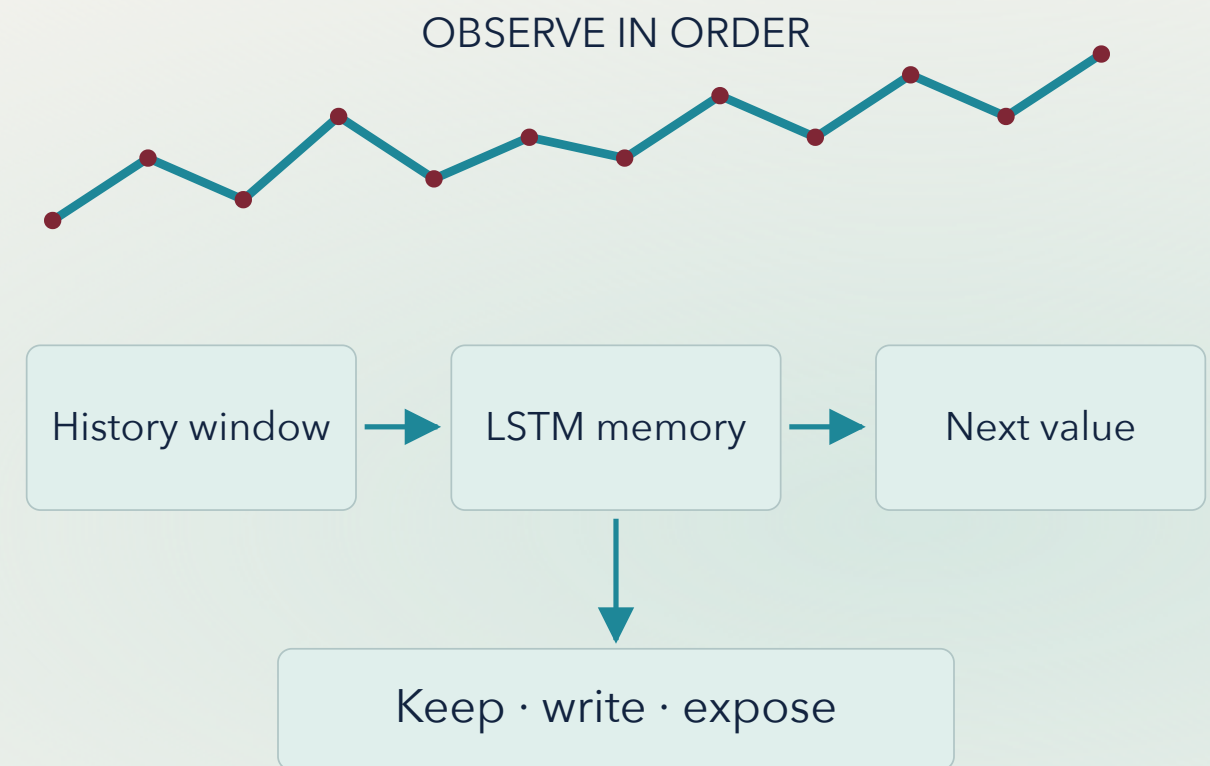


W07 · H1 · 14:00–15:00 · 19 October 2026

Sequential Data and LSTM

How do models turn ordered observations into useful memory?

Machine Learning in Engineering Science
Fall 2026 · Week 07 · Hour 1
Chi-Hua Jonny Yu · NCKU / LAiMM



A model of memory, not just a fitted curve

This hour: from sequences to memory

Define a valid sequence task, then explain and test how LSTM stores information.

Goal: trace one cell update and judge a forecast.

00–18 min

Sequence task + honest data



18–38 min

RNN limits + LSTM gates



38–56 min

Notebook + baseline evidence



56–60 min

Limits + summary

Mechanism diagram / illustrative teaching values

The same values can tell different stories

A sequence contains both values and an ordering rule.

$[1, 2, 4, 8] \neq [8, 1, 4, 2]$

What remains identical after shuffling?

A

$1 \rightarrow 2 \rightarrow 4 \rightarrow 8$

B

$8 \rightarrow 1 \rightarrow 4 \rightarrow 2$

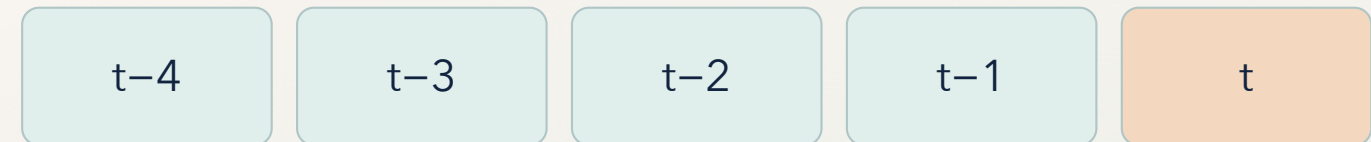
Mechanism diagram / illustrative teaching values

Define the forecast origin

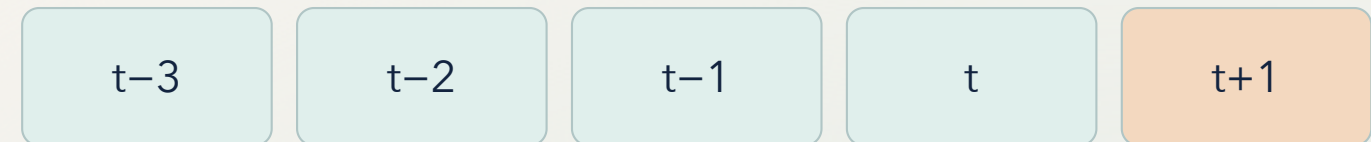
The prediction must use only information available at its origin.

history: $x[t-L:t]$ target: $x[t]$

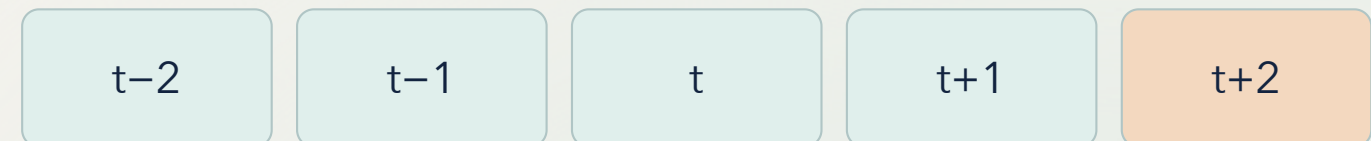
Window 1



Window 2



Window 3



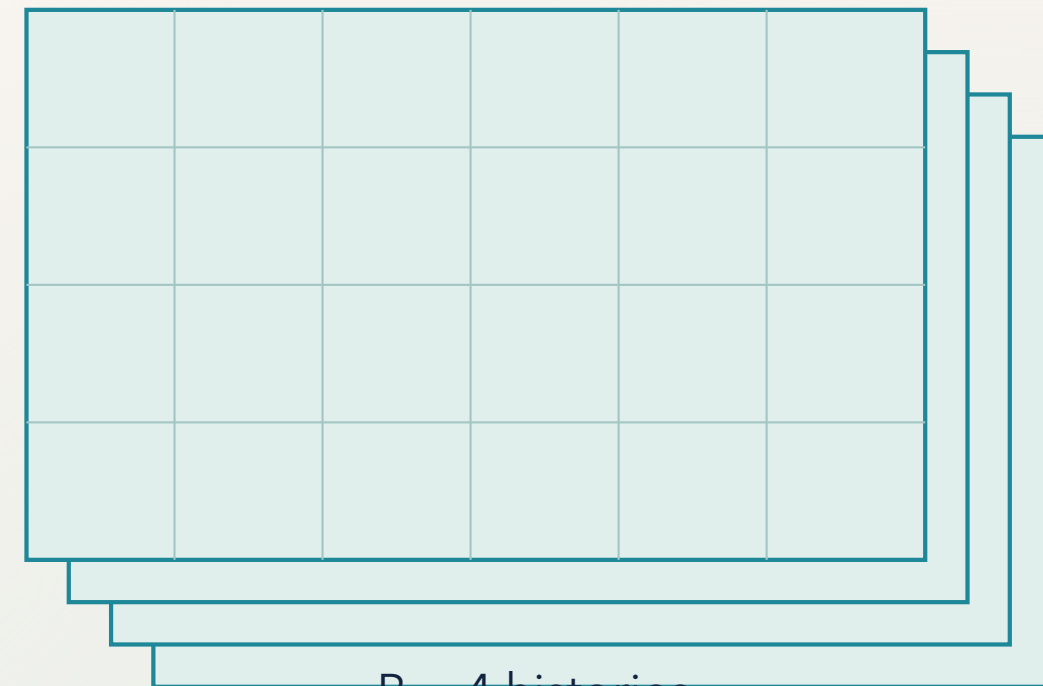
History shifts forward; target stays excluded

Mechanism diagram / illustrative teaching values

One sample is an entire window

Sequence models receive a batch of histories, with features at every time step.

$$X \in \mathbb{R}^{B \times T \times f}$$



B = 4 histories

T = 24 samples

F = 1 sensor

Mechanism diagram / illustrative teaching values

Targets determine the output structure

Many-to-one predicts one value; many-to-many predicts a value at multiple positions.

A

24 sensor samples → one forecast

B

token sequence → shifted token sequence

Mechanism diagram / illustrative teaching values

Split time before selecting a model

Train on earlier targets, select on later validation, then evaluate on test.

`train < validation < test`

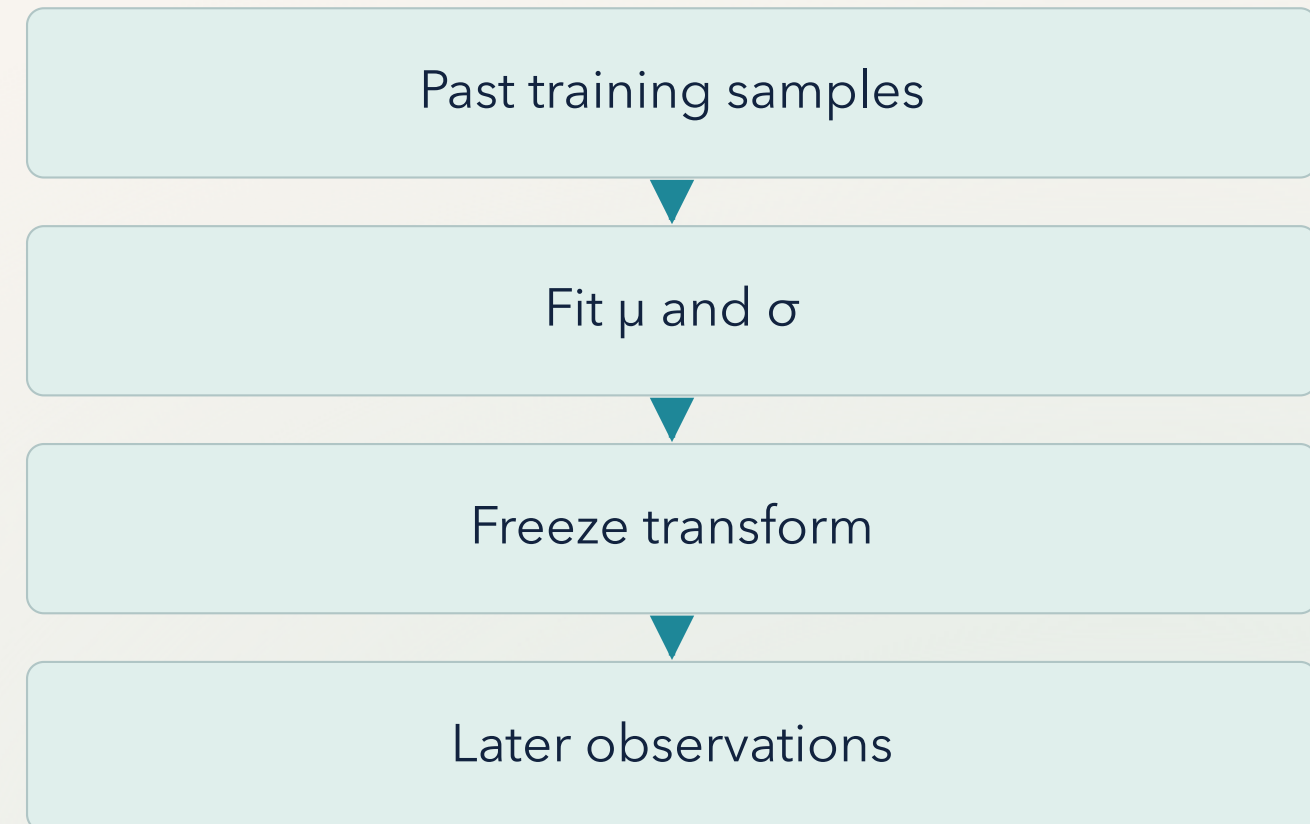


Mechanism diagram / illustrative teaching values

Fit normalization on the training past

Global means and variances can reveal future distribution information.

$$z = (x - \mu_{\text{train}}) / \sigma_{\text{train}}$$

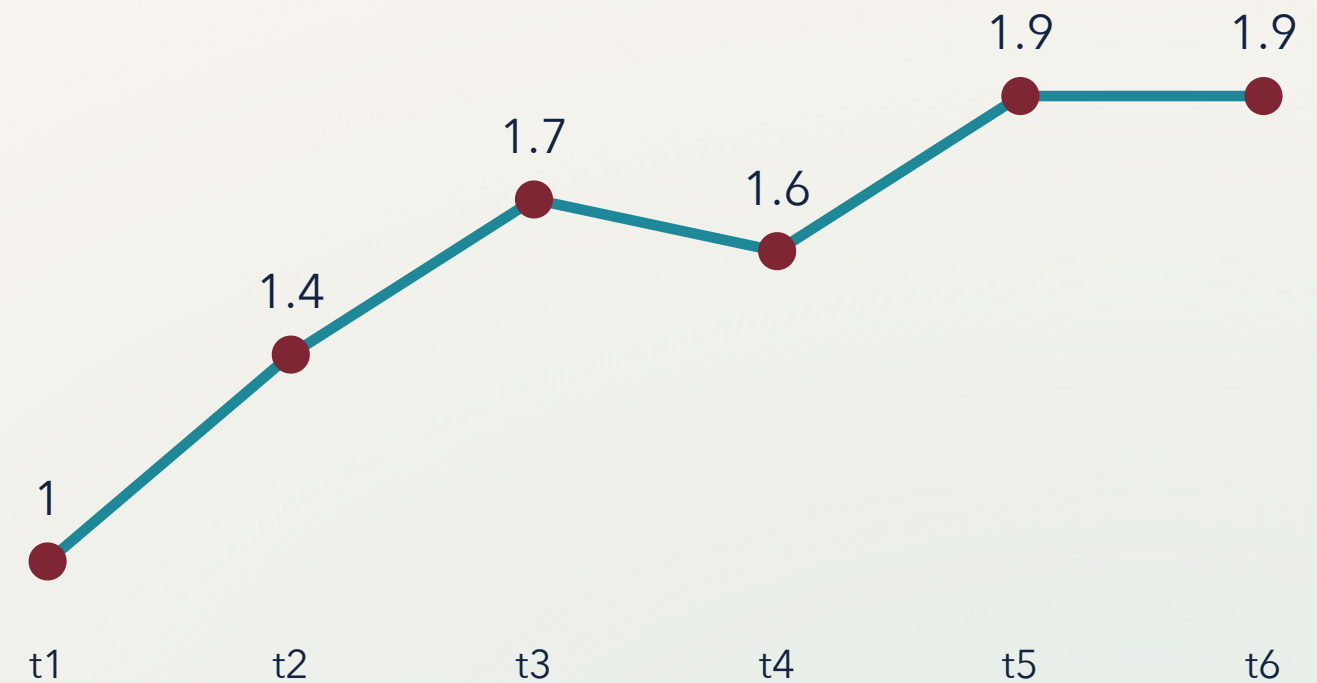


Mechanism diagram / illustrative teaching values

Persistence is a serious baseline

Predict the next observation using the last known observation.

$$\hat{y}[t] = x[t-1]$$



Order is part of the input

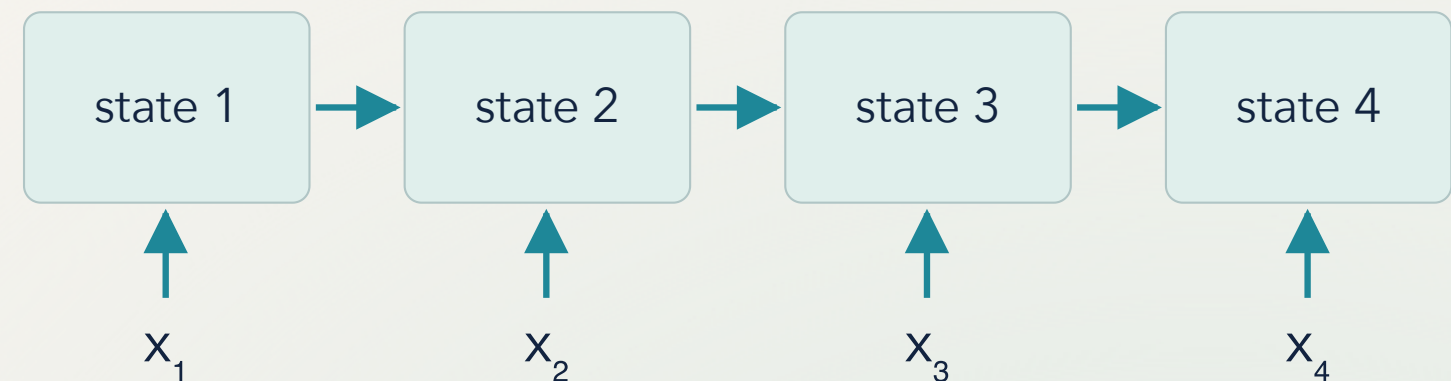
Mechanism diagram / illustrative teaching values

A recurrent state summarizes the past

An RNN applies the same state-update parameters at every time step.

$$h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$$

Shared update parameters across time



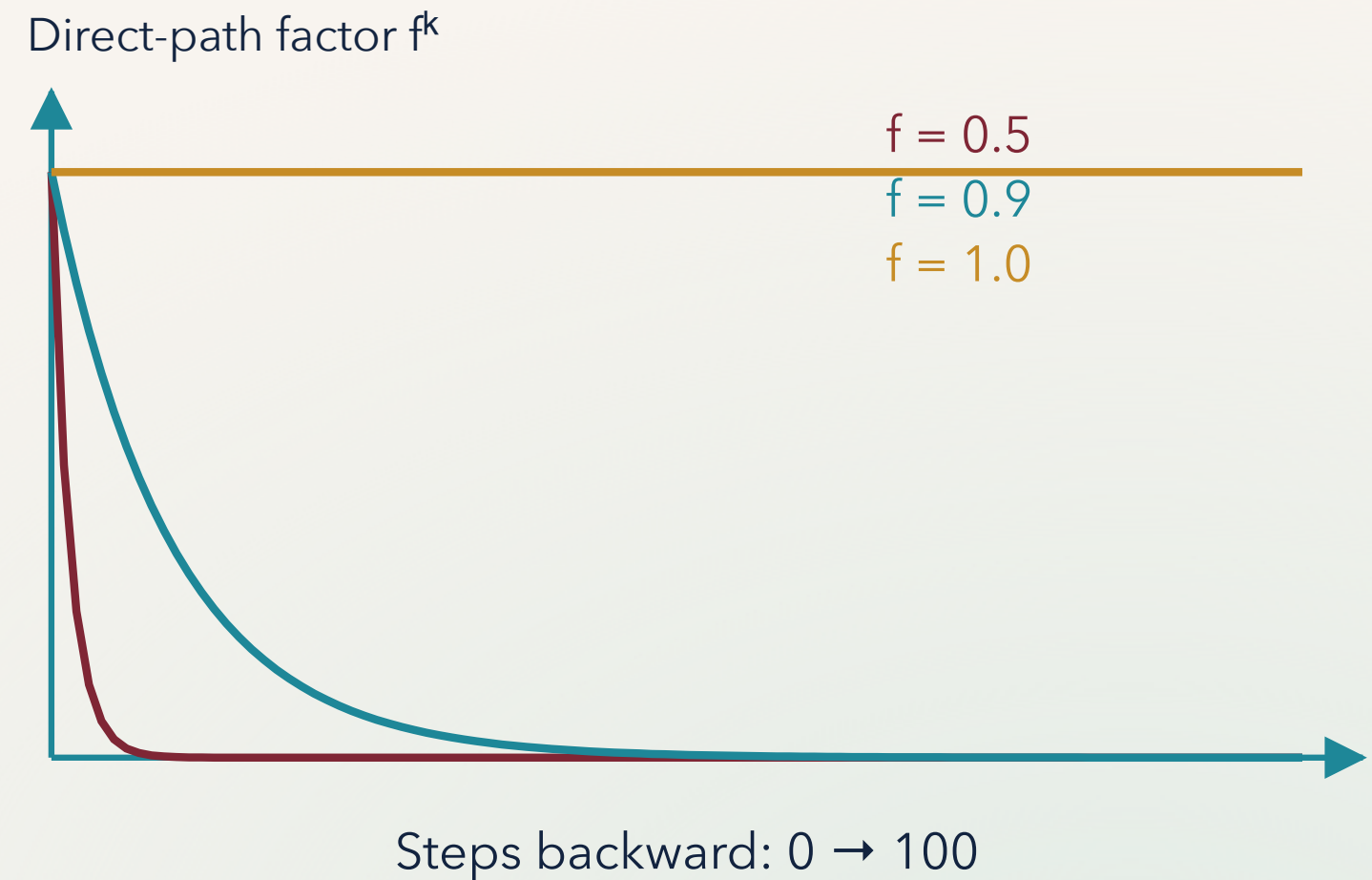
Past → running state → later state

Mechanism diagram / illustrative teaching values

Learning must cross the time chain

Backpropagation through time multiplies Jacobians along earlier states.

$$\partial h_t / \partial h_{t-k} = J_t J_{t-1} \dots J_{t-k+1}$$



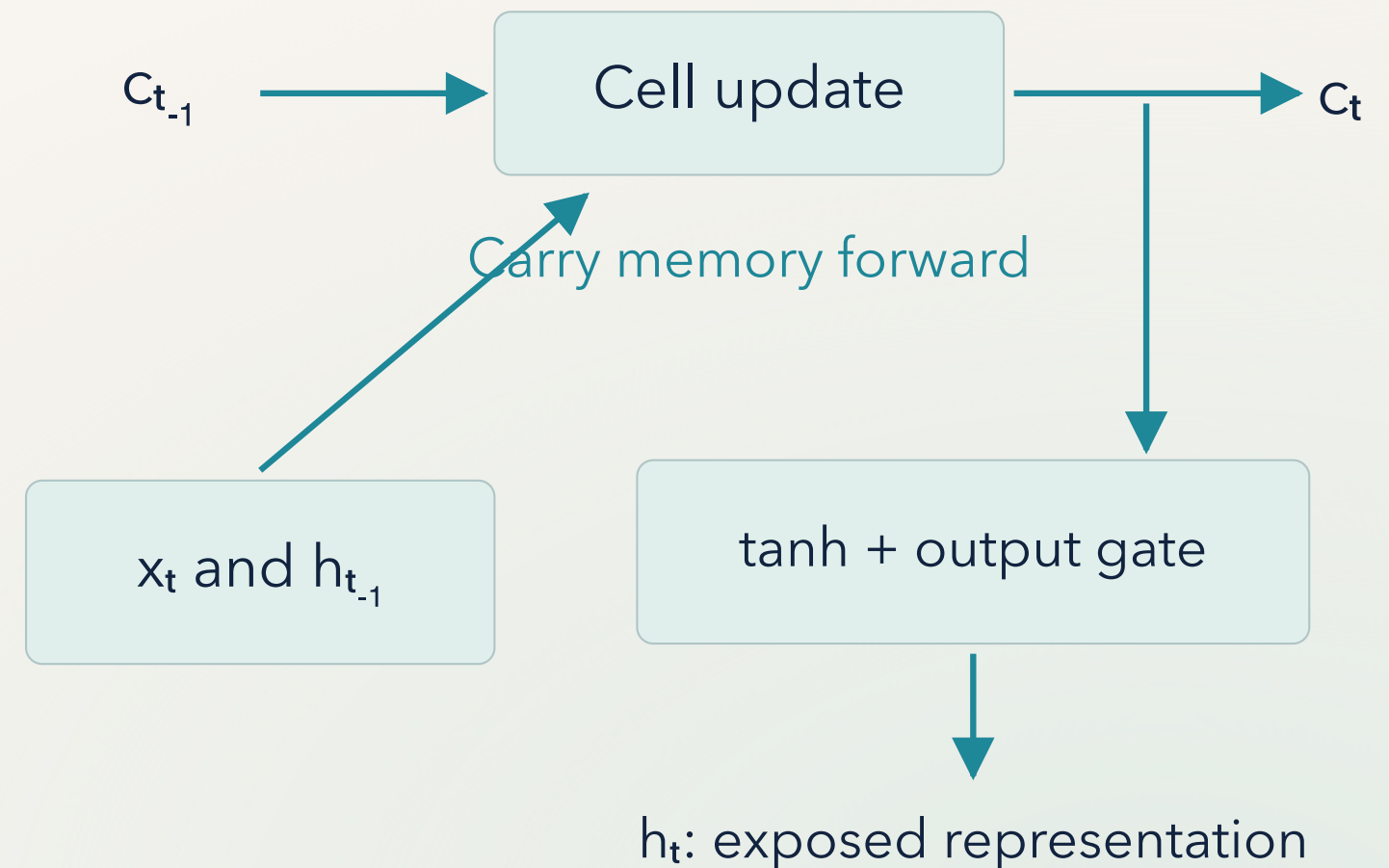
Mechanism diagram / illustrative teaching values

LSTM adds a separate memory path

The cell state carries memory; the hidden state exposes a filtered view.

cell: c_t hidden output: h_t

TWO STATES, TWO DIFFERENT JOBS

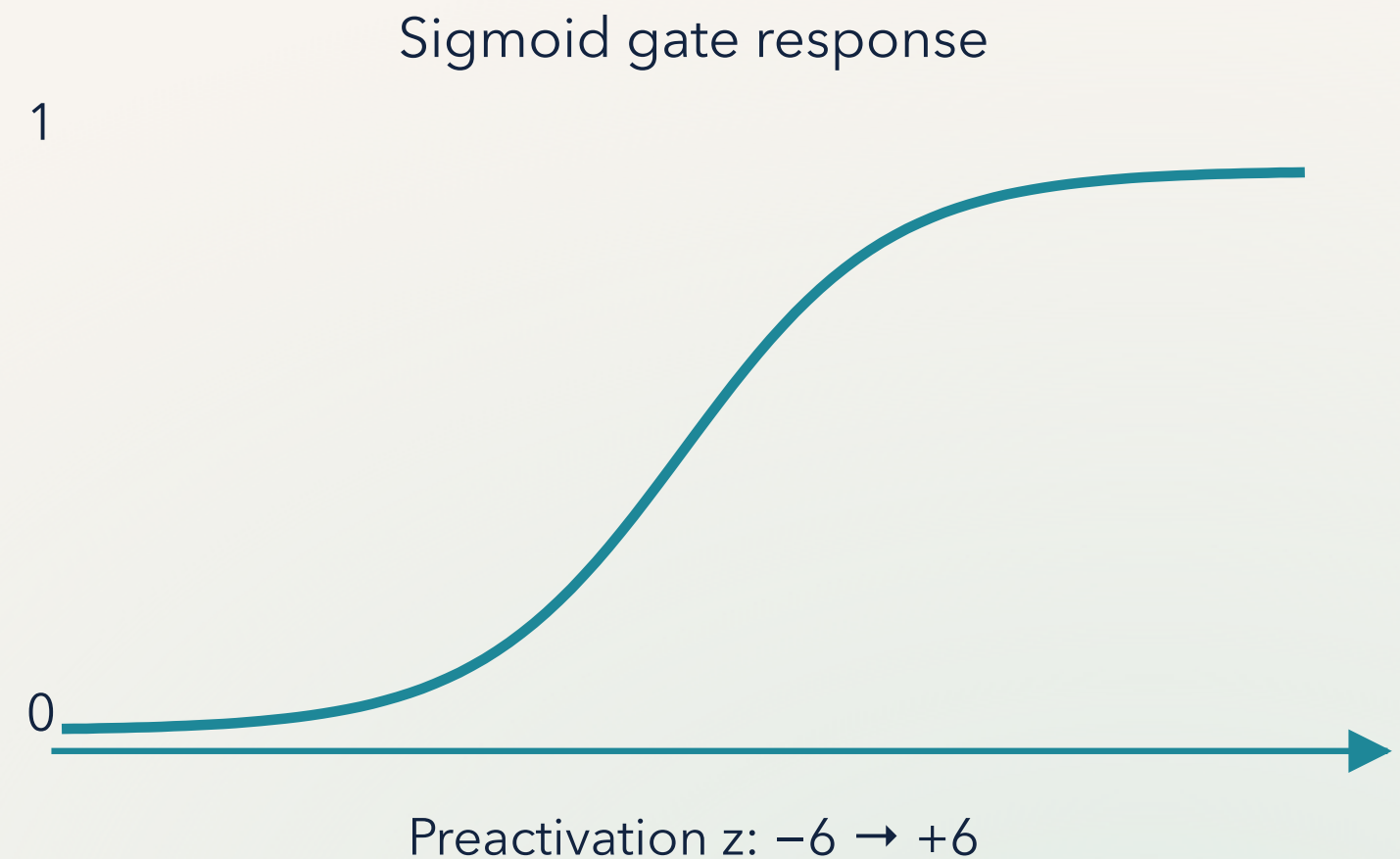


Mechanism diagram / illustrative teaching values

A gate is a learned vector of coefficients

Sigmoid outputs values between zero and one for each hidden dimension.

$$\sigma(z) = 1 / (1 + \exp(-z))$$



Mechanism diagram / illustrative teaching values

The forget gate controls retained memory

Multiply the previous cell state by a learned forget coefficient.

$$f_t = \sigma(W_f [h_{t-1}; x_t] + b_f)$$

KEEP A FRACTION OF OLD MEMORY

old c = 0.80

× 0



retained = 0.00

old c = 0.80

× 0.5



retained = 0.40

old c = 0.80

× 0.9



retained = 0.72

f is learned from $[h_{t-1}; x_t]$, dimension by dimension

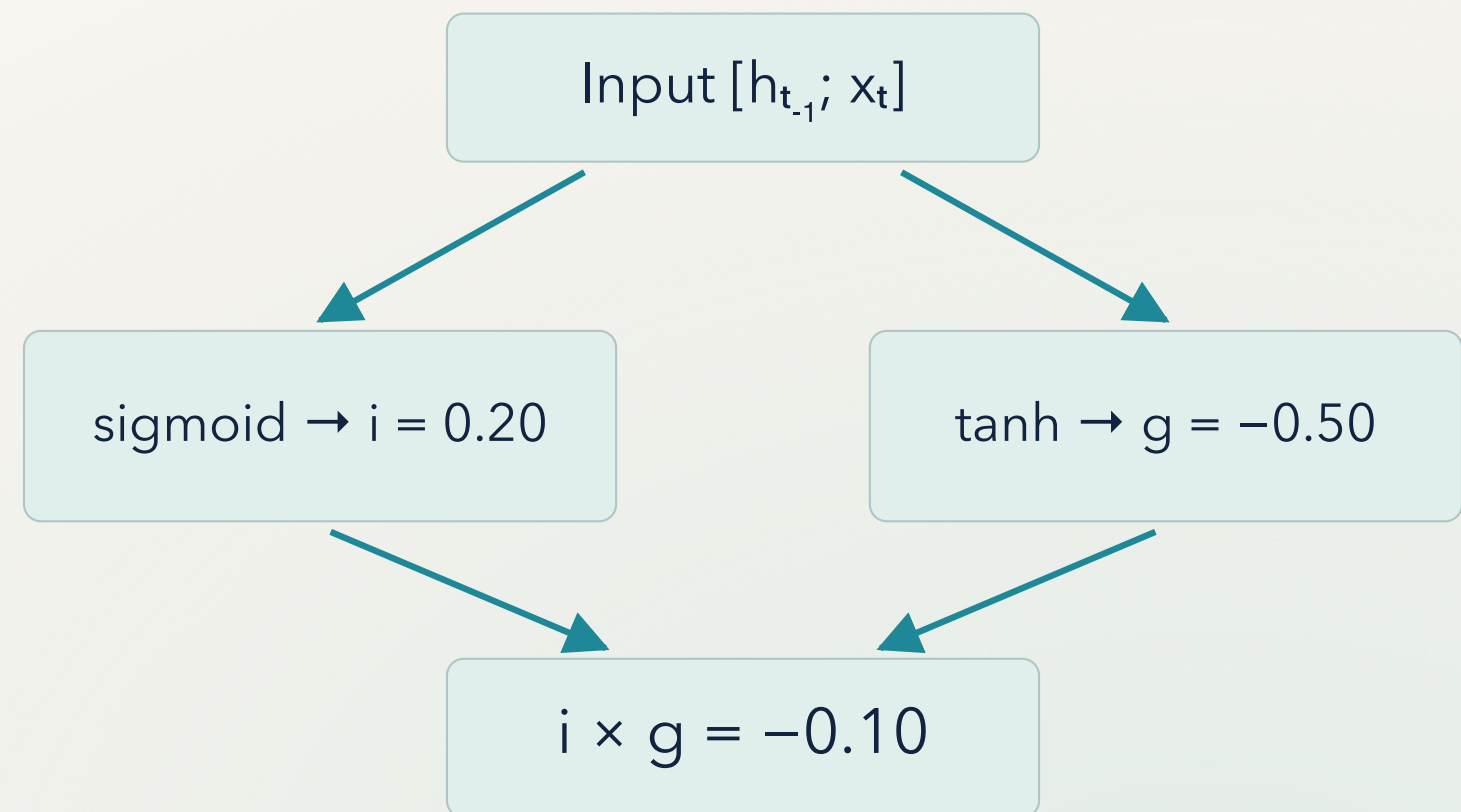
Mechanism diagram / illustrative teaching values

The input gate controls the write

A candidate proposes content; the input gate scales how much is written.

$$i_t = \sigma(W_i [h_{t-1}; x_t] + b_i)$$
$$g_t = \tanh(W_g [h_{t-1}; x_t] + b_g)$$

PROPOSE CONTENT, THEN SCALE THE WRITE



A nonnegative gate can write negative candidate content

Mechanism diagram / illustrative teaching values

The cell update adds two paths

Retain selected old memory and add selected new content.

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

MEMORY IS AN ADDITIVE UPDATE

Retain: 0.90×0.80

Write: $0.20 \times (-0.50)$

+0.72

-0.10

New $c = 0.62$

$\odot$ means multiply corresponding vector coordinates

Mechanism diagram / illustrative teaching values

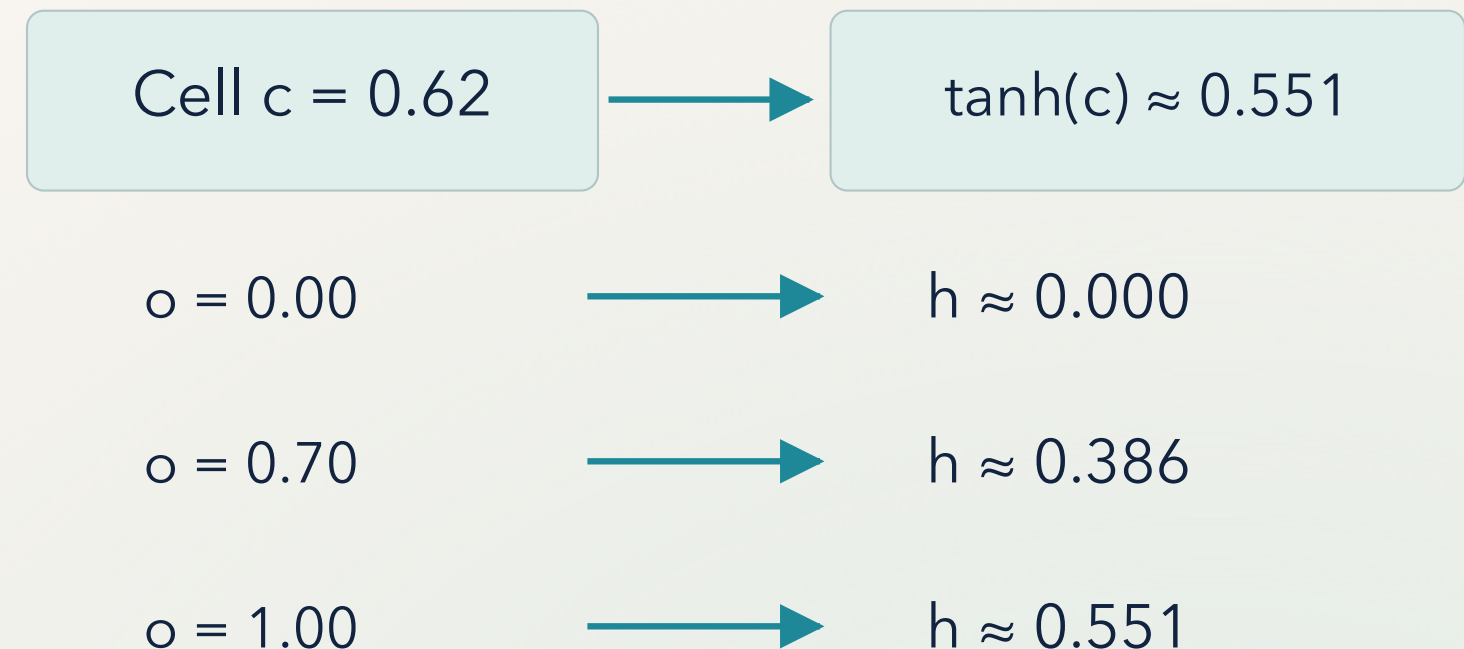
The output gate exposes memory

The hidden state is a bounded, gated view of the updated cell.

$$o_t = \sigma(W_o [h_{t-1}; x_t] + b_o)$$

$$h_t = o_t \odot \tanh(c_t)$$

EXPOSURE CAN CHANGE WITHOUT REWRITING c



Same cell content; different visible hidden output

Mechanism diagram / illustrative teaching values

Trace one numerical cell update

Chosen gates make every intermediate step visible.

$$0.9 \times 0.8 + 0.2 \times (-0.5) = 0.62$$

What is the hidden output?

Retain

$$0.90 \times 0.80 = 0.72$$



Write

$$0.20 \times (-0.50) = -0.10$$



Update

$$0.72 - 0.10 = 0.62$$



Expose

$$0.70 \times \tanh(0.62) \approx 0.386$$

Mechanism diagram / illustrative teaching values

Predict before moving a gate

A high forget coefficient does not prevent new content from changing memory.

$$c_{\text{new}} = f \times c_{\text{old}} + i \times g$$

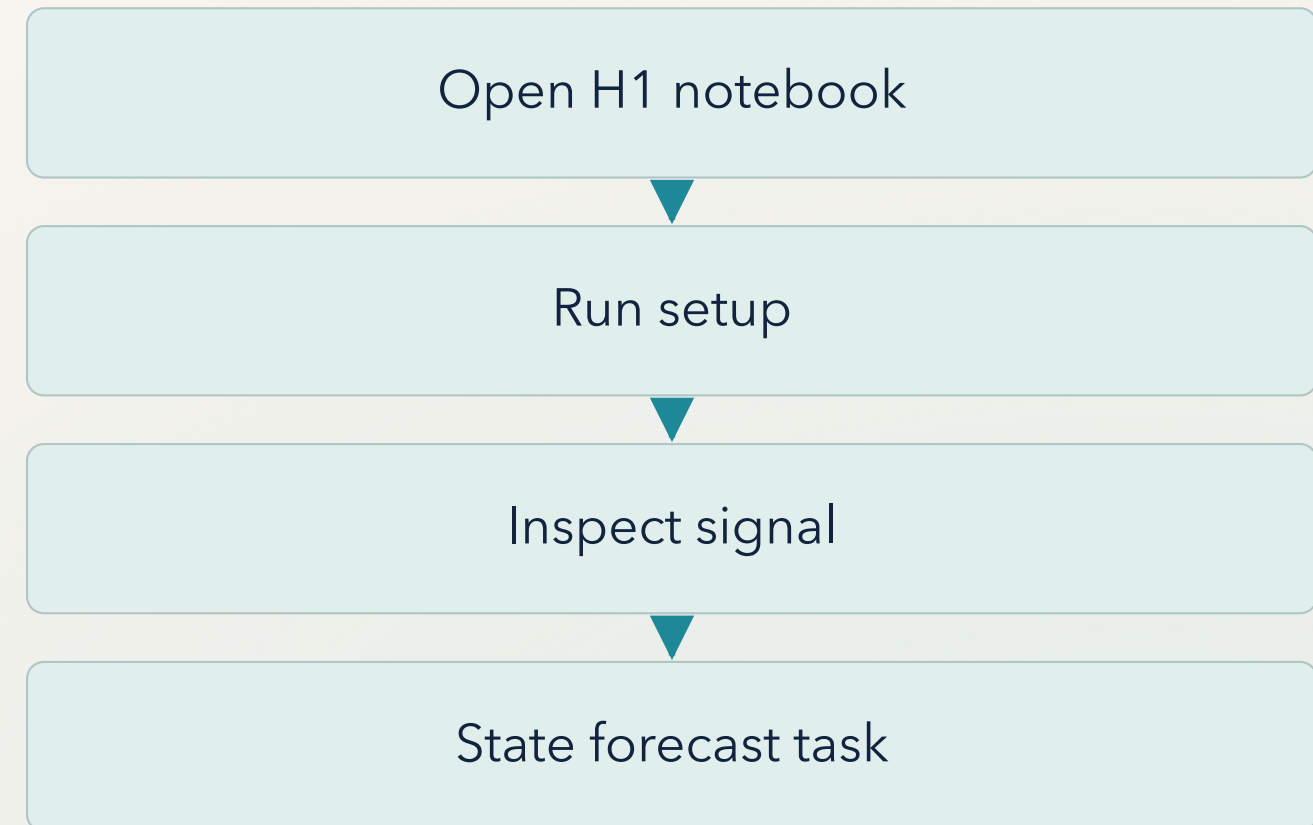


cell = 0.620; hidden = 0.386

Interactive chosen values – change one variable

Open the sequence notebook

Run setup and inspect the synthetic sensor before fitting anything.



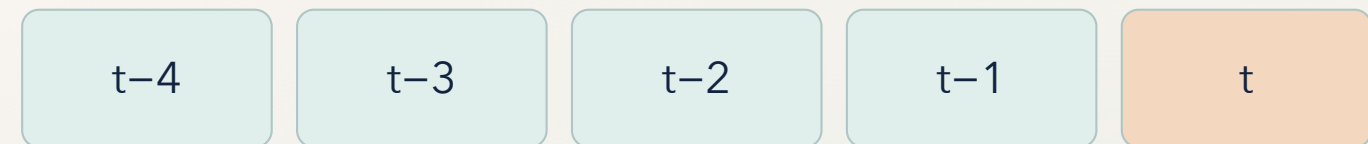
Mechanism diagram / illustrative teaching values

Check the window indices

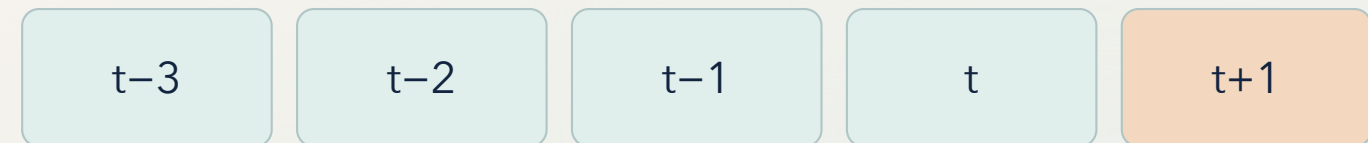
A correct shape is insufficient if the window contains its target.

```
X[i] = signal[t-24:t] y[i] =  
signal[t]
```

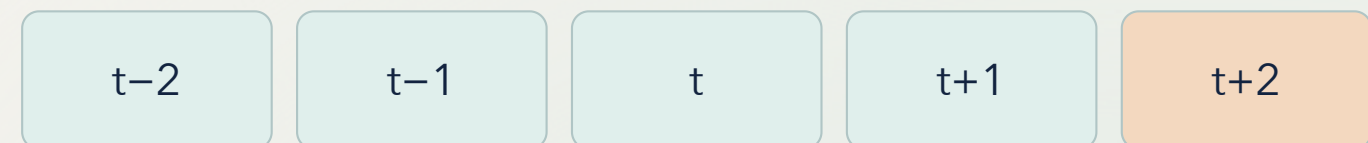
Window 1



Window 2



Window 3



History shifts forward; target stays excluded

Mechanism diagram / illustrative teaching values

Compare linear memory with learned state

Ridge uses fixed window positions;
LSTM updates a shared recurrent state.

A

Ridge: flattened 24-value window

B

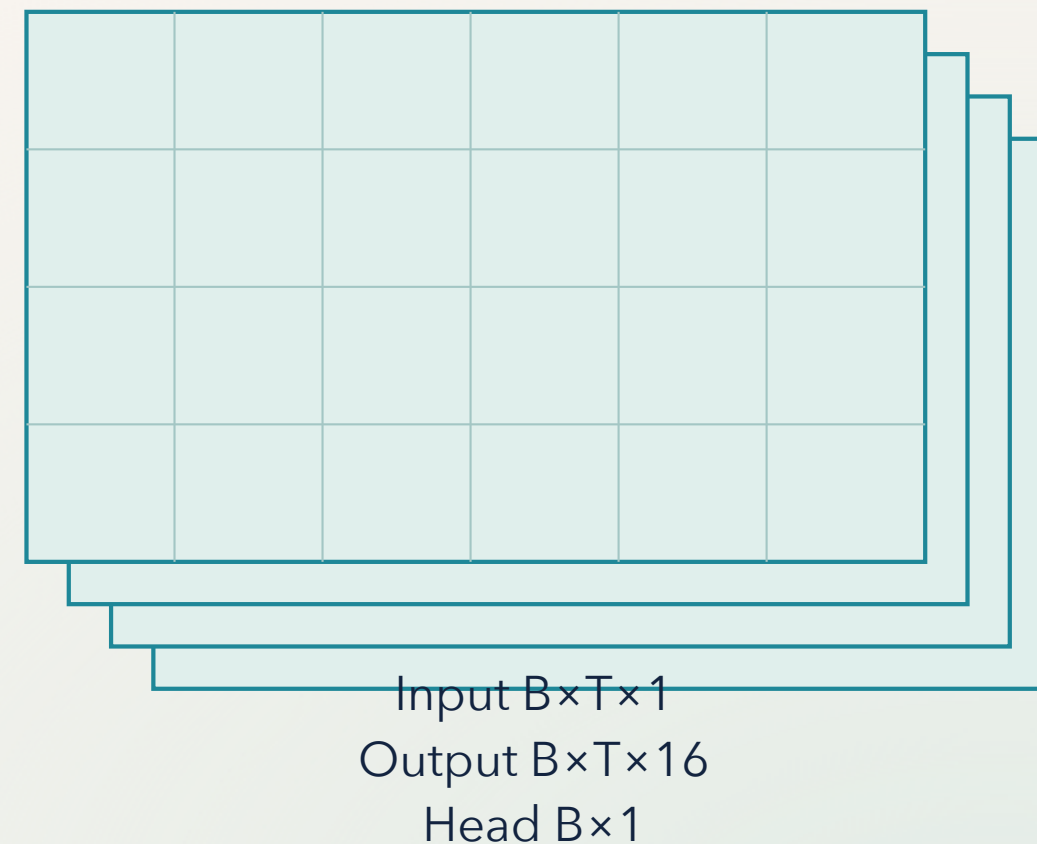
LSTM: 24 shared state updates

Mechanism diagram / illustrative teaching values

Map the implementation to tensor axes

batch_first changes input/output layout,
not hidden-state layout.

input: $(B, T, 1)$ states: (B, T, H) h_n :
 $(layers, B, H)$



Mechanism diagram / illustrative teaching values

Training and selection use different data

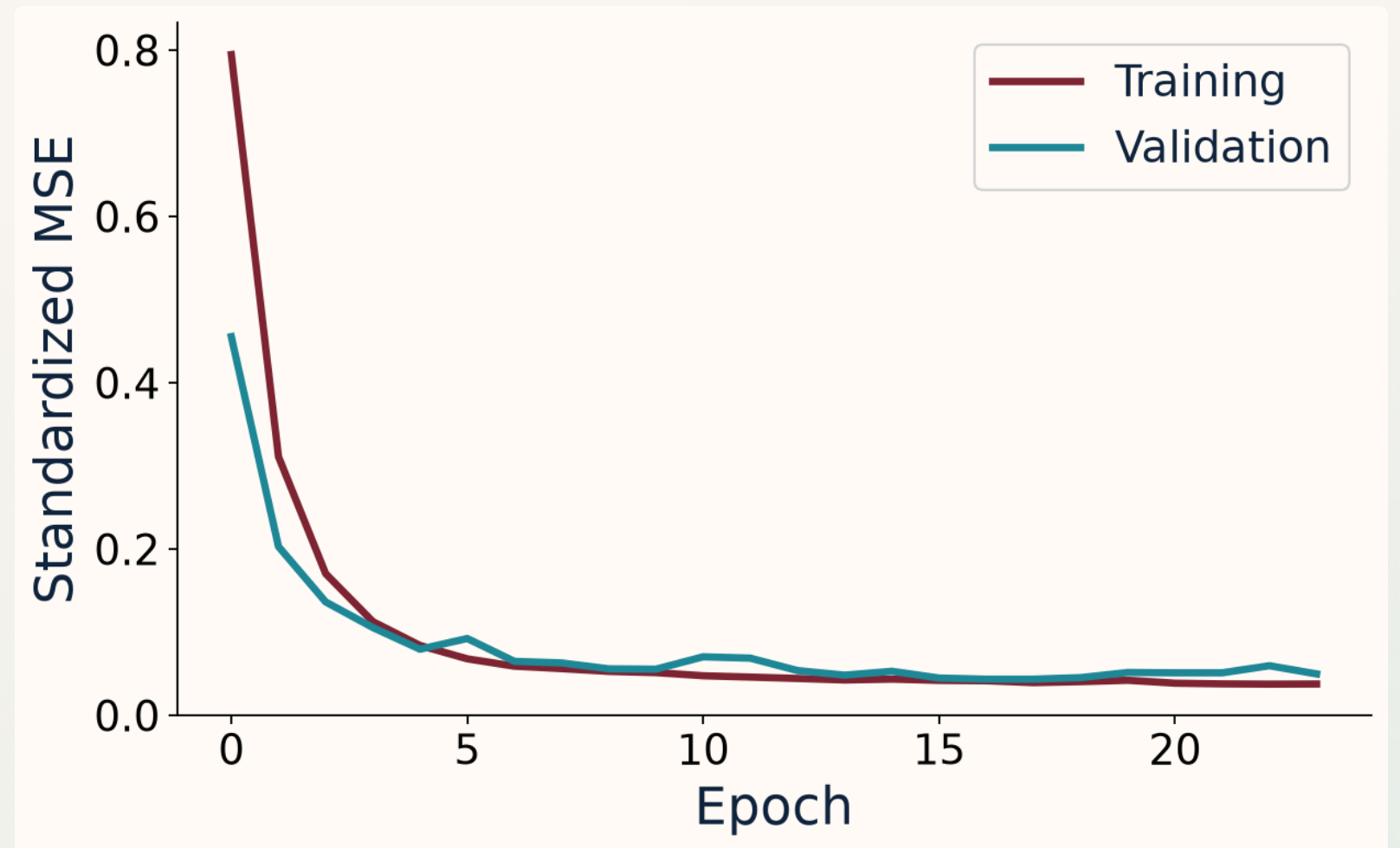
Gradient updates use training;
checkpoint selection uses validation.



Mechanism diagram / illustrative teaching values

A loss curve must come from the run

Inspect actual training and validation trajectories in your notebook.

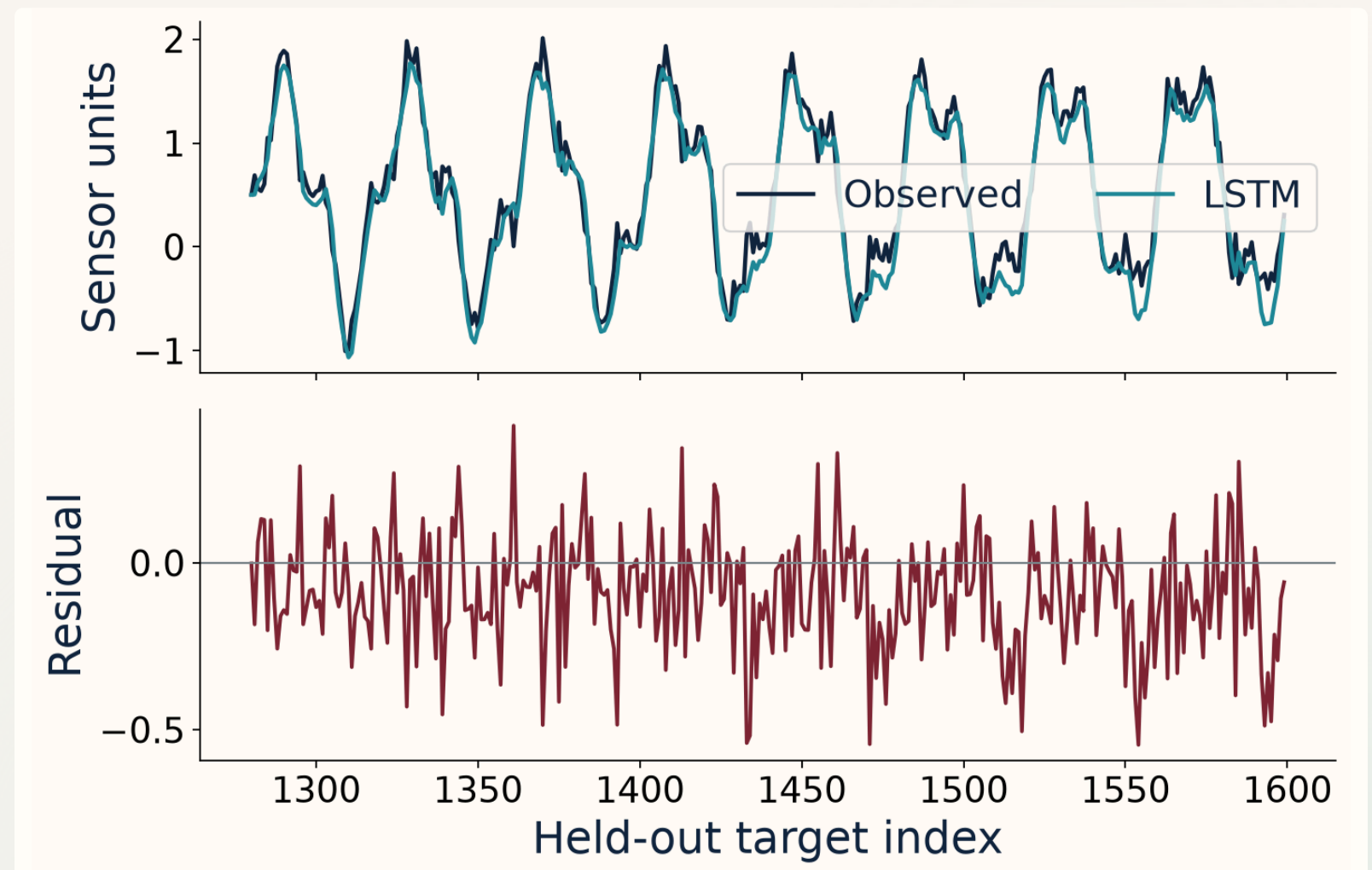


Actual CPU notebook run · synthetic teaching data

Judge errors in the original units

MAE averages error magnitude; RMSE emphasizes large residuals.

$$\text{MAE} = \text{mean } |\hat{y} - y| \quad \text{RMSE} = \sqrt{\text{mean}(\hat{y} - y)^2}$$

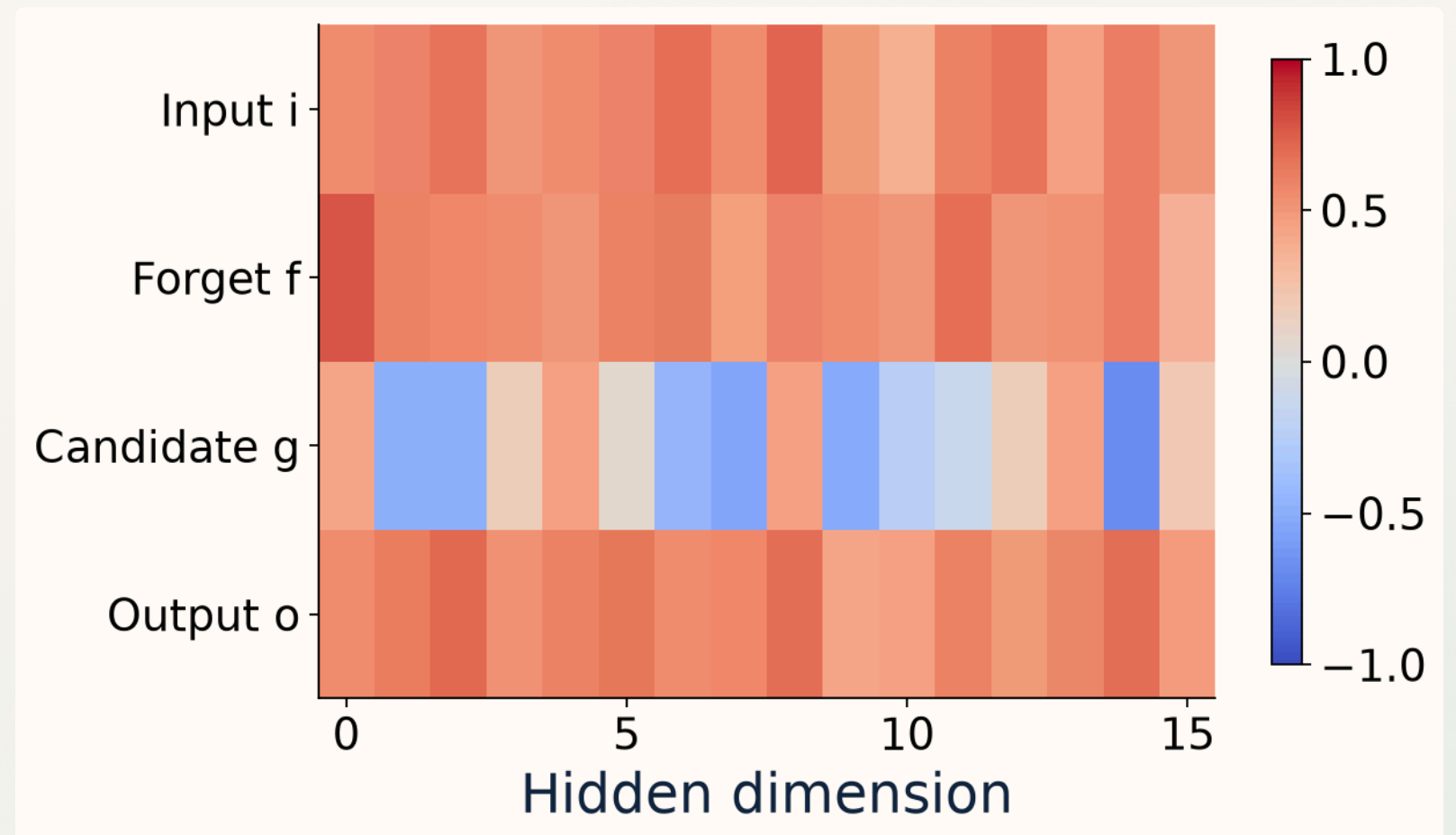


Actual CPU notebook run · synthetic teaching data

A trained gate can be reconstructed

PyTorch concatenates its learned gate parameters in i, f, g, o order.

linear projections → split →
activations → cell update



Actual CPU notebook run · synthetic teaching data

Reproduce one controlled change

Change window length while preserving the forecast question and split boundaries.

A

Reference: WINDOW = 24

B

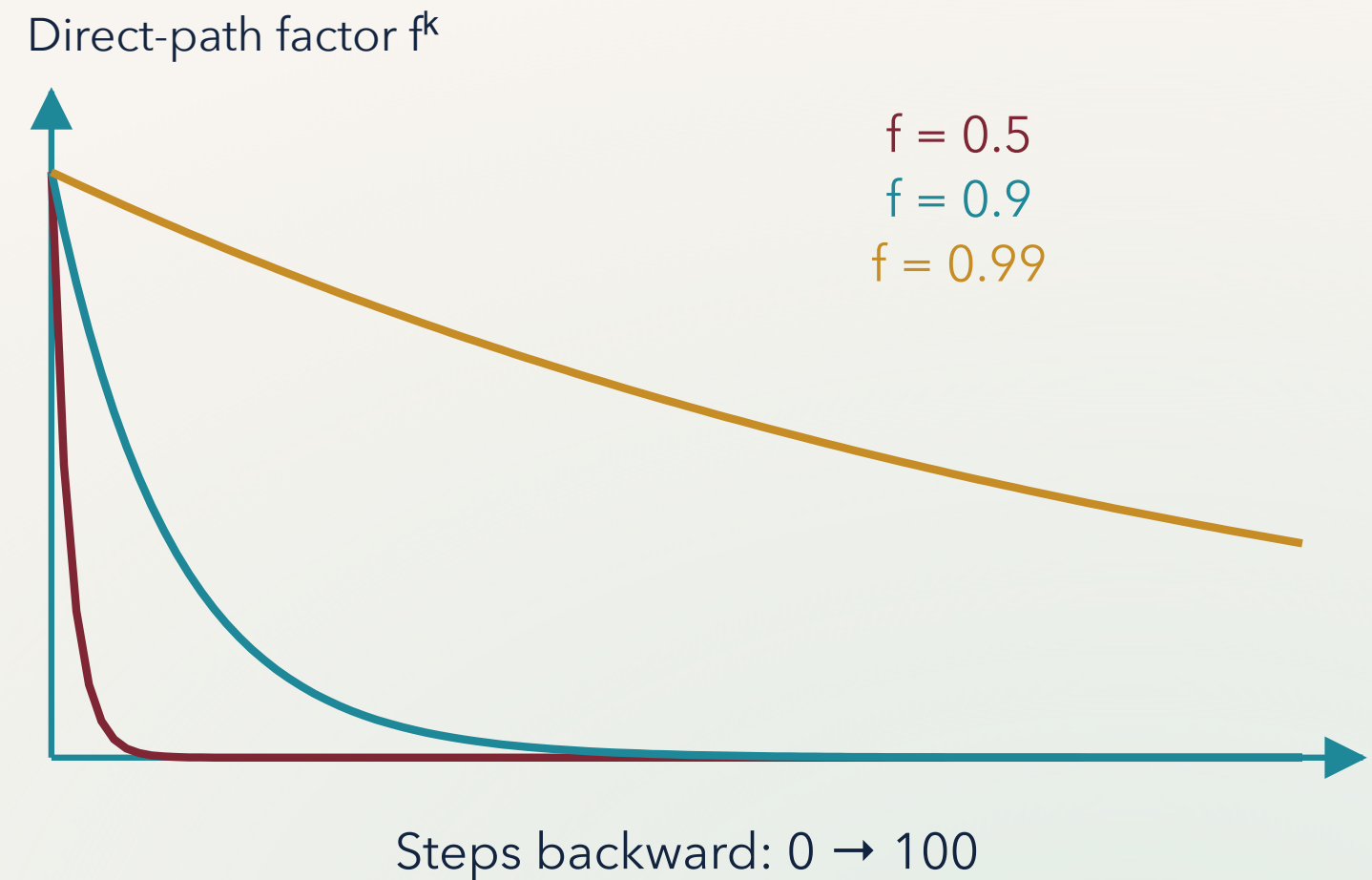
Intervention: WINDOW = 12 or 48

Mechanism diagram / illustrative teaching values

LSTM helps; it does not guarantee memory

Retention depends on learned gates,
sequence structure, and optimization.

direct memory factor $\approx \prod f_j$



Mechanism diagram / illustrative teaching values

Summary: order, memory, evidence

Order defines the task. LSTM gates control what memory is kept, written, and exposed.

$$\begin{aligned}c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\h_t &= o_t \odot \tanh(c_t)\end{aligned}$$

Ordered history

Use only the available past

Gated memory

Keep old state + write new content

Forecast evidence

Compare baselines; preserve the run

NEXT: attention retrieves relevant earlier states

Mechanism diagram / illustrative teaching values