Deep Learning for Time Series

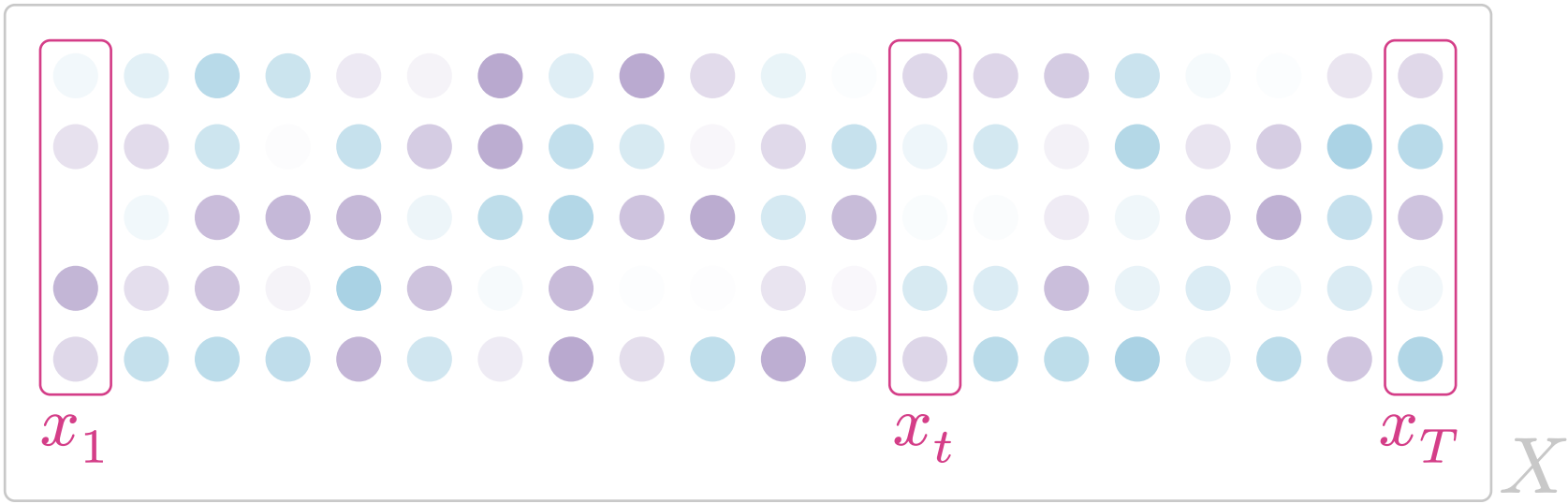
Session 1: Basics

Romain Tavenard

Definition

In this course, we will denote by *time series* any sequence of feature vectors:

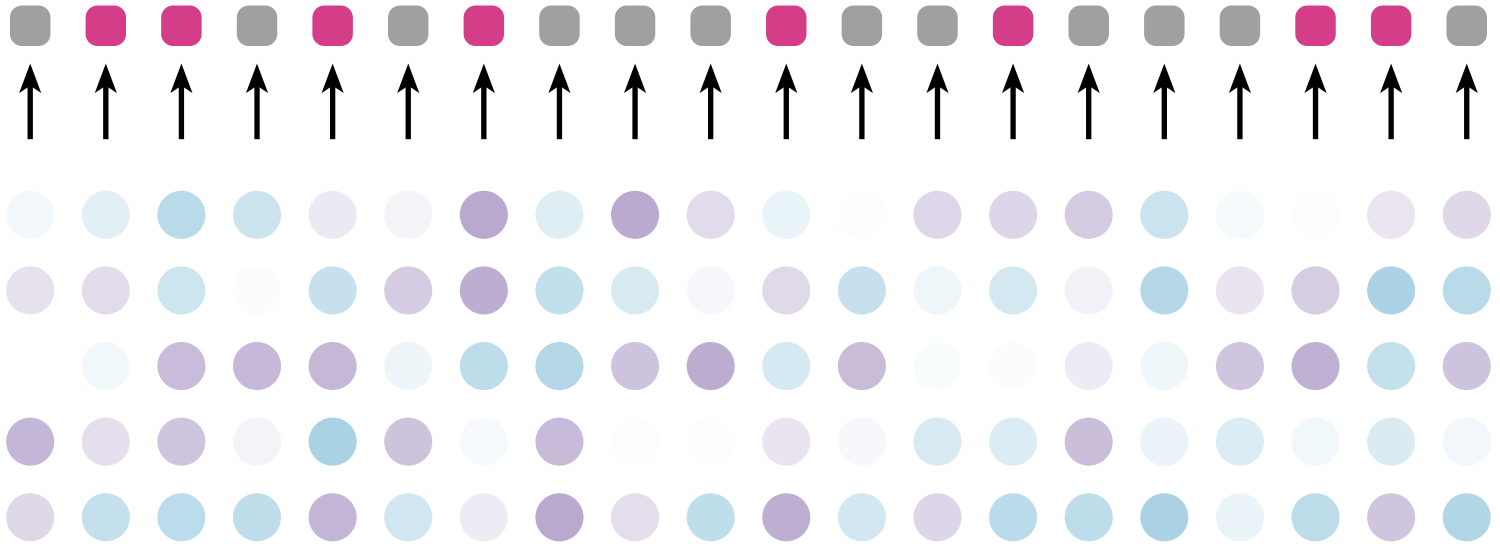
$$X = (x_t)_{t=1}^{T} \quad \text{such that} \quad x_t \in \mathbb{R}^d$$



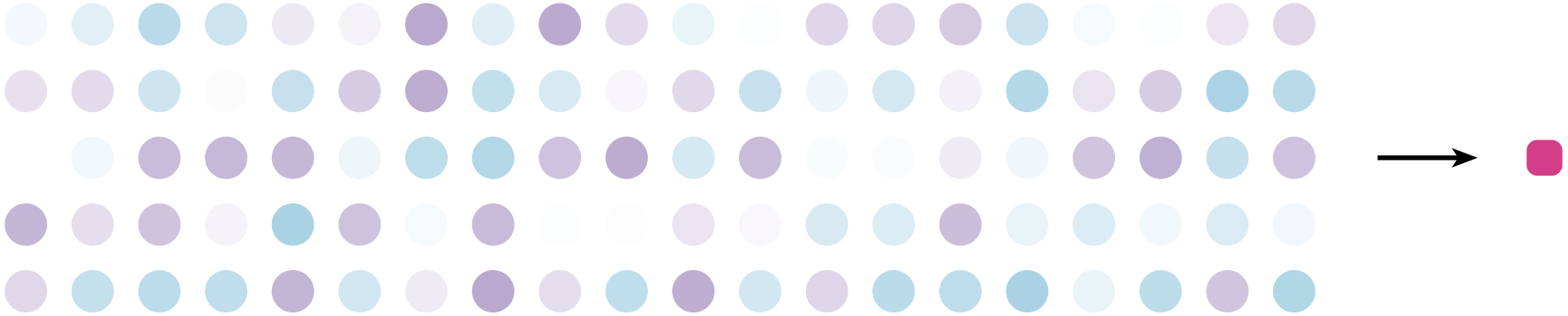
Time series tasks

Classification / labeling

- **Labeling**: Assign a class label to each time point (or segment)

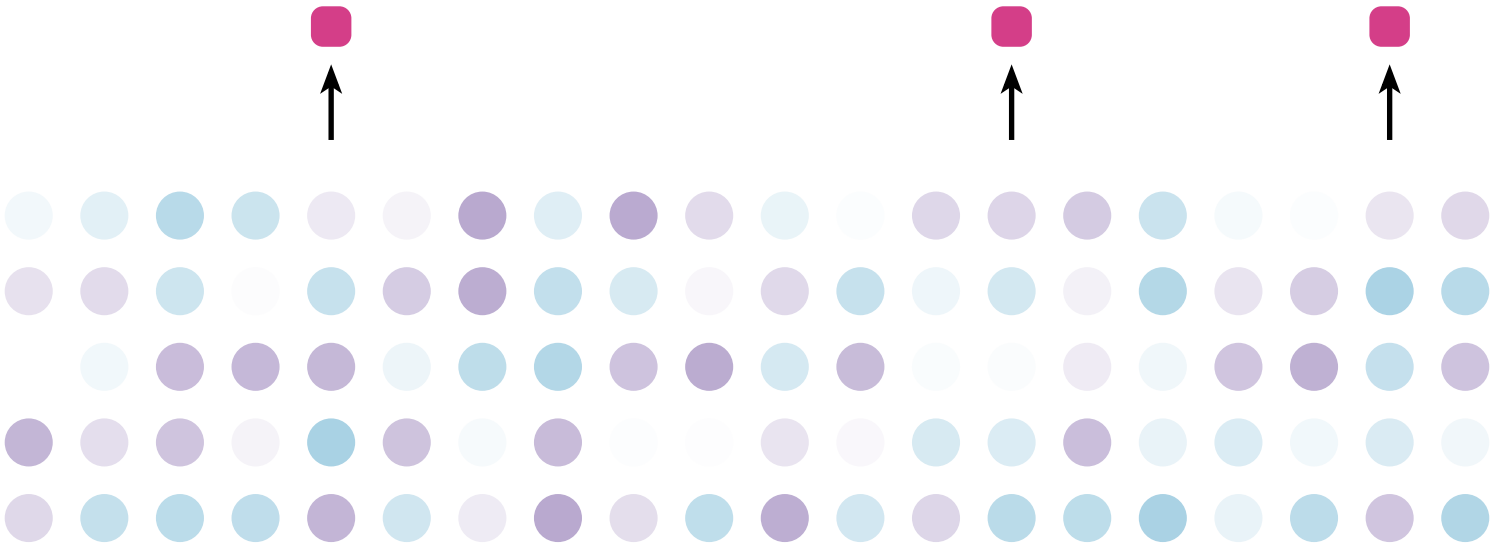


- **Classification**: Assign a class label to a full series



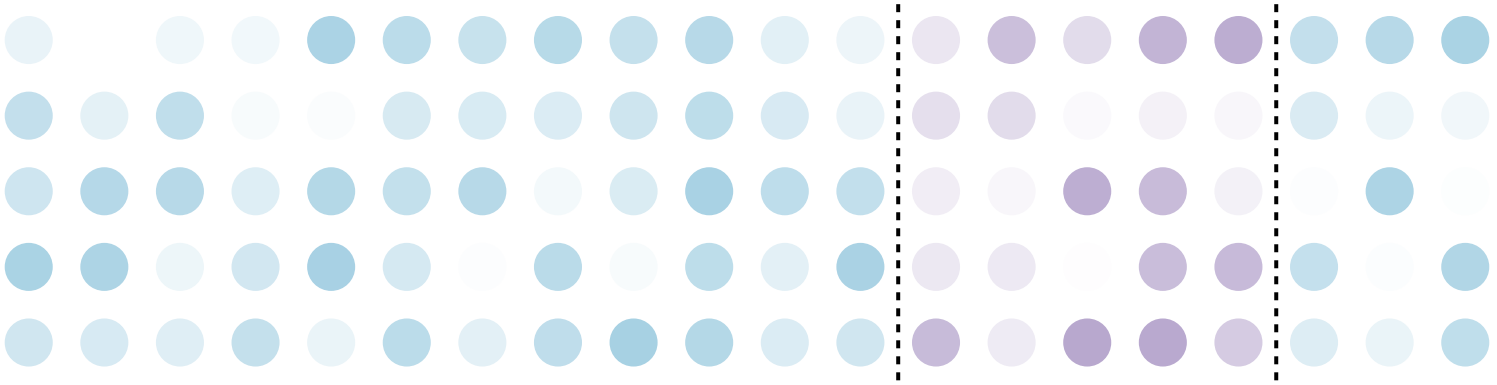
Anomaly detection

- Find unexpected points or segments
  - Usually unsupervised



Change-point detection

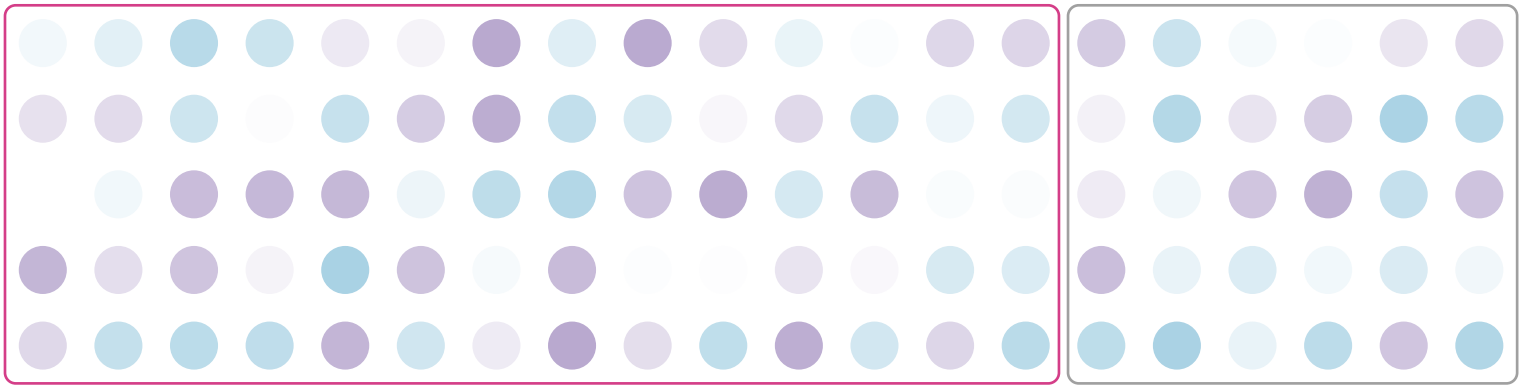
- Detect structural breaks in the sequence



Forecasting

- Regression task
- Target values are called **horizons**
- Model should capture temporal dependencies:

$$X_{\text{horizon}} \approx f\left(X_{\text{past}}\right)$$



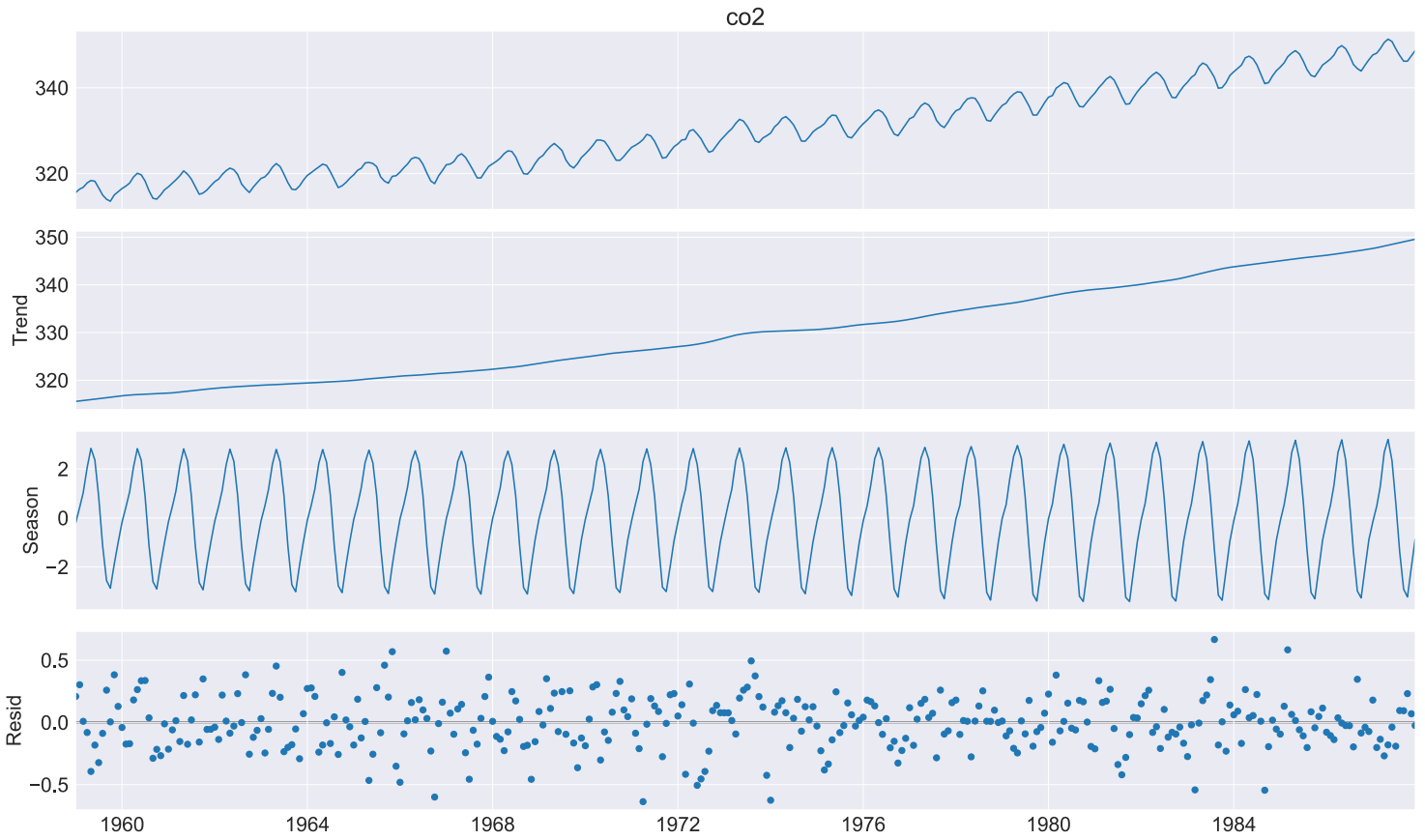
  - Modern pipelines: forecasting is **the** pretext task for feature learning

Old-school models

Additive decomposition

- Typical assumption: Time series can be decomposed into: Trend, Seasonality, and Residuals in an additive way:

$$x_t = T_t + S_t + R_t$$



ARIMA models

Autoregressive AR(p)

$$x_t = c + \sum_{i=1}^{p} \varphi_i x_{t-i} + \varepsilon_t$$

- Linear regression model based on a window of past values of size p.
- Captures temporal correlations via lags.

Moving Average MA(q)

$$x_t = c + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} + \varepsilon_t$$

- Models the effect of past residuals on the current value.

ARIMA models

ARMA(p,q)

$$x_t = c + \sum_{i=1}^{p} \varphi_i x_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} + \varepsilon_t$$

- Assumes stationarity.

ARIMA(p,d,q)

- Introduces differencing of order d to handle non-stationary trends.
- If $y_t = \nabla^d x_t$ (difference of order d), then y_t follows an ARMA(p,q).

Evaluation metrics and losses

Mean Squared Error MSE

$$\text{MSE}(x_t, \hat{x}_t) = \frac{1}{N}\sum_{t=1}^{N} \|x_t - \hat{x}_t\|_2^2$$

Mean Absolute Error MAE

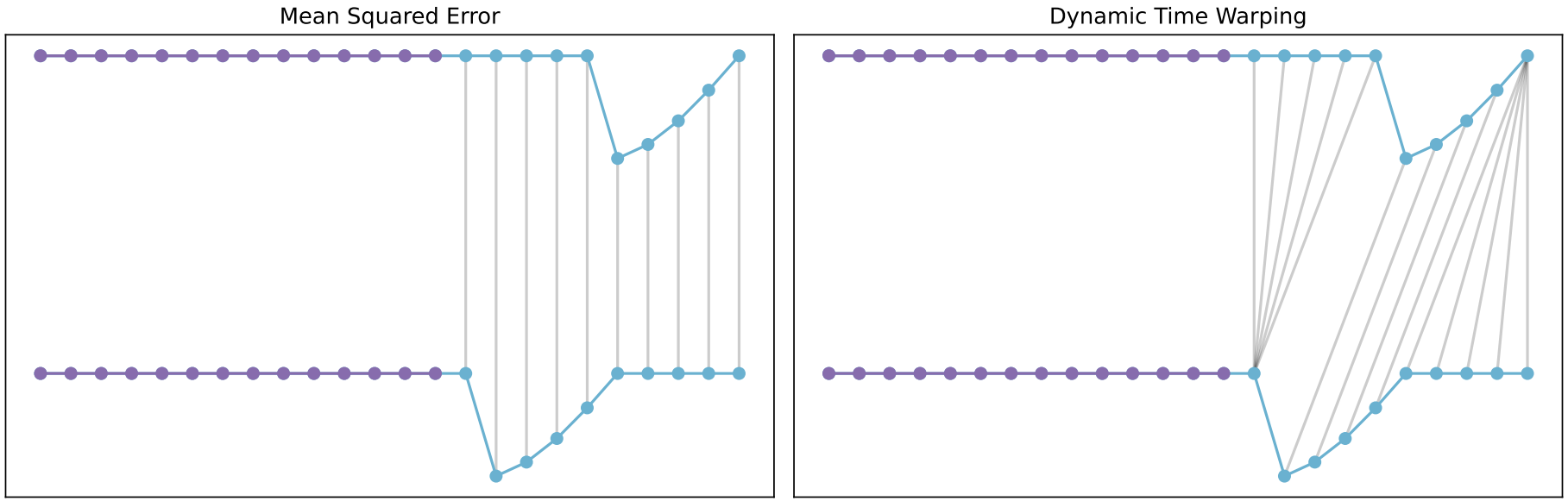
$$\text{MAE}(x_t, \hat{x}_t) = \frac{1}{N}\sum_{t=1}^{N} \|x_t - \hat{x}_t\|_1$$

Symmetric Mean Absolute Percentage Error sMAPE

$$\text{sMAPE}(x_t, \hat{x}_t) = \frac{2}{N}\sum_{t=1}^{N} \frac{\|x_t - \hat{x}_t\|_1}{\|x_t\|_1 + \|\hat{x}_t\|_1}$$

Differentiable losses

- MSE is the *de facto* standard loss for forecasting
- Shape-based losses exist however, eg:



- Aligns two time series by warping the time axis
- Useful when similar patterns occur at different time locations
- Relies on solving the following optimization problem:

$$\mathrm{DTW}\left(X, \hat{X}\right) = \min_{\pi \in \Pi} \sum_{t=1}^{T} \|x_{\pi_1(t)} - \hat{x}_{\pi_2(t)}\|_2^2$$

- Where Π is the set of all admissible alignments between the two series
- Can be solved via dynamic programming in $O(T^2)$ time

soft-DTW

- DTW is not differentiable
- soft-DTW is a differentiable variant
- Replaces min by soft-min in the DP recursion:

$$\mathrm{softDTW}^{\gamma}\left(X, \hat{X}\right) = \mathrm{softmin}^{\gamma}_{\pi \in \Pi} \sum_{t=1}^{T} \| x_{\pi_1(t)} - \hat{x}_{\pi_2(t)} \|_2^2$$

- Can be used as a loss in deep learning models
- Hyperparameter $\gamma > 0$ controls the smoothness
  - $\gamma \to 0$: soft-DTW approaches DTW
  - Large γ: more smoothing (MSE-like behaviour)