Power Flow Algorithm using a Second-order Differentation Approach

F. M. Echavarren, L. Rouco, A. Benítez, L. Sigrist
Instituto de Investigación Tecnológica — IIT
Escuela Técnica de Ingeniería — ICAI
Universidad Pontificia Comillas
Madrid, SPAIN
francisco.echavarren@iit.comillas.edu

***Abstract* — Load flow is the key tool for most of the studies related with electric power systems operation and planning. For a given electric network and power dispatch, power flow provides the state variables of the system, i.e. bus voltages magnitude and angle. In most of the cases, power flow must be run several times to cover different dispatches, topologies, outages, etc. Therefore, power flow algorithms must be fast and robust. This paper presents a second-order formulation of the Newton-Raphson method applied to the power flow problem. Power flow equations are reformulated as a homotopy that undergoes a manifold between initial point and the solution. The updating vector at each iteration is computed using first and second-order derivatives. The performance of the algorithm is illustrated using the IEEE 39 buses test network.**

***Index Terms* – convergence; Newton-Raphson method; power flow; second-order approach.**

I. INTRODUCTION

Power flow (PF) is key tool in for electric power systems planning and operation studies. PF provides the steady-state solution of a power system scenario. A scenario is defined by the network topology, the loads, the active power generation and the generator voltages. Basic state variables of the PF are bus voltage magnitudes and angles. Since bus power injections are nonlinear functions of the state variables, PF problem consists of solving a set of nonlinear equations, thus iterative algorithms are required to tackle the PF problem. Unfortunately, convergence problems may appear in ill-conditioned systems and the iterative process may either oscillate or diverge. The boundary between solvable and unsolvable case scenarios is known as the voltages collapse region. Consequently, fast and robust PF tools are always needed for higher level tools, such as security analysis, SCOPF, dynamic studies, etc.

Different approaches have been proposed in the literature to solve the problem of PF, and, what is even more useful, to verify the ability to solve the problem of PF and then, if the solution exists, calculate it regardless of how close the voltage collapse point system. Several of these works are based in the most commonly method used for power flow calculation, the Newton-Raphson (NR) method. The NR updating process is based on successive solutions of the linear approximation of the power flow equations. For most of the situations, NR is robust and fast as a consequence of the development of highly efficient sparse matrix techniques for direct solution of linear systems. However, close to the solvability boundary, NR presents convergence issues.

This paper presents a second-order approach for the NR algorithm applied to the PF problem. The PF equations are reformulated as a homotopy, which creates a trajectory that connects the current point of the system with the desired solution. By computing the first-order derivatives of the trajectories only, a linear updating vector is computed, i.e. a standard NR iteration is carried out. By computing the second-order derivatives of the trajectories, a parabolic updating vector is obtained, being this more accurate than the linear one.

The paper is organized as follows. First, section II presents the PF problem and review the techniques present in the literature to solve it. Then, section III introduces the formulation of the method to obtain the second-order PF updating vector presented in this paper. In section IV, the second-order PF algorithm used in this paper is described, remarking the integration of the first and second-order updating vectors considered. To illustrate the performance of the proposed parabolic predictor, the standard IEEE 39 buses test network is analyzed in section V, and the proposed second-order PF algorithm is compared with the standard NR. Finally, section VI contains the conclusions of the paper.

II. POWER FLOW PROBLEM FORMULATION AND SOLUTION

First task associated to almost every power system static or dynamic analysis is the power flow, the key tool in power system planning and operation. The resolution of the load flow problem provides the steady state point of operation of a power system scenario, that is, it gets the root of the power flow equations [1].

$$\begin{aligned} MP_m(\boldsymbol{\theta},\mathbf{V}) &= P_m(\boldsymbol{\theta},\mathbf{V}) - Pe_m = 0 \\ MQ_m(\boldsymbol{\theta},\mathbf{V}) &= Q_m(\boldsymbol{\theta},\mathbf{V}) - Qe_m = 0 \end{aligned} \tag{1}$$

Power flow equations (1) are formulated by making zero the difference between specified and real node power. The specified node power is the balance between generation and demand. The real node power is the injected power from the node into the network, and it is function of node voltages and network configuration. State variables of the system are node voltage magnitudes and angles, whereas any other magnitude of the system, such as generation, demand, scheduled voltages of generation units or even the parameters of the network branches, i.e. power lines and transformers, may be considered as control variables.

Due to the parabolic (voltage magnitudes) and trigonometric (voltage angles) nature of power flow equations, power flow is a nonlinear problem that may be feasible or infeasible. A

2023 IEEE Belgrade PowerTech | 978-1-6654-8778-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/POWERTECH55446.2023.10202924

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.

feasible power flow has multiple real solutions for the state variables, which become only two when the system is close to the feasibility boundary. On the contrary, an infeasible power flow presents complex solutions for the state variables, which are not admissible from a technical point of view [2], [3]. At the feasibility boundary, a saddle-node bifurcation occurs and thus the Jacobian of the system becomes singular.

The first power flow method, now known as Gauss-Seidel (GS), was developed during the fifties [4]. However, the most commonly used method for power flow calculation is the Newton-Rhapson (NR) method, based on the successive resolution of the linear approximation of the power flow equations [5]. The NR method exhibits more robustness and better convergence properties than the GS method. The NR method became very efficient due to the development of highly efficient sparse matrix techniques for direct solution of linear equations systems [6]. NR and its variations became the most widely used method to tackle power flow problems. Most recent proposals to address the power flow problem are the sequential conic programming [7], [8] or the holomorphic embedding [9], [10].

The divergence of conventional root-finding methods such as NR, can be associated either with an infeasible power flow, or with a feasible one but with the starting point out of the convergence radius of the solution method. To overcome the non-convergence of the standard NR method, several works have been carried out. In [11], the authors incorporate a deceleration factor to the state variables updating vector, controlled by the mismatches evolution during the convergence process. Later, this step size control has been optimized, not only improving the convergence, but also using the deceleration factor as a power flow solvability indicator. Braz, *et al*., presents in [12] a critical evaluation of three methods of step size optimization based polar coordinates power flow methods developed in [13]-[15], whereas step size optimization based rectangular coordinates power flow are developed in [16], [17].

However, these algorithms only identify when a case is unsolvable, but do not provide any insolvability measure. Overbye presents in [18] an algorithm that not only identifies unsolvable cases, but also provides a measure of how unsolvable the case is. This insolvability measure is based on the minimum Euclidean distance between the operating point, and the closest solvable operating point in the parameter space. In [19], [20] a general methodology to compute and improve power systems security margins is presented. One of the applications considered is a power flow algorithm based on a combination of continuation and optimization techniques. In case of a feasible power flow, the root of the system is obtained. In case of an unfeasible power flow, a margin to feasibility boundary is computed and improved using a sequential linear programming algorithm.

III. SECOND-ORDER DIFFERENTATION APPROACH TO THE POWER FLOW PROBLEM

A. Notation

Italic notation x_i represents scalars. Bold notation $\mathbf{x}$ represents matrices or vectors, e.g. $\mathbf{x} = [x_i]$.

i, j, k, m	bus indices
x_i	state variable
M_m	power mismatch equation
S_m	injected power into network
Se_m	specified power
x_{0i}	starting point for NR iteration
x_{SOLi}	solution of power flow problem
$\frac{\partial S_m}{\partial x_i}$	first-order derivative of power injections with respect to state variables (Jacobian)
$\frac{\partial x_i}{\partial S_m}$	inverse of the Jacobian matrix, i.e. first-order sensitivity of state variables with respect to power injection
λ	convergence factor
S_{0m}	injected power into network for the starting point $\mathbf{x}_0$
M_m^*	power mismatch equation for the starting point $\mathbf{x}_0$
g_m	homotopy built to connect the starting point with the power flow solution
$\frac{d^N x_i}{d\lambda^N}$	N^{th}-order derivative of the state variable with respect to the convergence factor
$\frac{\partial^2 S_m}{\partial x_j \partial x_k}$	second-order derivative of power injections with respect to state variables

B. Newton-Raphson applied to power flow problem

Power flow problem consists in balancing the power specified (Se_m), i.e. generation minus demand, and the power injected into network (S_m) at each of the buses. Balancing power equations means doing zero the corresponding power mismatches (M_m).

$$M_m(\mathbf{x}) = S_m(\mathbf{x}) - Se_m = 0 \tag{2}$$

In (2), power specified is fixed, but power injected into network is function of the system state variables $\mathbf{x}$, so is the corresponding mismatch. Let $\mathbf{x_{SOL}}$ be the solution of (2) and $\mathbf{x_0}$ the starting point for an iteration of the NR algorithm. Mismatches computed at $\mathbf{x_0}$ will be:

$$M_{0m} = S_m(\mathbf{x_0}) - Se_m \tag{3}$$

The further away is $\mathbf{x_0}$ from $\mathbf{x_{SOL}}$, the larger are the initial mismatches M_{0m}. To approach the solution, NR substitutes the set of nonlinear equations by their linear approximations.

$$0 = M_m(\mathbf{x}) \approx M_{0m} + \sum_i \frac{\partial S_m}{\partial x_i} \cdot dx_i \tag{4}$$

In (4), $\partial S_m/\partial x_i$ represents the first-order derivative of power mismatches with respect to state variables, which is equal to the derivative of power injections with respect to state variables $\partial S_m/\partial x_i$, called the Jacobian matrix. The NR iteration is run by substituting the differential dx_i by the update of the variable Δx_i and solving (4).

$$\Delta x_i = -\sum_m \frac{\partial x_i}{\partial S_m} \cdot M_{0m} \tag{5}$$

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.

In (5), $\partial x_i/\partial S_m$ represents the coefficients of the inverse of the Jacobian.

$$\sum_m \frac{\partial x_i}{\partial S_m}\cdot\frac{\partial S_m}{\partial x_j}=\begin{cases}1 & if \quad i=j\\ 0 & if \quad i\neq j\end{cases} \tag{6}$$

Once Δx_i is computed, state variables are updated, and mismatches are checked again. The process is repeated until a sufficiently precise value is reached, i.e. until mismatches are below a stated tolerance. Figure 1 depicts a flowchart of the NR power flow algorithm.

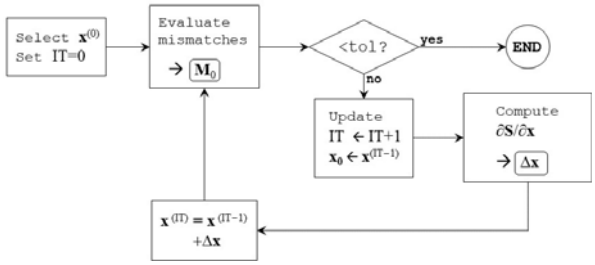

Figure 1. Flow chart of the NR power flow algorithm

C. Homotopy reformulation of the power flow problem

The aim is to build a new system where $\mathbf{x_0}$ and $\mathbf{x_{SOL}}$ are both roots of it. Let S_{0m} be the power injected computed at the starting point.

$$S_{0m}=S_m(\mathbf{x_0}) \tag{7}$$

Using S_{0m}, new mismatches M_m^* are defined as follows:

$$M_m^*(\mathbf{x})=S_m(\mathbf{x})-S_{0m}=0 \tag{8}$$

New mismatches M_m^* represent a new power flow problem that has the starting point as solution, i.e. $M_m^*(\mathbf{x}_0)=0$. The homotopy g_m is defined as.

$$g_m(\mathbf{x},\lambda)=M_m(\mathbf{x})\cdot\lambda+M_m^*(\mathbf{x})\cdot(\lambda-1)=0 \tag{9}$$

Homotopy (9) is function of the state variables and the convergence factor λ. If λ=0, the starting point $\mathbf{x_0}$ is root of the homotopy, and if λ=1, the power flow solution $\mathbf{x_{SOL}}$ is root of the homotopy.

$$g_m(\mathbf{x_0},0)=0 \quad ; \quad g_m(\mathbf{x_{SOL}},1)=0 \tag{10}$$

By using (2) and (8), the homotopy is rewritten as follows:

$$g_m(\mathbf{x},\lambda)=S_m(\mathbf{x})-\left[S_{0m}+\lambda\cdot\left(Se_m-S_{0m}\right)\right]=0 \tag{11}$$

Due to the degree of freedom introduced by the convergence factor, the homotopy defines a set of manifolds for the state variables $x_i=x_i(\lambda)$ that evolves from the starting point $x_i(0)=x_{0i}$ to the power flow solution $x_i(1)=x_{SOLi}$. Figure 2 provides two different perspectives of the manifolds associated to the homotopy. In Figure 2 a), the manifold $\mathbf{x}(\lambda)$ is represented as a single curve in the state variables space that evolves from the starting point $\mathbf{x_0}$ to the power flow solution $\mathbf{x_{SOL}}$. This manifold can be represented as a set of trajectories in the convergence factor-state variables plane, as they are depicted in Figure 2 b). Consequently, the different trajectories con be expanded using Taylor's series from the starting point (λ=0).

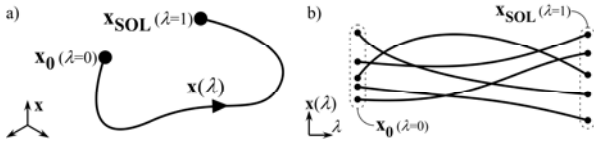

Figure 2. Manifolds $\mathbf{x}(\lambda)$ associated to the homotopy represented a) in the state variables space, and b) in the convergence factor-state variables plane

$$x_i(\lambda)=x_{0i}+\sum_N\frac{1}{N!}\cdot\frac{d^N x_i}{d\lambda^N}\cdot\lambda^N \tag{12}$$

The difficulty with this method is to compute the successive derivatives $d^Nx_i/d^N\lambda$. Since state variables are defined as roots of the homotopy, their derivatives are inaccessible. To obtain them, successive differentiations of the homotopy are required. In the next two subsections the first and second-order derivatives are cleared out and used to build the second-order update vector. In addition, first-order derivative will be demonstrated equal to the updating vector from NR.

D. First-order differentiation of the homotopy model

Starting from $(\mathbf{x_0},0)$, a differential variation $d\lambda$ in the convergence factor is considered that undergoes a variation in the state variables. Under this assumption, homotopy will tend to vary differentially too. However, to keep following the manifold $\mathbf{x}(\lambda)$, the homotopy must continue being zero, thus differential variation dg_m vanishes.

$$dg_m=0=\sum_i\frac{\partial g_m}{\partial x_i}\cdot dx_i+\frac{\partial g_m}{\partial\lambda}\cdot d\lambda \tag{13}$$

Considering (11), it can be noticed that $\partial g_k/\partial x_i=\partial S_k/\partial x_i$, i.e. the Jacobian matrix. On the other hand, the derivative of g_m with respect to the convergence factor is equal to the initial mismatches M_{0m}.

$$\frac{\partial g_m}{\partial\lambda}=-\left(Se_m-S_{0m}\right)=M_{0m} \tag{14}$$

With these considerations, the first-order differential variation of state variables is isolated from (13).

$$\sum_i\frac{\partial S_m}{\partial x_i}\cdot dx_i=-M_{0m}\cdot d\lambda \tag{15}$$

To clear out the first-order derivative of state variables with respect to the convergence factor, (15) is left-multiplied by the inverse of the Jacobian ($\partial x_i/\partial S_m$) and divided by $d\lambda$.

$$\boxed{\frac{dx_i}{d\lambda}=-\sum_m\frac{\partial x_i}{\partial S_m}\cdot M_{0m}} \tag{16}$$

E. Second-order differentiation of the homotopy model

Following the previous assumption, homotopy is differentiated and set equal to zero again.

$$\begin{aligned}d^2g_m=0=\ &\sum_i\frac{\partial g_m}{\partial x_i}\cdot d^2x_i+\sum_j\sum_k\frac{\partial^2 g_m}{\partial x_j\partial x_k}\cdot dx_j\cdot dx_k\\ &+\sum_j\frac{\partial^2 g_m}{\partial x_j\partial\lambda}\cdot dx_j\cdot d\lambda+\frac{\partial^2 g_m}{\partial\lambda^2}\cdot d\lambda^2\end{aligned} \tag{17}$$

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.

In (17), second-order derivatives $\partial^2 g_m/\partial x_j \cdot \partial\lambda$ and $\partial^2 g_m/\partial\lambda^2$ vanish because both of them are equal to zero. In addition, noticed that $\partial^2 g_k/\partial x_j \cdot \partial x_k = \partial^2 S_k/\partial x_j \cdot \partial x_k$. With these simplifications, the second-order differential variation of state variables is isolated from (17).

$$\sum_i \frac{\partial S_m}{\partial x_i} \cdot d^2 x_i = -\sum_j \sum_k \frac{\partial^2 S_m}{\partial x_j \partial x_k} \cdot dx_j \cdot dx_k \qquad (18)$$

To clear out the second-order derivative of state variables with respect to the convergence factor, (18) is left-multiplied by the inverse of the Jacobian ($\partial x_i/\partial S_m$) and divided by $d\lambda^2$. First-order derivatives $dx_j/d\lambda$ and $dx_k/d\lambda$ are computed using (16)

$$\boxed{\frac{d^2 x_i}{d\lambda^2} = -\sum_m \sum_j \sum_k \frac{\partial x_i}{\partial S_m} \cdot \frac{\partial^2 S_m}{\partial x_j \partial x_k} \cdot \frac{dx_j}{d\lambda} \cdot \frac{dx_k}{d\lambda}} \qquad (19)$$

First-order derivatives (16) of the manifolds $x_i = x_i(\lambda)$ have been proved very useful in several applications, thus second-order derivatives (19) can provide additional information that corrects linear approximations. Continuation techniques applied to different problems, such as TTC computation [21] or voltage stability analysis [22], use linear predictor vectors that forecast the distance to the boundary, thus second-order may provide more accuracy to the predictors. However, in this paper derivatives (16) and (19) will be used to formulate the second-order power flow algorithm presented in the next section.

IV. Homotopic second-order power flow algorithm

The power flow algorithm presented in this section has been designed as an iterative process. At each iteration, an updating vector is computed using derivatives (16) and (19), and applied to the state variables. The power flow solution $\mathbf{x_{SOL}}$ can be computed using (12) and doing λ=1:

$$x_{SOLi} = x_{0i} + \sum_N \Delta x_i^{(N)} \qquad (20)$$

where

$$\Delta x_i^{(N)} = \frac{1}{N!} \cdot \frac{d^N x_i}{d\lambda^N} \qquad (21)$$

If we only consider first-order approximations, the first-order updating vector for state variables is obtained by truncating (20) to degree 1, i.e. $\Delta x_i\big|_{1stO} = \Delta x_i^{(1)}$, which is equal to the NR updating vector (5). Therefore, first-order version of the algorithm proposed in this paper corresponds with standard NR algorithm. However, second-order derivatives provide additional information to build a second-order updating vector. Truncating (20) to degree 2, the second-order updating vector would be $\Delta x_i\big|_{2ndO} = \Delta x_i^{(1)} + \Delta x_i^{(2)}$ for state variables is obtained, where:

$$\Delta x_i^{(1)} = \frac{dx_i}{d\lambda} \quad ; \quad \Delta x_i^{(2)} = \frac{1}{2}\frac{d^2 x_i}{d\lambda^2} \qquad (22)$$

Figure 3 provides a graphical description of how the different updating vectors work at each of the iterations.

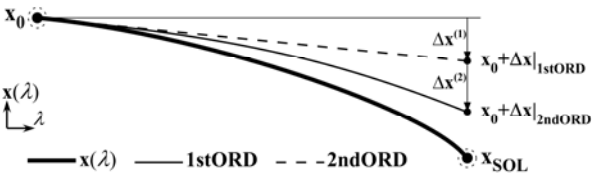

Figure 3. Manifold $\mathbf{x}(\lambda)$ with first and second-order approximations

In Figure 3, manifold $\mathbf{x}(\lambda)$ is represented from the starting point $\mathbf{x}(0) = \mathbf{x_0}$ to the power flow solution $\mathbf{x}(1) = \mathbf{x_{SOL}}$. In addition, the first and second-order approaches are included. In Figure 3, starting point is updated by using first and second-order updating vectors. Additional information from second-order derivatives provides a more accurate updating vector. Figure 4 depicts a flowchart of the algorithm presented in this paper.

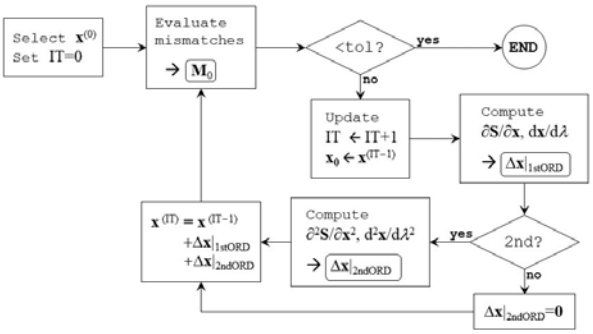

Figure 4. Flow chart of the second-order power flow algorithm

The algorithm depicted in Figure 4, if "second-order" flag is off, corresponds with the standard NR, since the first-order updating vector is equal to the NR updating vector (see Figure 1). If the "second-order" flag is activated, the second-order derivatives and the second-order updating vector are computed. This correction takes the next point computed closer to the solution. Since the second-order updating vector gets the new point closer to the solution, the goal of the proposed algorithm is to speed up the convergence process, i.e. to reduce the total number of iterations.

V. Results

To illustrate the performance of the second-order power flow algorithm presented in this paper it will be applied to the IEEE 39 test buses standard network [23], [24], and compared against the NR algorithm. The network includes 39 buses, 34 lines, 12 transformers, 10 generating units and 31 loads. Buses are distributed among three different areas, where area 3 behaves as importing area. TABLE I contains the power balance in MW for the base case, showing the total balance and breaking it down by areas, whereas Figure 5 shows the single-line diagram of the IEEE 39 test buses network. As TABLE I shows, the negative generation-demand balance in the area 3 leads to import 439 MW from the other areas, 106 MW

TABLE I. Active power balance for the base case [MW]

area	GEN	DEM	LOSS	EXP
1-AREA01	1620.0	1455.5	18.3	146.2
2-AREA02	3571.3	3068.0	22.5	480.8
3-AREA03	1000.0	1626.0	1.0	-627.0
TOTAL	6191.3	6149.5	41.8	

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.

from area 1 and 333 MW from area 2.

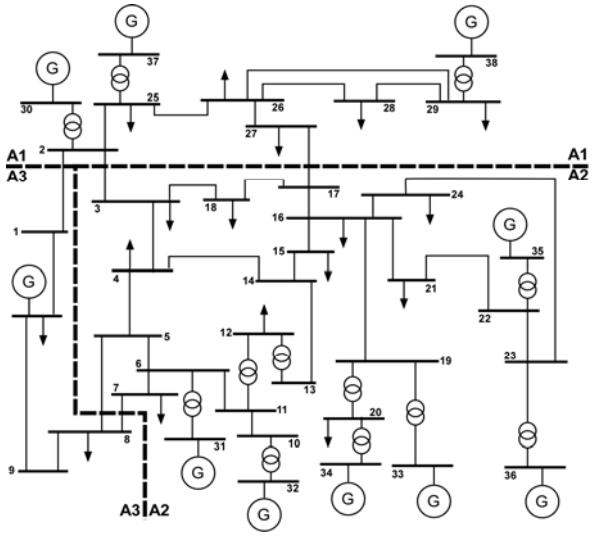

Figure 5. IEEE 39 test buses network single-line diagram

To test the power flow algorithms properly, different load levels must be considered. Active power load and generation have been proportionally increased. Reactive power demand has been increased too by keeping power factors constant. Considering that the voltages collapse point is reached with a demand increase of 142.5%, four different scenarios of demand growth have been considered: 0% (base case), 50%, 100% and 140%. TABLE II and TABLE III show the active and reactive power balance for each of the demand growth scenarios considered respectively.

TABLE II. ACTIVE POWER BALANCE [MW]

scen	GEN	DEM	LOSS
0%	6191.3	6149.5	41.8
50%	9323.4	9224.3	99.1
100%	12498.9	12299.0	199.9
140%	15133.0	14758.8	374.2

TABLE III. REACTIVE POWER BALANCE [MVAR]

scen	GEN	DEM	LOSS
0%	837.3	1408.9	-571.6
50%	2956.1	2113.4	842.8
100%	6193.2	2817.8	3375.4
140%	11448.8	3381.4	8067.4

The power flow algorithms have been applied to the four scenarios considered in TABLE II and TABLE III. The tolerance for the mismatches has been set to 10^{-12}, and the Flat Start has been considered as starting point. To illustrate the performance of the second-order algorithm presented in this paper, Figure 6 depicts the number of iterations run by both power flow algorithms. As Figure 6 illustrates, second-order power flow algorithm reduces the number of iterations by 2. In the base case (0%), the classic NR takes 5 iterations, while the second-order power flow only needs 3. In the extreme case (+140%), both algorithms take more iterations, 8 and 5 respectively.

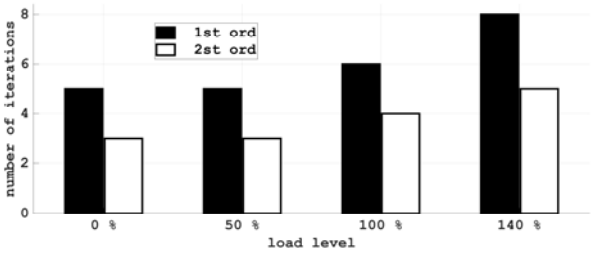

Figure 6. Number of iterations run by the 1st-order and the 2nd-order power flow algorithms

Figure 7 shows the evolution of the largest mismatch during convergence in logarithmic scale. As shown in Figure 7, the mismatches reduction during standard NR convergence is slower than the observed for the second-order power flow algorithm for all the scenarios considered. The behavior of the second-order convergence seems to be more robust and stable than the NR.

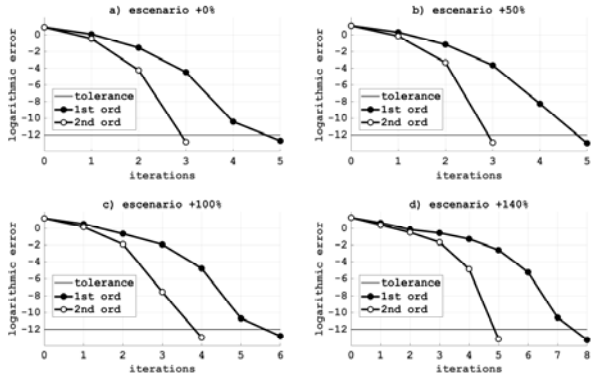

Figure 7. Convergence evolution of the 1st and 2nd order power flow algorithms

In terms of computational time effort, it is clear that a 2nd-order iteration takes more time than a 1st-order iteration. However, the computational time effort in a 1st-order iteration is mostly dedicated to solve the corresponding linear equations system to obtain the updating vector for the state variables. This process implies the LU factorization of the Jacobian matrix and the forward-backward substitution. However, while forward-backward substitution computational cost is $\mathcal{O}(n^2)$, the cost of the LU factorization is $\mathcal{O}(n^3)$, thus the total cost of a 1st-order iteration is $\mathcal{O}(n^3)+\mathcal{O}(n^2)\approx\mathcal{O}(n^3)$. Concerning the cost of a 2nd-order iteration, it implies to solve a linear equations system twice, but using the same Jacobian matrix. Therefore, the computational cost of a 2nd-order iteration would be would be $\mathcal{O}(n^3)+2\cdot\mathcal{O}(n^2)\approx\mathcal{O}(n^3)$, i.e. equivalent to the cost of a 1st-order iteration. For example, considering Figure 7a, the 1st-order algorithm requires 5 iterations, i.e. 5 LU factorizations and 5 forward-backward substitutions, while the 2nd-order algorithm requires only 3 iterations, i.e. 3 LU factorizations and 6 forward-backward substitutions. Using the 2nd-order algorithm, an extra forward-backward substitution must be run, but it requires two less LU factorizations, with the corresponding time saving.

To illustrate the difference between using first or second-order approaches to update the variables, Figure 8 represents the actual trajectories of the voltage at bus 24, and corresponding first and second-order approaches from first

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.

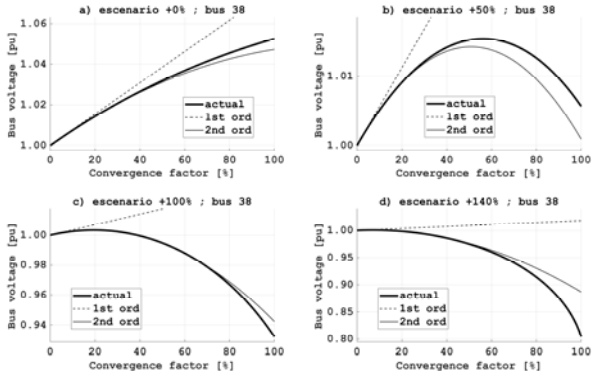

Figure 8. Trajectory of bus 24 voltage and its first and second-order approximations

iteration. To provide more detailed information of results in Figure 8, TABLE IV present the updating results after first iteration of the voltage at bus 24. Columns are, from left to right, the scenario analyzed, difference between solution and starting point (1.0 pu for bus 24 voltage), first and second-order updating components, and first and second-order updating vectors.

TABLE IV. BUS 24 VOLTAGE UPDATING RESULTS AFTER FIRST ITERATION [PU X 100]

scen	$V_{SOL} - V_0$	$\Delta V^{(1)}$	$\Delta V^{(2)}$	$\Delta V\|_{1stO}$	$\Delta V\|_{2ndO}$
0%	5.26	7.71	-2.98	7.71	4.73
50%	0.57	5.61	-5.51	5.61	0.09
100%	-6.79	3.50	-9.27	3.50	-5.76
140%	-19.56	1.82	-13.15	1.82	-11.33

Figure 8 and TABLE IV shows how the difference between the solution of the bus 24 voltage and its starting point (1.0 pu) changes the sign from positive to negative as long as the system load growths and the network evolves from generating to consuming reactive power (see). In this case, first-order derivatives present larger deviations. Due to the negative second-order derivatives, second-order updating is corrected and provide a new point closer to the solution.

VI. CONCLUSIONS

In this paper, a novel second-order formulation of the NR power flow is presented. The proposed second-order approach is based on the reformulation of the power flow equations as a homotopy that evolves from the starting point to the power flow solution. Thus, the homotopy defines a set of trajectories for the state variables. First-order derivative of these trajectories defines an updating vector equal to the standard NR. Second-order derivatives allow to compute a second-order updating vector that presents more accuracy and robustness than only using the first-order approach.

The performance of the second-order power flow algorithm is tested against the standard NR. Considering the IEEE 39 buses test network, four different load growth scenarios have been defined from the base case to close to the voltages collapse point. For all of them, second-order power flow almost halves the number of iterations required for convergence. Second-order approach provides more accurate updating vectors, with resultant faster and more robust convergence processes.

REFERENCES

[1] W. D. Stevenson, Elements of Power System Analysis. McGraw-Hill, 1982.

[2] H. Mori and F. Iizuka, "An efficient method for calculating power flow solutions and the closest bifurcation point using mathematical programming," ISCAS '98. Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. no. 98CH36187), vol. 3, pp. 570-573 vol.3, 1998.

[3] K. Iba et al, "A method for finding a pair of multiple load flow solutions in bulk power systems," IEEE Transactions on Power Systems, vol. 5, (2), pp. 582-591, 1990.

[4] B. Stott, "Review of load-flow calculation methods," Proceedings of the IEEE, vol. 62, (7), pp. 916-929, 1974.

[5] W. F. Tinney and C. E. Hart, "Power Flow Solution by Newton's Method," IEEE Transactions on Power Apparatus and Systems, vol. PAS-86, (11), pp. 1449-1460, 1967.

[6] W. F. Tinney and J. W. Walker, "Direct solutions of sparse network equations by optimally ordered triangular factorization," Proceedings of the IEEE, vol. 55, (11), pp. 1801-1809, 1967.

[7] R. A. Jabr, "A Conic Quadratic Format for the Load Flow Equations of Meshed Networks," IEEE Transactions on Power Systems, vol. 22, (4), pp. 2285-2286, 2007.

[8] R. A. Jabr, "Radial distribution load flow using conic programming," IEEE Transactions on Power Systems, vol. 21, (3), pp. 1458-1459, 2006.

[9] S. Rao et al, "The Holomorphic Embedding Method Applied to the Power-Flow Problem," IEEE Transactions on Power Systems, vol. 31, (5), pp. 3816-3828, 2016.

[10] A. Trias, "The Holomorphic Embedding Load Flow method," 2012 IEEE Power and Energy Society General Meeting, pp. 1-8, 2012.

[11] K. A. Clements et al, "Transmission system reliability methods - 1. Mathematical models, computing methods, and results." Electr Power Res Rep EPRI EL, vol. 1, 1982.

[12] L. M. C. Braz, C. A. Castro and C. A. F. Murati, "A critical evaluation of step size optimization based load flow methods," IEEE Transactions on Power Systems, vol. 15, (1), pp. 202-207, 2000.

[13] C. A. Castro and L. M. C. Braz, "A New Approach to the Polar Newton Power Flow using Step Size Optimization," Proceedings of the 29th Annual North American Power Symposium, 1997..

[14] J. M. Scudder, "Step Optimization in a Polar Newton Power Flow." University of Wisconsin, Madison, U.S.A., 1981.

[15] M. Dehnel and H. W. Dommel, "A method for identifying weak nodes in nonconvergent load flows," IEEE Transactions on Power Systems, vol. 4, (2), pp. 801-807, 1989.

[16] S. Iwamoto and Y. Tamura, "A Load Flow Calculation Method for Ill-Conditioned Power Systems," IEEE Transactions on Power Apparatus and Systems, vol. PAS-100, (4), pp. 1736-1743, 1981.

[17] P. R. Bijwe and S. M. Kelapure, "Nondivergent fast power flow methods," IEEE Transactions on Power Systems, vol. 18, (2), pp. 633-638, 2003.

[18] T. J. Overbye, "A power flow measure for unsolvable cases," IEEE Transactions on Power Systems, vol. 9, (3), pp. 1359-1365, 1994.

[19] F. M. Echavarren et al, "Formulation, computation and improvement of steady state security margins in power systems. Part I: Theoretical framework," International Journal of Electrical Power & Energy Systems, vol. 33, (2), pp. 340-346, FEB 2011, 2011.

[20] F. M. Echavarren et al, "Formulation, computation and improvement of steady state security margins in power systems. Part II: Results," International Journal of Electrical Power & Energy Systems, vol. 33, (2), pp. 347-358, 2, 2011.

[21] I. Dobson et al, "Electric power transfer capability: concepts, applications, sensitivity and uncertainty," 01, 2001.

[22] V. Ajjarapu and C. Christy, "The continuation power flow: a tool for steady state voltage stability analysis," [Proceedings] Conference Papers 1991 Power Industry Computer Application Conference, pp. 304-311, 1991.

[23] IEEE 39-Bus System, https://icseg.iti.illinois.edu/ieee-39-bus-system/.

[24] T. Athay, R. Podmore and S. Virmani, "A Practical Method for the Direct Analysis of Transient Stability," IEEE Transactions on Power Apparatus and Systems, vol. PAS-98, (2), pp. 573-584, 1979.

Authorized licensed use limited to: UNIVERSIDAD PONTIFICIA DE COMILLAS. Downloaded on September 11,2023 at 09:53:57 UTC from IEEE Xplore. Restrictions apply.