2nd IEEE International Challenge in Design Methods for Power Electronics

2025 IEEE Power Electronics Society

MagNet Challenge 2

"From Steady-State to Transient Models!"

Tutorial Session 3, May 30, 2025

Hyukjae Kwon, Shukai Wang, Haoran Li, Thomas Guillod, Minjie Chen, Charles R. Sullivan

GitHub Repository: https://github.com/minjiechen/magnetchallenge-2
pelsmagnet@gmail.com
MagNet 2025 Organizing Team

IEEE POWER ELECTRONICS SOCIETY
Powering a Sustainable Future

IEEE pels MagNetX Challenge 2

IEEE
Advancing Technology for Humanity

Agenda

- **Webinar 1 – Data and Neural Network Methods**
  - May 16th Friday, 9 AM EST
- **Webinar 2 – Analytical Methods (by Dr. Thomas Guillid)**
  - May 23rd, Friday, 9 AM EST
- **Webinar 3 – Model Testing and Evaluation Rules**
  - May 30th, Friday, 9 AM EST
- **Webinar 4 – Brainstorm and Q&A**
  - June 6th, Friday, 9 AM EST

Foundation Model for Power Magnetics

Read history and predict future based on new inputs

Foundation Model

Frequency agnostic

- Any arbitrary / non steady state waveforms

Universal time step

- Long- or short-time steps

Initial state impact

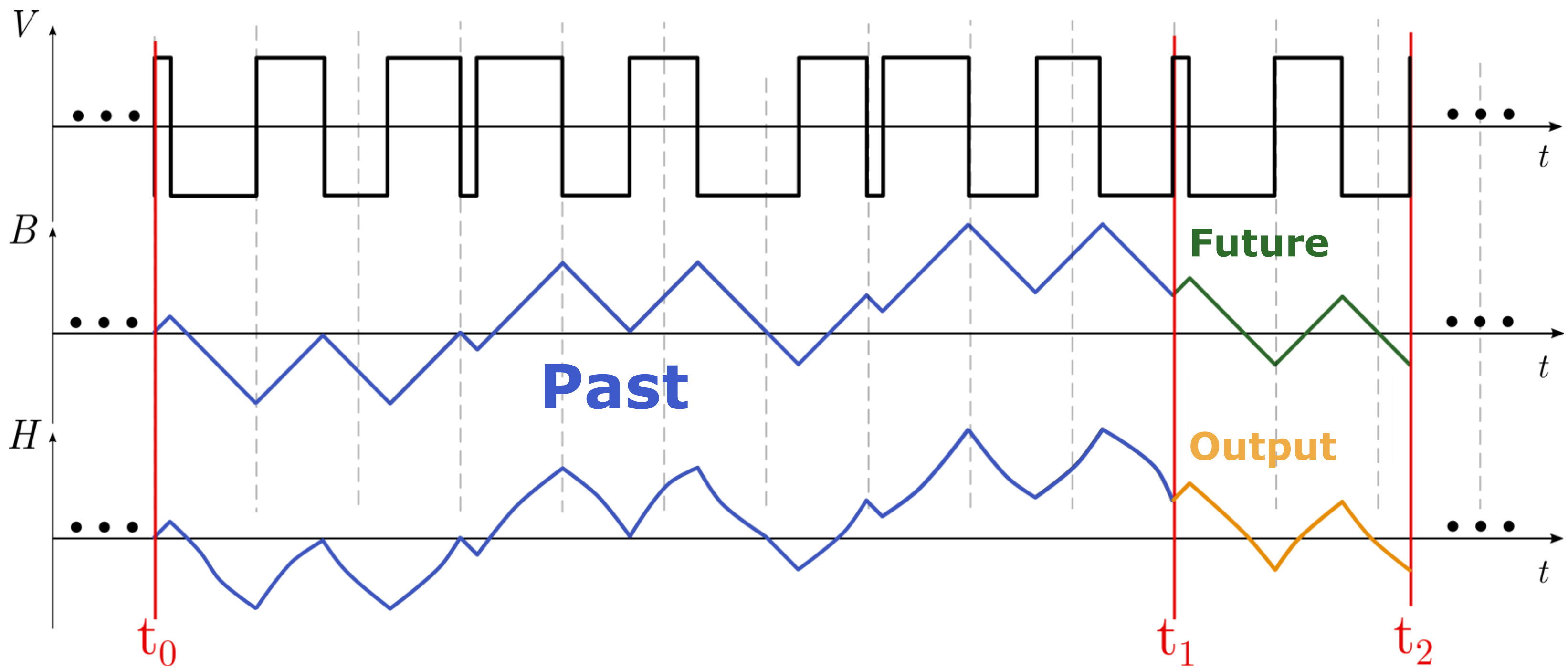
- Hypothesis: impact of initial state has finite time horizon

Rigorous Mathematical Framework

- Flexible, Accurate
- Converge over time to steady state condition
- Explainable modeling framework
- Physics-based, data-driven, or hybrid …

Outcome: A Callable Prediction Function

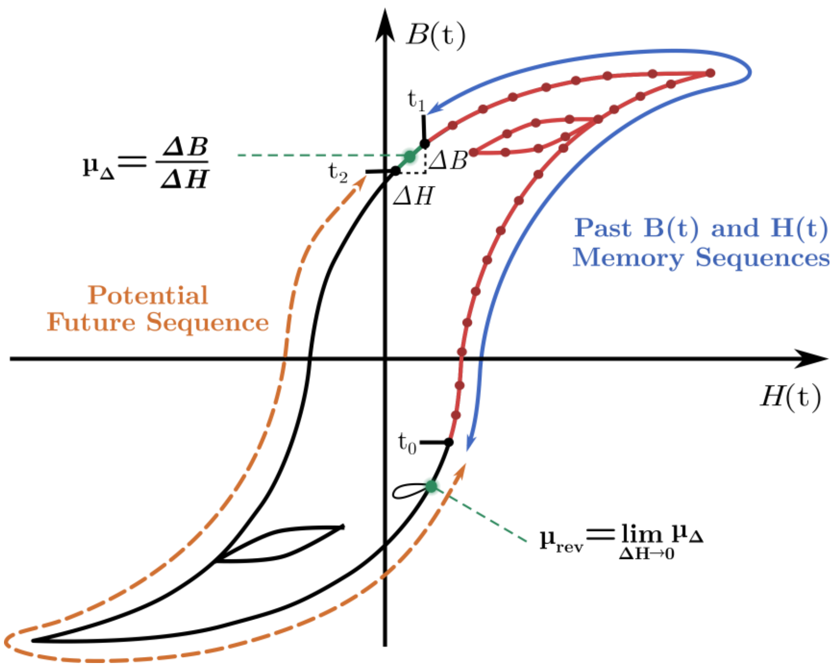
$$H_{t_1\to t_2} = function\,(B_{t_0\to t_1}(t), H_{t_0\to t_1}(t), B_{t_1\to t_2}, T)$$

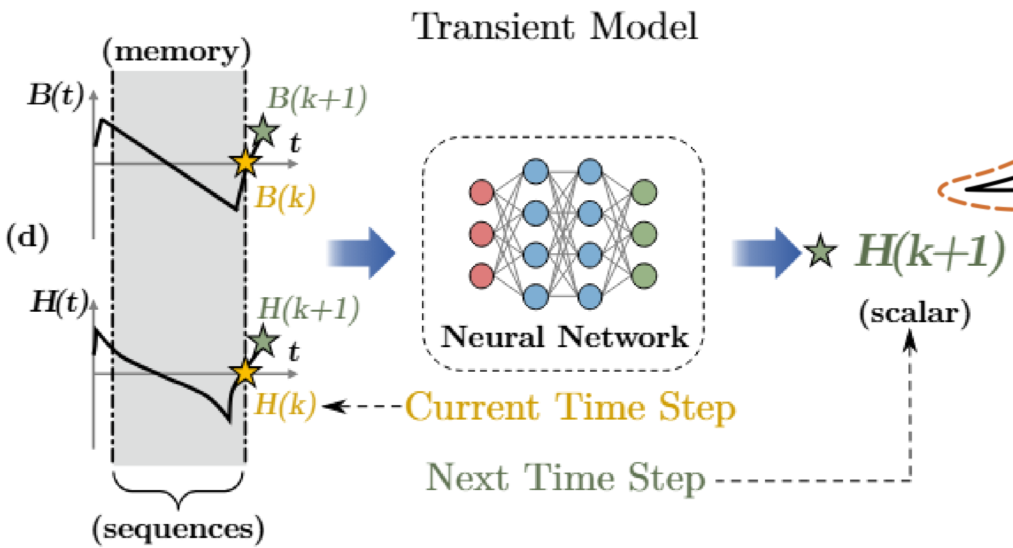


- Hyukjae Kwon, Shukai Wang, Haoran Li, et al. "**MagNetX: Extending the MagNet Database for Modeling Power Magnetics in Transient**," *TechRxiv*. December 11, 2024. Accepted to APEC 2025.

What we have developed (as tutorials)

Input: B(t) in the past 80 steps
H(t) in the past 80 steps
future B(t) from t_1 to t_2
Output: future H(t) from t_1 to t_2



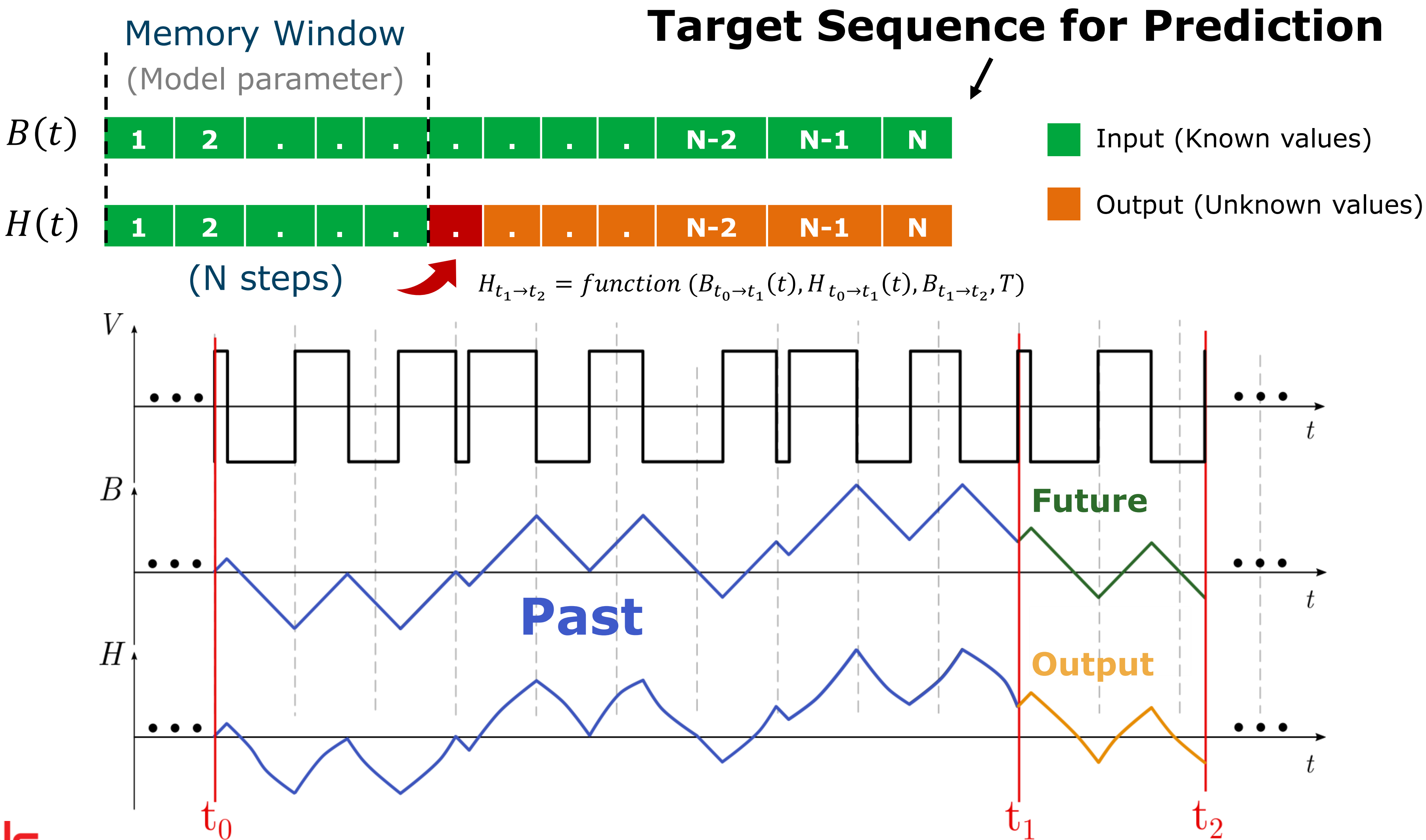


- Shukai Wang, Hyukjae Kwon, Haoran Li, et al. "**MagNetX: Foundation Neural Network Models for Simulating Power Magnetics in Transient**." *TechRxiv.* December 11, 2024. Accepted to APEC 2025.

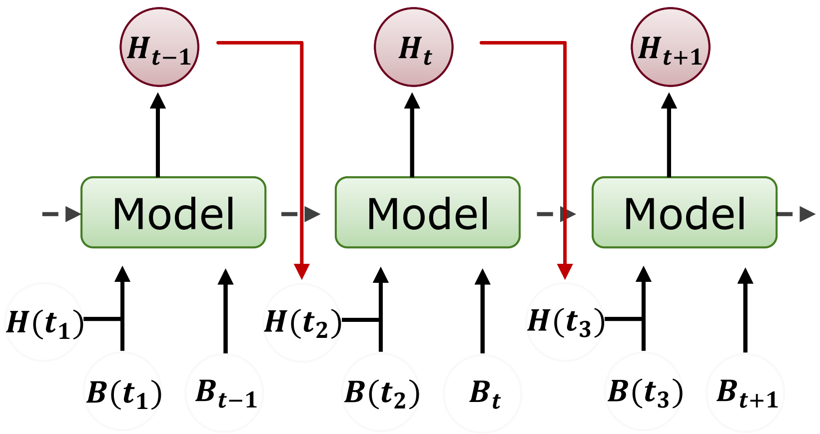
IEEE POWER ELECTRONICS SOCIETY
Powering a Sustainable Future

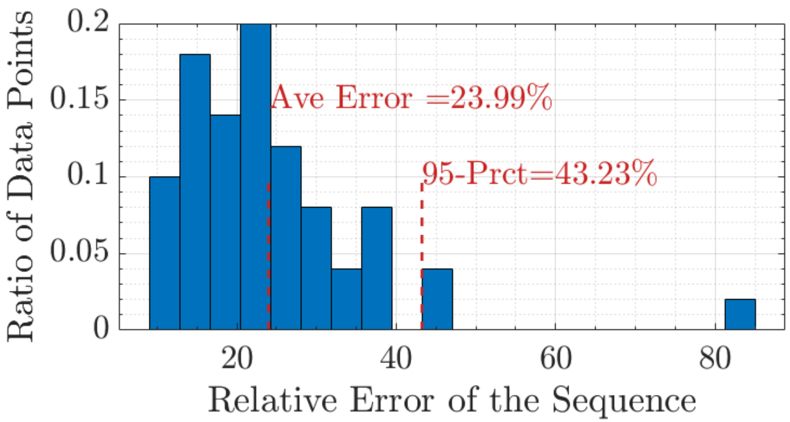
IEEE
Advancing Technology for Humanity

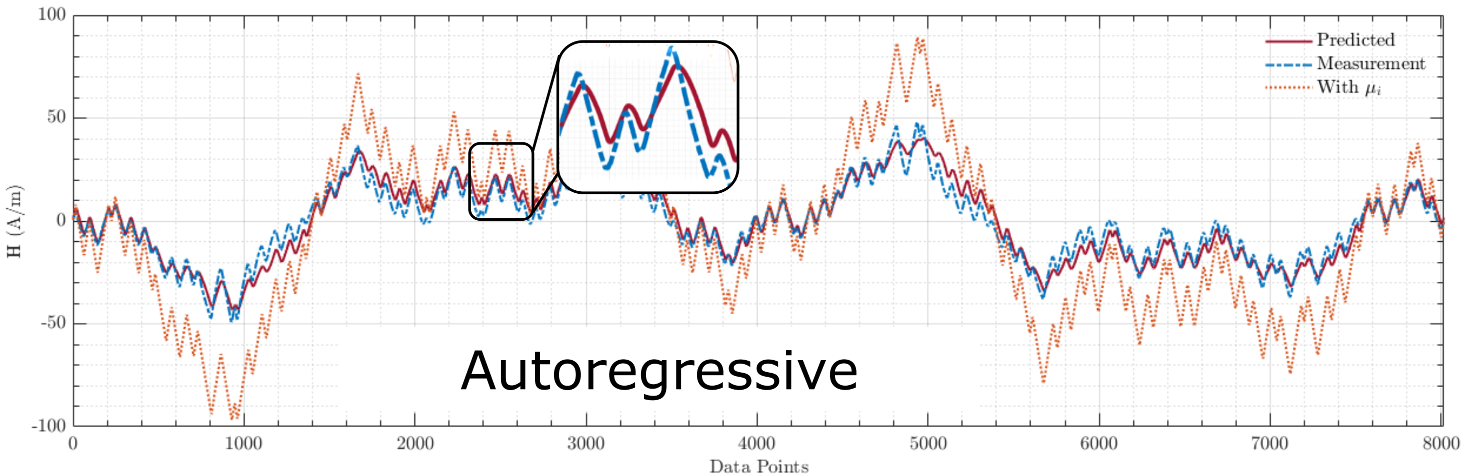
Model Testing



Autoregressive Prediction

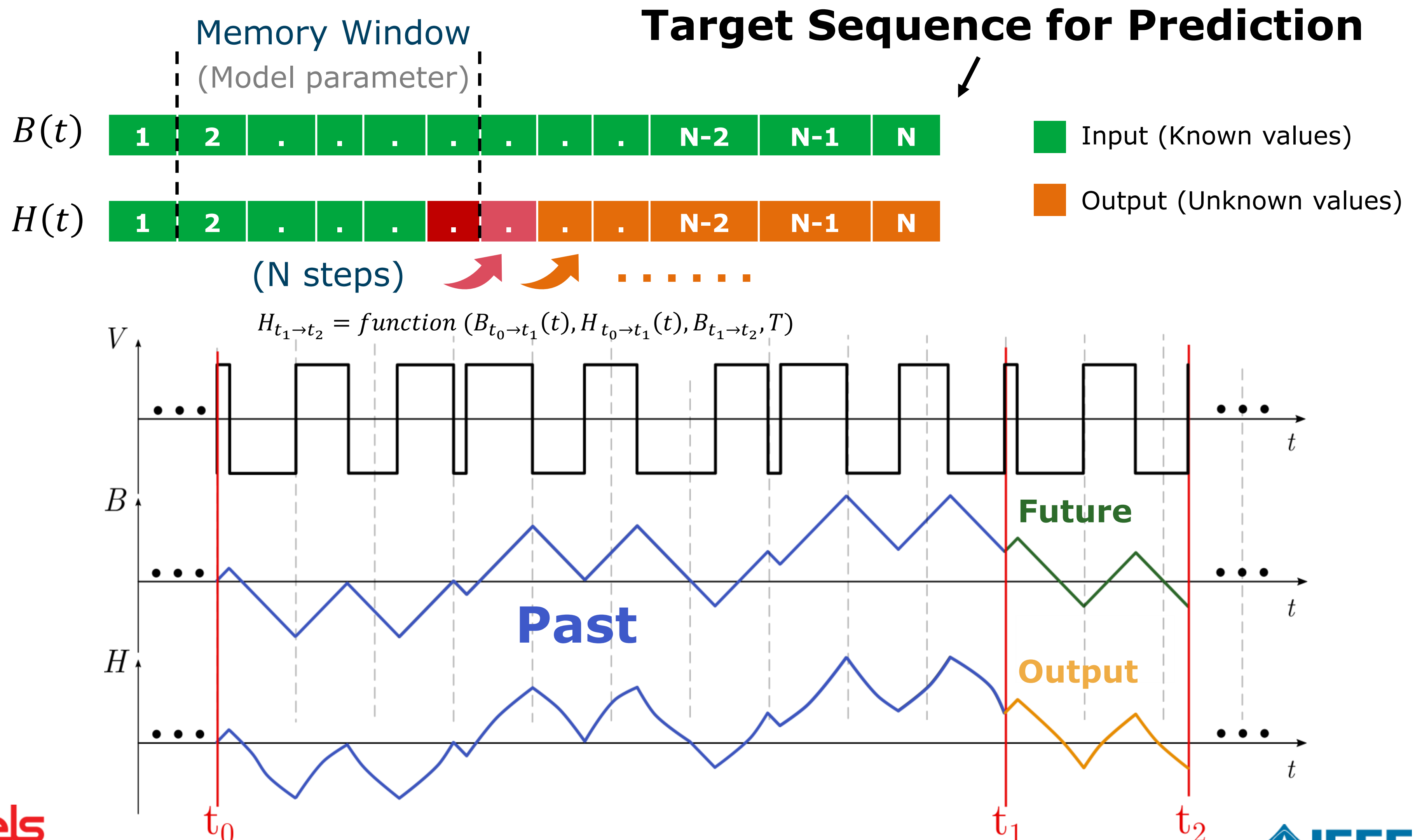






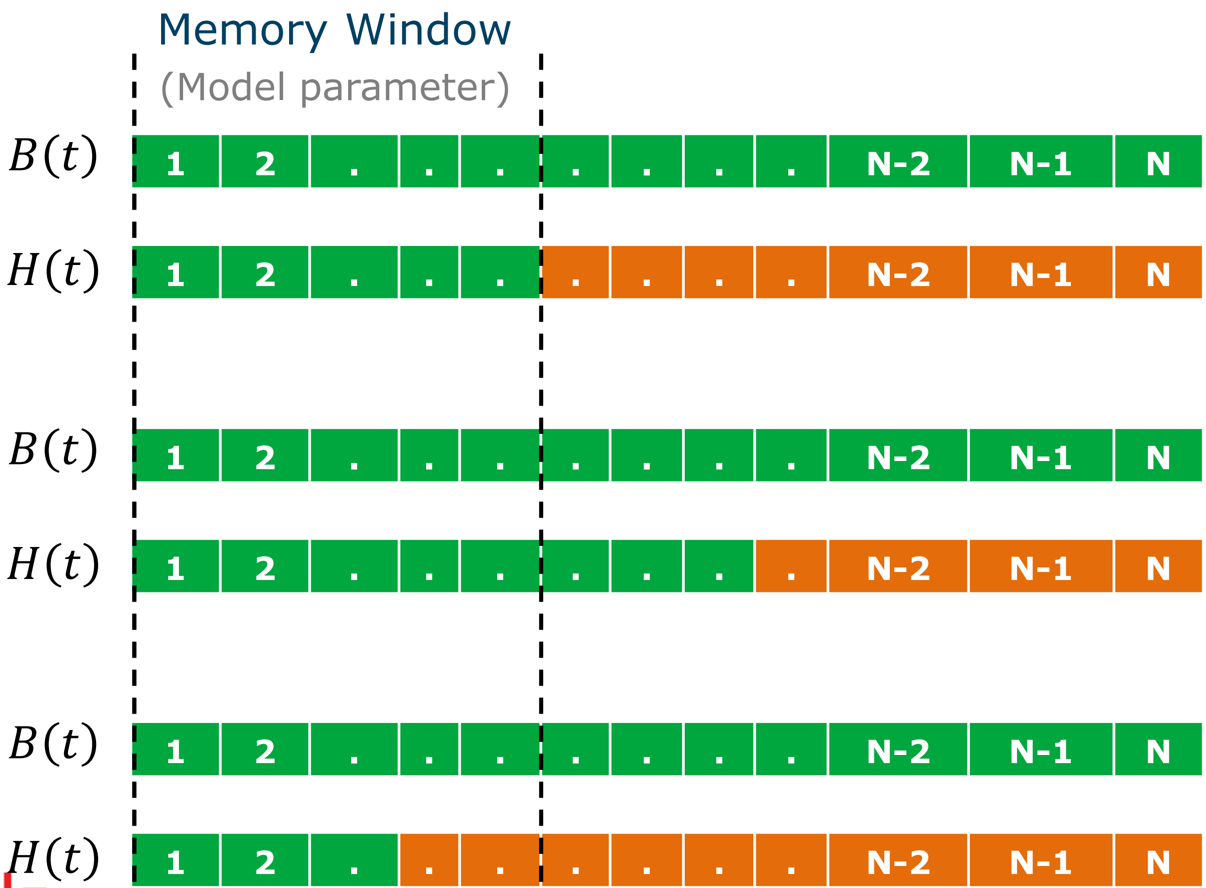
Autoregressive

Model Testing



Arbitrary 1. Memory Length Mismatch

What if memory window ≠ input time steps?



Input (Known values)

Output (Unknown values)

Ideal case:

Input = Window size

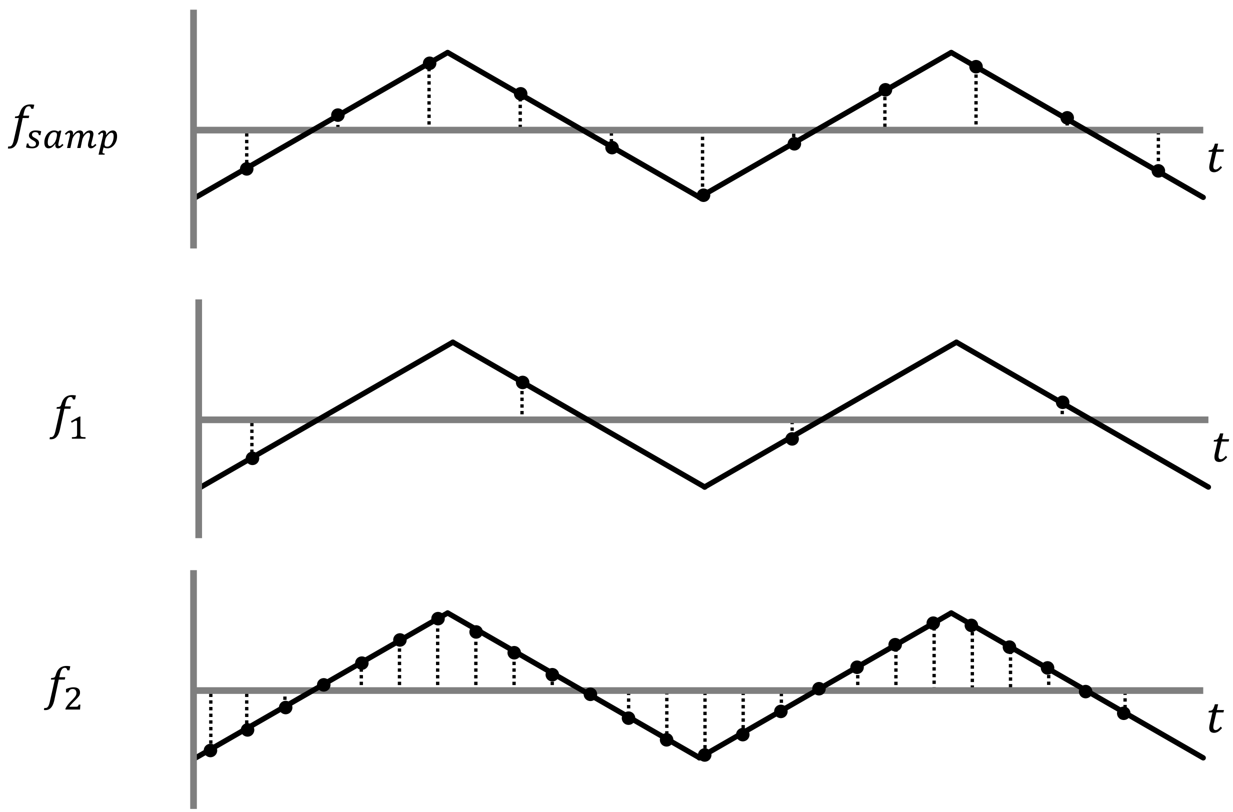
Input > Window size

Sequence Length Mismatch

Input < Window size

Arbitrary 2. Sampling frequency Mismatch

What if sampling frequency is different?



Model sampling f
16 MHz

Lower sampling f
Test dataset

Higher sampling f
Test dataset

Foundation Model Evaluation with Testing Data

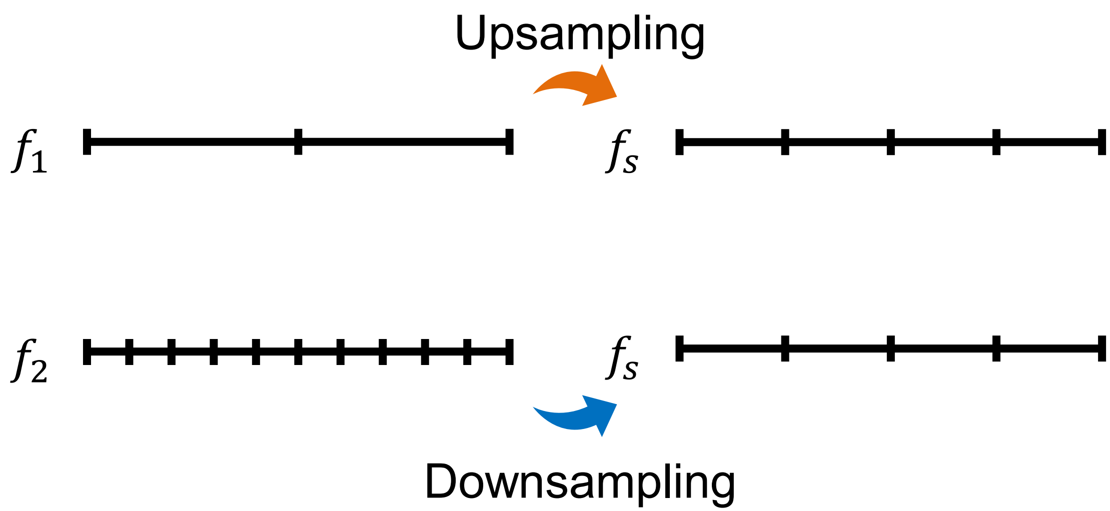
Foundation Model

- Frequency agnostic
- Universal time step

① Different sampling frequency

$f_{input} \; != \; f_{samp}$

- $f_1 < f_s$: Upsample
- $f_2 > f_s$: Downsample



Foundation Model Evaluation with Testing Data

MagNetX GitHub: https://github.com/PaulShuk/MagNetX

① Different sampling frequency

Foundation Model

Frequency agnostic

Universal time step

Demo_LSTM_Evaluation.ipynb

def get_dataset(…):

```
# 1. Sampling frequency difference: Interpolation if sampling frequency mismatch exists
if sample_freq_model != sample_freq_test:
    # Time axes for original and interpolated data
    t_B_old = np.arange(B_len) * sample_time_test
    t_H_old = np.arange(H_len) * sample_time_test
    t_H_true_old = np.arange(H_true_len) * sample_time_test

    t_B_new = np.arange(0, total_time_B, sample_time_model / time_factor)
    t_H_new = np.arange(0, total_time_H, sample_time_model / time_factor)
    t_H_true_new = np.arange(0, total_time_H_true, sample_time_model / time_factor)

    # Interpolate each sample
    B_inter = np.array([np.interp(t_B_new, t_B_old, B[i]) for i in range(B.shape[0])])
    H_inter = np.array([np.interp(t_H_new, t_H_old, H[i]) for i in range(H.shape[0])])
    H_true_inter = np.array([np.interp(t_H_true_new, t_H_true_old, H_true[i]) for i in range(H_true.shape[0])])

    # Downsample after interpolation
    B_seq = B_inter[:, ::time_factor]
    H_seq = H_inter[:, ::time_factor]
    H_true_seq = H_true_inter[:, ::time_factor]

    data_length_test = H_seq.shape[1]
    print("data_length_test:", data_length_test)
else:
    # If no interpolation is needed
    B_seq, H_seq, H_true_seq = B, H, H_true
```

1 Handling input with different **sampling rates**

Foundation Model Evaluation with Testing Data

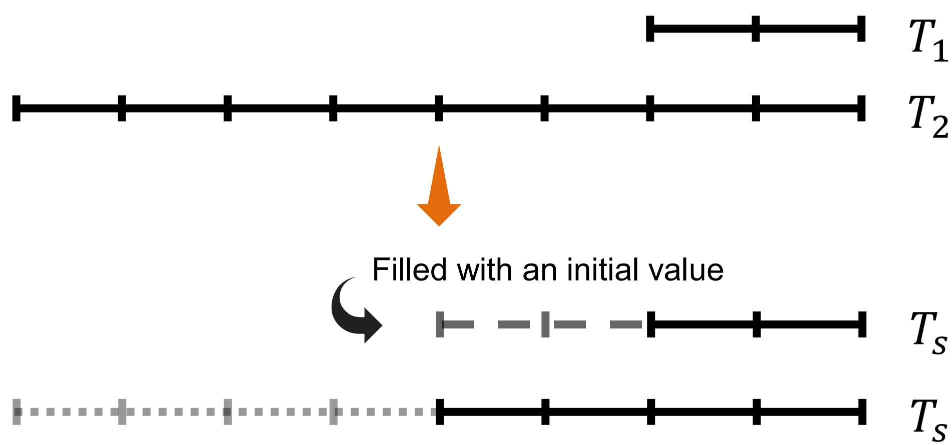
Foundation Model

Frequency agnostic

Universal time step

② Different total time steps

$$T_{input} \; != \; T_{samp} \quad \begin{cases} T_1 < T_s \\ T_2 > T_s \end{cases}$$



Foundation Model Evaluation with Testing Data

Foundation Model

Frequency agnostic

Universal time step

② Different total time steps

Demo_LSTM_Evaluation.ipynb

def get_dataset(…):

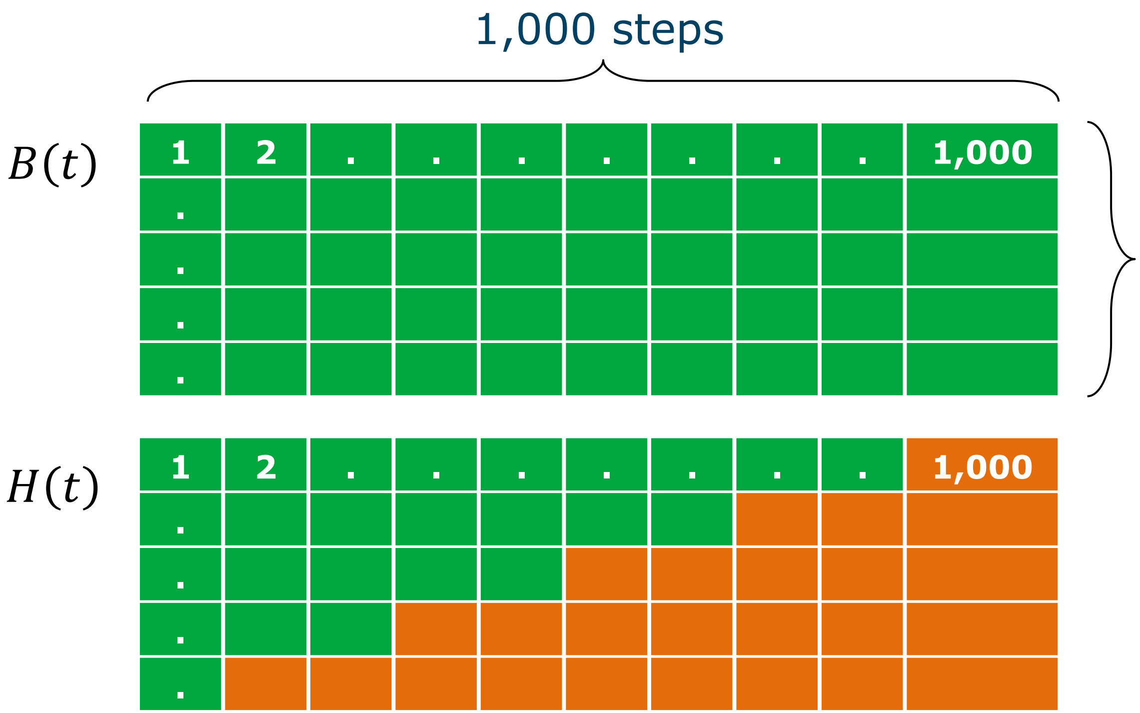
```
# 2. Time step difference: Adjust length of the sequence if different from model input
if data_length_model > data_length_test:
    # Pad the front with the first value if test data is shorter
    B_ftr = B_seq[:, data_length_test:]
    H_ftr = H_true_seq[:, data_length_test:]

    pad_len = data_length_model - data_length_test
    B_seq = np.concatenate([np.tile(B_seq[:, [0]], (1, pad_len)), B_seq[:, :data_length_test]], axis=1)
    H_seq = np.concatenate([np.tile(H_seq[:, [0]], (1, pad_len)), H_seq[:, :data_length_test]], axis=1)
else:
    # Truncate if test data is longer
    B_ftr = B_seq[:, data_length_test:]
    H_ftr = H_true_seq[:, data_length_test:]
    B_seq = B_seq[:, data_length_test - data_length_model:data_length_test]
    H_seq = H_seq[:, data_length_test - data_length_model:data_length_test]
```

2 **Handling input with different memory length**

Testing Dataset for Evaluation

- Input (Known values)
- Output (Unknown values)

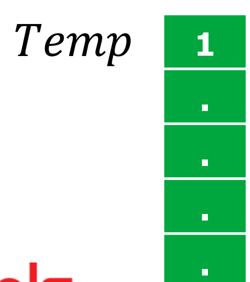


105 sequences

- 7 frequencies
- 3 temperatures
- 5 input/output ratio

Sampling frequency

- 16 MHz
- 62.5 ns



Material

- 3C90

Frequency

- 50 kHz
- 80 kHz
- 125 kHz
- 200 kHz
- 320 kHz
- 500 kHz
- 800 kHz

Temperature

- 25 ℃
- 50 ℃
- 70 ℃

Known : Unknown

- 90%:10%
- 70%:30%
- 50%:50%
- 30%:70%
- 10%:90%

Testing dataset is randomly sampled from released data

Testing and Evaluation Tutorial

- Please visit MagNet Challenge GitHub: https://github.com/minjiechen/magnetchallenge-2
- Or MagNetX GitHub: https://github.com/PaulShuk/MagNetX

2: Network Testing

This tutorial demonstrates how to test the double LSTM-based model for the sequence-to-scalar future hysteresis step prediction. The test set seqeunces include data that has the same sampling time steps as the training dataset. However the code is able to decipher the sequence length (provided that it has a sampling frequency), and upsample or downsample as needed to fit the model criteria. If the sequence is longer than the designed memory, then only the most recent time steps (amount to the total training memory time) are taken. If the data length is shorter, then the extra empty memory will take the constant of the most distant memory data point. The test set consists of multiple frequencies (7), temperatures (3), and 5 different combinations of past and future lengths, totaling 105 sequences, each with 1,000 time steps.

Step 0: Import Packages

In this step we import the important packages that are necessary for the testing.

```python
import torch
from torch import Tensor
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import random
import numpy as np
import json
import h5py
import math
import csv
import time
import matplotlib.pyplot as plt
```

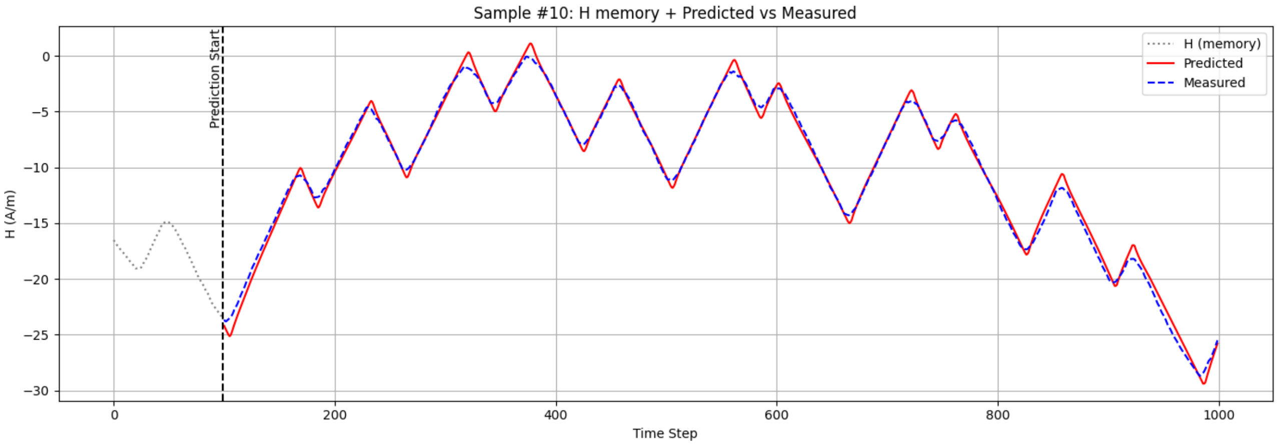
[8] Python

IEEE POWER ELECTRONICS SOCIETY
Powering a Sustainable Future

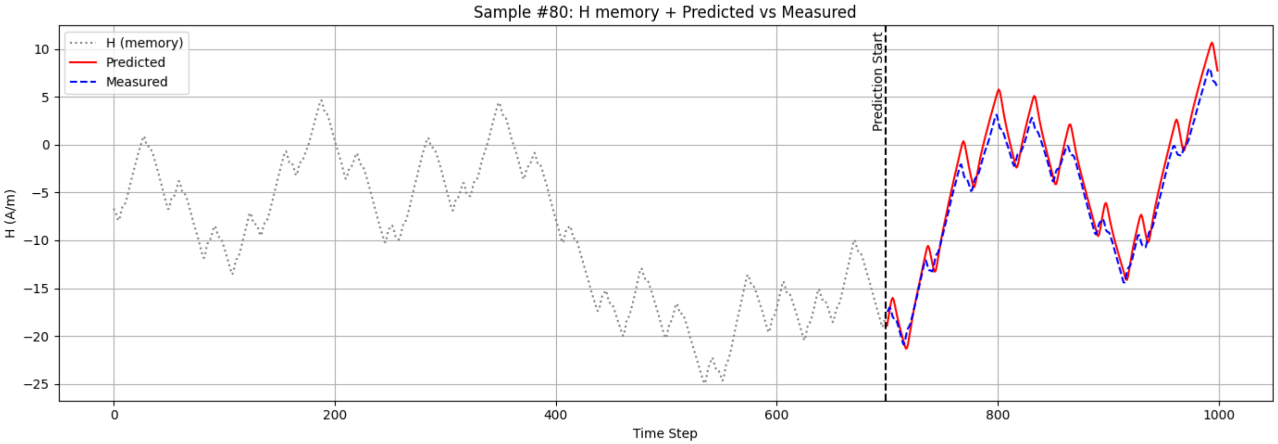
IEEE Advancing Technology for Humanity

Testing and Evaluation Results

- H sequence prediction vs measurement



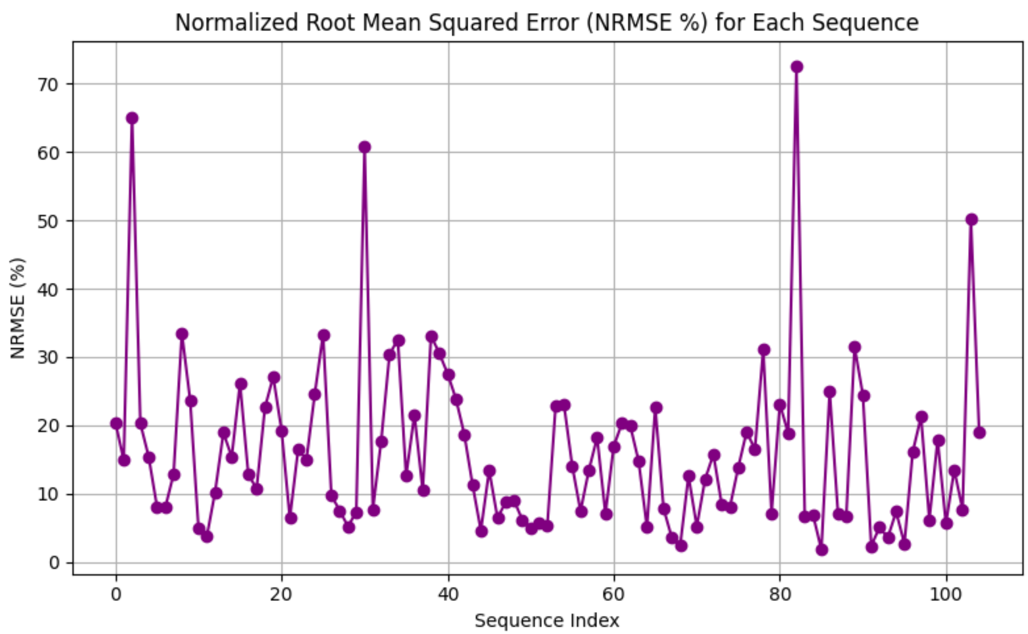
[3C90, 200 kHz, 50 ℃, 10%:90% format]

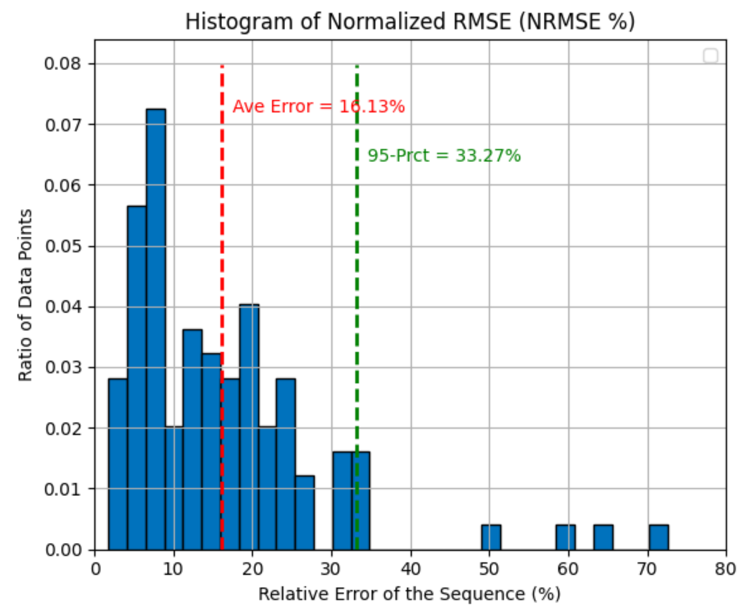


[3C90, 500 kHz, 70 ℃, 70%:30% format]

Testing and Evaluation Results

- NRMSE of H sequences






```
Memory length 100: Mean NRMSE = 18.74% (0 - 20)
Memory length 300: Mean NRMSE = 20.67% (21 - 41)
Memory length 500: Mean NRMSE = 12.25% (42 - 62)
Memory length 700: Mean NRMSE = 15.57% (63 - 83)
Memory length 900: Mean NRMSE = 13.40% (84 - 104)
```

IEEE POWER ELECTRONICS SOCIETY
Powering a Sustainable Future

IEEE
Advancing Technology for Humanity

Testing and Evaluation Results

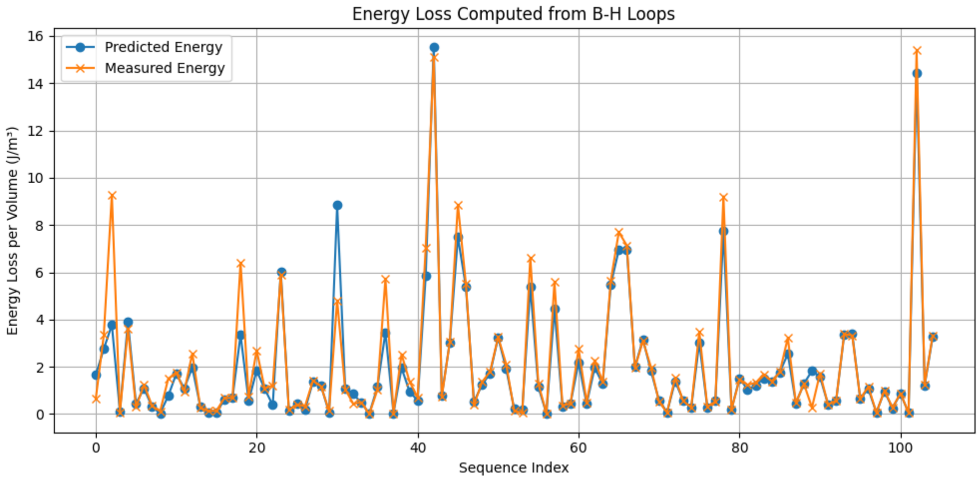
- Energy loss from B-H loops

$$\int H\, dB$$

$$\begin{bmatrix} dB: [V \cdot s/m^2] \\ H: [A/m] \\ HdB: [J/m^3] \end{bmatrix}$$

```python
def compute_total_energy(B, H):
    """
    Computes energy loss per unit volume [J/m³] for each sequence in B and H.
    B, H: numpy arrays of shape (N_sequences, N_timesteps)
    Returns: energy loss per sequence (N_sequences,)
    """
    dB = np.diff(B, axis=1)                         # shape: (N, T-1)
    H_mid = (H[:, :-1] + H[:, 1:]) / 2              # shape: (N, T-1)
    energy_density = np.sum(H_mid * dB, axis=1)   # shape: (N,)
    return np.abs(energy_density)                   # abs to ensure positive energy
```

$\int H\, dB$ (open loop): Energy exchanged ≠ total core loss (cycle not closed)



NRMSE
39.78 %

IEEE POWER ELECTRONICS SOCIETY
Powering a Sustainable Future

IEEE
Advancing Technology for Humanity

Conclusion

Training data

- 10 training materials
- Lots of long B(t)-H(t) pairs
- Temperature

Testing data

- 5 new materials
- Temperature
- Practicing: plenty of long B(t) and H(t) pairs

Input/Output

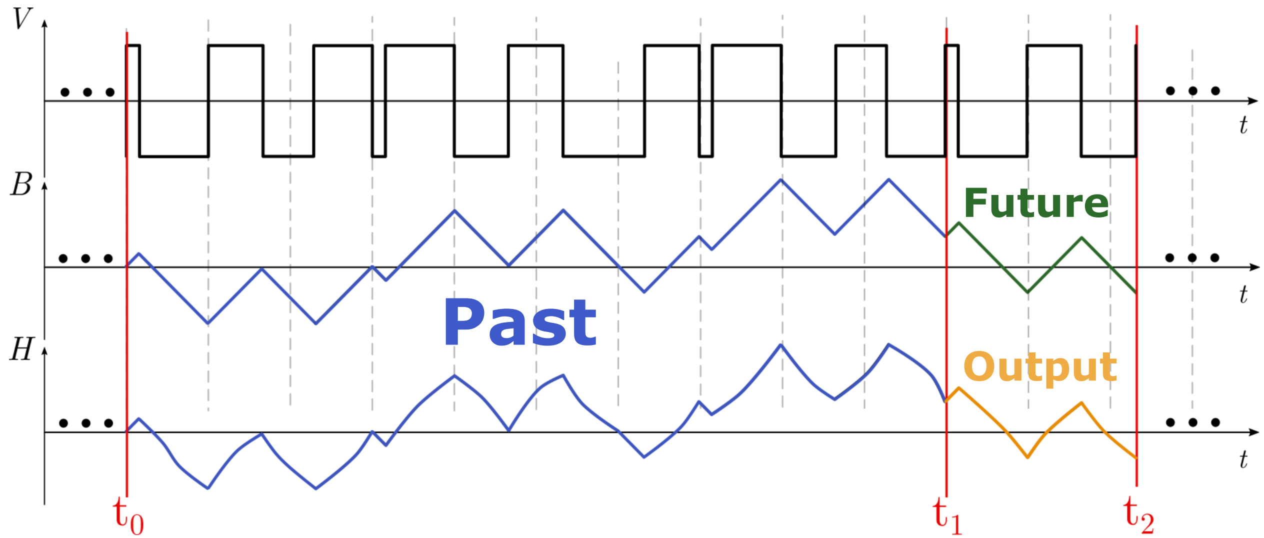
- Past B(t) and H(t)
- Future B(t)

↓

- Future H(t)

Evaluate

- Total core loss from t_1 to t_2
- H(t) RMS error from t_1 to t_2
- Model size (# of parameters)



$$H_{t_1 \to t_2} = \mathit{function}\left(B(t),\ H(t),\ B_{t_1 \to t_2},\ T \right)$$

- Hyukjae Kwon, Shukai Wang, Haoran Li, et al. "**MagNetX: Extending the MagNet Database for Modeling Power Magnetics in Transient**," *TechRxiv*. December 11, 2024. Accepted to APEC 2025.

GitHub: Tutorial_3

- MagNetX GitHub: https://github.com/PaulShuk/MagNetX

- Tutorial_3
  - 3C90_Testing_padded.h5
    - Provide input sequences (B, H, T)
    - 105 sequences with varying past and future lengths
  - 3C90_Testing_true.h5
    - Provide ground truth data of H sequence
  - Demo_LSTM_Evaluation.ipynb
    - Main test code to evaluate models
  - Model_LSTM.sd
    - Model trained using 3C90 data
  - Normalization_Params.json
    - Normalization Parameters made at Tutorial_1/ 3C90_Testing
  - Output
    - meas.csv
    - pred.csv
      - Measured data and predicted data created at Demo_LSTM_Evaluation.ipynb