Chapter 4
Globalized Newton–Krylov–Schwarz AC Load Flow Methods for Future Power Systems

Domenico Lahaye and Kees Vuik

Abstract The load flow equations express the balance of power in an electrical power system. The power generated must equal the power consumed. In the AC time-harmonic case, the load flow equations are non-linear in the voltage phasors associated with the nodes in the network. The development of future power systems urgently requires new, highly efficient and robust load flow solvers. In this contribution we aim at making the following three scientific contributions. We first show that the use of a globalization procedure is required to ensure the convergence of a Newton load flow simulation of a stressed network. Such operational conditions are more likely to occur in the future. We subsequently show that the use of an inexact Newton–Krylov method results in faster computations. We employ Quotient Minimal Degree (QMD) as a matrix reordering method, incomplete LU factorization (ILU) as a preconditioner, Generalized Minimal Residual (GMRES) as a Krylov acceleration, and the Dembo-Steihaus strategy to defined the accuracy of the linear solver at each Newton iteration. We finally show the results of iterative solution algorithms that allow to exploit the decomposition of a network into subnetworks. Decompositions with and without overlapping nodes are tested.

4.1 Introduction

The development of new power system architectures motivates the parallel development of new computational tools. These new tools will have to be able to give fast and reliable answers in challenging and unforeseen conditions. Emerging electrical power networks will be operated closer to limits specified in the design, allow bi-directional power flow, accommodate renewable sources and electrical vehicles and merge with gas and heat networks. More simulations will be required to decide on the type of control action to take in the grid. The size of the problems is expected

D. Lahaye (✉) · K. Vuik
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, Delft, The Netherlands
e-mail: d.j.p.lahaye@tudelft.nl

K. Vuik
e-mail: c.vuik@tudelft.nl

P. Palensky et al. (eds.), *Intelligent Integrated Energy Systems*,
https://doi.org/10.1007/978-3-030-00057-8_4

to grow with the extend of geographical interconnections and the level of detail in the network description. Computationally efficient and robust methods will therefore have to be developed.

The load-flow equations describe the flow of electrical power in an electrical power system [1–6]. In a time-harmonic formulation, these equation on non-linear in the complex-valued nodal voltage phasors. Various commercial and public domain software packages that implements solvers for these equations is given on the URL [7]. The Jacobian matrix of partial derivatives can be computed analytically. Newton's method [8] is therefore the solution method of choice. Newton's method is however only guaranteed to converge for initial guesses that are sufficiently close to the solution. In traditional power system analysis, the load-flow solution has an amplitude close to one and an angle close to zero degrees (in per unit values). By specifying an initial guess with unit amplitude and zero angle (referred to as a flat start), Newton's method is likely to converge. In the future power systems, more frequent and large deviations from the nominal solution of the load flow equations are expected to occur. This will lead to nodal voltage and line overloads. In these cases, Newton's method is no longer likely to converge with flat start initial guesses. We show that in these case the convergence can be guided by globalizing the Newton method. In a globalized form, the Newton's method is augmented with auxiliary procedures that ensure convergence from initial guesses that lie further way from the solution.

The Newton method required solving a linear system with a large and sparse Jacobian system at each iteration. Preconditioned Krylov subspace methods are ideally suited to solve these systems [9]. We give examples of how load-flow equations can efficiently be solved using preconditioned Krylov subspace methods. We show numerical results for the load flow solver implemented in the PETSc library [10]. The test cases are taken from the examples provided with the MatPower package [11]. We in particular show how the additive Schwarz method that decomposes the network with and without overlapping nodes can be employed.

This chapter is structured as follows. In Sect. 4.2 the concepts of voltage, current and power for the steady state description of a power system are introduced. In Sect. 4.3 the power system is represented as a graph with generator and load nodes and interconnecting edges. In Sect. 4.4 the load flow equations and the method of Newton to solve them are described. In Sect. 4.5 an example of the computational efficiency of the Newton–Krylov method is given. In Sect. 4.6 an example of solving the load flow equations using the Schwarz method is given. Concluding remarks are made in Sect. 4.7.

4.2 Power Systems

A system of hardware that provides for the generation of power, and the transmission to the consumers, is called a Power System. This section contains an introduction to voltage, current, and power, for the steady state of a power system operating on alternating current (AC).

4.2.1 Voltage and Current

The voltage and current, in a power system in steady state, can be assumed to be sinusoidal functions of time with constant frequency ω. It is conventional to use the cosine function to describe these quantities, i.e.,

$$\begin{aligned} v(t) &= V_{\max}\cos(\omega t + \delta_V) &&= \Re\left(V_{\max}e^{j\delta_V}e^{j\omega t}\right), \\ i(t) &= I_{\max}\cos(\omega t + \delta_I) &&= \Re\left(I_{\max}e^{j\delta_I}e^{j\omega t}\right), \end{aligned}$$

where j is the imaginary unit, and $\Re$ is the operator that takes the real part.

The complex numbers $V = V_{\max}e^{j\delta_V}$ and $I = I_{\max}e^{j\delta_I}$ are called the phasor representation of the voltage and current respectively, and are used to represent the voltage and current in circuit theory. In power system theory, instead the effective phasor representation is used:

$$V = |V|\,e^{j\delta_V}, \text{ with } |V| = \frac{V_{\max}}{\sqrt{2}}, \tag{4.1}$$

$$I = |I|\,e^{j\delta_I}, \text{ with } |I| = \frac{I_{\max}}{\sqrt{2}}. \tag{4.2}$$

Note that $|V|$ and $|I|$ are the RMS values of $v(t)$ and $i(t)$, and that the effective phasors differ from the circuit theory phasors by a factor $\sqrt{2}$.

As we are dealing with power system theory, we use V and I to denote the effective voltage and current phasors, as defined above. Further, as usual in the treatment of power systems, we use per unit quantities, and represent the balanced three-phase network of a power system as its equivalent single-phase system. For more details see for example [4] or [3].

4.2.2 Complex Power

Assume that t is chosen such that the voltage is $v(t) = V_{\max}\cos(\omega t)$, and the current is $i(t) = I_{\max}\cos(\omega t - \phi)$. The value $\phi = \delta_V - \delta_I$ is called the power factor angle, and $\cos\phi$ the power factor. Then the instantaneous power $p(t)$ is given by

$$\begin{aligned} p(t) &= v(t)\,i(t) \\ &= \sqrt{2}\,|V|\cos(\omega t)\,\sqrt{2}\,|I|\cos(\omega t - \phi) \\ &= 2\,|V|\,|I|\cos(\omega t)\cos(\omega t - \phi) \\ &= 2\,|V|\,|I|\cos(\omega t)\cos(\omega t - \phi) \\ &= 2\,|V|\,|I|\cos(\omega t)\left[\cos\phi\cos(\omega t) + \sin\phi\sin(\omega t)\right] \\ &= |V|\,|I|\left[2\cos\phi\cos^2(\omega t) + 2\sin\phi\sin(\omega t)\cos(\omega t)\right] \end{aligned}$$

$$\begin{aligned} &= |V|\,|I| \cos\phi \left[2\cos^2(\omega t)\right] + |V|\,|I| \sin\phi \left[2\sin(\omega t)\cos(\omega t)\right] \\ &= |V|\,|I| \cos\phi \left[1 + \cos(2\omega t)\right] + |V|\,|I| \sin\phi \left[\sin(2\omega t)\right] \\ &= P\left[1 + \cos(2\omega t)\right] + Q\left[\sin(2\omega t)\right], \end{aligned}$$

where $P = |V|\,|I| \cos\phi$, and $Q = |V|\,|I| \sin\phi$.

Thus the instantaneous power is the sum of a unidirectional component, that is sinusoidal with average value P, and amplitude P, and a component of alternating direction, that is sinusoidal with average value 0, and amplitude Q. Note that integrating the instantaneous power over one time period $T = \frac{2\pi}{\omega}$ gives

$$\int_0^T p(t)\,dt = P.$$

The magnitude P is called the active power, or also real power or average power, and is measured in W (watts). The magnitude Q is called the reactive power, or also imaginary power, and is measured in VAr (voltamperes reactive).

Using the complex representation of voltage and current, we can write

$$\begin{aligned} P &= |V|\,|I| \cos\phi = \Re\left(|V|\,|I|\, e^{j(\delta_V - \delta_I)}\right) = \Re\left(V\overline{I}\right), \\ Q &= |V|\,|I| \sin\phi = \Im\left(|V|\,|I|\, e^{j(\delta_V - \delta_I)}\right) = \Im\left(V\overline{I}\right), \end{aligned}$$

where $\overline{I}$ is the complex conjugate of I. Thus we can extend Joule's law to AC circuits, as

$$S = V\overline{I}. \tag{4.3}$$

Note that strictly speaking VA and VAr are the same unit as W, however it is very useful to use descriptive units to distinguish between the measured quantities.

4.2.3 Impedance and Admittance

Impedance is the extension of the notion resistance, to AC circuits, and is thus a measure of opposition to a sinusoidal current. The impedance is denoted by $Z = R + jX$, and measured in ohms (Ω). We call $R \geq 0$ the resistance, and X the reactance. If $X > 0$ the reactance is called inductive and we can write $jX = j\omega L$, where $L > 0$ is called the inductance. If $X < 0$ the reactance is called capacitive and we write $jX = \frac{1}{j\omega C}$, where $C > 0$ is called the capacitance.

The admittance $Y = G + jB$ is the inverse of impedance, i.e., $Y = \frac{1}{Z} = \frac{R}{|Z|^2} - j\frac{X}{|Z|^2}$, and is measured in siemens (S). We call $G = \frac{R}{|Z|^2} \geq 0$ the conductance, and $B = -j\frac{X}{|Z|^2}$ the susceptance.

The voltage drop over an impedance Z is equal to $V = ZI$. This is the extension of Ohm's law to AC circuits. Alternatively, using the admittance, we can write

$$I = YV. \tag{4.4}$$

The power consumed by the impedance is $S = V\overline{I} = ZI\overline{I} = |I|^2 Z = |I|^2 R + j\,|I|^2 X$.

4.2.4 Kirchhoff's Circuit Laws

To calculate the voltage and current in an electrical circuit, we use Kirchoff's circuit laws.

Kirchhoff's current law (KCL)
At any point in the circuit that does not represent a capacitor plate, the sum of currents flowing towards that point is equal to the sum of currents flowing away from that point, i.e., $\sum_k I_k = 0$.

Kirchhoff's voltage law (KVL)
The directed sum of the electrical potential differences around any closed circuit is zero, i.e., $\sum_k V_k = 0$.

4.3 Power System Model

Power systems are modeled as a network of buses (nodes) and lines (edges). At each bus i four electrical magnitudes are of importance:

$|V_i|$, the voltage amplitude,
δ_i, the voltage phase angle,
P_i, the injected active power,
Q_i, the injected reactive power.

Each bus can consist of a number of electrical devices. The bus is named according to the electrical magnitudes specified by its devices,

generator bus or PV-bus: P_i and $|V_i|$ specified, Q_i and δ_i unknown,
load bus or PQ-bus: P_i and Q_i specified, $|V_i|$ and δ_i unknown.

4.3.1 Generators, Loads, and Transmission Lines

Generally, a physical generator has P and $|V|$ controls and thus specifies these magnitudes. Likewise, a load will have specified negative injected active power P, and specified reactive power Q. However, the name of the bus does not necessarily

indicate what type of devices it consists of. A wind turbine, for example, is a generator but does not have PV controls. A wind turbine is instead modeled as a load, with positive injected active power P. If a PV generator and a PQ load are connected to the same bus, the result is a PV-bus with a voltage amplitude equal to that of the generator, and an active power equal to the sum of the active power of the generator and the load. Also there may be buses without a generator or load connected, such as transmission substations, which are modeled as load with $P = Q = 0$.

In any practical power system there are system losses. These losses have to be taken into account, but since they depend on the power flow they are not known in advance. Therefore one generator bus has to be assigned to supply these unknown losses. This bus is generally called the slack bus. Obviously it is no longer possible to specify the real power P for this bus. Instead the voltage magnitude $|V|$ and angle δ are specified. Note that δ does not have any practical meaning, it is merely the reference phase to which the other phase angles are related. As such, for the slack bus it is generally specified that $\delta = 0$.

Lines are the network representation of the transmission lines, that connect the buses in the power system. From a modeling viewpoint, lines define how to relate buses through Kirchhoff's circuit laws. Lines can incur losses on the transported power and must be modeled as such.

A transmission line from bus i to bus j has some impedance. The total impedance over the line is modeled as a single impedance z_{ij} of the line. The admittance of that line is $y_{ij} = \frac{1}{z_{ij}}$. Further there is a shunt admittance from the line to the neutral ground. The total shunt admittance y_s of the line is modeled to be evenly divided between bus i and bus j.

It is usually assumed that there is no conductance from the line to the ground. This means that the shunt admittance is due only to the electrical field between line and ground, and is thus a capacitive susceptance, i.e., $y_s = jb_s$, with $b_s > 0$. For this reason, the shunt admittance y_s is also sometimes referred to as the shunt susceptance b_s. See also the notes about modeling a shunt in Sect. 4.3.2. For details on how to calculate y_s for a given line, we refer to [4].

4.3.2 Shunts, Tap Transformers, and Phase Shifters

Three other devices, that are also commonly found in power systems, are shunts, tap transformers and phase shifters. Shunt capacitors can be used to inject reactive power, resulting in a higher node voltage, whereas shunt inductors consume reactive power, thus lowering the node voltage.

A shunt is modeled as a reactance $z_s = jx_s$ between the bus and the ground. The shunt admittance is thus $y_s = \frac{1}{z_s} = -j\frac{1}{x_s} = jb_s$. If $x_s > 0$ the shunt is inductive, and if $x_s < 0$ the shunt is capacitive. Note that the shunt susceptance b_s has the opposite sign of the shunt reactance x_s.

A tap transformer is a transformer that can be set to different turns ratios. The taps are the connection points along the transformer winding on one side, that allow a certain number of turns to be selected. Tap transformers are generally used to control the voltage magnitude, dealing with fluctuating industrial and domestic demands, or with the effects of switching out a circuit for maintenance. Note that transformers without taps do not need to be modeled specifically, as the per unit system reduces their numerical effect to that of a series impedance.

Phase shifters are devices that can change the voltage phase angle, while keeping the voltage magnitude constant. As such they can be used to control the active power.

The transformer ratio is given by $T : 1$. For a tap transformer T is a positive real number, typically between 0.8 and 1.2. In the case of a phase shifter, T is a rotation in the complex plane, i.e., $T = e^{j\delta_T}$, where δ_T is the phase shift.

4.3.3 Admittance Matrix

The admittance matrix Y, is a matrix that relates the injected current I at each bus to bus voltages V, such that

$$I = YV, \tag{4.5}$$

where I is the vector of injected currents at each bus, and V is the vector of bus voltages. This is in fact Ohm's law (4.4) in matrix form. As such we can also define the impedance matrix $Z = Y^{-1}$.

To calculate the admittance matrix Y, we look at the injected current I_i at each bus i. If $I_i > 0$ power is generated, if $I_i < 0$ power is consumed, and if $I_i = 0$ there is no current injected, for example at a transmission substation. Let I_{ij} denote the current flowing from bus i, in the direction of bus j, or to the ground in case of a shunt. Applying Kirchoff's current law now gives

$$I_i = \sum_k I_{ik}. \tag{4.6}$$

Let y_{ij} denote the admittance of the line between bus i and j, with $y_{ij} = 0$ if there is no line between these buses. For a simple transmission line from bus i to bus j, without shunt admittance, Ohm's law states that

$$I_{ij} = y_{ij}\left(V_i - V_j\right), \text{ and } I_{ji} = -I_{ij}, \tag{4.7}$$

or in matrix notation:

$$\begin{bmatrix} I_{ij} \\ I_{ji} \end{bmatrix} = y_{ij} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} V_i \\ V_j \end{bmatrix}. \tag{4.8}$$

Note that, if a power system would consist of such simplified lines only, the admittance matrix for that system is a Laplacian matrix given by

$$Y_{ij} = \begin{cases} \sum_{k \neq i} y_{ik} & \text{if } i = j, \\ -y_{ij} & \text{if } i \neq j. \end{cases} \tag{4.9}$$

Indeed, then we have

$$I_i = \sum_k I_{ik} = \sum_k y_{ik} \left(V_i - V_k\right) = \sum_{k \neq i} y_{ik} V_i - \sum_{k \neq i} y_{ik} V_k = \sum_k Y_{ik} V_k = (YV)_i \,. \tag{4.10}$$

Now suppose there is a shunt s connected to bus i. Then, according to Eq. (4.6), an extra term I_{is} is added to the injected power I_i. It is clear that

$$I_{is} = y_s \left(V_i - 0\right) = y_s V_i. \tag{4.11}$$

This means that an extra term y_s has to be added to Y_{ii} element of the admittance matrix. Recall that $y_s = jb_s$, and that the sign of b_s depends on the shunt being inductive or capacitive.

Knowing how to deal with shunts, it is now easy to incorporate the line shunt admittance model. In the admittance matrix, for a transmission line between the buses i and j, the halved line shunt admittance $\frac{y_s}{2}$ of that line, has to be added to both Y_{ii} and Y_{jj}. For a transmission line with shunt admittance y_s, we thus find

$$\begin{bmatrix} I_{ij} \\ I_{ji} \end{bmatrix} = \left(y_{ij} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + y_s \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \right) \begin{bmatrix} V_i \\ V_j \end{bmatrix}. \tag{4.12}$$

The influence on the admittance matrix, of a device t between the buses i and j, that is either a tap transformer or a phase shifter, can be derived from the model. Let E be the voltage induced by t, then

$$V_i = TE. \tag{4.13}$$

Thus, the current from bus j to t in the direction of bus i is

$$I_{ji} = y_{ij} \left(V_j - E\right) = y_{ij} \left(V_j - \frac{V_i}{T}\right). \tag{4.14}$$

Conservation of power gives

$$V_i \overline{I}_{ij} = -E \overline{I}_{ji} \Rightarrow T \overline{I}_{ij} = -\overline{I}_{ji} \Rightarrow \overline{T} I_{ij} = -I_{ji}. \tag{4.15}$$

and thus, for the current from bus i to t in the direction of j we find

$$I_{ij} = -\frac{I_{ji}}{\overline{T}} = y_{ij} \left(\frac{V_i}{|T|^2} - \frac{V_j}{\overline{T}}\right). \tag{4.16}$$

If the device t that connects bus i to bus j is a tap transformer, then $\overline{T} = T$, and $|T|^2 = T^2$, and thus instead of the admittance matrix values from Eq. (4.12), we find

$$\begin{bmatrix} I_{ij} \\ I_{ji} \end{bmatrix} = y_{ij} \begin{bmatrix} \frac{1}{T^2} & -\frac{1}{T} \\ -\frac{1}{T} & 1 \end{bmatrix} \begin{bmatrix} V_i \\ V_j \end{bmatrix}. \tag{4.17}$$

If instead t is a phase shifter, then $\overline{T} = e^{-j\delta_T} = \frac{1}{T}$, and $|T|^2 = 1$, and we find

$$\begin{bmatrix} I_{ij} \\ I_{ji} \end{bmatrix} = y_{ij} \begin{bmatrix} 1 & -T \\ -\overline{T} & 1 \end{bmatrix} \begin{bmatrix} V_i \\ V_j \end{bmatrix}. \tag{4.18}$$

4.4 Load Flow

The load flow problem, or power flow problem, is the problem of computing the flow of electrical power in a power system in steady state. In practice, this amounts to calculate all node voltages and line currents in the power system. The load flow is regarded as the most important network computation. The problem arises in many applications in power system analysis, and is treated in many books on power systems, see for example [4] or [3].

The mathematical equations describing the load flow problem can be obtained by combining Ohm's law from Eq. (4.5), with Joule's law:

$$S_i = V_i \overline{I}_i = V_i \left(\overline{YV}\right)_i = V_i \sum_{k=1}^{N} \overline{Y}_{ik} \overline{V}_k. \tag{4.19}$$

Note that the admittance matrix Y is easy to obtain, and generally very sparse. Therefore a formulation using the admittance matrix has preference over using the impedance matrix, which is generally a lot harder to obtain and not sparse.

Further note that for each bus i that has no injected power, i.e., $S_i = 0$, we also have that injected current $I_i = 0$. Therefore, we can use the linear equation $(YV)_i = 0$, to eliminate the variable V_i from the problem. This method is called Kron reduction, see [12] Sect. 9.3. The elimination can be done by simple Gaussian elimination, but in the case of many such buses, using a Schur complement may provide better results.

Recall that voltage and current are complex numbers, and substitute $V_i = |V_i| e^{j\delta_i}$, $Y = G + jB$, and $\delta_{ij} = \delta_i - \delta_j$ into Eq. (4.19). This gives

$$S_i = |V_i| e^{j\delta_i} \sum_{k=1}^{N} (G_{ik} - jB_{ik}) |V_k| e^{-j\delta_k} = \sum_{k=1}^{N} |V_i| |V_k| (\cos\delta_{ik} + j \sin\delta_{ik}) (G_{ik} - jB_{ik}) \tag{4.20}$$

Define the real vector of the voltage variables of the load flow problem as

$$\mathbf{V} = [\ \delta_1, \ldots, \delta_N, |V_1|, \ldots, |V_N|\ ]^T. \tag{4.21}$$

For the purpose of notational comfort, we further define

$$P_{ij}(\mathbf{V}) = |V_i|\left|V_j\right|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right), \tag{4.22}$$

$$Q_{ij}(\mathbf{V}) = |V_i|\left|V_j\right|\left(G_{ij}\sin\delta_{ij} - B_{ij}\cos\delta_{ij}\right). \tag{4.23}$$

Then we can write equation (4.20) as

$$S_i = \sum_{k=1}^{N} P_{ik}(\mathbf{V}) + j\sum_{k=1}^{N} Q_{ik}(\mathbf{V}). \tag{4.24}$$

Splitting the real and imaginary parts of the equation, and defining $P_i(\mathbf{V}) = \sum_{k=1}^{N} P_{ik}(\mathbf{V})$ and $Q_i(\mathbf{V}) = \sum_{k=1}^{N} Q_{ik}(\mathbf{V})$, we have

$$P_i = P_i(\mathbf{V}), \tag{4.25}$$

$$Q_i = Q_i(\mathbf{V}). \tag{4.26}$$

Equations (4.25)–(4.26) relate the complex power in each node to the node voltages, using the admittance matrix of the power system network. It consists of $2N$ non-linear real equations. Each node has four variables, $|V_i|$, δ_i, P_i and Q_i, two of which have a specified value, as discussed in Sect. 4.3. Thus we have $2N$ non-linear real equations in $2N$ real unknowns, commonly known as the load flow equations, or power flow equations.

4.4.1 Newton–Raphson

It is generally not possible to solve a system of non-linear equations analytically. However, there are many iterative techniques to approximate the solution. The problem of solving a system of equations is trivially equivalent to find the simultaneous roots of a set of functions in the same variables. As such, we can deploy root finding algorithms as a tool to solve a system of equations. Of these algorithms, Newton's method, also known as the Newton–Raphson method, is widely accepted to be the best for many applications. Newton's method is the base algorithm of choice when solving load flow equations.

First, define a real vector of the voltage variables of the load flow problem:

$$\mathbf{V} = [\,\delta_1, \ldots, \delta_N, |V_1|, \ldots, |V_N|\,]^T. \tag{4.27}$$

Then write the load flow equations as a set of functions $\mathcal{F}_i$, whos simultaneous roots are the solution of the load flow equations:

$$\mathcal{F}_i(\mathbf{V}) = \begin{bmatrix} \Delta P_i(\mathbf{V}) \\ \Delta Q_i(\mathbf{V}) \end{bmatrix} = \begin{cases} P_i - P_i(\mathbf{V}), & i = 1 \ldots N, \\ Q_i - Q_i(\mathbf{V}), & i = N+1 \ldots 2N, \end{cases} \tag{4.28}$$

where $S_i = P_i + jQ_i$ are the desired values of the power, whereas $S_i(\mathbf{V}) = P_i(\mathbf{V}) + jQ_i(\mathbf{V})$ are the values dictated by the bus voltages and network admittance matrix, i.e., $S_i(\mathbf{V})$ is the right-hand side in Eq. (4.20). The function $\mathcal{F}$ is called the power mismatch.

Next, we start with some guess or approximation $\mathbf{V}^0$, and update it iteratively by the rule

$$\mathbf{V}^{k+1} = \Phi\left(\mathbf{V}^k\right), \tag{4.29}$$

where the function Φ is such that

$$\Phi(\mathbf{V}) = \mathbf{V} \Leftrightarrow \mathcal{F}(\mathbf{V}) = \mathbf{0}, \tag{4.30}$$

with $\mathcal{F}$ the vector of functions $\mathcal{F}_i$. Condition (4.30) is always satisfied, if we choose

$$\Phi(\mathbf{V}) = \mathbf{V} - A(\mathbf{V})^{-1}\mathcal{F}(\mathbf{V}), \tag{4.31}$$

with $A(\mathbf{V})$ a non-singular matrix. Different choices of $A(\mathbf{V})$ correspond to different methods. For example, $A = I$ leads to the Gauss–Seidel method.

Newton's method is based on the first order Taylor expansion of $\mathcal{F}$, and as such uses $A(\mathbf{V}) = J(\mathbf{V})$, with J the Jacobian of $\mathcal{F}$. Thus, Newton's method is characterized by the update rule

$$\mathbf{V}^{k+1} = \mathbf{V}^k + \Delta\mathbf{V}^k, \tag{4.32}$$

where $\Delta\mathbf{V}^k$ is the solution of the linear system of equations

$$-J\left(\mathbf{V}^k\right)\Delta\mathbf{V}^k = \mathcal{F}\left(\mathbf{V}^k\right). \tag{4.33}$$

Convergence of Newton's method is not guaranteed. However, when the starting approximation is sufficiently close to the solution, the convergence is quadratic. Note that the Jacobian generally differs in each iteration. We will refer to this method as full Newton, as opposed to inexact Newton methods which make some kind of approximation, often in the Jacobian. Inexact Newton methods generally have linear convergence.

4.4.2 Generator Buses

When solving a linear system of the form $A\mathbf{x} = \mathbf{b}$, we assume the coefficient matrix A, and right-hand side vector $\mathbf{b}$ to be known, and the variable vector $\mathbf{x}$ to be unknown. In the case of the linear system from Eq. (4.33), each bus i of the power system makes for two equations, begin row i and row $N + i$. If bus i is a load bus, then indeed all coefficients, and the right-hand side value, of row i and row $N + i$ are known, whereas δ_i and $|V_i|$ are unknown. However, if bus i is a generator bus, then we have a different situation.

First let us consider the slack bus, of which every power system has exactly one, and assume that it is located at bus 1. Then, δ_1 and $|V_1|$ are known, whereas P_1 and Q_1 are not. As a result, we can eliminate the variables δ_1 and $|V_1|$ from the system, by substituting their values into the coefficient matrix, and bringing the resulting values to the right-hand side. Also, we can take row 1, which corresponds to the equation for P_1, and row $N+1$, which corresponds to the equation for Q_1, out of the system. Note that the resulting coefficient matrix, is a principal minor of the original matrix.

Once the Newton–Raphson process has converged, P_1 and Q_1 are easily calculated by substituting the found solution into the original equations 1 and $N+1$, that were taken out.

Now consider a generator bus i that is not the slack bus. Then P_i and $|V_i|$ are known, whereas Q_i and δ_i are unknown. With $|V_i|$ known, we can eliminate this variable, and column $N+i$ from the linear system. And since Q_i is unknown, we also take row $N+i$ out of the system, like we did in the case of a slack bus. Note that again, the resulting coefficient matrix is a principal minor of the original matrix.

4.4.3 Full Newton Jacobian

Applying Newton's method to the load flow problem, in each iteration we have to solve the linear system of equations (4.33), and update the approximate solution according to Eq. (4.32). All terms in these equations are straightforward to calculate, except for the Jacobian.

In this section we will derive the explicit form of the Jacobian of the load flow equations. The general structure of the Jacobian is

$$J(\mathbf{V}) = -\left[\begin{array}{ccc|ccc} \frac{\partial P_1}{\partial \delta_1} & \cdots & \frac{\partial P_1}{\partial \delta_{N_1}} & \frac{\partial P_1}{\partial |V_1|} & \cdots & \frac{\partial P_1}{\partial |V_{N_2}|} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial P_{N_1}}{\partial \delta_1} & \cdots & \frac{\partial P_{N_1}}{\partial \delta_{N_1}} & \frac{\partial P_{N_1}}{\partial |V_1|} & \cdots & \frac{\partial P_{N_1}}{\partial |V_{N_2}|} \\ \hline \frac{\partial Q_1}{\partial \delta_1} & \cdots & \frac{\partial Q_1}{\partial \delta_{N_1}} & \frac{\partial Q_1}{\partial |V_1|} & \cdots & \frac{\partial Q_1}{\partial |V_{N_2}|} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Q_{N_2}}{\partial \delta_1} & \cdots & \frac{\partial Q_{N_2}}{\partial \delta_{N_1}} & \frac{\partial Q_{N_2}}{\partial |V_1|} & \cdots & \frac{\partial Q_{N_2}}{\partial |V_{N_2}|} \end{array}\right], \tag{4.34}$$

where the dependance of P_i and Q_i of $\mathbf{V}$ has been left out for readability. The dimensions N_1 and N_2 are due to the elimination of the slack bus, and other generator buses.

Below the first order derivatives, of which the Jacobian exists, are derived. Note that we assume that $i \neq j$ wherever applicable.

$$\frac{\partial P_i(\mathbf{V})}{\partial \delta_j} = |V_i|\left|V_j\right|\left(G_{ij}\sin\delta_{ij} - B_{ij}\cos\delta_{ij}\right) = Q_{ij}(\mathbf{V}) \tag{4.35}$$

$$\frac{\partial P_i(\mathbf{V})}{\partial \delta_i} = \sum_{k\neq i}|V_i||V_k|\left(-G_{ik}\sin\delta_{ik} + B_{ik}\cos\delta_{ik}\right) = -Q_i(\mathbf{V}) - |V_i|^2 B_{ii} \tag{4.36}$$

$$\frac{\partial Q_i(\mathbf{V})}{\partial \delta_j} = |V_i|\left|V_j\right|\left(-G_{ij}\cos\delta_{ij} - B_{ij}\sin\delta_{ij}\right) = -P_{ij}(\mathbf{V}) \tag{4.37}$$

$$\frac{\partial Q_i(\mathbf{V})}{\partial \delta_i} = \sum_{k\neq i}|V_i||V_k|\left(G_{ik}\cos\delta_{ik} + B_{ik}\sin\delta_{ik}\right) = P_i(\mathbf{V}) - |V_i|^2 G_{ii} \tag{4.38}$$

$$\frac{\partial P_i(\mathbf{V})}{\partial \left|V_j\right|} = |V_i|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right) = \frac{P_{ij}(\mathbf{V})}{\left|V_j\right|} \tag{4.39}$$

$$\frac{\partial P_i(\mathbf{V})}{\partial |V_i|} = 2|V_i|G_{ii} + \sum_{k\neq i}|V_k|\left(G_{ik}\cos\delta_{ik} + B_{ik}\sin\delta_{ik}\right) = \frac{P_i(\mathbf{V})}{|V_i|} + |V_i|G_{ii} \tag{4.40}$$

$$\frac{\partial Q_i(\mathbf{V})}{\partial \left|V_j\right|} = |V_i|\left(G_{ij}\sin\delta_{ij} - B_{ij}\cos\delta_{ij}\right) = \frac{Q_{ij}(\mathbf{V})}{\left|V_j\right|} \tag{4.41}$$

$$\frac{\partial Q_i(\mathbf{V})}{\partial |V_i|} = -2|V_i|B_{ii} + \sum_{k\neq i}|V_k|\left(G_{ik}\sin\delta_{ik} - B_{ik}\cos\delta_{ik}\right) = \frac{Q_i(\mathbf{V})}{|V_i|} - |V_i|B_{ii} \tag{4.42}$$

Summarizing, again leaving out the dependance on $\mathbf{V}$:

$$\begin{array}{ll|ll}
\dfrac{\partial P_i}{\partial \delta_j} = Q_{ij} & & \left|V_j\right|\dfrac{\partial P_i}{\partial \left|V_j\right|} = P_{ij} \\
\dfrac{\partial P_i}{\partial \delta_i} = -Q_i - |V_i|^2 B_{ii} & & |V_i|\dfrac{\partial P_i}{\partial |V_i|} = P_i + |V_i|^2 G_{ii} \\
\hline
\dfrac{\partial Q_i}{\partial \delta_j} = -P_{ij} & & \left|V_j\right|\dfrac{\partial Q_i}{\partial \left|V_j\right|} = Q_{ij} \\
\dfrac{\partial Q_i}{\partial \delta_i} = P_i - |V_i|^2 G_{ii} & & |V_i|\dfrac{\partial Q_i}{\partial |V_i|} = Q_i - |V_i|^2 B_{ii}
\end{array}$$

Algorithm 1 Newton–Raphson Method

1: $k := 0$
2: given initial iterate $\mathbf{V}_0$
3: **while** not convergenced **do**
4: solve $-J(\mathbf{V}_k)\Delta\mathbf{V}_k = \mathcal{F}(\mathbf{V}_k)$
5: update $\mathbf{V}_{k+1} := \mathbf{V}_k + \Delta\mathbf{V}_k$
6: k := k+1
7: **end while**

4.4.4 Globalized Newton

The convergence of the Newton–Raphson method is known to critically depend on the choice of the initial guess $\mathbf{V}_0$. Convergence is likely to fail in case that the initial guess is too far from the solution. In load flow computations the initial guess is typically chosen as $\mathbf{V}_0 = \mathbf{1}$ (in per unit values). This is referred to as the *flat start* solution. With this initial guess, the Newton iteration is likely to converge in a few iterations in problems for which the solution $\mathbf{V}$ satisfies $|V_i| \approx 1$ and $\delta_i \approx 0$. In optimization studies and in Monte-Carlo simulations, the load flow simulation is placed in a outer loop [13]. For these problems the flat start is likely to lie outside the basin of attraction of the Newton method. The same is true for stressed networks for which the solution is likely to deviate substantially from $|V_i| \approx 1$ and $\delta_i \approx 0$. Some kind of save guarding from bad initial guesses is therefore mandatory.

To saveguard against bad initial guesses, a code that implements the Newton–Raphson algorithm is typically augmented with other code that iteratively improves the current initial guess using either line-search or trust-region algorithms. The update formula (4.32) is replaced by a trust-region subproblem. The Powell dogleg procedure [8] is an efficient method to solve this subproblem. Other globalization methods are discussed in [14].

To give an example of the effect of saveguarding against bad initial guesses, we give in Fig. 4.1 the convergence history of the Newton algorithm with and without saveguarding [15]. As example we took the test case **case2383wp.m** from the MatPower [11] test collection from to which we added a generator. Figure 4.1 shows the convergence of the Newton's method to solve the load flow equations with and without globalization. The algorithm without globalization is the implementation in the function `newtonpf.m` in MatPower package. The method with globalization using the Powell dogleg procedure is obtained using the function `fsolve` in Matlab. Without globalization the plot shows that Newton's method does not converge. With globalization instead convergence is obtained.

Algorithm 2 Inexact Newton–Raphson Method

1: $k := 0$
2: given initial iterate $\mathbf{V}_0$
3: **while** not convergenced **do**
4: minimize $\mathcal{F}(\mathbf{V}_k + s\mathbf{W}_k)$
5: update $\mathbf{V}_{k+1} := \mathbf{V}_k + \Delta\mathbf{V}_k$
6: k := k+1
7: **end while**

4.5 Newton–Krylov Method

Newton's method requires solving a linear system at each iteration. In literature [3, 4] this linear system solve is typically performed by a direct solution method. Cur-

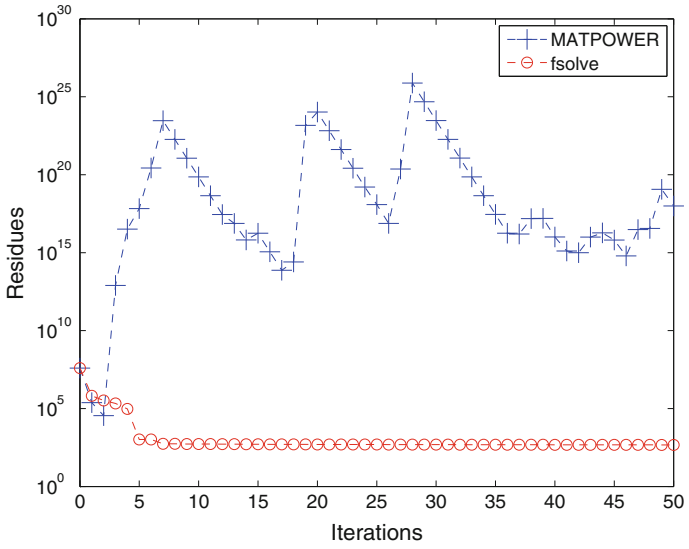

Fig. 4.1 Comparison of Newton-only (`newtonpf.m` in MatPower) and globalized Newton (`fsolve` in Matlab) algorithm. Plot of the residuals versus iterations for both Newton's method and the trust-region method for a version of **case2383wp.m** from MatPower [11] to which a generator was added. The converge of the trust-region method is seen to be more stable

rently however there exists a pressing need to compute the power flow in networks that are large in size. Such networks originate for instance in the modeling of the interconnection of the network Europe with that in Russia or in Northern Africa. It is well known that direct solution methods are not suited for such large scale problems [9, 16]. A new generation of load flow solvers in which the LU-factorization of Jacobian $J(\mathbf{V}_k)$ is replaced by a preconditioned Krylov subspace iteration is available in literature [2]. The term Newton–Krylov for these new solvers derives from the fact the accuracy of the inner linear solved is linked to the outer non-linear iteration residual. The PETSc software library is a public domain software library that implements a load flow solver called `pflow` [17] that allows to employ these modern iterative solvers. This solver employs the `DMNetwork` data structures to manage the network data, the System of Nonlinear Equation Solvers (`SNES`) component to handle the non-linearity of the equations and the Preconditioning (`PC`) and Krylov Subspace (`KSP`) components to provide the preconditioners and the Krylov subspace solvers.

In Table 4.1 we present statistics on the performance of the `pflow` solver applied to the test example **case6468rte.m** taken from MatPower [11]. As Krylov subspace solver we employ the Generalized Minimal Residual Method (GMRES). We vary the level of fill-in in the incomplete LU factorization from 0 to 16. We list the number of Newton iterations, the overall number of GMRES iterations (sum of 5 runs), the number of non-zero elements in the preconditioner, the level of fill-in in the preconditioner relative to the number of non-zero elements in the original

Table 4.1 Numerical results for the Newton–Krylov AC load flow method for the **case6468rte.m** test case from MatPower [11] for various level of fill-in in the ILU preconditioner without QMD reordering (top) and with QMD reordering (bottom). Tabulated are the number of Newton iteration, the total number of GMRES iterations, the number of non-zeros in the preconditioner, the fill-in ratio in the preconditioner and the total simulation time

Newton–Krylov AC load flow on **case6468rte.m**					
Without QMD reordering of J ($\mathbf{V}_k$)					
Preconditioner	ILU(0)	ILU(2)	ILU(4)	ILU(8)	ILU(16)
Newton iterations	–	5	5	5	5
GMRES iterations	–	541	152	73	20
nnz(L+U)	–	215.648	413.080	1.449.264	7.063.408
Fill ratio	–	2.39	4.57	16.03	78.14
Time(s)	–	2.133	1.930	4.658	77.58
With QMD reordering of J ($\mathbf{V}_k$)					
Preconditioner	ILU(0)	ILU(2)	ILU(4)	ILU(8)	ILU(16)
Newton iterations	5	5	5	5	5
GMRES iterations	1213	163	76	40	15
nnz(L+U)	90.392	115.104	126.520	138.600	1.495.584
Fill ratio	1	1.27	1.49	1.53	1.65
Time(s)	3.025	1.277	1.092	1.044	0.988

matrix and the overall CPU time. We compare runs without (top) and with (bottom) use of Quotient Minimal Degree reordering. In all runs, 5 Newton iterations are required to reach convergence. Both with and without QMD reordering, the number of GMRES iterations decreases as expected as the level of fill-in in the preconditioner is increased. Without the use of QMD reordering, however, the number of non-zeros in the preconditioner, the level of fill-in in the preconditioner and the overall CPU time increases significantly. With the use of QMR reordering in contrast, the fill ratio of the preconditioner remains bounded by 1.65. When QMD reordering is used, the highest level of fill-in in the preconditioner is seen to deliver the shortest overall run time. More results on the Newton–Krylov method for load flow computations are available in [2, 18].

4.6 Newton–Krylov–Schwarz Method

In particular applications, it is useful to view a power system network as an interconnection of various subnetworks. An illustrative example is given in Fig. 4.2. This figure shows the decomposition of the **case9.m** test case from MatPower into

subnetworks. Subnetworks can be owned and operated by different entities. Despite the shared ownership, one would like to simulate the entire network without violating sensitive issues on the propriety of the network data. This problem is referred to as co-simulation and is discussed extensively elsewhere in this book.

The Newton–Krylov methodology discussed in this chapter can be adapted to allow some form of co-simulation of an interconnection of various power systems. Indeed, by adopting the Schwarz method as a preconditioner for the linear system solve at each Newton iteration, the linear solver can exploit the network decomposition. The Schwarz method was developed as a domain decomposition method for solving large scale elliptic and parabolic partial differential equations in parallel [19, 20]. In its most simple form, the Schwarz preconditioner is equivalent to be block Jacobi (additive form) or block Gauss–Seidel (multiplicative form). Schwarz methods decompose the problem into subproblems, perform local solves and employ interface conditions to stitch the individual parts together to obtain the global solution.

The load flow solver `pflow` implemented in PETSc [10] does provide Schwarz-type preconditioners. This solver can therefore be used to solve the load flow equation in a distributed fashion. We applied the Newton–Krylov–Schwarz algorithm implemented implemented in `pflow` to a synthetically constructed test case that we will refer to as test case **case5timespegase.m**. This test case was constructed by doubling the test case **case9241pegase.m** from MatPower five times and establishing links between these copies. It resulting network has $9.242 * 32 = 295.712$ nodes. The procedure followed to construct the test case is the same as the one outlined in [2]. Numerical results are given in Table 4.2. This table lists the number of Newton iterations, the total amount of linear iterations and the total CPU time for a decomposition into two (top) and four (bottom) subnetworks. For both decompositions, we tested an additive form of the Schwarz method for various amounts of overlapping nodes ranging between 0 and 2. The results in Table 4.2 show that the number of Newton iterations does not vary with the number of subnetworks and the amount of overlap. This can be explained by the fact that, at each Newton iteration, the linear system is solved to full accuracy for all parameter settings. The number of linear iterations decreases significantly by increasing the overlap from 0 to 1. Increasing the amount of overlap even more does not result in a significant further reduction in the number of iterations. The number of linear iterations increases with the number of subnetworks. More iterations are indeed required to pass information from one network to all the others. This report on the number of iterations is similar to what is reported in the literature on the Schwarz method for discretized elliptic partial differential equations, see [19, 20] and the references cited therein. The CPU time is a complex function of the number of iterations, the amount of overlap and the number of subnetworks. More results of distributed load flow computations are available in [21].

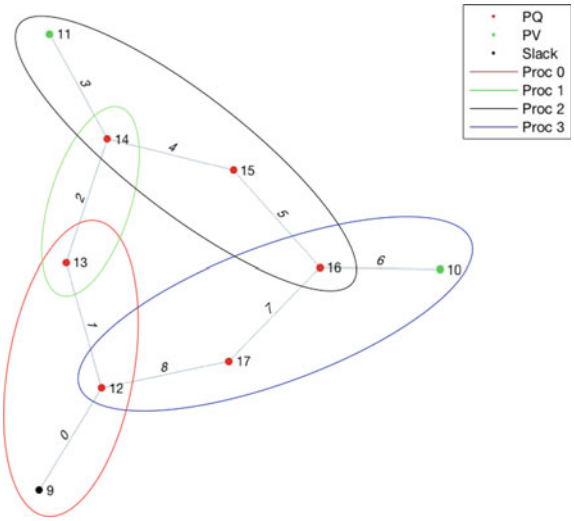

Fig. 4.2 Decomposition of the **case9.m** network into four subnetworks

Table 4.2 Numerical results for the Newton–Krylov–Schwarz AC load flow method applied to the **case5timespegase.m** test case. This test case was constructed by copying **case9241pegase.m** from MatPower [11] five times and interconnecting these copies. Decompositions into two and four subnetworks with various amounts of overlapping nodes ranging between 0 and 3 were tested. Tabulated are the number of Newton iteration, the total number of GMRES iterations and the total simulation time

Newton–Krylov–Schwarz AC Load Flow on **case5timespegase.m**			
Decomposition in two subnetworks			
Overlapping edges	Non-linear its.	Linear its.	CPU time (s)
0	7	72	81
1	7	38	77
2	7	30	72
3	7	30	76
Decomposition in four subnetworks			
Overlapping edges	Non-linear its.	Linear its.	CPU time (s)
0	7	100	54
1	7	49	51
2	7	39	50
3	7	37	62

4.7 Conclusions

We discussed the globalized Newton–Krylov–Schwarz method to solve the AC load flow equation in power systems. We showed that the globalization of the Newton method is required to obtain convergence for stressed networks. We argued that Newton–Krylov methods yield efficient solvers and that Schwarz methods allow to distribute computations over subnetworks.

Acknowledgements The results on the globalization of the Newton method using the `fsolve` function in Matlab resulted from a fruitful collaboration with VVTP Applied Physics student association. Results from the Newton–Krylov method using `pflow` implemented in PETSc resulted from the master thesis project of Jonathan Aviles. Results from the Newton–Krylov–Schwarz using again `pflow` resulted from the master thesis project of the students Andrea Ceresoli and Stefano Guido Rinaldo.

References

1. J. Arrillaga, C.P. Arnold, *Computer Analysis of Power Systems* (Wiley, New Jersey, 1990)
2. R. Idema, D. Lahaye, *Computational Methods in Power System Analysis* (Atlantis Press, Amsterdam, 2014)
3. W.H. Kersting, *Distribution System Modeling and Analysis*, 3rd edn. (Taylor & Francis, Abingdon, 2012)
4. P. Schavemaker, L. van der Sluis, *Electrical Power System Essentials* (Wiley, New Jersey, 2008)
5. M.H. Bollen, F. Hassan, *Integration of Distributed Generation in the Power System*, IEEE Press Series on Power Engineering (Wiley, New Jersey, 2011)
6. N. Hatziargyriou, *Microgrids: Architectures and Control* (IEEE, Wiley, New Jersey, 2014)
7. http://www.openelectrical.org/wiki/index.php?title=Power_Systems_Analysis_Software
8. J. Nocedal, S. Wright, *Numerical Optimization* (Springer, New York, 2006)
9. Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edn. (SIAM, Philadelphia, 2003)
10. S. Balay, S. Abhyankar, M.F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, K. Rupp, B.F. Smith, S. Zampini, H. Zhang, H. Zhang, PETSc Web Page (2016), http://www.mcs.anl.gov/petsc
11. C.E. Murillo-Sánchez, R.D. Zimmerman, C.L. Anderson, R.J. Thomas, Secure planning and operations of systems with stochastic sources, energy storage and active demand. IEEE Trans. Smart Grid **4**(4), 2220–2229 (2013)
12. A.R. Bergen, V. Vittal, *Power Systems Analysis* (Pearson/Prentice Hall, New Jersey, 2000)
13. M. de Jong, G. Papaeffhymiou, D. Lahaye, C. Vuik, L. van der Sluis, Impact of correlated infeeds on risk-based power system security assessment, in *Power Systems Computation Conference (PSCC)* (Wroclaw, Poland, 2014). https://doi.org/10.1109/PSCC.2014.7038439
14. P.J. Lagacé, M.H. Vuong and I. Kamwa, Improving power flow convergence by Newton Raphson with a Levenberg-Marquardt method, in *IEEE Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century* (2008), pp. 1–6
15. M. De Beurs, P. De Graaf, P. Hansler, S. Hermans, K. Van Walstijn, J. De Winter, D.J.P. Lahaye, Optimal configuration of the future electricity grid. DIAM TU Delft Technical Report 16-01 (2016)
16. R. Idema, G. Papaefthymiou, D. Lahaye, C. Vuik, L. van der Sluis, Towards faster solution of large power flow problems. IEEE Trans. Power Syst. **28**(4), 4918–4925 (2013)
17. S. Abhyankar, B.F. Smith, PETSc: an advanced math and computing framework for rapidly developing parallel smart grid applications, in *Proceedings of the IEEE PES General Meeting* (2013)

18. J. Aviles Cedeño, A three-phase unbalanced load flow solver for large-scale distribution power systems. TU Delft Master thesis (2017). (uuid:0d750fa1-b349-4459-8ba7-5f2a3bbf0c87)
19. A. Toselli, O. Widlund, *Domain Decomposition Methods - Algorithms and Theory*, Springer Series in Computational Mathematics (Springer, Berlin, 2004)
20. B. Smith, P. Bjorstad, W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations* (Cambridge University Press, Cambridge, 2004)
21. S. Guido Rimaldo, A. Ceresoli, Newton-Krylov-Schwarz methods for distributed load flow and related applications. Master thesis report, School of Industrial and Information Engineering, Politecnico di Milano (2018)