Physics-Informed Closed-form Twinning of Power Flow

Putting Network Graph into Gaussian Processes

Parikshit Pareek
Post-Doctoral Research Associate
Theoretical Division (T-5), ANSI
Los Alamos National Laboratory (LANL)

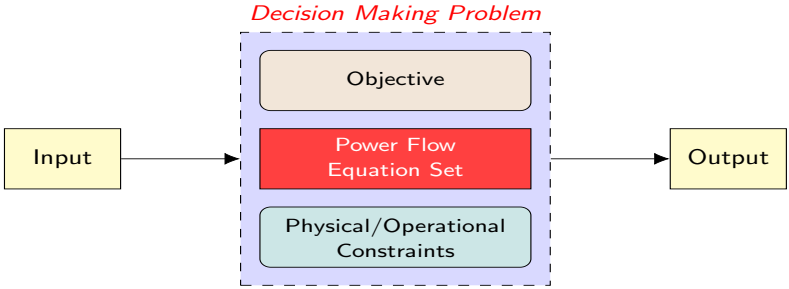
VDK-GP: Deepjyoti Deka & Sidhant Misra @LANL
CFPF: Hung D. Nguyen @NTU Singapore

Sixth Workshop on Autonomous Energy Systems. NREL
LA-UR-23-29848

Power System Learning

Motivation

– Interpretations about Causalities via 'Mimicking' physics
– Implanting Learning into Optimization models $\Longrightarrow$ Closed-Form



A Closed-form Power Flow (CFPF) Framework for Power Balance Equations

Motivation for Physics-Informed Learning for Power Flow

$$\texttt{Power Flow:} \implies \overset{\text{Net Power Injection}}{S_i} = \sum_{j \in \mathcal{N}} \overset{\text{Network Parameter}}{Y_{ij}^{\dagger}} (v_i v_i^{\dagger} - \overset{\text{Complex Voltage}}{v_i} v_j^{\dagger})$$

Power flow equation set allows to obtain the complex voltage values at each network node, given the power injection at each node.

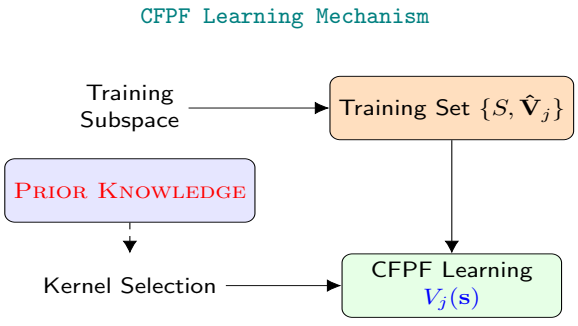
Motivation

A Closed-form Power Flow Approximation Framework which gives

- `Flexible Forms` $\implies$ Non-linear forms with *complexity-accuracy* trade-off
- `Easy to Evaluate Forms` $\implies$ Faster numerical calculations
- `Non-parametric Forms` $\implies$ Works within a power injection range or hypercube
- `Differentiable Forms` $\implies$ Can be fed into optimization problems
- `Interpretable Forms` $\implies$ Should provide insights into physical system

Essentially an explicit expression of voltage as a function of power injection is needed

Power Flow Approximation: Analytical Form



Mean : $\mathbb{E}[f(\mathbf{s})] = V_j(\mathbf{s}) = \mathbf{k}^T[K + \sigma_\epsilon^2 I]^{-1}\hat{\mathbf{V}}_j$

Variance : $\sigma^2[f(\mathbf{s})] = k(\mathbf{s}, \mathbf{s}) - \mathbf{k}^T[K + \sigma_\epsilon^2 I]^{-1}\mathbf{k}$

Training Data– $\{S, \hat{\mathbf{V}}_j$; N Samples Design Matrix– $S = [\mathbf{s}^1 \dots \mathbf{s}^i \dots \mathbf{s}^N]$

– $\mathbf{s}^i$ is i-th power injection vector

Target Vector– $\hat{\mathbf{V}}_j = [V_j^1 \dots V_j^N]$ Power Flow as a Function:

$$\hat{V}_j = f(\mathbf{s}) + \epsilon$$

Gaussian Process (GP) function view [1]

$$f(\mathbf{s}^i) \sim \mathcal{GP}\big(\underbrace{0}_{\text{Zero Mean}}, \underbrace{k}_{\text{Kernel Function}}(\mathbf{s}^i, \mathbf{s}^j)\big)$$

Learn the kernel hyper-parameters using maximum log-likelihood

CFPF provides mean prediction of voltage and confidence in that prediction

Training Data

$$V_j(\mathbf{s}) = \begin{bmatrix} k(\mathbf{s}^1, \mathbf{s}) & \dots & k(\mathbf{s}^N, \mathbf{s}) \end{bmatrix} \boldsymbol{\alpha}_j$$

Variable $\mathbf{s} = [\mathbf{p}; \mathbf{q}]$ — Constant

How Do Forms Look Then?

Simple, Standard Forms

– *Linear:* $\mathbf{v} = A\mathbf{s} + b$ – *Quadratic:* $V_j(\mathbf{s}) = \mathbf{s}^T M \mathbf{s} + \mathbf{m}^T \mathbf{s} + r$

More Complex but Accurate Forms

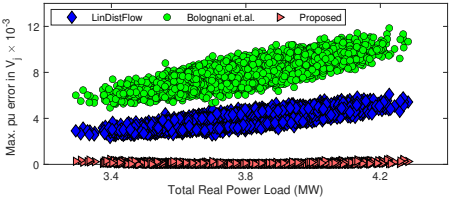
$$V_j(\mathbf{s}) = \sum_{i=1}^{N} \alpha_j(i) \beta^i$$

where, $\beta^i = \tau^2 \exp\left(-\|\mathbf{s}^i - \mathbf{s}\|^2 / 2\ell^2\right)$: Gaussian Kernel

P. Pareek and H. D. Nguyen "A Framework for Analytical Power Flow Solution using Gaussian Process Learning", *IEEE Trans. on Sustainable Energy*, Vol.13 (1), Jan 2022.

CFPF: Features (& Reasons of using Gaussian Process)

- Subspace-wise Approximations & Non-Parametric



Error Not a Function of Load

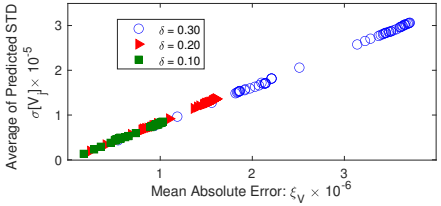
For 69-Bus System

Non-parametric Nature

Distribution	Max. MAE (pu)
Normal	1.86E-05
Beta	2.39E-05
Laplace	2.29E-05
Mixed	1.74E-05

For 33-Bus System

- Inherent Accuracy Indicator



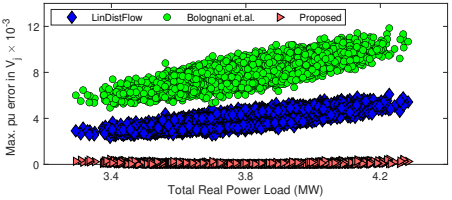
Use of Predictive Variance

For 56-Bus System

- Differentiable Functions
- Faster Numerical Evaluations

CFPF: Features (& Reasons of using Gaussian Process)

- Subspace-wise Approximations & Non-Parametric



Error Not a Function of Load

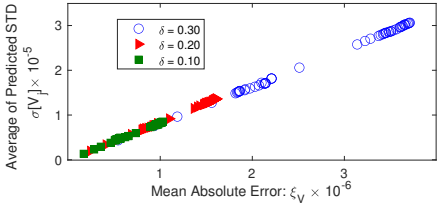
For 69-Bus System

Non-parametric Nature

Distribution	Max. MAE (pu)
Normal	1.86E-05
Beta	2.39E-05
Laplace	2.29E-05
Mixed	1.74E-05

For 33-Bus System

- Inherent Accuracy Indicator



Use of Predictive Variance

For 56-Bus System

- Differentiable Functions
- Faster Numerical Evaluations
- Model Interpretability via *Hyper-parameters*
- Independent from Network Type Assumptions

Curse-of-Dimensionality

- Exact inference has complexity N^3 with N training samples
- Sample requirement tends to grows exponentially with size of system
- So, with all loads varying, we cannot apply exact inference

Mesh Network Flows

- Not so rigid patters as in radial with single source
- Injection-voltage relationship is not direct
- Chances of non-linearity beyond quadratic is higher

Limited Explainability and Use

- Relative effect of node or cluster of nodes is hard to analyze
- Full GP is restrictive in use within Bayesian Optimization

An Ideal Situation Would be

A high-dimensional voltage function is breakable into low-dimensional sub-functions

Learn individual low-dimensional sub-function and combine
- Lower Curse-of-Dimensionality

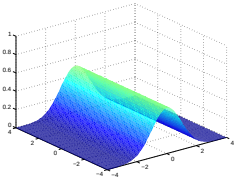
Capturing localized voltage-injection relationship will be easier
- Suitable for Mesh Network Flows

Better understanding of impact of a smaller set of injections a voltage
- Improved Explainability & Interpretability

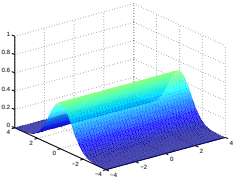
Low-dimensional GPs are perfect fit for active learning
- Useful as surrogate in Bayesian optimization approach

$$
\begin{aligned}
V_j(\mathbf{s}) &= V_j^1(\mathbf{s}_1) + \cdots + V_j^m(\mathbf{s}_m) \\
&\equiv \mathcal{GP}_1(0, K_1(\mathbf{s}_1, \cdot)) + \cdots + \mathcal{GP}_m(0, K_m(\mathbf{s}_m, \cdot)) \\
&\equiv \texttt{Additive Gaussian Process}
\end{aligned}
$$

Additive Gaussian Process

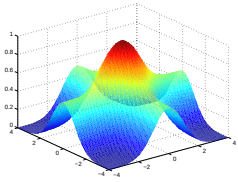


$k_1(x_1, x_1')$
1-D Kernel



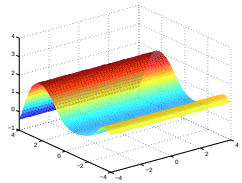
$k_2(x_2, x_2')$
1-D Kernel

$=$

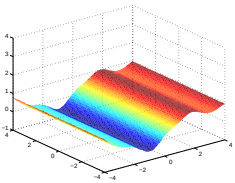


$k_1(x_1, x_1') + k_2(x_2, x_2')$
Additive Kernel

$\downarrow$ $\downarrow$ $\downarrow$

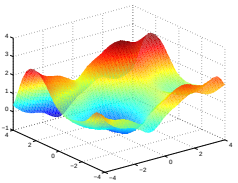


$f_1(x_1)$
1-D GP Prior



$f_2(x_2)$
1-D GP Prior

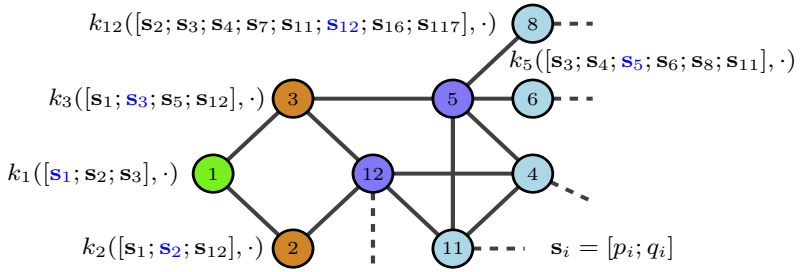
$=$



$f_1(x_1) + f_2(x_2)$
Additive GP Prior

Using Graph-structure to Achieve Additive GP Architecture

- **Neighboring power injections** have highly correlated effect on node voltages
- Effect of **far away power injections** is approximately equal to sum of individual effects



Power Injection Vectors

Number of Nodes

$$k_v(\mathbf{s}^i, \mathbf{s}^j) = \sum_{b=1}^{|\mathcal{B}|} k_b(\mathbf{x}_b^i, \mathbf{x}_b^j)$$

Node Neighborhood Kernel

Neighborhood Aggregated Injection Vectors

More of Vertex Degree Kernel (VDK)

Target Node Voltage

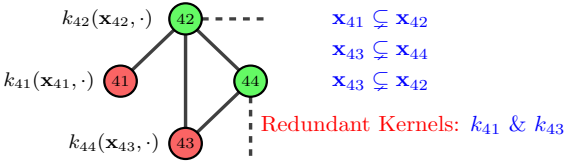
$$V_1(\mathbf{s}) = \underbrace{V_{11}(\mathbf{x}_1) + \cdots + V_{1|\mathcal{B}|}(\mathbf{x}_{|\mathcal{B}|})}_{\text{Latent Node Voltage Functions}}$$

Features

Dimension Reduction: Maximum kernel dimension is equal to maximum vertex degree

Neighborhood Correlation: Aggregated vectors capture correlated injection effects

Constant Kernel Structure: No redesign needed for a constant network structure



Idea of VDK reduction by removing proper subsets.

Size Reduction via Reduced VDK Representation

System†	Reduced VDK	Reduction
118-Bus	97	17.7%
500-Bus	238	52.4%
1354-Bus	786	41.9%

† VDK size is equal to system size i.e. $|\mathcal{B}|$

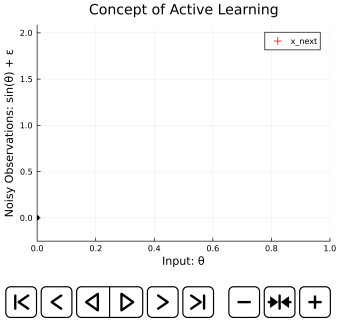
Active Learning

What: Learning by successively selecting the next training point 'intelligently'

Why: To speed-up the learning process using unlabeled data i.e. only input data needed $\implies$ Low Sample Complexity $\implies$ Less Power Flow Samples Needed

Concept: Next training point is the one which has maximum information of underlying function

$$\mathbf{s}^{t+1} = \underset{\mathbf{s}\in\mathcal{L}}{\arg\max}\ \sigma_f^t(\mathbf{s}) \rightarrow \texttt{Only Function Evaluation}$$



$\sigma_f^t(\mathbf{s})$ Submodular Function

Greedy Optimization $1 - 1/e$ Approximation

Finding maximum variance point for large-dimensional input space is hard

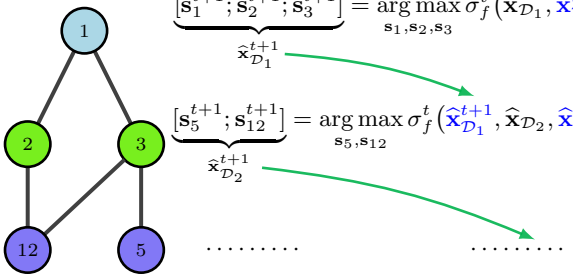
$\implies$ Used mostly up to 20-dimensions

$\implies$ Power systems have 100s of uncertain power injections

Network-swipe Active Learning

$$\text{Variance:}\quad \sigma^2[f(\mathbf{s})] = \underbrace{\sum_{b=1}^{|\mathcal{B}|} k_b(\mathbf{x_b},\mathbf{x_b})}_{\text{Constant}} - \underbrace{\left[\sum_{b=1}^{|\mathcal{B}|} k_b(X_b,\mathbf{x_b})\right]^T}_{} \underbrace{[K_1+\cdots+K_{|\mathcal{B}|}]^{-1}}_{\text{Constant Matrix}} \underbrace{\left[\sum_{b=1}^{|\mathcal{B}|} k_b(X_b,\mathbf{x_b})\right]}_{\text{Variable Vector}}$$

Neighborhood Aggregated Injection Vectors $\mathbf{x_b}$'s have Overlap



Idea of network-swipe algorithm for AL.

$$\widehat{\mathbf{x}}^{t+1}_{\mathcal{D}_i} = \arg\max_{\widehat{\mathbf{x}}_{\mathcal{D}_i}\in\mathcal{L}_i} \sigma_f^t(\widehat{\mathbf{x}}^{t+1}_{\mathcal{D}_{1\ldots i-1}}, \widehat{\mathbf{x}}_{\mathcal{D}_i}, \widehat{\mathbf{x}}^t_{\mathcal{D}_{i+1\ldots d}})$$

$$\widehat{\mathbf{x}}_{\mathcal{D}_i} = \{\mathbf{s}_j | \mathbf{s}_j \in \overline{\mathbf{x}}_{\mathcal{D}_i} \text{ and } \mathbf{s}_j \notin \overline{\mathbf{x}}_{\mathcal{D}_i} \forall\, j < i\}$$

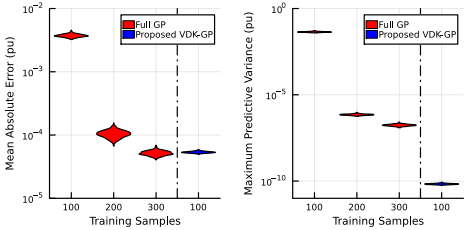
Algorithm 1 Network-Swipe Algorithm for AL

Require: T, $\mathcal{D}$, $\{\mathbf{s}^1, V^1\}$
1: Initialize GP model (2) with VDK (5)
2: **for** $t = 1$ to T **do**
3: Solve (7) for $\widehat{\mathbf{x}}^{t+1}_{\mathcal{D}_i}$, sequentially for $i = 1 \ldots d$
4: Solve ACPF for load $\mathbf{s}^{t+1}$ to get V^{t+1}
5: Update GP model with $(\mathbf{s}^{t+1}, V^{t+1})$
6: **end for**
Output: Compute $\mu_f(\mathbf{s}), \sigma_f^2(\mathbf{s})$ for final GP

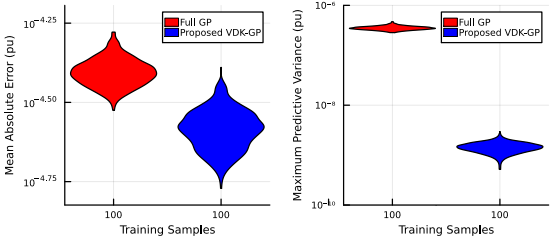
P. Pareek, D. Deka, and S. Misra, "Graph-Structured Kernel Design for Power Flow Learning using Gaussian Processes". Arxiv.2308.07867

Benchmarking

500 independent trials with 100 training samples; Testing: 1000 unique samples



V_1 within $\pm 10\%$ hypercube for 500-Bus system.



V_1 within $\pm 10\%$ hypercube for 118-Bus system.

Three Times Lower Sample Complexity

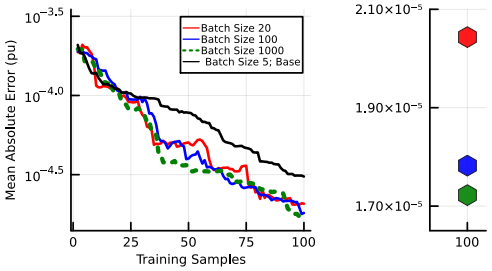
50% Lower Error & 100 Times Confident Model

Proposed VDK-GP *outperforms* a *3-layer, 1000-neuron Deep Neural Network* for using 100 training samples in 118-Bus system

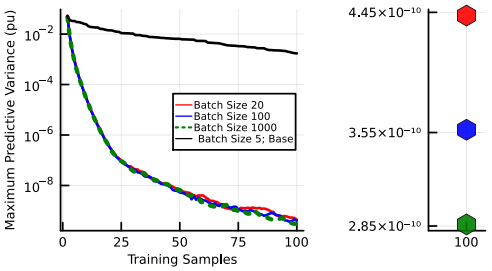
Node	*MAE (pu)*	
	Proposed	*DNN*
1	$\mathbf{5.22 \times 10^{-5}}$	1.89×10^{-4}
43	$\mathbf{8.70 \times 10^{-5}}$	9.77×10^{-4}
117	$\mathbf{2.26 \times 10^{-5}}$	9.05×10^{-4}

Accurate, loading independent power flow model with extremely low sample complexity
Useful for power system operation under uncertainty

Active Learning Performance: 118-Bus System



(a) MAE (pu) over 1000 testing samples.



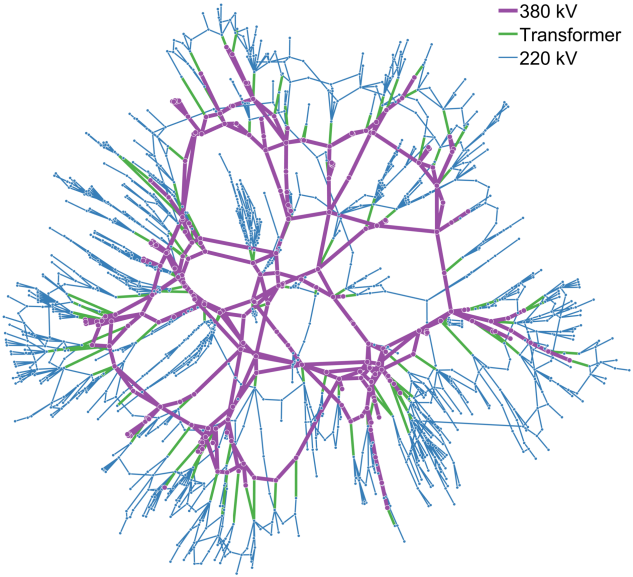
(b) MPV (pu) over 1000 testing samples.

	Learning Method (Training Samples)		
	Full GP (100)	VDK-GP (100)	AL (100)
MAE $\times 10^{-5}$	3.98	2.66	**1.72**
ME $\times 10^{-5}$	17.1	11.5	**6.98**
#ACPF Samples	5×10^4	5×10^4	**100**

– Full GP & VDK-GP: Mean over 500 random trails

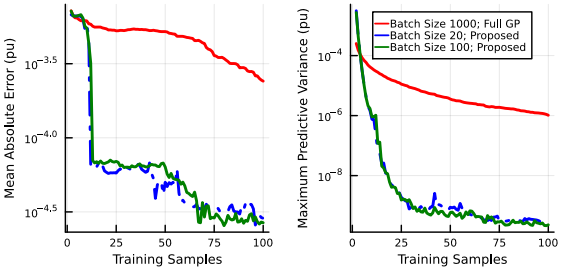
Network-swipe is successful in achieving near smooth decay in Variance

1354-Bus portion of European transmission system



1,354-buses, 260 generators, and 1,991-branches and it operates at 380 and 220 kV.

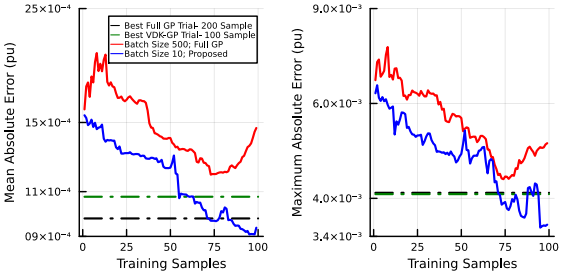
Active Learning Performance: Larger Systems



Learning V_1 for 500-Bus system.

	Learning Method (Training Samples)		
	Full GP (300)	VDK-GP (100)	AL (100)
MAE $\times 10^{-5}$	5.22	5.35	**2.68**
ME $\times 10^{-5}$	22.4	23.3	**11.0**
#PF	15×10^4	5×10^4	**100**

– Full GP & VDK-GP: Mean over 500 random trails



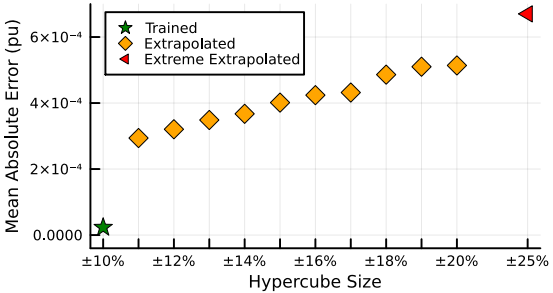
Learning V_5 for 1354-Bus system.

	Learning Method (Training Samples)		
	Full GP (200)	VDK-GP (100)	AL (100)
#PF	5000	2500	**100**

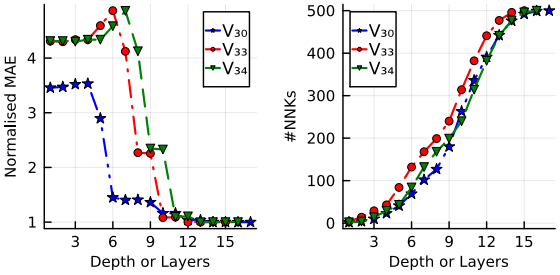
– #PF: Total number of ACPF solutions required

A random trial of active learning is better than large number of passive learning attempts.

More Insights: Extrapolation & Depth Effect



Extrapolation of VDK-GP model trained within ±10% hypercube for 118-Bus system.



Effect of depth on learning quality of three different voltage function in 500-Bus system.

References I

[1] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[2] P. Pareek, D. Deka, and S. Misra, "Graph-structured kernel design for power flow learning using gaussian processes," *arXiv preprint arXiv:2308.07867*, 2023.

[3] P. Pareek and H. D. Nguyen, "A framework for analytical power flow solution using gaussian process learning," *IEEE Transactions on Sustainable Energy*, vol. 13, no. 1, pp. 452–463, 2021.

[4] P. Pareek *et al.*, "Privacy-preserving feasibility assessment for p2p energy trading and storage integration," in *IEEE PESGM*. IEEE, 2022, pp. 1–5.

[5] P. Pareek, W. Yu, and H. D. Nguyen, "Optimal steady-state voltage control using gaussian process learning," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 7017–7027, 2020.

[6] P. Pareek and H. D. Nguyen, "Gaussian process learning-based probabilistic optimal power flow," *IEEE Transactions on Power Systems*, vol. 36, no. 1, pp. 541–544, 2020.

[7] P. Pareek, L. M. I. Sampath, H. D. Nguyen, and E. Y. Foo, "Locating critical prosumers in p2p dominant grids using state-sensitivity function," *IEEE Transactions on Smart Grid*, 2023.

A part of this work is funded by LANL's Directed Research and Development (LDRD) project: "High-Performance Artificial Intelligence" (20230771DI) and and The Department of Energy (DOE), USA under the Advanced grid Modeling (AGM) program. The research work conducted at Los Alamos National Laboratory is done under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy under Contract No. 89233218CNA000001

Summary: Till Now & Next?

Key Take Away

Through Application Specific Physics-inspired Kernels, GPs can be Excellent Interpretable Learning Tools in Low Data Regimes.

What Closed-forms Are Useful For?

- Large scale power flow learning [2]
- Uncertainty Quantification and Behavior Characterization [3]
- Privacy-preserving Probabilistic Feasibility Assessment [4]
- Voltage Control with Linear Forms under Uncertainty [5]
- Optimal Power Flow Proxy [6]
- Locating Critical Nodes in Distribution Systems [7]

What's Next?

- Learning Power Flow under Topological and Injection Uncertainties : Next Week
- GPU Trainable Models of Full GP and VDK-GP for Power Flow : Later This Month