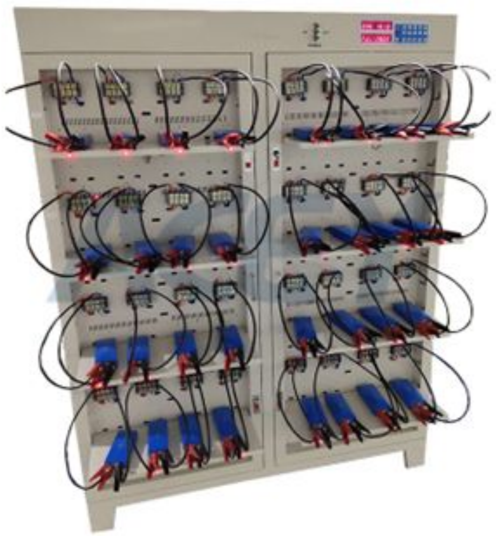
Deep Surrogate Models for Fast Prediction and Parameter Inference of Li-ion Batteries

Siddhant Singh, Haonan Zhao, Atila Haimiti

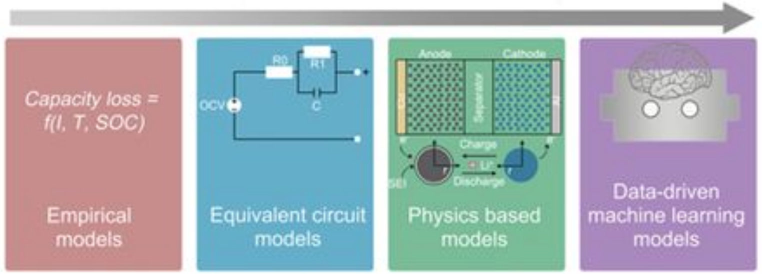
Motivation

Motivation for Creating Battery Models

- Real-World experiments are expensive



Battery cycling can be expensive and time-consuming



Battery modeling can reduce the need for testing as extensively

$$\varepsilon_k \;\; c*_{\{k,max\}} \;\; \sigma*_k \;\; D*_{\{s,k\}} \;\; R*_k \;\; a*_k \;\; m*_k \;\; L*_k$$
$$U*_{\{k,ref\}} \;\; c*_{\{e,typ\}} \;\; D*_{\{e,typ\}} \;\; \kappa*_{\{e,typ\}} \;\; b \;\; t^+$$

Requires accurate parameter identification to use in the models

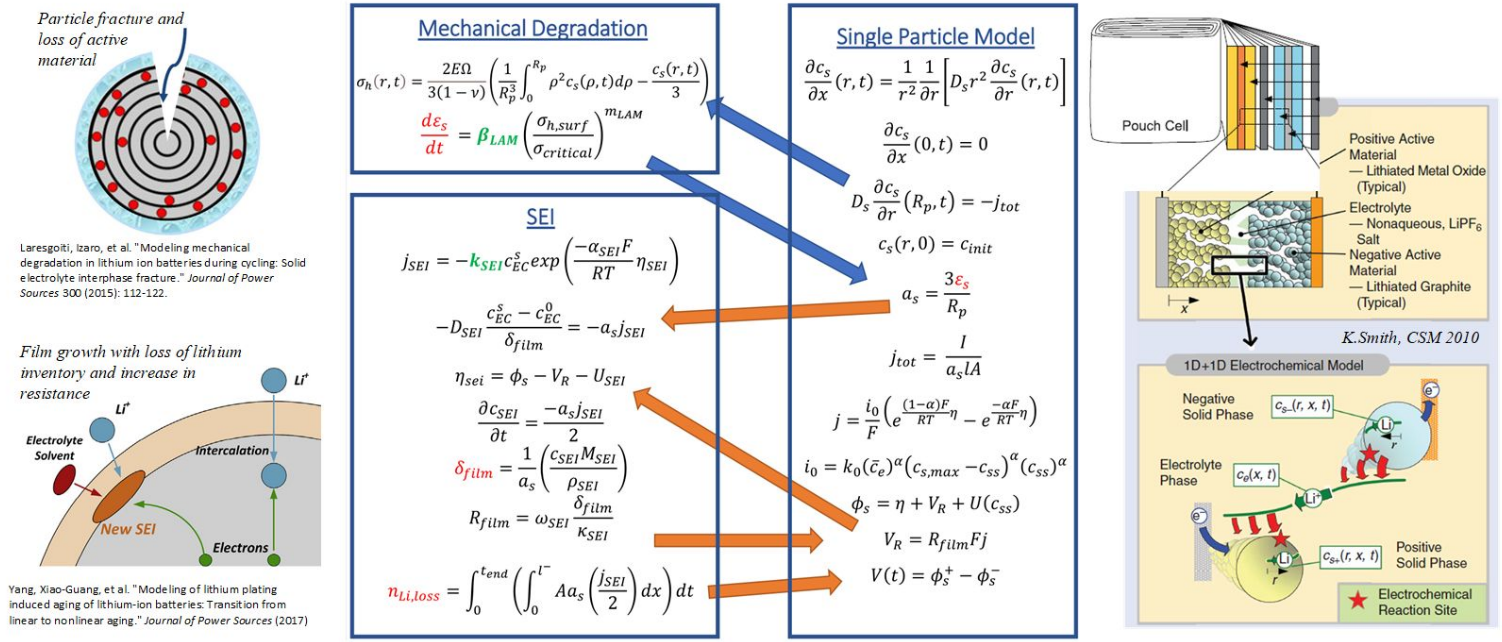
Motivation-continued

Motivation for Creating Surrogate Model for Battery Models

- Even reduced-ordered systems involves system of several highly interdependent ODEs → Computationally Costly

Solution: Use AI for Science methods to build a data-driven, and to what extent possible, a physics-informed model.

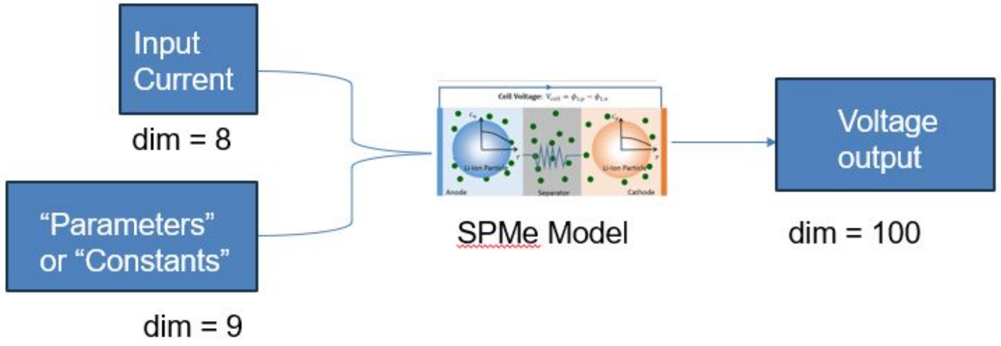
SPM model combined with degradation



Marquis, Scott G., Valentin Sulzer, Robert Timms, Colin P. Please, and S. Jon Chapman. *Journal of The Electrochemical Society* 166, no. 15 (November 8, 2019): A3693.

Datasets Descriptions

Physical System Characteristics/Assumptions



- ❑ Single Particle Model with Electrolyte Dynamics
- ❑ Solver: Python library - Pybamm

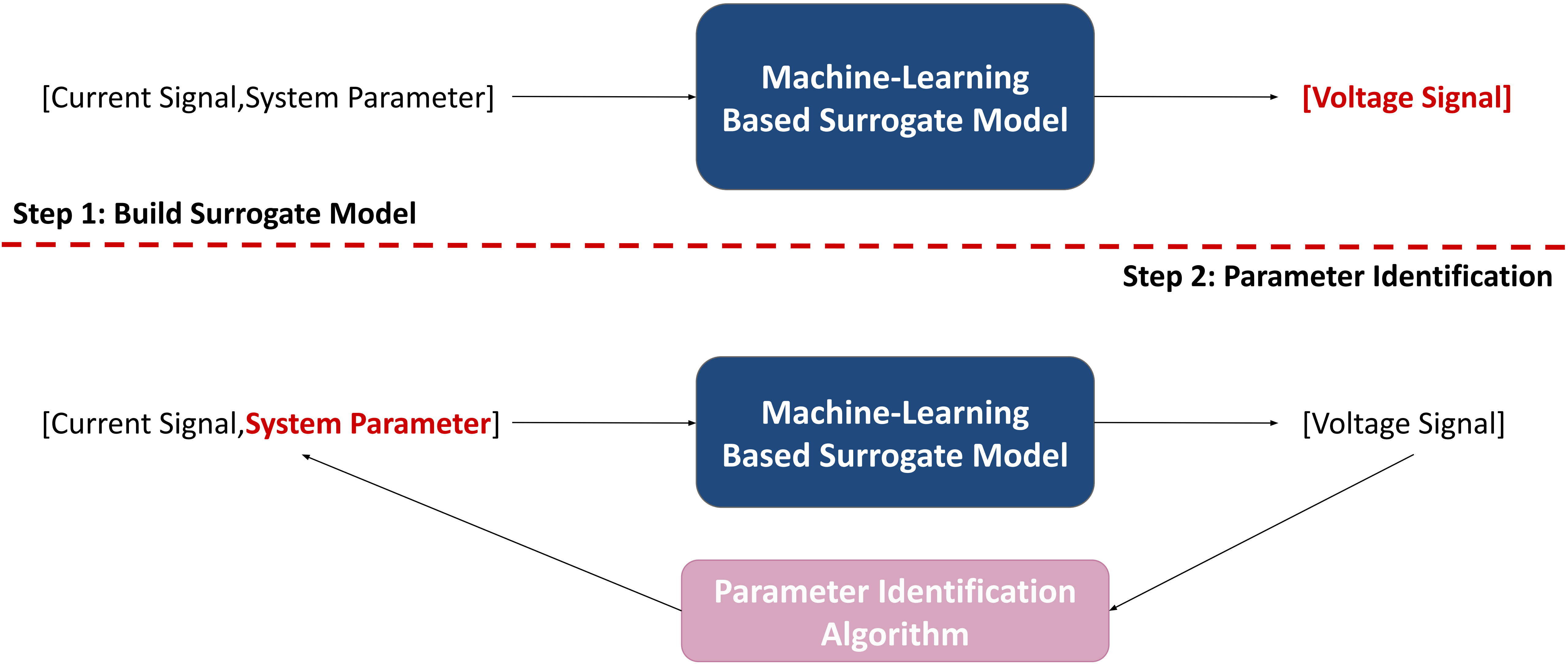
System Parameters

- ❑ Inputs: Nine parameters define the behavior of the model
- ❑ Inputs:Eight parameters definite the current signal (time-series)
- ❑ Outputs: Voltage response (time-series)

Number of Sample and Size

- ❑ 4000 samples for training and testing
- ❑ Each samples has 105 time-steps (used adaptive time-stepping for data generation, so timesteps are non-uniformly spaced)
- ❑ Occupies around 20 MB

Overview Proposed Approach



Overview Proposed Approach - For Surrogate

Baseline Approach

- ❑ **DeepOnet** → Standard neural operator

Proposed Approaches

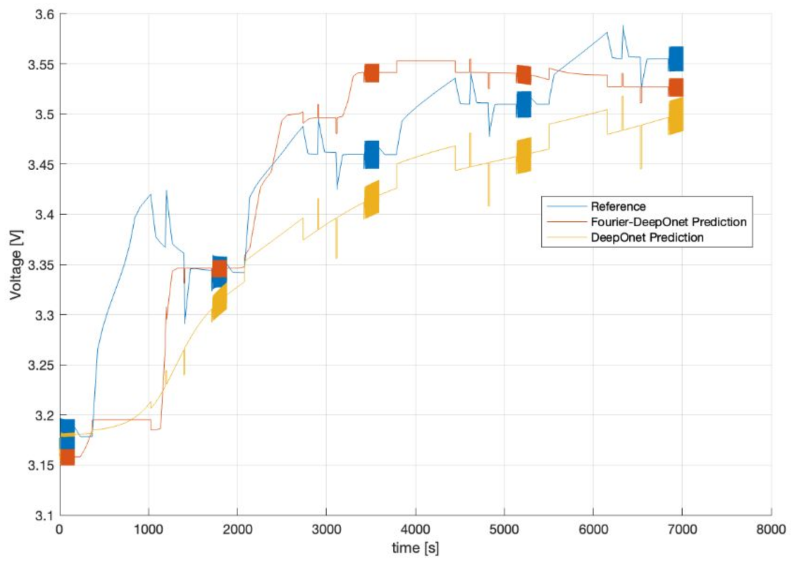
- ❑ **LSTM** → Good at sequence-to-sequence learning
- ❑ **Mamba-DeepOnet** → Neural operators natural for solving the dynamic systems.

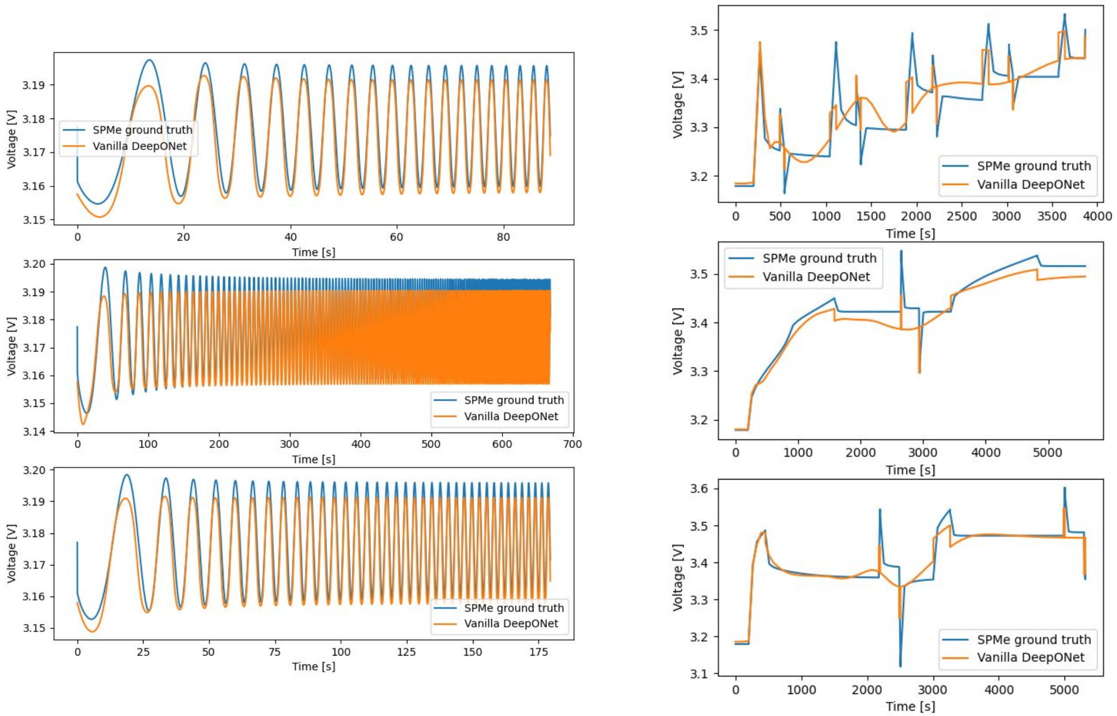
Baseline Approach - DeepONet

The full battery dynamics:

- ❑ **Hybrid pulse power characterization (low frequency) + chirp signal (broad-band excitation).**
- ❑ **DeepONet fails to capture both of them at the same time.**

Baseline: DeepONet





Proposed Approach - LSTM

Inputs-Outputs to the LSTM

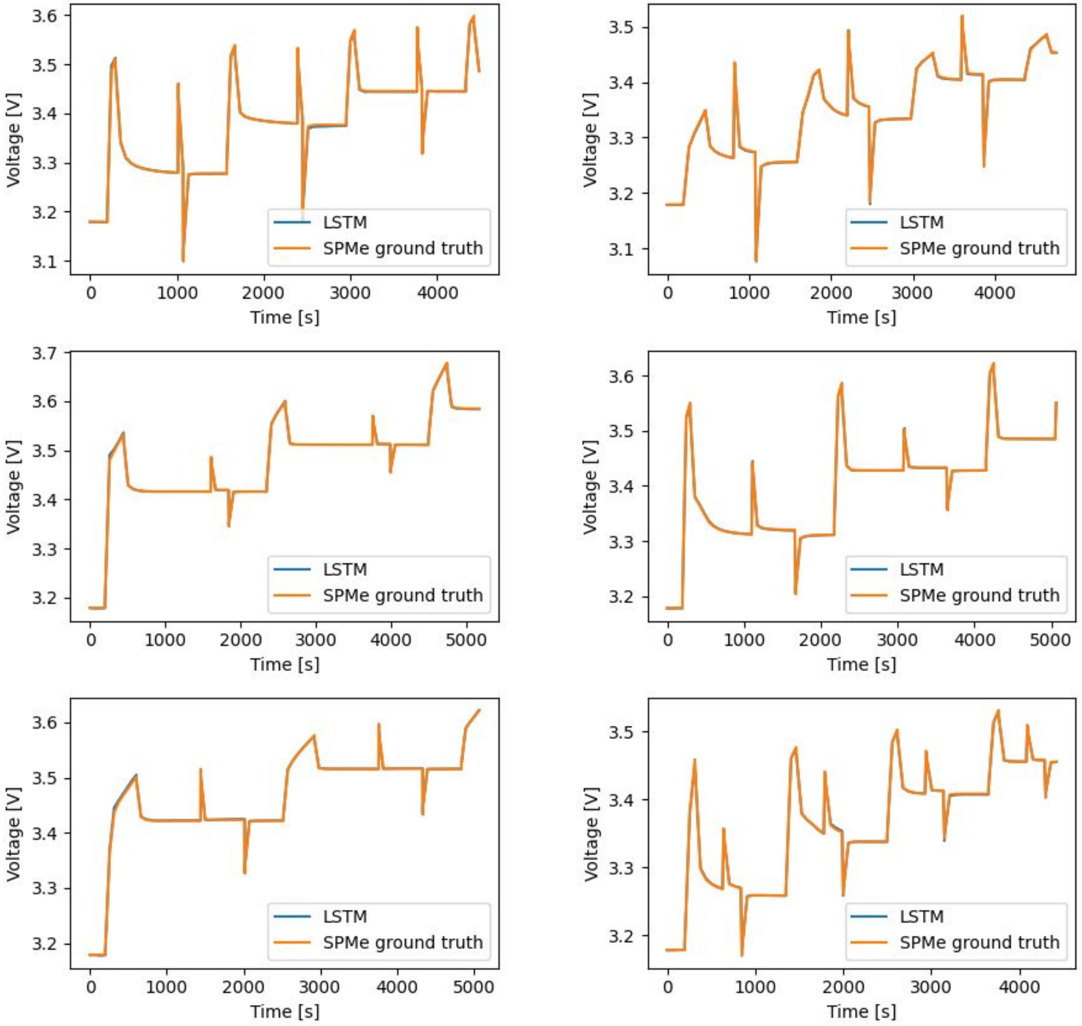
- Inputs: Current sequence along with the system parameters
- Output: Voltage sequence
- Sequence-Length: 105

Why Use LSTM?

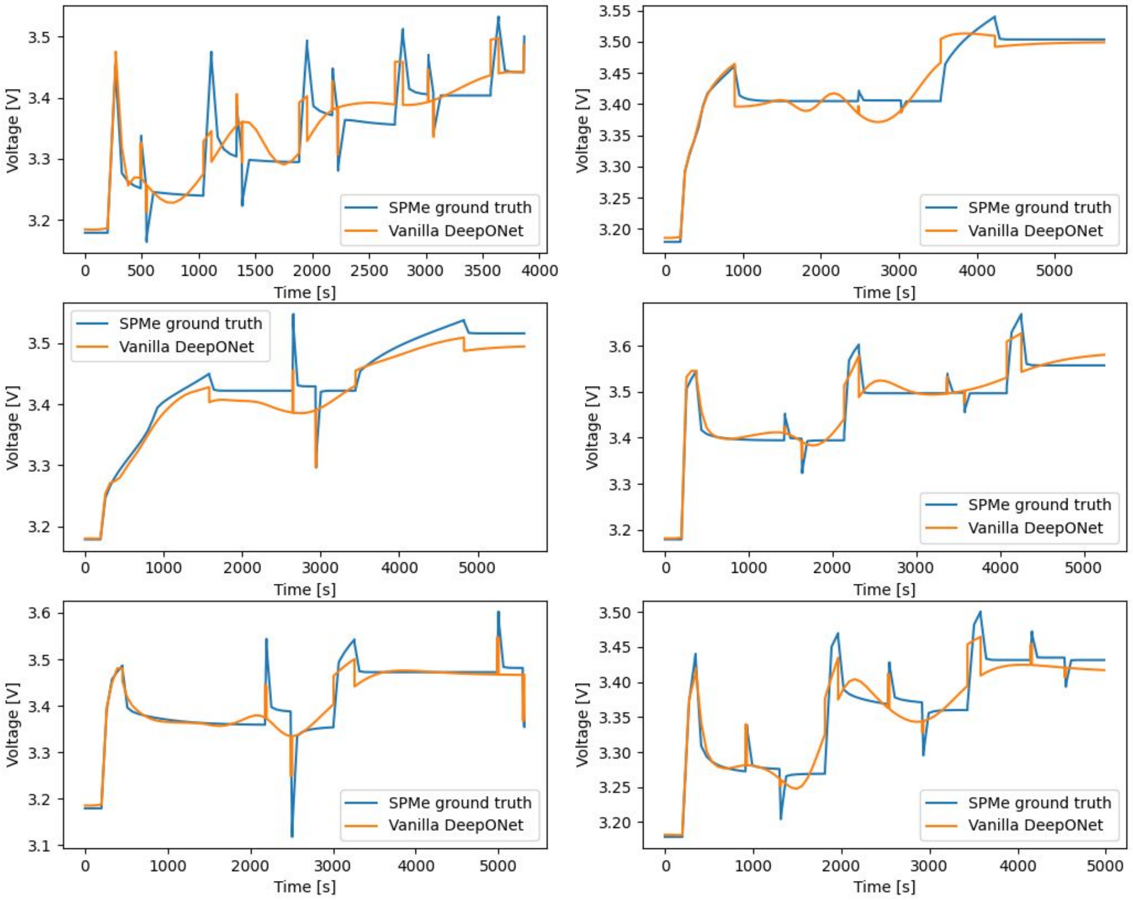
- The process of the physical battery computational model resembles recurrent networks
- LSTM is suitable for mitigating gradient explosion and vanishing

Case Study Results For Surrogate - LSTM

LSTM



Vanilla DeepONet



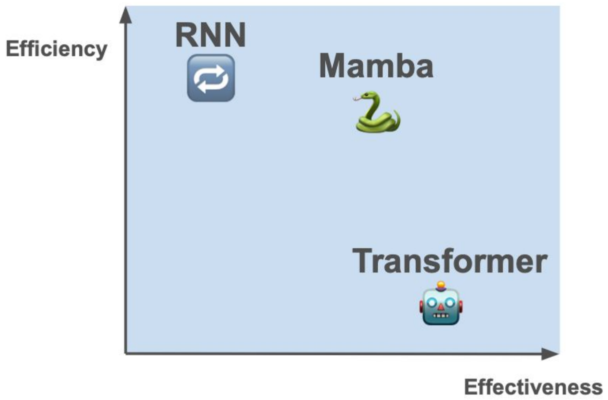
Proposed Approach - Mamba-DeepOnet

What is Mamba

- State-space models for dynamic systems.
- Traditional SSM expressivity and speed have trade-offs -> hidden layer size.
- Discretization of A,B,C,Δ is adaptive, Mamba balances between efficiency and effectiveness.

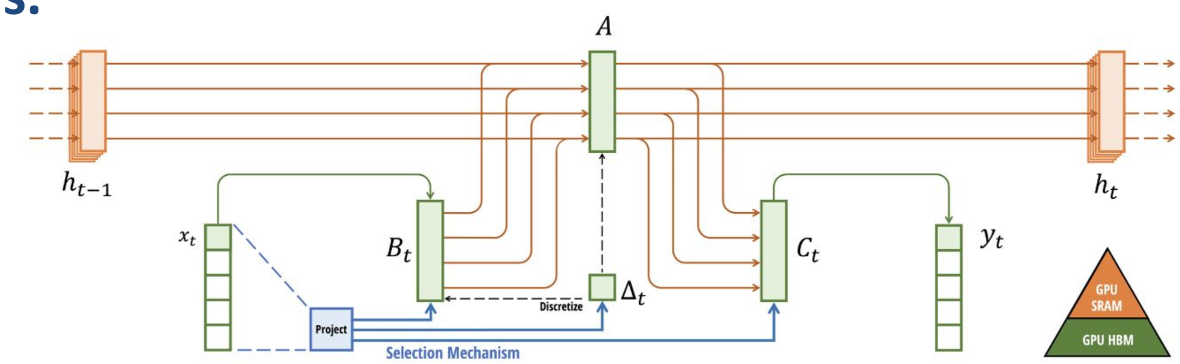
$$\overline{A} = \exp(\Delta A) \qquad \overline{B} = (\Delta A)^{-1}(\exp(\Delta A) - I) \cdot \Delta B$$

- Parallelized hardware-aware selective scan



Selective State Space Model

with Hardware-aware State Expansion

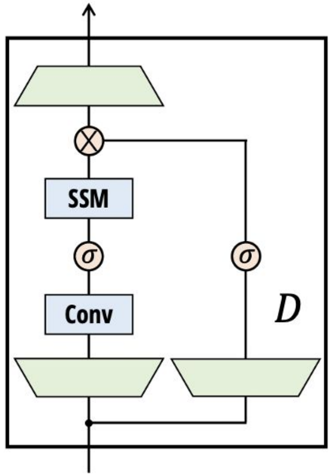


$$h_t = \bar{A}h_{t-1} + \bar{B}x_t$$
$$y_t = Ch_t + Dx_t$$

Algorithm 2 SSM + Selection (S6)

Input: x : (B, L, D)
Output: y : (B, L, D)

A : (D, N) $\leftarrow$ Parameter
▹ Represents structured $N \times N$ matrix
B : (B, L, N) $\leftarrow s_B(x)$
C : (B, L, N) $\leftarrow s_C(x)$
Δ : (B, L, D) $\leftarrow \tau_\Delta(\text{Parameter}+s_\Delta(x))$
$\overline{A}, \overline{B}$: (B, L, D, N) $\leftarrow$ discretize(Δ, A, B)
$y \leftarrow$ SSM($\overline{A}, \overline{B}, C$)($x$)
▹ Time-varying: recurrence (*scan*) only
return y



Mamba

Gu, Albert, and Tri Dao. "Mamba: Linear-Time Sequence Modeling with Selective State Spaces." arXiv, May 31, 2024.

Proposed Approach - Mamba-DeepOnet

Why combine Mamba and DeepONet

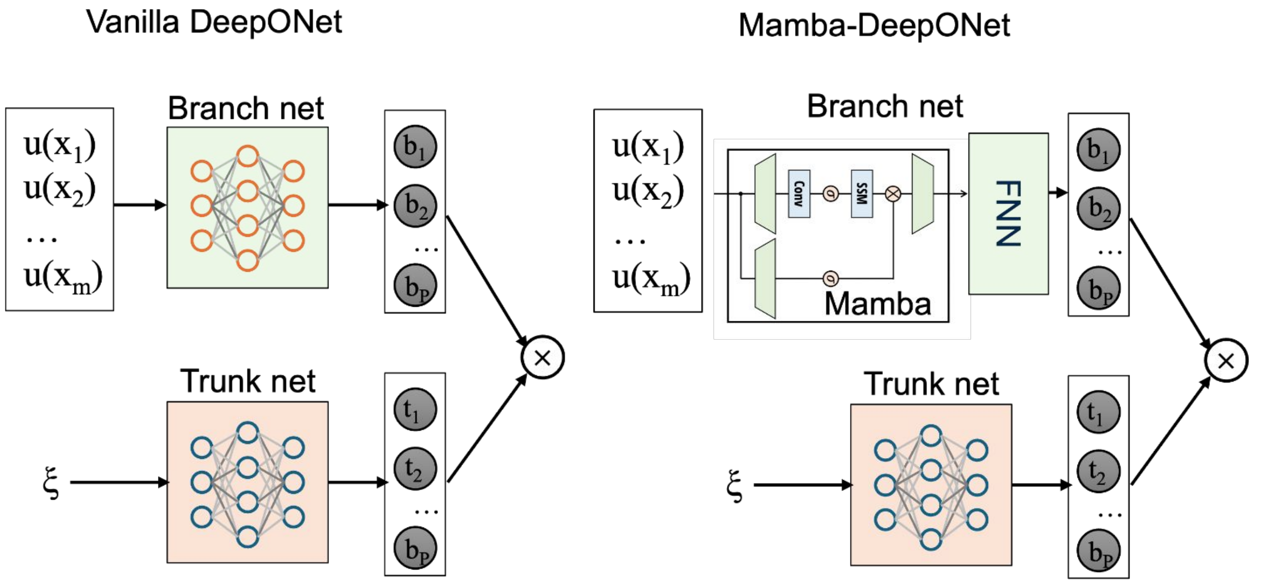
- If one model learns a dynamic system fast and accurately:

$$\dot{\boldsymbol{x}} = \boldsymbol{v}(t, \boldsymbol{x})$$
$$\boldsymbol{x}\big|_{t=0} = \boldsymbol{x}_0$$

- Then the neural operator for this family of dynamic systems should learn the following fast and accurately:

$$\dot{x} = v[\xi(t), x]$$
$$\xi = f(t)$$
$$x\Big|_{\partial\xi} = x_0$$
$$\xi\Big|_{t=0} = \xi_0$$

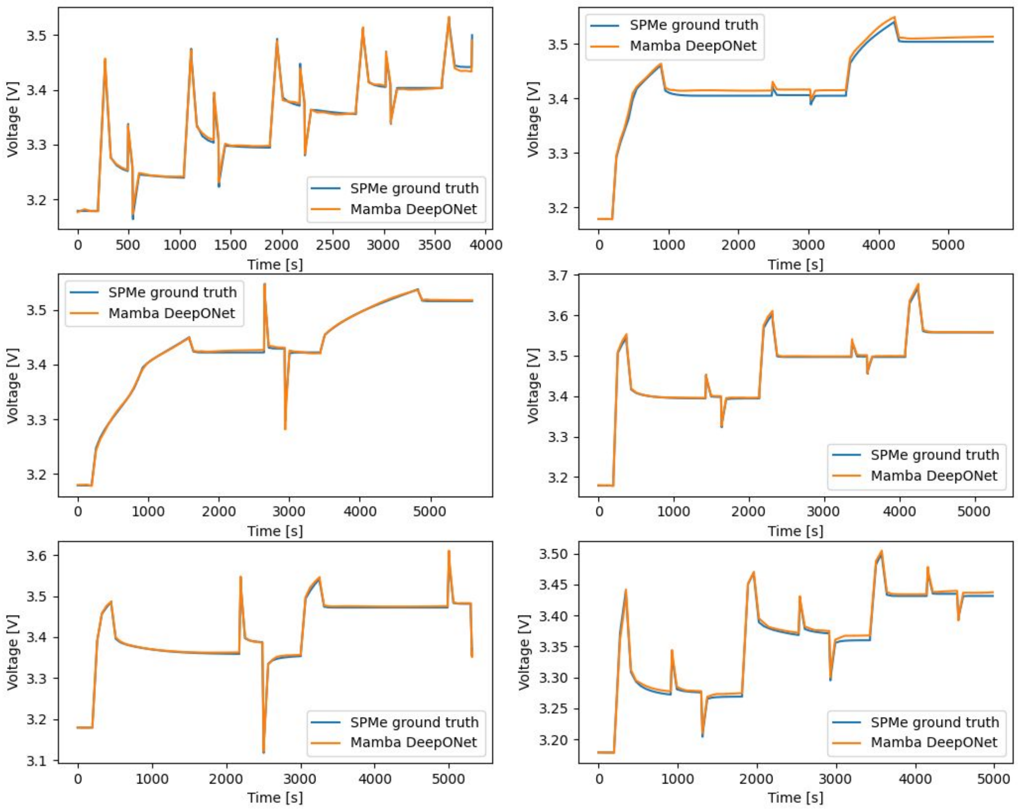
- Mamba branch net to encode the function space, trunk to encode the output domain.



$$\left| G(u)(y) - \sum_{k=1}^{p} \underbrace{\sum_{i=1}^{n} c_i^k \sigma\left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k\right)}_{\text{branch}} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{\text{trunk}} \right| < \epsilon$$

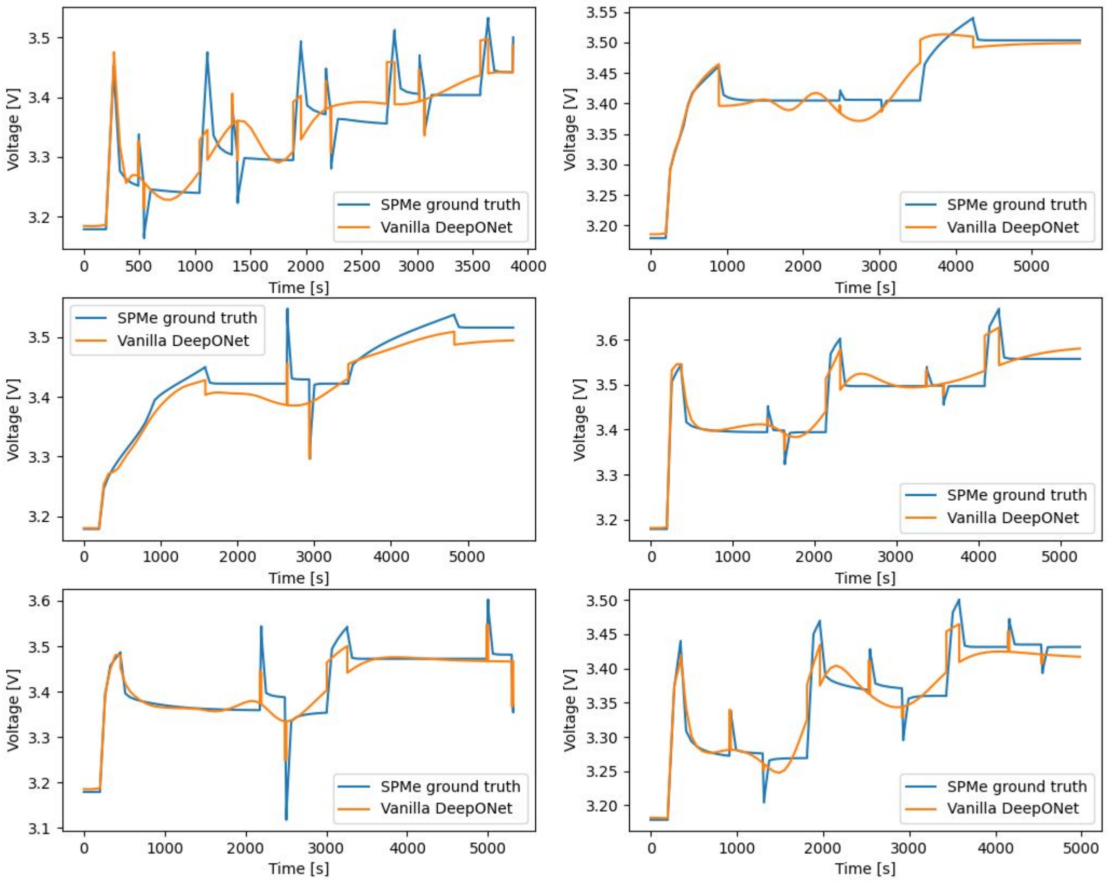
Results For Surrogate - Mamba-DeepONet

Mamba-DeepONet



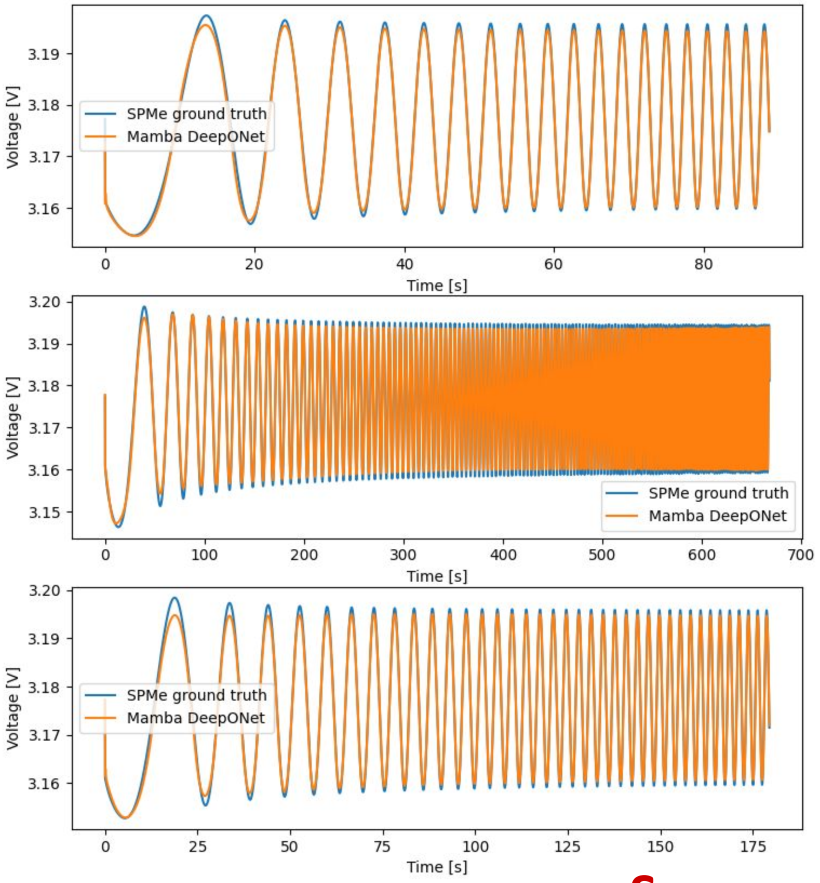
Sequence length = 105

Vanilla DeepONet

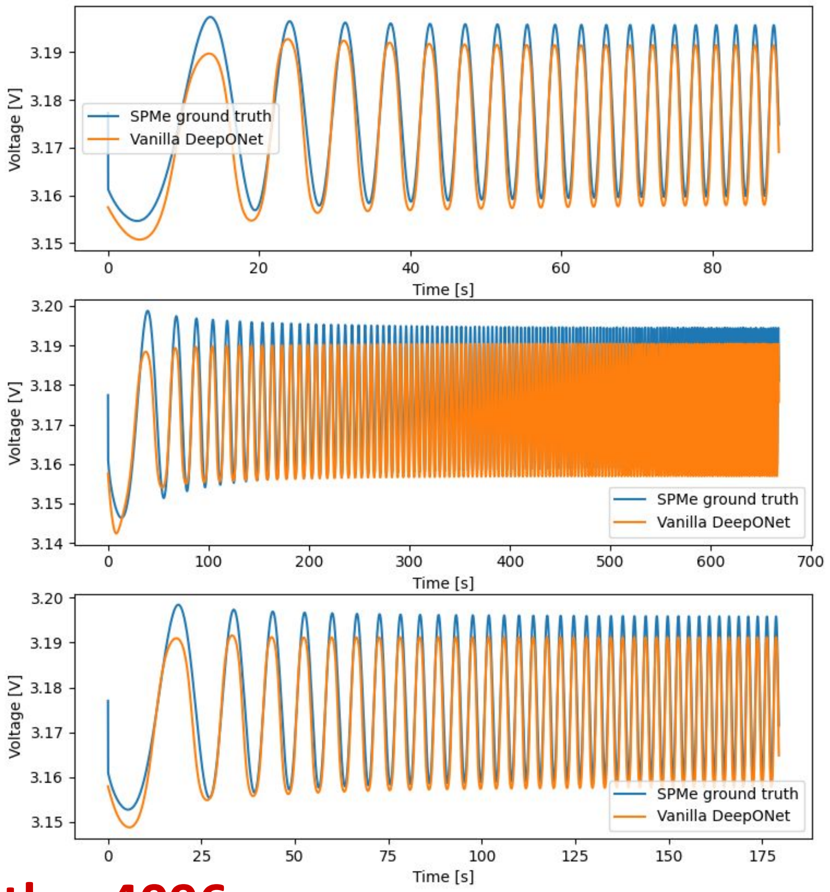


Results For Surrogate - Mamba-DeepONet

Mamba-DeepONet



Vanilla DeepONet



Sequence length = 4096

Results For Surrogate - Summary

	MSE Loss of HPPC fitting	MSE Loss of chirp
DeepOnet	1.41e-03	7.33e-06
Mamba-DeepOnet	1.43e-05	7.57e-07
LSTM	3.71e-06	N/A

Proposed Approach - For Parameter Identification

Tree-structured Parzen Estimator Algorithm (TPE)

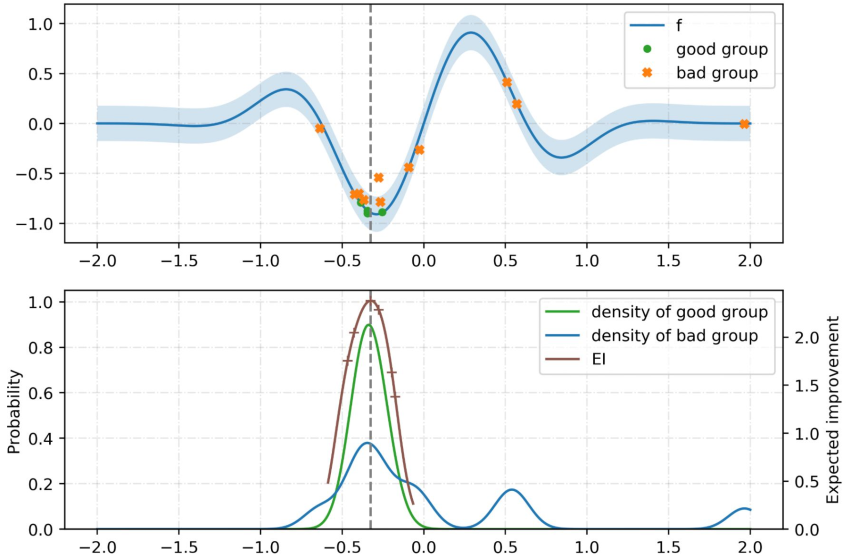
- Given past search history, suggests the next trial for a single parameter search (ignore correlation)
- Model P(single parameter value | loss) instead of P(loss|single parameter value)
- Chose next trial single parameter value base on "Promisingness" → Proportional to Expected Improvment (EI)

density of good parameter configurations

$$\text{Promisingness} = \frac{P(\text{single parameter value} \mid \text{loss} < \text{loss}^*)}{P(\text{single parameter value} \mid \text{loss} > \text{loss}^*)}$$

density of bad parameter configurations

argmax to find next parameter value

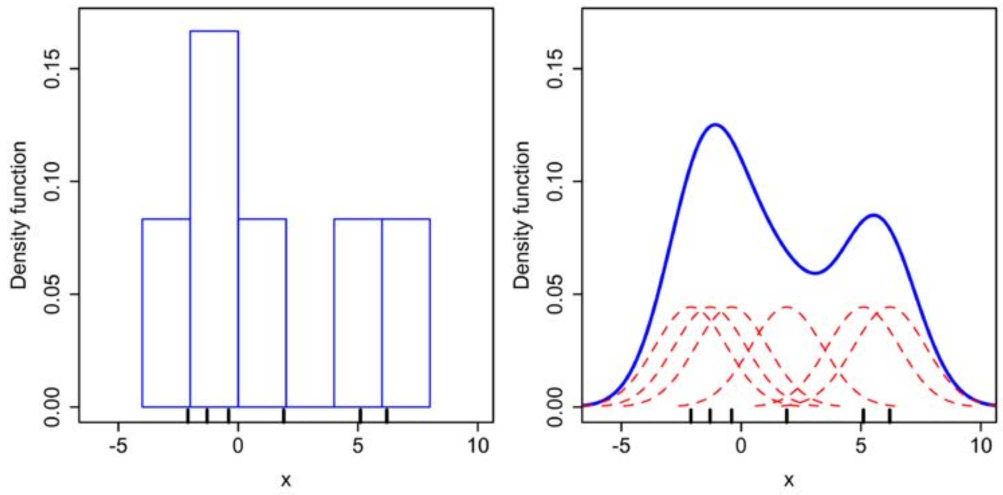


Watanabe, Shuhei. "Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance." arXiv, May 26, 2023.

https://techblog.criteo.com/hyper-parameter-optimization-algorithms-2fe447525903

More on Tree-Structured Parzen Estimator (TPE)

- Non-parametric method (does not assume any pre-defined distribution)
- Uses Kernel Density Estimation (KDE) to estimate the densities of good parameter values (low loss, numerator in promisingness) and bad parameter values (high loss)
- Uses a Gaussian Kernel with a adaptively-determined bandwidth
- Balances exploration and exploitation by varying the loss threshold. A lower threshold prioritizes exploitation. A higher loss threshold prioritizes exploration.

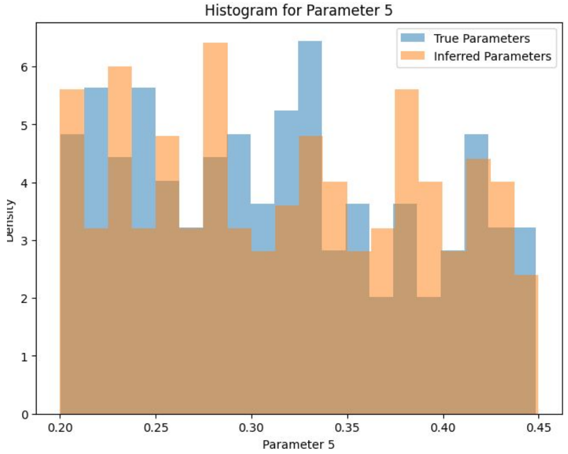


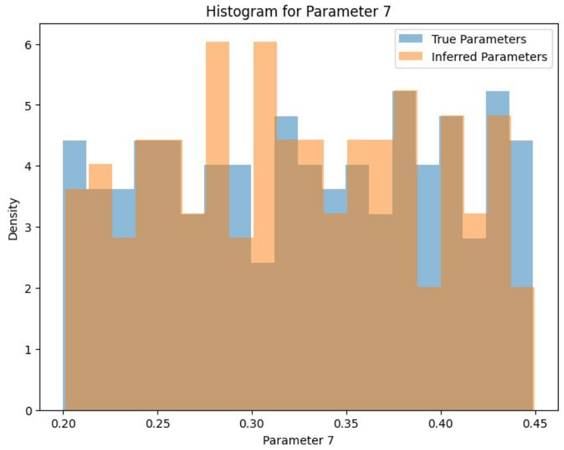
Parameter Identification Results

Parameter Index	LSTM Error (Normalized)	Mamba-DeepONet Error (Normalized)
0	1.285372	1.91261
1	13.12026	18.138
2	30.7648	35.69853
3	2.425946	3.031413
4	0.011579	0.012378
5	0.245476	0.245792
6	0.202349	0.251153
7	0.250003	0.232978
8	0.567224	0.607218

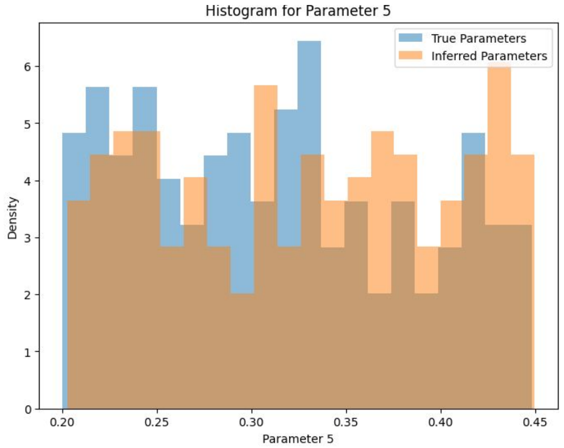
Case Study Results - Example Histograms

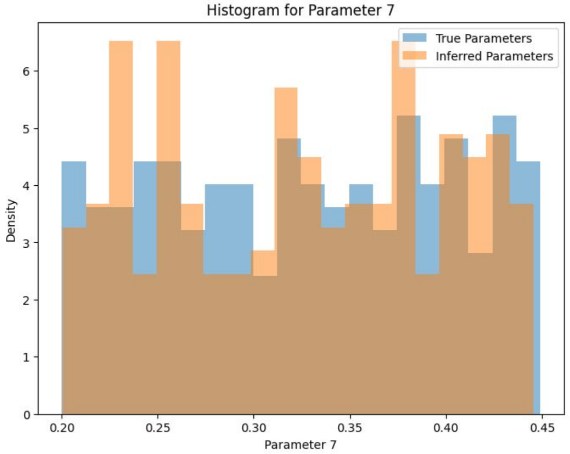
LSTM





Mamba-DeepONet

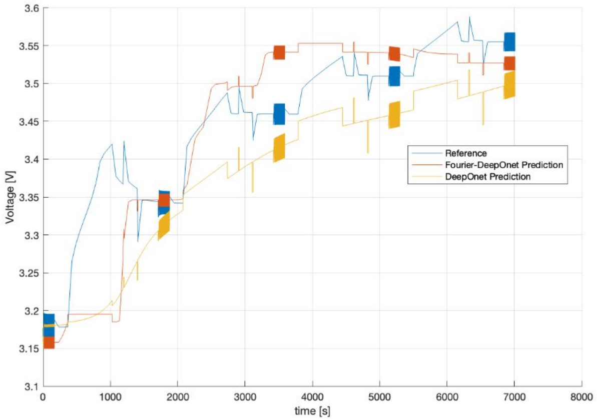




Future Work: Incorporate Chirp Signal

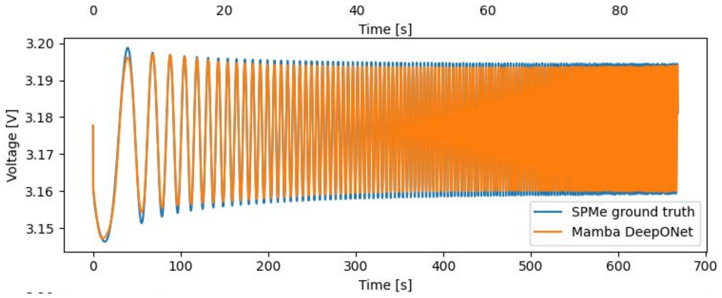
With chirp, fast and increasing frequency

- There is interest in using chirp signals (a high frequency to low frequency sweep) for greater excitation of battery dynamics enabling faster parameter identification
- Mamba-DeepONet can handle long sequences better than LSTM
- Mamba-DeepONet can handle multidimensional domain (trunk net)



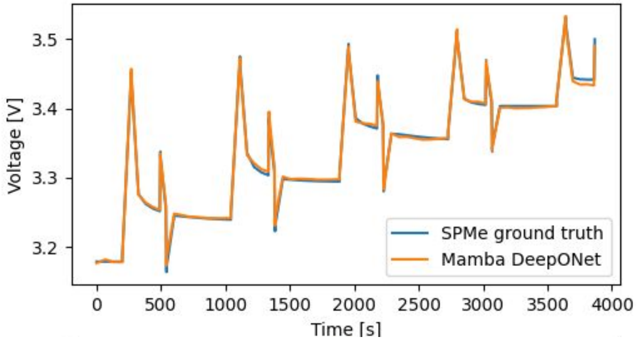
Broadband frequency data

=



Chirp signal
long sequence length
short time step

+



HPPC
short sequence length
long time step

Conclusion

1. **Effective Surrogate Modeling**:
    - Mamba-DeepONet and LSTM outperform DeepONet as surrogate models for the lithium-ion battery single particle model with electrolyte dynamics.
2. **Machine Learning for Parameter Inference using Probabilistic Methods**:
    - Accurate machine-learning-based surrogate models enable advanced parameter inference techniques.
    - Bayesian optimization methods, such as Tree-structured Parzen Estimators (TPE), are well-suited for this purpose.
3. **TPE Effectiveness**:
    - TPE demonstrates strong performance in estimating key model parameters, and is computationally efficient and parallelizable.

Appendix

Proposed Approach - For Parameter Identification

Algorithm 1 Tree-structured Parzen estimator (TPE)

N_{init} (The number of initial configurations, `n_startup_trials` in Optuna), N_s (The number of candidates to consider in the optimization of the acquisition function, `n_ei_candidates` in Optuna), Γ (A function to compute the top quantile γ, `gamma` in Optuna), W (A function to compute weights $\{w_n\}_{n=0}^{N+1}$, `weights` in Optuna), k (A kernel function), B (A function to compute a bandwidth b for k).

$\mathcal{D} \leftarrow \emptyset$
for $n = 1, 2, \ldots, N_{\text{init}}$ **do** ▷ Initialization
    Randomly pick $\boldsymbol{x}_n$
    $y_n := f(\boldsymbol{x}_n) + \epsilon_n$ ▷ Evaluate the (expensive) objective function
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{x}_n, y_n)\}$
while Budget is left **do**
    Compute $\gamma \leftarrow \Gamma(N)$ with $N := |\mathcal{D}|$ ▷ Section 3.1 (Splitting algorithm)
    Split $\mathcal{D}$ into $\mathcal{D}^{(l)}$ and $\mathcal{D}^{(g)}$
    Compute $\{w_n\}_{n=0}^{N+1} \leftarrow W(\mathcal{D})$ ▷ See Section 3.2 (Weighting algorithm)
    Compute $b^{(l)} \leftarrow B(\mathcal{D}^{(l)}), b^{(g)} \leftarrow B(\mathcal{D}^{(g)})$ ▷ Section 3.3.4 (Bandwidth selection)
    Build $p(\boldsymbol{x}|\mathcal{D}^{(l)}), p(\boldsymbol{x}|\mathcal{D}^{(g)})$ based on Eq. (5) ▷ Use $\{w_n\}_{n=0}^{N+1}$ and $b^{(l)}, b^{(g)}$
    Sample $\mathcal{S} := \{\boldsymbol{x}_s\}_{s=1}^{N_s} \sim p(\boldsymbol{x}|\mathcal{D}^{(l)})$
    Pick $\boldsymbol{x}_{N+1} := \boldsymbol{x}^\star \in \operatorname{argmax}_{\boldsymbol{x} \in \mathcal{S}} r(\boldsymbol{x}|\mathcal{D})$ ▷ The evaluations by the acquisition function
    $y_{N+1} := f(\boldsymbol{x}_{N+1}) + \epsilon_{N+1}$ ▷ Evaluate the (expensive) objective function
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{x}_{N+1}, y_{N+1}\}$

Watanabe, Shuhei. “Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance.” arXiv, May 26, 2023.