Implementation-driven Mathematics

My standpoint: Prediction via filtering

- stochastic generation (diffusion models)
- prediction via filtering

$\Rightarrow$ duality relationship: essentially doing the same thing (I think).

- Diffusion models: Learn the process of adding noise to data and then gradually denoising it to reconstruct the original structure.
- Prediction via filtering: Suppress noise and learn the underlying deterministic structure. (explain our novel filtering from the next)

Nonparametric data-driven filter

Nonparametric data-driven filter

Weight function of the moving average

$$\Psi(t) = \underbrace{\left(d_1 \cos\left(\frac{t}{\pi r_1}\right) + d_2 \cos\left(\frac{t}{\pi r_2}\right)\right)}_{\textit{passing freq}} \underbrace{\frac{(w-t)^c}{w^c}}_{\textit{band width}} \quad \text{for} \quad t \in [0, w]$$

$d_1, d_2, r_1, r_2, c, w > 0$: learnable parameters

Aim: construct the objective function

$\Omega \subset \mathbb{Z}$: prescribed finite sequence (input space).
Original data: $y : \Omega \to \mathbb{R}$,

$$\text{filtered data:} \quad y^*(t) := \sum_{t'=0}^{w} y(t-t')\Psi(t') \quad \text{for} \quad t \in \Omega.$$

Point: the band width cannot have a unique threshold due to the Fourier uncertainty principle!!

Range quantization

Choose $\{a_k\}_{k=1}^K \subset \mathbb{R}$ such that $a_1 < a_2 < \cdots < a_K$.
Using this $\{a_k\}_{k=1}^K$, we classify the filtered data y^* as follows:

Range quantization

$$\tilde{y}(t) := \text{argmin}_{a \in \{a_k\}_{k=1}^K} |y^*(t) - a| \quad \text{for} \quad t \in \Omega.$$

Let σ_n^L $(n = 1, 2, \cdots, N_L)$ be a permutation operator such that

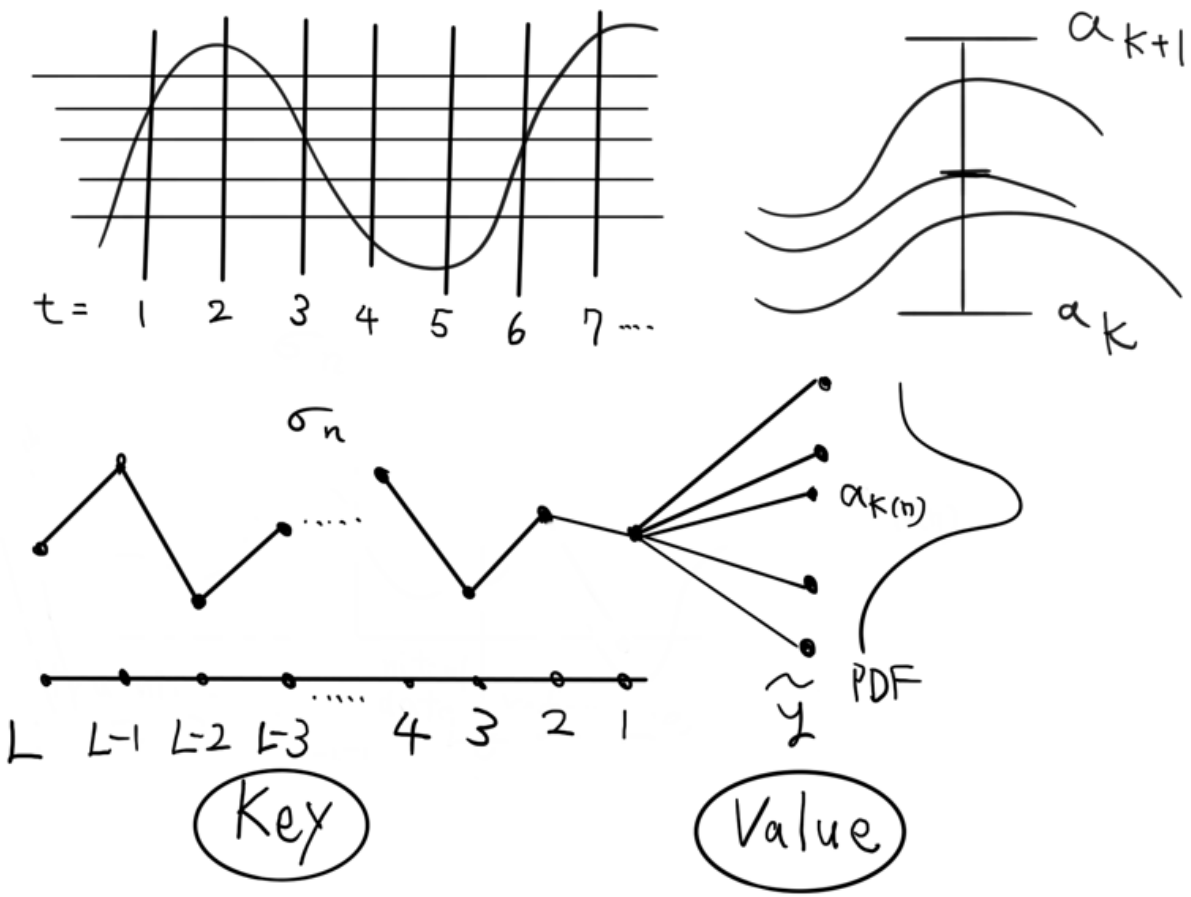
Permutation operator

$$\sigma_n^L : \{1, 2, \cdots, L\} \to \{a_1, a_2, \cdots, a_{K-1}, a_K\} \quad (\ell \mapsto \sigma_n^L(\ell)) \quad \text{with } \sigma_n^L \neq \sigma_{n'}^L \ (n \neq n').$$

n means just labeling and there exists a suitable N_L such that

$$\begin{cases} \text{For any } t \in \Omega, \text{ there is } n \in \{1, \cdots, N_L\} \text{ such that} \\ \qquad \sigma_n^L(\ell) = \tilde{y}(t - \ell) \text{ for } \ell = 1, 2, \cdots, L, \\ \text{For any } n \in \{1, \cdots, N_L\} \text{ there is } t \in \Omega \text{ such that} \\ \qquad \sigma_n^L(\ell) = \tilde{y}(t - \ell) \text{ for } \ell = 1, 2, \cdots, L. \end{cases}$$

Image of quantization



- Detect $L = L_{max}, L_{max} - 1, \cdots, L_{min} + 1, L_{min}$
- Roughly, optimize the learnable params so that deviations of each PDF values become small. (but not precise, due to nonparametric)

Aim: reduce the indefinite factor

Nonparametric setting

- $\Sigma_n^L := \{t \in \Omega : \sigma_n^L(\ell) = \tilde{y}(t-\ell) \quad \text{for} \quad \ell = 1, 2, \cdots, L\}$
- $\Pi_n^L(k) := \{t \in \Omega : \sigma_n^L(\ell) = \tilde{y}(t-\ell) \quad \text{for} \quad \ell = 1, 2, \cdots, L \quad \text{and} \quad \tilde{y}(t) = a_k\}$

Maximize the following objective function under the condition that the autocorrelation between the original and the filtered data is not small.

Objective function

$$\frac{\displaystyle\sum_{L=L_{min}}^{L_{max}} \#\left\{n : \text{there is a } k \in \{1, 2, \cdots, K\} \text{ such that } \frac{\#\Pi_n^L(k)}{\#\Sigma_n^L} \geq \gamma\right\}}{\displaystyle\sum_{L=L_{min}}^{L_{max}} N_L}$$

$\gamma \in (0, 1]$ is the match rate of a_k for each σ_n^L. Here, γ, as well as L_{max} and L_{min} are prescribed parameters in the numerical implementation.

Why Recurrent Neural Network is excellent?

Math framework for ML

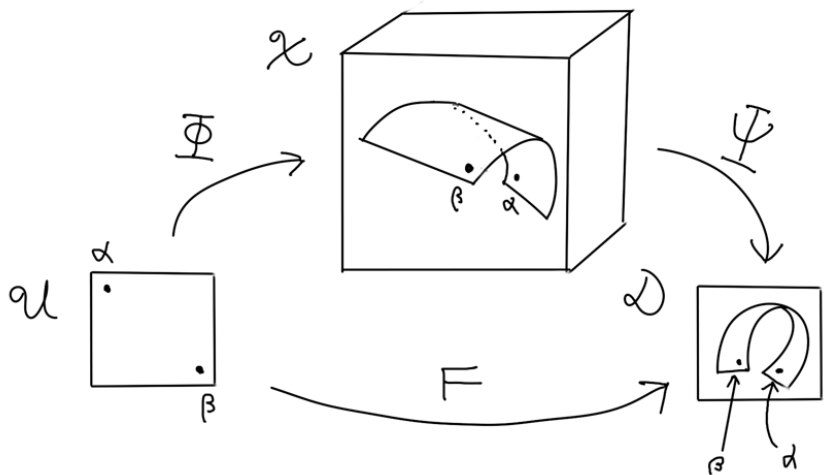
- input dataset $\mathcal{U}$, output space $\mathcal{D}$ and feature space $\mathcal{X}$
- $F : \mathcal{U} \to \mathcal{D}$: underlying system, might be disconti, really want to know
- $\dim(\mathcal{U}), \dim(\mathcal{D}) \ll \dim(\mathcal{X})$ (except for Encoder)

Example:

- $u \in \mathcal{U}$: picture ($\mathcal{U} \subset 1000px \times 1000px \times 3colors$ dimension)
- $d = 1$: pic is cat, $d = 0$: pic is NOT cat ($\mathcal{D} \cong \mathbb{R}$)

Purpose of ML in pure math description

Find continuous maps $\Phi_{in} : \mathcal{U} \to \mathcal{X}$ and $\Phi_{out} : \mathcal{X} \to \mathcal{D}$ such that
$|\Phi_{out} \circ \Phi_{in}(u) - d|$ is small enough for $\forall (u, d) \in \mathcal{U} \times F(\mathcal{U})$.



Mathematical important question for ML

There may exist robust structures (independent of Φ_{in}) such that the following holds: there exist $(u_1, d_1), (u_2, d_2) \in \mathcal{U} \times F(\mathcal{U})$ satisfying $|u_1 - u_2| \gtrsim 1$ such that the corresponding feature vectors must be close to each other.

How can we structure this into Φ_{in} with $|\Phi_{in}(u_1) - \Phi_{in}(u_2)| \ll 1$?

Mathematical point: Opposite direction $\gtrsim$ and $\ll$

To address this question, maximize the following value (measure of nonlinearity):

$$\frac{|\Phi_{in}(u) - \Phi_{in}(v) - D\Phi_{in}(v)(u - v)|}{|u - v|} \quad \text{for any} \quad u, v \in \mathcal{U}$$

It is well-known that Deep Neural Network achieves large value of it.

- See also: Amari, Information geometry and its applications (2016)

Mathematical important question for ML

There may exist robust structures (independent of Φ_{in}) such that the following holds: there exist $(u_1, d_1), (u_2, d_2) \in \mathcal{U} \times F(\mathcal{U})$ satisfying $|u_1 - u_2| \gtrsim 1$ such that the corresponding feature vectors must be close to each other.

How can we structure this into Φ_{in} with $|\Phi_{in}(u_1) - \Phi_{in}(u_2)| \ll 1$?

Examples: convolutional NN, ESP, Category-theoretic structures in encoder–decoder.

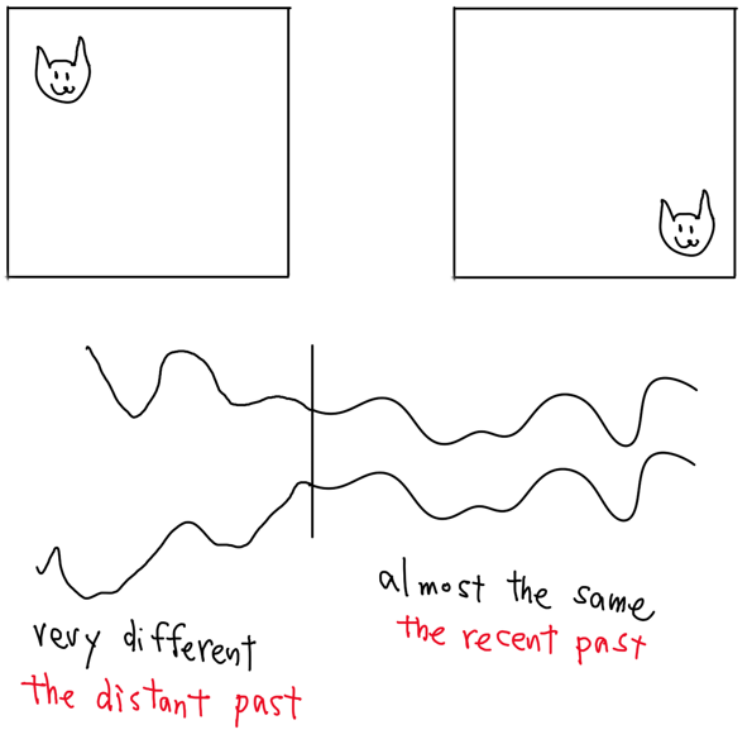


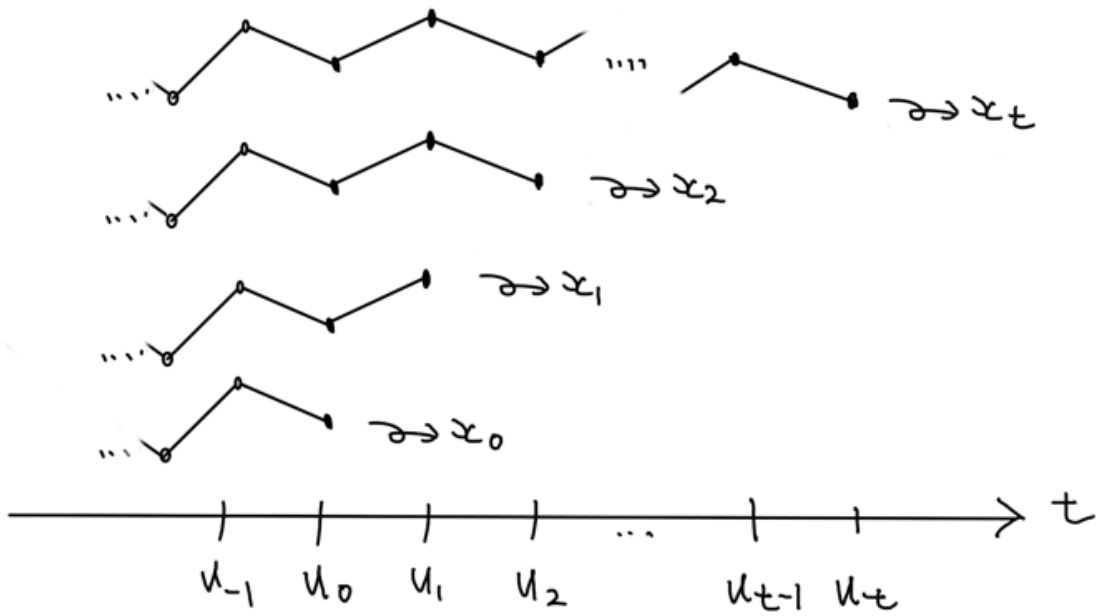
Figure: Left cat is α, Right cat is β.

ESP: Echo state property (for time series data)

The tricky point is the following definition: For $\Phi_{in} : \mathcal{U} \times \mathcal{X} \to \mathcal{X}$, let

$$x_t^1 = \Phi_{in}(u_t, \cdot) \circ \cdots \circ \Phi_{in}(u_2, \cdot) \circ \Phi_{in}(u_1, x_0^1),$$
$$x_t^2 = \Phi_{in}(u_t, \cdot) \circ \cdots \circ \Phi_{in}(u_2, \cdot) \circ \Phi_{in}(u_1, x_0^2).$$

Feature vectors x_0^1 and x_0^2 are expressing time series before $t = 0$.



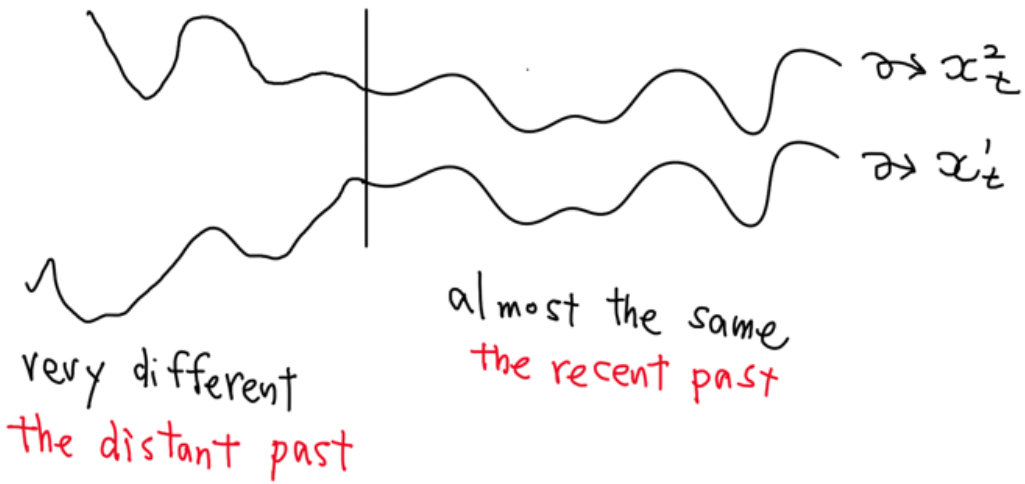
Theorem

Assume there is $\lambda \in (0,1)$ such that

$$|\Phi_{in}(u, x^1) - \Phi_{in}(u, x^2)| \leq \lambda |x^1 - x^2|$$

for $(u, x^1), (u, x^2) \in \mathcal{U} \times \mathcal{X}$ with $x_1 \neq x_2$. Then for any $x_0^1, x_0^2 \in \mathcal{X}$, we have

$$|x_t^1 - x_t^2| \to 0 \quad (t \to \infty).$$



The proof is obvious. We have the following for $t \to \infty$:

$$|x_{t+1}^1 - x_{t+1}^2| \leq |\Phi_{in}(u_t, x_t^1) - \Phi_{in}(u_t, x_t^2)| \leq |x_t^1 - x_t^2| \leq \cdots \leq \lambda^t |x_0^1 - x_0^2| \to 0.$$

Remark: probabilistic ESP (LSTM $\approx$ RNN)

$$\begin{aligned}
\text{input gate:} \quad & i_t = tanh(W_{datain} U_t) \\
\text{forget gate:} \quad & f_t = h(W_{dataforget} U_t + bias) \\
\text{memory cell:} \quad & c_t = i_t + f_t \odot c_{t-1} \\
\text{Ridge regression:} \quad & U_{t+1} \simeq W_{out} c_t
\end{aligned}$$

$$\text{induction arg:} \quad c_t = \sum_{s=0}^{t} \left(i_s \prod_{r=s+1}^{t} (\odot f_r) \right), \quad \prod_{r=t+1}^{t} f_r := 1.$$

Now assume that the following probability:

$$\gamma := P(f_r \cdot e_j \neq 0) \quad (0 < \gamma < 1)$$

is independent of j and r, where e_j is the j'th unit vector. Also assume f_r and $f_{r'}$ $(r \neq r')$ are probabilistically independent. Then we have ESP:

$$P\left(\prod_{r=s+1}^{t} f_r \cdot e_j \neq 0 \right) = \prod_{r=s+1}^{t} P(f_r \cdot e_j \neq 0) = \gamma^{t-s} \to 0 \quad (s \ll t \to \infty).$$

Remark (linear structure might not be good)

ML framework

Recall that, the underlying system F could be discontinuous, on the other hand, Φ_{in} and Φ_{out} are continuous maps.

From this ML framework, We see that Φ_{in} and Φ_{out} may not be appropriate to be expressed as Fourier sums (or, more generally, any kind of sums). The evidence is the following: For the Fourier series of the indicator functions of several dimensional balls, we observe

- Gibbs phenomena,
- Pinsky phenomena 1993 (diverging at the origin),
- Kuratsubo phenomena 2010 (preventing pointwise convergence).

$$\Rightarrow |\Phi_{out} \circ \Phi_{in}(u) - d| \quad \text{cannot be small for} \quad \forall (u, d) \in \mathbb{Q} \cap \mathcal{U} \times F(\mathbb{Q} \cap \mathcal{U})$$

To understand Kuratsubo phenomena intuitively, see numerical computations in Section 7 in Kuratsubo-Nakai-Ootsubo (2010).

Why Reservoir computing is excellent?

Brief explanation of standard gradient descent

Add learnable parameters $W_{in} \in \mathbb{R}^{M_{in}}$ and $W_{out} \in \mathbb{R}^{M_{out}}$ into Φ_{in} and Φ_{out}.

ML model with learnable params

$$\Phi(W, u) := \Phi_{out}(W_{out}, \cdot) \circ \Phi_{in}(W_{in}, u), \quad W = (W_{in}, W_{out}) \in \mathbb{R}^{M_{in}} \oplus \mathbb{R}^{M_{out}}$$

We wish to choose W so that $|\Phi(W, u) - d|$ becomes small for almost any $(u, d) \in \mathcal{U} \times F(\mathcal{U})$. This intention can be reformulated using the energy:

Energy

$$E(W) := \frac{1}{2} \sum_{(u,d) \in \mathcal{U} \times F(\mathcal{U})} |\Phi(W, u) - d|^2$$

From

$$\partial_\epsilon E(W + \epsilon w)\big|_{\epsilon=0} = \nabla E(W) \cdot w,$$

we see that W should be updated iteratively in the direction of steepest descent, namely $w = -\nabla E(W)$.

Why Reservoir is excellent?

Reservoir: Φ_{out} is a linear transformation

Ridge regression: $W_{out} = R(\mathcal{U}, F(\mathcal{U}), W_{in})$ (composed of inverse matrix)

Substituting this explicit representation R into Φ_{out} gives

$$\Phi(W, u) = \Phi_{out}(R(\mathcal{U}, F(\mathcal{U}), W_{in}), \cdot) \circ \Phi_{in}(W_{in}, u).$$

Advantage of the Reservoir

On the subspace $\mathbb{R}^{M_{out}}$, the minimization is always attained, and moreover, the learnable parameters are reduced from W to W_{in}.

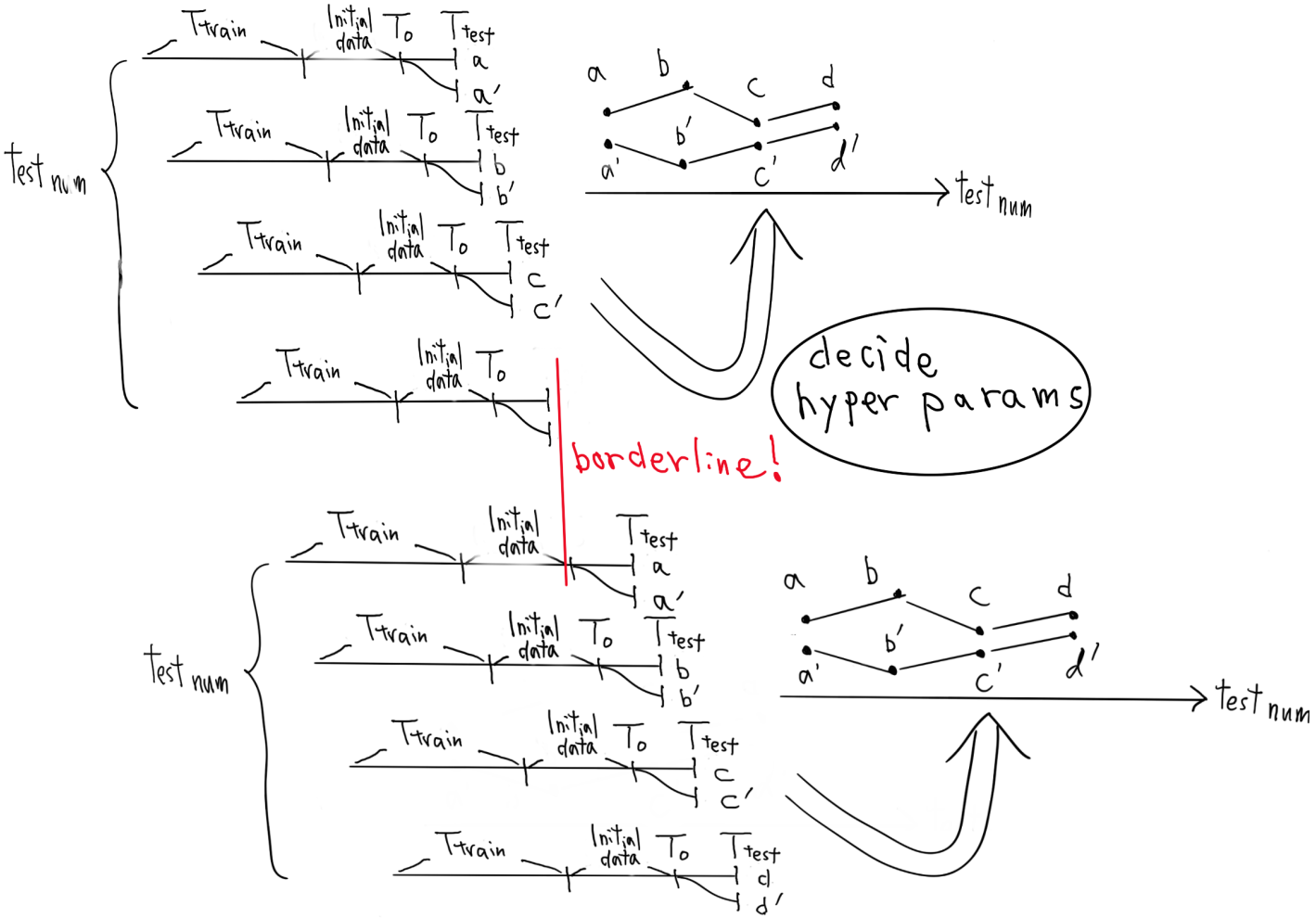
However, conventional backpropagation (i.e., standard gradient descent) encounters significant difficulty:

- Reason: R involves high-dimensional matrix inversion, making its explicit form prohibitively complicated.
- To overcome this difficulty: we adopt Bayesian opt (Optuna).
- Advantage: we can add model selection phase!!

ML Results
(Reservoir computing)

Learning framework: ElNino and MJO

Pearson correlation coefficient (PCC) between real data and prediction



Generalization performance evaluation: $T_{test} = 1, 2, \cdots$

Threshold of the correlation is 0.5 in this competition!!

El Nino prediction (Jinno-Mitsui-Nakai-Saiki-Y 2025)

Nino 3.4 (this time series expresses the strength of El Nino)

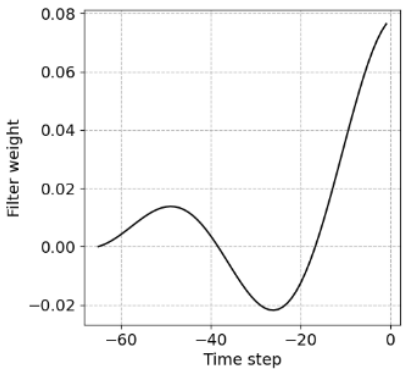
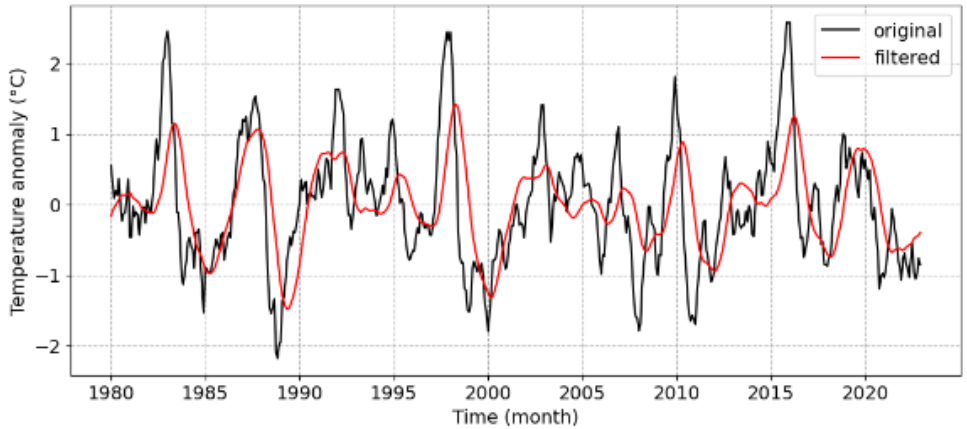
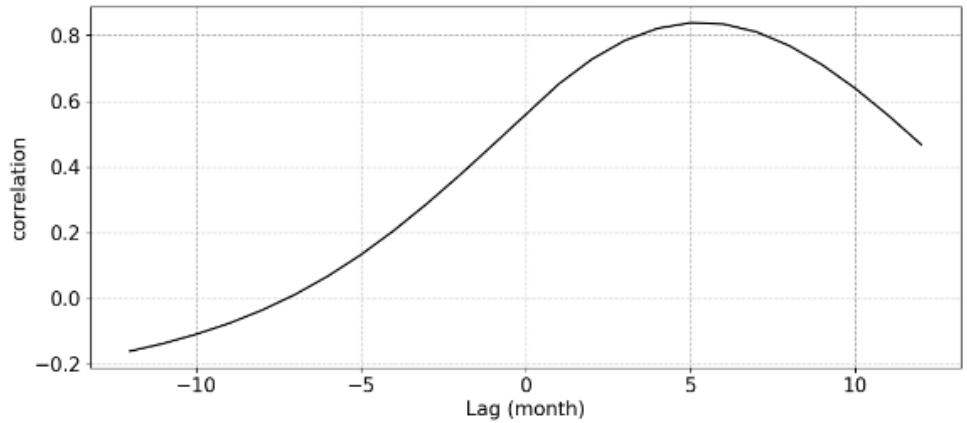


Figure: data-driven filter



(a)



(b)

(a): original (black) and filtered (red)

(b): Lag correlation between original and filtered : correlation > 0.8

Result (El Nino)

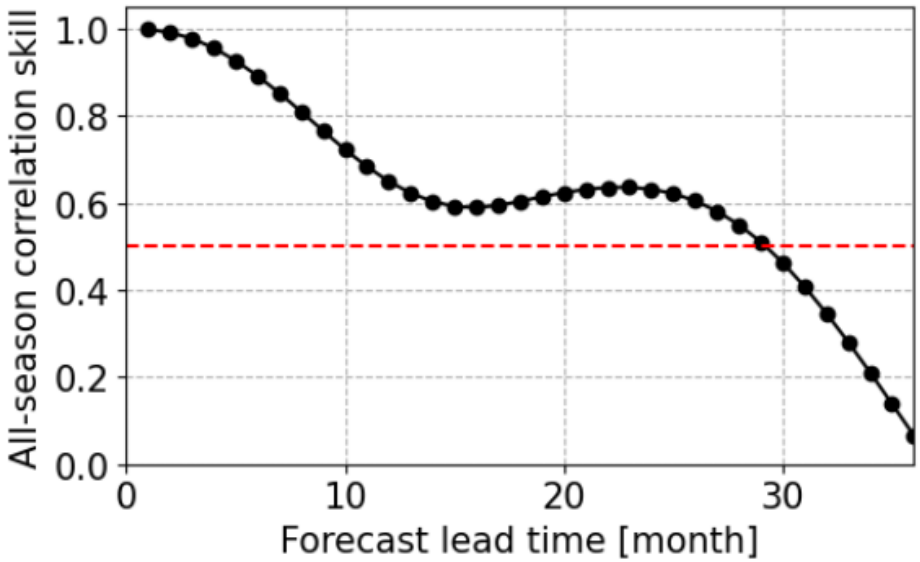


Figure: Prediction with our novel filter
(29 minus 5(lag)=24 months)

- Reservoir Hassanibesheli-Kurths-Boers (2022): 19 months
(but incorporating future values due to the band-pass filter)
- CNN Ham-Kim-Luo (2019): 17 months
- Physical model SINTEX-F: 12 months

Result (MJO prediction)

Nakai-Suematsu-Miura-Y-Takasuka-Jinno-Saiki (arXiv)

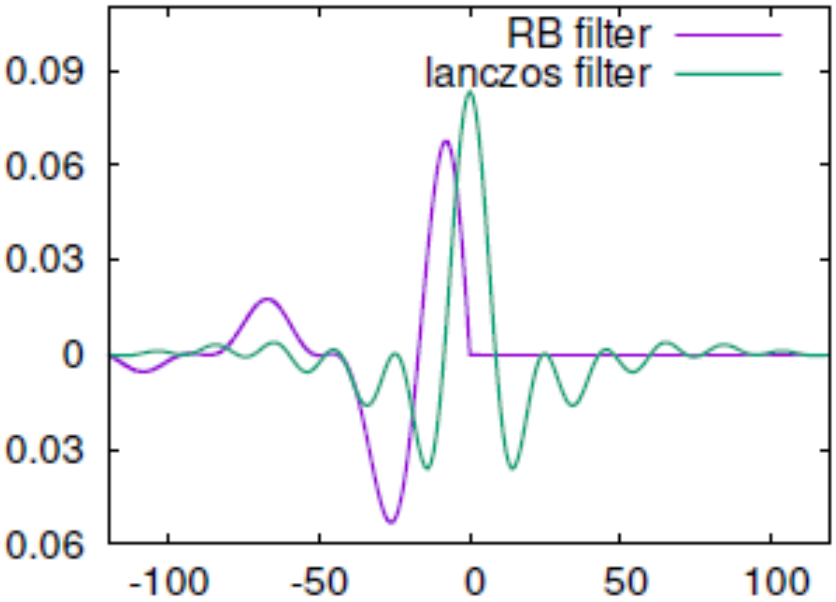


Figure: Lag (weighted center): -7.9

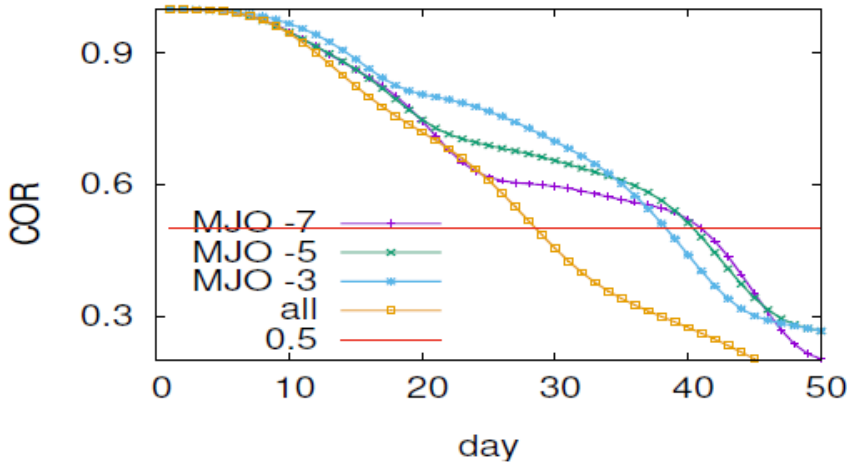
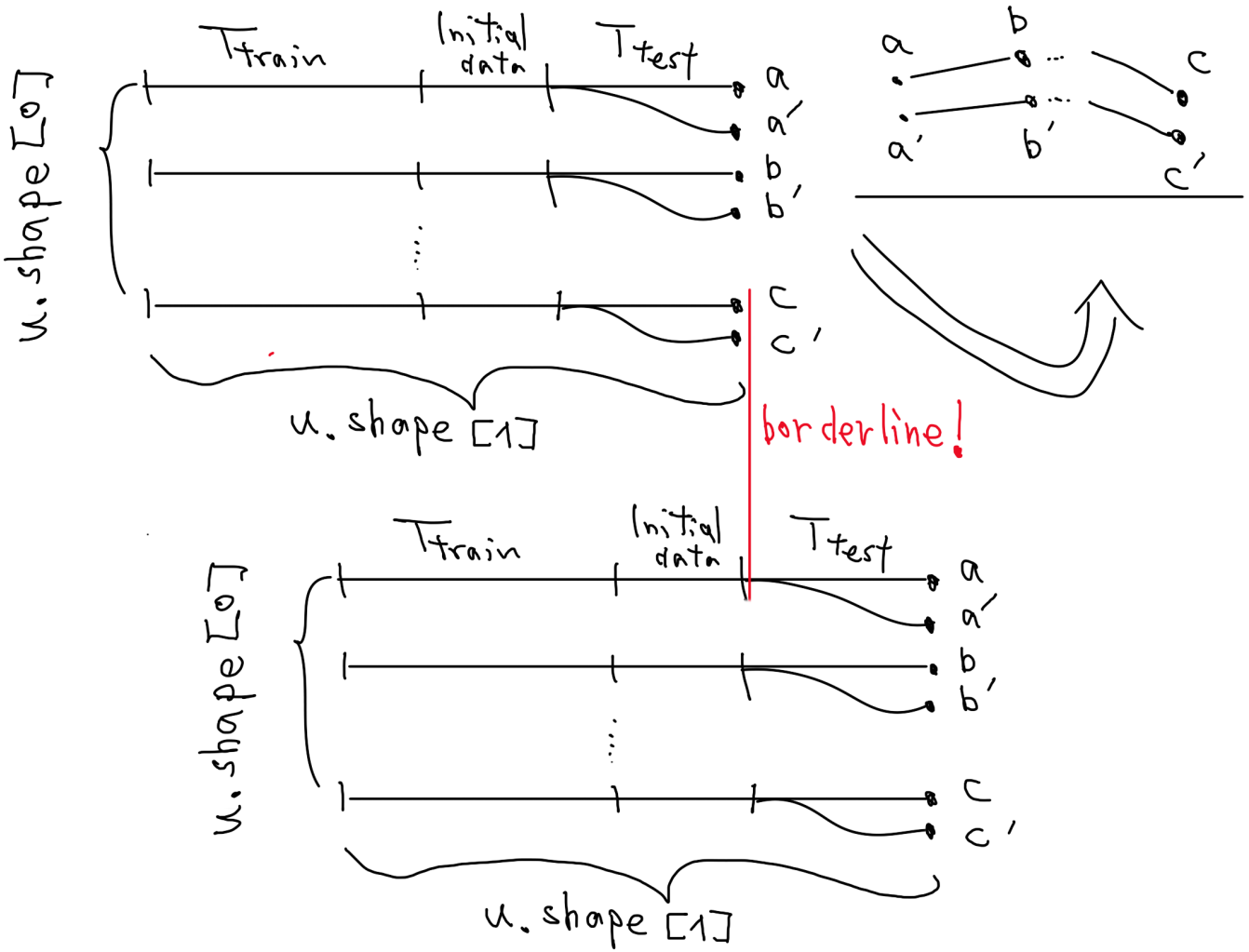


Figure: Pred with our RB filter (29 minus 8(lag)=21)

Even when using state-of-the-art physical models with high computational cost, the prediction accuracy is only on the order of about 20 days!

Learning framework: Tokyo WindSpeed and SP500

Mean absolute error (MAE) between real data and prediction

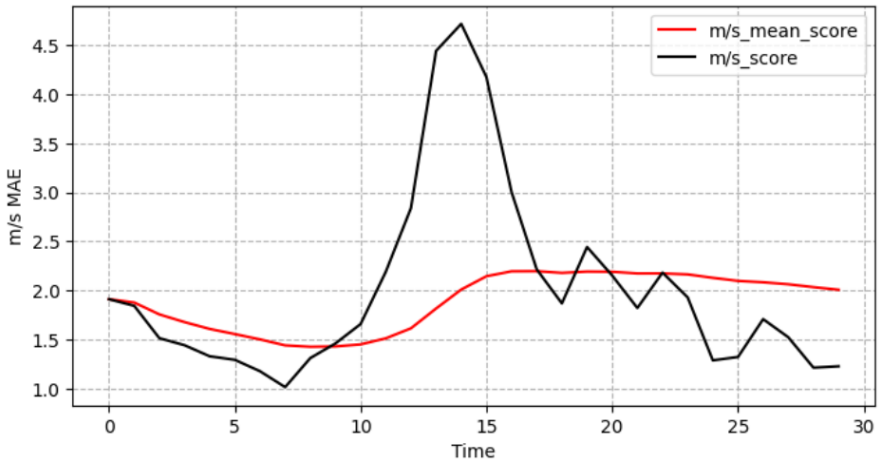


Generalization performance evaluation: choose one T_{test}

Result (windspeed in Tokyo region)

- predict 3 hours ahead ($T_{test} = 3$)
- most recent 1,000 hours of data are used to train($=: T_{train}$)
- Thirty independent trials were performed using a rolling-window validation scheme

Result MAE: 2.01 m/s (data-driven filter is applied: correlation 0.83, Jinno-Mitsui-Nakai-Saiki-Yoneda 2025)



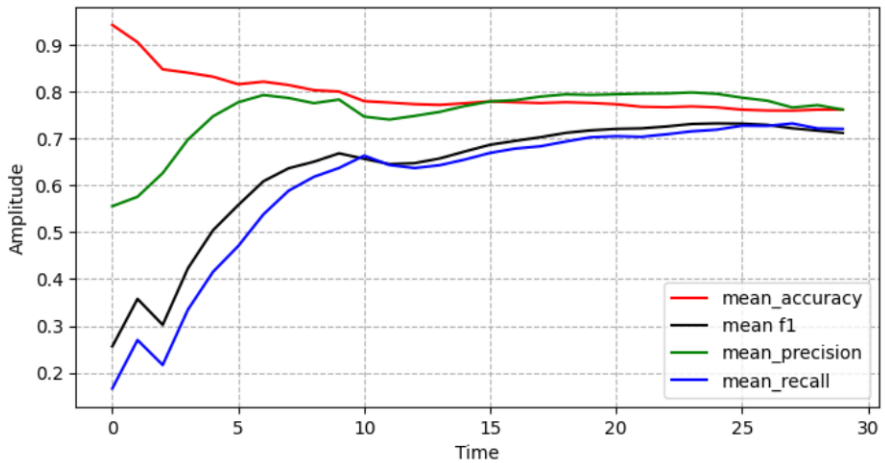
Cheng and Steenburgh (2005) CIRP, WRF (physical models)

MAE: 1.6 $\sim$ 1.9 m/s, but unclear whether a filter is applied...

Result (stock S&P500)

- predict 1 day ahead ($T_{test} = 1$)
- most recent 60 days of data are used to train($=: T_{train}$)
- learnable params $\approx$ 1,000
- Thirty independent trials were performed using a rolling-window validation scheme

Result Accuracy 0.762, F1 0.712, Recall 0.721 (data-driven filter is applied: correlation 0.985, Jinno-Mitsui-Nakai-Saiki-Yoneda 2025)

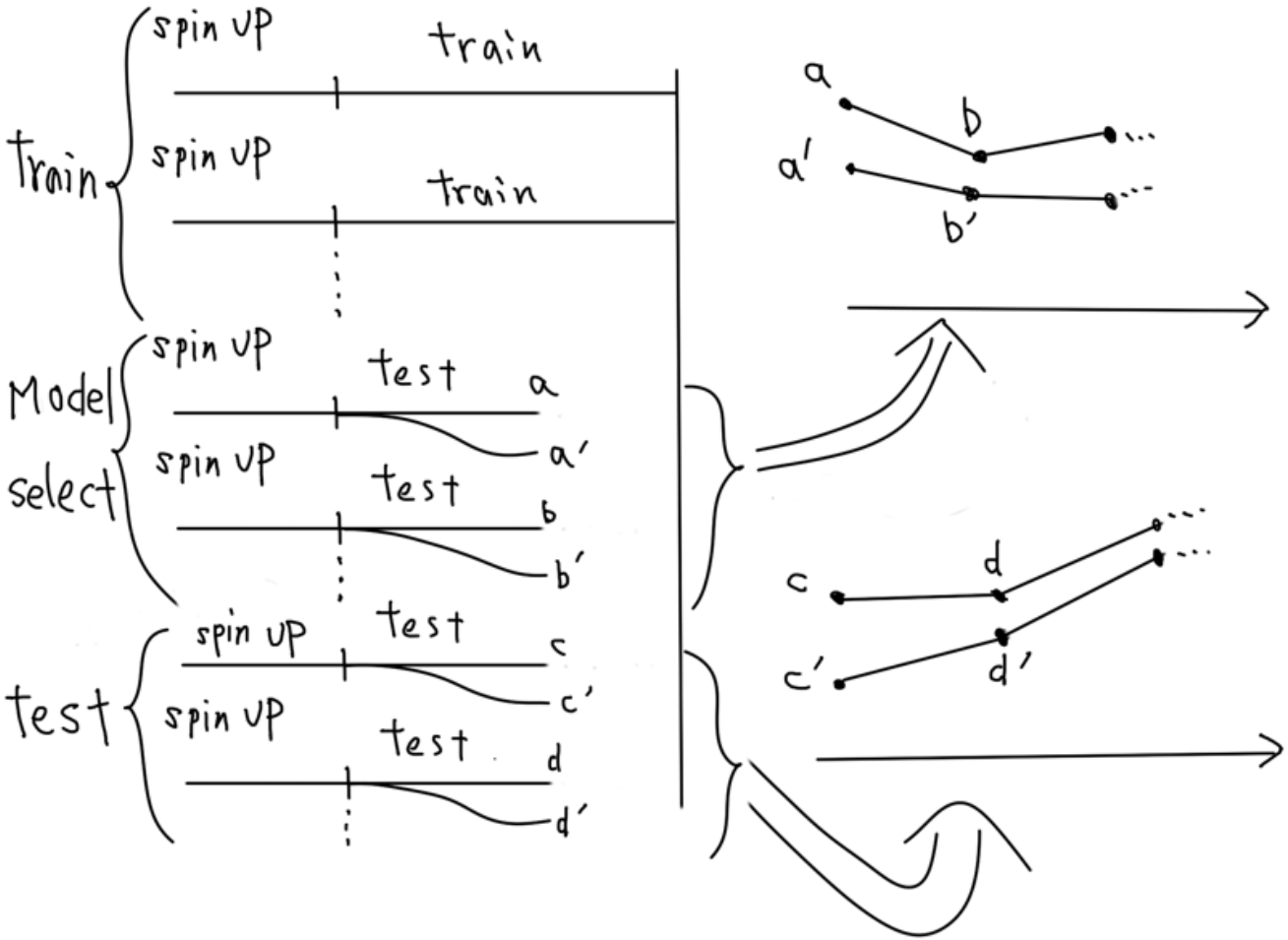


Gil, Duhamel-Sebline, McCarren (2024) xLSTM-TS

Accuracy 0.709, F1 0.730, Recall 0.768, wavelet filter is applied.
learnable params = 125,389, very huge!!

Learning framework: Lorenz 63

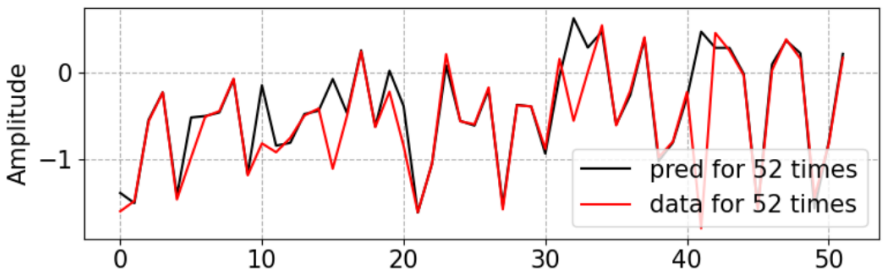
Mean absolute error (MAE) between real data and prediction



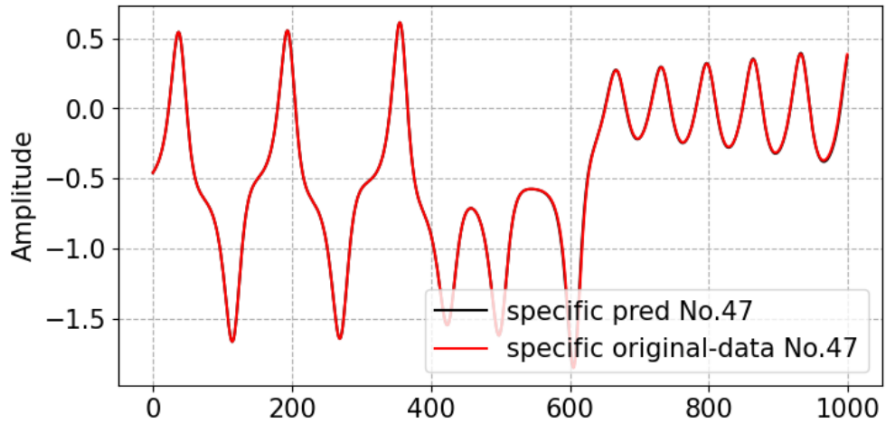
Generalization performance evaluation: take average between 1 and T_{test}

Result (Lorenz system)

- training dataset$\approx 600,000 \times 3$
- predict 1000-steps ahead ($T_{test} = 1000$), nondimensional time of 10
- standardized dataset
- MAE as the mean error over the entire prediction horizon



1000-step-ahead pred vs. ground truth



ParallelReservoir: mean-MAE 0.014,
Prediction: 1000 steps ahead

Feng et al. (2025) Physics-Guided Learning (PGL) method

A standardized mean-MAE of approximately $0.10 \sim 0.12$.

- training dataset$\approx 6,000 \times 3$
- incorporating physical laws as gentle hints, as a softer way of integrating physics compared to PINNs.

Thank you!