2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)
Virtual, October 26-28, 2020

Probabilistic Power Flow Solution with Graph Convolutional Network

Dawei Wang, Kedi Zheng, Qixin Chen
Dept. Electrical Engineering
Tsinghua University
Beijing, China
qxchen@tsinghua.edu.cn

Gang Luo, Xuan Zhang
Power Dispatch and Control Centre
Guangdong Power Grid Co., Ltd
Guangzhou, China
Luogang@gd.csg.cn

Abstract—**One fundamental issue in power grid is the power flow calculation. Due to the uncertainty in system variables, recent research works often concentrate on the probabilistic power flow (PPF). But traditional algorithms cannot combine high accuracy with fast calculation speed. In this paper, we revisit the probabilistic power flow problem, aiming to boost the computational accuracy leveraging the breakthroughs of deep learning. We propose a quasi Monte-Carlo method based on graph convolutional network (GCN) to calculate distribution characteristics of system power flow, where the key novelty is viewing the power system from a graph perspective. More specifically, the graph structure and features of power system are adopted as input of graph convolutional network. Through construction of the weight adjacency matrix with an impedance matrix, the relationship among nodes is redefined. We apply the convolutional layers on the mixed graph to compute probabilistic power flow. Using IEEE Case 69 data for experiments, our model outperforms existing model-based and data-driven methods.**

Index Terms—**Probilistic Power Flow, Graph Convolutional Network, Deep Learning, Impedance Matrix**

I. Introduction

Power flow calculation is foundational work for the power system [1]. Generally, power flow calculation is only conducted from a system operator's perspective. The network topology, transformer ratios, node power injection and other electrical configurations are all given. In the era of energy internet, both the scale and complexity of power systems are increasing, which make probabilistic power flow (PPF) algorithms attract more attention. In this way, we can obtain the distribution characteristics (e.g., mean, variance and probability density function) of unknown variables from the information of known variables with the theory of probability. There are two conventional PPF solutions: Numerical and sampling methods(i.e. Monte-Carlo (MC) method and its improved method based on Latin hypercube sampling(LHS)), analytical methods and approximate methods [2]. Analytical methods include the cumulant method [3], while approximate methods include the point estimation method (PE) [4] and Parzen windows methods [5], etc. Simulation results of MC are often used as benchmarks. However, MC is not a practical solution due to its high computation consumption in solving numerous power flow equations with the Newton-Raphson method. The cumulant method uses the linearised power flow equations to solve the problem, but such simplification increases the computational error. Meanwhile, it performs even worse with variables not serving Gaussian distribution, which are very common in systems with renewable energy generators like wind turbines.

There has been some attempts to solve power flow problems with data-driven methods in literature. The support vector regression (SVR) is used to reveal the mapping rules between different variables in distribution power grid, and shows a good generalization ability [6]. The power flow equation of data-driven method is linearized by using partial least squares and Bayesian linear regression [7]. Besides, ANN algorithms (e.g., back Propagation Neural Network [8], chaotic neural network [9],Radial Basis Function Neural Network [10], etc.) are also applied to power flow calculation. Reference [11] uses an input-output network without hidden layer to solve power flow, and points out that its network weighting coefficients are formed from Jacobian matrix in Newton-Raphson method. As shown in the aforementioned articles, data-driven attempts in this field are limited to shallow models, which limits their ability in fitting the highly non-linear power flow equations.

Similar to the common non Euclidean structure in social network and communication network, the power system can also be abstracted as a graph composed of nodes and branches. Like the problem in Euclidean data structure, such as image recognition, traditional convolution neural network can be used to achieve good classification results. However it can not deal with the non-Euclidean problem directly. To solve this problem, Joan proposed two networks constructions, one based upon a hierarchical clustering of the domain, and the other based on the spectrum of the graph Laplacian [12]. Defferrard presented a formulation of CNNs in the context of spectral graph theory, greatly reduced the complexity of parameters [13]. Hammond presented a fast Chebyshev polynomial approximation algorithm for computing the transform that avoided the need for complex eigendecomposition [14]. Kipf motivated the choice of convolutional architectures via a localized first-order approximation of spectral graph con-

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0900100, the science and technology project of Guangdong Power Grid(No.036000KK52180031) and Key-Area Research and Development Program of Guangdong Province under Grant No.2019B111109002.

978-1-7281-7100-5/20/$31.00 ©2020 IEEE

volutions [15]. Federico proposed MotifNet, a graph CNN capable of dealing with directed graphs by exploiting local graph motifs [16]. At present, spatial domain analysis and spectral domain analysis are still the two main methods for GCN [17].

Deep learning on graphs and in particular, graph convolutional neural networks, have recently attracted significant attention in the machine learning society. GCN-based methods have been explored in many fields of smart grid, including power outages forecasting [18] and fault location [19]. For high-dimensional irregular domains, the advantages of convolution include two parts: one is that the convolution kernel has fewer parameters, and the computational efficiency is higher than that of fully connected network. The second is that the convolution process is performed around a vertex along the topology, and multi-layer convolution is carried out for each vertex, so that the convolution covers the whole network. This kind of relationship from near to far is similar to the characteristics of topology.

Based on the coupling characteristics of system topology, this paper has the following contributions:

- In accordance with the characteristics of power systems, a data-driven power flow calculation method based on GCN is proposed. Then the GCN-based method is embedded into PPF calculation to achieve fast and accurate results.
- Comprehensive numerical cases based on IEEE 69-bus system are conducted, compared with some existing PPF methods. Different distributions and variable dependencies are tested. The numerical results demonstrate the efficacy of the proposed method.

The rest of this paper is structured as follows. The proposed architecture of GCN is described in Section II. Three numerical cases are designed and presented in Section III. Finally, Section IV draws the conclusion.

II. GCN-based Power Flow Solution

A. Linearised Power Flow Solution

Let the non-linear expressions of power flow equations be like Function (1) and (2), where $\boldsymbol{S}$, $\boldsymbol{V}$, $\boldsymbol{R}$ represent the nodal power injection, node voltage, bi-directional branch power flow respectively.

$$\boldsymbol{S} = g(\boldsymbol{V}), \tag{1}$$

$$\boldsymbol{R} = h(\boldsymbol{V}). \tag{2}$$

At the baseline voltage vector $\boldsymbol{V}_0$, we get the linearised equations with the Taylor series expansion, where J_1 is the final Jacobian matrix, and J_2 is a sensitivity matrix:

$$\boldsymbol{S}_0 + \Delta\boldsymbol{S} = g(\boldsymbol{V}_0 + \Delta\boldsymbol{V}) = g(\boldsymbol{V}_0) + J_1\Delta\boldsymbol{V} + o(\Delta\boldsymbol{V}), \tag{3}$$

$$\boldsymbol{R}_0 + \Delta\boldsymbol{R} = h(\boldsymbol{V}_0 + \Delta\boldsymbol{V}) = h(\boldsymbol{V}_0) + J_2\Delta\boldsymbol{V} + o(\Delta\boldsymbol{V}). \tag{4}$$

Since $\boldsymbol{S}_0 = g(\boldsymbol{V}_0)$ and $\boldsymbol{R}_0 = h(\boldsymbol{V}_0)$, (3) and (4) are transformed as follows ignoring the higher order terms:

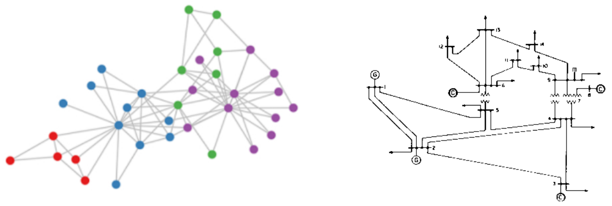
$$\Delta\boldsymbol{V} = J_1^{-1}\Delta\boldsymbol{S}, \tag{5}$$

$$\Delta\boldsymbol{R} = J_2\Delta\boldsymbol{V} = J_2J_1^{-1}\Delta\boldsymbol{S}. \tag{6}$$

The problem in PPF is based on the inherent shortcoming of traditional model-based methods for solving non-linear equations. Although the linearised algorithm (e.g. the cumulant method) is fast, its accuracy is limited on principle. The non-linear algorithm (e.g. Newton-Raphson) needs a lot of time in the process of iterative convergence. The problem to be solved in this paper is using the ANN method for fitting the non-linear power flow equation efficiently, which is important for PPF solution.

B. Fast Spectral Convolutions on Graphs

Drawing on the ideas of GCN in the field of high-dimensional irregular domains, as shown in Figure 1a, the graph topology has the following two basic characteristics: attribute and structure information of nodes. Similarly, in the power system topology, like Figure 1b, nodes with electrical parameters are connected with others by power lines. We can get the undirected graph from the system configurations and introduce spectral convolutions into electrical graph networks.



(a) Non Euclidean Structure (b) Grid Diagram

Fig. 1: The comparison of an ordinary Non Euclidean Structure and the topology of power system.

For simplicity, consider the power grid as a weighted undirected graph. It is formally defined as $\boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}})$, where $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{n\times1}$ is the set of n vertices, and $\boldsymbol{\mathcal{E}} \subseteq \boldsymbol{\mathcal{V}} \times \boldsymbol{\mathcal{V}}$is the set of edges. Moreover, suppose $\boldsymbol{W}$ as the weighted adjacency matrix, so we can associate vertex i and j with a weight $w_{ij} \geq 0$.

In the traditonal signal processing realm, a discrete convolution of two signals $\boldsymbol{g}$ and $\boldsymbol{f}$ relates with fourier transform as follows,

$$\boldsymbol{g} * \boldsymbol{f} = \boldsymbol{\mathcal{F}}^{-1}\{\boldsymbol{\mathcal{F}}\{\boldsymbol{g}\} \cdot \boldsymbol{\mathcal{F}}\{\boldsymbol{f}\}\}$$

Generally, the orthogonal basis of Fourier transform is a set of eigenvectors of Laplacian operator $\boldsymbol{\Delta}$.

$$\boldsymbol{\Delta} e^{2\pi ix\cdot v} = \boldsymbol{\Lambda} e^{2\pi ix\cdot v}, \boldsymbol{\Lambda} = diag\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$$

By analogy with the algorithms above in Euclidean spaces, the basis of Fourier transform on graphs is given by the eigendecomposition of Laplacian operator $\boldsymbol{\Delta}$, which is a positive semi-definite symmetric matrix.

$$\boldsymbol{\Delta} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T$$

Then the Laplacian operator on graphs is needed. Given that the unnormalized graph Laplacian operator can be written in matrix form as $\boldsymbol{\Delta}_u = \boldsymbol{D} - \boldsymbol{W}$, denoting by the degree matrix $\boldsymbol{D} = diag(\sum_{j:j\neq i} w_{ij})$, we further get the normalized operator as

$$\boldsymbol{\Delta} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{\Delta}_u\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{\Delta}\boldsymbol{D}^{-\frac{1}{2}}$$

With the Orthogonal basis $\boldsymbol{\Phi} = (\phi_1, \phi_2, \ldots, \phi_n)$, the graph Fourier transform and its inverse can be expressed as,

$$\begin{cases} \boldsymbol{\mathcal{GF}}\{\boldsymbol{f}\} = \boldsymbol{U}^T\boldsymbol{f} \\ \boldsymbol{\mathcal{IGF}}\{\boldsymbol{f}\} = \boldsymbol{U}\boldsymbol{f} \end{cases}$$

As a result, the convolution on graphs in the spectral domain can be conducted with Fourier transform, expressed as follows, where g can be regarded as a filter of the graph convolution. At the same time, to get good spacial localization on graphs, we define g as a function of $\boldsymbol{\Lambda}$.

$$\boldsymbol{g} * \boldsymbol{f} = \boldsymbol{U}(\boldsymbol{U}^T\boldsymbol{g} \cdot \boldsymbol{U}^T\boldsymbol{f})$$

Considering $\boldsymbol{U}^T g$ as $\boldsymbol{g}_\theta(\boldsymbol{\Lambda})$, a function of eigenvalues of $\boldsymbol{\Delta}$, parametrized by $\boldsymbol{\theta} \in \mathbb{R}^n$, then the convolution on graph can be rewritten as

$$\boldsymbol{g} * \boldsymbol{f} = \boldsymbol{U}\boldsymbol{g}_\theta(\boldsymbol{\Lambda})\boldsymbol{U}^T\boldsymbol{x}$$

Evaluating expensive computation consumption with the multiplication of the eigenvector matrix U and eigendecomposition of Δ prohibitively for large graphs. We consider the approximation of filter function,

$$\boldsymbol{g}_\theta(\boldsymbol{\Lambda}) \approx \sum_{k=0}^{K} \boldsymbol{\theta}_k^{'}\boldsymbol{T}_k(\widetilde{\boldsymbol{\Lambda}})$$

where the rescaled $\widetilde{\boldsymbol{\Lambda}}$ can be represented as $\widetilde{\boldsymbol{\Lambda}} = \frac{2}{\lambda_{max}}\boldsymbol{\Lambda} - \boldsymbol{I_N}, \lambda_{max} = max\boldsymbol{\Lambda}$, $\boldsymbol{\theta}^{'}$ denetes an array of Chebyshev coefficients. And the Chebyshev polynomials $\boldsymbol{T}_k(\widetilde{\boldsymbol{\Lambda}})$ up to order K can be expressed as follows,

$$\begin{cases} \boldsymbol{T}_k(x) = 2x\boldsymbol{T}_{k-1}(x) - \boldsymbol{T}_{k-2}(x) \\ \boldsymbol{T}_1(x) = x \\ \boldsymbol{T}_0(x) = 1 \end{cases}$$

As $\boldsymbol{\Delta}^k = (\boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T)^k = \boldsymbol{\Phi}\boldsymbol{\Lambda}^k\boldsymbol{\Phi}^T$, now we have the the graph convolution expressed with $\widetilde{\boldsymbol{\Delta}} = \frac{2}{\Delta_{max}}\boldsymbol{\Delta} - \boldsymbol{I}_N$,

$$\boldsymbol{g} * \boldsymbol{f} \approx \sum_{k=0}^{K} \boldsymbol{\theta}_{\boldsymbol{k}}^{'}\boldsymbol{T}_{\boldsymbol{k}}(\widetilde{\boldsymbol{\Delta}})$$

Supposing K = 1, we finally simplify the graph convolution operation as follows, where $\widetilde{\boldsymbol{W}} = \boldsymbol{W} + \boldsymbol{I}_N, \widetilde{\boldsymbol{D}} = diag(\sum_{j:j\neq i} \widetilde{w}_{ij})$

$$\begin{aligned} \boldsymbol{g} * \boldsymbol{f} &\approx \boldsymbol{\theta}(\boldsymbol{I}_N + \boldsymbol{\Delta})\boldsymbol{f} & (7) \\ &\approx \boldsymbol{\theta}(\boldsymbol{I}_N + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}})\boldsymbol{f} & (8) \\ &\approx \boldsymbol{\theta}(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}})\boldsymbol{f} & (9) \end{aligned}$$

C. GCN Design for Power Flow

1) Electrical Graph Construction: Noticing the neighbour relationship on electrical nodes, we use KNN algorithm to construct the weight adjacency matrix of the graph according to the impedance matrix of the power grid, as the numerical relationship between mutual impedance z_{ij} and self impedance z_{ii} well reflecting the degree of electrical coupling between node i and j. In general, the smaller weight represents the closer relationship. Nodes with lager mutual impedance are contrarily more sensitive to the reference node. In order to keep their properties consistent, we obtain $\hat{\boldsymbol{Z}} \in \mathbb{R}^{n\times n}, \hat{z}_{ij} = |z_{ij} - z_{ii}|$. Then we get the bandwidth $\boldsymbol{\delta} \in \mathbb{R}^n$ and as follows,

$$\hat{w}_{ij} = e^{-\frac{\hat{z}_{ij}2}{\delta_i{}^2}}, \delta_i = \frac{1}{n}\sum_{j=0}^{n}\hat{z}_{ij}$$

We sort and keep the smallest K_n values in each row of $\hat{\boldsymbol{W}}$ to obtain $\boldsymbol{W} \in \mathbb{R}^{n\times n}$.

$$w_{ij} = \begin{cases} \hat{w}_{ij}, & j \subset \boldsymbol{\mathcal{K}} \\ 0, & otherwise \end{cases}$$

In fact, $\boldsymbol{\Delta}$ is a second order differential operator. So the operation on it is equivalent to the propagation of a neighbor node on the graph. In Figure 2, it's obvious that the influence of nodes on each other is gradually deepened with the increase of power m,

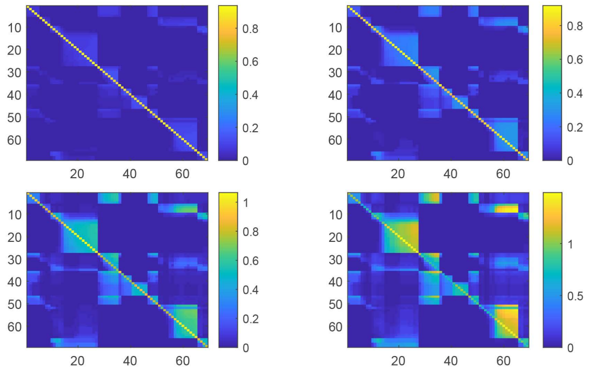


Fig. 2: Visualization of $|\boldsymbol{\Delta}^m|$ (m = 1,2,3,4)

2) Port Design: Our convolutional neural network acting on graph signals aims at fitting the non-linear relationship between system variables (i.e. the equations (1) and (2) listed in subsection A). Then we remark that the input to the first layer are $\boldsymbol{S}$ (i.e., the active injection vector $\boldsymbol{P}$ and the reactive injection vector $\boldsymbol{Q}$). Usually, the last layer is a fully connected layer with output vectors $\boldsymbol{V}$ (i.e., the magnitude ($\boldsymbol{v}_m$) and phase angle ($\boldsymbol{\theta}_v$)) and $\boldsymbol{R}$ (i.e., bi-directional active power $\boldsymbol{P}_f$ and reactive power $\boldsymbol{Q}_f$).

3) Architecture Design: For the whole deep neural network, one forward layer of its graph convolution part can be expressed simply as follows,

$$\boldsymbol{H}^{l+1} = \sigma(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{H}^l\Theta^l) \qquad (10)$$

where $\boldsymbol{H}^l$ is the output of the l_{th} convolution layer, $\boldsymbol{H}^0$ is original input. and $\boldsymbol{\Theta}^l$ denotes the filter parameters that should be trained, $\sigma(\cdot)$ represents the activation function [17].

The GCN model applied to PPF task is illustrated in Figure 3. Besides l_c convolution layers like the above, the data will be fed into l_f fully connected layers after being flattened to an array. leaky ReLU(Rectified Linear Unit) and ReLU are successively used as activation functions in two types of neuron computing operations.

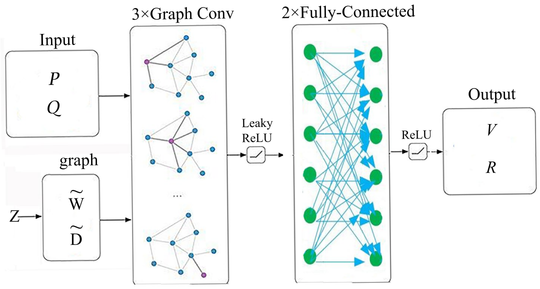


Fig. 3: Graph Convolutional Network Architecture

III. Case Study and Comparison with Other Methods

The proposed method is essentially an improved MC method, utilizing GCN to perform fast power flow calculation. This section contains three parts: in Part A we will compare the performance of accuracy and speed in single power flow calculation; then some evaluation indexes of PPF problem are introduced; Part C shows the performance of GCN-based method on dealing with PPF problem compared with others. For data-driven cases, a total of 800 sets of data are generated, 400 for training and 400 for test. Both of them are designed to be calculated by the baseline of $\boldsymbol{S}$ multiplied with a factor serving uniform distribution over the interval [0.8, 1.2], running on MATPOWER 7.0 platform [21]. The baseline data is from the IEEE 69-bus distribution system. In the training process of GCN, the Adamax optimizer with an initial learning rate of 0.0001 is used for 200 epochs with Pytorch.

A. Case Study of Power Flow Computation

We use GCN for power flow calculation, taking SVR and fully connected neural network(FCNN) as the benchmark. The results are shown as Table I (specifically, the complexity mentioned here refers to the test complexity of the algorithm rather than the training process). As a benchmark, the computing time using Newton-Raphson method for test dataset is 1.05e-2s. During the training process, there are massive repeated calculation to modify network parameters, so the time is much longer than that for the testing. And in the practical application, the training is finished once for all. We only need to use the trained model for the feedforward calculation.

As Figure 4 shows, different colors are stacked without covering each other in a broken line stacked graph. It presents the average value of calculation results ($\boldsymbol{\theta}_v$ and $\boldsymbol{v}_m$) on each bus using different methods. It indicates that the GCN model proposed fits the nonlinear power flow equation well, with high calculative efficiency [22].

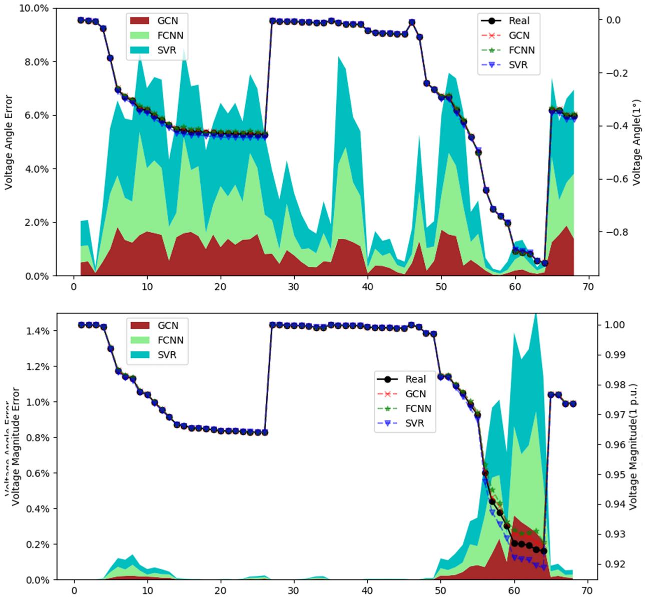


Fig. 4: Calculation results of $\boldsymbol{\theta}_v$ and $\boldsymbol{v}_m$ of each bus

TABLE I: Comparation of Power Flow Computation

Network Type	SVR	FCNN	GCN
Algorithm Complexity	$O(m^2 \cdot n)$	$O(n^2)$	$O(n)$
Parameter Amount	——	20115638	613254
Traning Time	43s	241s	167s
Test Time	3.1e-2s	2.0e-3s	9.2e-4s
Error($\boldsymbol{\theta}_v$)	1.9%	1.6%	0.8%
Error($\boldsymbol{v}_m$)	8.1e-4	6.7e-4	4.5e-4

m is the feature dimension.

B. Evaluation Index of PPF Problem

There are two kinds of metrics adopted, so as to provide a thorough evaluation of different methods:

- Moment estimation error metrics:

$$\varepsilon_\mu^x = \left| \frac{\mu_{MC}^x - \mu_{others}^x}{\mu_{MC}^x} \right| \times 100[\%], \tag{11}$$

$$\varepsilon_\sigma^x = \left| \frac{\sigma_{MC}^x - \sigma_{others}^x}{\sigma_{MC}^x} \right| \times 100[\%]. \tag{12}$$

In (11) and (12), μ_{MC}^x and σ_{MC}^x denotes the x-order origin moment and central moment of a variable in PPF calculation results. The subscripts indicate corresponding algorithms. Generally x is taken to be 2, and the index can be understood as standard deviation accuracy (second-order central moment estimation error, SDA).

- Average Root Mean Square Error (ARMSE)

$$\text{ARMSE} = \frac{\sqrt{\sum_{i=1}^{N_{point}} (OM_i - MC_i)^2}}{N_{point}} \tag{13}$$

In (13), MC_i (i.e. Monte-Carlo) and OM_i (i.e. other methods) are the values of the i-th point of the probability density function (PDF) curve, and N_{point} is the points quantity of the PDF curve.

C. Case Study of PPF Computation

In order to get a practical conclusion, the new test dataset is designed as all nodes fitting norm distribution added with 10% Gaussian white noise. Consider the influence of new energy resources like Wind farms, we replace a generation unit serving a two-parameter Weibull distribution. The scale parameter of Weibull distribution is set to C = 0.9 and the shape parameter is set to k = 1.1. Results obtained from the Monte-Carlo simulation with 10,000 deterministic power flow calculations are the common benchmark. As traditional methods, cumulant method will be compared with the proposed approach. As an improved quasi-MC method, we utilize Latin hypercube sampling for dataset reduction [20].

TABLE II: Comparation of algorithm performance in Case 1

(a) Result with a Generator Serving Weibull Distribution

algorithm	Cumulant	GCN-LHS
time/s	0.16	1.8
SDA/%	35	0.14
ARMSE/%	43	3.1

The benchmark method takes time of 144.2s, LHS with N-R method takes time of 6.1s.

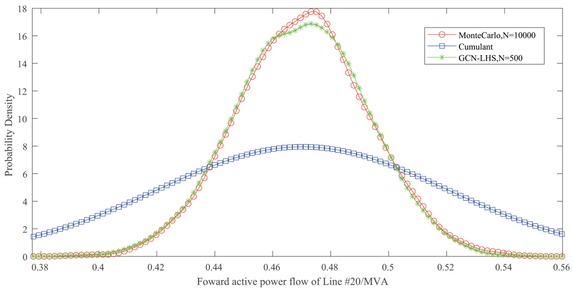


Fig. 5: Local View of Calculative PDF Curve

We visualize the PDF curves with different methods shown in Figure 5. More details about the evaluation index are listed in Table II. The best performing algorithm, both in terms of SDA and ARMSE is the GCN-based method. We observe an improvement of 70% with respect to the cumulant method.

IV. Conclusion

In this paper, we propose a data-driven approach based on GCN as a new quasi-MC solution to the PPF problem. There are two novel aspects of our method. The first part is GCN framework utilizing no prior electrical knowledge but the topology of power grid. The second part is reduce computing time greatly with the high accuracy, compared classical MC simulation method or improved method based on LHS.

The proposed method can be extensively applied in power system planning, operation, and dispatch. As a pioneering effort to employ GCN on PPF, there are still many issues to investigate in the coming months, including the construction of weight adjacency matrix with other model-based manners.

References

[1] X.-F. Wang, Y. Song, and M. Irving, Modern power system analysis. Springer: Springer, 2008.

[2] B. R. Prusty and D. Jena, “A critical review on probabilistic load flow studies in uncertainty constrained power systems with photovoltaic generation and a new approach,” Renewable and Sustainable Energy Reviews, vol. 69, pp. 1286–1302, 2017.

[3] L. A. Sanabria and T. S. Dillon, “Stochastic power flow using cumulants and vonmises functions,” International Journal of Electrical Power & Energy Systems, vol. 8, no. 1, pp. 47–60, 1986.

[4] J. M. Morales and J. Perez-Ruiz, “Point estimate schemes to solve the probabilistic power flow,” IEEE Transactions on Power Systems, vol. 22, no. 4, pp. 1594–1601, 2007.

[5] M. Rouhani, M. Mohammadi, and A. Kargarian, “Parzen window density estimatorbased probabilistic power flow with correlated uncertainties,” IEEE Transactions on Sustainable Energy, vol. 7, no. 3, pp. 1170–1181, 2016. [Online].

[6] Y. Jiafan, W. Yang, and R. Rajagopal, “Robust mapping rule estimation for power flow analysis in distribution grids,” 2017 North American Power Symposium (NAPS), pp. 6 pp.–6 pp., 2017.

[7] L. Yuxiao, Z. Ning, W. Yi, Y. Jingwei, and K. Chongqing, “Data-driven power flow linearization: a regression approach,” IEEE Transactions on Smart Grid, vol. 10, no. 3, pp. 2569–80, 2019.

[8] N. Kumar, R. Wangneo, P. K. Kalra, and S. C. Srivastava, “Application of artificial neural networks to load flow solutions,” in TENCON ’91. Region 10 International Conference on EC3-Energy, Computer, Communication and Control Systems, vol. 1, pp. 199–203, 1991.

[9] M. S. Cai, R. X. Chen, L. S. Kong, and S. O. C. IEEE Computer, Hyper-chaotic neural network based on Newton iterative method and its application in solving load flow equations ofpower system. 2009 International Conference on Measuring Technology and Mechatronics Automation, Vol Iii, Los Alamitos: IEEE Computer Soc, 2009.

[10] A. Karami and M. S. Mohammadi, “Radial basis function neural network for power system load-flow,” International Journal of Electrical Power & Energy Systems, vol. 30, no. 1, pp. 60–66, 2008.

[11] T. T. Nguyen, “Neural-network load-flow,” IEEE Proceedings-Generation Transmission and Distribution, vol. 142, no. 1, pp. 51–58, 1995.

[12] J. B. LeCun, W. Zaremba, A. Szlam, and Yann, “Spectral networks and locally connected networks on graphs,” CoRR, vol. abs/1312.6203, 2013.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering, vol. 29 of Advances in Neural Information Processing Systems. 2016.

[14] D. K. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” vol. 30, no. 2, pp. 129–150, 2011.

[15] T. N. Kipf and M.Welling, “Semi-supervised classification with graph convolutional networks arxiv,” arXiv, pp. 10 pp.–10 pp., 2016.

[16] F. Monti, K. Otness, and M. M. Bronstein, “Motifnet: A motif-based graph convolutional network for directed graphs,” in 2018 IEEE Data Science Workshop (DSW), pp. 225–228, June 2018.

[17] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, “Geometric deep learning: going beyond euclidean data arxiv,” arXiv, pp. 20 pp.–20 pp., 2016.

[18] D. Owerko, F. Gama, and A. Ribeiro, “Predicting power outages using graph neural networks,” pp. 743–747, 11 2018.

[19] C. Kunjin, H. Jun, Z. Yu, Y. Zhanqing, and H. Jinliang, “Fault location in power distribution systems via deep graph convolutional networks arxiv,” arXiv, pp. 8 pp.– 8 pp., 2018.

[20] H. Yu, C. Y. Chung, K. P. Yong, H. W. Lee, and J. H. Zhang, “Probabilistic load flow evaluation with hybrid latin hypercube-sampling and cholesky decomposition,” IEEE Transactions on Power Systems, vol. 24, no. 2, pp. 661–667, 2009.

[21] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, “Matpower: steadystate operations, planning, and analysis tools for power systems research and education,” IEEE Transactions on Power Systems, vol. 26, no. 1, pp. 12–19, 2011.

[22] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016. http: //www.deeplearningbook.org.