Performance analysis of Newton-Raphson Power flow computations based on Power and Current Mismatches

L Hakim[1], Khairudin[2], and Z Huda[3*]

[1,2,3]*Deparment of Electrical Engineering, Universitas Lampung Jalan Prof. Soemantri Brojonegoro No. 1, Bandar Lampung 35145, Indonesia*

[*]*Email: zulmiftah.huda@eng.unila.ac.id*

Article Information:

Received:
12 August 2022

Received in revised form:
15 November 2022

Accepted:
19 December 2022

Volume 4, Issue 2 , December 2022
pp. 67 – 73

http://dx.doi.org/10.23960/jesr.v4i2.117

Abstract

This work studies the performance of single-phase power flow computations implemented using vector-based Python scripting language. Two approaches are considered, namely the current mismatch and power mismatch based on Newton-Raphson method. Both approaches are developed using rectangular coordinates and tested through a variety of IEEE test systems modelling high voltage transmission networks. The computational burden and accuracy of both implementations are duly discussed. This work will be later incorporated into our vector-based Non-linear Primal-Dual Interior Point Method as constraints.

Keywords: *power flow; vector form; current mismatch; power mismatch; Newton-Raphson.*

I. INTRODUCTION

Vector-based programming consists in the definition of highly efficient implementation of element-wise vector calculations. This approach has been made popular by the scientific programming language Matlab, but is common to many other languages. Noteworthy examples are Fortran 95 and newer Fortran versions, as well as modern scripting languages, such as Python and Julia.

Since the power flow analysis is based on vectorized equations, its implementation through vector-oriented programming languages appears as a natural and convenient choice. As a matter of fact, several software tools for power system analysis have been developed in the last decade. Alvarado in [1] developed a vector-based version of power flow in Matlab. Other well-known Matlab-based implementations of power flow analysis are described in [2, 3]. Vector-based implementations are also possible in other scripting languages. For example, a high-performance power system analysis tool implemented in Python is described in [4]. This paper focuses on the development of a simple yet robust and efficient, power flow analysis software tool that solves the power flow problems fully based on vector-based programming.

Most power flow analysis tools rely on minimizing power mismatches at every iteration [5, 6]. Others minimize current mismatches [7-9]. The substantial equivalency, in terms of computational burden, of these two formulations when utilized for time domain analysis is given in [10]. However, when dealing with the power flow problem, some differences can be observed. The power-injection model, for example, is known to work well for high-voltage transmission systems, where the ratio R/X is low. On the other hand, the current-injection approach, when it converges, may require fewer iterations than the power-injection approach [11]. Moreover, the current approach tends to work better for distribution systems, and might show numerical issues when imposing PV-bus constraints. PV-bus constraints, in fact, can lead to numerical stability issues of the current mismatch-based formulation. The work in [12] suggests an improved representation of PV-buses by combining a power mismatch-based PV-bus representation into current mismatch equations for PQ-bus. In [12], the vector-based programming paradigm was not adopted and instead of utilizing full rectangular coordinate, the work combined rectangular for current mismatch equations for PQ-bus and for PV-bus representation the polar coordinate was implemented. Other attempts in dealing with PV-bus model for current mismatch method are reported in [13, 14].

In this work, we focus on the performance, in terms of convergence rate, of two vector-based implementations using power mismatches as well as current mismatches. The two approaches are compared using standard IEEE Test. The proposed tool utilizes the Newton-Raphson method because it is a model and topology independent technique. We also use a rectangular form for voltages and currents. All vector-based operations are implemented in Scipy [15] and Numpy [16] libraries of the Python programming language.

II. Materials and Methods

A. Full Rectangular Power Flow

In this work, two power flow approaches based on current mismatches and power mismatches are developed and compared. All complex variables are expressed in rectangular form.

B. Current Mismatch Method

Current mismatch is computed by subtracting calculated complex current ($I_{inj} = Y \cdot V$) from scheduled or pre-specified current injection.

$$\Delta I = I_{sch} - I_{inj} \tag{1}$$

Rearranging (1) and having $\Delta I = 0, I_{inj} - I_{sch} = 0$

$$I_{inj}{}^{(0)} + \left(\frac{\partial I_{inj}}{\partial e}\right)^{(0)} \Delta e^{(0)} + \left(\frac{\partial I_{inj}}{\partial f}\right)^{(0)} \Delta f^{(0)} - I_{sch}{}^{(0)} - \left(\frac{\partial I_{sch}}{\partial e}\right)^{(0)} \Delta e^{(0)} - \left(\frac{\partial I_{sch}}{\partial f}\right)^{(0)} \Delta f^{(0)} = 0 \tag{2}$$

$$\left(\frac{\partial I_{inj}}{\partial e}\right)^{(0)} \Delta e^{(0)} + \left(\frac{\partial I_{inj}}{\partial f}\right)^{(0)} \Delta f^{(0)} - \left(\frac{\partial I_{sch}}{\partial e}\right)^{(0)} \Delta e^{(0)} - \left(\frac{\partial I_{sch}}{\partial f}\right)^{(0)} \Delta f^{(0)} = I_{sch}{}^{(0)} - I_{inj}{}^{(0)} \tag{3}$$

Knowing that $I_{inj} = Y \cdot V$, $I_{sch} = {S_{sch}}^{*}/_{V^*}$, and $V = e + j\,f$, equation (3) can be derived below:

$$\left(Y\frac{\partial V}{\partial e} + V\frac{\partial Y}{\partial e}\right)\Delta e^{(0)} + \left(Y\frac{\partial V}{\partial f} + V\frac{\partial Y}{\partial f}\right)\Delta f^{(0)} - \frac{\left(\frac{\partial {S_{sch}}^*}{\partial e}V^* - \frac{\partial V^*}{\partial e}{S_{sch}}^*\right)}{(V^*)^2}\Delta e^{(0)} - \frac{\left(\frac{\partial {S_{sch}}^*}{\partial f}V^* - \frac{\partial V^*}{\partial f}{S_{sch}}^*\right)}{(V^*)^2}\Delta f^{(0)} = I_{sch}{}^{(0)} - I_{inj}{}^{(0)} \tag{4}$$

$$\left(Y + \frac{{S_{sch}}^*}{(V^*)^2}\right)\Delta e^{(0)} + j\left(Y - \frac{{S_{sch}}^*}{(V^*)^2}\right)\Delta f^{(0)} = I_{sch}{}^{(0)} - I_{inj}{}^{(0)} \tag{5}$$

In vector form, these partial derivatives can be written as:

$$\frac{\partial \Delta I}{\partial e} = \mathbf{Y} + \frac{diag(\mathbf{S_{sch}^*})}{diag((\mathbf{V}^*)^2)} \tag{6}$$

$$\frac{\partial \Delta I}{\partial f} = j\left(\mathbf{Y} - \frac{diag(\mathbf{S_{sch}^*})}{diag((\mathbf{V}^*)^2)}\right) \tag{7}$$

Equations (6) and (7) are then arranged to form Newton's correction equations to be solved iteratively by stacking the real parts and imaginary parts of the current mismatch Jacobian matrices.

$$\begin{bmatrix} Real\left\{\frac{\partial \Delta I}{\partial e}\right\} & Real\left\{\frac{\partial \Delta I}{\partial f}\right\} \\ Imag\left\{\frac{\partial \Delta I}{\partial e}\right\} & Imag\left\{\frac{\partial \Delta I}{\partial f}\right\} \end{bmatrix} \cdot \begin{bmatrix} \Delta e \\ \Delta f \end{bmatrix} = \begin{bmatrix} Real\{\Delta I\} \\ Imag\{\Delta I\} \end{bmatrix} \tag{8}$$

C. Power Mismatch Method

Power mismatch is computed by subtracting calculated complex power ($S = V \cdot I^*$) from scheduled or pre-specified power.

$$\Delta S = S_{sch} - S_{inj} \tag{9}$$

$$S_{inj} = S_{sch} \tag{10}$$

$$S_{inj}{}^{(0)} + \left(\frac{\partial S_{inj}}{\partial e}\right)^{(0)} \Delta e^{(0)} + \left(\frac{\partial S_{inj}}{\partial f}\right)^{(0)} \Delta f^{(0)} = S_{sch} \tag{11}$$

$$\left(\frac{\partial S_{inj}}{\partial e}\right)^{(0)} \Delta e^{(0)} + \left(\frac{\partial S_{inj}}{\partial f}\right)^{(0)} \Delta f^{(0)} = S_{sch} - S_{inj}{}^{(0)} \tag{12}$$

$$\left(V\left(\frac{\partial I^*}{\partial e}\right)^{(0)} + I^*\left(\frac{\partial V}{\partial e}\right)^{(0)}\right)\Delta e^{(0)} + \left(V\left(\frac{\partial I^*}{\partial f}\right)^{(0)} + I^*\left(\frac{\partial V}{\partial f}\right)^{(0)}\right)\Delta f^{(0)} = S_{sch} - S_{inj}{}^{(0)} \tag{13}$$

Partial derivatives of the power mismatch equations with respect to variables e and f can be written in vector forms as follows:

$$\frac{\partial S_{inj}}{\partial e} = V\frac{\partial I^*}{\partial e} + I^*\frac{\partial V}{\partial e} = diag(\mathbf{V})\cdot\mathbf{Y}^* + diag(\mathbf{I}^*) \qquad (14)$$

$$\frac{\partial S_{inj}}{\partial f} = V\frac{\partial I^*}{\partial f} + I^*\frac{\partial V}{\partial f} = j\big(-diag(\mathbf{V})\cdot\mathbf{Y}^* + diag(\mathbf{I}^*)\big) \qquad (15)$$

Equations (14) and (15) are then arranged to form Newton's correction equations to be solved iteratively by stacking the real parts and imaginary parts of the power mismatch Jacobian matrices.

$$\begin{bmatrix} Real\left\{\frac{\partial \Delta S}{\partial e}\right\} & Real\left\{\frac{\partial \Delta S}{\partial f}\right\} \\ Imag\left\{\frac{\partial \Delta S}{\partial e}\right\} & Imag\left\{\frac{\partial \Delta S}{\partial f}\right\} \end{bmatrix}\cdot\begin{bmatrix}\Delta e\\ \Delta f\end{bmatrix} = \begin{bmatrix} Real\{\Delta S\} \\ Imag\{\Delta S\}\end{bmatrix} \qquad (16)$$

Both the current and power mismatches methods above follow voltage updates as in (17):

$$e^{(k+1)} = e^{(k)} + \Delta e^{(k)} \qquad (17)$$
$$f^{(k+1)} = f^{(k)} + \Delta f^{(k)}$$

D. PV-bus Model

Different for PV-bus where voltage magnitude is maintained at the same level as its pre-specified value, the imaginary row of both current and power mismatches correction equations is replaced with voltage magnitude constraint (21).

$$V^2 = e^2 + f^2 \qquad (18)$$

$$V_{is}{}^2 - V^2 = 0 \qquad (19)$$

$$V^{2(0)} + \frac{\partial V^2}{\partial e}\Delta e^{(0)} + \frac{\partial V^2}{\partial f}\Delta f^{(0)} = V_{is}{}^2 \qquad (20)$$

$$\frac{\partial V^2}{\partial e}\Delta e^{(0)} + \frac{\partial V^2}{\partial f}\Delta f^{(0)} = V_{is}{}^2 - V^{2(0)} \qquad (21)$$

$$\frac{\partial V^2}{\partial e} = 2e \qquad (22)$$

$$\frac{\partial V^2}{\partial f} = 2f \qquad (23)$$

Therefore, equation (8) is modified to equation (24) for PV-bus.

$$\begin{bmatrix} Real\left\{\frac{\partial \Delta I}{\partial e}\right\} & Real\left\{\frac{\partial \Delta I}{\partial f}\right\} \\ \frac{\partial V^2}{\partial e} & \frac{\partial V^2}{\partial f} \end{bmatrix}\cdot\begin{bmatrix}\Delta e\\ \Delta f\end{bmatrix} = \begin{bmatrix} Real\{\Delta I\} \\ V_{is}{}^2 - V^{2(0)}\end{bmatrix} \qquad (24)$$

Consequently, equation (25) substitutes equation (16).

$$\begin{bmatrix} Real\left\{\frac{\partial \Delta S}{\partial e}\right\} & Real\left\{\frac{\partial \Delta S}{\partial f}\right\} \\ \frac{\partial V^2}{\partial e} & \frac{\partial V^2}{\partial f} \end{bmatrix}\cdot\begin{bmatrix}\Delta e\\ \Delta f\end{bmatrix} = \begin{bmatrix} Real\{\Delta S\} \\ V_{is}{}^2 - V^{2(0)}\end{bmatrix} \qquad (25)$$

III. Results and Discussions

The developed power flow models both for current and power mismatches were then tested on the standard IEEE Test Systems of 14-bus, 30-bus, 57-bus, 118-bus and 300-bus as available on the University of Washington page [17].

Table 1. Performance Comparison for Power Flow Calculation in Current and Power Mismatch Methods

Test-System	Computing Time (s)		Iteration Number	
	Current Mismatch	Power Mismatch	Current Mismatch	Power Mismatch
IEEE 14-bus	0.02141	0.02135	4	4
IEEE 30-bus	0.02738	0.03071	4	(17) 4
IEEE 57-bus	0.05397	0.07081	4	5
IEEE 118-bus	0.15613	0.15818	6	5
IEEE 300-bus	0.31671	0.37524	5	6

Overall performance of both current mismatch and power mismatch methods in vector forms is indicated in Table 1. The two methods show similar performance but the current mismatch method appears to be slightly superior than power mismatch method. In terms of mismatch progression over iterations, the current mismatch method indicates faster decay than the power mismatch method. This can be observed in Figures. 1 to 5.

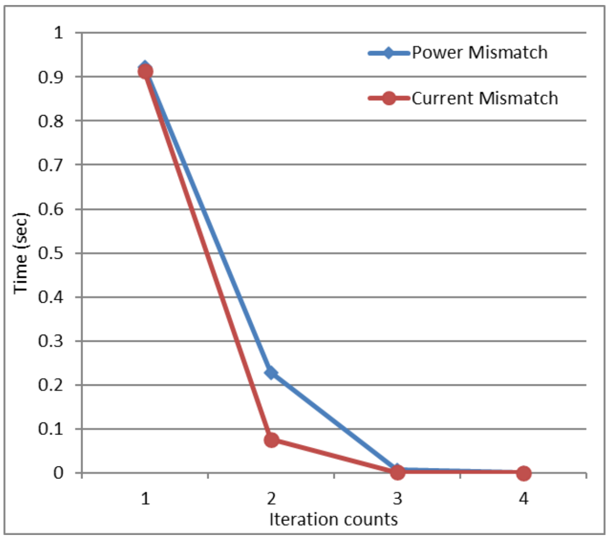

Figure 1. Mismatch progression for the IEEE 14-bus System (24)

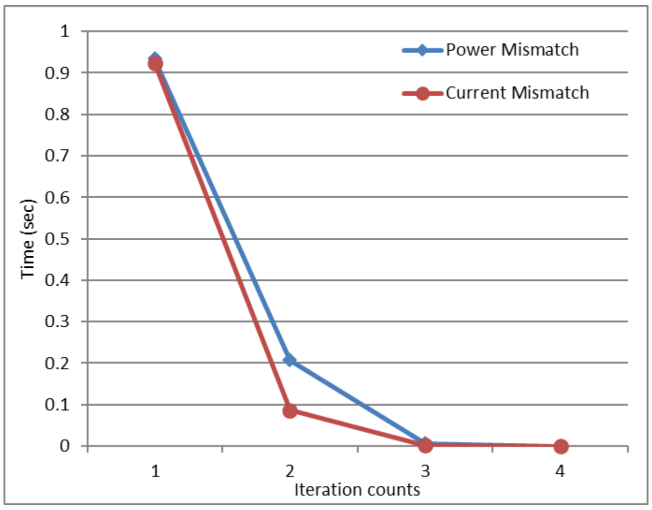

Figure 2. Mismatch progression for the IEEE 30-bus System

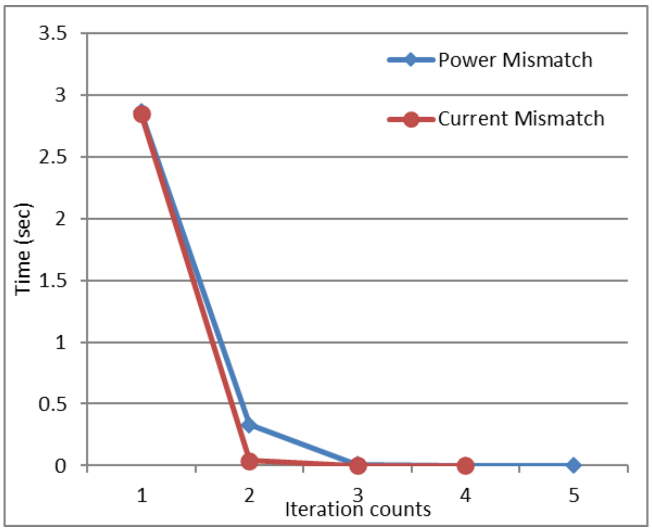

Figure 3. Mismatch progression for the IEEE 57-bus System

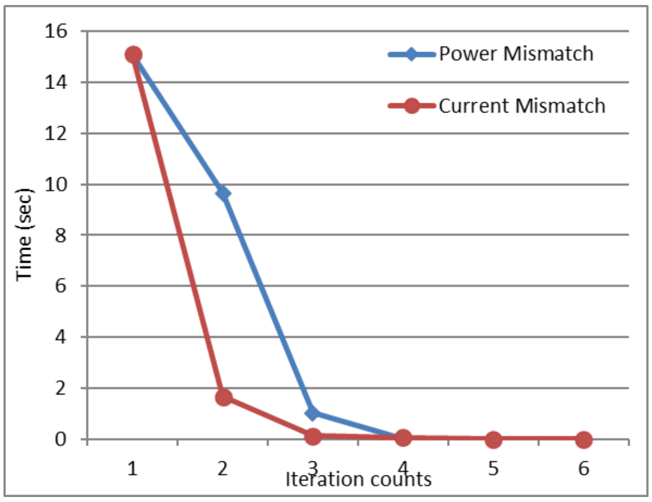

Figure 4. Mismatch progression for the IEEE 118-bus System

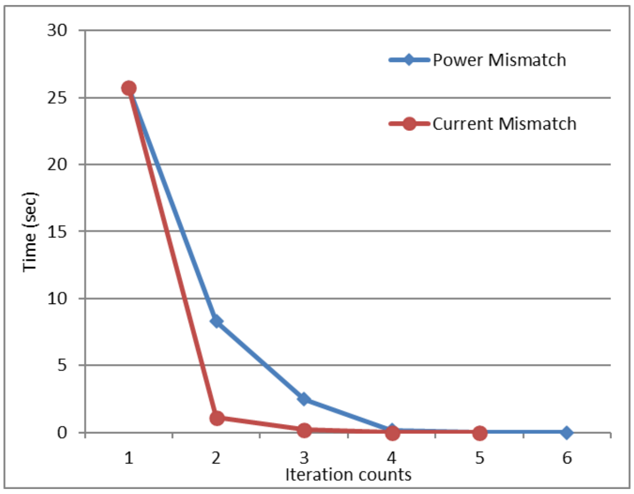

Figure 5. Mismatch progression for the IEEE 300-bus System.

Comparisons are also made for the voltages obtained from both current mismatch and power mismatch methods and the original IEEE final voltages as available in the input data are taken as the reference. Interestingly, both methods produce approximately similar results and close to the final voltage of the IEEE test system data. Figures 6 to 10 show the absolute difference in voltage resulted from the current mismatch and power mismatch methods as compared to the IEEE final voltages. The dotted-red line shows absolute difference between power mismatch method and IEEE final voltage in the input data while the solid-blue line is for the absolute difference between the current mismatch method and the IEEE final voltages.

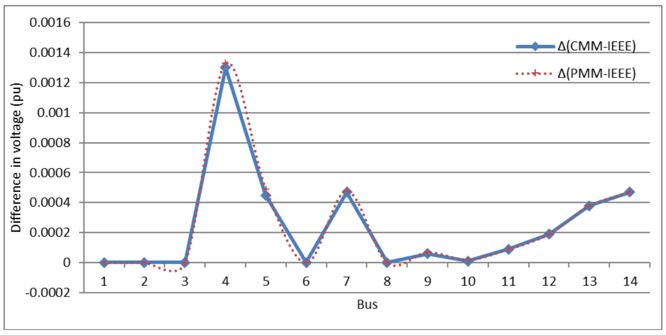

Figure 6. Differences from IEEE Final Voltages for the IEEE 14-bus System

It can also be observed that as the size of the system increases, the power mismatch method produces results consistently very similar to the original IEEE test data. For the cases of IEEE 14-bus, 30-bus and 57-bus, both methods show exact match with negligible differences from the IEEE final voltages in the input data. However, for IEEE 118-bus and 300-bus, the power mismatch method results in very narrow differences from the IEEE final voltage as suggested in the input data.

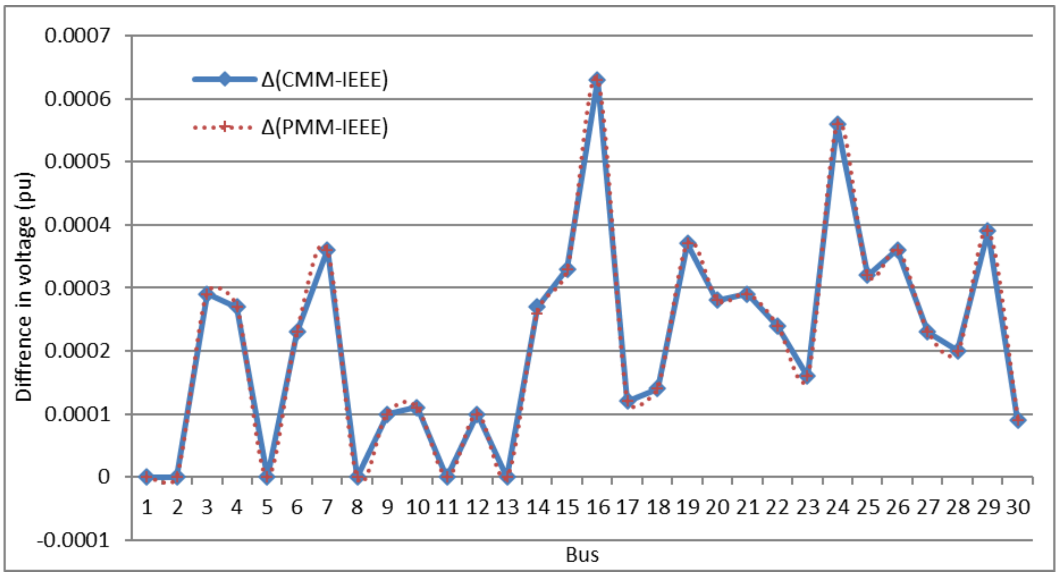

Figure 7. Differences from IEEE Final Voltages for the IEEE 30-bus System

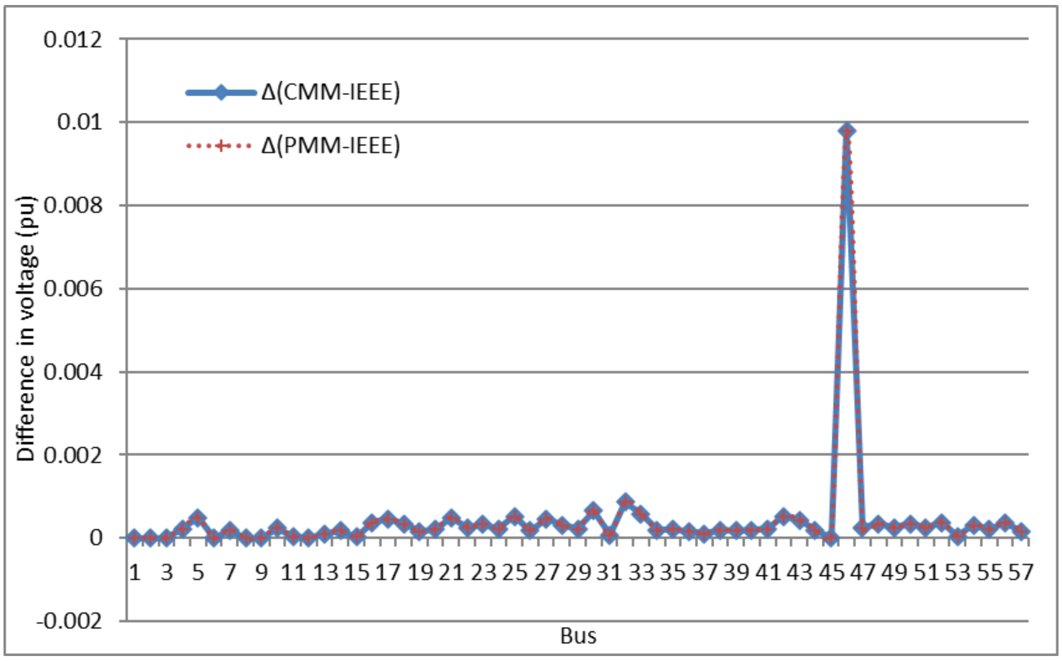

Figure 8. Differences from IEEE Final Voltages for the IEEE 57-bus System

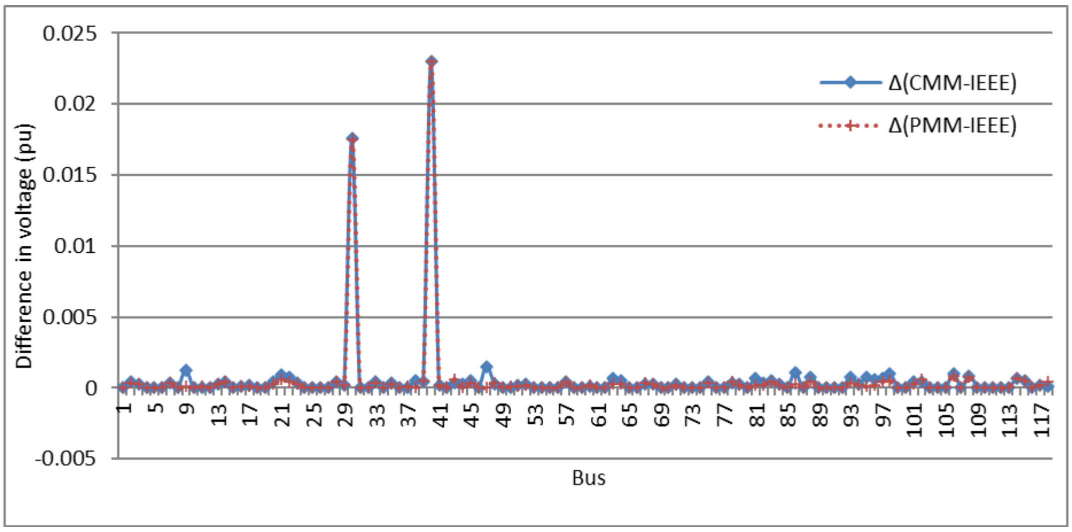

Figure 9. Differences from IEEE Final Voltages for the IEEE 118-bus System

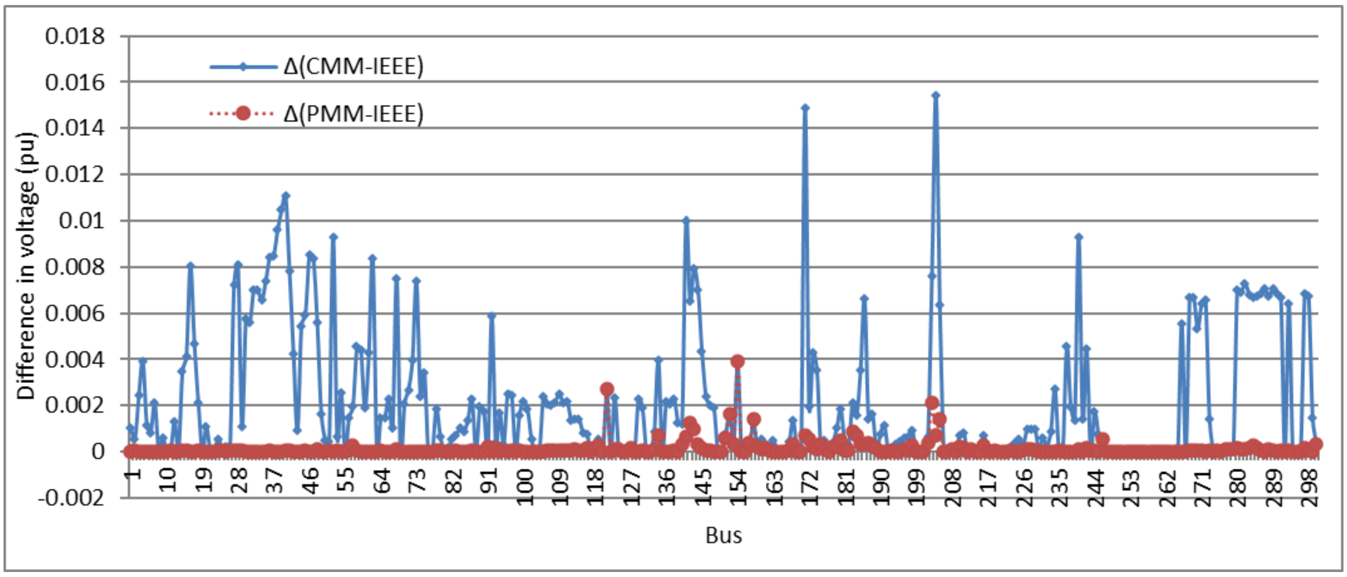

Figure 10. Differences from IEEE Final Voltages for the IEEE 300-bus System

IV. Conclusions

The paper compares the convergence properties of two implementations of power flow analysis, namely, the current- and power-mismatch methods. Both implementations are based on full rectangular form and well-known vector-based Python libraries. These will later be incorporated into an optimal power flow calculation using the non-linear primal-dual interior point method based on vector-form. Overall, the mismatch progression over iterations shows that the current-mismatch method decays faster than the power mismatch method. In future work, we will consider the robustness of each method with respect to network parameters, topology and constraints.

Acknowledgment

Authors wishing to acknowledge LPPM Universitas Lampung for its support through Hibah Penelitian Dasar (No. 1330/UN26.21/PN/2020) as this work is a part of this research fund.

References

[1] F. L. Alvarado, "Solving power flow problems with a Matlab implementation of the power system applications data dictionary", Decision Support Systems, vol. 30, pp. 243-54, 2001.

[2] F. Milano, "An Open-Source Power System Analysis Toolbox," in IEEE Transactions on Power Systems, vol. 20, no. 3, pp. 1199-1206, Aug. 2005.

[3] R. D. Zimmerman, C. E. Murillo-Sanchez and R. J. Thomas, "MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education," in IEEE Transactions on Power Systems, vol. 26, no. 1, pp. 12-19, Feb. 2011.

[4] F. Milano, "A Python-based software tool for power system analysis," 2013 IEEE Power & Energy Society General Meeting, Vancouver, BC, 2013, pp. 1-5.

[5] Tinney, W.F., Hart, C.E., “Power flow solution by Newton’s Method,” IEEE Trans. On Power Apparatus and Systems, vol. PAS-86, no. 11, Nov. 1967, pp. 1449-1460.

[6] Stott, B., Alsac, O., “Fast decoupled load flow,” IEEE Trans. On Power Apparatus and System, vol. PAS-93, no. 3, May 1974, pp. 859-862.

[7] Da Costa, V.M., Martins, N., Pereira, J.L.R., “Developments in the Newton Raphson power flow formulation based on current injections,” IEEE Trans. On Power Systems, vol. 14, no. 4, Nov. 1999, pp. 1320-1326.

[8] Ferreira, C.A., Da Costa, V.M., “A second order power flow based on current injection equations,” Int’l Journal of Electrical Power and Energy Systems, vol. 27, no. 4, May 2005, pp. 254-263.

[9] Exposito, A.G., Ramos, E.R., Dzafic, I., “Hybrid real-complex current injection-based load flow formulation,” Electric Power Systems Research, vol. 119, Feb. 2015, pp. 237-246.

[10] Milano, F., “On current and power injection models for angle and voltage stability analysis of power systems,” IEEE Trans. on Power Systems, vol. 31, no. 3, May 2016, pp. 2503-2504.

[11] Da Costa, V.M., Rosa, A.L.S., “A comparative anlysis of different power flow methodologies,” Proc. of 2008 IEEE/PES Trans. and Dist. Conference and Exposition: Latin America, 13-15 Aug. 2008, pp. 1090-1096.

[12] Kamel, S., Abdel-Akher, M., Jurado, F., “Improved NR current injection load flow using power mismatch representation of PV bus,” Electrical Power and Energy Systems, vol. 53, Dec. 2013, pp. 64-68.

[13] Garcia, P.A.N., Pereira, J.L.R., Cameiro Jr, S., Vinagre, M.P., Gomes, F.V., “Improvements in the representation of PV buses on three-phase distribution power flow,” IEEE Trans. on Power Delivery, vol. 19,

no. 2, April 2004, pp. 894-896.

[14] Oliveira, C.C., Bonini Neto, A., Minussi, C.R., Alves, D.A., Castro, C.A., "New representation of PV buses in the current injection Newton power flow," Electrical Power and Energy Systems, vol. 90, 2017, pp. 237-244.

[15] Jones E, Oliphant E, Peterson P, et al., "SciPy: Open-Source Scientific Tools for Python," 2001-, URL: http://www.scipy.org/ [Online; accessed 2017-04-15].

[16] Van der Walt, S., Colbert, S.C., Varoquaux, G., "The NumPy Array: a Structure for Efficient Numerical Computation," Computing in Science and Engineering, vol. 13, no. 2, March 2011, pp. 22-30.

[17] University of Washington, "Power System Test Archive," Aug. 1999, URL: http://www2.ee.washington.edu/research/pstca/, [Online; accessed 2020-04-12]