Cellular Computational Network for Distributed Power Flow Inferencing in Electric Distribution Systems

Hasala Dharmawardena, *Student Member, IEEE* *
Ganesh K. Venayagamoorthy, *Senior Member, IEEE* * †
*Real-Time Power and Intelligent System Laboratory
Clemson University, South Carolina, USA
†School of Engineering, University of KwaZulu-Natal, Durban, South Africa
hasala@ieee.org and gkumar@ieee.org

***Abstract*—The modern power system is undergoing a rapid transformation from centralized generation to distributed generation. The distributed generation consists of a large number of small distributed energy resource (DER) based generators with stochastic power output connected at different geographic locations throughout the feeder. Therefore, the operation of the future distribution grid requires efficient, scalable, and robust techniques for system analysis and control. This study presents a data driven approach to solve the electric power flow problem based on the framework known as the Cellular Computational Network (CCN). It is scalable since diakoptics resolves the system to computational cells, which are then connected together to form the full system. The results for the IEEE 4 Bus system shows that this approach can generate accurate power flow solutions.**

***Index Terms*—Smart grid, electric power distribution system, power flow, diakoptics, CCN**

I. Introduction

The exponential growth in the unconventional technologies is driving the power system to a state that was not envisioned by the planners and the operators of the past. This smart grid revolution spans through the full spectrum of power engineering, starting from generation, through transmission to distribution and loads. The system is increasing in size as well as in complexity as a result of this revolution. The traditional technologies could prove to be insufficient to address the engineering problems associated with the smart grid.

This paper demonstrates a novel technology to solve the power flow problem in a scalable and efficient way, by tearing the larger system into sub units following the Cellular Computational Network (CCN) framework (extension of CCN theory to the power flow problem). By using this approach, power flow for a large network can be solved in a High Performance Computing (HPC) framework. In the near future there will be stage where the system is so large that the power flow problem cannot be solved without a piecewise diakoptics approach.

This study is supported by the National Science Foundation (NSF) of the United States, under grants IIP #1312260, #1408141 and #1638321, and the Duke Energy Distinguished Professorship Endowment Fund. Any opinions, findings and conclusions or recommendations expressed in this material are those of author(s) and do not necessarily reflect the views of National Science Foundation and Duke Energy.

Diakoptics refers to the piecewise method of solution. It is a procedure for solving large-scale system problems by tearing or decomposing. The problems to be analyzed seem to grow at a very rapid pace. Neither the increased computer speeds nor the computer core storages seem to keep up with the size and complexity of the problems to be solved. Diakoptics approach has been applied in the past to solve large-scale system problems [1].

The technique presented in this paper is a data driven diakoptic approach and not the model based approach which was first proposed by Kron [1], [2], [3]. Since the smart grid revolution will inherently have significant amount of embedded sensors generating vast amounts of data, a data driven approach makes more sense than a model based approach, where the system is 'learned' from data rather than 'modeled' from physics. Additionally, advanced controls can be applied when the system can be learned.

The diakoptics is based on the cellular computational networks (CCN) framework first introduced in [4], and then developed into a complete framework in [5]. This framework has been successfully applied to voltage estimation [4], frequency estimation [6], [7] and wind forecasting [8], as well as power system contingency studies [9] in the past. The framework is based on mapping the physical connections directly to a computational network, which helps to recreate the total system using the connections. The connections between cells, however, are limited by the physical connections. CCN can be layered when more than one variable needs to be estimated. There will be connections between the layers to represent the dynamics of the connections between different state variables.

The rest of the paper is organized as follows: Section II describes the general power flow problem that this study is based on. Section III describes the case study that is analyzed in this paper. Section IV discusses the simulation results comparing the different methods applied. Finally, the conclusions and future work is given in Section V.

978-1-7281-2009-6/$31.00 paper N-20374.pdf

II. Power flow problem

Where, $V_k = |V_k|\angle\delta_k$, represents the node voltage and $Y_{rt} = |Y_{rt}|\angle\theta_{rt}$, represents components of the Bus admittance matrix, the power flow equations are defined by (1).

$$P_i - jQ_i = V_i^* \sum_{j=1}^{n} Y_{ij} V_j \tag{1}$$

$$P_i = \sum_{j=1}^{n} |V_i|\,|V_j|\,|Y_{ij}|\,cos(\theta_{ij} - \delta_i + \delta_j) \tag{2}$$

$$Q_i = \sum_{j=1}^{n} |V_i|\,|V_j|\,|Y_{ij}|\,sin(\theta_{ij} - \delta_i + \delta_j) \tag{3}$$

The power flow problem consists of solving a set of non-linear simultaneous equations. The power flow equations for a balanced power system takes the form given in (1), (2) and (3). One typical way to solve this is to apply the Newton Raphson (NR) method [10]. This method calculates the Jacobian based on (4) and then applies (5), to iteratively correct the error. This approach, however, requires the calculation of the inverse of the jacobian, which is computationally demanding and thereby places a constraint on the size of the system that can be solved.

$$J = \begin{bmatrix} \frac{\partial P}{\partial V} & \frac{\partial P}{\partial \delta} \\ \frac{\partial Q}{\partial V} & \frac{\partial Q}{\partial \delta} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = J \begin{bmatrix} \Delta V \\ \Delta \delta \end{bmatrix} \tag{5}$$

III. Case Study

A. Diakoptics of a Power system based on CCN for power flow

Here a large power system is cut into a number of computational units based on the physical interconnections between the buses. Each sub unit is trained separately and after it is sufficiently trained the networks are connected. Therefore, step one is to use a good dataset to train the cells separately. Step two is to connect the network in the CCN structure and iterate till systems states converge. This technique could be applied both in parallel as well as in a sequential manner.

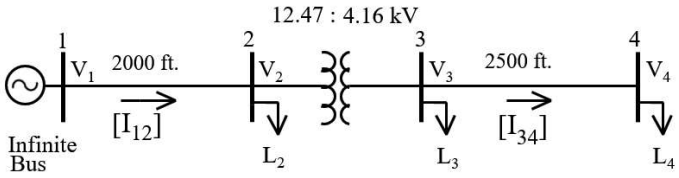

Fig. 1. Single line diagram of the modified IEEE 4 Bus test feeder [11].

In this study a modified IEEE 4 bus case [11], shown in Fig. 1, is used to demonstrate the approach. This is the simplest possible unbalanced feeder that can be analyzed and therefore a good choice to set up the proof of concept. The loads at Bus 2, 3 and 4 are 1600 kW, 1800 kW and 2000 kW at 0.9 power factor respectively (balanced loading case considered). The rated voltage for Bus 1 and 2 (primary) is 12.47 kV and for Bus 3 and 4 (secondary) is 4.16 kV. The transformer configuration is grounded wye - grounded wye. The input variables are the loads at Bus 2, 3 and 4: L_2, L_3, L_4. The output variables are the Bus voltages at Bus 2, 3 and 4: V_2 ,V_3 ,V_4. Each of these variables are made up of six primary units, since each variable has three phases and each of these phase quantities has a real and an imaginary component. The data for training, as well as validation, was generated using the OpenDSS simulator [12].

The traditional approach is to learn the system as one single network. In this approach there will be one large computation unit to represent the full network as shown in Fig. 2.

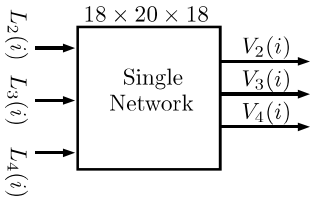

Fig. 2. Computation engine using a single neural network.

B. CCN Framework

The basic building block of the learning system in CCN is the cell. In this study the inputs to these cells are the voltages and the loads. The voltage and the loads are processed in the Cartesian form (real and imaginary values separately). Since the system under study is a three phase system, both loads and voltages have three dimensions corresponding to phases A, B and C.

CCN is a new technology that was first introduced in [4], and then developed into a complete framework in [5]. In simple terms the CCN modeling framework can be explained as mimicking the physical topology to develop the distributed model of the system. The distributed model for Fig. 1 that was derived using the CCN framework is shown in Fig. 3. The computational cell for this network is chosen as a fully connected feed-forward neural network (multi-layer perceptron) with one hidden layer consisting of ten sigmoid activated neurons. The number of inputs to each cell depends on the CCN connections, whereas the output is the calculated bus voltage. The neural networks that make up the cells, with inputs and outputs, are shown in Fig. 4.

C. Training CCN Cells

The cells are trained by applying time varying loads to Bus 2, 3 and 4, which are different in shape and characteristics. A large input data set of 1 million points was generated by choosing random input load values in the interval of $[L_{min}, L_{max}]$. Here, the minimum load, L_{min}, is 0, and, maximum load, L_{max}, is the rated load value. This input data set was used as the time-series input to the physical model of the system that was set-up in OpenDSS [12] to generate the corresponding output data set. Out of the full data set,

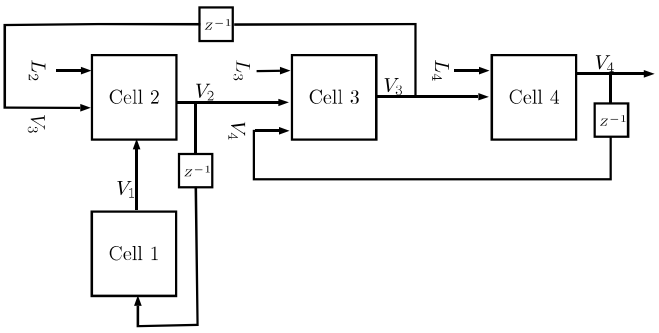

Fig. 3. Overview of the Cellular Computational Network.

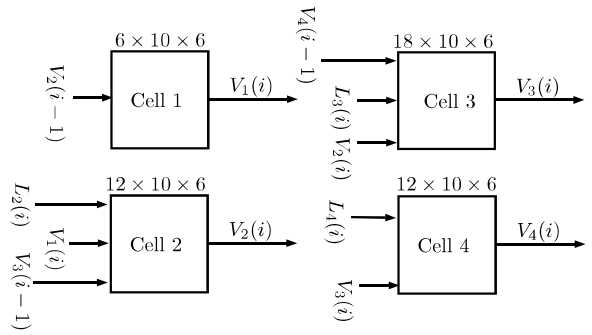

Fig. 4. Structure of individual computational cells.

70% of the data was used for training, 15% was used for validation and the rest was used for testing. The Levenberg-Marquardt algorithm [13], [14], which uses an approximation to Newton's method, was used for training the neural networks. This algorithm uses the sum of squares error function, $V(\mathbf{w})$, given in (6). Here e_i is the output error for every input pattern and w the weight vector.

$$V(\mathbf{w}) = \sum_{i=1}^{N} e_i^2(\mathbf{w}) \tag{6}$$

The weight update equation is:

$$\Delta\mathbf{w} = [J^T(\mathbf{w})J(\mathbf{w}) + \mu I]^{-1} J(\mathbf{w})e(\mathbf{w}) \tag{7}$$

Where J is the Jacobian matrix of the weights.

D. CCN Power Flow Engine

Once the cells are trained, they are connected based on the CCN derivation that was shown in Fig. 3. The connected CCN network takes the form shown in Fig. 5. After the cells are connected, the bus voltages are calculated based on the algorithm shown in Fig. 6.

The first step in the calculation is to initialize the voltages to 1 per unit (pu), since otherwise there is no estimation available for the first iteration. The choice for using 1 pu is based on the fact that any realistic power system Bus voltage will be close to 1 pu. The second step is to calculate the voltages of Bus 2, 3 and 4, in that order. The V_3 calculated by cell 2 serves as an input to cell 2 and the V_4 calculated by cell 3 serves as an input to cell 4. The stop condition for iterations is based on the calculated difference in voltage magnitude in all voltages between two consecutive iterations. When each and every value in this set is smaller than a specified ϵ, the iterations will stop.

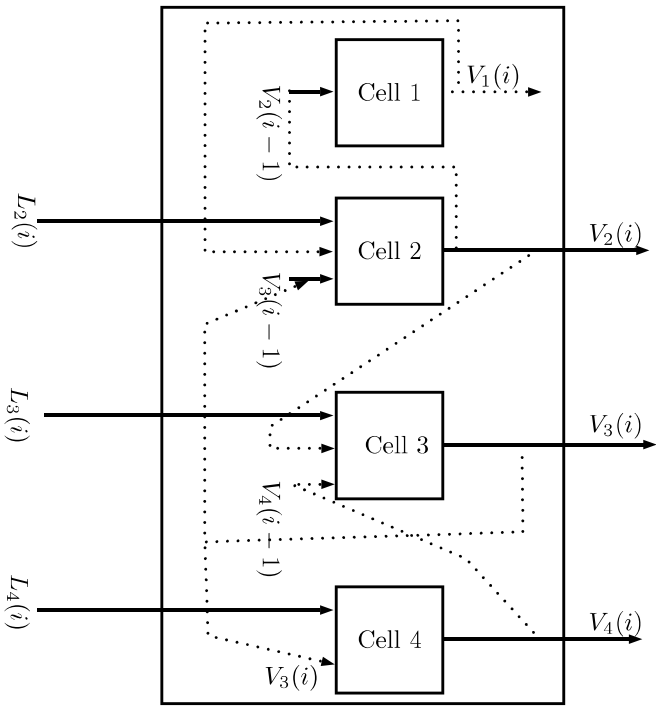

Fig. 5. CCN Power flow inference engine.

Note that the only one initial value for Bus 2 is required to start the iterations in this method. The full initialization is only required if a parallel approach is used.

IV. Results and Discussion

A. Results

The performance of the CCN inference engine is compared with the MLP inference engine by applying the balanced random load profile shown in Fig. 7 to the three loads. To ensure a fair comparison, the number of hidden layer neurons in the MLP implementation is set to 20. As a result both methods have an equal 360 tuned weights.

Voltage magnitude and APE of Bus 2 phase A is compared in Figs. 8 and 9. Since this is closest to the infinite Bus (Bus 1), the voltages are very close to 1 pu and shows minimal variation from 1 pu. The CCN network accuracy compares with the MLP results. Note that APE is calculated using (8). Here, $V_{k,estimate}^{Ph}$ is the estimated voltage magnitude of phase Ph of Bus k and V_k^{Ph} is the actual voltage calculated through OpenDSS.

$$APE = 100 \times |(V_{k,estimate}^{Ph} - V_k^{Ph})/V_k^{Ph}| \tag{8}$$

Voltage magnitude and APE of Bus 3 phase A are compared in Figs. 10 and 11. Whereas the APE has increased for both

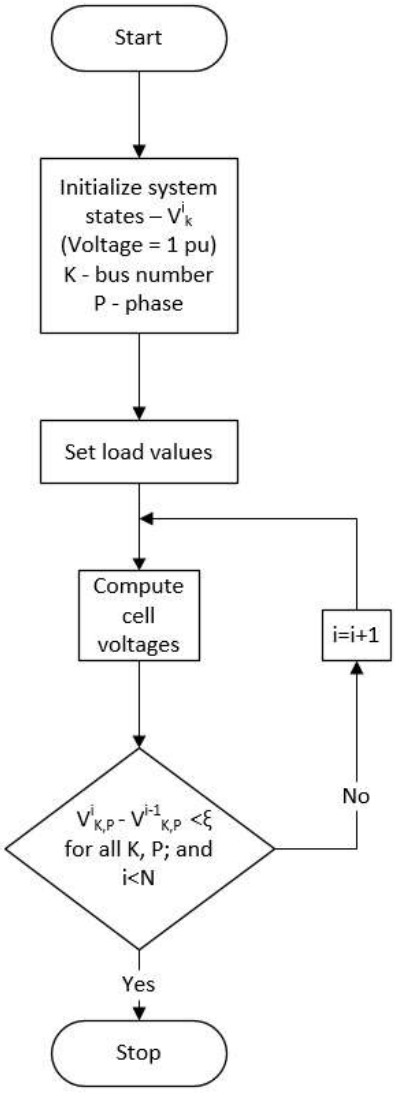

Fig. 6. CCN power flow inference flowchart.

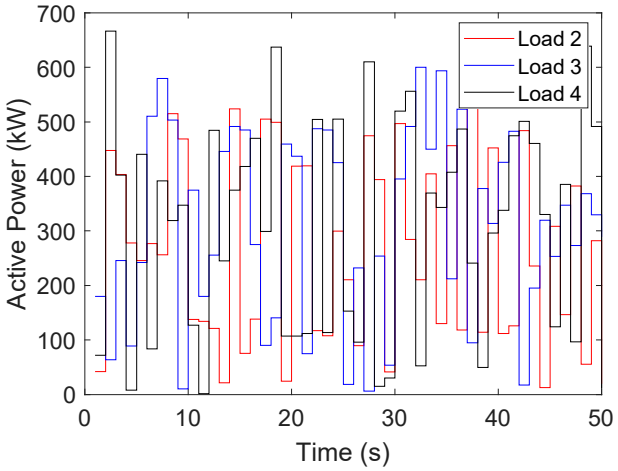

Fig. 7. Load profile (phase A) used to evaluate performance.

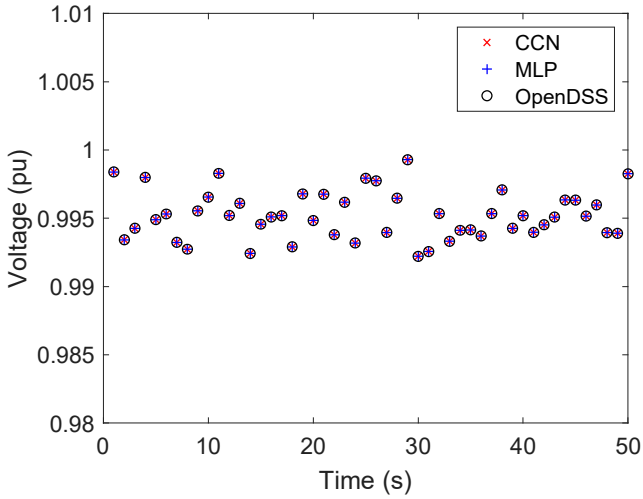

Fig. 8. Voltage magnitude (Bus 2 phase A).

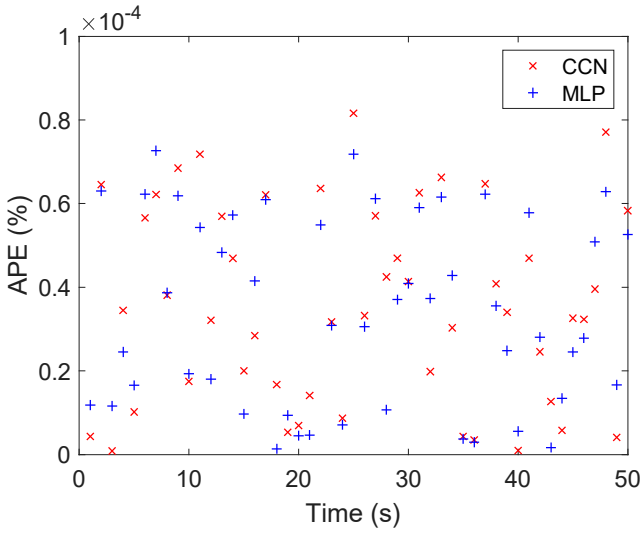

Fig. 9. Absolute Percentage Error of the voltage magnitude of Bus 2 phase A.

MLP and CCN, the resulting APE is very low and the error magnitude for the two methods are comparable.

From the set of buses Bus 4 is the farthest away from the infinite bus, and as a result it is the most challenging bus to infer voltage. Therefore, this bus is analyzed in more detail.

Figs. 12 and 13 shows that errors has increased in the estimation. These plots also show that for Bus 4 phase A, CCN accuracy is consistently comparable with MLP accuracy. Additionally, the variation of error follows a similar pattern across the two different methods. Figs. 14, 15, 16 and 17 shows that a similar pattern holds for phase B as well as phase C of Bus 4.

Fig. 18 compares the phase angle inferred from MLP and CCN with the reference generated by OpenDSS for Bus 4 phase A while Fig. 19 compares the inference errors of CCN with MLP. Fig. 20 compares the phase angle inferred from

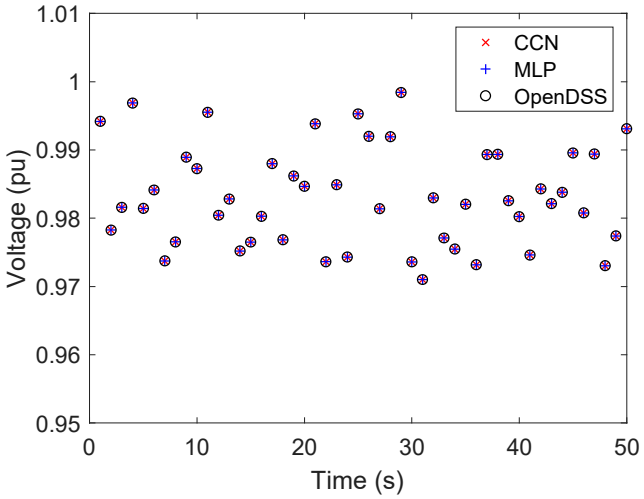

Fig. 10. Voltage magnitude (Bus 3 phase A).

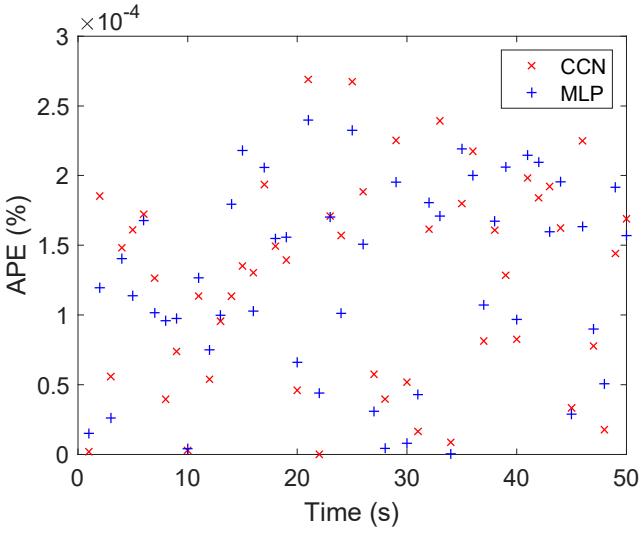

Fig. 11. Absolute Percentage Error of the Bus 3 phase A voltage magnitude.

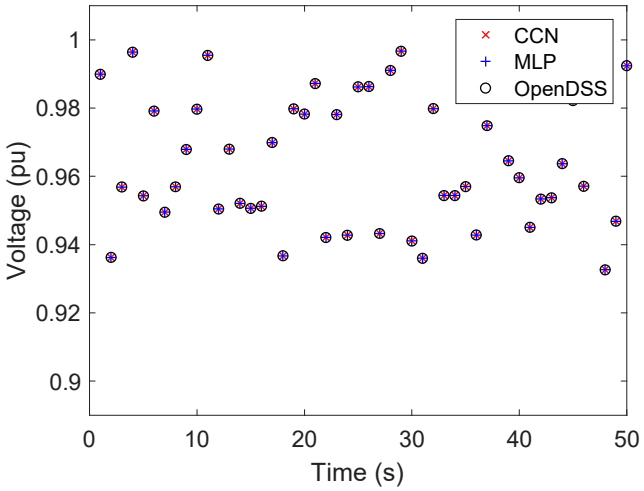

Fig. 12. Voltage magnitude (Bus 4 phase A).

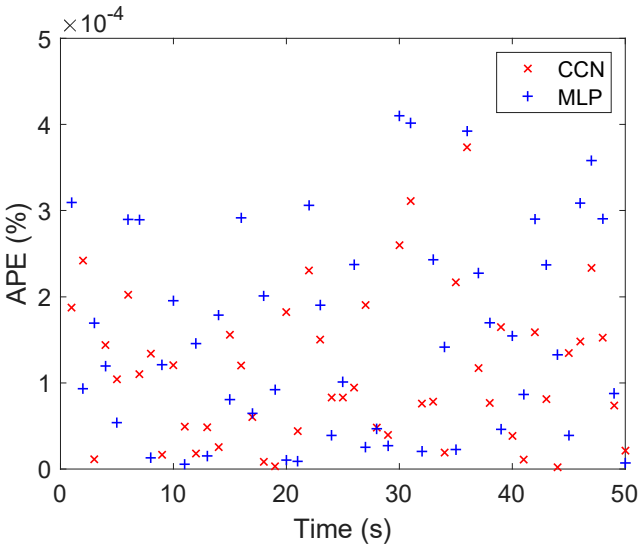

Fig. 13. Absolute Percentage Error of the Bus 4 phase A voltage magnitude.

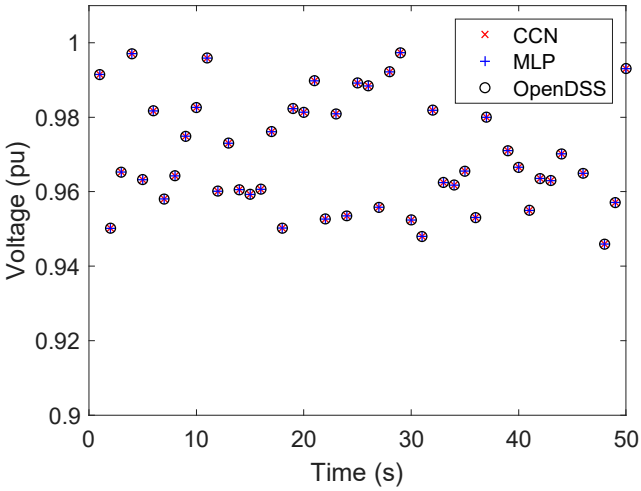

Fig. 14. Voltage magnitude (Bus 4 phase B).

MLP and CCN with the reference generated by OpenDSS for Bus 4 phase B. Fig. 21 compares the inference errors of CCN with MLP. Fig. 22 compares the phase angle inferred from MLP and CCN with the reference generated by OpenDSS for Bus 4 phase C. Fig. 23 compares the inference errors of CCN with MLP.

The absolute percentage error for Bus 4 voltage is comparatively high in comparison to Bus 2 and 3. However, across all phases, it is evident that CCN has a better performance in comparison to MLP with respect to both voltage magnitude and angle.

In order to gauge overall performance of the two different approaches, the mean of the errors over the 50 second result set is calculated and resulting values for MLP are compared with CCN. The results looking at both inferenced voltage magnitude as well as voltage angle, are given in Table I.

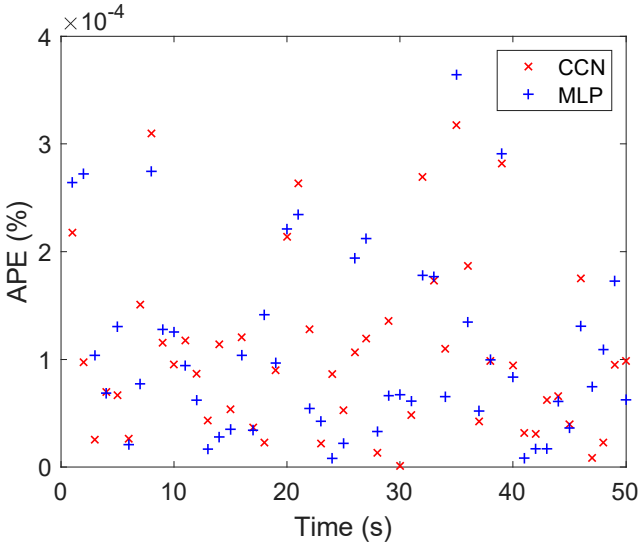

Fig. 15. Absolute Percentage Error of the Bus 4 phase B voltage magnitude.

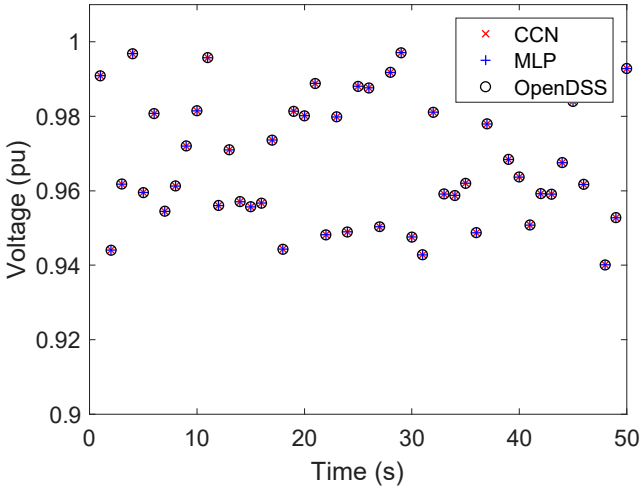

Fig. 16. Voltage magnitude (Bus 4 phase C).

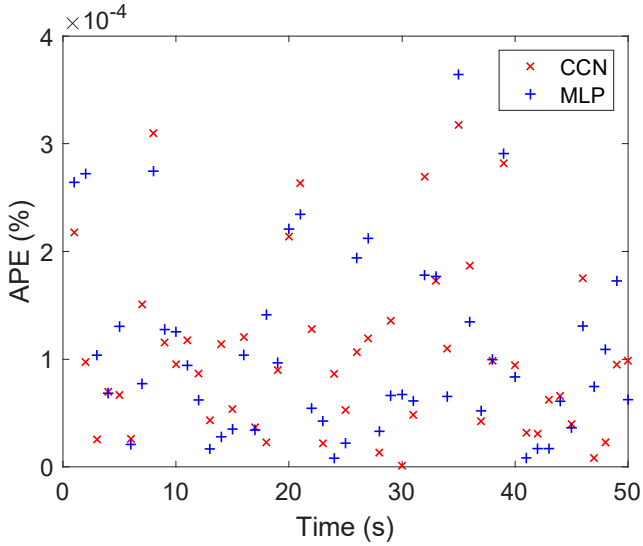

Fig. 17. Absolute Percentage Error of the Bus 4 phase C voltage magnitude.

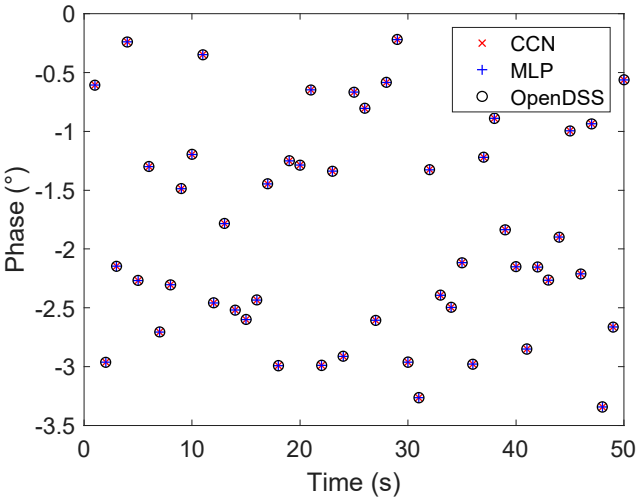

Fig. 18. Phase Angle (Bus 4 phase A angle).

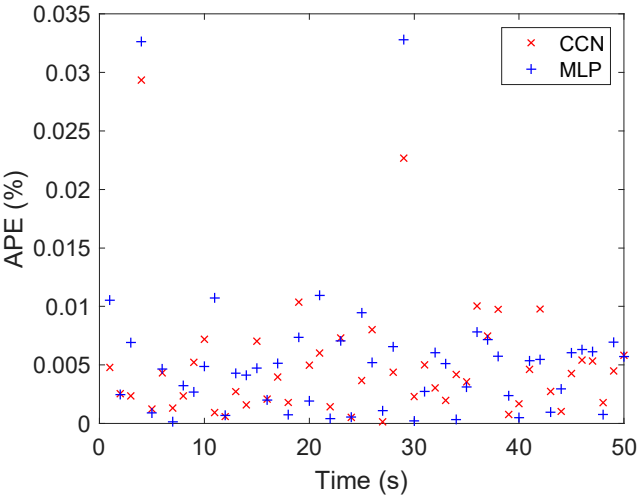

Fig. 19. Absolute Percentage Error of Bus 4 phase A Phase Angle.

B. Discussion

The mean absolute percentage error of the voltage magnitude as well as the mean absolute error of the voltage angle has increased when moving from Bus 2 to Bus 4. The results show that both CCN and MLP perform at a very high accuracy. Both methods accurately estimate the system output and the observed error values are negligible in the context of a real application.

The errors of both methods have increased for the tested data set with increase in distance to the infinite bus. Even though the network is unbalanced, the results show similar voltages in the three phases since the network was designed assuming balanced loads.

The CCN method is a distributed and scalable approach. The training is faster and requires minimum computational resources, yet shows comparable performance in accuracy to the MLP network. The CCN approach does require a

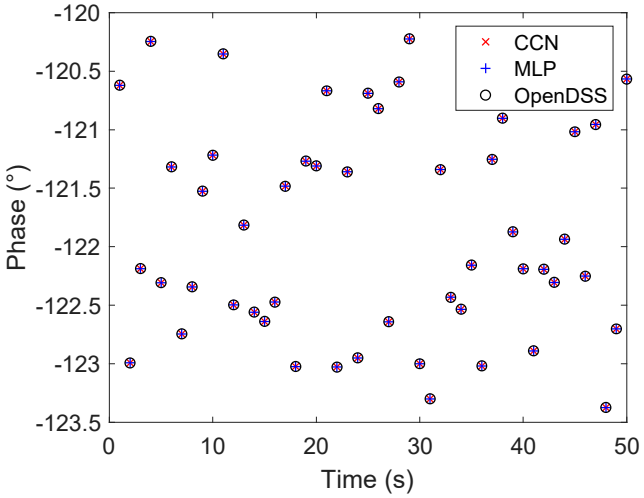

Fig. 20. Phase Angle (Bus 4 phase B angle).

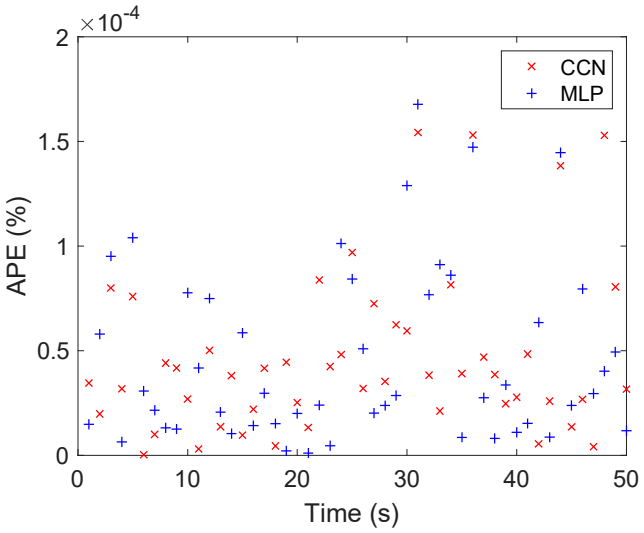

Fig. 21. Absolute Percentage Error of Bus 4 phase B Phase Angle.

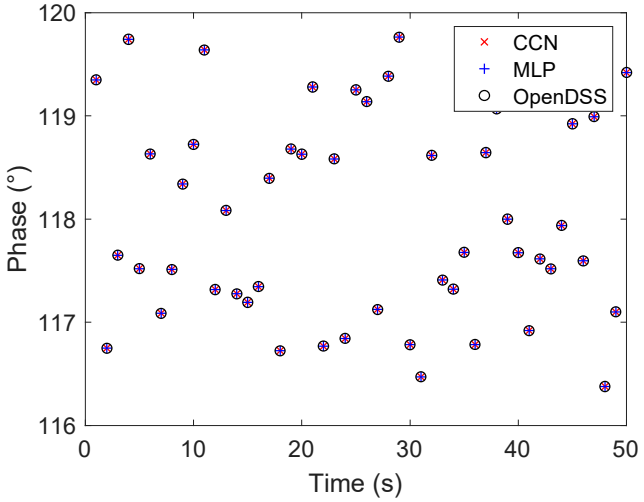

Fig. 22. Phase Angle (Bus 4 phase C angle).

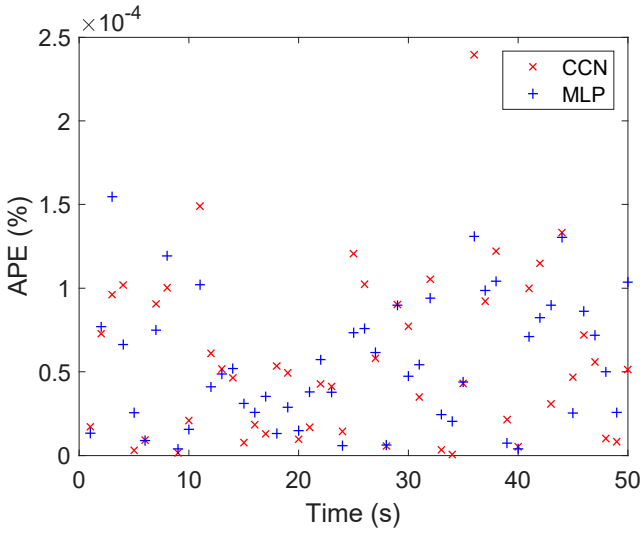

Fig. 23. Absolute Percentage Error of Bus 4 phase C Phase Angle.

TABLE I
COMPARISON OF THE AVERAGE MEAN ABSOLUTE PERCENTAGE ERROR (MAPE) OF THE VOLTAGE MAGNITUDE (V) AND VOLTAGE PHASE ANGLE (PH).

State	CCN	MLP
2 V A	$(3.5 \pm 2.3) \times 10^{-5}$	$(3.4 \pm 2.2) \times 10^{-5}$
2 V B	$(3.2 \pm 2.4) \times 10^{-5}$	$(3.1 \pm 2.5) \times 10^{-5}$
2 V C	$(3.9 \pm 2.2) \times 10^{-5}$	$(4.0 \pm 2.3) \times 10^{-5}$
2 Ph A	$(4.4 \pm 6.5) \times 10^{-3}$	$(5.4 \pm 8.2) \times 10^{-3}$
2 Ph B	$(1.7 \pm 1.0) \times 10^{-5}$	$(1.8 \pm 1.0) \times 10^{-5}$
2 Ph C	$(1.5 \pm 2.0) \times 10^{-5}$	$(1.7 \pm 1.1) \times 10^{-5}$
3 V A	$(1.2 \pm 0.7) \times 10^{-4}$	$(1.3 \pm 0.7) \times 10^{-4}$
3 V B	$(9.6 \pm 6.0) \times 10^{-5}$	$(1.0 \pm 0.7) \times 10^{-4}$
3 V C	$(9.8 \pm 7) \times 10^{-5}$	$(1.0 \pm 0.7) \times 10^{-4}$
3 Ph A	$(3.4 \pm 3.0) \times 10^{-3}$	$(2.0 \pm 3.0) \times 10^{-3}$
3 Ph B	$(5.5 \pm 3.4) \times 10^{-5}$	$(5.3 \pm 3.8) \times 10^{-5}$
3 Ph C	$(6.1 \pm 4.2) \times 10^{-5}$	$(5.7 \pm 3.8) \times 10^{-5}$
4 V A	$(1.1 \pm 0.9) \times 10^{-4}$	$(1.6 \pm 1.2) \times 10^{-4}$
4 V B	$(1.1 \pm 0.8) \times 10^{-4}$	$(1.1 \pm 0.9) \times 10^{-4}$
4 V C	$(1.1 \pm 0.70) \times 10^{-4}$	$(1.2 \pm 0.8) \times 10^{-4}$
4 Ph A	$(4.9 \pm 5.1) \times 10^{-3}$	$(5.5 \pm 6.3) \times 10^{-3}$
4 Ph B	$(4.6 \pm 3.9) \times 10^{-5}$	$(4.6 \pm 4.2) \times 10^{-5}$
4 Ph C	$(5.7 \pm 4.9) \times 10^{-5}$	$(5.5 \pm 3.8) \times 10^{-5}$

few iterations to converge, whereas the MLP method directly estimates the load flow solution. However, when the network nodes increase, MLP will be limited by the availability of computational resources.

V. CONCLUSION

This paper presented a new method to distribute the power flow using a data driven approach. The larger power flow problems is broken down into smaller sub-problems using the CCN framework. In a situation where the models are data

driven, it also provides an avenue to efficiently tune the cells in an asynchronous manner.

The model that was used to set up the proof of concept was a very simple radial network. In order to prove the applicability of the method in a general load flow problem, the CCN needs to be applied to a larger network which represents an actual system operating in the real world.

The advantage of using a data driven approach is the possibility of learning from data even in circumstances where physics based models are unavailable. The disadvantage is that a high quality data set is required.

The initial data set used to train the models cost some computational resources. However, The approach used in this study does not require matrix inversion. Therefore, once trained, the model requires minimal computational resources to solve load flow.

The results show that CCN and MLP can accurately identify the power flow model. CCN is more scalable, efficient and can be computed in a distributed manner, which makes it a better approach for application in the future smart grid.

One key extension to this study is to find the applicability of using algebraic connectivity or similar index from the graph theory domain to understand characteristics such as stability and speed of convergence of the CCN.

REFERENCES

[1] H. Happ, "Z diakoptics-torn subdivisions radially attached," *IEEE Transactions on Power Apparatus and Systems*, no. 6, pp. 751–769, 1967.

[2] G. Kron, *Diakoptics: the piecewise solution of large-scale systems*. MacDonald, 1963, vol. 2.

[3] F. M. Uriarte, "On kron's diakoptics," *Electric Power Systems Research*, vol. 88, pp. 146–150, 2012.

[4] L. L. Grant and G. K. Venayagamoorthy, "Cellular multilayer perceptron for prediction of voltages in a power system," in *Intelligent System Applications to Power Systems, 2009. ISAP'09. 15th International Conference on*. IEEE, 2009, pp. 1–6.

[5] B. Luitel and G. K. Venayagamoorthy, "Cellular computational networksa scalable architecture for learning the dynamics of large networked systems," *Neural Networks*, vol. 50, pp. 120–123, 2014.

[6] Y. Wei, I. Jayawardene, G. K. Venayagamoorthy *et al.*, "Frequency prediction of synchronous generators in a multi-machine power system with a photovoltaic plant using a cellular computational network," in *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 2015, pp. 673–678.

[7] Y. Wei and G. K. Venayagamoorthy, "A lite cellular generalized neuron network for frequency prediction of synchronous generators in a multimachine power system," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 3085–3092.

[8] C. Pathiravasam and G. K. Venayagamorthy, "Spatio-temporal characteristics based wind speed predictions," in *Information and Automation for Sustainability (ICIAfS), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[9] L. Wu, G. K. Venayagamoorthy, and R. G. Harley, "Cellular computational networks based voltage contingency ranking regarding power system security," in *Power Systems Conference (PSC), 2018 Clemson University*. IEEE, 2018, pp. 1–6.

[10] J. J. Grainger and W. D. Stevenson, *Power system analysis*. McGraw-Hill, 1994.

[11] W. H. Kersting, "Radial distribution test feeders," *IEEE Transactions on Power Systems*, vol. 6, no. 3, pp. 975–985, 1991.

[12] R. C. Dugan, "Reference guide: The open distribution system simulator (opendss)," *Electric Power Research Institute, Inc*, vol. 7, 2012.

[13] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.

[14] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.