Time Series and Dynamic Time Warping

CSE 4309 – Machine Learning

Vassilis Athitsos
Computer Science and Engineering Department
University of Texas at Arlington

Sequential Data

- **Sequential data**, as the name implies, are **sequences**.
- What is the difference between a **sequence** and a **set**?
- A sequence $\boldsymbol{X}$ is a set of elements, together with a **total order** imposed on those elements.
  - A **total order** describes, for any two elements $\boldsymbol{x_1}, \boldsymbol{x_2}$, which of them comes before and which comes after.
- Examples of sequential data:
  - Strings: sequences of characters.
  - Time series: sequences of vectors.

Time Series

- A **time series** is a **sequence** of observations made over time.
- Examples:
  - Stock market prices (for a single stock, or for multiple stocks).
  - Heart rate of a patient over time.
  - Position of one or multiple people/cars/airplanes over time.
  - Speech: represented as a sequence of audio measurements at discrete time steps.
  - A musical melody: represented as a sequence of pairs (note, duration).

Applications of Time Series Classification

- Predicting future prices (stock market, oil, currencies…).
- Heart rate of a patient over time:
  - Is it indicative of a healthy heart, or of some disease?
- Position of one or multiple people/cars/airplanes over time.
  - Predict congestion along a route suggested by the GPS.
- Speech recognition.
- Music recognition.
  - Sing a tune to your phone, have it recognize the song.
  - Recognize the genre of a song based on how it sounds.

Time Series Example: Signs

- 0.5 to 2 million users of American Sign Language (ASL) in the US.
- Different regions in the world use different sign languages.
  - For example, British Sign Language (BSL) is different than American Sign Language.
- These languages have vocabularies of thousands of signs.
- We will use sign recognition as our example application, as we introduce methods for time series classification.

Example: The ASL Sign for "again"

Example: The ASL Sign for "bicycle"

Representing Signs as Time Series

- A sign is a video (or part of a video).
- A video is a sequence of frames.
  - A sequence of images, which, when displayed rapidly in succession, create the illusion of motion.
- At every frame i, we extract a **feature vector** $\boldsymbol{x}_i$.
  - How should we define the feature vector? This is a (challenging) computer vision problem.
  - The methods we will discuss work with **any** choice we make for the feature vector.
- Then, the entire sign is represented as time series $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_D)$.
  - D is the length of the sign video, measured in frames.

Feature Vectors

- Finding good feature vectors for signs is an active research problem in computer vision.

Feature Vectors

- We choose a simple approach: the feature vector at each frame is the (x, y) pixel location of the right hand at that frame.

Frame 1:
Feature vector
$\boldsymbol{x}_1 = (192,205)$

Feature Vectors

- We choose a simple approach: the feature vector at each frame is the (x, y) pixel location of the right hand at that frame.

Frame 2:
Feature vector
$\boldsymbol{x}_2 = (190{,}201)$

Feature Vectors

- We choose a simple approach: the feature vector at each frame is the (x, y) pixel location of the right hand at that frame.

Frame 3:
Feature vector
$\boldsymbol{x}_3 = (189,197)$

Time Series for a Sign

- Using the previous representation, for sign "AGAIN" we end up with this time series:

((192, 205),(190, 201),(190, 194),(191, 188),(194, 182),(205, 183),
(211, 178),(217, 171),(225, 168),(232, 167),(233, 167),(238, 168),
(241, 169),(243, 176),(254, 177),(256, 179),(258, 186),(269, 194),
(271, 202),(274, 206),(276, 207),(277, 208)).

- It is a sequence of 22 2D vectors.

Time Series for a Sign

- Using the previous representation, for sign "AGAIN" we end up with this time series:

((192, 205),(190, 201),(190, 194),(191, 188),(194, 182),(205, 183),
(211, 178),(217, 171),(225, 168),(232, 167),(233, 167),(238, 168),
(241, 169),(243, 176),(254, 177),(256, 179),(258, 186),(269, 194),
(271, 202),(274, 206),(276, 207),(277, 208))

- It is a sequence of 22 2D vectors.

Time Series Classification

Training sign for "AGAIN".

Time Series Classification

Training sign for "BICYCLE".

Time Series Classification

Suppose our training set only contains those two signs: "AGAIN" and "BICYCLE".

We get this test sign.

Does it mean "AGAIN", or "BICYCLE"?

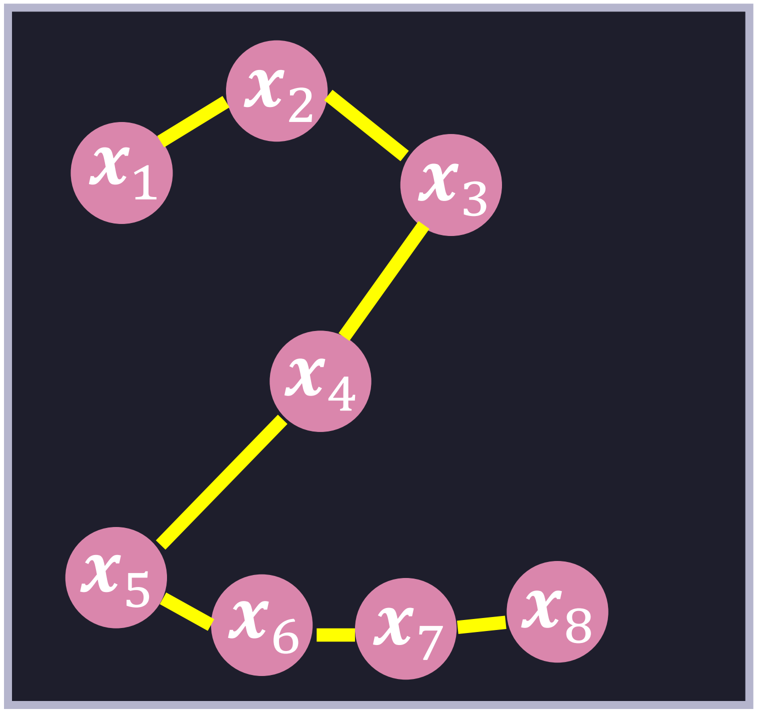
Time Series Classification

Does this test sign mean "AGAIN", or "BICYCLE"?

We can use nearest-neighbor classification.

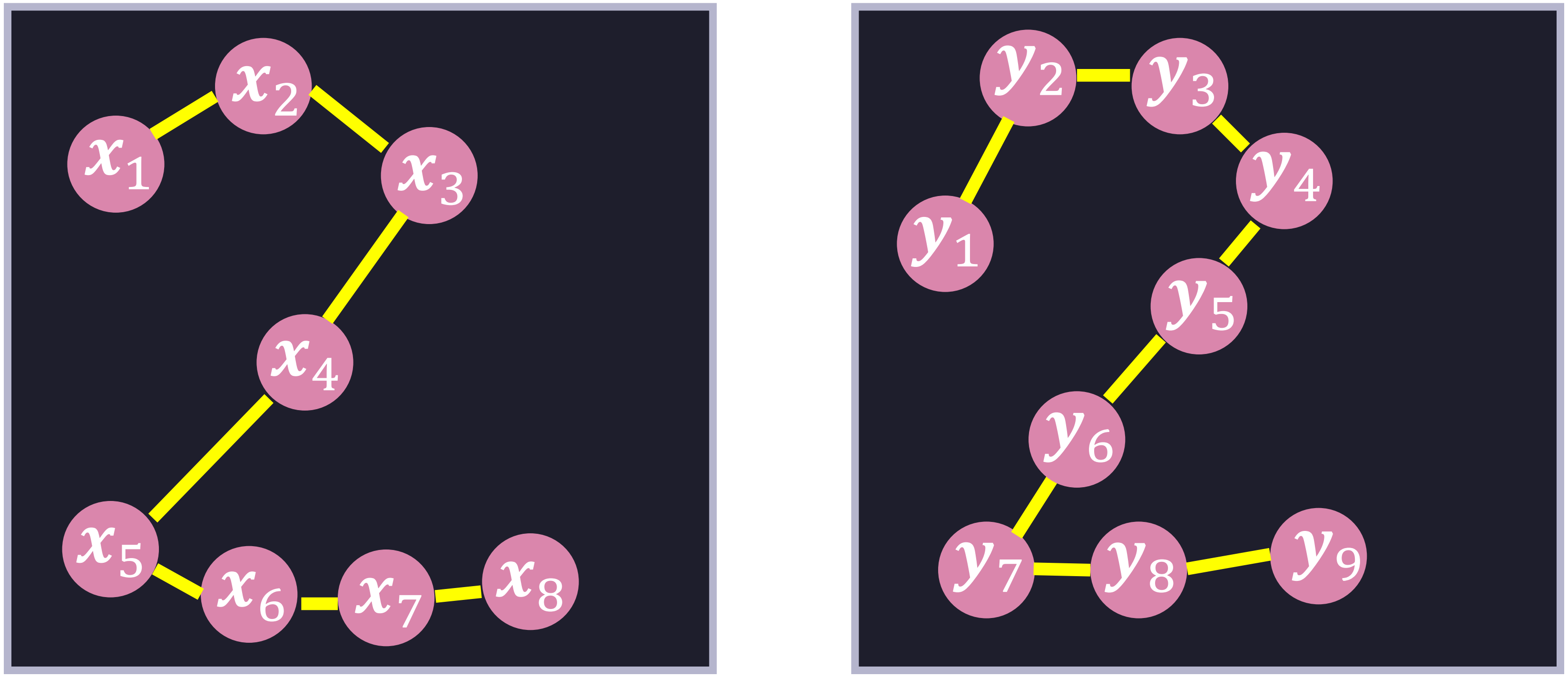
Big question: what distance measure should we choose?

Visualizing a Time Series



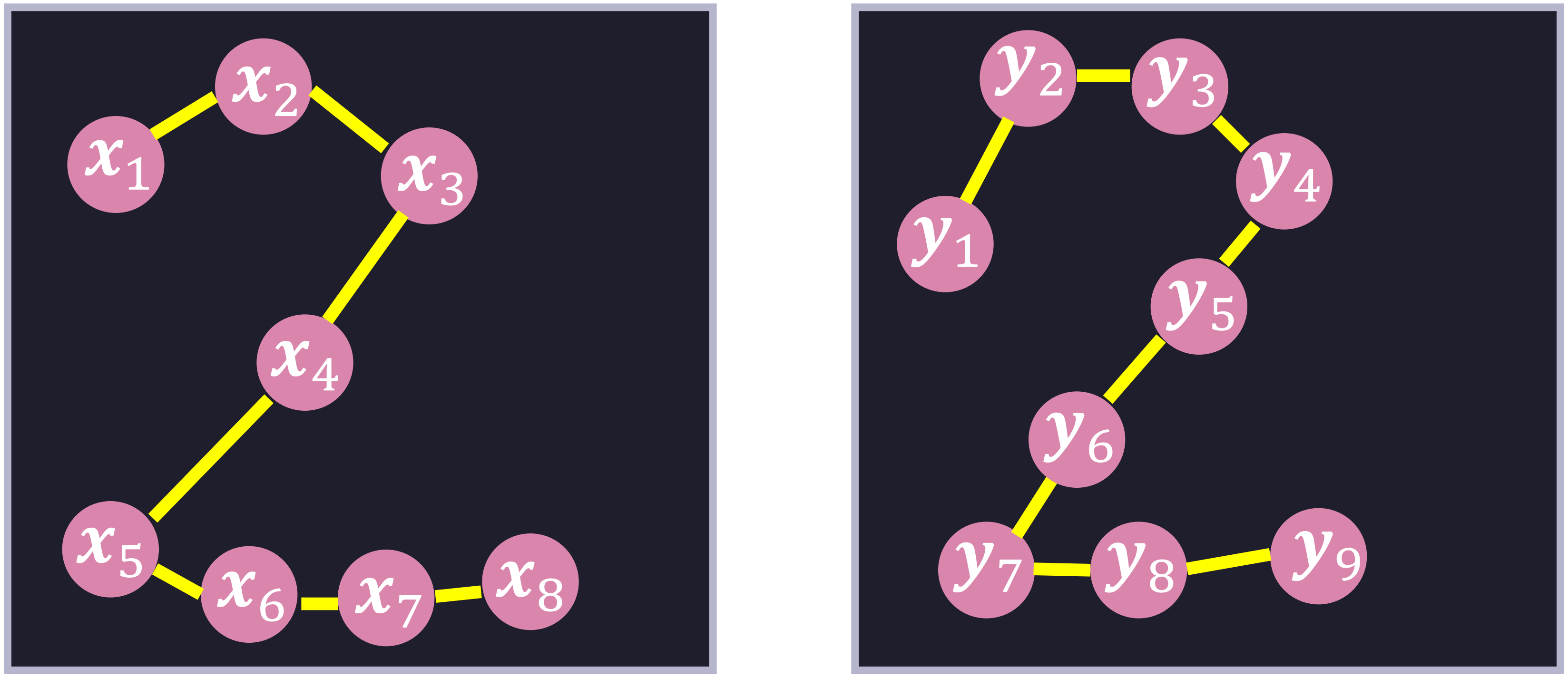
- Here is a visualization of one time series $\boldsymbol{X}$.
- Each feature vector (as before) is a point in 2D.
- The time series has eight elements. $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_D)$.
- For example, x_4 indicates the position of 2D point $\boldsymbol{x}_4$.

Comparing Time Series



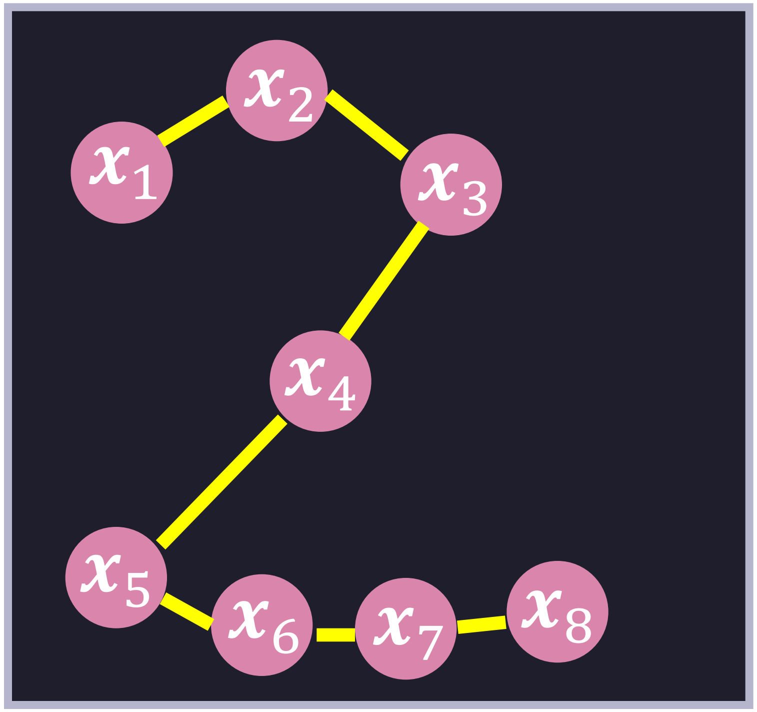
- Here is a visualization of two time series, $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- How do we measure the distance between two time series?

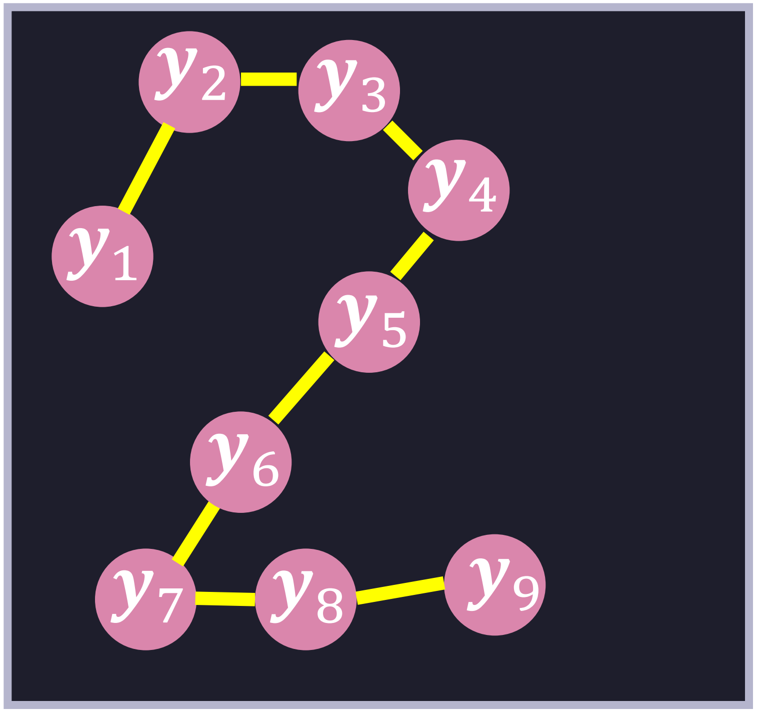
Comparing Time Series



- We could possibly use the Euclidean distance.
  - The first time series is a 16-dimensional vector.
  - The second time series is an 18-dimensional vector.
- However, there are issues…

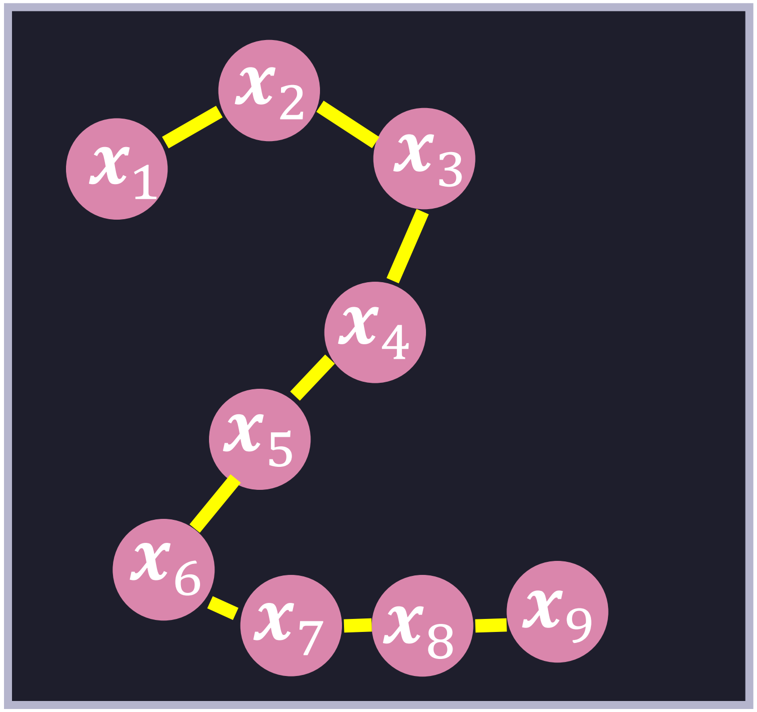
Problems with the Euclidean Distance

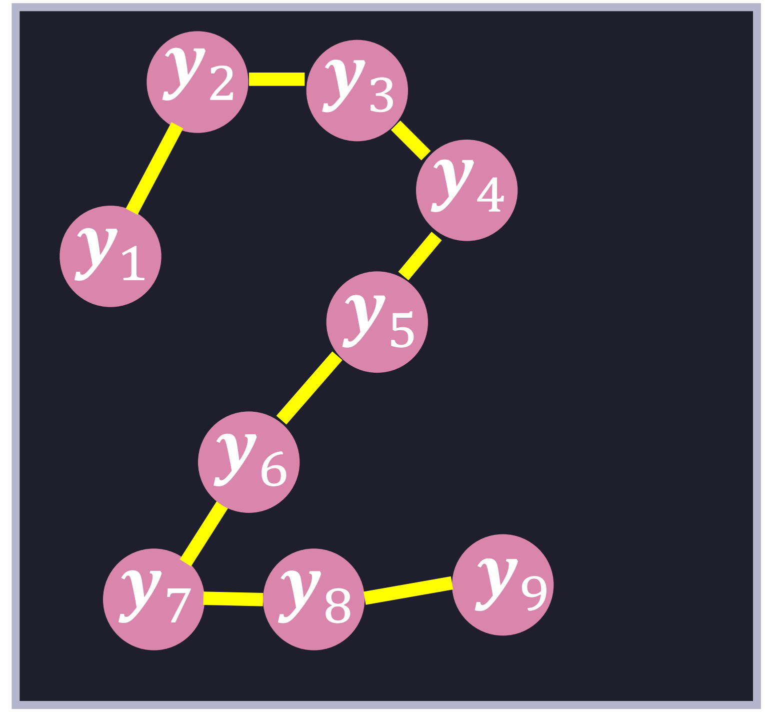




- The two vectors have different numbers of dimensions.
- We could fix that by "stretching" the first time series to also have 9 elements.

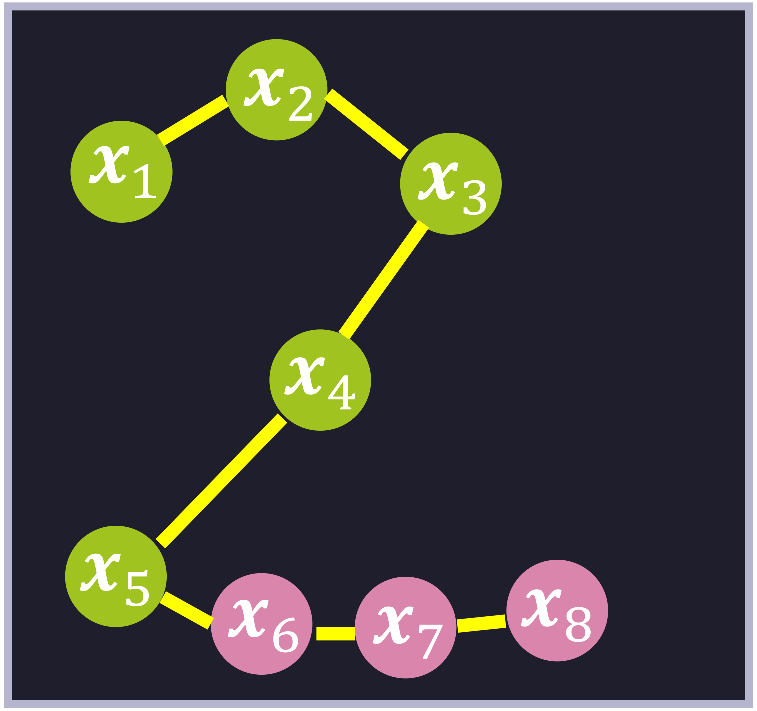
Problems with the Euclidean Distance

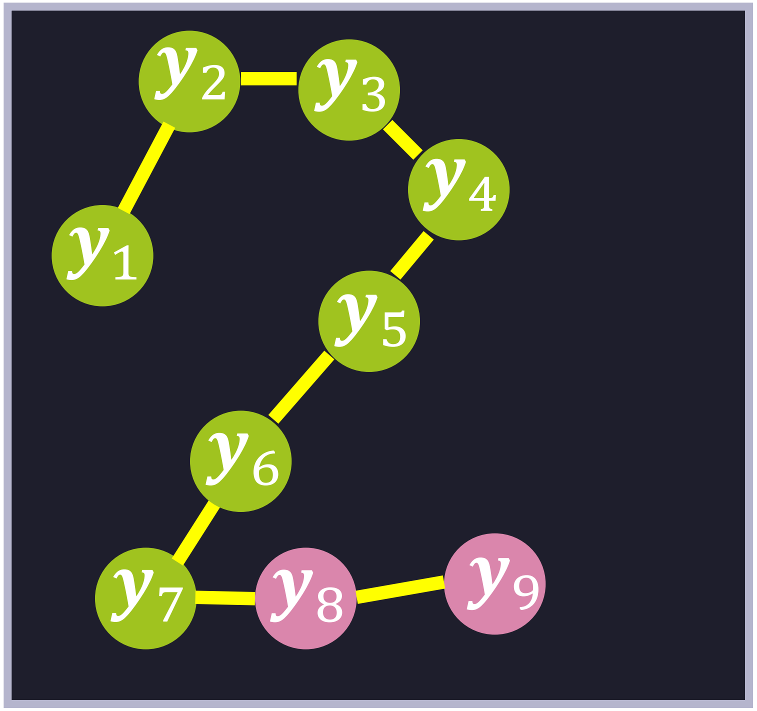




- Here, the time series on the left has been converted to have nine elements, using interpolation.
  - We will not explore that option any further, because…

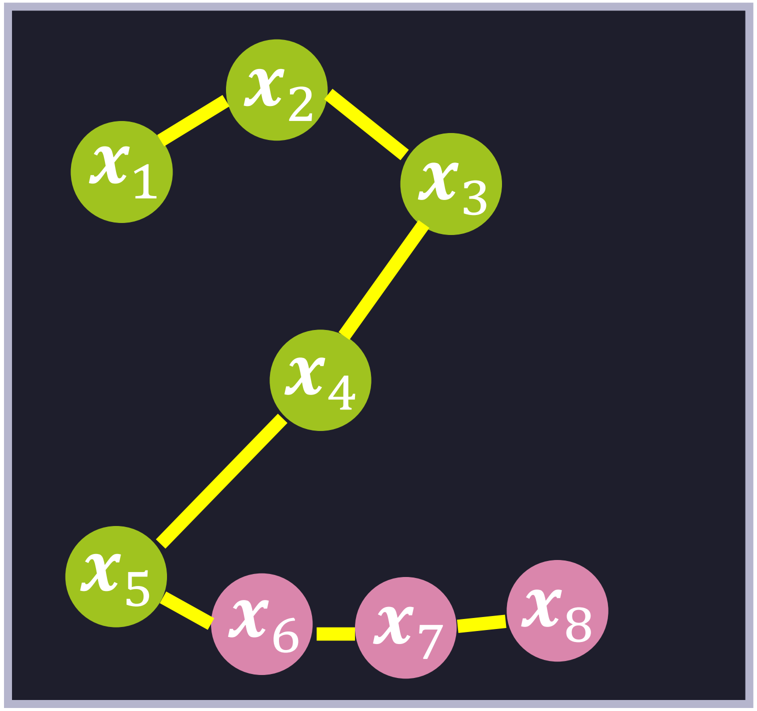
Problems with the Euclidean Distance

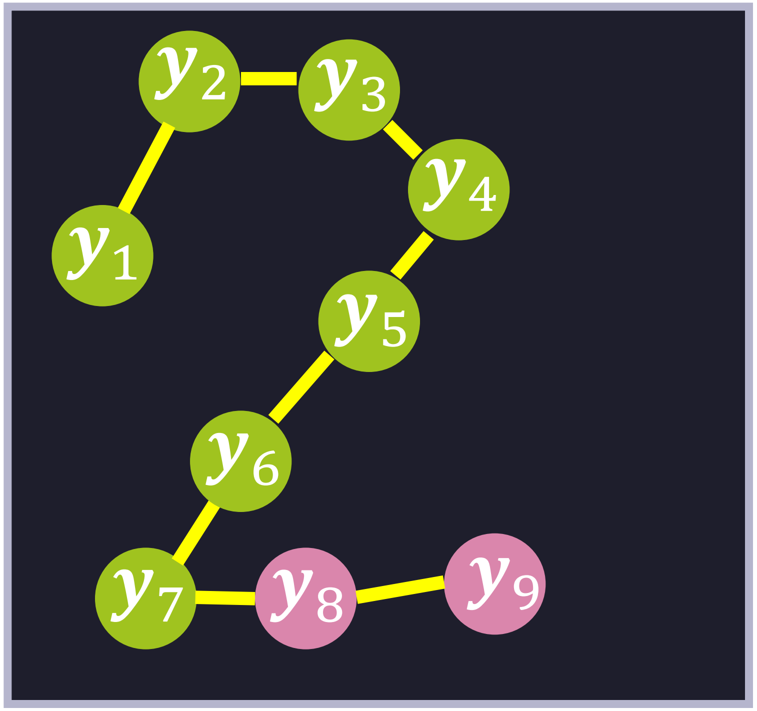




- Bigger problem: the two time series correspond to a similar pattern being performed with **variable speed**.
  - The first five elements of the first time series visually match the first seven elements of the second time series.

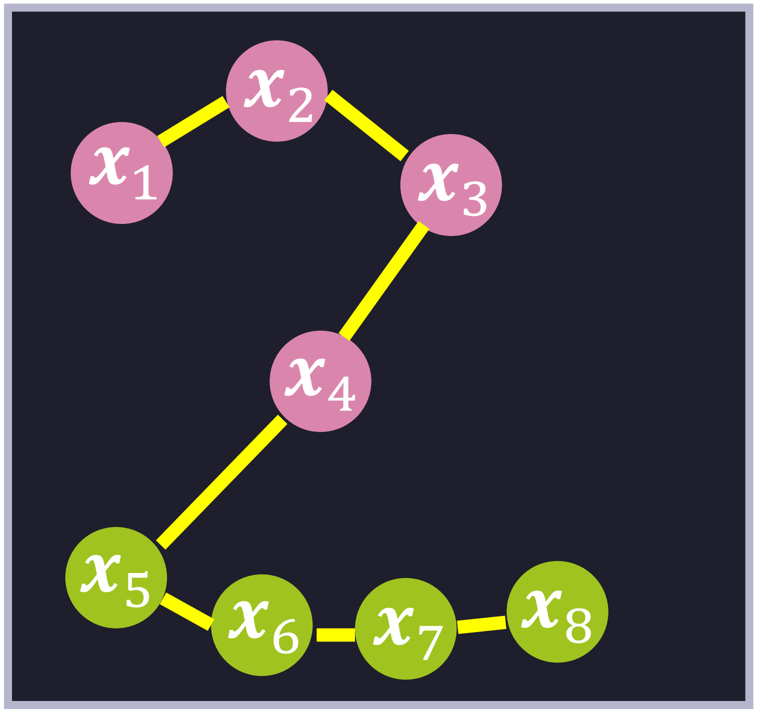
Problems with the Euclidean Distance

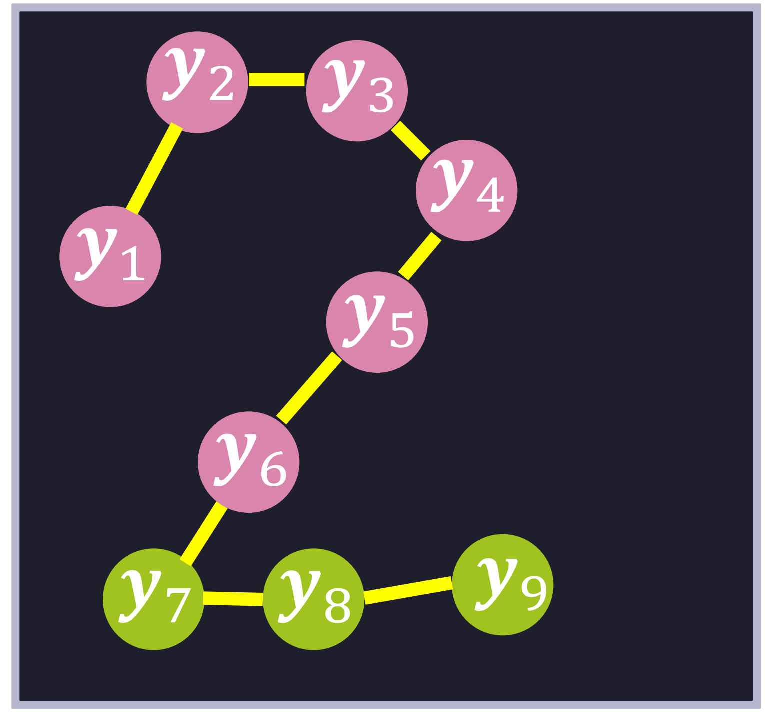




- So, the first time series had a faster speed than the second time series initially.
  - It took 5 time ticks for the first time series, and seven time ticks for the second time series, to move through approximately the same trajectory.

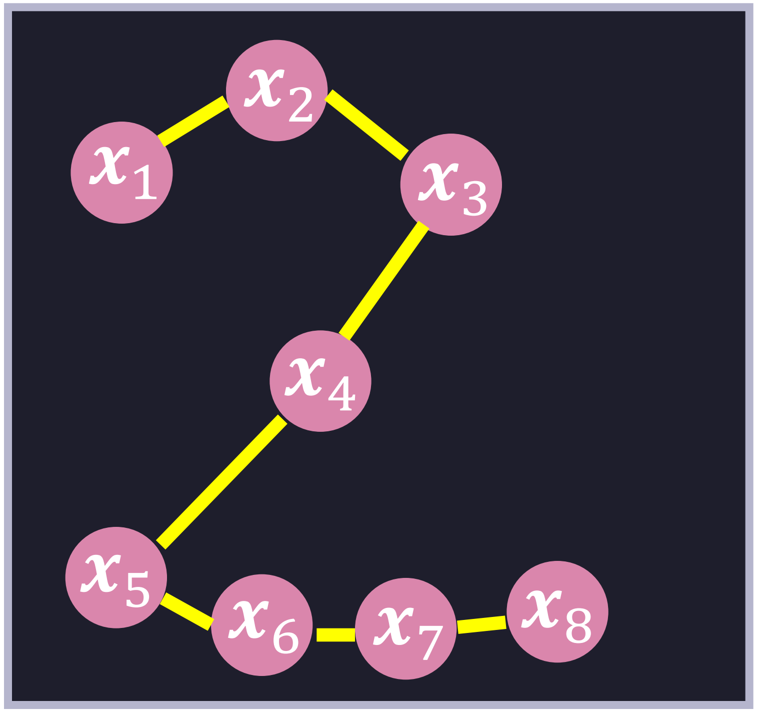
Problems with the Euclidean Distance

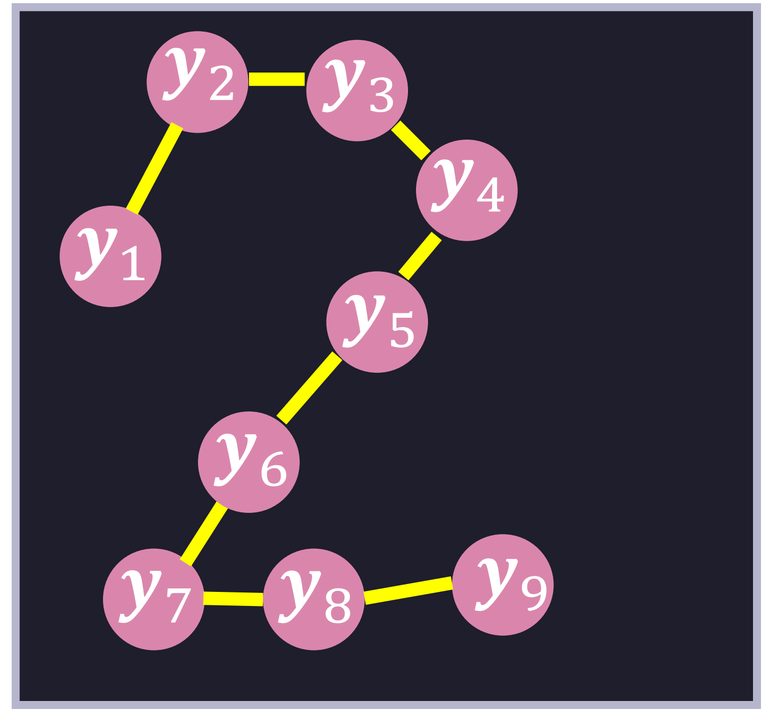




- Similarly, the last four elements of the first time series visually match the last three elements of the second time series.
- So, the first time series had a faster speed than the second time series in the last part.
  - 4 time ticks for the first time series, 3 time ticks for the second one.

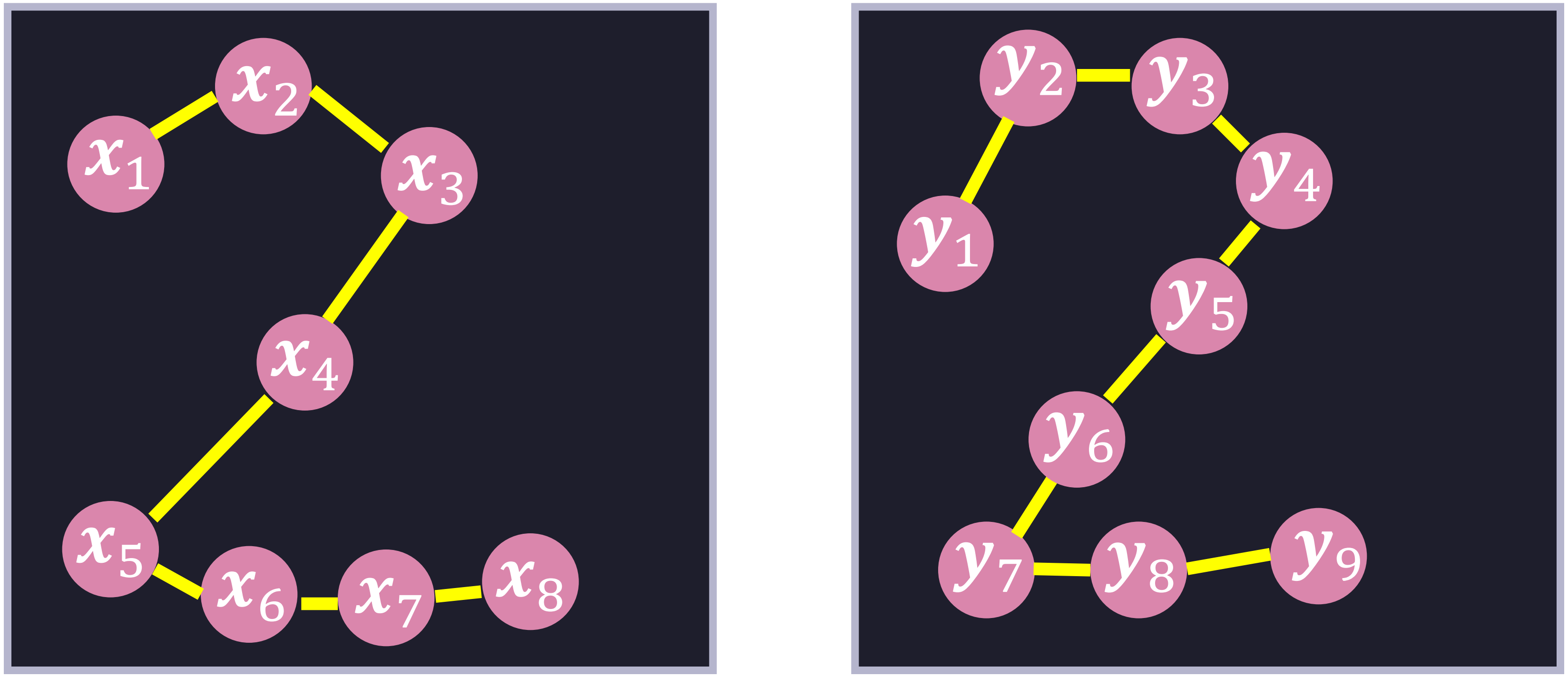
Time Series Alignment





- An alignment is a correspondence between elements of two time series.
- To reliably measure the similarity between two time series, we need to figure out, for each element of the first time series, which elements of the second time series it **<u>corresponds</u>** to.

Time Series Alignment

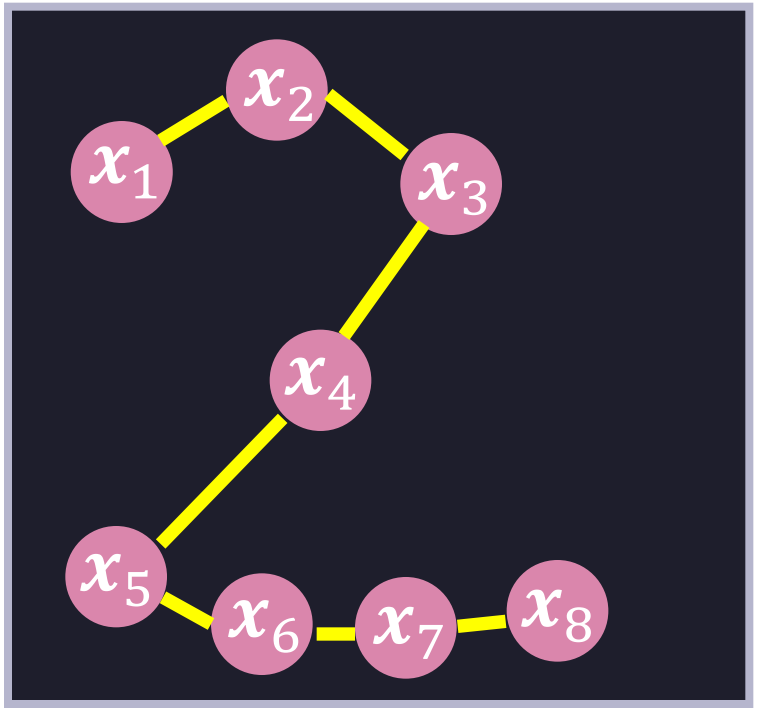


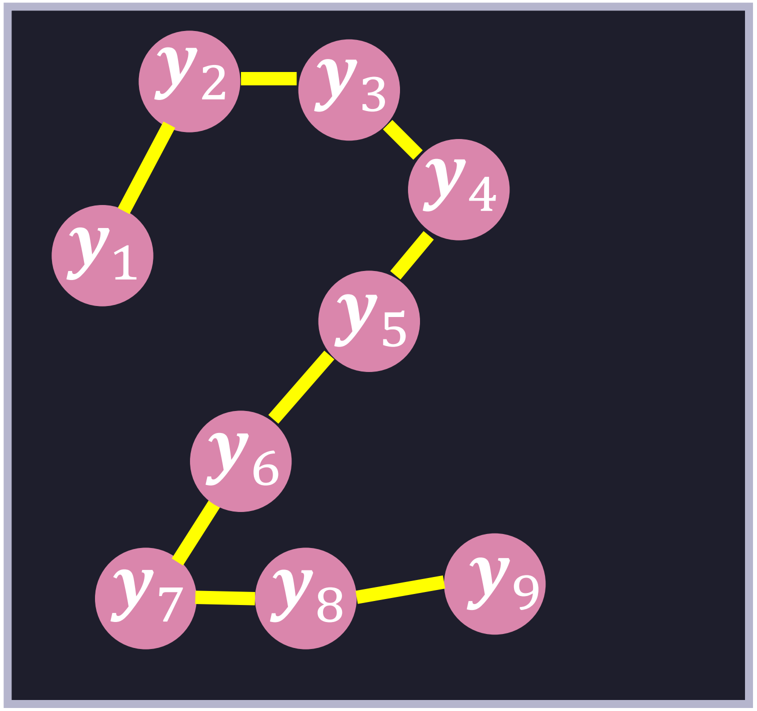
- An alignment $\boldsymbol{A}$ between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is a sequence of pairs of indices:

$$\boldsymbol{A} = \Big((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \dots, (a_{R,1}, a_{R,2})\Big)$$

- Element $(a_{i,1}, a_{i,2})$ of alignment $\boldsymbol{A}$ specifies that element $\boldsymbol{x}_{a_{i,1}}$ of $\boldsymbol{X}$ corresponds to element $\boldsymbol{y}_{a_{i,2}}$ of the other time series.

Time Series Alignment



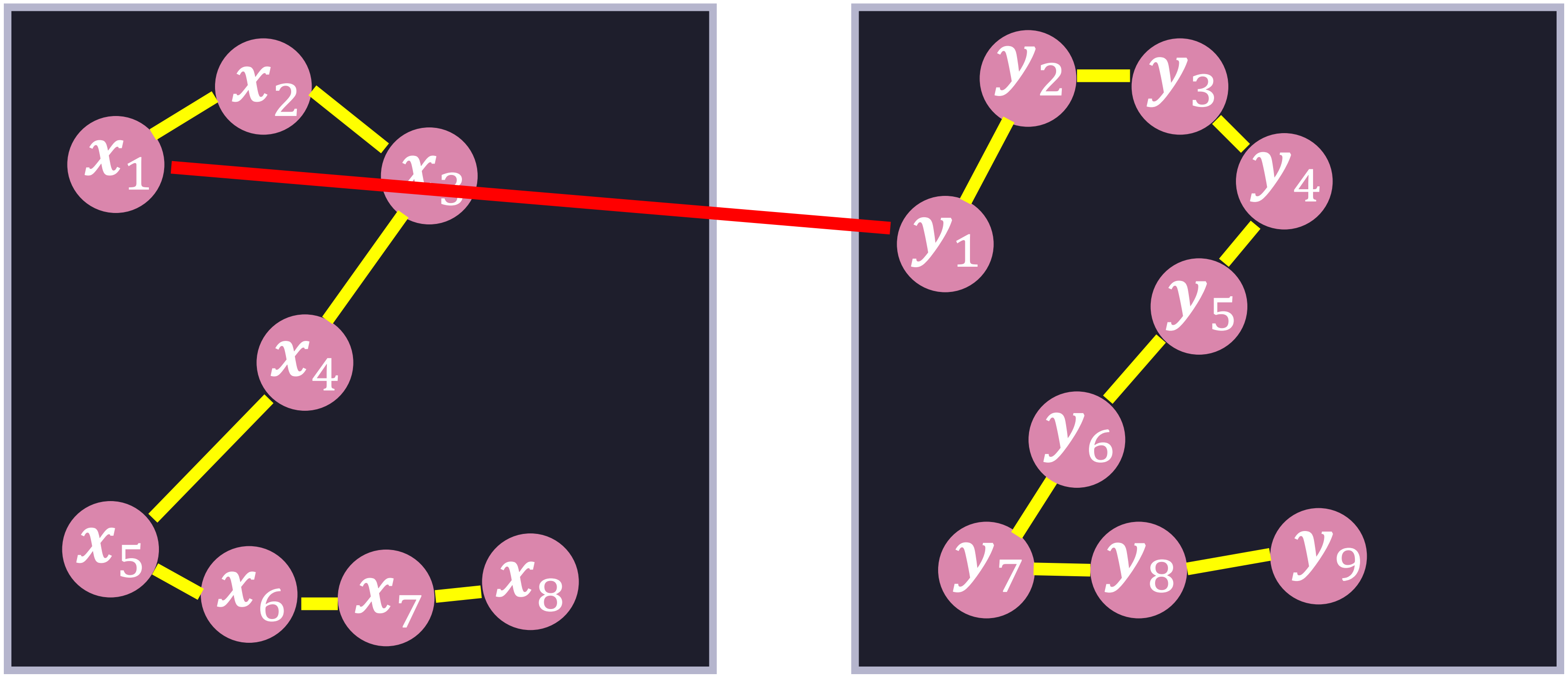


- For example, here is a "good" alignment between the two series:

$$\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$$

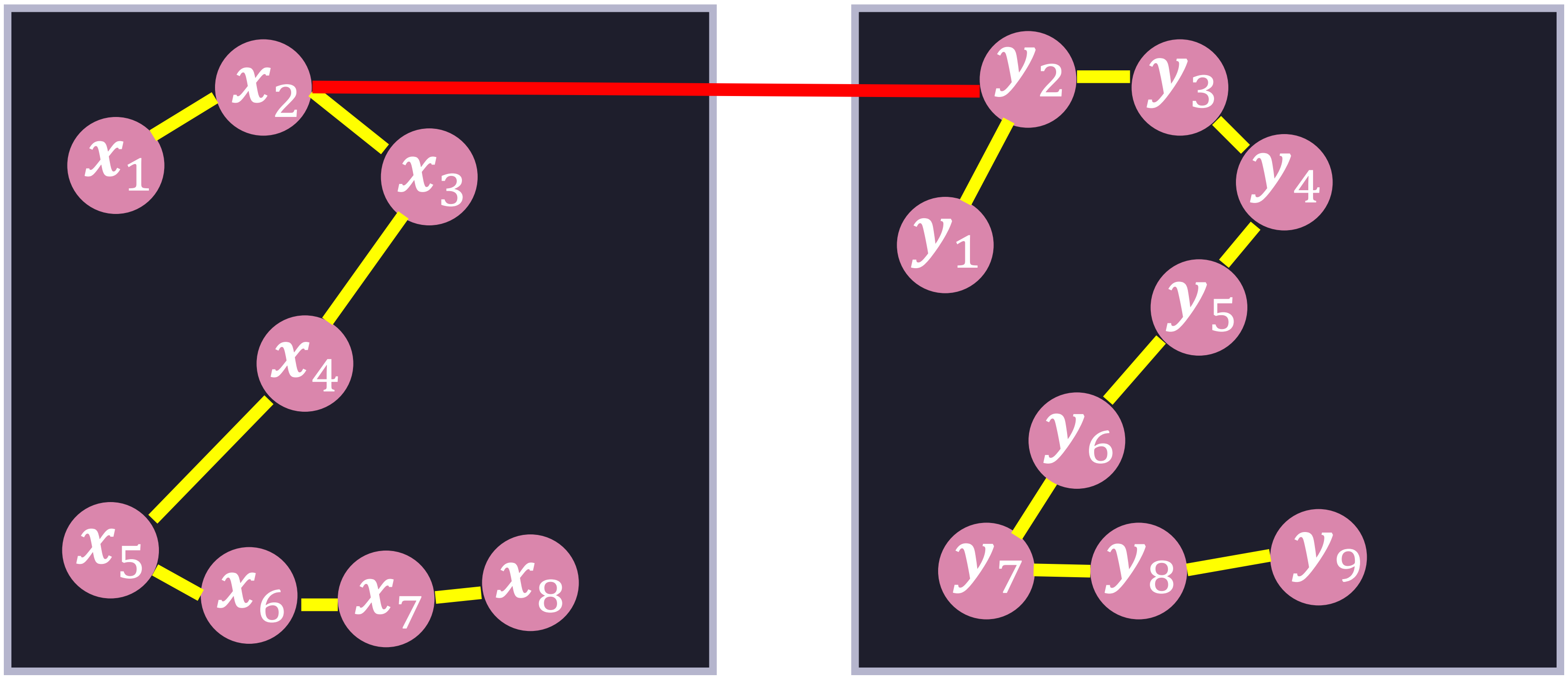
- We will discuss later how to find such an alignment automatically.

Time Series Alignment



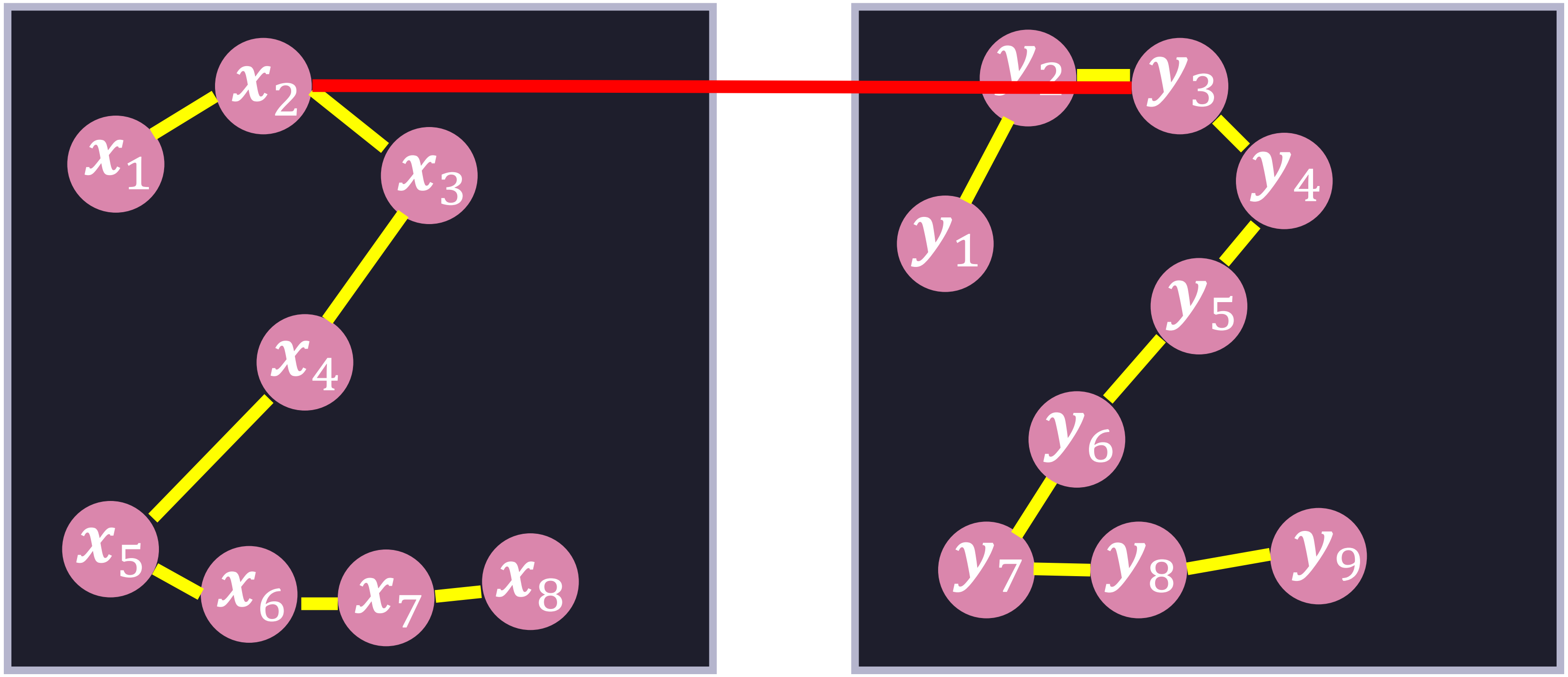
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_1$ corresponds to $\boldsymbol{y}_1$.

Time Series Alignment



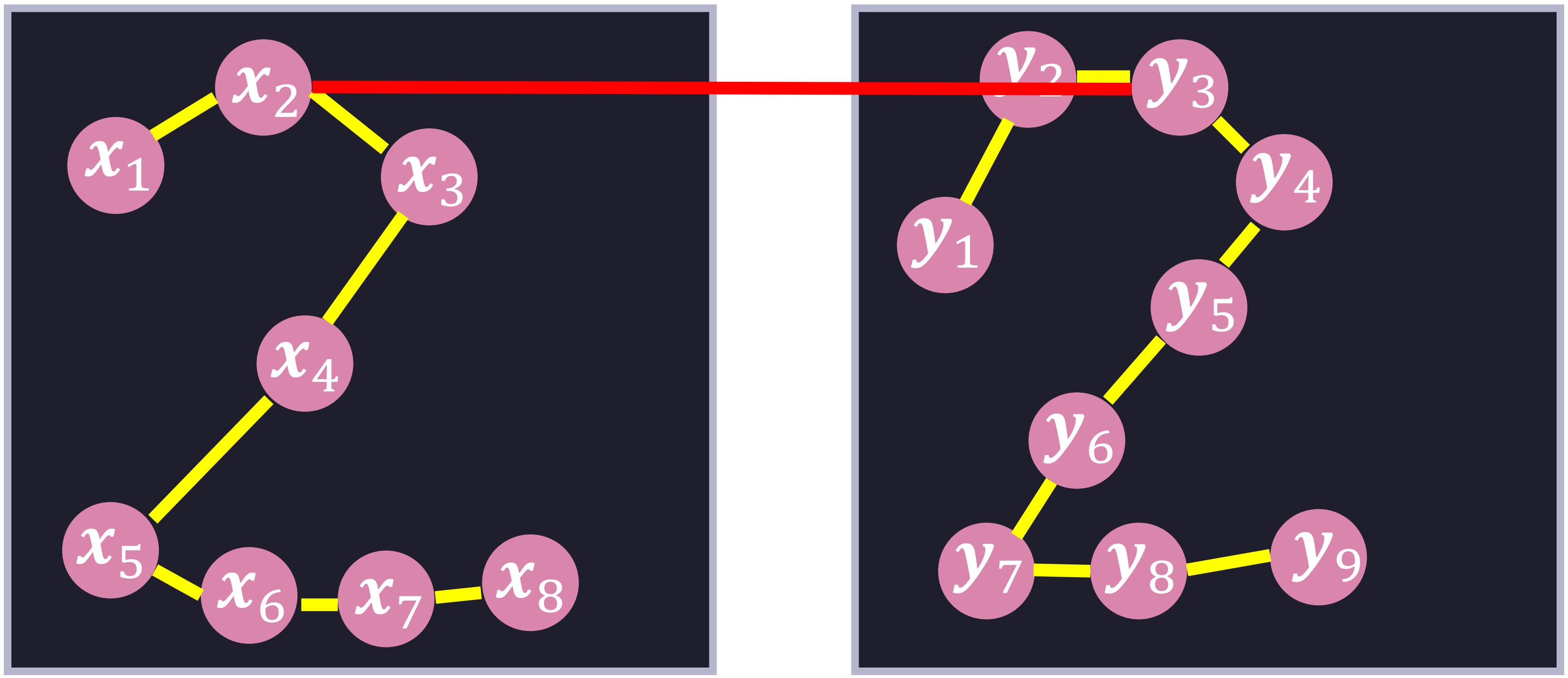
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_2$ corresponds to $\boldsymbol{y}_2$.

Time Series Alignment



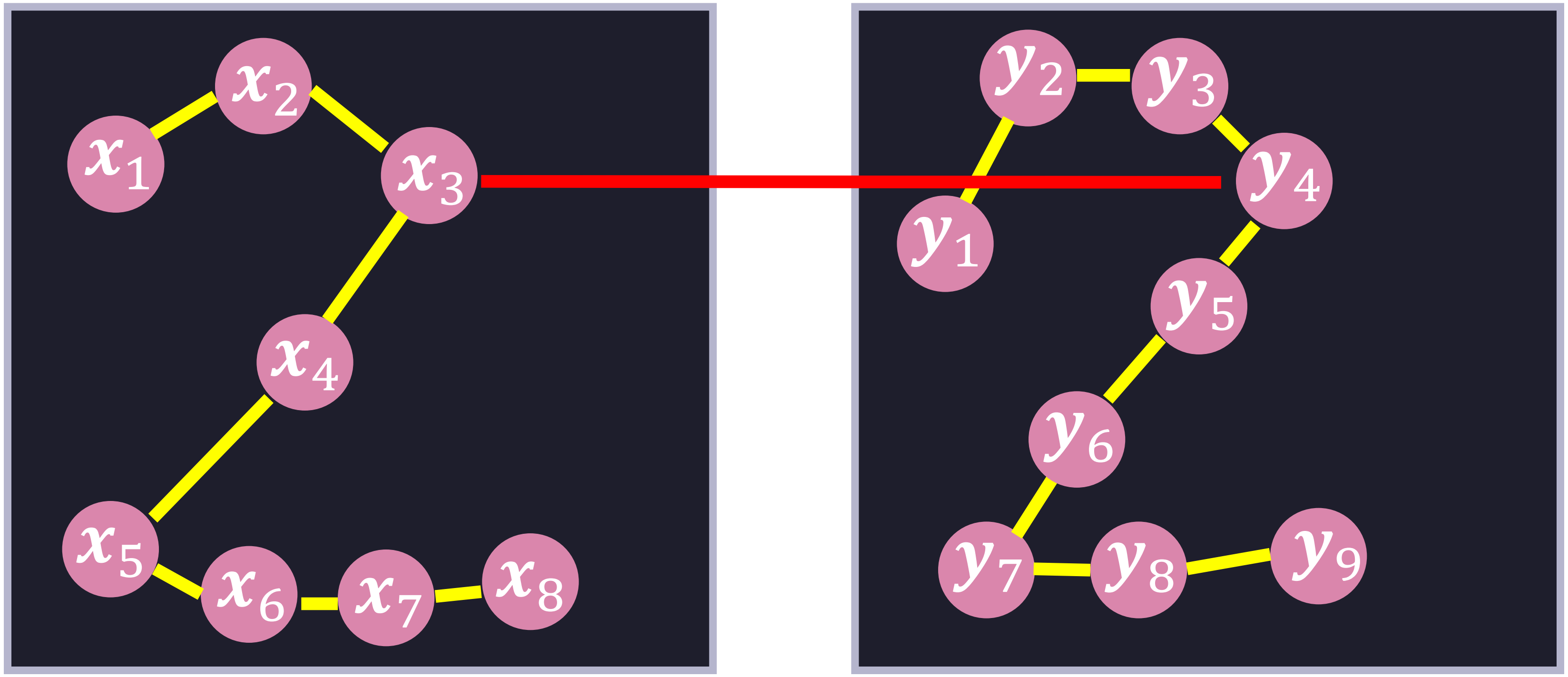
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - x_2 **also corresponds to** y_3.

Time Series Alignment



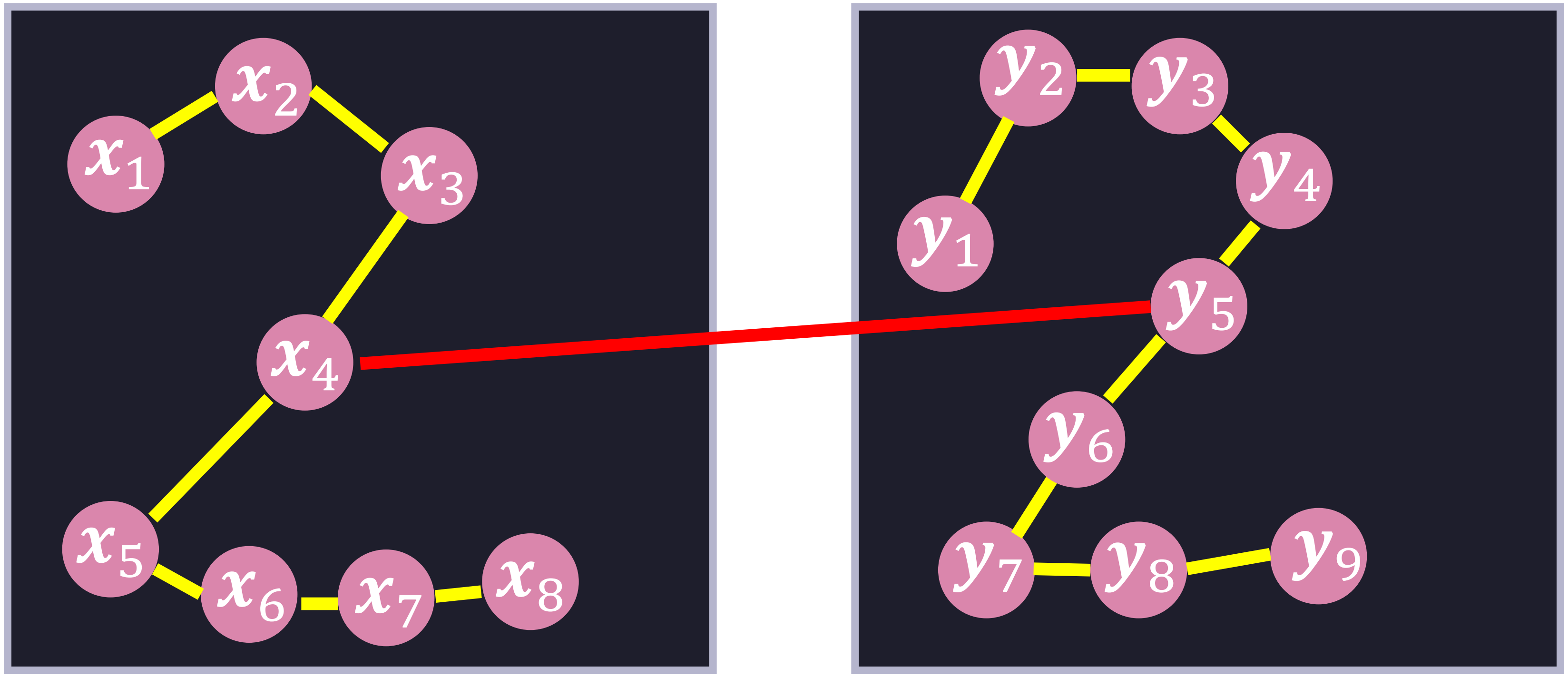
- x_2 **also corresponds to** y_3.
- One element from one time series can correspond to **multiple consecutive elements** from the other time series.
  - This captures cases where one series moves slower than the other.
  - For this segment, the first series moves faster than the second one.

Time Series Alignment



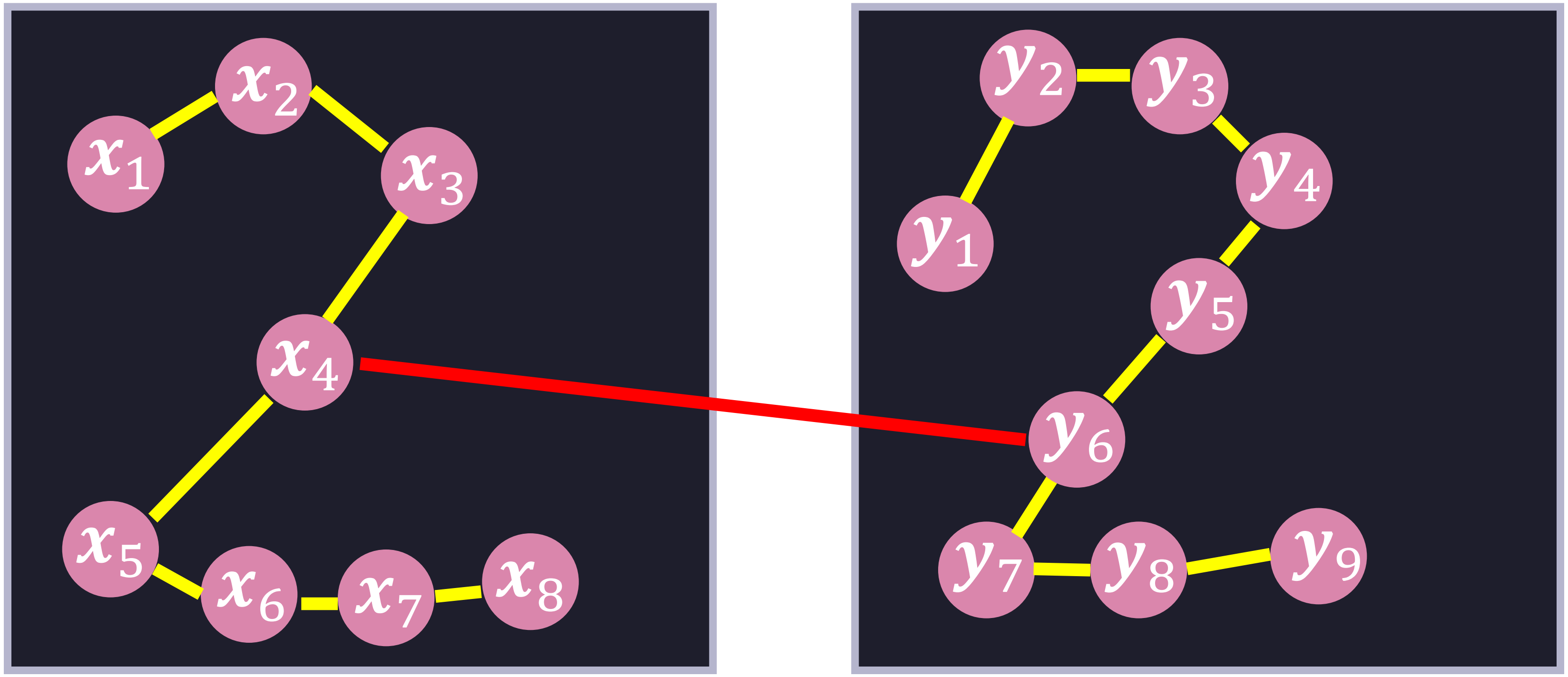
- Alignment: $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_3$ corresponds to $\boldsymbol{y}_4$.

Time Series Alignment



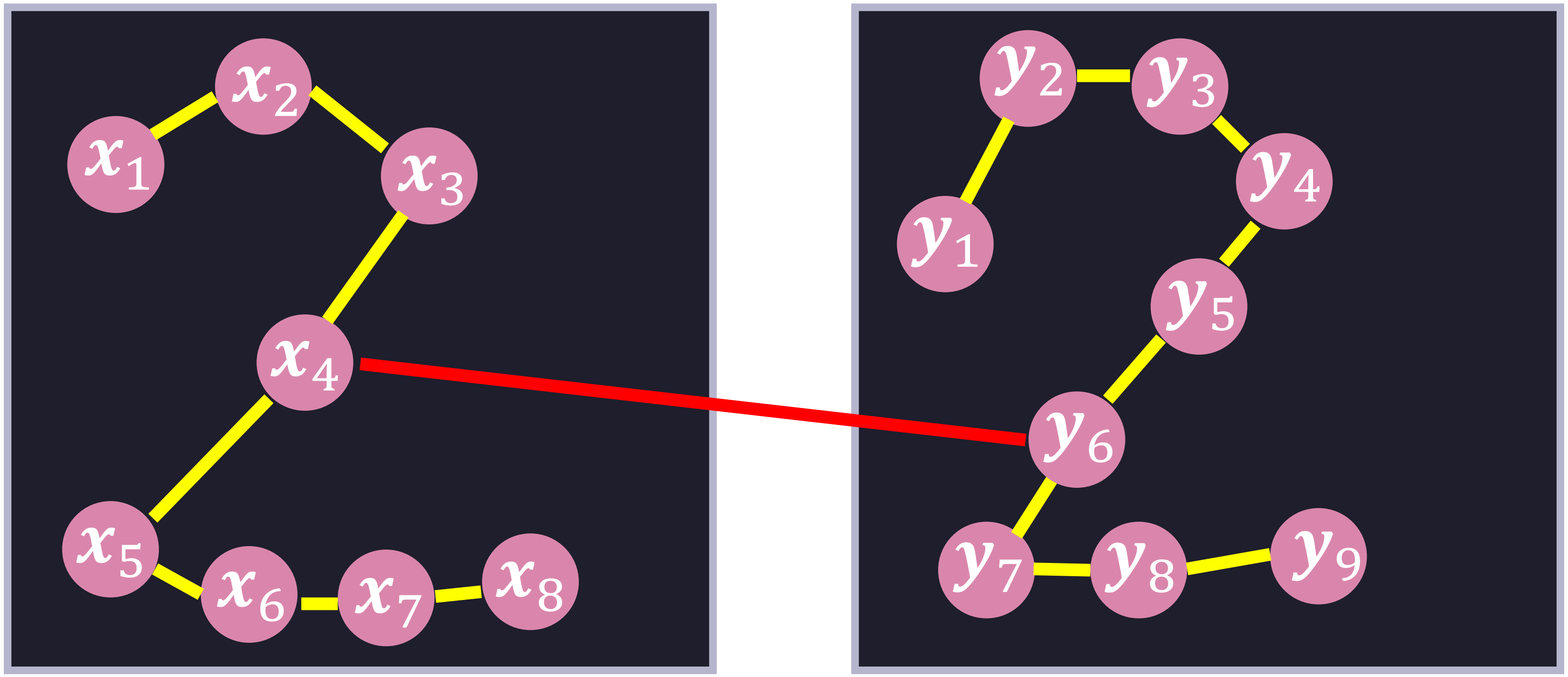
- Alignment: $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_4$ corresponds to $\boldsymbol{y}_5$.

Time Series Alignment



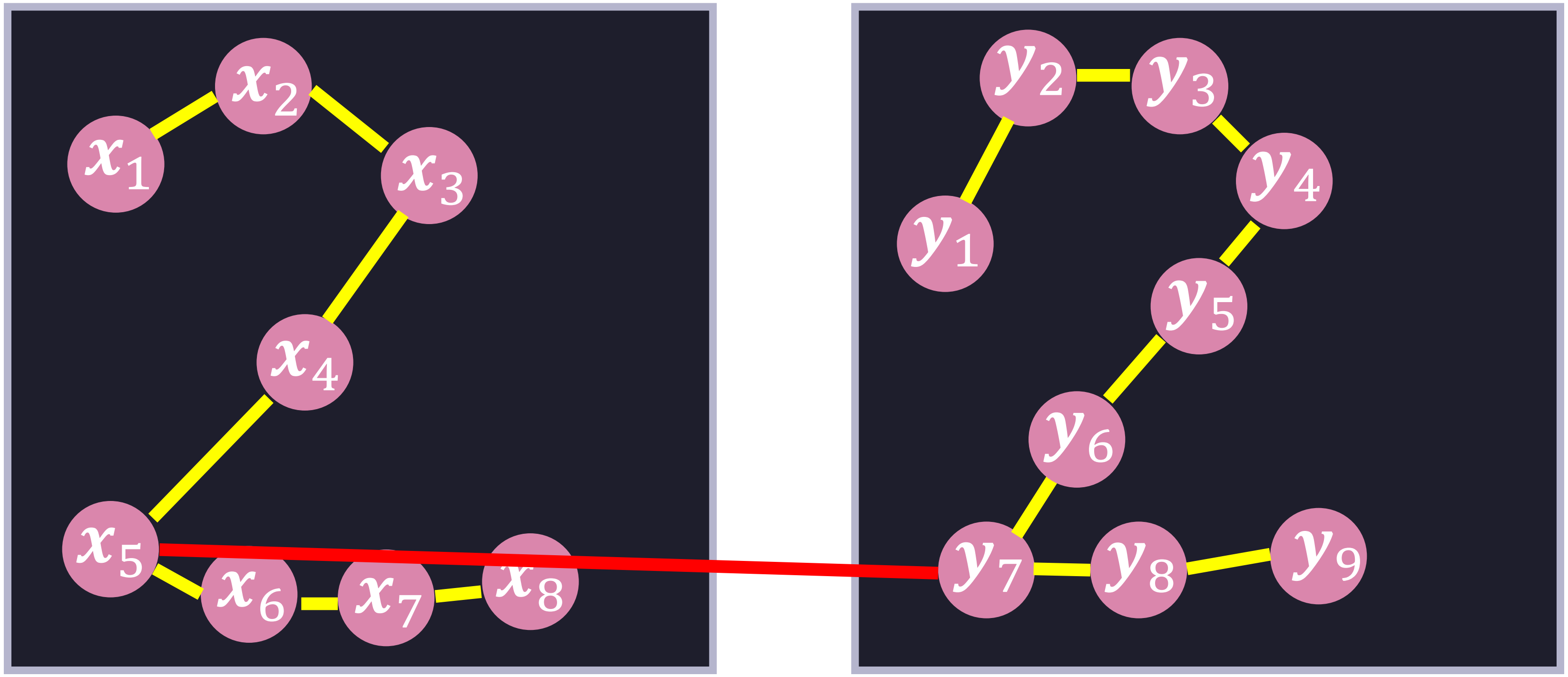
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_4$ **also corresponds to** $\boldsymbol{y}_6$.

Time Series Alignment



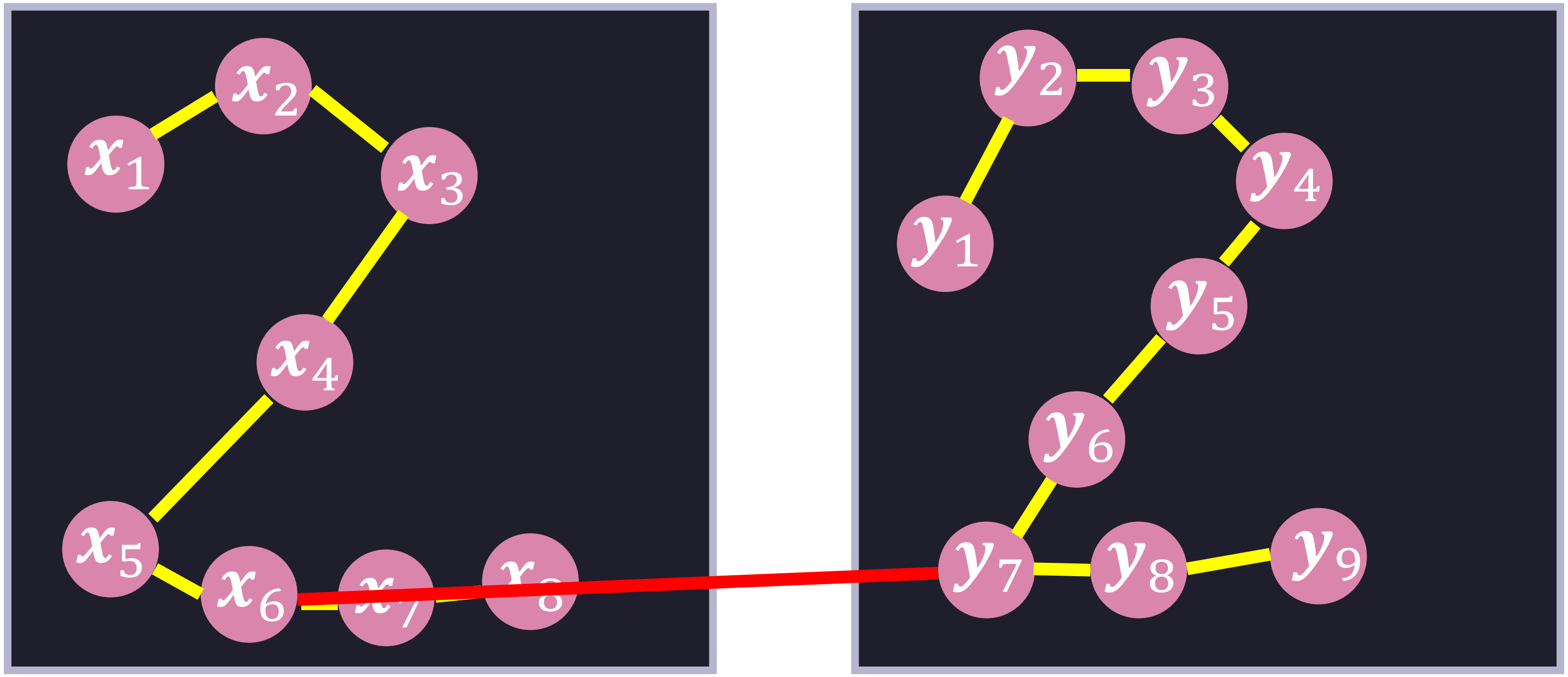
- x_4 **<u>also corresponds to</u>** y_6.
  - As before, for this segment, the first time series is moving faster than the second one.

Time Series Alignment



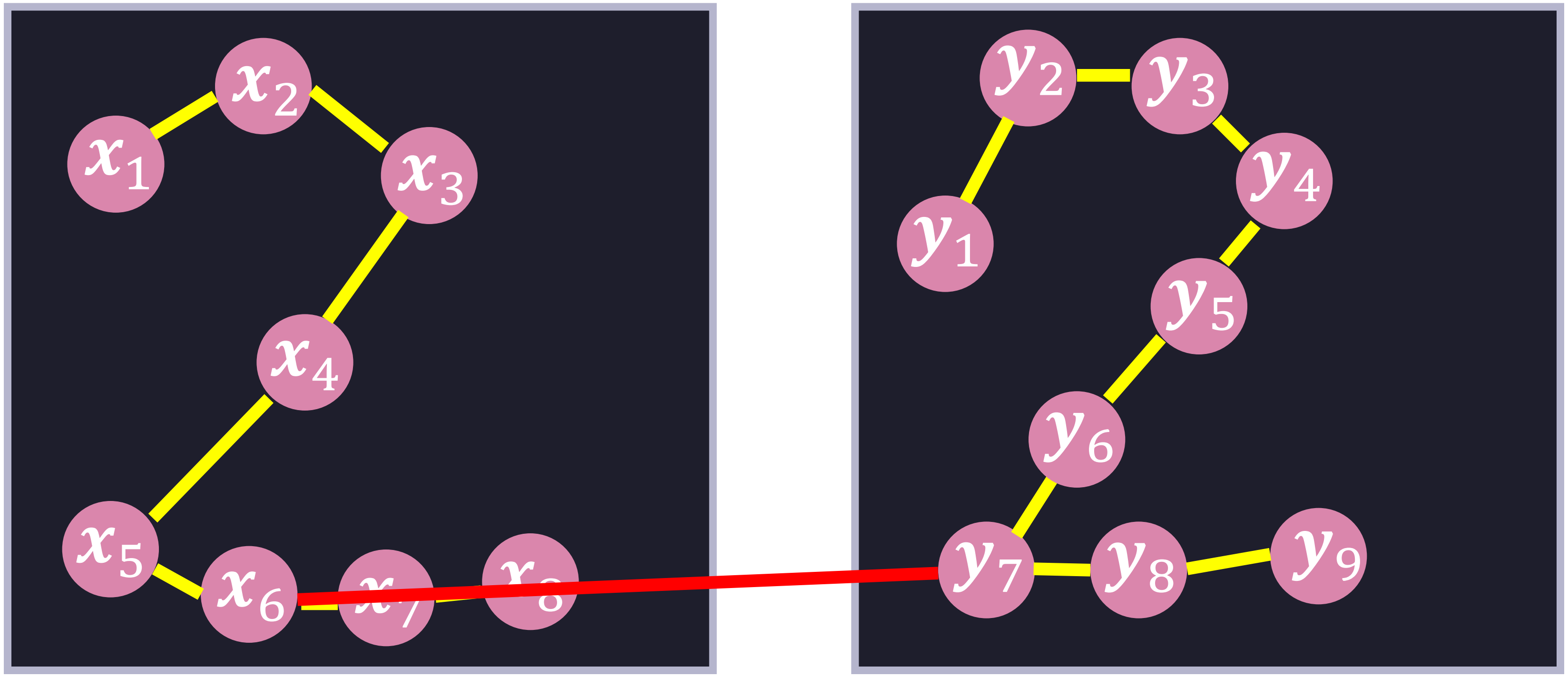
- Alignment:
$\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_5$ corresponds to $\boldsymbol{y}_7$.

Time Series Alignment



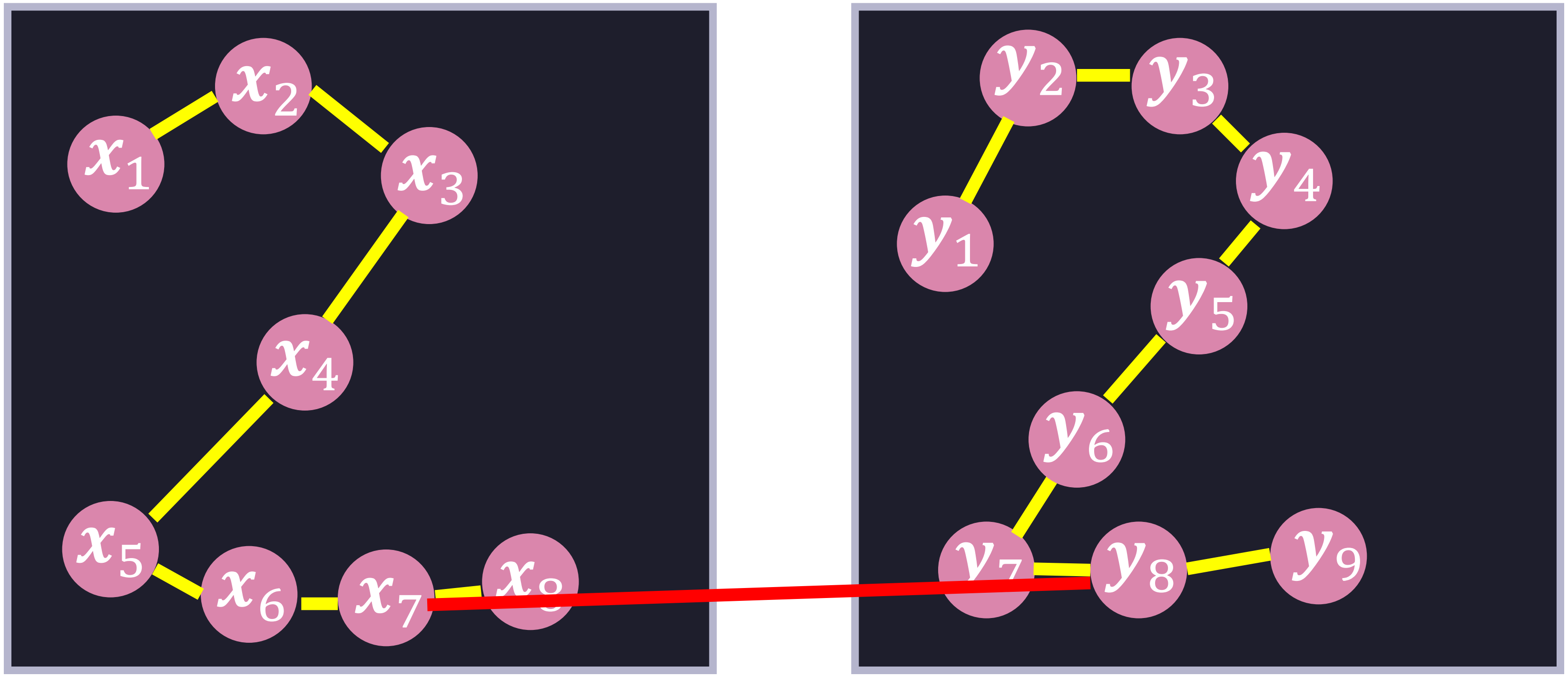
- Alignment: $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - x_6 **also corresponds to** y_5.

Time Series Alignment



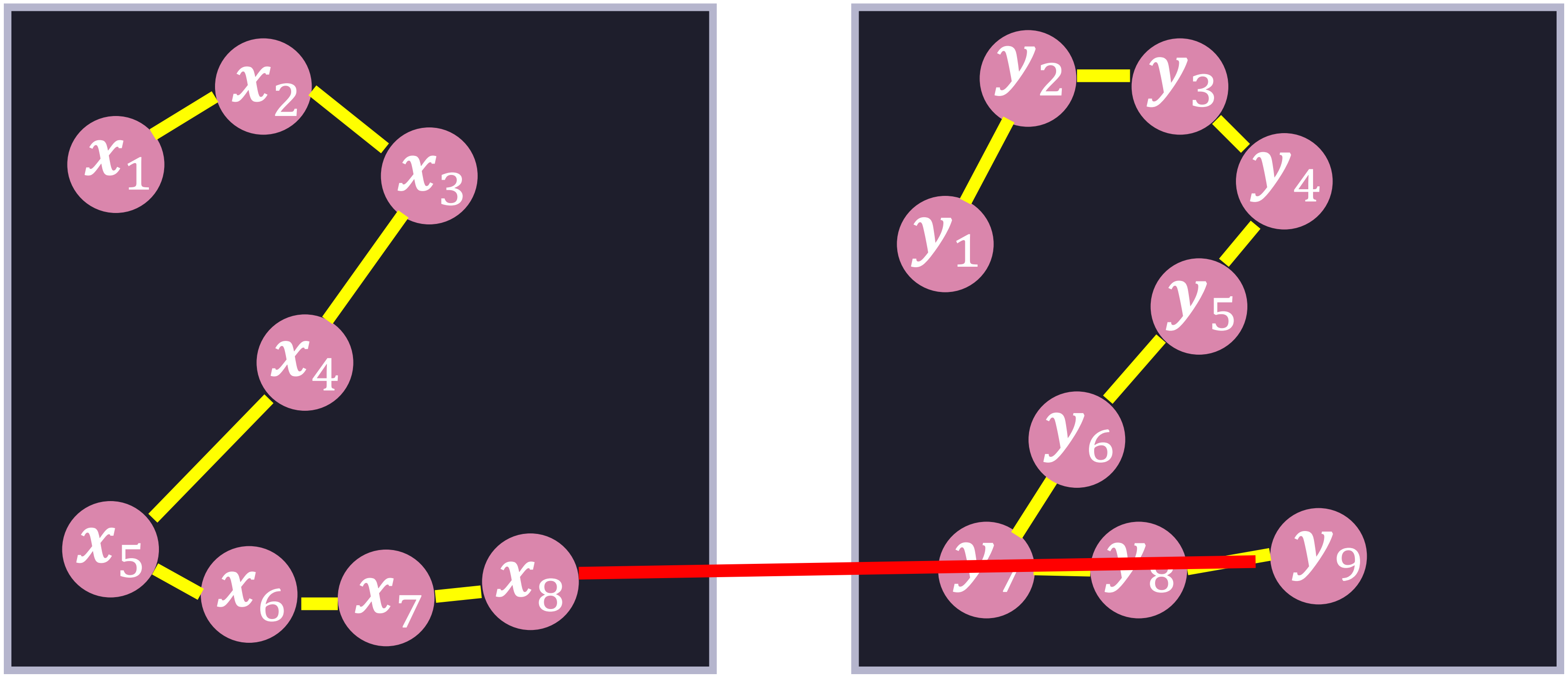
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- Here, the first time series is moving slower than the second one, and thus $\boldsymbol{x}_5$ and $\boldsymbol{x}_6$ are both matched to $\boldsymbol{y}_7$.

Time Series Alignment



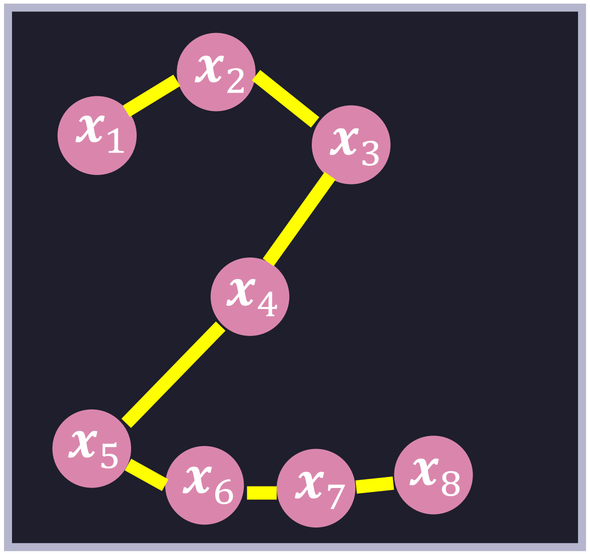
- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_7$ corresponds to $\boldsymbol{y}_8$.

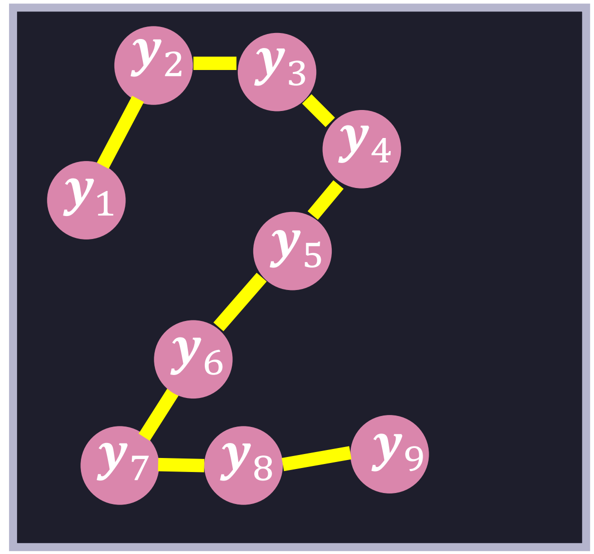
Time Series Alignment



- Alignment:
  $\big((1,1),(2,2),(2,3),(3,4),(4,5),(4,6),(5,7),(6,7),(7,8),(8,9)\big)$
- According to this alignment:
  - $\boldsymbol{x}_8$ corresponds to $\boldsymbol{y}_9$.

The Cost of an Alignment





- An alignment $\boldsymbol{A}$ is defined as a sequence of pairs

$$\boldsymbol{A} = \left((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \dots, (a_{R,1}, a_{R,2})\right)$$

- If we are given an alignment $\boldsymbol{A}$ between two time series $\boldsymbol{X}$ and $\boldsymbol{Y}$, we can compute the cost $C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A})$ of that alignment as:

$$C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A}) = \sum_{i=1}^{R} \mathrm{Cost}(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}})$$

The Cost of an Alignment

- An alignment is defined as a sequence of pairs:

$$\boldsymbol{A} = \big((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \ldots, (a_{R,1}, a_{R,2})\big)$$

- If we are given an alignment $\boldsymbol{A}$ between two time series $\boldsymbol{X}$ and $\boldsymbol{Y}$, we can compute the cost $C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A})$ of that alignment as:

$$C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A}) = \sum_{i=1}^{R} \text{Cost}(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}})$$

- In this formula, $\text{Cost}\big(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}}\big)$ is a black box.
  - You can define it any way you like.
  - For example, if $\boldsymbol{x}_{a_{i,1}}$ and $\boldsymbol{y}_{a_{i,2}}$ are vectors, $\text{Cost}(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}})$ can be the Euclidean distance between $\boldsymbol{x}_{a_{i,1}}$ and $\boldsymbol{y}_{a_{i,2}}$.

The Cost of an Alignment

- An alignment is defined as a sequence of pairs:

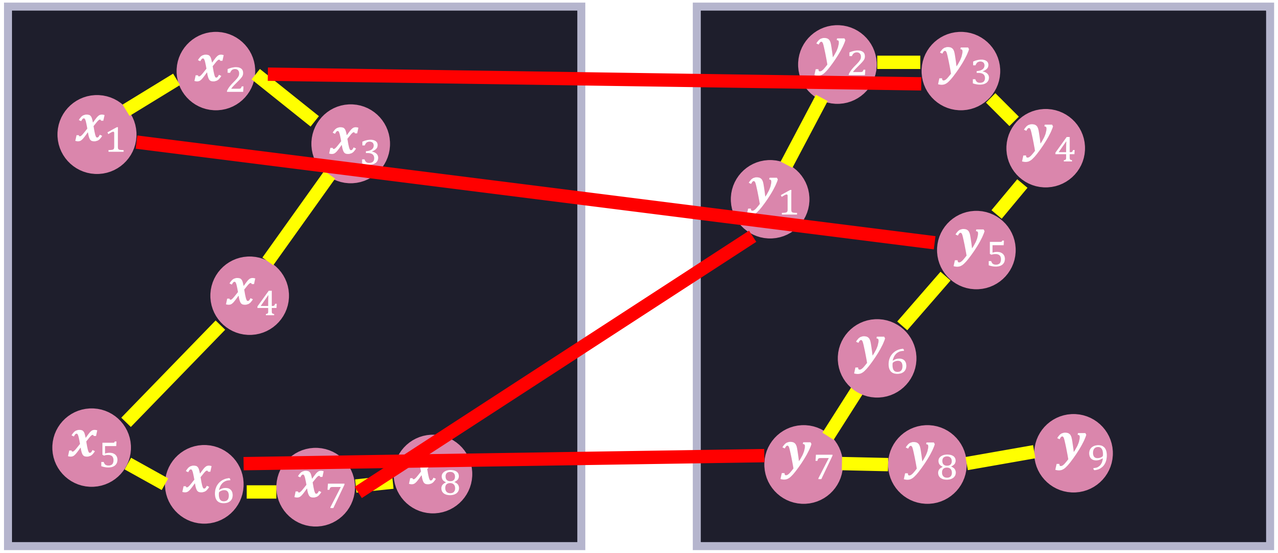
$$\boldsymbol{A} = \Big((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \ldots, (a_{R,1}, a_{R,2})\Big)$$

- If we are given an alignment $\boldsymbol{A}$ between two time series $\boldsymbol{X}$ and $\boldsymbol{Y}$, we can compute the cost $C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A})$ of that alignment as:

$$C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A}) = \sum_{i=1}^{R} \mathrm{Cost}(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}})$$

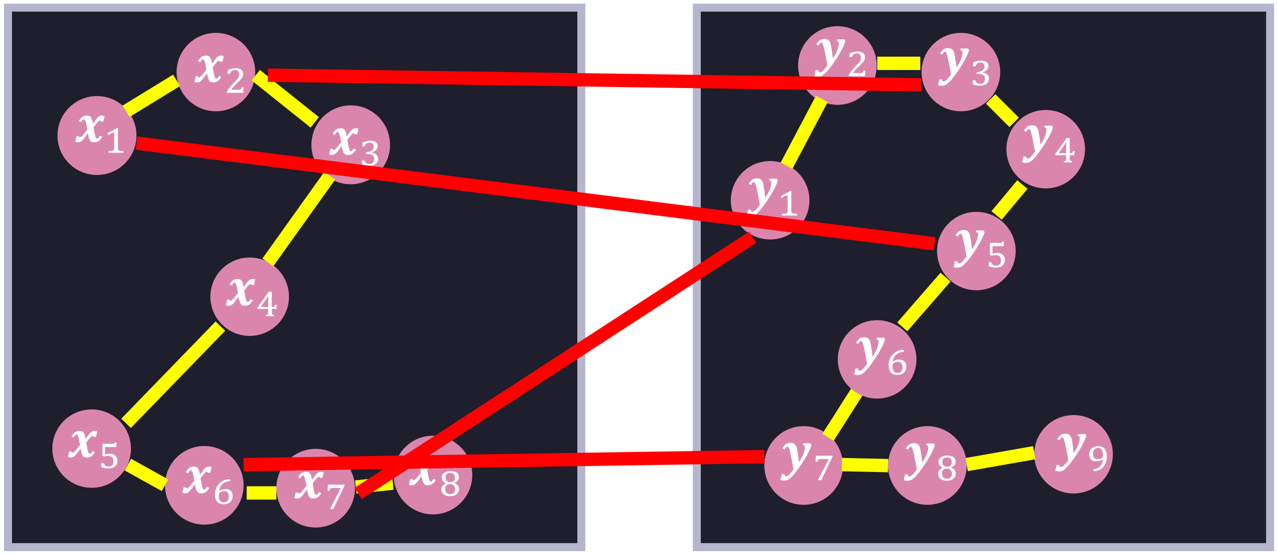
- Notation $C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A})$ indicates that the cost of alignment $\boldsymbol{A}$ depends on the specific pair of time series that we are aligning.
  - The cost of an alignment $\boldsymbol{A}$ depends on the alignment itself, as well as the two time series $\boldsymbol{X}$ and $\boldsymbol{Y}$ that we are aligning.

Rules of Alignment



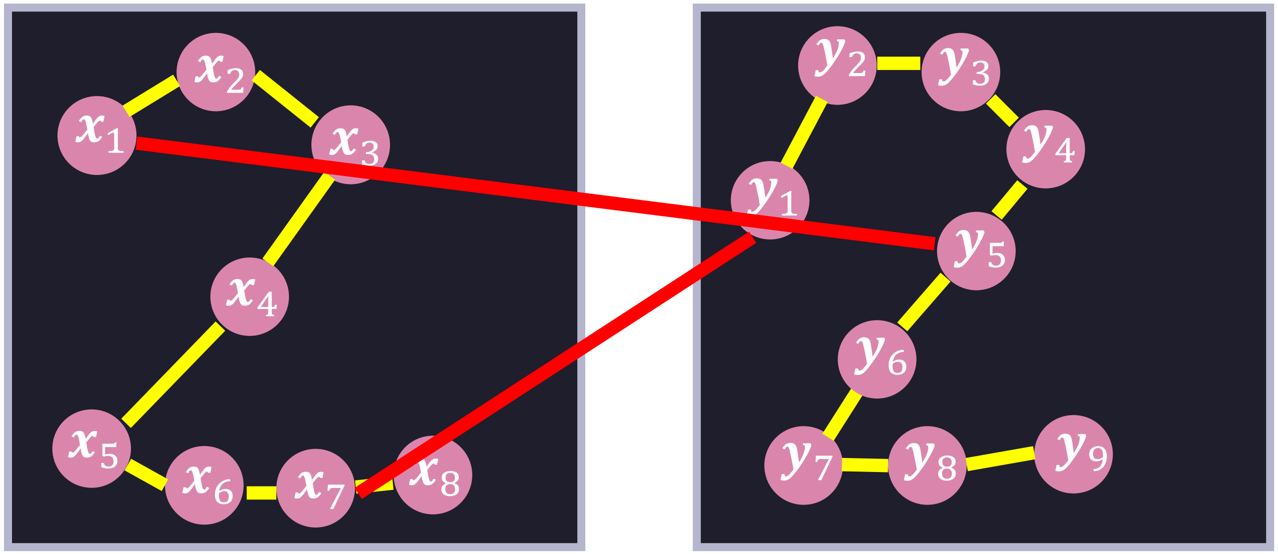
- Should alignment ((1, 5), (2, 3), (6, 7), (7, 1)) be legal?
- It always depends on what makes sense for your data.
- Typically, for time series, alignments have to obey certain rules, that this alignment violates.

Rules of Alignment



- We will require that legal alignments obey three rules:
  - Rule 1: Boundary Conditions.
  - Rule 2: Monotonicity.
  - Rule 3: Continuity
- The next slides define these rules.

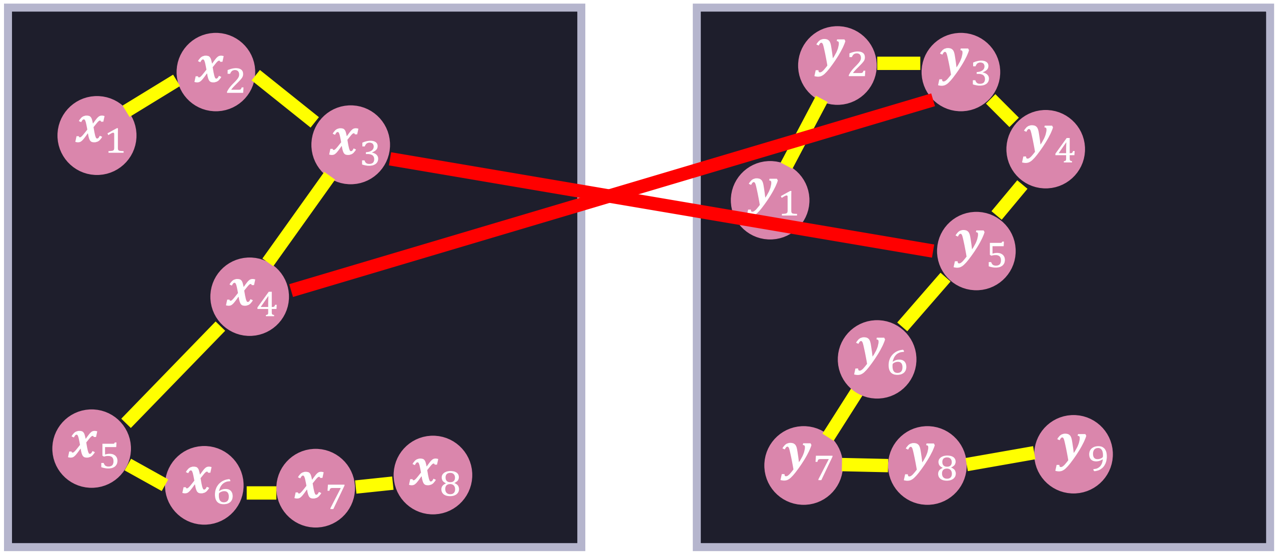
Rule 1: Boundary Conditions



- Illegal alignment (violating boundary conditions):
  - $((1,5), \ldots, (7,1))$.
  - $\left((s_1, t_1), (s_2, t_2), \ldots, (s_R, t_R)\right)$
- Alignment rule #1 (boundary conditions):
  - $s_1 = 1,\ t_1 = 1.$
  - $s_R = M =$ length of first time series
  - $t_R = N =$ length of second time series

first elements match
last elements match

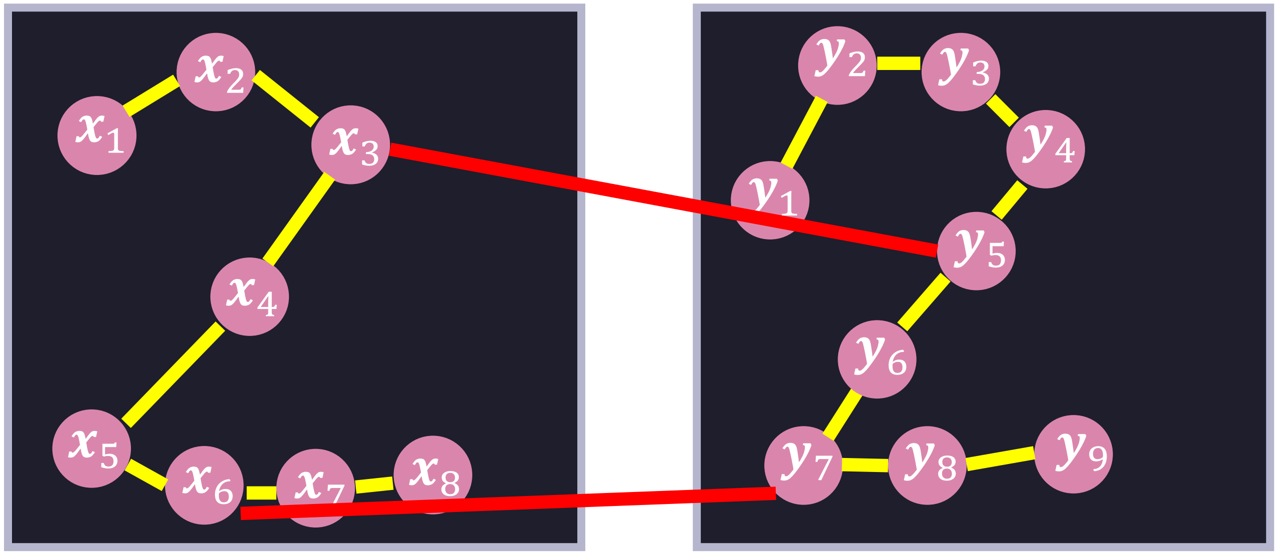
Rule 2: Monotonicity



- Illegal alignment (violating monotonicity):
  - $(\dots, (3, 5), (4, 3), \dots)$.
  - $\big((s_1, t_1), (s_2, t_2), \dots, (s_R, t_R)\big)$
- Alignment rule #2: sequences $s_1, \dots, s_R$ and $t_1, \dots, t_R$ are monotonically increasing.
  - $(s_{i+1} - s_i) \geq 0$
  - $(t_{i+1} - t_i) \geq 0$

The alignment cannot go backwards.

Rule 3: Continuity



- Illegal alignment (violating continuity):
  - $(\dots, (3, 5), (6, 7), \dots)$.
  - $\left((s_1, t_1), (s_2, t_2), \dots, (s_R, t_R)\right)$
- Alignment rule #3: sequences $s_1, \dots, s_R$ and $t_1, \dots, t_R$ cannot increase by more than one at each step.
  - $(s_{i+1} - s_i) \leq 1$
  - $(t_{i+1} - t_i) \leq 1$

The alignment cannot skip elements.

Visualizing a Warping Path

- **Warping path** is an alternative term for an alignment

Visualizing a Warping Path

$$\boldsymbol{A} = ((1,1), (2,2), (2,3), (3,4), (4,5), (4,6), (5,7), (6,7), (7,8), (8,9))$$

- We can visualize a warping path $\boldsymbol{A}$ in 2D as follows:
- An element of $\boldsymbol{A}$ corresponds to a colored cell in the 2D table.

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$									
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

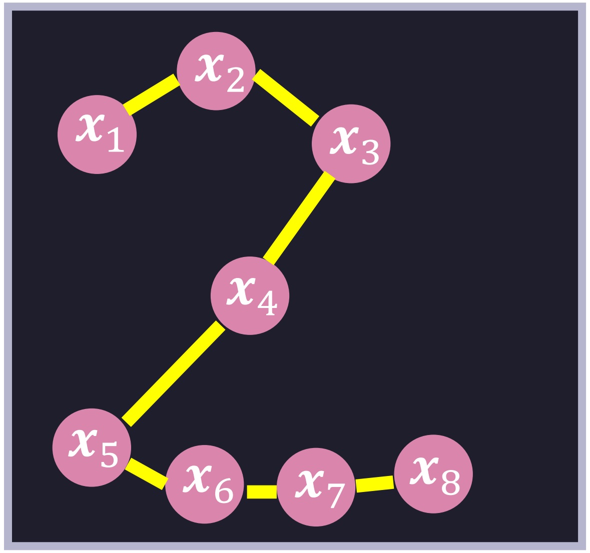
Dynamic Time Warping

- **Dynamic Time Warping** (DTW) is a distance measure between time series.
- The DTW distance is the cost of the **optimal legal alignment** between the two time series.
- The alignment must be legal: it must obey the three rules of alignments.
  - Boundary conditions.
  - Monotonicity.
  - Continuity.
- The alignment must minimize $C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A}) = \sum_{i=1}^{R} \mathrm{Cost}(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}})$
- So:

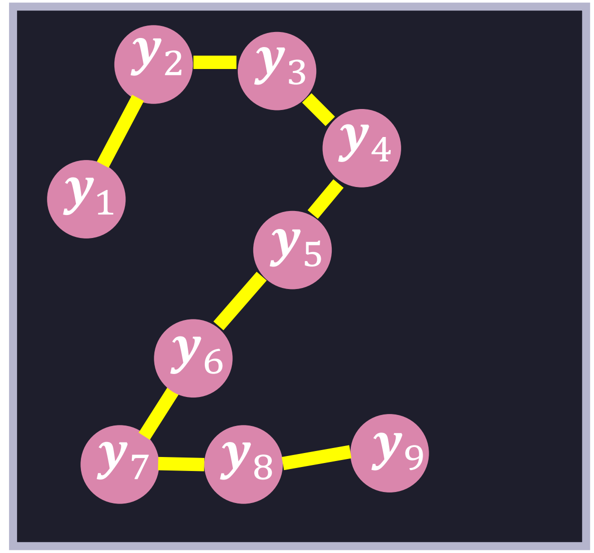
$$\mathrm{DTW}(\boldsymbol{X}, \boldsymbol{Y}) = \min_{\boldsymbol{A}} C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A})$$

where $\boldsymbol{A}$ ranges over all legal alignments between $\boldsymbol{X}$ and $\boldsymbol{Y}$.

Finding the Optimal Alignment



$\boldsymbol{X}$



$\boldsymbol{Y}$

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
  - $\boldsymbol{X}$ is a time series of length M.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
  - $\boldsymbol{Y}$ is a time series of length N.
- Each $\boldsymbol{x}_i$ and each $\boldsymbol{y}_i$ are feature vectors.

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- Dynamic programming strategy:
  - Break problem up into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M$, $1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
- We need to solve every problem(i, j) for every i and j.
- Then, the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is the solution to problem(M, N).

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M$, $1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
- How do we start computing the optimal alignment?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M, 1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(1, 1)$:
  - How is problem$(1, 1)$ defined?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M$, $1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
- Solve problem$(1, 1)$:
  - Find optimal alignment between $(\boldsymbol{x}_1)$ and $(\boldsymbol{y}_1)$.
  - What is the optimal alignment between $(\boldsymbol{x}_1)$ and $(\boldsymbol{y}_1)$?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M$, $1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(1, 1)$:
  - Find optimal alignment between $(\boldsymbol{x}_1)$ and $(\boldsymbol{y}_1)$.
  - The optimal alignment is the **only legal alignment**: $((1, 1))$.

Alignment for Problem $(1, 1)$

$$\boldsymbol{A} = ((1, 1)\ )$$

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$									
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M$, $1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
- Solve problem$(1, j)$:
  - How is problem$(1, j)$ defined?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M$, $1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
- Solve problem$(1, j)$:
  - Find optimal alignment between $(\boldsymbol{x}_1)$ and $(\boldsymbol{y}_1, \dots \boldsymbol{y}_j)$.
  - What is the optimal alignment between $(\boldsymbol{x}_1)$ and $(\boldsymbol{y}_1, \dots, \boldsymbol{y}_j)$?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M$, $1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(1, j)$:
  - Find optimal alignment between $(\boldsymbol{x}_1)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
  - Optimal alignment: $((1,1),(1,2), \dots, (1, j))$.
  - Here, $\boldsymbol{x}_1$ is matched with each of the first j elements of $\boldsymbol{Y}$.

Alignment for Problem $(1, 5)$

$$\boldsymbol{A} = ((1, 1), (1,2), (1,3), (1,4), (1,5))$$

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$									
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M, 1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(i, 1)$:
  - How is problem$(i, 1)$ defined?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M$, $1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(i, 1)$:
  - Find optimal alignment between $(\boldsymbol{x}_1, \dots \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1)$.
  - What is the optimal alignment between $(\boldsymbol{x}_1, \dots \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1)$?

Finding the Optimal Alignment

- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \leq i \leq M$, $1 \leq j \leq N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem$(i, 1)$:
  - Find optimal alignment between $(\boldsymbol{x}_1, \dots \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1)$.
  - Optimal alignment: $((1,1), (2,1), \dots, (i, 1))$.
  - Here, $\boldsymbol{y}_1$ is matched with each of the first i elements of $\boldsymbol{X}$.

Alignment for Problem $(5,1)$

$$\boldsymbol{A} = ((1,1),(2,1),(3,1),(4,1),(5,1))$$

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$									
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

Finding the Optimal Alignment

- $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
- We want to find the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- We break up the problem into a 2D array of smaller, interrelated problems (i, j), where $1 \le i \le M$, $1 \le j \le N$.
- Problem(i, j):
  - find optimal alignment between $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ and $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.
- Solve problem(i, j):
  - Here is where dynamic programming comes into play.
  - The solution can be obtained from the solutions to problem$(i, j-1)$, problem$(i-1, j)$, problem$(i-1, j-1)$.

Finding the Optimal Alignment

- Solving problem(i, j):
- Look at the following three alignments:
  - Let $\boldsymbol{A}_{i,j-1}$ be the solution to problem$(i, j - 1)$.
  - Let $\boldsymbol{A}_{i-1,j}$ be the solution to problem$(i - 1, j)$.
  - Let $\boldsymbol{A}_{i-1,j-1}$ be the solution to problem$(i - 1, j - 1)$.
- Let $\boldsymbol{A^*}_{i,j}$ be the alignment among $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i,j-1}$, and $\boldsymbol{A}_{i-1,j-1}$ with the smallest cost.
- Then, the solution to problem(i, j) is obtained by appending pair (i, j) to the end of $\boldsymbol{A^*}_{i,j}$.

Computing DTW$(\boldsymbol{X}, \boldsymbol{Y})$

- Input:
  - $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_M)$.
  - $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_N)$.
- Initialization:
  - C = zeros(M, N). % Zero matrix, of size $M \times N$.
  - $C(1,1)$ = Cost$(\boldsymbol{x}_1, \boldsymbol{y}_1)$.
  - For $i = 2$ to m: $C(i,1) = C(i-1,1)$ +Cost$(\boldsymbol{x}_i, \boldsymbol{y}_1)$.
  - For $j = 2$ to n: $C(1,j) = C(1,j-1)$ +Cost$\left(\boldsymbol{x}_1, \boldsymbol{y}_j\right)$.
- Main loop:
  - For $i = 2$ to m, for $j = 2$ to n:

    $C(i,j) = \min\{C(i-1,j),\ C(i,j-1), C(i-1,j-1)\} + \mathrm{Cost}\left(\boldsymbol{x}_i, \boldsymbol{y}_j\right)$.
- Return $C(M,N)$.

Cost vs. Alignment

- Note: there are two related but different concepts here: **optimal alignment** and **optimal cost**.
  - We may want to find the optimal **alignment** between two time series, to visualize how elements of those two time series correspond to each other.
  - We may want to compute the DTW distance between two time series $\boldsymbol{X}$ and $\boldsymbol{Y}$, which means finding the **cost** of the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$. We can use such DTW distances, for example, for nearest neighbor classification.
- The pseudocode in the previous slide computes the **cost** of the optimal alignment, **without** explicitly outputting the optimal alignment itself.

Cost vs. Alignment

- If you want to compute both the cost of the optimal alignment, and the optimal alignment itself, there are two ways you can modify the pseudocode to do that:
- Option 1: Maintain an **alignment array** W of size $M \times N$.
  - Every time we save a cost at $C(i, j)$, we should save the corresponding optimal alignment at $W(\mathrm{i}, \mathrm{j})$.
  - This alignment is obtained by adding pair (i, j) to the **end** of the optimal alignment stored at one of $W(\mathrm{i}-1, \mathrm{j})$, $W(\mathrm{i}, \mathrm{j}-1)$, $W(\mathrm{i}-1, \mathrm{j}-1)$.
- Option 2: Maintain a **backtrack array** B of size $M \times N$**.**
  - Every time we save a cost to $C(i, j)$, we should record at $B(\mathrm{i}, \mathrm{j})$ the choice, among $(i-1, j)$, $(i, j-1)$, $(i-1, j-1)$, that we made.
  - When we are done computing $C(M, N)$, we use array B to **backtrack** and recover the optimal alignment between $\boldsymbol{X}$ and $\boldsymbol{Y}$.

DTW Complexity

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$					?				
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

- For each (i, j):
  - Compute optimal alignment between $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \ldots \boldsymbol{y}_j)$.
  - The solution is computed based on the solutions for $(i-1, j), (i, j-1), (i-1, j-1)$.
  - Time complexity?

DTW Complexity

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$									
$\boldsymbol{x}_5$					?				
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

- For each (i, j):
  - Compute optimal alignment between $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i)$ and $(\boldsymbol{y}_1, \ldots \boldsymbol{y}_j)$.
  - The solution is computed based on the solutions for $(i-1, j), (i, j-1), (i-1, j-1)$.
  - Time complexity? Linear to the size of the C array ($M \times N$).

DTW Complexity

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
x_1									
x_2									
x_3									
x_4									
x_5					?				
x_6									
x_7									
x_8									

- Assume that both time series have approximately the same length M.
- Then, the time complexity of DTW is $O(M^2)$.
  - Improvements and variations have been proposed, with lower time complexity.

DTW Finds the Optimal Alignment

- Proof:

DTW Finds the Optimal Alignment

- Proof: by induction.
- Base cases:

DTW Finds the Optimal Alignment

- Proof: by induction.
- Base cases:
  - $i = 1$ OR $j = 1$.

DTW Finds the Optimal Alignment

- Proof: by induction.
- Base cases:
  - $i = 1$ OR $j = 1$.
- Proof of claim for base cases:
  - For any problem$(1, j)$ and problem$(i, 1)$, only one legal warping path exists.
    - Unless we consider warping paths with repetitions, like $((1,1),(1,1),(1,2))$, which are legal, but can never have lower cost than their non-repeating versions, like $((1,1),(1,2))$.
  - Therefore, DTW finds the optimal path for problem$(1, j)$ and problem$(i, 1)$.
    - It is optimal since it is the *only one*.

DTW Finds the Optimal Alignment

- Proof: by induction.
- General case:
  - (i, j), for $i \geq 2, j \geq 2$.
- Inductive hypothesis:

DTW Finds the Optimal Alignment

- Proof: by induction.
- General case:
  - (i, j), for $i \geq 2, j \geq\ 2$.
- Inductive hypothesis:
  - What we want to prove for (i, j) is true for $(i-1, j), (i, j-1), (i-1, j-1)$:

DTW Finds the Optimal Alignment

- Proof: by induction.
- General case:
  - (i, j), for $i \geq 2, j \geq\ 2$.
- Inductive hypothesis:
  - What we want to prove for (i, j) is true for $(i-1, j), (i, j-1), (i-1, j-1)$:
  - DTW has computed optimal solutions for problems $(i-1, j), (i, j-1), (i-1, j-1)$.

DTW Finds the Optimal Alignment

- Proof: by induction.
- General case:
  - (i, j), for $i \geq 2, j \geq 2$.
- Inductive hypothesis:
  - What we want to prove for (i, j) is true for $(i-1, j), (i, j-1), (i-1, j-1)$:
  - DTW has computed optimal solutions for problems $(i-1, j), (i, j-1), (i-1, j-1)$.
- Proof by contradiction:

DTW Optimality Proof

- Proof: by induction.
- General case:
  - (i, j), for $i \geq 2, j \geq 2$.
- Inductive hypothesis:
  - What we want to prove for (i, j) is true for $(i-1, j), (i, j-1), (i-1, j-1)$:
  - DTW has computed optimal solutions for problems $(i-1, j), (i, j-1), (i-1, j-1)$.
- Proof by contradiction:
  - Summary: if solution for (i, j) is not optimal, then one of the solutions for $(i-1, j)$, $(i, j-1)$, or $(i-1, j-1)$ was not optimal, which violates the inductive hypothesis.

DTW Optimality Proof

- Let $\boldsymbol{A}_{i,j}$ be the DTW solution to problem(i, j).
  - We want to prove that $\boldsymbol{A}_{i,j}$ is optimal, using the inductive hypothesis.
- Let $\boldsymbol{A}^{*}{}_{i,j}$ be the alignment among $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i,j-1}$, and $\boldsymbol{A}_{i-1,j-1}$ with the smallest cost.
- Then, $\boldsymbol{A}_{i,j} = \boldsymbol{A}^{*}{}_{i,j} \oplus (i, j)$.
  - $\boldsymbol{Q} \oplus r$ is the result of appending item r to the end of list $\boldsymbol{Q}$.
- Let $\boldsymbol{B}_{i,j}$ be the **<u>optimal</u>** solution to problem(i, j).
- $\boldsymbol{B}_{i,j} = \boldsymbol{B}^{*}{}_{i,j} \oplus (i, j)$ for some $\boldsymbol{B}^{*}$.
- Assume contradiction ($\boldsymbol{B}_{i,j} \neq \boldsymbol{A}_{i,j}$).

DTW Optimality Proof

- DTW solution for problem(i, j):

$$\boldsymbol{A}_{i,j} = \left((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \dots, (a_{R,1}, a_{R,2})\right)$$

- Optimal solution for problem(i, j):

$$\boldsymbol{B}_{i,j} = \left((b_{1,1}, b_{1,2}), (b_{2,1}, b_{2,2}), \dots, (b_{S,1}, b_{S,2})\right)$$

- Assume contradiction: $\boldsymbol{A}_{i,j}$ is not optimal. Then,

$$C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{A}_{i,j}) > C_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{B}_{i,j})$$

- Why?
  - Because $\boldsymbol{B}_{i,j}$ is optimal and $\boldsymbol{A}_{i,j}$ is not optimal.

DTW Optimality Proof

- $\boldsymbol{A}_{i,j} = \left((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \dots, (a_{R,1}, a_{R,2})\right)$
- $\boldsymbol{B}_{i,j} = \left((b_{1,1}, b_{1,2}), (b_{2,1}, b_{2,2}), \dots, (b_{S,1}, b_{S,2})\right)$

$C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i,j}\right) > C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{B}_{i,j}\right) \Rightarrow$

$$\sum_{i=1}^{R} \text{Cost}\left(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}}\right) > \sum_{i=1}^{S} \text{Cost}\left(\boldsymbol{x}_{b_{i,1}}, \boldsymbol{y}_{b_{i,2}}\right)$$

- Why?
  - We are just using the definition of the cost of an alignment.

DTW Optimality Proof

- $\boldsymbol{A}_{i,j} = \left((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \dots, (a_{R,1}, a_{R,2})\right)$
- $\boldsymbol{B}_{i,j} = \left((b_{1,1}, b_{1,2}), (b_{2,1}, b_{2,2}), \dots, (b_{S,1}, b_{S,2})\right)$

$$\sum_{i=1}^{R} \mathrm{Cost}\left(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}}\right) > \sum_{i=1}^{S} \mathrm{Cost}(\boldsymbol{x}_{b_{i,1}}, \boldsymbol{y}_{b_{i,2}}) \Rightarrow$$

$$\sum_{i=1}^{R-1} \mathrm{Cost}\left(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}}\right) > \sum_{i=1}^{S-1} \mathrm{Cost}(\boldsymbol{x}_{b_{i,1}}, \boldsymbol{y}_{b_{i,2}})$$

- Why? The last element of both $\boldsymbol{A}_{i,j}$ and $\boldsymbol{B}_{i,j}$ is (i, j).
  - Both $\boldsymbol{A}_{i,j}$ and $\boldsymbol{B}_{i,j}$ align $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_i)$ with $\left(\boldsymbol{y}_1, \dots \boldsymbol{y}_j\right)$.

DTW Optimality Proof

- $\boldsymbol{A}_{i,j} = \big((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \ldots, (a_{R,1}, a_{R,2})\big)$
- $\boldsymbol{B}_{i,j} = \big((b_{1,1}, b_{1,2}), (b_{2,1}, b_{2,2}), \ldots, (b_{S,1}, b_{S,2})\big)$

$$\sum_{i=1}^{R-1} \mathrm{Cost}\big(\boldsymbol{x}_{a_{i,1}}, \boldsymbol{y}_{a_{i,2}}\big) > \sum_{i=1}^{S-1} \mathrm{Cost}(\boldsymbol{x}_{b_{i,1}}, \boldsymbol{y}_{b_{i,2}}) \Rightarrow$$

$$C_{\boldsymbol{X},\boldsymbol{Y}}\big(\boldsymbol{A}^*{}_{i,j}\big) > C_{\boldsymbol{X},\boldsymbol{Y}}\big(\boldsymbol{B}^*{}_{i,j}\big)$$

- Why? We are just using the definitions of $\boldsymbol{A}^*{}_{i,j}$ and $\boldsymbol{B}^*{}_{i,j}$.
  - $\boldsymbol{A}_{i,j} = \boldsymbol{A}^*{}_{i,j} \oplus (i,j)$.
  - $\boldsymbol{B}_{i,j} = \boldsymbol{B}^*{}_{i,j} \oplus (i,j)$

DTW Optimality Proof

- $\boldsymbol{A}_{i,j} = \left((a_{1,1}, a_{1,2}), (a_{2,1}, a_{2,2}), \ldots, (a_{R,1}, a_{R,2})\right)$
- $\boldsymbol{B}_{i,j} = \left((b_{1,1}, b_{1,2}), (b_{2,1}, b_{2,2}), \ldots, (b_{S,1}, b_{S,2})\right)$

$C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}^{*}{}_{i,j}\right) > C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{B}^{*}{}_{i,j}\right) \Rightarrow$

$\min\left(C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i,j-1}\right), C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i-1,j}\right), C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i-1,j-1}\right)\right) > C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{B}^{*}{}_{i,j}\right)$

- Why? We are just using the definitions of $\boldsymbol{A}^{*}{}_{i,j}$ and $\boldsymbol{B}^{*}{}_{i,j}$.
  - $\boldsymbol{A}^{*}{}_{i,j} = \operatorname{argmin}\left(C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i,j-1}\right), C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i-1,j}\right), C_{\boldsymbol{X},\boldsymbol{Y}}\left(\boldsymbol{A}_{i-1,j-1}\right)\right)$

DTW Optimality Proof

$\min(C_{X,Y}(\boldsymbol{A}_{i,j-1}), C_{X,Y}(\boldsymbol{A}_{i-1,j}), C_{X,Y}(\boldsymbol{A}_{i-1,j-1})) > C_{X,Y}(\boldsymbol{B}^*{}_{i,j})$

Contradiction!!!

- The inductive hypothesis was that alignments $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i-1,j}$, $\boldsymbol{A}_{i-1,j-1}$ were the optimal alignments for problems $(i-1,j), (i,j-1), (i-1,j-1)$.
- However, $\boldsymbol{B}^*{}_{i,j}$ is also an alignment for one of the three problems.
  - Otherwise, $\boldsymbol{B}_{i,j}$ would break one of the three rules of legal alignments.
- Plus, $\boldsymbol{B}^*{}_{i,j}$ has a lower cost than each of $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i-1,j}$, $\boldsymbol{A}_{i-1,j-1}$, so one of those three alignments is not optimal..
- This contradicts the inductive hypothesis.

Visualizing $\boldsymbol{B}^*{}_{i,j}$ for $i = 4,\ j = 5$

	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\boldsymbol{y}_4$	$\boldsymbol{y}_5$	$\boldsymbol{y}_6$	$\boldsymbol{y}_7$	$\boldsymbol{y}_8$	$\boldsymbol{y}_9$
$\boldsymbol{x}_1$									
$\boldsymbol{x}_2$									
$\boldsymbol{x}_3$									
$\boldsymbol{x}_4$					?				
$\boldsymbol{x}_5$									
$\boldsymbol{x}_6$									
$\boldsymbol{x}_7$									
$\boldsymbol{x}_8$									

- $\boldsymbol{B}_{i,j}$ aligns $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i)$ with $(\boldsymbol{y}_1, \ldots \boldsymbol{y}_j)$.
- $\boldsymbol{B}^*{}_{i,j}$ is the result of removing the last element from $\boldsymbol{B}_{i,j}$.
- Based on the rules of continuity and monotonicity, the last element of $\boldsymbol{B}^*{}_{i,j}$ must be $(i-1, j), (i, j-1)$, or $(i-1, j-1)$.

DTW Optimality Proof - Recap

- Proof: by induction.
- The base cases are easy to prove.
- The tricky part is to prove the inductive case.
  - I.e., prove that $\boldsymbol{A}_{i,j}$, which is the DTW solution problem (i, j), is indeed the optimal alignment for that problem.
- We prove the inductive case by contradiction:
  - Inductive hypothesis: alignments $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i-1,j}$, $\boldsymbol{A}_{i-1,j-1}$ were the optimal alignments for problems $(i-1, j), (i, j-1), (i-1, j-1)$.
  - If $\boldsymbol{A}_{i,j}$ is not optimal, then we prove that one of alignments $\boldsymbol{A}_{i,j-1}$, $\boldsymbol{A}_{i-1,j}$, $\boldsymbol{A}_{i-1,j-1}$ must not be optimal, which contradicts the inductive hypothesis.

Dynamic Time Warping, Recap

- It is oftentimes useful to use nearest neighbor classification to classify time series.
  - Especially when we have very few training examples per class, or, in the extreme, only one training example per class.
- However, to use nearest neighbor classification, we need to define a meaningful distance measure between time series.
- The Euclidean distance works poorly because even slight misalignments can lead to large distance values.
- Dynamic Time Warping is based on computing an optimal alignment between two time series, and thus it can tolerate even large misalignments.
- The complexity of Dynamic Time Warping is quadratic to the length of the time series, but more efficient variants can also be used.

Other Distance Measures

- There are more distance measures that can be used on time series data. For example:
  - Edit Distance with Real Penalty (ERP).

Lei Chen and Raymond Ng. "On the marriage of lp-norms and edit distance." In *International Conference on Very Large Data Bases (VLDB)*, pp. 792-803, 2004.

  - Time Warp Edit Distance (TWED).

Pierre-François Marteau. "Time warp edit distance with stiffness adjustment for time series matching." *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI),* 31, no. 2 (2009): 306-318.

  - Move-Split-Merge (MSM).

Alexandra Stefan, Vassilis Athitsos, and Gautam Das. "The move-split-merge metric for time series." *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 25, no. 6 (2013): 1425-1438.