(    0.6611185413) pu
            S_G                        =     1.5622604325 + i(    0.6611185413) pu
            eps_S                      =       0.0000e+00 + i(      0.0000e+00) pu
Node    1c: S_br (L_12               )  =     1.5622604325 + i(    0.6611185413) pu
            S_G                        =     1.5622604325 + i(    0.6611185413) pu
            eps_S                      =       0.0000e+00 + i(      0.0000e+00) pu
Node    2a: S_br (L_12               )  =    -1.5334830786 + i(   -0.3733450029) pu
            S_br (TrYd1 TC           )  =     1.5334830786 + i(    0.3733450029) pu
            eps_S                      =       1.5543e-15 + i(      5.5511e-16) pu
Node    2b: S_br (L_12               )  =    -1.5334830786 + i(   -0.3733450029) pu
            S_br (TrYd1 TC           )  =     1.5334830786 + i(    0.3733450029) pu
            eps_S                      =      -1.3323e-15 + i(     -3.2196e-15) pu
Node    2c: S_br (L_12               )  =    -1.5334830786 + i(   -0.3733450029) pu
            S_br (TrYd1 TC           )  =     1.5334830786 + i(    0.3733450029) pu
            eps_S                      =      -1.3323e-15 + i(     -4.9960e-15) pu
Node    3a: S_br (L_34               )  =     0.9097241331 + i(    0.1972413306) pu
            S_br (TrYd1 TC           )  =    -1.5334830786 + i(   -0.1081329098) pu
            S_sh                       =     0.6237589456 + i(   -0.0891084208) pu
            eps_S                      =       5.5511e-16 + i(      4.8572e-16) pu
Node    3b: S_br (L_34               )  =     0.9097241331 + i(    0.1972413306) pu
            S_br (TrYd1 TC           )  =    -1.5334830786 + i(   -0.1081329098) pu
            S_sh                       =     0.6237589456 + i(   -0.0891084208) pu
            eps_S                      =       0.0000e+00 + i(     -1.0131e-15) pu
Node    3c: S_br (L_34               )  =     0.9097241331 + i(    0.1972413306) pu
            S_br (TrYd1 TC           )  =    -1.5334830786 + i(   -0.1081329098) pu
            S_sh                       =     0.6237589456 + i(   -0.0891084208) pu
            eps_S                      =       4.4409e-15 + i(      5.9119e-15) pu
Node    4a: S_br (L_34               )  =    -0.9000000000 + i(   -0.1000000000) pu
            S_L                        =     0.9000000000 + i(    0.1000000000) pu
            eps_S                      =      -1.2212e-15 + i(      1.4155e-15) pu
Node    4b: S_br (L_34               )  =    -0.9000000000 + i(   -0.1000000000) pu
            S_L                        =     0.9000000000 + i(    0.1000000000) pu
            eps_S                      =      -2.2204e-16 + i(      2.5813e-15) pu
Node    4c: S_br (L_34               )  =    -0.9000000000 + i(   -0.1000000000) pu
            S_L                        =     0.9000000000 + i(    0.1000000000) pu
            eps_S                      =      -2.1094e-15 + i(     -3.0809e-15) pu

_____________________________________ END OF REPORT _____________________________________
```

Figure 4.6 Section 4 of the report of results for the network of fig. 4.4

adjustment column, the numbers separated by a slash sign are n_{adj1} on the left hand side, and n_{adj2} on the right hand side.

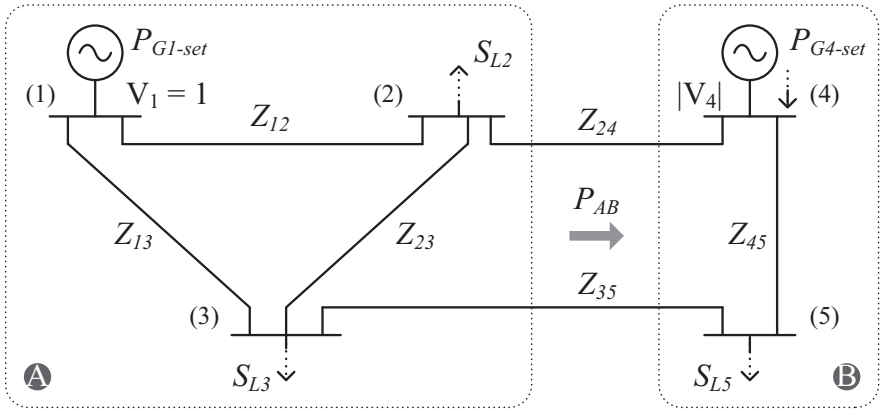
When the tap changing functionality is activated, additional information pertinent to the tap changing activity is provided (in section 2.2). In particular, the controlled bus number, its associated voltage limits, its voltage prior to making a change (indicated by the suffix '0'), and the tap positions before and after the change are shown. Remarks may also be added, e.g. in the present case $dn = -1$ indicates a one unit decrease in tap position. Such behaviour is expected because the controlled bus is on the secondary side of the transformer, and its voltage (initially 0.900532 pu) is below the minimum required value (0.93 pu). The third section of the report

shows, among other things, the final voltage at the controlled bus, i.e. 0.9440 pu, which lies within the required range.

Lastly, as seen in the latter part of the report (fig. 4.6), the balance in power is reached at every node, which is an additional indicator of the soundness of the results.

4.3 Regulation of power transferred and frequency on a five-bus network

4.3.1 Constant load



Basic conditions:

$Z_{12} = j0.4$ pu
$Z_{13} = j0.2$ pu
$Z_{23} = 0.05 + j0.25$ pu
$Z_{24} = 0.1 + j0.4$ pu
$Z_{35} = 0.05 + j0.3$ pu
$Z_{45} = j0.2$ pu

$S_{L2} = 0.2 + j0.15$ pu
$S_{L3} = 0.8 - j0.3$ pu
$S_{L5} = 0.6 + j0.2$ pu

Regulation conditions:

$P_{G1\text{-}set} = 1.1$ pu
$R_1 = 0.15$
$\beta_1 = 0.7$
$P_{G4\text{-}set} = 0.4$ pu, $|V_4| = 1$ pu
$R_4 = 0.3$
$\beta_4 = 0.3$

Figure 4.7 Two-zone five-bus network analysed using the classic model and the power and frequency regulation model

The system of fig. 4.7 is used to exemplify the *nr3r* computations in the regulation model framework. The network is composed of five three-phase buses. The loads are balanced. The

Table 4.1 Results of the five-bus network with constant load in the classic and regulation models (values are given in pu)

	Classic	**Power and frequency regulation ($P_{AB\,sch} = 0.2$ pu, $f_{sch} = 1$ pu)**		
		$a_p = 0, a_f = 5$	$a_p = 1, a_f = 0$	$a_p = 1, a_f = 5$
V_{2a}	$0.9858 \angle -7.8253°$	$0.9856 \angle -7.5423°$	$0.9858 \angle -7.8037°$	$0.9856 \angle -7.5587°$
V_{3a}	$1.0056 \angle -9.9435°$	$1.0057 \angle -9.7210°$	$1.0056 \angle -9.9265°$	$1.0056 \angle -9.7339°$
V_{4a}	$1.0000 \angle -8.2537°$	$1.0000 \angle -7.5786°$	$1.0000 \angle -8.2022°$	$1.0000 \angle -7.6178°$
V_{5a}	$0.9708 \angle -13.0866°$	$0.9711 \angle -12.5889°$	$0.9709 \angle -13.0486°$	$0.9711 \angle -12.6178°$
S_{24}	$0.0091 - j0.0373$	$-0.0069 - j0.0337$	$0.0079 - j0.0370$	$-0.0059 - j0.0339$
S_{35}	$0.1933 + j0.0891$	$0.1779 + j0.0902$	$0.1921 + j0.0892$	$0.1788 + j0.0901$
P_{G1}	1.2037	1.1724	1.2013	1.1743
P_{G4}	0.4000	0.4310	0.4024	0.4292
P_{AB}	0.2024	0.1710	**0.2000**	0.1729
P_r	-	0.3105	2.8985	0.4734
f	-	**1.0000**	1.0862	1.0054

network is also divided into two zones (A and B) linked by two tie lines, i.e. one connecting the buses #2 and #4, the other the buses #3 and #5. Bus #1 is treated as the reference bus, bus #4 as a generator bus, and the remaining ones as load buses.

Three variants of the regulation model are examined: frequency regulation alone ($a_p = 0$, $a_f \neq 0$), transferred power regulation alone ($a_p \neq 0$, $a_f = 0$), and the regulation of a linear combination of both transferred power and frequency ($a_p \neq 0$, $a_f \neq 0$). As a reference, the network is also solved based on the classic model. The main results are given in table 4.1, where the powers are given in three-phase pu. Furthermore, the report for the power regulation analysis is included in figures 4.9 and 4.10 at the end of the chapter, to show in detail the results provided by the program *nr3r* (recall that, except for the power transfer value, the *nr3r* report gives the powers in single-phase pu). As it should, since the system $-\varepsilon = J\Delta x$ is solved iteratively to ultimately obtain vanishing ε elements, the scheduled frequency and transferred power are fulfilled respectively in the first two cases. In the third case, though ε_{pf} vanishes, the individual gaps in frequency $f - f_{sch}$ and transferred power $P_T - P_{T\,sch}$ are non-zero. The extent of those gaps is determined by the relative weight of a_f with respect to a_p and vice

versa. By comparing the flat tie lie control approach ($a_p = 1$, $a_f = 0$) with the classic model approach, one also notices that the excess active transferred power (0.0024 pu) can be relieved by transferring an equal share of the generated active power from zone A to zone B.

The validity of the solutions is again supported by the facts that the imposed conditions as well as the balance in power at all buses are fulfilled. Furthermore, as an extra verification step, the three regulation cases were solved by the method of constrained optimisation (refer to sec. 2 of the appendix I) in the MATLAB environment, and the results were verified to coincide with the ones of *nr3r*.

4.3.2 Frequency and voltage dependent load

The network of fig. 4.7 is examined once again by replacing the load at bus #5 by a frequency and voltage dependent load characterized by $P_{L1} = 0.6$ pu, $Q_{L1} = 0.2$ pu, $P_{L\text{-}Z} = 0.3$ pu, $Q_{L\text{-}Z} = 0.1$ pu, and $K_P = K_Q = 7$ pu. The results are shown in table 4.2.

Table 4.2 Results of the five-bus network with a frequency and voltage dependent load in the regulation model (values are given in pu)

	Power and frequency regulation ($P_{ABsch} = 0.2$ pu, $f_{sch} = 1$ pu)		
	$a_p = 0, a_f = 5$	$a_p = 1, a_f = 0$	$a_p = 1, a_f = 5$
V_{2a}	$0.9826 \angle -8.9829°$	$0.9850 \angle -7.7364°$	$0.9839 \angle -8.3875°$
V_{3a}	$0.9979 \angle -11.3444°$	$1.0042 \angle -9.9775°$	$1.0010 \angle -10.6921°$
V_{4a}	$1.0000 \angle -10.1968°$	$1.0000 \angle -7.8685°$	$1.0000 \angle -9.0841°$
V_{5a}	$0.9498 \angle -16.8737°$	$0.9669 \angle -13.2948°$	$0.9582 \angle -15.1602°$
S_{24}	$0.0391 - j0.0519$	$-0.0033 - j0.0360$	$0.0188 - j0.0442$
S_{35}	$0.3245 + j0.1203$	$0.2033 + j0.0962$	$0.2671 + j0.1082$
S_{L5}	$0.8707 + j0.2902$	$0.6580 + j0.2193$	$0.7701 + j0.2567$
P_{G1}	1.3650	1.2014	1.2874
P_{G4}	0.5136	0.4607	0.4885
P_{AB}	0.3635	**0.2000**	0.2860
P_r	1.1357	−0.5965	0.3120
f	**1.0000**	0.9639	0.9828

As seen in the previous section, the scheduled frequency and transferred power are fulfilled in the first two cases respectively; in the combined regulation case, how close they are from their scheduled values depends on the relative weight of their coefficients in ε_{pf}. Due to the different voltage moduli and frequency gaps between the three regulation cases, the load at bus #5 takes differing values.

4.4 Power network composed of 3000 three-phase buses

The power system utilised in Lagacé (2012) to study the convergence properties of various power flow methods is used here to evaluate the computation efficiency of *nr3r*. As required, the system is large; specifically it is composed of 3000 three-phase buses (or 9000 nodes) connected in a meshed and radial way typical of a North American network. The loads are balanced. The computations are characterized as shown in table 4.3, and performed with an initial absolute mismatch of the order of 2.6×10^{-2}.

Table 4.3 Significant characteristics of the 3000-bus system

Y_{bus} dimensions	9000×9000
Y_{bus} elements (non-zero)	$35,514$
Jacobian matrix dimensions	$17,494 \times 17,494$
Jacobian matrix elements (non-zero)	$139,504$

A large part of the methodology described in ch. 3 was developed in a series of steps aimed at improving the computation speed of the program. Among other things, in its original state *nr3r* relied on two imbedded *for* loops to construct the Y_{bus} matrix, and on a series of *for* loops (each associated with a type of mismatch) to obtain the Jacobian matrix. Furthermore, a computationally expensive function to identify the indices of non-zero array elements, namely NumPy's *nonzero* function, was used throughout the program. As a result, the computing time (including the Y_{bus} construction) to solve the 3000-bus network was around 225s, a duration

definitely inappropriate in view of the project's constraints. In fact, in general terms the maximum computing time should be inferior to 3s.

Three main sets of measures were implemented to increase computation efficiency: the Y_{bus} procedure was completely vectorized and the Y_{bus} matrix created directly as a sparse matrix of the type *csr* (sec. 3.3); the Jacobian matrix procedure was also vectorized, owing to the identification at once of the non-zero Y_{bus} elements and the use of boolean filter arrays and mapping arrays, and constructed by a single call of the *csr matrix* function (sec. 3.6); in several cases the utilization of the *nonzero* function was replaced by a mapping array. Consequently, the computation time could be reduced by more than a six hundredfold to a time of the order of 0.35s. Added to the calculation time is also the time to read and organize the input data, measured to be of the order of 0.44s. Therefore, excluding the time needed to construct the report (which is arbitrary in the sense that it depends on the information sought), the relevant processing time is around 0.79s.[1]

For the sake of comparison, it is interesting to note that the calculation time recorded with *nr3r* is significantly shorter than the one obtained with the program used to conduct the analyses of Lagacé (2012), i.e. around 6.5s when employing Newton's method, even though such program operates in a unifilar framework with a smaller Y_{bus} matrix ($11,838$ non-zero elements) and a smaller Jacobian matrix ($46,568$ non-zero elements).

As far as the validity of the results is concerned, the imposed conditions as well as the balance in powers are met. Furthermore, the results were confirmed to agree with the ones yielded by the program used by Lagacé (2012). In fact, the voltage moduli and phases at all nodes were verified to agree within 10^{-8} pu. The distribution of the complex voltages obtained with the 9000 nodes is also shown in fig. 4.8. The moduli are found to lie between 0.9 pu and 1.1 pu. The spread in phases is relatively large, yet it is a consequence of the network topology and conditions.

---

[1] It should be noted, however, that such durations are dependent on the processor and memory of the computer. In the present case, a computer with a 2.5 GHz Intel Core i5 processor and 16 GB 1333 MHz memory was used.

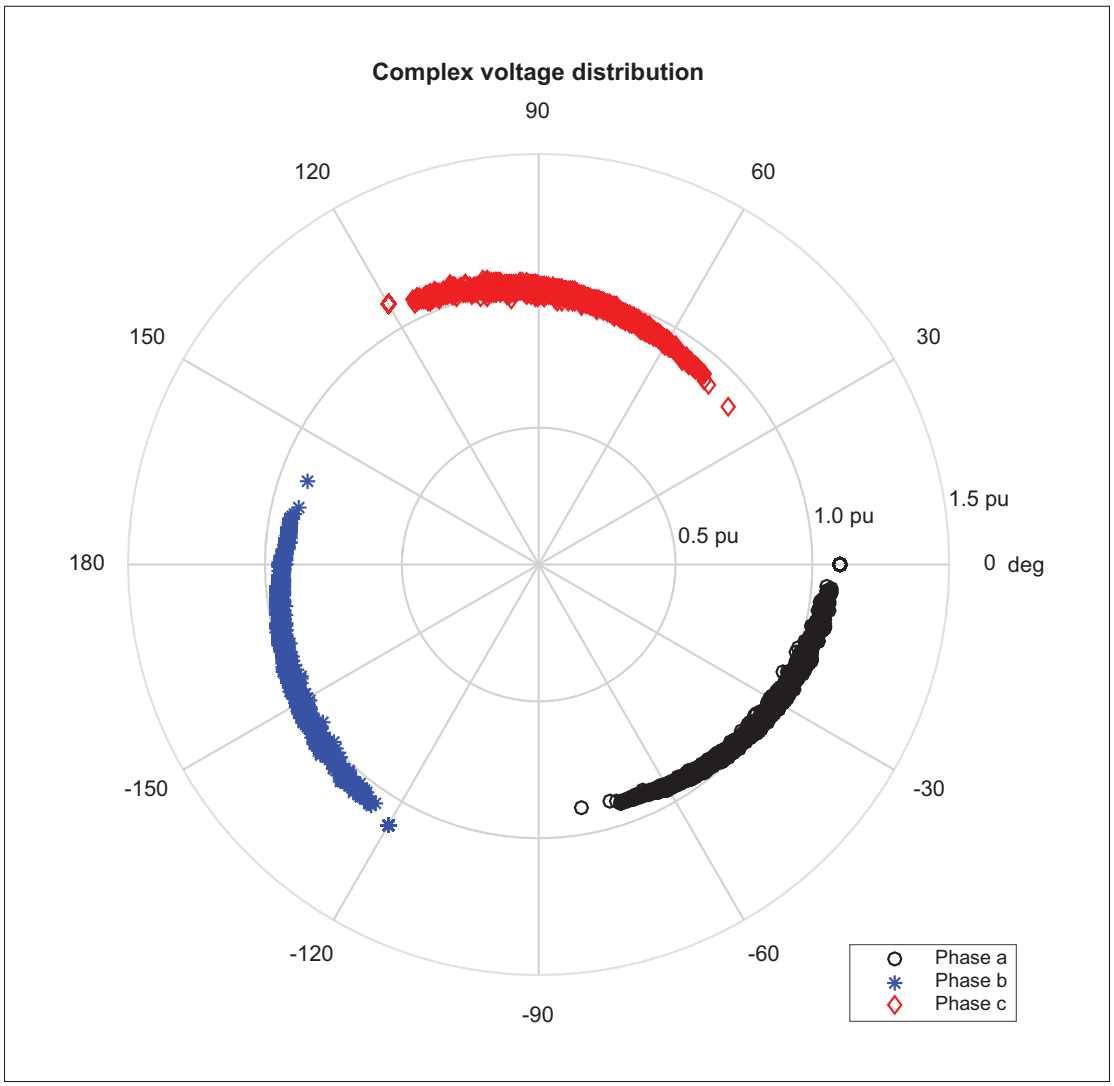


Figure 4.8 Distribution of complex voltages resulting from the resolution of the 3000 three-phase buses network

```
_____________________________ NR3R Ver. 1.1 (01 February 2016) _____________________________
Time: 15.06.2016 17:41
Path: /home/saubert2/Documents/Python/nr3r

1. INPUT SUMMARY
Input file: ../test_files/test_files_r/nr3r_rm_p_test3.dat
Number of nodes:                          15
Number of buses:                           5 (5 three-ph. and 0 single-ph.)
Number of branches:                        6 (6 three-ph. and 0 single-ph.)
Number of zigzag earth. transformers:      0

Transferred power coeff. (a_p):  1.0000000000 pu
Scheduled power transfer:        0.2000000000 pu
Frequency bias coeff. (a_f):     0.0000000000 pu
Scheduled frequency:             1.0000000000 pu
Primary tie branch bus(es):      2, 3
Secondary tie branch bus(es):    4, 5
Circuit label(s):                1, 1

2.CONVERGENCE BEHAVIOUR
Model: Transferred power regulation (flat tie-line control)
Calculation method: Newton-Raphson

It.     eps_Pmax      (Node)      eps_Qmax      (Node)       eps_pf           ||eps||       Adj.
  0 -3.3000000000e+00   (1a)  -3.0000000000e-01   (3c)  -6.0000000000e-01  4.033298898e+00  -/-
  1 -2.6820563063e-02   (4a)   8.8069424551e-02   (3a)   8.6246781647e-04  1.710507701e-01  -/-
  2 -2.4501912552e-04   (4a)   9.7802431839e-04   (3c)  -3.2685295084e-04  1.897180985e-03  -/-
  3 -7.2714894883e-08   (1a)   1.2812357736e-07   (3b)  -9.7683604317e-08  2.772728268e-07  -/-
  4  2.3314683517e-15   (5b)   2.9420910153e-15   (3a)  -4.4408920985e-15  8.974584149e-15  -/-

Calculation time until reaching tolerance threshold for eps_max: 0.0168s

3. POWER TRANSFER AND NETWORK FREQUENCY
Power transfer (PT):             0.2000000000 pu
PT - PT_sch:                    -0.0000000000 pu
Network frequency (f):           1.0862462145 pu
f - f_sch:                       0.0862462145 pu
Dist. power insufficiency (Pr):  0.9661761644 pu

4. VOLTAGES
Bus#   Name   Type Ph. |V0| (pu) th0 (deg)    |V| (pu)       |V| (kV)       th. (deg)    Eb (kV)
---- -------- ---- --- --------- --------- -------------- -------------- --------------- -------
   1   REF    REF   a   1.00000     0.0000   1.0000000000   1.0000000000    0.0000000000  1.000
                    b   1.00000  -120.0000   1.0000000000   1.0000000000 -120.0000000000  1.000
                    c   1.00000   120.0000   1.0000000000   1.0000000000  120.0000000000  1.000
   2   PQ2    LOAD  a   1.00000     0.0000   0.9857650311   0.9857650311   -7.8036927022  1.000
                    b   1.00000  -120.0000   0.9857650311   0.9857650311 -127.8036927022  1.000
                    c   1.00000   120.0000   0.9857650311   0.9857650311  112.1963072978  1.000
   3   PQ3    LOAD  a   1.00000     0.0000   1.0055780126   1.0055780126   -9.9265077711  1.000
                    b   1.00000  -120.0000   1.0055780126   1.0055780126 -129.9265077711  1.000
                    c   1.00000   120.0000   1.0055780126   1.0055780126  110.0734922289  1.000
   4   PV4    GEN   a   1.00000     0.0000   1.0000000000   1.0000000000   -8.2022469249  1.000
                    b   1.00000  -120.0000   1.0000000000   1.0000000000 -128.2022469249  1.000
                    c   1.00000   120.0000   1.0000000000   1.0000000000  111.7977530751  1.000
   5   PQ5    LOAD  a   1.00000     0.0000   0.9708568612   0.9708568612  -13.0486325421  1.000
                    b   1.00000  -120.0000   0.9708568612   0.9708568612 -133.0486325421  1.000
                    c   1.00000   120.0000   0.9708568612   0.9708568612  106.9513674579  1.000
```

Figure 4.9 Report of results (sec. 1 to 4) of the transferred power regulation model applied to the system of fig. 4.7

```
5. POWERS AND THEIR BALANCE AT EVERY NODE
Node    1a: S_br (Line 1-2           )  =     0.3346165244 + i(    0.0584101214) pu
            S_br (Line 1-3           )  =     0.8667320271 + i(    0.0473790700) pu
            S_G                         =     1.2013485515 + i(    0.1057891914) pu
            eps_S                       =      -4.4409e-16 + i(      8.8818e-16) pu
Node    1b: S_br (Line 1-2           )  =     0.3346165244 + i(    0.0584101214) pu
            S_br (Line 1-3           )  =     0.8667320271 + i(    0.0473790700) pu
            S_G                         =     1.2013485515 + i(    0.1057891914) pu
            eps_S                       =      -6.6613e-16 + i(      3.3307e-16) pu
Node    1c: S_br (Line 1-2           )  =     0.3346165244 + i(    0.0584101214) pu
            S_br (Line 1-3           )  =     0.8667320271 + i(    0.0473790700) pu
            S_G                         =     1.2013485515 + i(    0.1057891914) pu
            eps_S                       =      -2.2204e-16 + i(      3.7470e-16) pu
Node    2a: S_br (Line 1-2           )  =    -0.3346165244 + i(   -0.0122581371) pu
            S_br (Line 2-4           )  =     0.0078938521 + i(   -0.0369946767) pu
            S_br (Line 2-3           )  =     0.1267226723 + i(   -0.1007471861) pu
            S_L                         =     0.2000000000 + i(    0.1500000000) pu
            eps_S                       =       2.2204e-16 + i(     -4.1633e-16) pu
Node    2b: S_br (Line 1-2           )  =    -0.3346165244 + i(   -0.0122581371) pu
            S_br (Line 2-4           )  =     0.0078938521 + i(   -0.0369946767) pu
            S_br (Line 2-3           )  =     0.1267226723 + i(   -0.1007471861) pu
            S_L                         =     0.2000000000 + i(    0.1500000000) pu
            eps_S                       =      -1.9429e-16 + i(      1.1380e-15) pu
Node    2c: S_br (Line 1-2           )  =    -0.3346165244 + i(   -0.0122581371) pu
            S_br (Line 2-4           )  =     0.0078938521 + i(   -0.0369946767) pu
            S_br (Line 2-3           )  =     0.1267226723 + i(   -0.1007471861) pu
            S_L                         =     0.2000000000 + i(    0.1500000000) pu
            eps_S                       =       1.3323e-15 + i(      4.1633e-16) pu
Node    3a: S_br (Line 3-5           )  =     0.1921061479 + i(    0.0891952898) pu
            S_br (Line 1-3           )  =    -0.8667320271 + i(    0.1033147666) pu
            S_br (Line 2-3           )  =    -0.1253741209 + i(    0.1074899435) pu
            S_L                         =     0.8000000000 + i(   -0.3000000000) pu
            eps_S                       =      -5.5511e-16 + i(      2.0539e-15) pu
Node    3b: S_br (Line 2-3           )  =    -0.1253741209 + i(    0.1074899435) pu
            S_br (Line 3-5           )  =     0.1921061479 + i(    0.0891952898) pu
            S_br (Line 1-3           )  =    -0.8667320271 + i(    0.1033147666) pu
            S_L                         =     0.8000000000 + i(   -0.3000000000) pu
            eps_S                       =      -1.1102e-15 + i(      7.2164e-16) pu
Node    3c: S_br (Line 3-5           )  =     0.1921061479 + i(    0.0891952898) pu
            S_br (Line 2-3           )  =    -0.1253741209 + i(    0.1074899435) pu
            S_br (Line 1-3           )  =    -0.8667320271 + i(    0.1033147666) pu
            S_L                         =     0.8000000000 + i(   -0.3000000000) pu
            eps_S                       =      -2.3315e-15 + i(      1.2212e-15) pu
Node    4a: S_br (Line 2-4           )  =    -0.0077465977 + i(    0.0375836943) pu
            S_br (Line 4-5           )  =     0.4101120652 + i(    0.1630707864) pu
            S_G                         =     0.4023654675 + i(    0.2006544807) pu
            eps_P (sum abc) + i eps_Q   =      -1.9984e-15 + i(     -1.3045e-15) pu
Node    4b: S_br (Line 4-5           )  =     0.4101120652 + i(    0.1630707864) pu
            S_br (Line 2-4           )  =    -0.0077465977 + i(    0.0375836943) pu
            S_G                         =     0.4023654675 + i(    0.2006544807) pu
            eps_P (sum abc) + i eps_Q   =      -1.9984e-15 + i(      3.8858e-16) pu
Node    4c: S_br (Line 2-4           )  =    -0.0077465977 + i(    0.0375836943) pu
            S_br (Line 4-5           )  =     0.4101120652 + i(    0.1630707864) pu
            S_G                         =     0.4023654675 + i(    0.2006544807) pu
            eps_P (sum abc) + i eps_Q   =      -1.9984e-15 + i(      4.1633e-16) pu
Node    5a: S_br (Line 3-5           )  =    -0.1898879348 + i(   -0.0758860111) pu
            S_br (Line 4-5           )  =    -0.4101120652 + i(   -0.1241139889) pu
            S_L                         =     0.6000000000 + i(    0.2000000000) pu
            eps_S                       =       1.6653e-15 + i(      1.9429e-15) pu
Node    5b: S_br (Line 4-5           )  =    -0.4101120652 + i(   -0.1241139889) pu
            S_br (Line 3-5           )  =    -0.1898879348 + i(   -0.0758860111) pu
            S_L                         =     0.6000000000 + i(    0.2000000000) pu
            eps_S                       =       1.9984e-15 + i(      1.9429e-15) pu
Node    5c: S_br (Line 3-5           )  =    -0.1898879348 + i(   -0.0758860111) pu
            S_br (Line 4-5           )  =    -0.4101120652 + i(   -0.1241139889) pu
            S_L                         =     0.6000000000 + i(    0.2000000000) pu
            eps_S                       =       2.2204e-15 + i(      2.4980e-16) pu

_____________________________________ END OF REPORT _____________________________________
```

Figure 4.10 Report of results (sec. 5) of the transferred power regulation model applied to the system of fig. 4.7

CONCLUSION

The work has focused on formulating and implementing a power flow model that is adapted to the resolution of three-phase (unbalanced) power systems by taking into account the effects of network frequency variations as well as economic power transactions between areas of a network. In addition, single-phase components have been implemented in order to facilitate the treatment of unbalanced loads, and to allow for the solution of systems that comprise both transmission and distribution elements. For comparative purposes, a more basic model that assumes a constant network frequency was similarly formulated and implemented.

The implementation of the power flow models in the program *nr3r* resulted from a series of optimization steps aimed at reaching high computation speeds to satisfy the needs of Hydro-Québec. Several efficient array based procedures were developed. Among other things, such procedures eliminated the need of *for* loops in the construction of the bus admittance matrix from the component level admittance matrices, as well as in the construction of the Jacobian matrix. A compact yet comprehensible algorithm was also devised to reproduce the operations of voltage regulation tap changers. A number of numerical examples were presented to illustrate the main functionalities of the program. In particular, the program's computation speed was tested with a large network composed of 3000 three-phase buses (9000 nodes). In each case, as shown by the reports generated by *nr3r*, the imposed conditions were fulfilled, and the balance in power was met at all buses (up to a specified tolerance).

The program presents certain limitations that could be addressed in future work. As hinted previously, power constraints at generator buses could be taken into account to respect the physical limits of generators. This would require the conversion of bus types during the iterative process. Moreover, the treatment of the bus admittance matrix may be modified to take into consideration neutral currents and voltages, particularly in the power distribution portion of a network. Besides, the admittance matrix for the untransposed transmission line may be

derived and included in the program to enhance its versatility. Computation speed may also be increased by freeing the program code from complex numbers. This could also facilitate the code migration to a lower level language if need be. Last but not least, the option to use the Levenberg-Marquardt method should be added to the present power flow analysis tool in order to widen its region of convergence.

As regards future research avenues, analyses of systems under the effect of phenomena, such as line tripping, load shedding (e.g. due to a major loss of power generation or tie-line support), and generation shedding (e.g. after a major loss of load or tie-line export), may be carried out. Furthermore, the program should be utilized to investigate the behaviour of networks with respect to load profiles that incorporate customer energy consumption (and/or production) over time. Specifically large and ill-conditioned systems, representative of combined transmission and distribution networks, should also be solved and analysed in detail to find new ways to improve the convergence rate and the region of convergence of the program. In this respect, a potential approach is to distinguish branches with comparable impedance values, and to break up the iterative process accordingly.

APPENDIX I

MATHEMATICAL TOOLS

1. Newton's method

If $g(x)$ is a vector function that is differentiable with respect to the elements x_i, then $g(x+\Delta x)$ may be approximated by keeping the first two terms of the Taylor series of g about x:

$$g(x+\Delta x) \approx g(x) + \nabla g(x)\ \Delta x$$

Assuming that $g(x)$ needs to be driven to zero, corrections Δx can be sought such that

$$g(x+\Delta x) = g(x) + \nabla g(x)\ \Delta x = 0$$

or equivalently

$$\Delta x = -[\nabla g(x)]^{-1} g(x). \tag{A I-1}$$

An algorithm based on Newton's method consists in setting once the initial values $x = x_0$; computing iteratively $g(x)$, $\nabla g(x)$, and Δx by (A I-1); and each time using Δx to correct the unknowns, i.e. $x = x + \Delta x$. At the end of the iterative process, convergence is confirmed by evaluating $g(x)$. The elements of x_0 should be chosen carefully (relatively close to the solution) in order to prevent divergence of the results.

2. Constrained optimisation

Let $f(x)$ be a function to be minimised or maximised under the constraints $p_j(x) = 0$ ($j = 1,2,\ldots,n$), where x is a vector of the independent variables x_i ($i = 1,2,\ldots,m$). Thus a *Lagrangian* function can be defined as follows:

$$\mathscr{L}(x,\lambda) = f(x) + \sum_j \lambda_j\, p_j(x) \tag{A I-2}$$

where the variables λ_j are referred to as the Lagrange multipliers. Naturally, there may be only one constraint, in which case the sum in (A I-2) reduces to a single term.

The optimum of $f(x)$ under the constraints $p_j(x)$ can be realised when $f(x)$ is tangent to the functions $p_j(x)$ (in the multivariate space of x). In that case the gradients of the objective function $f(x)$ and constraints $p_j(x)$ should be non-zero and have the same orientation. Mathematically, there should be a λ such that the following equations hold:

$$\nabla_x f(x) + \sum_j \lambda_j \nabla_x\, p_j(x) = 0$$

Furthermore, we note that differentiating the Lagrangian function with respect to any Lagrange multiplier yields the corresponding constraint, namely

$$\frac{\partial \mathscr{L}}{\partial \lambda_j} = p_j(x)$$

Hence the constrained optimisation problem may be summarised as finding the solution to the following system of equations:

$$\frac{\partial \mathscr{L}}{\partial x_i} = 0 \qquad \text{(A I-3)}$$

$$\frac{\partial \mathscr{L}}{\partial \lambda_j} = 0 \qquad \text{(A I-4)}$$

APPENDIX II

PROGRAMMING LANGUAGE

The specifications of the programming language used as well as key programming tools are given below.

Programming language:	Python, version 3.5.2
Scientific computing package:	NumPy, version 1.11.1
Sparse matrix package:	*scipy.sparse*, from the SciPy library, version 0.17.1
Key classes:	*scipy.sparse.csr_matrix*, to create the bus admittance matrix and the Jacobian matrix
	scipy.sparse.spsolve, to solve the linear system resulting from the application of Newton's method

APPENDIX III

PROGRAM INSTRUCTION MANUAL

1. Input file

Given a power network to be analysed, the input file is a text file that contains the necessary data to describe the network and its state. The extension .dat is typically used.

2. Types of data

In the following sections, the typewriter typeface indicates the different data items (and their syntax) that can be inserted in the input file. In most cases, these items take the form of a list (in the computing sense), or of a list of lists. The order in which they are entered is irrelevant.

The bus data particularly take the latter form, where each inner list corresponds to a distinct bus in the network (labelled by the index *i*). There are no constraints as to the order in which the inner lists are entered. The ordered elements of each inner list represent different parameters applicable to the bus. Their descriptions are provided in the following sections, based on position number. If a parameter designation depends on the calculation model, then the one that is specific to the classic model is placed in parentheses. The branch data and the zigzag earthing transformer data are organised similarly.

The regulation of power transferred and/or of network frequency is achieved by means of the expression $\varepsilon_{pf} = a_p(P_T - P_{T\text{-}sch}) + a_f(f - f_{sch})$, which is brought to zero by Newton's method. The parameters a_p and a_f are specified in the input item *regfPT* described in sec. 2.4 of this appendix. Among other things, the values $a_p = 1$ and $a_f = 0$ correspond to the case where power transferred is strongly regulated, commonly known as *flat tie line control*.

Insertion of the item *regfPT* in the input file calls the power and frequency regulation model. Otherwise, the calculations are performed by default in the classic model.

2.1 Bus data

From sec. 2.5, recall the general expressions for the generation of power, i.e.

$$P_{Gi} = P_{G\text{-}set\,i} - \Delta f / R_i + \beta_i P_r$$
$$Q_{Gi} = Q_{G\text{-}set\,i}$$

and the load, i.e.

$$P_{Li} = P_{L0i} + (1 + K_{Pi}\Delta f)(P_{L1\,i} + P_{L\text{-}I\,i}\,|V_i| + P_{L\text{-}Z\,i}\,|V_i|^2 + P_{L\text{-}NP\,i}\,|V_i|^{NP_i})$$
$$Q_{Li} = Q_{L0i} + (1 + K_{Qi}\Delta f)(Q_{L1\,i} + Q_{L\text{-}I\,i}\,|V_i| + Q_{L\text{-}Z\,i}\,|V_i|^2 + Q_{L\text{-}NQ\,i}\,|V_i|^{NQ_i})$$

where i is a node index. Note that those expressions can also accommodate the classic model, through the omission of a few terms.

The shunt conductance $G_{sh\,i}$ and susceptance $B_{sh\,i}$ enter the construction of the bus admittance matrix.

2.1.1 Three-phase buses

```
B3 = [
[x0, x1, x2, ... , x24, x25],
[y0, y1, y2, ... , y24, y25]
]
```

0 : Bus number (a nonnegative integer)

1 : Bus name (a *string*)

2 : Bus type: '1' for reference (or swing), '2' for generator (or PV), and '3' for load (or PQ)

3 : Voltage base (kV)

4 : Initial voltage modulus (pu_{kV})

5 : Initial voltage phase (degrees)

6 : Shunt conductance $G_{sh\,i}$ (pu_{S})

7 : Shunt susceptance $B_{sh\,i}$ (pu_{S})

8 : Set active power generation $P_{G\text{-}set\,i}$ (or $P_{G\,i}$) (pu_{MW})

9 : Set reactive power generation $Q_{G\text{-}set\,i}$ (or $Q_{G\,i}$) (pu_{MVar})

10 : Constant active load $P_{L0\,i}$ (pu_{MW})

11 : Constant reactive load $Q_{L0\,i}$ (pu_{MVar})

12 : Active load with a constant power characteristic $P_{L1\,i}$ (pu_{MW})

13 : Reactive load with a constant power characteristic $Q_{L1\,i}$ (pu_{MVar})

14 : Coefficient $P_{L\text{-}I\,i}$ of the active load with a constant current characteristic ($\text{pu}_{\text{MW/kV}}$)

15 : Coefficient $Q_{L\text{-}I\,i}$ of the reactive load with a constant current characteristic ($\text{pu}_{\text{MVar/kV}}$)

16 : Coefficient $P_{L\text{-}Z\,i}$ of the active load with a const. impedance characteristic ($\text{pu}_{\text{MW}/\text{kV}^2}$)

17 : Coefficient $Q_{L\text{-}Z\,i}$ of the reactive load with a const. imped. characteristic ($\text{pu}_{\text{MVar}/\text{kV}^2}$)

18 : Coefficient $P_{L\text{-}NP\,i}$ of the active load with a $|V_i|^{NP_i}$ characteristic ($\text{pu}_{\text{MW}/\text{kV}^{NP_i}}$)

19 : Coefficient $Q_{L\text{-}NQ\,i}$ of the reactive load with a $|V_i|^{NQ_i}$ characteristic ($\text{pu}_{\text{MVar}/\text{kV}^{NQ_i}}$)

20 : Exponent NP_i of the active load exhibiting a $|V_i|^{NP_i}$ characteristic

21 : Exponent NQ_i of the reactive load exhibiting a $|V_i|^{NQ_i}$ characteristic

22 : Factor K_{Pi} of the frequency dependent load model ($\mathrm{pu}_{1/\mathrm{Hz}}$)

23 : Factor K_{Qi} of the frequency dependent load model ($\mathrm{pu}_{1/\mathrm{Hz}}$)

24 : Constant R_i reflecting the speed droop characteristics of the generator ($\mathrm{pu}_{\mathrm{Hz}}/\mathrm{pu}_{\mathrm{MW}}$)

25 : Participation factor β_i of the generator

Remarks:

a. Data 1–21 must be provided. Data 22–25 are only required to perform calculations in the power and frequency regulation model.

b. The fields 22–25 are allowed but ignored in the classic model, which permits to use the same data structure independently of the calculation model.

c. An unbalanced three-phase load can be modelled by specifying three single-phase buses with different load parameters.

d. In the classic model, any P_{Gi} or Q_{Gi} value assigned to a swing bus is ignored. In the power and frequency regulation model, any value Q_{Gi} assigned to a reference bus is ignored.

e. Any value Q_{Gi} assigned to a *PV* or generator bus is ignored.

2.1.2 Single-phase buses

```
B1 = [
[x0, x1, x2, ... , x24, x25],
[y0, y1, y2, ... , y24, y25]
]
```

Remarks:

a. The data structure is the same as the one of the three-phase buses, except that the element in position '0' is a single-phase bus number, i.e. a node number. A node number is written as a complex number in cartesian form composed of nonnegative integers as the real and imaginary parts. The imaginary part indicates the phase, where +0j, -1j, +1j correspond to phases *a*, *b*, *c* respectively. For example, the node number associated with phase-*c* of bus 120 is written as 120+1j.

b. Power is expressed in single-phase pu.

2.2 Branch data

The branches are always connected between two buses. In particular, a single-phase branch is bounded by two single-phase buses, or equivalently two nodes. By definition, the bus (or node) specified in the data position '0' is referred to as the primary bus (node), and the bus (node) specified in position '1' is referred to as the secondary bus (node).

2.2.1 Lines

2.2.1.1 Three-phase line – short line model

The short line model of the three-phase line is introduced at the beginning of sec. 2.3.4.2. It implies representing the three-phase line as three uncoupled π circuits. The line impedance ($Z = R + jX$) and shunt admittance ($Y = G + jB$) provided are directly used to build the admittance matrix (2.9).

```
L3 = [
[x0, x1, x2, x3, x4, x5, x6, x7],
[y0, y1, y2, y3, y4, y5, y6, y7]
]
```

0 : Primary bus number I (a nonnegative integer)

1 : Secondary bus number J (a nonnegative integer)

2 : Parameter *ckt* used to differentiate branches connected between the same buses (a nonnegative integer).

3 : Name of the line (a *string*)

4 : Resistance R (pu_Ω)

5 : Reactance X (pu_Ω)

6 : Shunt conductance G (pu_S)

7 : Shunt susceptance B (pu_S)

2.2.1.2 Three-phase line – transposed long line model

The long line model is presented in the latter part of sec. 2.3.4.2. The admittance matrix is given by (2.14), where the zero and direct sequence impedances per unit length are expressed in terms of their self and mutual counterparts as $z_0 = z_s + 2z_m$ and $z_1 = z_s - z_m$. Based on the same reasoning, $y_0 = y_s + 2y_m$ and $y_1 = y_s - y_m$. In the input data, the self and mutual impedances and admittances per unit length are broken down into their real and imaginary components, i.e. $z_s = r_s + jx_s$, $z_m = r_m + jx_m$, $y_s = g_s + jb_s$, and $y_m = g_m + jb_m$.

```
L3l = [
[x0, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12],
[y0, y1, y2, y3, y4, y5, y6, y7, y8, y9, y10, y11, y12]
]
```

0 : Primary bus number I (a nonnegative integer)

1 : Secondary bus number J (a nonnegative integer)

2 : Parameter *ckt* used to differentiate branches connected between the same buses (a non-negative integer).

3 : Name of the line (a *string*)

4 : Self resistance per unit length r_s (pu_Ω/km)

5 : Self reactance per unit length x_s (pu_Ω/km)

6 : Mutual resistance per unit length r_m (pu_Ω/km)

7 : Mutual reactance per unit length x_m (pu_Ω/km)

8 : Self conductance per unit length g_s (pu_S/km)

9 : Self susceptance per unit length b_s (pu_S/km)

10 : Mutual conductance per unit length g_m (pu_S/km)

11 : Mutual susceptance per unit length b_m (pu_S/km)

12 : Length of the line ℓ (km)

2.2.1.3 Single-phase line – short line model

The short single-phase line model is introduced at the beginning of sec. 2.3.4.1. It implies representing the single-phase line as a π circuit. The line impedance ($Z = R + jX$) and shunt admittance ($Y = G + jB$) provided are directly used to build the admittance matrix (2.6).

```
L1 = [
[x0, x1, x2, x3, x4, x5, x6, x7],
[y0, y1, y2, y3, y4, y5, y6, y7]
]
```

The data structure is similar to the one of the short three-phase line model, except that positions '0' and '1' refer to node numbers. Information on how node numbers should be specified as complex numbers can be found in sec. 2.1.2 of this appendix.

2.2.1.4 Single-phase line – long line model

The long line model is described in sec. 2.3.4.1. It consists essentially in substituting $Y/2$ and Z in (2.6) by (2.7) and (2.8), where the impedance and admittance per unit length can be decomposed into their real and imaginary parts, i.e. $z = r + jx$ and $y = g + jb$.

```
L1l = [
[x0, x1, x2, x3, x4, x5, x6, x7, x8],
[y0, y1, y2, y3, y4, y5, y6, y7, y8]
]
```

0 : Primary node number I (a complex number)

1 : Secondary node number J (a complex number)

2 : Parameter *ckt* used to differentiate branches connected between the same buses (a nonnegative integer)

3 : Name of the line (a *string*)

4 : Resistance per unit length r (pu_Ω/km)

5 : Reactance per unit length x (pu_Ω/km)

6 : Conductance per unit length g (pu_S/km)

7 : Susceptance per unit length b (pu_S/km)

8 : Length of the line ℓ (km)

Remark: Refer to sec. 2.1.2 of this appendix regarding how node numbers should be specified as complex numbers.

2.2.2 Transformers

For compatibility purposes a single data structure is used for all three-phase transformers, despite their differences in construction and functionality. Wherever a parameter is irrelevant, it is ignored by the program.

2.2.2.1 *Yy0* transformer (star-star connection)

Refer to sec. 2.3.5.3 for details regarding the *Yy0* transformer model, and to sec. 2.3.6 for the generalisation reflecting the tap changer mechanism.

```
TrYy0 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

0 : Primary bus number I (a nonnegative integer)

1 : Secondary bus number J (a nonnegative integer)

2 : Parameter *ckt* used to differentiate branches connected between the same buses (a nonnegative integer).

3 : Name of the transformer (a *string*)

4 : Primary winding impedance Z_1 (pu_Ω)

5 : Secondary winding impedance Z_2 (pu_Ω)

6 : Magnetizing admittance Y_m (pu_S)

7 : Primary side grounding impedance Z_N (pu_Ω)

8 : Secondary side grounding impedance Z_n (pu_Ω)

9 : Minimum primary transformation ratio $a_{1\,min}$

10 : Primary transformation ratio a_1 (initial value)

11 : Maximum primary transformation ratio $a_{1\,max}$

12 : Secondary transformation ratio a_2

13 : Number of taps n_t (a nonnegative integer)

14 : Controlled bus number (a nonnegative integer)

15 : Minimum allowed voltage modulus at the controlled bus V_{min} (pu_{kV})

16 : Maximum allowed voltage modulus at the controlled bus V_{max} (pu_{kV})

Remarks:

a. Assigning a value of '0' to the controlled bus number field specifies that the transformer does not have a tap changer.

b. In the current version of the program, only the primary or the secondary bus are accepted as valid controlled buses. The regulation of voltage at distant buses may be implemented in a future version of the program.

c. The number of taps must be a nonnegative integer. If a positive decimal number is entered, then only its integer part is kept.

2.2.2.2 *Yd1* transformer (star-delta connection)

Refer to sec. 2.3.5.4 for details regarding the *Yd1* transformer model, and to sec. 2.3.6 for the generalisation reflecting the tap changer mechanism.

```
TrYd1 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

Remarks:

a. The data structure is the same as for the *Yy0* transformer.

b. Z_n is inapplicable, therefore its value (in position '8') is ignored by the program.

2.2.2.3 *Dd0* transformer (delta-delta connection)

Refer to sec. 2.3.5.5 for details regarding the *Dd0* transformer model, and to sec. 2.3.6 for the effect of introducing the transformation ratios a_1 and a_2.

```
TrDd0 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

Remarks:

a. The data structure is the same as for the *Yy0* transformer.

b. Z_N and Z_n are inapplicable, therefore their values (in positions '7' and '8') are ignored by the program.

c. The parameters concerned with the voltage regulation (in positions '9', '11', '13–16') are ignored, because tap changing is not implemented with this transformer type.

2.2.2.4 *Yz11* transformer

Refer to sec. 2.3.5.6 for details regarding the *Yz11* transformer model, and to sec. 2.3.6 for the effect of introducing the transformation ratios a_1 and a_2.

```
TrYz11 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

Remarks:

a. The data structure is the same as for the *Yy0* transformer.

b. The parameters concerned with the voltage regulation (in positions '9', '11', '13–16') are ignored, because tap changing is not implemented with this transformer type.

2.2.2.5 *Dz0* transformer

Refer to sec. 2.3.5.7 for details regarding the *Dz0* transformer model, and to sec. 2.3.6 for the effect of introducing the transformation ratios a_1 and a_2.

```
TrDz0 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

Remarks:

a. The data structure is the same as for the *Yy0* transformer.

b. Z_N is inapplicable, therefore its value (in position '7') is ignored by the program.

c. The parameters concerned with the voltage regulation (in positions '9', '11', '13–16') are ignored, because tap changing is not implemented with this transformer type.

2.2.2.6 *Dz10* transformer

Refer to sec. 2.3.5.8 for details regarding the *Dz10* transformer model, and to sec. 2.3.6 for the effect of introducing the transformation ratios a_1 and a_2.

```
TrDz10 = [
[x0, x1, x2, ... , x15, x16],
[y0, y1, y2, ... , y15, y16]
]
```

Remarks: The remarks made for the *Dz0* transformer are also applicable to the *Dz10* transformer.

2.2.2.7 The single-phase transformer

Refer to sec. 2.3.5.1 for details regarding the single-phase transformer model, and to sec. 2.3.6.1 for the generalisation reflecting the tap changer mechanism.

```
Tr1 = [
[x0, x1, x2, ... , x13, x14],
[y0, y1, y2, ... , y13, y14]
]
```

0 : Primary node number I (a complex number)

1 : Secondary node number J (a complex number)

2 : Parameter *ckt* used to differentiate branches connected between the same buses (a nonnegative integer).

3 : Name of the transformer (a *string*)

4 : Primary winding impedance Z_1 (pu_Ω)

5 : Secondary winding impedance Z_2 (pu_Ω)

6 : Magnetizing admittance Y_m (pu_S)

7 : Minimum primary transformation ratio $a_{1\,min}$

8 : Primary transformation ratio a_1 (initial value)

9 : Maximum primary transformation ratio $a_{1\,max}$

10 : Secondary transformation ratio a_2

11 : Number of taps n_t (a nonnegative integer)

12 : Controlled node number (a complex number)

13 : Minimum allowed voltage modulus at the controlled node V_{min} (pu_kV)

14 : Maximum allowed voltage modulus at the controlled node V_{max} (pu_kV)

Remarks:

a. Assigning a value of '0' to the controlled node number field specifies that the transformer does not have a tap changer.

b. In the current version of the program, only the primary or secondary node are accepted as valid controlled nodes. The regulation of voltage at distant nodes may be implemented in a future version of the program.

c. Refer to sec. 2.1.2 of this appendix regarding how node numbers should be specified as complex numbers.

d. The number of taps must be a nonnegative integer. If a positive decimal number is entered, then only its integer part is kept.

2.3 The zigzag earthing transformer (shunt component)

Refer to sec. 2.3.7 for details regarding the zigzag earthing transformer model.

```
TrZg = [
[x0, x1, x2, x3, x4, x5],
[y0, y1, y2, y3, y4, y5]
]
```

0 : Bus number (a nonnegative integer)

1 : Parameter *ckt* used to differentiate multiple zigzag earthing transformers connected between the specified bus and the ground (a nonnegative integer)

2 : Name of the transformer (a *string*)

3 : Winding impedance Z (pu_{Ω})

4 : Magnetizing admittance Y_m (pu_{S})

5 : Grounding impedance Z_n (pu_{Ω})

2.4 Regulation of transferred power and/or frequency

The following data item provides the necessary parameters to evaluate the regulation equation

$$\varepsilon_{pf} = a_p(P_T - P_{T\,sch}) + a_f(f - f_{sch})$$

presented in sec. 2.5.4.

The power and frequency regulation model is selected by specifying the data item *regfPT*. Otherwise, the classic model is assumed.

```
regfPT = [af, fsch, ap, PTsch, i0, j0, ckt0, w0, ... , in, jn, cktn, wn]
```

af : Coefficient of the gap in frequency ($\text{pu}_{\text{MW3}\phi}/\text{pu}_{\text{Hz}}$)

fsch : Scheduled frequency (pu_{Hz})

ap : Coefficient of the gap in power transferred. Specifying the null value ($a_p = 0$) implies a strong regulation in frequency so as to meet the scheduled frequency constraint. In that case, the following parameters do not need to be specified (they are ignored by the program).

PTsch : Scheduled transfer of power between two areas of the electric network ($\text{pu}_{\text{MW3}\phi}$)

i : Primary three-phase bus number of a tie branch (a nonnegative integer)

j : Secondary three-phase bus number of a tie branch (a nonnegative integer)

ckt : Parameter used to differentiate tie branches connected between the same buses *i* and *j* (e.g. given two branches that connect between *i0* and *j0*, the first one may be assigned $ckt = 1$ and the second one $ckt = 2$) (a nonnegative integer).

w : Index that reflects the reference direction of power transfer; a value of $+1$ indicates that positive power flows from bus *i* to *j*, and vice versa for a value of -1.

Remarks:

a. Only three-phase power transfer is accepted. Therefore, *i* and *j* label three-phase buses. Power related units are assumed to be three-phase (denoted by 3ϕ).

b. An arbitrary number of tie branches can be specified. In the sample data list above, *n* tie branches are implied, with *n* corresponding sets of *i*, *j*, *ckt*, *w* parameters.

c. In the power calculations, power is taken to enter a tie branch at its bus *i*.

d. If several tie branches are specified, *w* may be used to vary artificially the importance of its associated branch with respect to the total power transferred. In other words, the total power transferred is the sum of the products of *w* and the power transfer per tie branch.

2.5 Initial network frequency

```
freq = [fi, fbase]
```

fi : Initial network frequency (Hz) (60Hz by default)

fbase : Base frequency (Hz) (60Hz by default)

This data item is optional.

2.6 Initial value of the power insufficiency distributed amongst the generators

Refer to the generator model presented in sec. 2.5.2.

```
Pr = x
```

x : Initial value of the total active power insufficiency distributed amongst the generators ($\text{pu}_{\text{MW}1\phi}$) (by default, $P_r = 0$)

Remarks:

a. The power insufficiency is expressed in single-phase pu (denoted by 1ϕ).

b. This data item is optional.

2.7 Base power

```
Sbase = x
```

x : Base power (MW or MVar)

This item is reserved to implement the report of power in MW and MVar in a future version of the program. Currently, the power is reported in pu.

2.8 A few syntactic remarks

- Comments are preceded by the sharp symbol (#).
- A complex number in cartesian form is written by juxtaposing its imaginary part and the letter 'j' (placed on the right), e.g. 1+2j. The letter 'i' cannot be used to that effect.
- The bus data, the branch data, and the zigzag earthing transformer data each take the form of a list of lists. A list consists of a set of elements separated by commas, and bounded by square brackets.

3. Launch command

The program can be launched from the command prompt (terminal) using the following syntax:

```
C:\path_to_nr3r> nr3r filename options
```

filename : Name of the data file, including or not its path relative to the location of *nr3r* (in quotation marks if necessary)

options : *t* to activate the transformer tap changing functionality;
a to display only the phase-*a* voltages of the three-phase buses, and all single-phase bus voltages.

NB Options must be specified in a single term (not separated by spaces).

Remarks:

a. The program can be called by including or not its extension, i.e. *nr3r.exe* and *nr3r* are both valid.

b. This command structure is also applicable to the Linux package.

Example: `C:\Users\saubert\Documents>nr3r test_files\nr3r_test_0.dat ta`

LIST OF REFERENCES

Acha, E. and J. Usaola. 2009. "Three-Phase Linear and Nonlinear Models of Power System Components". In *Electric Energy Systems: Analysis and Operation*, Gómez-Expósito, Antonio, Antonio J. Conejo and Claudio Cañizares (Eds.), p. 265-302. Boca Raton, FL: CRC Press.

Ćalović, M. S. and V. C. Strezoski. 1981. "Calculation of Steady-State Load Flows Incorporating System Control Effects and Consumer Self-Regulation Characteristics". *International Journal of Electrical Power & Energy Systems*, vol. 3, n° 2, p. 65-74.

Glover, J. D., M. S. Sarma and T. J. Overbye. 2012. *Power System Analysis and Design*, 5th ed. Stamford, CT: Cengage Learning, 828 p.

IEC. 2011. *Power transformers – Part 1: General*. International Standard, IEC60076-1. Geneva: International Electrotechnical Commission, 147 p.

Kersting, W. H. 2002. *Distribution System Modeling and Analysis*, 1st ed. Boca Raton, FL: CRC Press, 314 p.

Kirschen, D., R. Allan and G. Strbac. 1997. "Contributions of Individual Generators to Loads and Flows". *IEEE Transactions on Power Systems*, vol. 12, n° 1, p. 52-60.

Kocar, I., J. Mahseredjian, U. Karaagac, G. Soykan and O. Saad. 2014. "Multiphase Load-Flow Solution for Large-Scale Distribution Systems Using MANA". *IEEE Transactions on Power Delivery*, vol. 29, n° 2, p. 908-915.

Lagacé, P.-J. 2012. "Power Flow Methods for Improving Convergence". In *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*. (Montreal, QC, 2012), p. 1387 - 1392.

Okamura, M., Y. O-ura, S. Hayashi, K. Uemura and F. Ishiguro. 1975. "A New Power Flow Model and Solution Method – Including Load and Generator Characteristics and Effects of System Control Devices". *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-94, n° 3, p. 1042-1049.

Penido, D. R. R., L. R. de Araujo, S. Carneiro, J. L. R. Pereira and P. A. N. Garcia. 2008. "Three-Phase Power Flow Based on Four-Conductor Current Injection Method for Unbalanced Distribution Networks". *IEEE Transactions on Power Systems*, vol. 23, n° 2, p. 494-503.

Saadat, M. H. 1979. "Steady State Analysis of Power Systems Including the Effects of Control Devices". *Electric Power Systems Research*, vol. 2, n° 2, p. 111-118.

Slemon, G. R. and A. Straughen. 1982. *Electric Machines*, 1st ed. Reading, MA: Addison-Wesley Publishing Company, 575 p.

Strbac, G., D. Kirschen and S. Ahmed. 1998. "Allocating Transmission System Usage on the Basis of Traceable Contributions of Generators and Loads to Flows". *IEEE Transactions on Power Systems*, vol. 13, n° 2, p. 527-534.

Tinney, W. F. and C. E. Hart. 1967. "Power Flow Solution by Newton's Method". *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-86, n° 11, p. 1449-1460.

Tong, S. and K. N. Miu. 2005. "A Network-Based Distributed Slack Bus Model for DGs in Unbalanced Power Flow Studies". *IEEE Transactions on Power Systems*, vol. 20, n° 2, p. 835-842.

van Ness, J. E. and J. H. Griffin. 1961. "Elimination Methods for Load-Flow Studies". *Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems*, vol. 80, n° 3, p. 299-302.

Wood, A. J. and B. F. Wollenberg. 1996. *Power Generation, Operation, and Control*, 2nd ed. New York, NY: John Wiley & Sons, Inc., 592 p.

Zobian, A. and M. D. Ilić. 1997. "Unbundling of Transmission and Ancillary Services. Part I: Technical Issues". *IEEE Transactions on Power Systems*, vol. 12, n° 2, p. 539-548.