# Time Series Clustering

Machine Learning for Behavioral Data

May 1, 2023

# Today's Topic

| Week | Lecture/Lab |
|---|---|
| 8 | Spring Break |
| 9 | Time Series Prediction |
| **10** | **Unsupervised Learning** |
| 11 | Unsupervised Learning |
| 12 | Fairness |
| 13 | Explainability |
| 14 | Project Presentations |
| 15 | Whit Monday |

- **K-Means, Spectral Clustering**
- **Choosing the optimal K***
- **Clustering time-series data**

# Getting ready for today's lecture…

- **If not done yet**: clone the repository containing the Jupyter notebook and data for today's lecture into your Noto workspace.
- SpeakUp room for today's lecture:

**https://go.epfl.ch/speakup-mlbd**

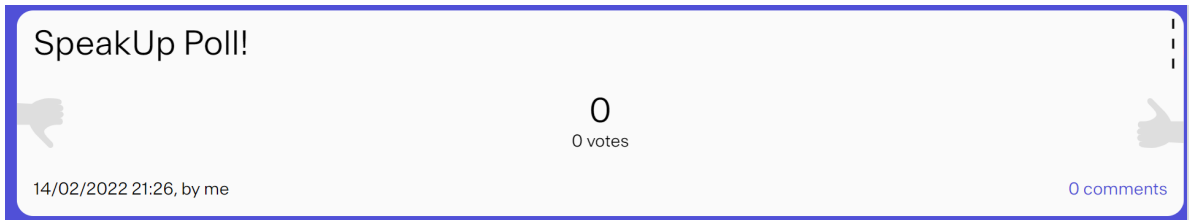
# Short quiz about the past…

In K-Means, which of the following parameters affect the goodness of the solution?

a) Number of iterations

b) Initial positioning of cluster centers

c) Choice of k

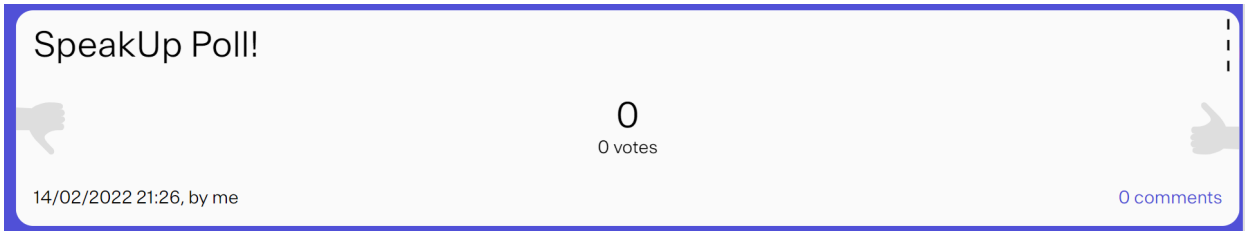


# Short quiz about the past…

K-Means is useful when dealing with non-convex clusters:

a) True

b) False



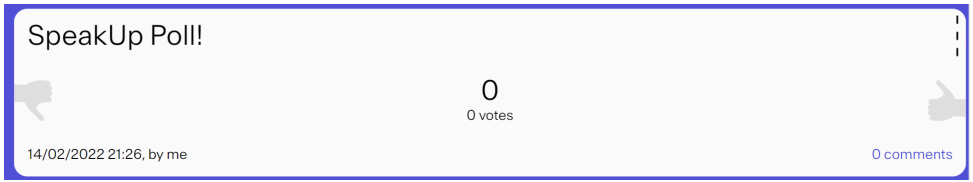
# Short quiz about the past…

In a binary classification problem, it is appropriate to use the following activation function for the output layer:

a) Linear

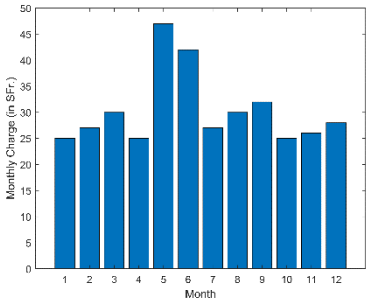
b) Tanh

c) Sigmoid

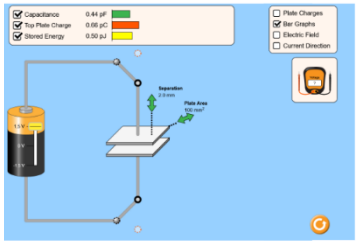


# Today – Clustering Time Series Data

1. Aggregating features over time
2. Defining fixed time intervals (weeks, levels in a game, etc.)
3. Dynamic Time Warping




4. String Metrics
5. Markov Models



Action Sequences

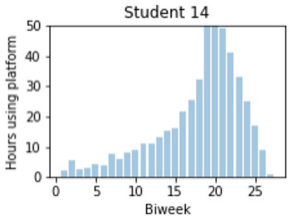
# Learning Objectives

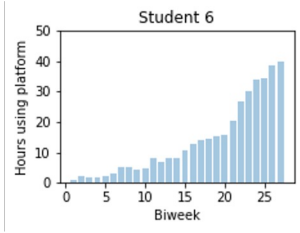
You should be able to:

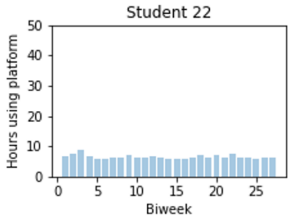
- Explain the different approaches to time series clustering
- Describe their advantages and disadvantages and when it is appropriate to use them
- Implement these approaches (lecture/lab session)
- Apply them to real-world data (lab session)

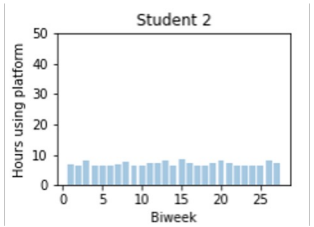
# Today's Use Case

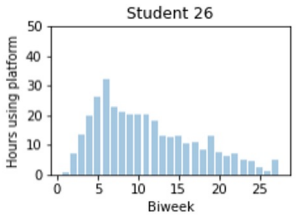
- Synthetic data of 30 high school students
- Time spent on an e-learning platform over one year (computed per biweek)
- Three clusters: 1) precrastinators, 2) regular, 3) procrastinators

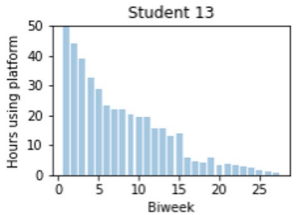


# Agenda

- **Aggregating features over time**
- Defining fixed time intervals (weeks, levels in a game, etc.)
- Dynamic Time Warping
- String Metrics
- Markov Models

# Aggregating features over time

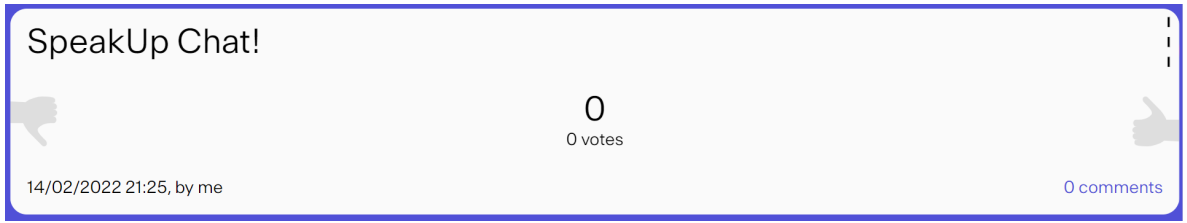
- We compute the value of the feature over the whole time series (average, maximum, range, standard deviation)
- We do not explicitly represent changes in features over time

➡ We can use standard distance/similarity measures

# Your Turn – Aggregated Data

Run spectral clustering on the average number of hours:

- Can we interpret the different clusters?
- Are we able to retrieve the procrastination patterns? If not, why not?



# Agenda

- Aggregating features over time
- **Defining fixed time intervals (weeks, levels in a game, etc.)**
- Dynamic Time Warping
- String Metrics
- Markov Models
- Additional Practice

# Using fixed time intervals

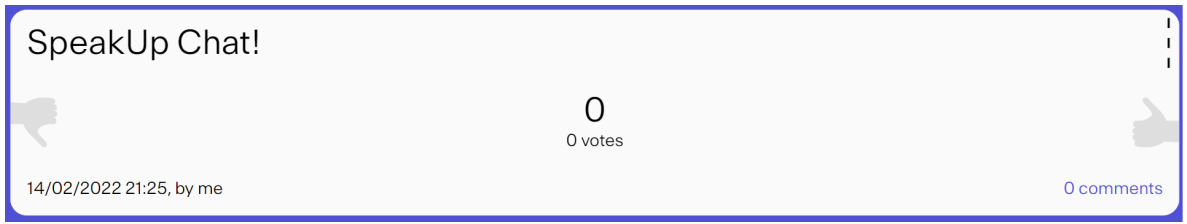
- Compute the feature value at fixed points in time (e.g., weeks, level in a game)
- We obtain feature vectors with the same length for every student

➡ We can use standard distance measures

# Your Turn – Fixed Time Intervals

Run spectral clustering on the vectors of biweeks (dimension = 27) using Euclidean distance:

- What is the optimal number of clusters?
- How do the results differ from the aggregated feature results?



# Agenda

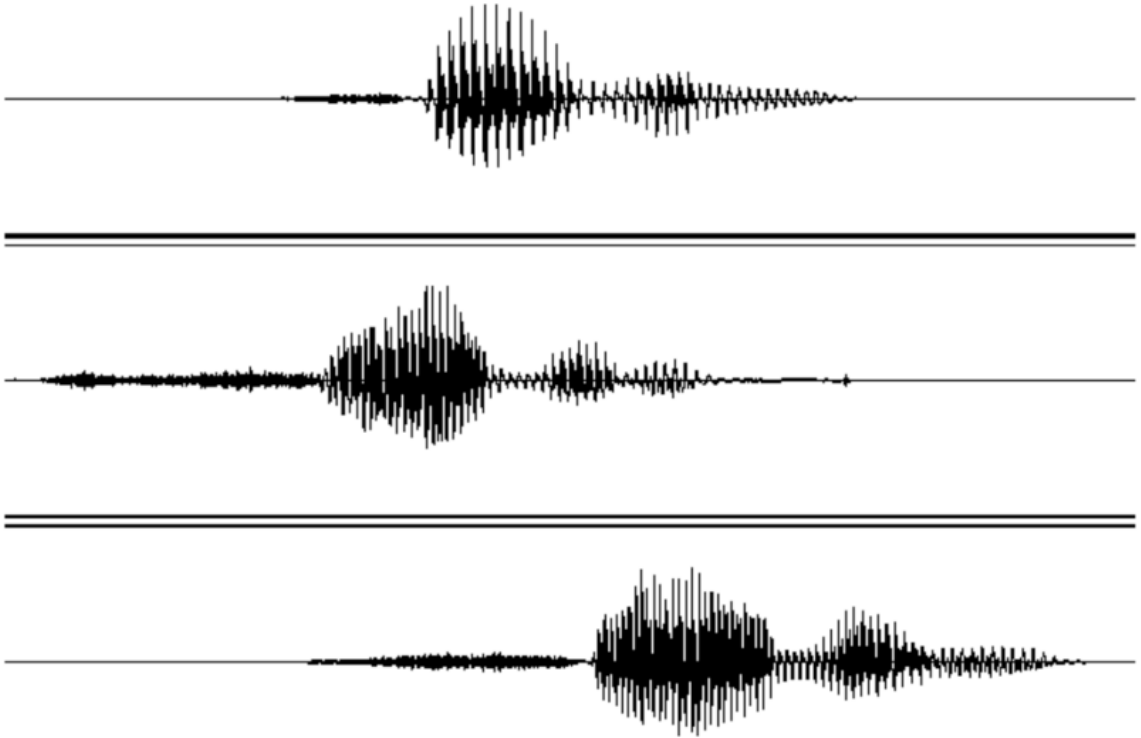
- Aggregating features over time
- Defining fixed time intervals (weeks, levels in a game, etc.)
- **Dynamic Time Warping**
- String Metrics
- Markov Models
- Additional Practice (if time permits)

# Dynamic Time Warping

- Compute distance between two time series, which may vary in speed
- Time series can have different lengths

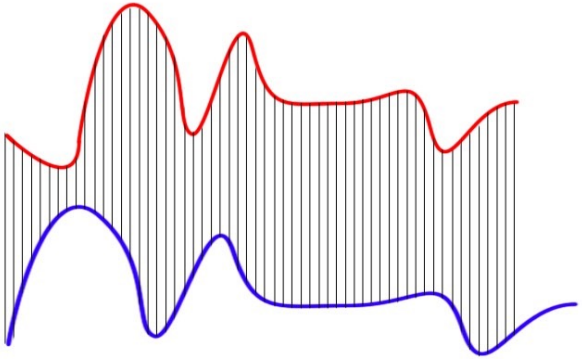
➡ Develop a **one-to-many** match, i.e. find an *optimal alignment* between two time series

# Example: Spoken Digits

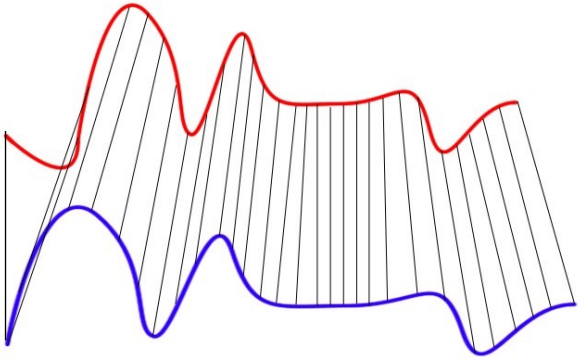


[*Image Credit: CS 498, Signals AI, University of Illinois*]

# Dynamic Time Warping vs. Euclidean Distance



Euclidean Distance



Dynamic Time Warping

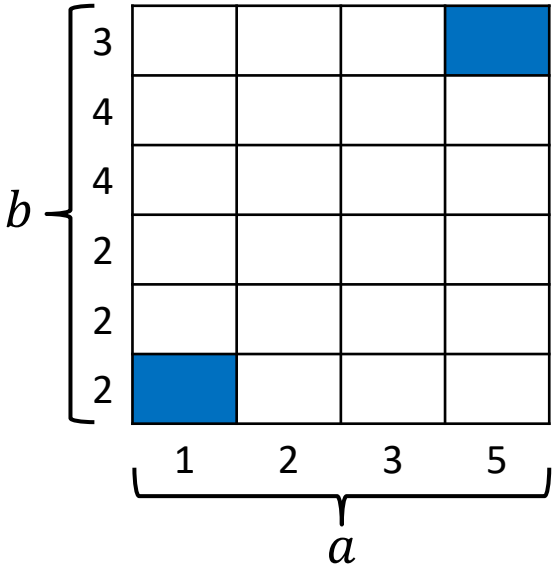
[*Image Credit: https://towardsdatascience.com/dynamic-time-warping-3933f25fcdd*]

# Dynamic Time Warping: Rules

- **Goal**: minimize $D(a, b) = \min_{\emptyset} \sum_k d(a_{\emptyset(k)}, b_{\emptyset(k)})$
- **Rules** (given two sequences $\boldsymbol{a}$ and $\boldsymbol{b}$):
  - Every index of $\boldsymbol{a}$ must be matched with one or more indices from $\boldsymbol{b}$, and vice versa
  - The first index from $\boldsymbol{a}$ must be matched with the first index from $\boldsymbol{b}$ (but it does not have to be its only match)
  - The last index from $\boldsymbol{a}$ must be matched with the last index from $\boldsymbol{b}$ (but it does not have to be its only match)
  - The mapping of the indices from $\boldsymbol{a}$ to indices from $\boldsymbol{b}$ must be monotonically increasing, and vice versa, i.e. if $j > i$ are indices from $\boldsymbol{a}$, then there must not be two indices $m > n$ in $\boldsymbol{b}$, such that index $i$ is matched with index $m$ and index $j$ is matched with index $n$, and vice versa

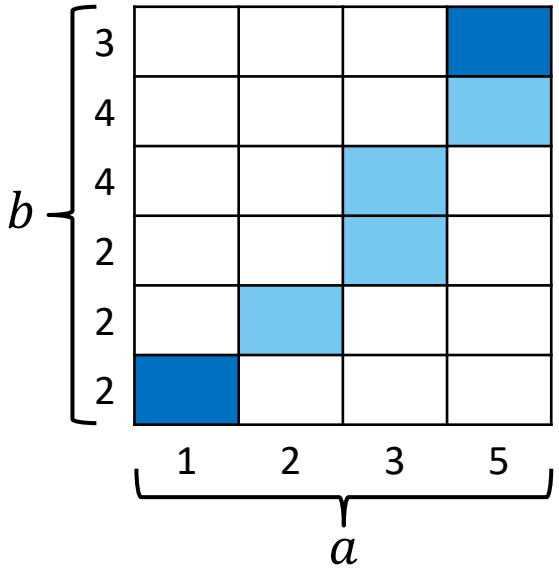
# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



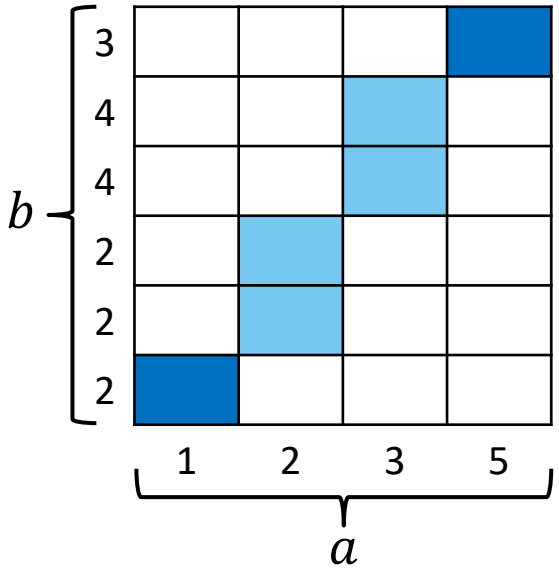
# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$



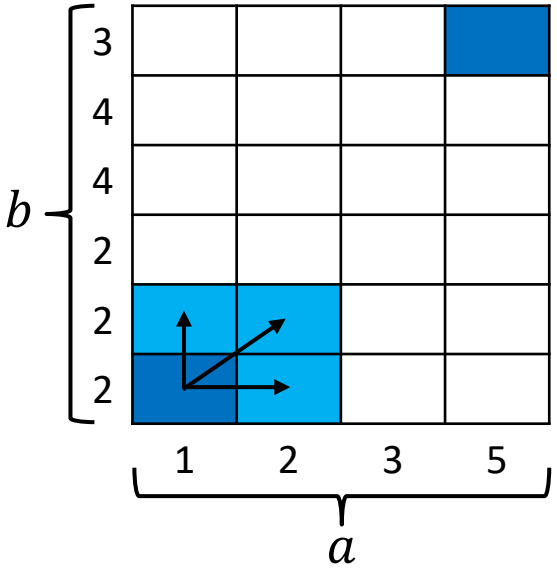
# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$



# Dynamic Time Warping: Possible Paths

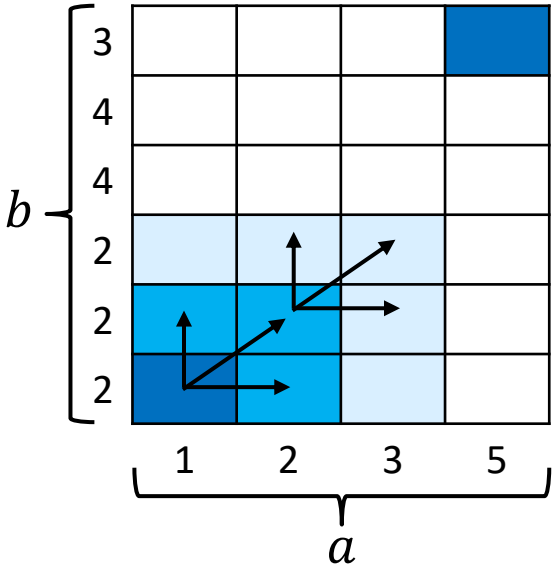
$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




- Three possible paths from each square

# Dynamic Time Warping: Possible Paths

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$

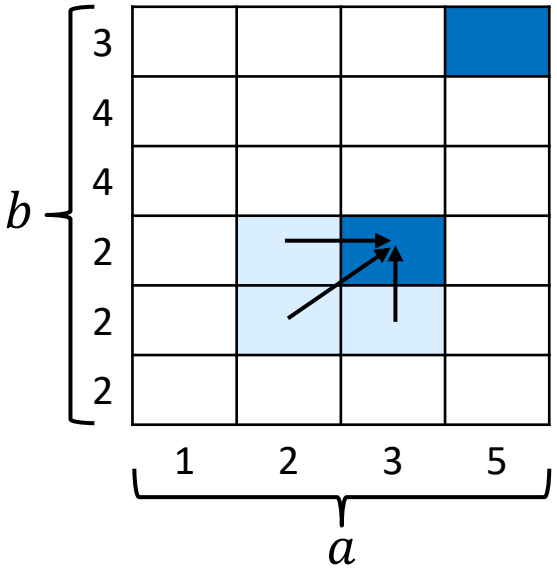



- Three possible paths from each square
- Every choice leads to three more possible paths

➡ $\approx 3^{4 \cdot 6}$ options

# Dynamic Time Warping: Minimum Path

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$

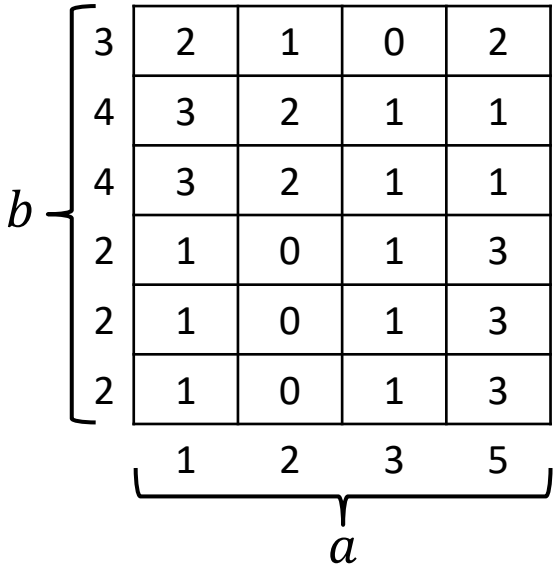



- For any cell $C$ (matching indices $i, j$): three possible precursor cells
- Minimum cost (distance) for getting to $C$

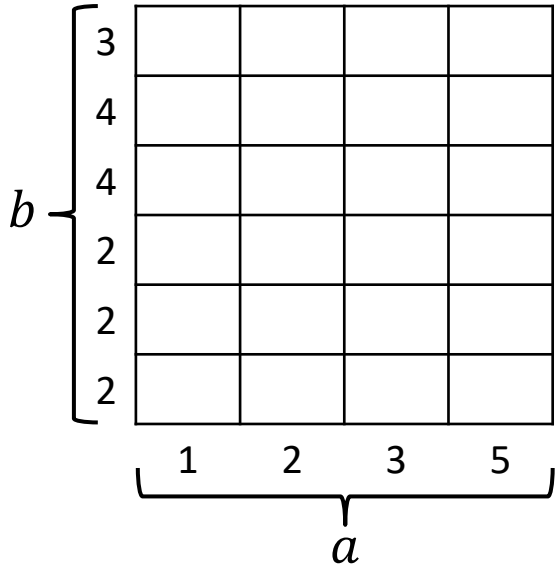
$$d(i,j) + \min(D(i-1,j), D(i-1,j-1), D(i,j-1))$$

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



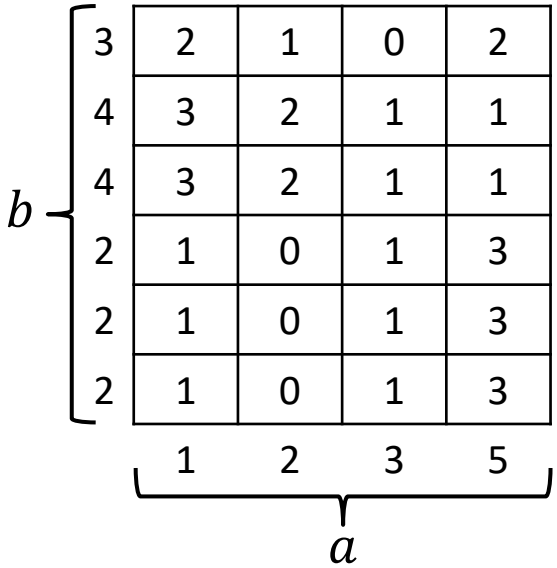

1. Compute pairwise distances



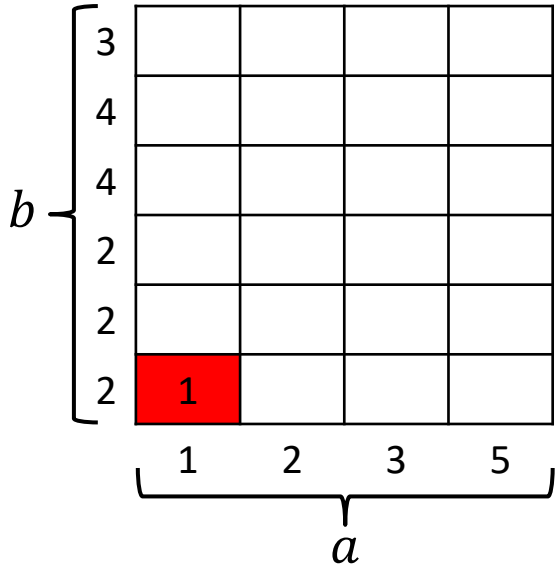

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



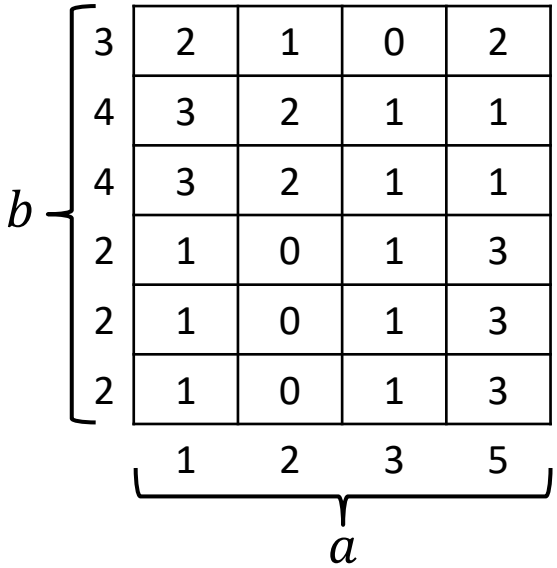

1. Compute pairwise distances



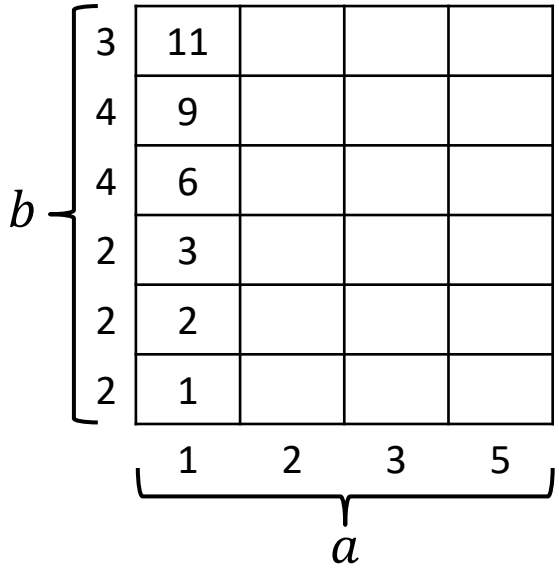

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



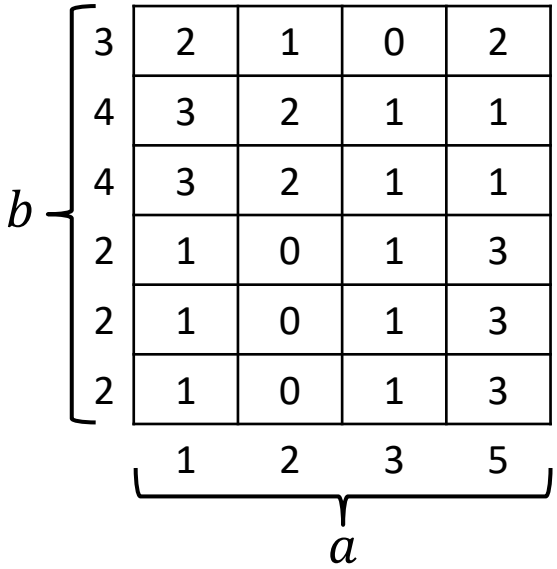

1. Compute pairwise distances



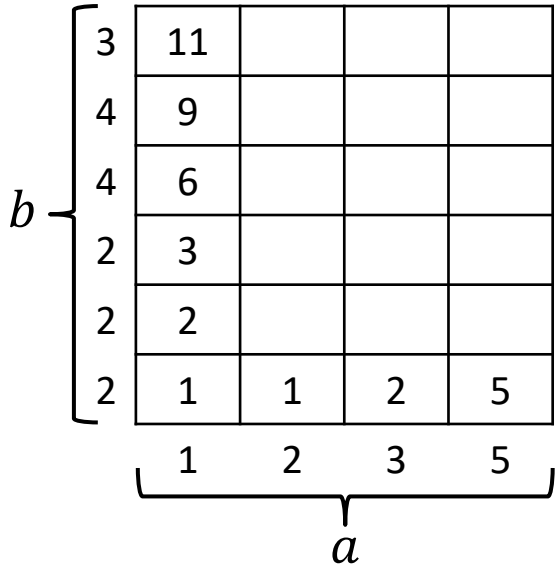

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



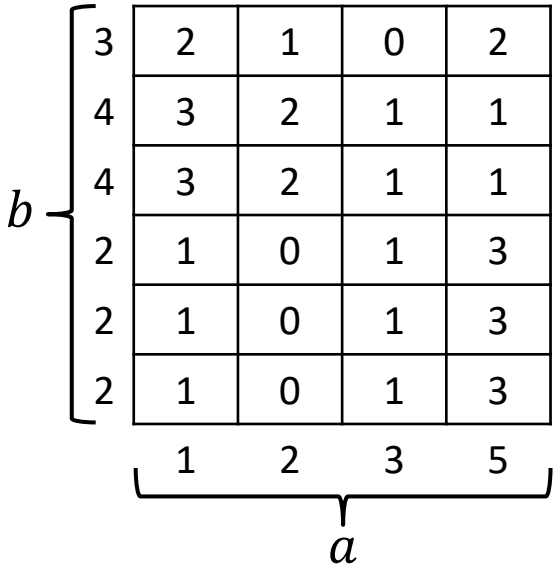

1. Compute pairwise distances



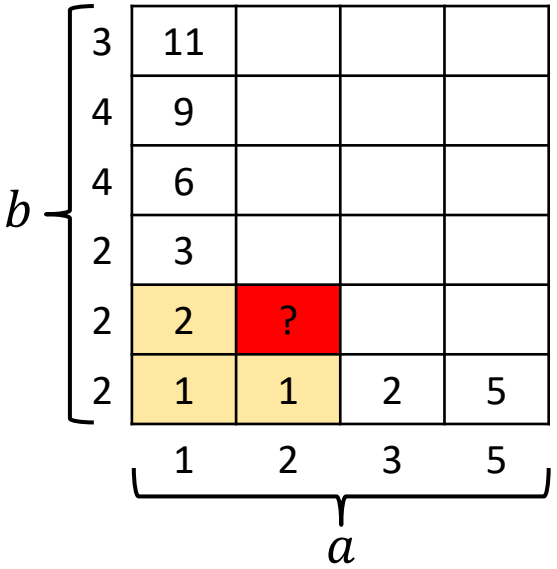

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



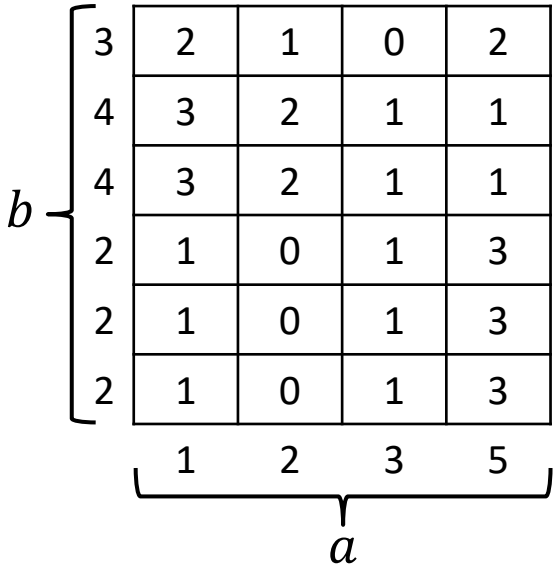

1. Compute pairwise distances



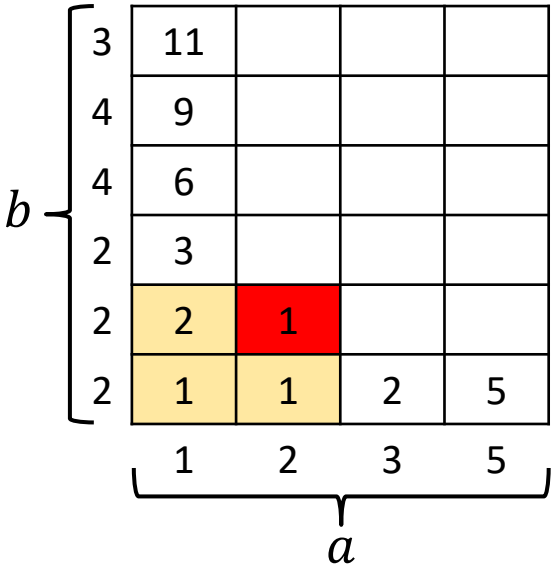

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$




1. Compute pairwise distances




2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

| $b$ | | | | |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| | 1 | 2 | 3 | 5 |
| | $a$ | | | |

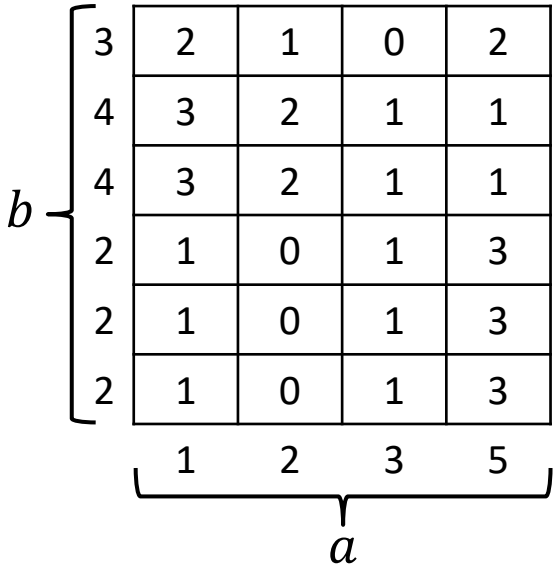
1. Compute pairwise distances

| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | | | |
| 4 | 9 | | | |
| 4 | 6 | | | |
| 2 | 3 | | | |
| 2 | 2 | 1 | ? | |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |
| | $a$ | | | |

2. Compute minimum distance path

# Dynamic Time Warping: Example

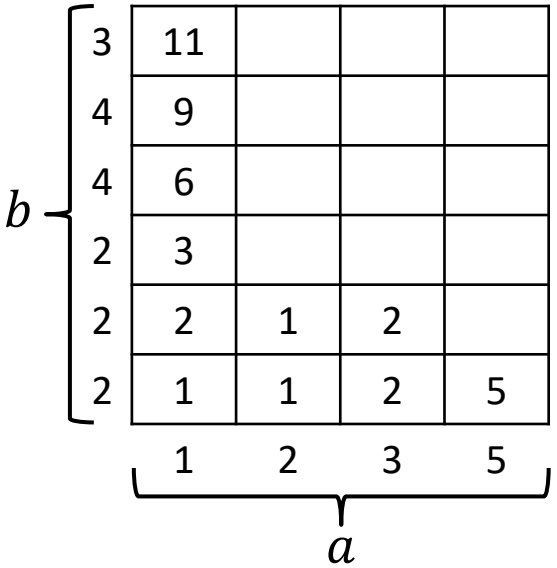
$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$



| $b$ | | | | |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| | 1 | 2 | 3 | 5 |

$a$

1. Compute pairwise distances



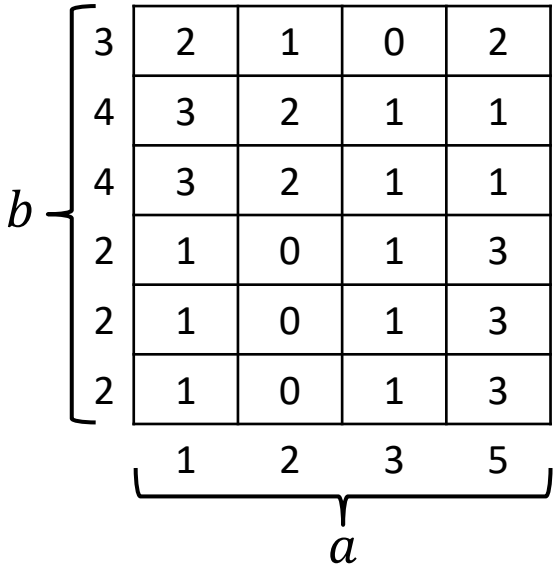
| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | | | |
| 4 | 9 | | | |
| 4 | 6 | | | |
| 2 | 3 | | | |
| 2 | 2 | 1 | 2 | |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |

$a$

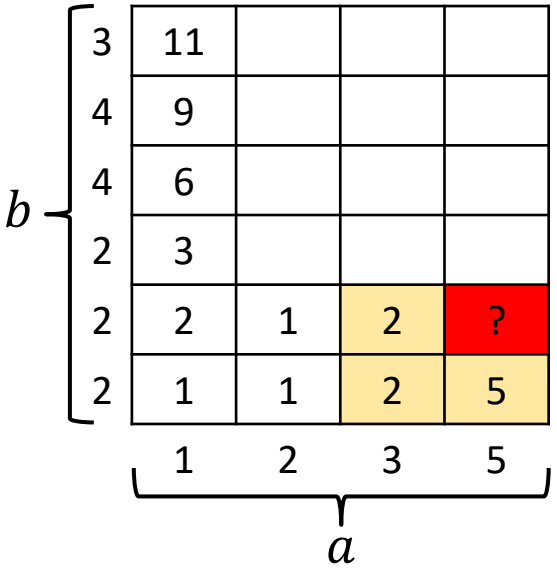
2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$



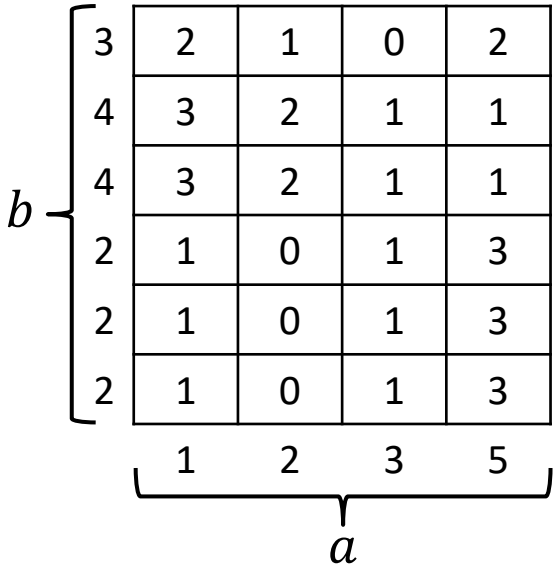

1. Compute pairwise distances



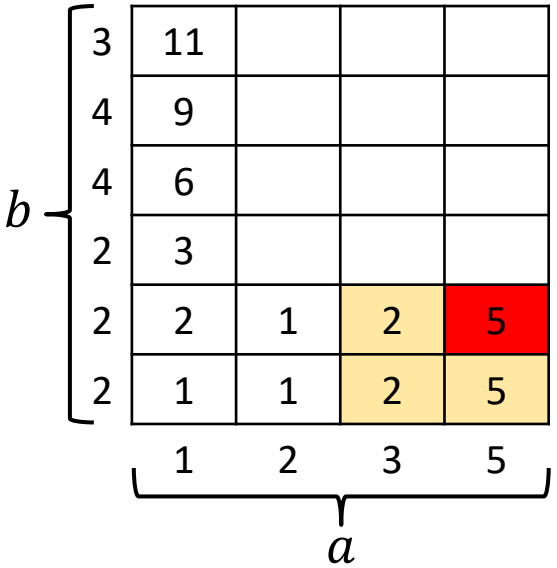

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$



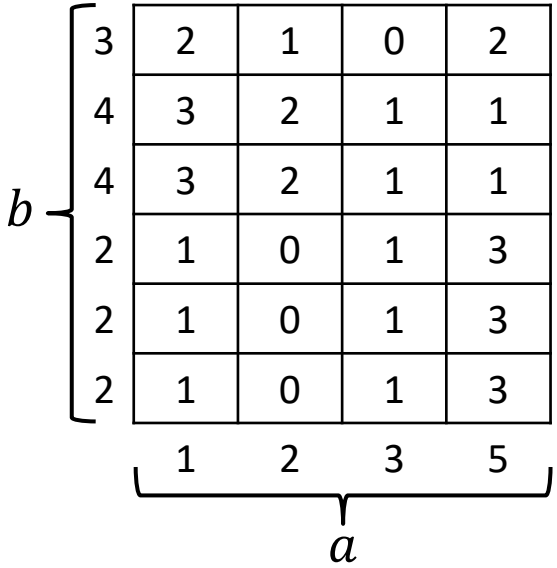

1. Compute pairwise distances




2. Compute minimum distance path

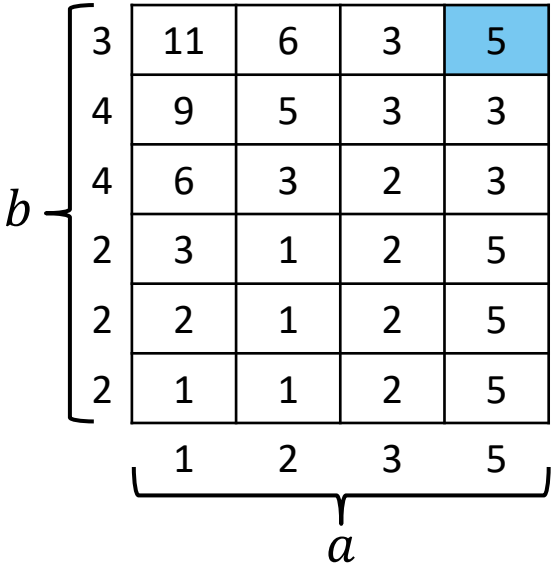
# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




1. Compute pairwise distances

Goal




2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

Goal

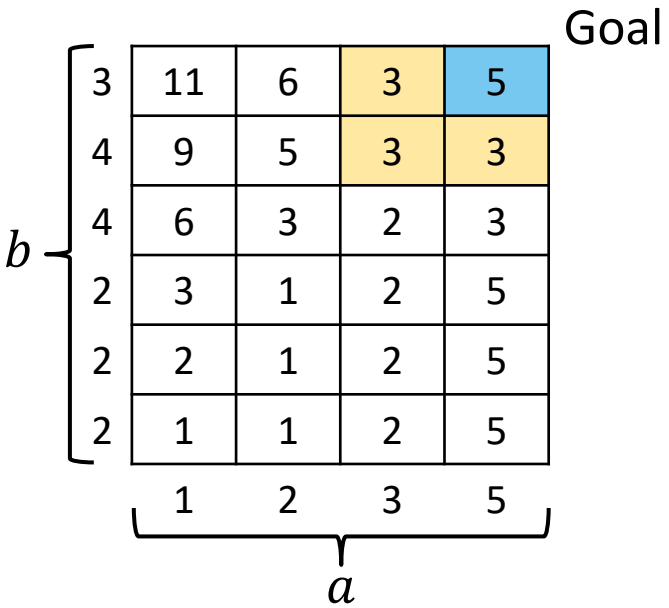
| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | 6 | 3 | 5 |
| 4 | 9 | 5 | 3 | 3 |
| 4 | 6 | 3 | 2 | 3 |
| 2 | 3 | 1 | 2 | 5 |
| 2 | 2 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |

$a$

2. Compute minimum distance path

# Dynamic Time Warping: Example

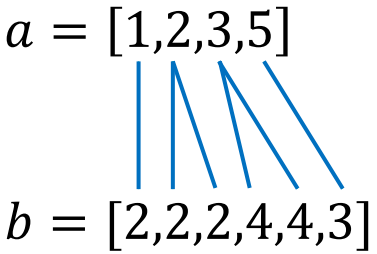
$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




Goal

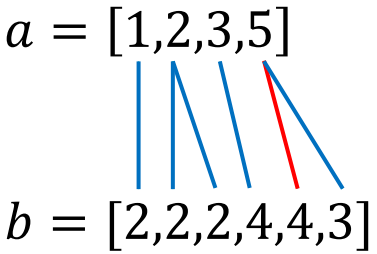
| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | 6 | 3 | 5 |
| 4 | 9 | 5 | 3 | 3 |
| 4 | 6 | 3 | 2 | 3 |
| 2 | 3 | 1 | 2 | 5 |
| 2 | 2 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |

$a$

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




Goal

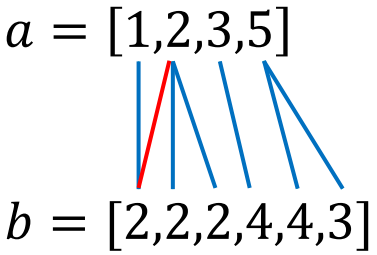
| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | 6 | 3 | 5 |
| 4 | 9 | 5 | 3 | 3 |
| 4 | 6 | 3 | 2 | 3 |
| 2 | 3 | 1 | 2 | 5 |
| 2 | 2 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |

$a$

2. Compute minimum distance path

# Dynamic Time Warping: Example

$a = [1,2,3,5] \quad b = [2,2,2,4,4,3]$




Goal

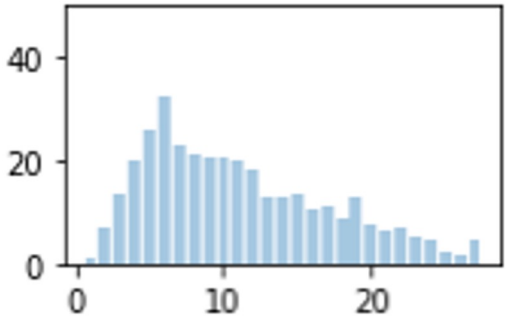
| $b$ | | | | |
|---|---|---|---|---|
| 3 | 11 | 6 | 3 | 5 |
| 4 | 9 | 5 | 3 | 3 |
| 4 | 6 | 3 | 2 | 3 |
| 2 | 3 | 1 | 2 | 5 |
| 2 | 2 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | 5 |
| | 1 | 2 | 3 | 5 |

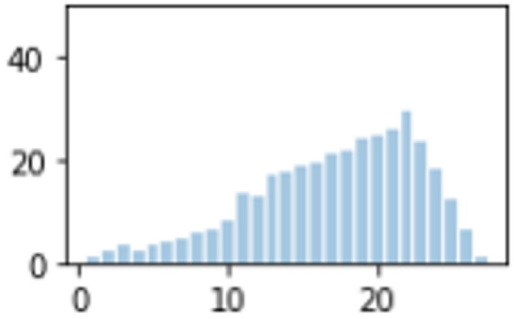
$a$

2. Compute minimum distance path

# Dynamic Time Warping: Window

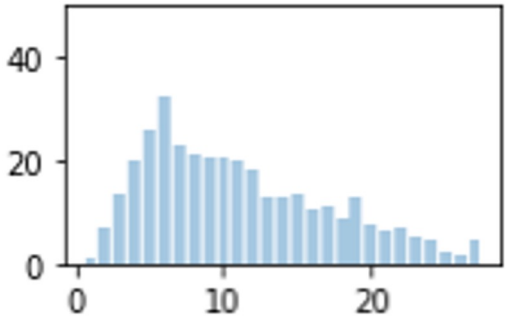
- Sometimes, we might want to constrain the mapping

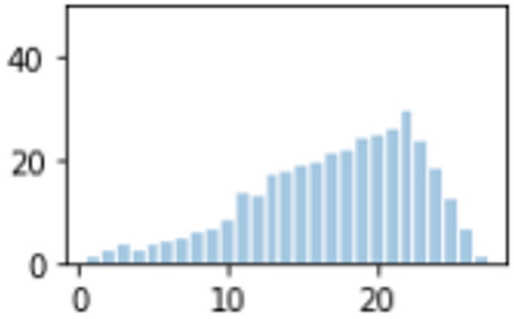


# Dynamic Time Warping: Window

- Sometimes, we might want to constrain the mapping




- We introduce an window size $w$: an element in sequence $\boldsymbol{a}$ at index $i$ can only be mapped to elements at index $i - w, \ldots, i + w$ in sequence $\boldsymbol{b}$

# Dynamic Time Warping: $w = 2$

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

| $b$ | 1 | 2 | 3 | 5 |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |

$a$

1. Compute pairwise distances

| $b$ | 1 | 2 | 3 | 5 |
|---|---|---|---|---|
| 3 | | | | |
| 4 | | | | |
| 4 | | | | |
| 2 | | | | |
| 2 | | | | |
| 2 | | | | |

$a$

2. Compute minimum distance path

# Dynamic Time Warping: $w = 2$

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

| $b$ | | | | |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| | 1 | 2 | 3 | 5 |

$a$

1. Compute pairwise distances

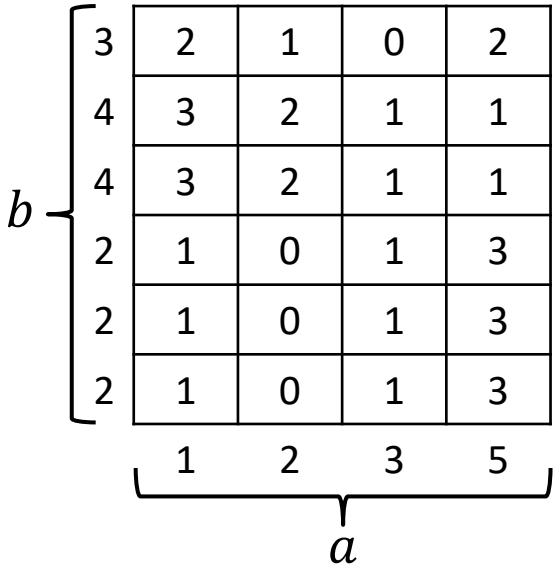
| $b$ | | | | |
|---|---|---|---|---|
| 3 | - | | | |
| 4 | - | | | |
| 4 | - | | | |
| 2 | 3 | | | |
| 2 | 2 | | | |
| 2 | 1 | | | |
| | 1 | 2 | 3 | 5 |

$a$

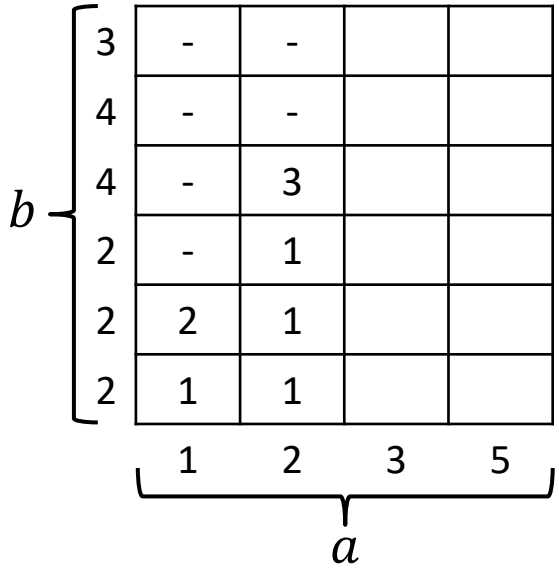
2. Compute minimum distance path

# Dynamic Time Warping: $w = 2$

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$




1. Compute pairwise distances




2. Compute minimum distance path

# Dynamic Time Warping: $w = 2$

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

| $b$ | | | | |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| | 1 | 2 | 3 | 5 |

$a$

1. Compute pairwise distances

| $b$ | | | | |
|---|---|---|---|---|
| 3 | - | - | - | |
| 4 | - | - | 3 | |
| 4 | - | - | 2 | |
| 2 | - | 1 | 2 | |
| 2 | 2 | 1 | 2 | |
| 2 | 1 | 1 | 2 | |
| | 1 | 2 | 3 | 5 |

$a$

2. Compute minimum distance path

# Dynamic Time Warping: $w = 2$

$a = [1,2,3,5]$ $b = [2,2,2,4,4,3]$

| $b$ | | | | |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | 2 |
| 4 | 3 | 2 | 1 | 1 |
| 4 | 3 | 2 | 1 | 1 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| 2 | 1 | 0 | 1 | 3 |
| | 1 | 2 | 3 | 5 |

$a$

1. Compute pairwise distances

| $b$ | | | | |
|---|---|---|---|---|
| 3 | - | - | - | 5 |
| 4 | - | - | - | 3 |
| 4 | - | - | 2 | 3 |
| 2 | - | 1 | 2 | 5 |
| 2 | 2 | 1 | 2 | 5 |
| 2 | 1 | 1 | 2 | - |
| | 1 | 2 | 3 | 5 |

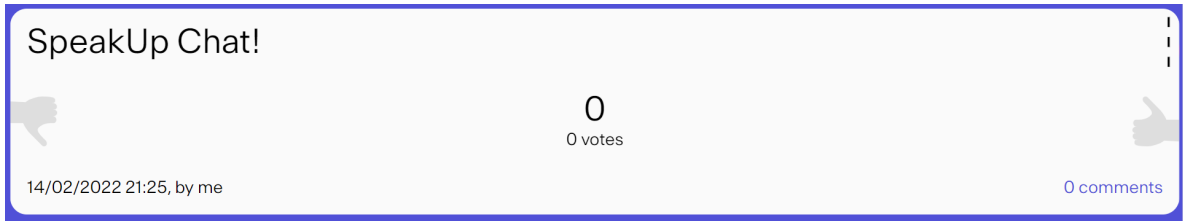
$a$

2. Compute minimum distance path

# Your Turn – Dynamic Time Warping

Run spectral clustering using DTW with a window size of $w = 3$:

- How do the results differ from previous results?
- What happens if you set $w = 0$?
- And if you set $w = 27$?



# Agenda

- Aggregating features over time
- Defining fixed time intervals (weeks, levels in a game, etc.)
- Dynamic Time Warping
- **String Metrics**
- Markov Models

# Example from Research: String Metrics


$$C1 \rightarrow C2 \rightarrow C3 \rightarrow C4 \rightarrow R4 \rightarrow P \rightarrow C5 \rightarrow R5 \rightarrow P$$

$$C4 \rightarrow R4 \rightarrow P \rightarrow C5 \rightarrow R5 \rightarrow P$$

[*Visualization and Confirmatory Clustering of Sequence Data from a Simulation-Based Assessment Task*, Berger et al., EDM 2014]

# Example from Research: String Metrics

- Levensthein distance: minimal number of single character edits (insertion, deletion, substitution) to change one string into the other
- Longest common subsequence (LCS): string similarity measure, find the longest common subsequence between two sequences

[*Visualization and Confirmatory Clustering of Sequence Data from a Simulation-Based Assessment Task*, Berger et al., EDM 2014]

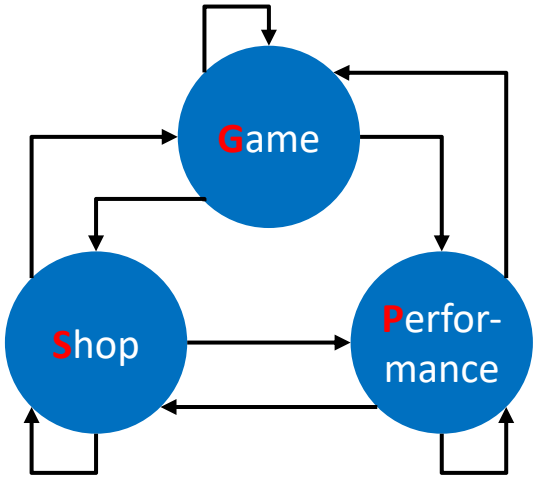
# Agenda

- Aggregating features over time
- Defining fixed time intervals (weeks, levels in a game, etc.)
- Dynamic Time Warping
- String Metrics
- **Markov Models**

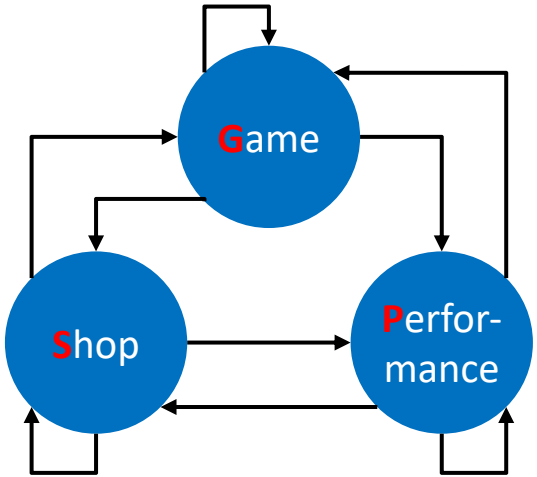
# Markov Models

- Detailed action sequences provide rich temporal information
- Might contain a considerable amount of noise
- We might be interested not in the detailed sequence, but in patterns (which actions tend to follow each other)

# Markov Models



# Markov Models




$$G \to S \to G \to S \to G \to P \to S \to G$$

$$G \to G \to G \to G \to P \to G \to G$$

$$G \to P \to S \to G \to P \to S$$

# Parameters: Maximum Likelihood Estimation

$$G \rightarrow S \rightarrow G \rightarrow S \rightarrow G \rightarrow P \rightarrow S \rightarrow G \rightarrow S \rightarrow S \rightarrow G \rightarrow G \rightarrow S \rightarrow G \rightarrow P \rightarrow G \rightarrow P \rightarrow S \rightarrow G \rightarrow S \rightarrow \mathrm{G} \rightarrow S \rightarrow S \rightarrow G \rightarrow S \rightarrow \mathrm{G} \rightarrow G \rightarrow S \rightarrow \mathrm{G} \rightarrow S \rightarrow S \rightarrow G \rightarrow S$$

$$p(S|G) = \frac{10}{15} = 0.67$$

$$p(G|G) = \frac{2}{15} = 0.13$$

$$p(P|G) = \frac{3}{15} = 0.20$$

$$\begin{array}{c} \\ \boldsymbol{G} \\ \boldsymbol{S} \\ \boldsymbol{P} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{G} & \boldsymbol{S} & \boldsymbol{P} \end{array} \\ \begin{pmatrix} 0.13 & 0.67 & 0.20 \\ 0.79 & 0.11 & 0 \\ 0.33 & 0.67 & 0 \end{pmatrix} \end{array}$$

# Stationary Distribution

$$G \to S \to G \to S \to G \to P \to S \to G \to S \to S \to G \to G \to S \to G \to P \to G \to$$
$$P \to S \to G \to S \to \mathrm{G} \to S \to S \to G \to S \to \mathrm{G} \to G \to S \to \mathrm{G} \to S \to S \to G \to S$$

$$\begin{array}{c} \\ \boldsymbol{G} \\ \boldsymbol{S} \\ \boldsymbol{P} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{G} & \boldsymbol{S} & \boldsymbol{P} \end{array} \\ \begin{pmatrix} 0.13 & 0.67 & 0.20 \\ 0.89 & 0.11 & 0 \\ 0.33 & 0.67 & 0 \end{pmatrix} \end{array}$$

$$\pi \mathrm{T} = \pi$$

# Stationary Distribution

$$G \to S \to G \to S \to G \to P \to S \to G \to S \to S \to G \to G \to S \to G \to P \to G \to P \to S \to G \to S \to \mathrm{G} \to S \to S \to G \to S \to \mathrm{G} \to G \to S \to \mathrm{G} \to S \to S \to G \to S$$

$$\begin{array}{c} \\ \boldsymbol{G} \\ \boldsymbol{S} \\ \boldsymbol{P} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{G} & \boldsymbol{S} & \boldsymbol{P} \end{array} \\ \begin{pmatrix} 0.13 & 0.67 & 0.20 \\ 0.89 & 0.11 & 0 \\ 0.33 & 0.67 & 0 \end{pmatrix} \end{array}$$

$$\pi \mathrm{T} = \pi$$

$$\pi = [0.48 \quad 0.43 \quad 0.09]$$

# Expected Frequencies

- When sequences get very long (n gets large), how often do we expect to observe the transitions?

$$\begin{array}{c} \\ \boldsymbol{G} \\ \boldsymbol{S} \\ \boldsymbol{P} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{G} & \boldsymbol{S} & \boldsymbol{P} \end{array} \\ \begin{pmatrix} 0.06 & 0.32 & 0.10 \\ 0.38 & 0.05 & 0 \\ 0.03 & 0.06 & 0 \end{pmatrix} \end{array}$$

# Distance Metrics

- Based on Frobenius Norm: equivalent to Euclidean distance over vectors

$$D_2(A,B) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m}\left(a_{ij} - b_{ij}\right)^2}$$

# Distance Metrics

- Kullback-Leibler Divergence: measures difference between two probability distributions

$$D_{KL}(P||Q) = \sum_{x \in \mathrm{X}} P(x) \cdot \log(\frac{P(x)}{Q(x)})$$

- Jensen-Shannon Divergence: measures difference between two probability distributions

$$D_{JS}(P||Q) = \frac{1}{2} D_{KL}(P||M) + \frac{1}{2} D_{KL}(M||Q) \qquad \mathrm{M} = \frac{1}{2}(P + Q)$$

# Distance Metrics

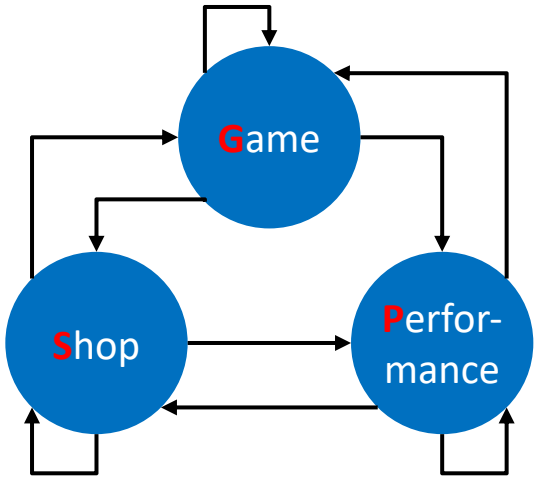
- Hellinger Distance: measures difference between two probability distributions

$$D_H(P||Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{n} (\sqrt{p_i} - \sqrt{q_i})^2}$$

# Distance between samples: Options

- Compute distance between stationary distributions: use Hellinger Distance (or Jensen-Shannon Divergence)
- Compute distance between transition matrices: use Frobenius Distance
- Compute distance between expected frequencies: use Hellinger Distance (or Jensen-Shannon Divergence)

# Example from Research: Spelling Learning



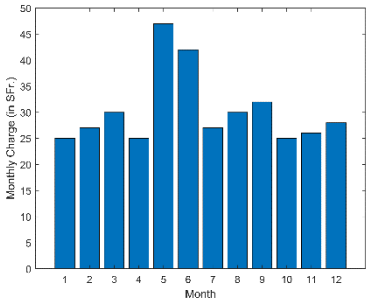

**Three clusters**:

- Focused on the task
- Children, who frequently check performance/shop in-between tasks
- Spend long amounts of time off-task

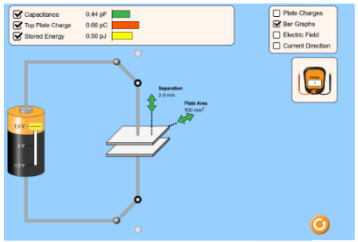
[*Temporally Coherent Clustering of Student Data*, Klingler et al., EDM 2016]

# Summary - Handling Time Series Data

1. Aggregating features over time
2. Defining fixed time intervals (weeks, levels in a game, etc.)
3. Dynamic Time Warping




4. String Measures
5. Markov Models



Action Sequences