Jurnal Nasional Teknik Elektro, Vol. 8, No. 1, Maret 2019
p-ISSN: 2302-2949, e-ISSN: 2407 - 7267

Vector Form Implementation in Three-Phase Power Flow Analysis Based on Power Injection Rectangular Coordinate

Lukmanul Hakim*, Fandi Prayoga, Khairudin, Herri Gusmedi
Jurusan Teknik Elektro, Fakultas Teknik, Universitas Lampung
*Corresponding author, e-mail: lukmanul.hakim@eng.unila.ac.id

Abstract— This paper aims to proposed a vector form implementation into three-phase power flow analysis. The developed algorithm is based on Newton-Raphson method with voltage is represented in rectangular coordinate. The Python programming language and its mathematical libraries are used in this works. Three-phase power flow analysis in vector form utilizes sparse matrix ordering algorithm, hence the elements of the coefficient correction matrix can be rearranged easily. This method was used to solve three-phase power flow for balance or unbalance network in two actual distribution system feeders in Lampung, *i.e.* 119 nodes and 191 nodes. Comparison with traditional Newton-Raphson method (non-vector) shows the vector form is able to solve computation up to eight times faster than the non-vector.

Keywords: Three-phase power flow, Vector form, Newton-Raphson, Rectangular, Python

1. Introduction

Power balance between generation and demand is the goal of power system operation. This goal is not easy to be achieved due to unpredictable power losses in power system components. It is therefore, power flow calculation is necessary. A robust and accurate power flow analysis software is then become compulsory [1]. The computation must be fast without sacrifice the accuracy of the result, hence correct decisions can be taken immediately[2].

Generally, the Newton methods are often used to solve the power flow analysis problem. In the traditional Newton-Raphson method, the coefficient matrix of correction equations (the Jacobian matrix), which are partial derivatives of power injection with respect to the real part and imaginary part of voltage variables, are represented in the program by having derivativ e equations for each element of the matrix, as in equations (17) to (24). In addition, the technique requires many for-loop functions to accommodate loops on each bus and branch. When this technique is used to solve problems with a large number of buses, a for-loop function is executed against the number of buses. This results in longer time required to complete the computation and sometimes it is difficult to converge[3, 4].

In reference [5], Power System Applications Data Dictionary (PSADD) has been implemented to solving power flow problem. The software employed vectorized computation technique in MATLAB and was very efficient and easy to solve simple power flow. In reference [6], the vectorization mode was implemented in power flow calculation based on non-linear programming. The method using Approximate Minimum Degree (AMD) [7] and sparse Cholesky factorization (LDL^T) [8] reordering algorithm methods to calculate the matrix of correction equation, thus the fill-in element can be reduced. This technique greatly increase the calculation speed. The vectorization method was also implemented in the optimal power flow problem. It was based on an object-oriented library (C++ power system vectorization calculation and OPF program) in rectangular coordinate [9].

In this research, the vectorization form is implemented in three-phase power flow analysis. Python software package and its mathematical libraries like NumPy and SciPy are used in this works. NumPy supports array objects and routines that make indexing the matrix and solve linear algebra problems become easier. Since the coefficient matrix is not a full matrix, SciPy with support for sparse matrix manipulation is utilized. It reduces memory usage significantly and

Received date 2018-12-1, Revised date 2019-3-26, Accepted date 2019-3-26
https://doi.org/10.25077/ jnte.v8n1.612.2019

results in faster computing time [10]. Although three-phase power flow analysis has exised for long [11], it is still a challenging topic due to the complexity and size of the problem. This research emphasizes on modeling of computation technique using a Newton-Raphson method based on rectangular coordinate in vector form for three-phase power flow analysis. It is expected that the computation time and the convergence are better demonstrated with this method.

2. Problem Formulation

2.1. Mismatch Equations

The power balance problem is a criterion that must be solved in the power flow analysis. In previous research, the power mismatch method in each node has been implemented in [2, 4, 9, 10]. For active and reactive power are part of the complex power, in vector form, the mathematical operations take place in the full matrix. The power balance equation problem of three-phase power flow analysis in vector form can be represented as follows

$$\Delta\bar{\boldsymbol{s}}_{abc} = \bar{\boldsymbol{s}}_{sch,abc} - \bar{\boldsymbol{s}}_{abc} = 0 \tag{1}$$

where $\bar{\boldsymbol{s}}_{sch,abc}$ is three-phase complex power scheduled between power generation and loading, and $\bar{\boldsymbol{s}}_{abc}$ is three-phase complex power injection. All variables in a vector matrix with $3 \times n_b$ bus size, with n_b is a number of buses. Equation (1) can be applied for PQ buses. For PV buses, in the rectangular coordinate, the imaginary parts of equation (1) can be changed by voltage magnitude. So, the equation (1), according to reference [12], for PV buses become

$$\Delta\bar{\boldsymbol{s}}_{abc} = \Re\{\Delta\bar{\boldsymbol{s}}_{abc}\} + j\Delta\bar{\boldsymbol{v}}^2_{abc} = 0 \tag{2}$$

where, $\bar{\boldsymbol{v}}_{abc}$ is three-phase voltage magnitude, with $\Delta\bar{\boldsymbol{v}}^2_{abc} = \bar{\boldsymbol{v}}^2_{sch,abc} - \bar{\boldsymbol{v}}^2_{abc}$.

The contents of the power injection equation (1) are

$$\bar{\boldsymbol{s}}_{abc} = \bar{\boldsymbol{V}}_{abc} \cdot \bar{\boldsymbol{\iota}}^*_{abc} \tag{3}$$

with

$$\bar{\boldsymbol{\iota}}_{abc} = \bar{\boldsymbol{Y}}_{abc} \cdot \bar{\dot{\boldsymbol{v}}}_{abc} \tag{4}$$

from equation (3) and (4),

$$\bar{\boldsymbol{s}}_{abc} = \bar{\boldsymbol{V}}_{abc} \cdot \left(\bar{\boldsymbol{Y}}_{abc} \cdot \bar{\dot{\boldsymbol{v}}}_{abc}\right)^* \tag{5}$$

where $\bar{\boldsymbol{V}}_{abc}$ is diagonal matrix of three-phase voltage ($\bar{\boldsymbol{v}}_{a,1}, \dots, \bar{\boldsymbol{v}}_{c,1}, \bar{\boldsymbol{v}}_{a,2}, \dots, \bar{\boldsymbol{v}}_{c,2}, \dots \bar{\boldsymbol{v}}_{a,n_b}, \dots, \bar{\boldsymbol{v}}_{c,n_b}$) with a size of $3n_b \times 3n_b$, and $\bar{\boldsymbol{Y}}_{abc}$ is three-phase admittance matrix of the system with a size of $3n_b \times 3n_b$, and $\bar{\dot{\boldsymbol{v}}}_{abc}$ is vector matrix of three-phase voltage with a size of $3n_b$.

2.2. Correction Equations

The power balance equation (1) can be expanded using the Taylor series theorem with the higher order terms are neglected as follows,

$$\begin{aligned}\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Re\{\bar{\dot{\boldsymbol{v}}}_{abc}\}}\Re\{\Delta\bar{\dot{\boldsymbol{v}}}_{abc}\} \\ + \frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Im\{\bar{\dot{\boldsymbol{v}}}_{abc}\}}\Im\{\Delta\bar{\dot{\boldsymbol{v}}}_{abc}\} \\ = \bar{\boldsymbol{s}}^{(0)}_{G,abc} - \bar{\boldsymbol{s}}^{(0)}_{L,abc} \\ - \bar{\boldsymbol{s}}^{(0)}_{i,abc}\end{aligned} \tag{6}$$

Power balance equation is a non-linear simultaneous equation of voltage phasor. Newton-Raphson method is an efficient algorithm for solving that problem. In general, the correction equation is

$$\boldsymbol{f}\left(\boldsymbol{x}^{(i)}\right) = \nabla^T_x \boldsymbol{f}^{\ (i)}\Delta\boldsymbol{x}^{(i)} \tag{7}$$

or

$$\Delta\boldsymbol{x}^{(i)} = \left[\nabla^T_x \boldsymbol{f}^{(i)}\ \right]^{-1}\boldsymbol{f}\left(\boldsymbol{x}^{(i)}\right) \tag{8}$$

where $\nabla^T_x \boldsymbol{f}$ is the coefficient of the correction equation which are derivatives of the variable $\boldsymbol{x}$**,** and i is the i^{th} iterations. The correction equations can be represented in vector form by the real and imaginary part of (6). The correction equation for three-phase power flow analysis is as follows.

$$\begin{bmatrix}\Re\{\Delta\bar{\boldsymbol{s}}_{abc}\}\\ \Im\{\Delta\bar{\boldsymbol{s}}_{abc}\}\end{bmatrix} = \begin{bmatrix}\Re\left\{\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Re\{\bar{\boldsymbol{v}}_{abc}\}}\right\} & \Re\left\{\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Im\{\bar{\boldsymbol{v}}_{abc}\}}\right\}\\ \Im\left\{\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Re\{\bar{\boldsymbol{v}}_{abc}\}}\right\} & \Im\left\{\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Im\{\bar{\boldsymbol{v}}_{abc}\}}\right\}\end{bmatrix}\begin{bmatrix}\Re\{\Delta\bar{\boldsymbol{v}}_{abc}\}\\ \Im\{\Delta\bar{\boldsymbol{v}}_{abc}\}\end{bmatrix} \tag{9}$$

The elements of the coefficient matrix of the correction equation are partial derivatives of power injection with respect to the real and imaginary part of the voltages.

For PQ buses, the contents of the coefficient of the correction equation as

$$\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Re\{\bar{\boldsymbol{v}}_{abc}\}} = \bar{\boldsymbol{I}}^*_{abc} + \bar{\boldsymbol{V}}_{abc}\cdot\bar{\boldsymbol{Y}}^*_{abc} \tag{10}$$

$$\frac{\partial\bar{\boldsymbol{s}}_{abc}}{\partial\Im\{\bar{\boldsymbol{v}}_{abc}\}} = j\bar{\boldsymbol{I}}^*_{abc} - j\bar{\boldsymbol{V}}_{abc}\cdot\bar{\boldsymbol{Y}}^*_{abc} \tag{11}$$

where $\bar{\boldsymbol{I}}^*_{abc}$ is diagonal matrix of three-phase conjugate current ($\bar{\boldsymbol{\iota}}_{a,1}$, ... , $\bar{\boldsymbol{\iota}}_{c,1}$, $\bar{\boldsymbol{\iota}}_{a,2}$, ... , $\bar{\boldsymbol{\iota}}_{c,2}$, ... $\bar{\boldsymbol{\iota}}_{a,n_b}$, ... , $\bar{\boldsymbol{\iota}}_{c,n_b}$) with a size of $3n_b \times 3n_b$.

For PV bus, $\Im\left\{\frac{\partial\bar{s}_{abc}}{\partial\Re\{\bar{v}_{abc}\}}\right\}$ is replaced by $\Im\left\{\frac{\partial v^2_{abc}}{\partial\Re\{\bar{v}_{abc}\}}\right\}$, and $\Im\left\{\frac{\partial\bar{s}_{abc}}{\partial\Im\{\bar{v}_{abc}\}}\right\}$ is replaced by $\Im\left\{\frac{\partial v^2_{abc}}{\partial\Im\{\bar{v}_{abc}\}}\right\}$.

$$\frac{\partial\boldsymbol{v}^2_{abc}}{\partial\Re\{\bar{\boldsymbol{v}}_{abc}\}} = 2\Re\{\bar{\boldsymbol{v}}_{abc}\} \tag{12}$$

$$\frac{\partial\boldsymbol{v}^2_{abc}}{\partial\Im\{\bar{\boldsymbol{v}}_{abc}\}} = 2\Im\{\bar{\boldsymbol{v}}_{abc}\} \tag{13}$$

2.3. Updating Variables

Rewrite equation (8), the coefficient of the correction equation is a linear equation, it can be solved by SciPy solver to get a value of $\Delta\boldsymbol{x}$. Here, $\Delta\boldsymbol{x}$ used as $\Delta\bar{\boldsymbol{v}}_{abc}$ variable, so that a new voltage is obtained

$$\bar{\boldsymbol{v}}^{(i+1)}_{abc} = \bar{\boldsymbol{v}}^{(i)}_{abc} + \Delta\bar{\boldsymbol{v}}^{(i)}_{abc} \tag{14}$$

with $\bar{\boldsymbol{v}}^{(i+1)}_{abc}$ is three-phase new voltage, $\bar{\boldsymbol{v}}^{(i)}_{abc}$ and $\Delta\bar{\boldsymbol{v}}^{(i)}_{abc}$ is three-phase voltage and voltage different at current iteration.

This computing process continues until the power balance equation meets the following conditions as

$$max\{|\Delta\bar{\boldsymbol{s}}_{abc}|\} < \varepsilon \tag{15}$$

or

$$i = i^{max} \tag{16}$$

where ε is the error that tolerated and i is a number of iterations.

3. Proposed Method

Figure 1 shows a flow chart of three-phase power flow in vector form. The process of calculating the power flow is outlined as follows:

(i) Read input data. Input data include initial voltage on each bus, generator data, sequence impedance of the line, and load data.

(ii) Make an admittance matrix of the three-phase system ($\bar{\boldsymbol{Y}}_{abc}$) with a size of $3n_b \times 3n_b$. The matrix value is calculated from the impedance of the line data.

(iii) Determine the initial value for the voltage magnitude ($\bar{\boldsymbol{v}}_{abc}$) and phase angle ($\theta_{abc}$) for each phase. The magnitude and phase angle of the voltage values are initially considered to be 1.0 *p.u.* (per unit) with an angle of $30°$ in phase a, $-90°$ in phase b, and $150°$ in phase c. Then the voltage is changed from polar to rectangular coordinates by $\Re\{\bar{\boldsymbol{v}}_{abc}\} = \boldsymbol{v}\cos\boldsymbol{\theta}$ and $\Im\{\bar{\boldsymbol{v}}_{abc}\} = \boldsymbol{v}\sin\boldsymbol{\theta}$.

(iv) Specify the initial iteration value of $i = 0$.

(v) Calculate the three-phase power mismatch ($\Delta\bar{\boldsymbol{s}}_{abc}$) with the equation (1).

(vi) Compare the mismatch value of the i-th iteration with the prespecified error. If the mismatch value is less than specified error (10^{-5}) or the number iteration of $i = i^{max}$, then go to step (xi). If it does not meet these conditions, proceed to the next stage.

(vii) Calculate the matrix element of the coefficient of the correction equation (Jacobian) based on the equation (10) and (11).

(viii) Solve simultaneous linear equations of (8) using SciPy solver to get a value of $\Delta\bar{\boldsymbol{v}}_{abc}$.

(ix) Calculate the new voltage as in equation (14).

(x) Add iteration $i = i + 1$and return to step (v).

(xi) Displays computational results such as voltage and power profiles on each bus.

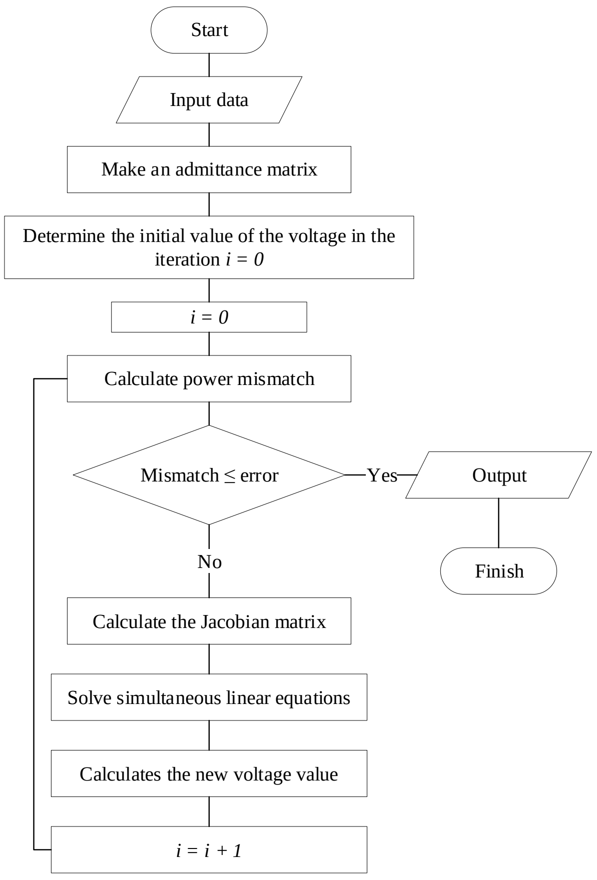
Start
Input data
Make an admittance matrix
Determine the initial value of the voltage in the iteration i = 0
i = 0
Calculate power mismatch
Mismatch ≤ error
Yes
Output
Finish
No
Calculate the Jacobian matrix
Solve simultaneous linear equations
Calculates the new voltage value
i = i + 1

Figure 1. Flowchart of three-phase power flow modeling in vector form.

4. Results and Discussions

In this work, the performance of the method offered is compared with the conventional method. The first test was carried out on the 11-node system as Figure 2.

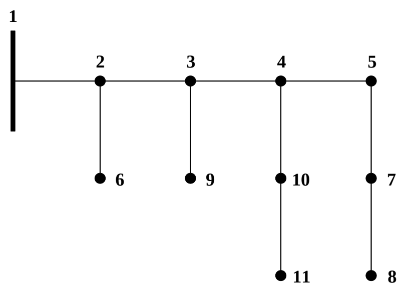

Figure 2. 11-node test system.

Figure 2 is the 11-node test system. The first bus is a slack bus, and a total of a load is 1.77 MW active power load and 1.35 MVAr reactive power. All load is assumed as constant power. The line to the neutral profile voltage magnitude of 11-node test system shown in Figure 3.

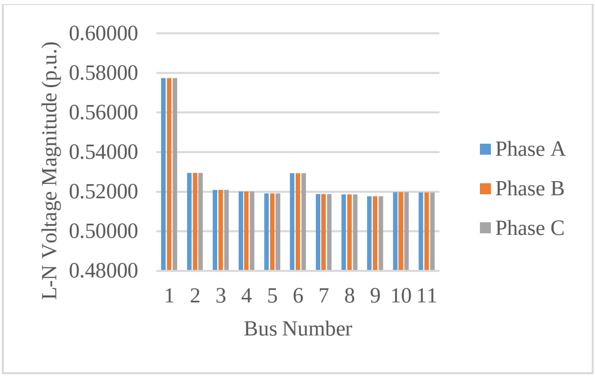

Figure 3. Line to neutral voltage magnitude profile of 11-node test system.

The 119-node test system is an actual distribution feeder in Lampung for Rawajitu area. The total of a load is 4.04 MW active power load and 2.46 MVAr reactive power load. In this case, there is 1 PV bus with capacity 5 MW. Figure 4. is the line to the neutral profile voltage magnitude of the 119-node test system.

The 191-node test system is another distribution feeder in Lampung towards Mesuji area. The total load, in this case, is 3.77 MW active power load and 2.34 MVAr reactive power load. In this case, there is one PV bus with capacity of 5 MW. Figure 5 is the line to the neutral profile voltage magnitude of the 191-node test system.

Figure 6 is the convergence graph between vector form and non-vector form when simulated for each case. In the 11-node test system, the computation finished at the third iteration, while in the 119-node and 191-node test system, the computation finished at the fourth iteration.

In the 11-node test system, the mismatch values between vector form and non-vector are the same for all iterations. On the first iteration, the mismatch value is 0.00400, and the second iteration, the mismatch value is 0.00036. The computation is finished in the third iterations when the mismatch value less than the error (10^{-5}).

In the 119-node test system, the mismatch values between vector form and non-vector are slightly different. On the first iteration, the mismatch value between vector form and non-vector form is the same, that is 0.00653. On the second iteration, the mismatch value using vector form is 0.00632, and non-vector 0.00148. In the 191-node test system, the mismatch value on the first iteration using vector form and non-vector is the same, that is 0.01649. On the second iteration, the mismatch value using vector form is 0.00142, and non-vector 0.00028. On the third iteration, the mismatch value using vector form is 0.00005, and non-vector is 0.00001

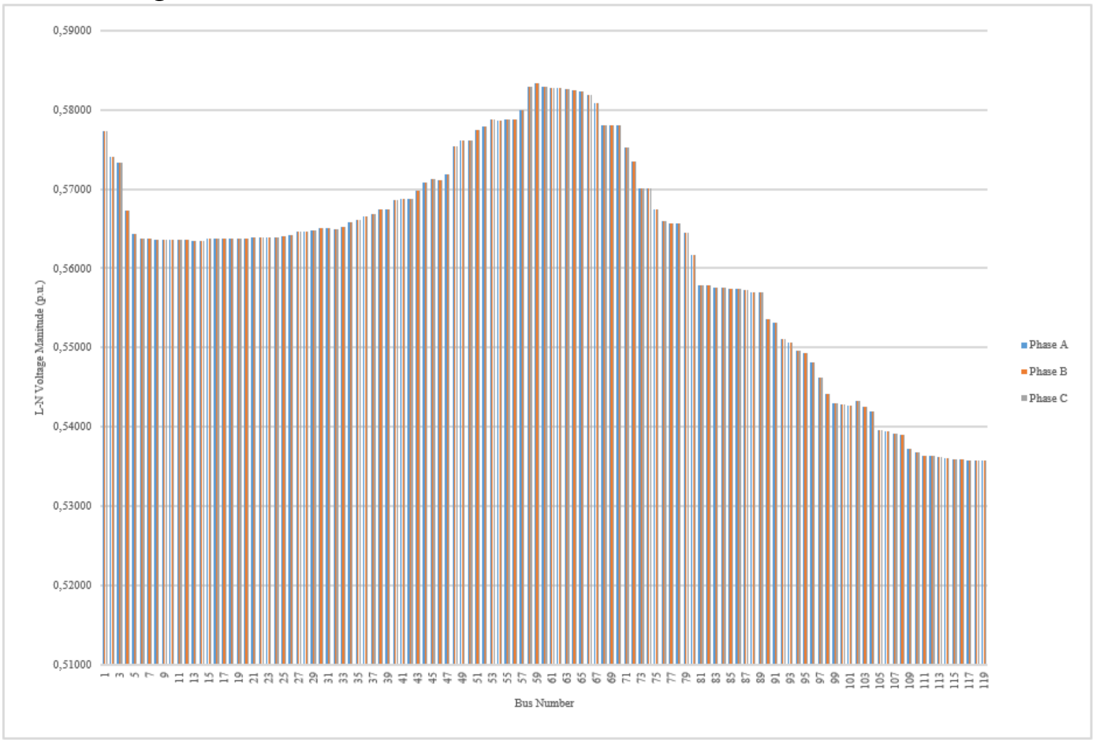

Figure 4. The line to neutral voltage magnitude profile of 119-node test system.

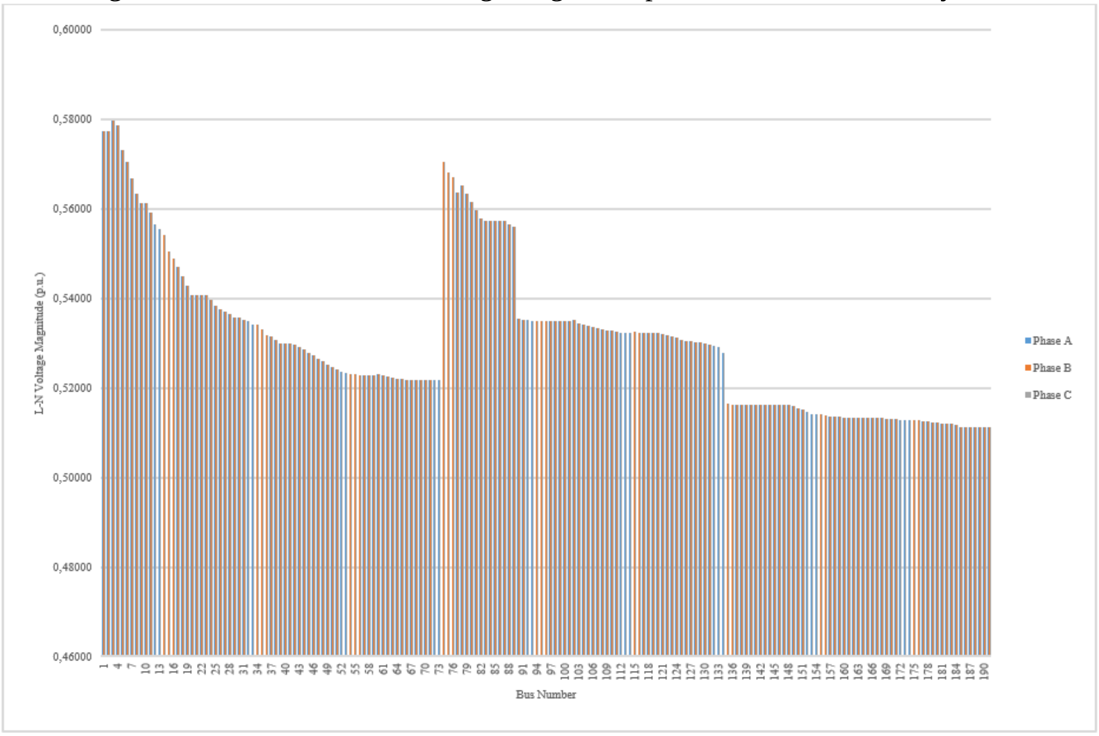

Figure 5. The line to neutral voltage magnitude profile of 191-node test system.

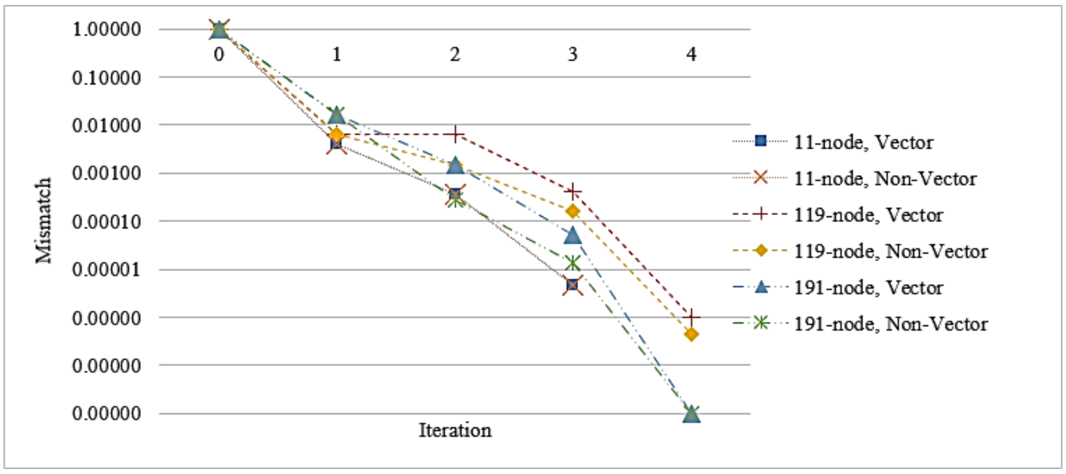

Figure 6. Convergence of Vector Form and Non-Vector Form when simulated on each of cases.

Figure 7 is the computation time between vector form and non-vector in each of the cases.

In the 11-node test system, power flow analysis in vector form finished the computation in 0.00248 seconds, and the non-vector in 0.02149 seconds. This means, the vector form was faster around eight times than non-vector. For the 119-node test system, the vector form finished the computation with time 0.02471 seconds, and the non-vector in 0.20840 seconds. For this case, vector forms faster around eight times than non-vector. While for the 191-node test system, the vector form finished the computation with time 0.03120 seconds, and the non-vector in 0.13229 seconds. In this case, the vector forms was faster around four times than non-vector. Efficient computation by having the vector form relies greatly on efficient implementation of vector-matrix manipulation routine provided by Numpy and Scipy. It is important to reduce the number of for-loops in Python by utilizing vector-matrix operations provided by the computing libraries in Python.

The operation of the coefficient correction equation calculation is done as follows. For the diagonal element of coefficient correction matrix $(h = k)$ in PQ bus,

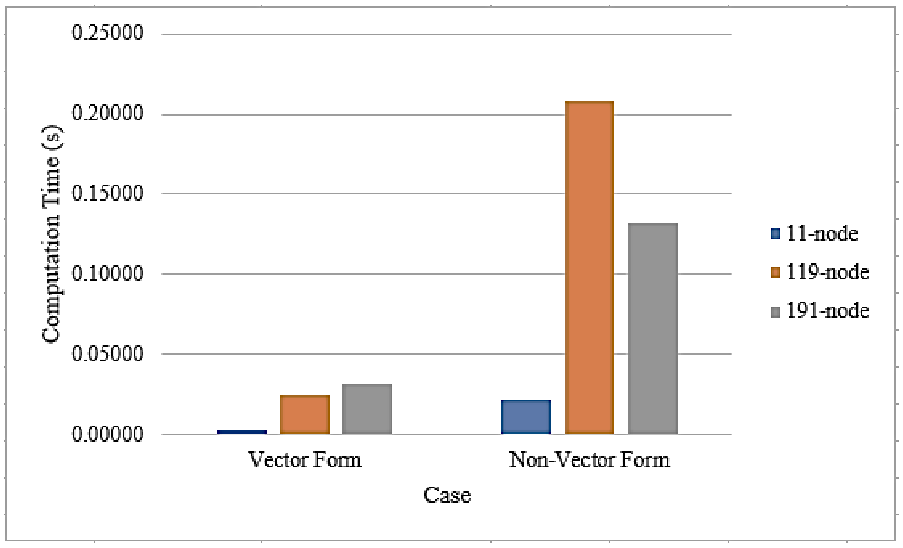

Figure 7. The computation time of three-phase power flow in vector form and non-vector.

$$\frac{\partial \Delta P_{h,abc}}{\partial e_{h,abc}} = \sum_{k\in h}\left[\left(e_{k,abc}\cdot G_{hk,abc} - f_{k,abc}\cdot B_{hk,abc}\right)\right] + e_{h,abc}\cdot G_{hh,abc} + f_{h,abc}\cdot B_{hh,abc} \qquad (17)$$

$$\frac{\partial \Delta P_{h,abc}}{\partial f_{h,abc}} = \sum_{k\in h}\left[\left(e_{k,abc}\cdot B_{hk,abc} + f_{k,abc}\cdot G_{hk,abc}\right)\right] - e_{h,abc}\cdot B_{hk,abc} + f_{h,abc}\cdot G_{hk,abc} \qquad (18)$$

$$\frac{\partial \Delta Q_{h,abc}}{\partial e_{h,abc}} = \sum_{k\in h}\left[\left(e_{k,abc}\cdot B_{hk,abc} + f_{k,abc}\cdot G_{hk,abc}\right)\right] - e_{h,abc}\cdot B_{hk,abc} + f_{h,abc}\cdot G_{hk,abc} \qquad (19)$$

$$\frac{\partial \Delta Q_{h,abc}}{\partial f_{h,abc}} = \sum_{k\in h}\left[\left(e_{k,abc}\cdot G_{hk,abc} - f_{k,abc}\cdot B_{hk,abc}\right)\right] - e_{h,abc}\cdot G_{hh,abc} - f_{h,abc}\cdot B_{hh,abc} \qquad (20)$$

where ΔP and ΔQ are active and reactive power mismatch in every bus. In PV bus applies the equation as follows.

$$\frac{\partial \Delta v_{h,abc}^2}{\partial e_{h,abc}} = 2\cdot e_{k,abc} \qquad (21)$$

$$\frac{\partial v_{h,abc}}{\partial f_{h,abc}} = 2\cdot f_{k,abc} \qquad (22)$$

For off-diagonal element of coefficient correction equation ($h \neq k$),

$$\frac{\partial \Delta P_{h,abc}}{\partial e_{k,abc}} = -\frac{\partial \Delta Q_{h,abc}}{\partial f_{k,abc}} = e_{h,abc}\cdot G_{hk,abc} + f_{h,abc}\cdot B_{hk,abc} \qquad (23)$$

$$\frac{\partial \Delta P_{h,abc}}{\partial f_{k,abc}} = \frac{\partial \Delta Q_{h,abc}}{\partial e_{k,abc}} = -e_{h,abc}\cdot B_{hk,abc} + f_{h,abc}\cdot G_{hk,abc} \qquad (24)$$

where $e = \Re\{v_{abc}\}$ and $f = \Im\{v_{abc}\}$, while h and k is a bus number.

Equation (17) until (24) is applied in the program will require many *for-loop* functions. If these techniques implemented in systems with a large number of buses, *for-loop* functions will be executed against the number of buses. So, for that reason, non-vector requires more computing time. The total of the command line for this technique around 269 lines.

While in vector form, coefficient correction matrix arranged in full matrix based on partial derivative results, so the use of the *for-loop* function can be reduced. The elements of the coefficient correction equation in vector form are shown in the equations (10) and (11).

By using matrix manipulation algorithm (*hstack* and *vsatck*), the element of the coefficient correction matrix can be rearranged horizontally and vertically. The format of the coefficient correction matrix in vector form is shown in the equation (9).

5. Conclusions

The formulation three-phase power flow analysis in vector form using Python programming language has proven capable to solve power flow analysis with promising result. By comparing between this technique and non-vector technique, vector form is eight times faster than the non-vector technique. Three-phase power flow analysis in vector form utilize reorder algorithm, therefore the element of the coefficient correction matrix can be rearranged easily. In further research, this technique will be implemented for a more complex system i.e. missing phase of lines, heavy unbalance load, and higher R/X ratio.

References

[1] Demirok, E; Kjær, S. B.; Dezso, S.; and Teodorescu, R.,"Three-Phase Unbalanced Load Flow Tool for Distribution Networks," *Proc. of 2nd Int'l. Workshop on Integration of Solar Power System*, p. 9, 2012.

[2] Sereeter, B.; Vuik, K.; and Witteveen, C.; "Newton Power Flow Methods for Unbalanced Three-Phase Distribution Networks," *Energies*, vol. 10, no. 10, p. 1658, 2017.

[3] Milano, F., "Continuous Newton ' s Method for Power Flow Analysis," IEEE. Trans. on Power Systems, vol. 24, no. 1, pp. 50–57, Feb. 2009.

[4] Hakim, L.; Wahidi, M.; Murdika, U.; Milano, F.; Kubokawa, J.; Yorino, N.,"A

Three-phase Power Flow Analysis for Electrical Power Distribution System with Low Voltage Profile," *Proc. 2015 2nd Int. Conf. Inf. Technol. Comput. Electr. Eng. (ICITACEE),* Semarang, 16-18 Oct. 2015.

[5] Alvarado, F. L.,"Solving Power Flow Problems with a Matlab Implementation of the Power System Applications Data Dictionary," *Proc. of the 32nd Annual Hawaii Int'l Conf. on Systems Sciences*, Maui, 5-8 Jan 1999.

[6] Yude, Y.; Zhijun, Q.; and Wei, H., "Power Flow Calculation Based on Non-linear Programming Model and Vectorization Mode," *Proc. of 2007 Int'l Conf. on Mechatronics and Automation,* Harbin, 5-8 Aug. 2007.

[7] P. R. Amestoy, I. Enseeiht, T. A. Davis, and I. S. Duff, "Algorithm 837: AMD, an Approximate Minimum Degree Ordering Algorithm," *ACM Trans. on Mathematical Software (TOMS)*, vol. 30, no. 3, pp. 381–388, Sep. 2004.

[8] Gaviano, M.; Kvasov, D. E.; Lera, D.; and Sergeyev, Y.D.; "Algorithm 829: Software for generation of classes of test functions with known local and global minima for global optimization," *ACM Trans. on Mathematical Software (TOMS)*, vol. 29, no. 4, pp. 469–480, Dec. 2003.

[9] Qin, Z.; and Yang, Y.; "Vectorization implementation of optimal power flow in rectangular form based on interior point method," *Proc. of 2008 IEEE PES GM - Convers. Deliv. Electr. Energy in the 21st Century, Pittsburgh*, 20-24 Jul. 2008.

[10] Milano, F.,"A python-based software tool for power system analysis," *Proc. of 2013 IEEE PES Gen. Meet.,* Vancouver, 21-25 Jul. 2013

[11] Birt, K. A.; Graffy, J. J.; Mcdonald, J. D.; El-Abiad, A. H.,"Three Phase Load Flow Program," *IEEE Trans. on Power App. and Systs.,* vol. 95, no. 1, pp. 59–65, Jan. 1976.

[12] Xi-Fan Wang, Y. Song, and M. Irving, *Modern Power Systems Analysis.* , Springer, India, 2013.

Biographies

Lukmanul Hakim received Bachelor of Engineering from Universitas Sriwijaya, Indonesia, in 1996. He finished his M.Sc. at the University of Manchester (formerly UMIST), the United Kingdom, in 1999 and his Dr. Eng. from Hiroshima University in 2010. He is currently a senior lecturer at Universitas Lampung since 2000.

Fandi Prayoga received his Bachelor of Engineering from Electrical Engineering Department, Engineering Faculty, Universitas Lampung in 2018.

Khairudin received Bachelor of Engineering from Universitas Sriwijaya, Indonesia, in 1994. He finished his M.Sc. at the University of Manchester (formerly UMIST), the United Kingdom, in 1999 and his Ph.D.Eng. from Kyushu Institute of Technology in 2016. He is currently a lecturer at Universitas Lampung since 2000.

Herri Gusmedi received Bachelor of Engineering from Universitas Sriwijaya, Indonesia, in 1995. He finished his M.T. at Institut Teknologi Bandung in 2000. He is currently a senior lecturer at Universitas Lampung since 1999.