Time series

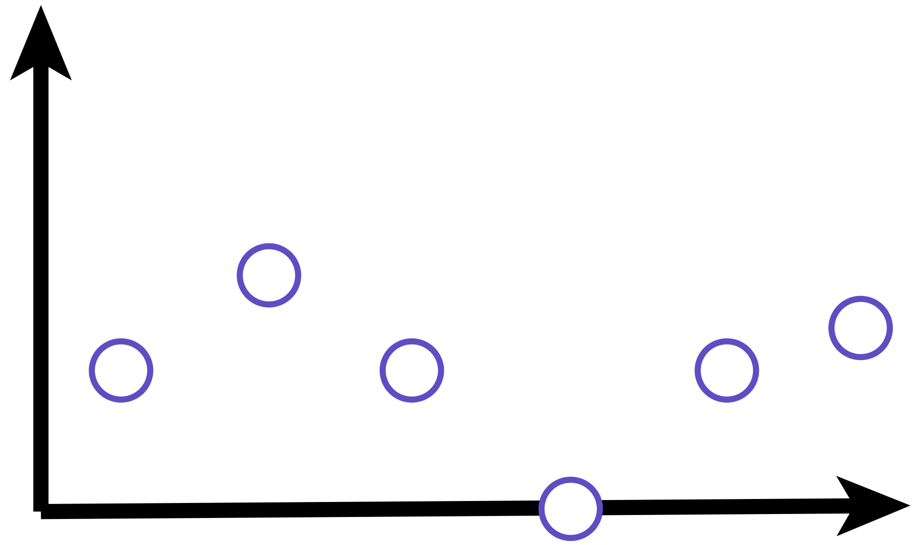
Università di Pisa

Data... over time.

Time series

A univariate series $s \in \mathcal{X}^n$ is a sequence $s = [s_1, \ldots, s_n]$ of n values in a domain $\mathcal{X}$. A series is defined by:

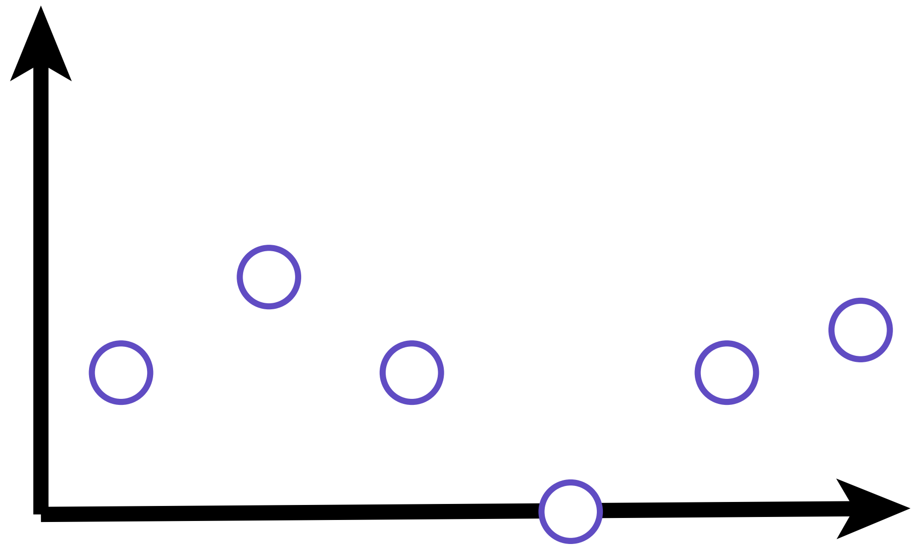
- *Type*: discrete, e.g., nucleotide bases, or continuous, e.g., stock values in a financial market
- *Sampling rate*: How often values are sampled, e.g., daily
- *Amplitude*: Values sampled, e.g., value of the stock on a particular day


A time series.

Time series

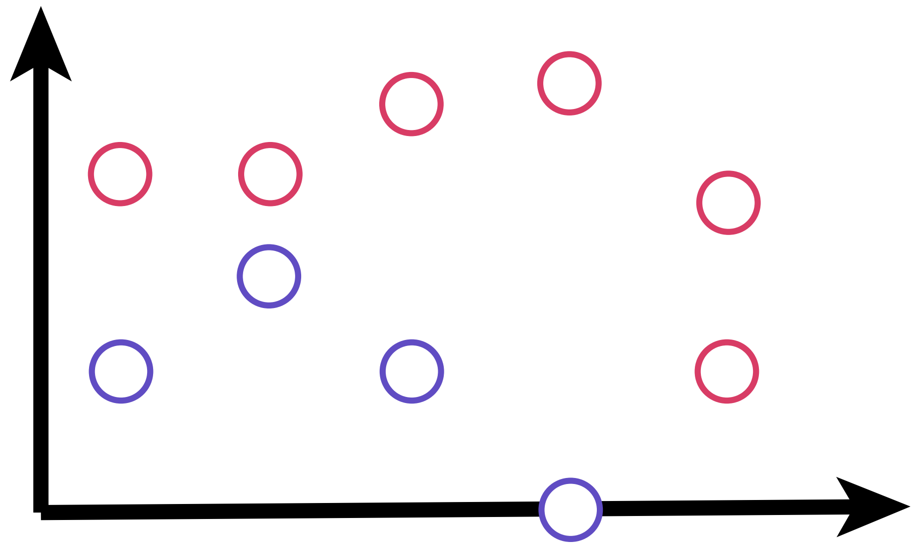
A univariate series $s \in \mathcal{X}^K$ is a sequence $s = [s_1, \ldots, s_K]$ of K values in a domain $\mathcal{X}$. A series is defined by:

- *Seasonality*: Series repeat (or almost repeat) over time, e.g., temperature
- *Period*: How much does it take for the series to repeat itself, e.g., length of a calendar year


A time series.

Time series

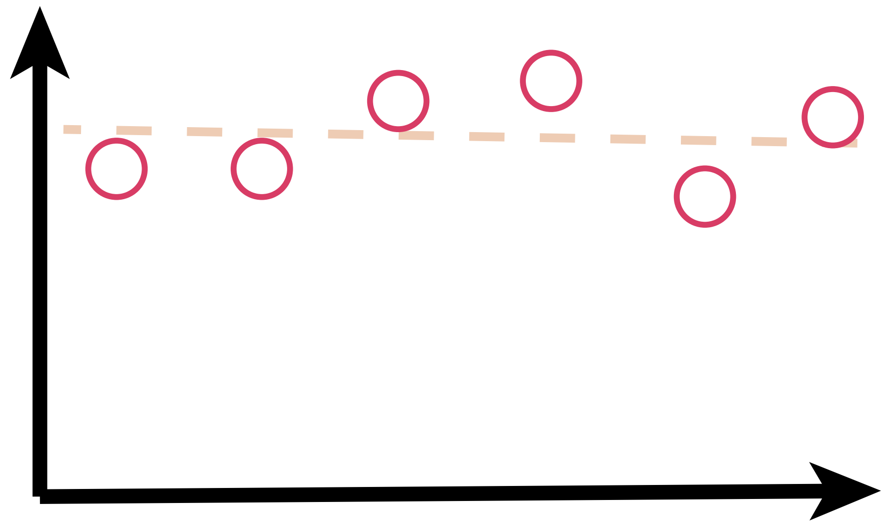
A multivariate time series s generalizes time series to multiple variables. Each instance is comprised of multiple time series, each representing a different feature.


A multivariate time series, with a variable in red, one in blue.

Time series statistics

UNIVERSITÀ DI PISA

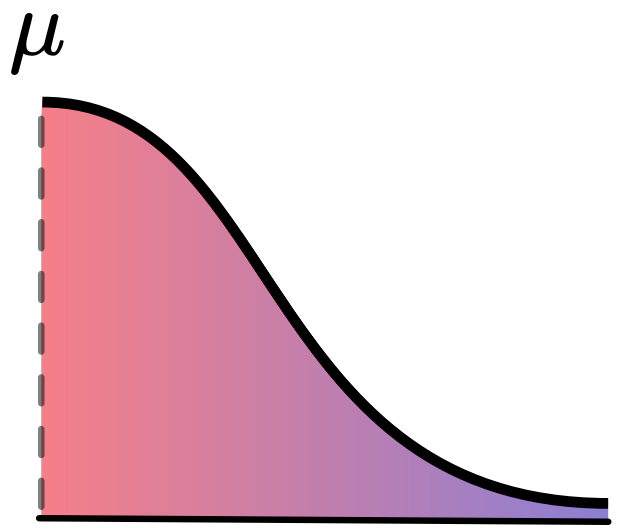
- Mean. Expected value $\mathbb{E}[s]$ of s
- Variance. Variance of s
- Trend: slope Δ of a linear model modeling s



A time series, and its trend (in beige).

Time series statistics

- Interquantile ranges. $q_a(s) - q_b(s)$
- Skewness: is the distribution symmetric? $\mathbb{E}[(\frac{s-\mu}{\sigma})^3]$
- Kurtosis: what is the probability mass on the tails? $\mathbb{E}[(\frac{s-\mu}{\sigma})^4]$



Empirical distribution of a time series components, values color-coded by density.

Time series: local behaviors

Time series can be affected by:

- *Seasonality*: the series has some repeated periodic behavior, e.g., temperatures fall every winter
- *Trends*: the series tends to have a monotonic behavior, with values increasing/decreasing

We need to analyze time series on a local scale.

Time series statistics: rolling statistics

Rolling indicates the act of extracting a series of consecutive subsequences of given dimensionalities $w^1, \dots, w^k$. Each subseries gives a different view on s, and is thus named *window*. Given a series of windows, we can now *locally* describe a time series!

- Rolling mean
- Rolling variance
- ...

Time series statistics: sliding statistics

Unlike rolling statistics, which compute a statistic over a subseries of a given series, *sliding* statistics slide a series t over a series s, computing a statistic between the two. The act of sliding a window through a function is called *convolution*.

Auto-covariance
How much does a component of a time series correlate with previous and future components? Slides a series over itself, computing covariance between the two components:

$$K_s = cov(s_{t:}, s_{t+\Delta:}) \sum_{i=1}^{n} s_i \cdot s_{i+\Delta}.$$

High autocovariances may indicate seasonality in the series.

Time series: sliding statistics

Cross-correlation

Shifted pointwise correlation of the two series a, b, measured as a *sliding* inner product:

$$CC_\Delta(a, b) = \sum_{i=1}^{n} a_i \cdot b_{i+\Delta}.$$

For univariate time series, the inner product is simply a multiplication.

From analyzing to transforming

Representing by segmenting

Università di Pisa

Given a set of time series $S = \{s^1, \ldots, s^n\}$, let $S^\subset$ be a set of k subseries $s^1_\subset, \ldots, s^k_\subset$. We define an alphabet Σ of symbols, each symbol assigned to a subseries. Then, we can *segment* each series into a sequence of subseries, and represent each time series as a series of symbols in Σ^*.

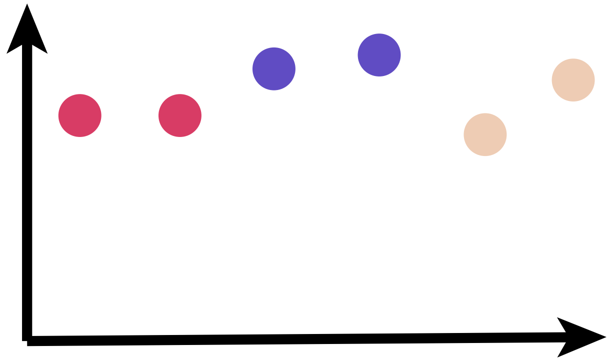
$$\boldsymbol{s}^i \to \underbrace{\left[\boldsymbol{s}^{i,1} \mid \cdots \mid \boldsymbol{s}^{i,k_i}\right]}_{\in \Sigma^*}$$

We can tackle two problems independently:

- Segmentation: how to split a series into subseries?
- Transformation: what symbols do we use to define each segment?

Fixed window-based segmentation

Each series s^i has an often intractable number of possible subseries, which makes exhaustive search unfeasible. Instead, we choose an arbitrary window w, and segment each series into a set of $\frac{n}{w}$ subseries.



A fixed window of size $w = 2$, segmenting a time series in three non-overallping adjacent windows (color-coded).

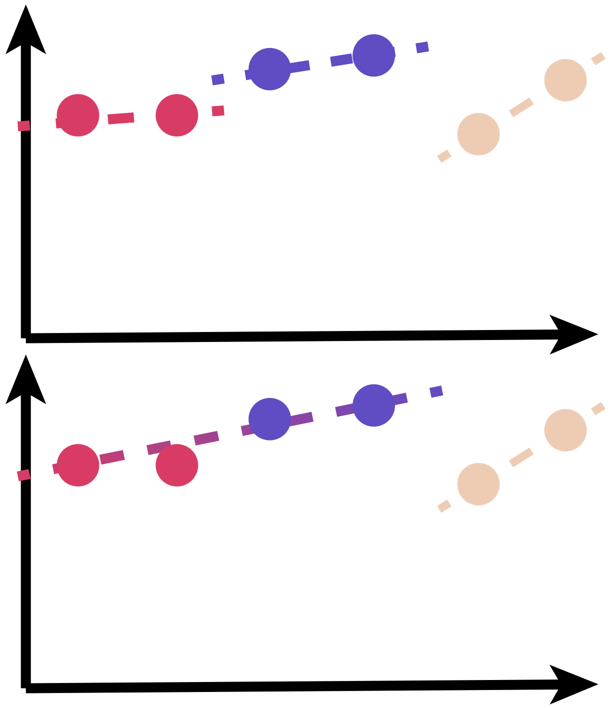
Learned segmentation

UNIVERSITÀ DI PISA

We *learn* segmentations which minimizing an approximation error.

Piecewise Linear Approximation (PLA) defines a segmentation minimizing segment-local linear models of the data.

- Given number of segments: segmentation which distributes approximation error as evenly as possible among segments
- Given error bound: segmentation which generates the *minimum* number of segments within given error bound ε


Segmentation on minimum error (top) and given number of segments (bottom).

Learned equidistributional windows

We *learn* segmentations by their likelihood: given a desired number of segments, segments are learned to maximize their probability $p(s^i_\subset)$, which results in segments as equiprobable as possible, ideally following a uniform distribution. By maximizing probability, we maximize entropy, and thus the diversity of the subseries.

Continuous transformations

	Description	**Type**	f^{Σ}	Σ
Piecewise Average (PA)	Average value of the segment	Continuous	$\mu_{s^i_{\subset}}$	$\mathbb{R}$
Piecewise Linear (PL)	Slope ∇ of the segment	Continuous	$\nabla_{s^i_{\subset}}$	$\mathbb{R}, \mathbb{R}^2$

$\mathbb{R}^2$ given by storing both slope and intercept of the linear model.

Two-tier segmentations

Window segmentations create series of symbols themselves! We can learn representations through a two-tier algorithm:

- continuous transformation, yielding segments $S^{\subset}$ with symbols in $\Sigma^{\subset}$
- transformation partitioning, mapping symbols in $\Sigma^{\subset}$ to discrete symbols in Σ

Symbols can be either categorical, or ordinal.

Symbolic aggregate approximation (SAX)

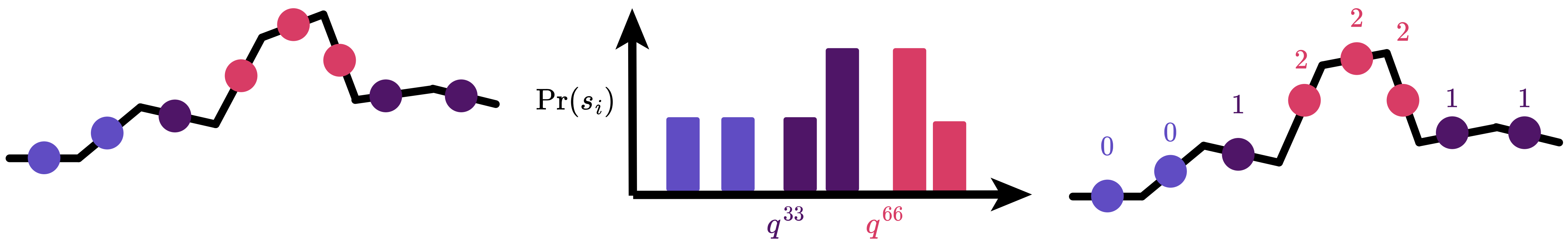
Symbolic aggregate approximation (SAX) implements a two-tier transformation:

- fixed-window segmentation followed by piecewise average transformation in subsymbols $(\Sigma^{\subset})^*$
- aggregation of subsymbols into equiprobable symbols in Σ: we partition the subsymbol distribution into equiprobable buckets, each defined by a symbol in Σ

SAX symbols are discrete *and* ordinal!

Symbolic aggregate approximation (SAX)

Università di Pisa



The three steps of SAX: subsymbol induction with fixed-window average segmentation, partitioning of the subsymbol distribution, and transformation.

Signal representations

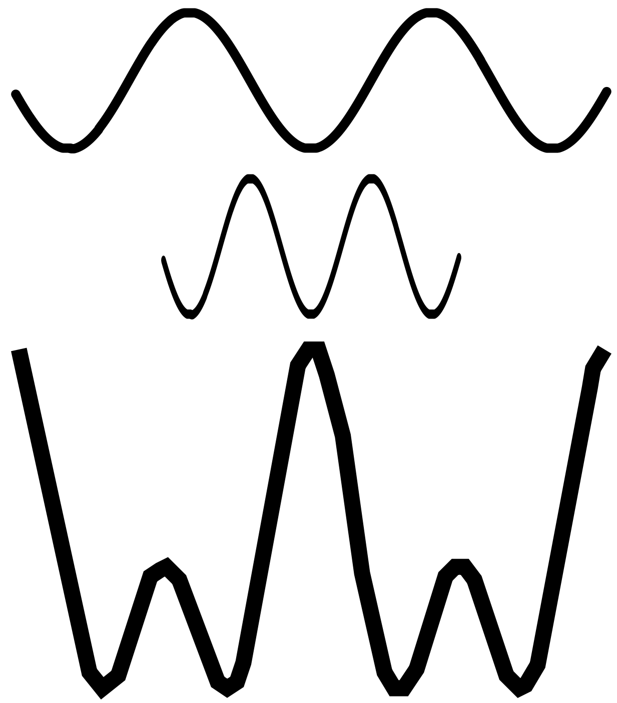
Segmentations can be tricky to handle, and simply offer a representation in a domain quite different from the original. Signal representations, on the other hand, aim to define a series in terms of other series. To stick with the signal processing literature, where they are most prevalent, we'll refer to series as *signals*.

- What other signals can we leverage?
- How do we combine them to represent the original series?

Fourier analysis

Fourier analysis tackles **periodic** (also known as stationary) series, i.e., series which periodically repeat themselves.

- What other signals can we leverage? Sine and cosine signals at different frequencies
- How do we combine them to represent the original signal? Linear combination



Two signals $\cos(x)$, $\cos(2x)$, and a linear combination $\cos(x) + cos(2x)$.

Fourier analysis

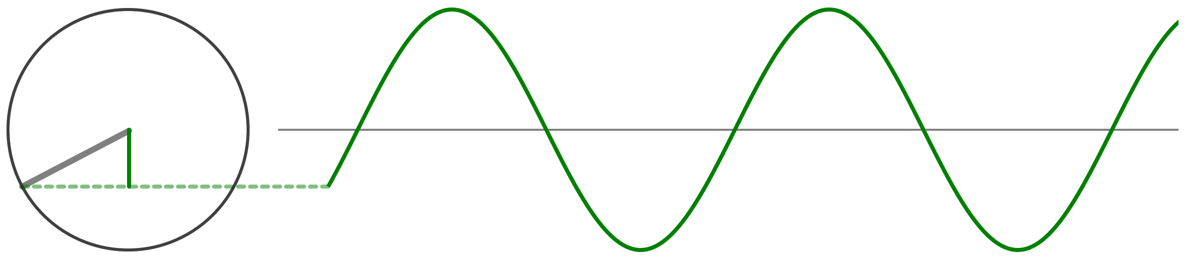
As a linear transformation, we need to learn a set of coefficients α which map the basis to the signal s. In Fourier analysis, we constrain $\alpha \in \mathbb{R}^+$:

- basis signals which do not contribute to the signal have a 0 coefficient
- basis signals which do contribute do so with a positive coefficient $\alpha_j > 0$

To compute α we can thus use *inner products*, e.g., dot product, which are guaranteed to satisfy both.

The basis: sinusoids

First, we define the basis of the transformation. We use sinusoids, i.e., $\sin/cos$ signals defined by a phasor ψ.

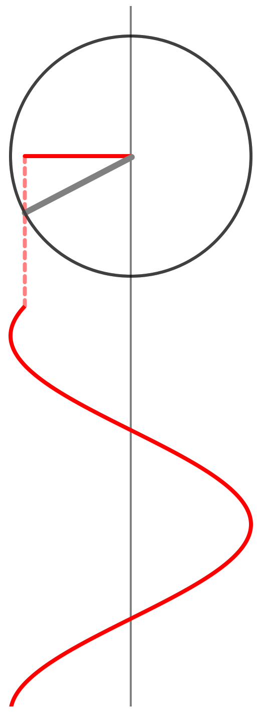


A phasor ψ, and the $\sin$ series generated.

The basis: sinusoids

UNIVERSITÀ DI PISA

First, we define the basis of the transformation. We use sinusoids, i.e., sin/cos signals defined by a phasor ψ.



A phasor ψ, and the $\cos$ series generated.

Sinusoids define amplitude

Università di Pisa

For a phasor ψ, we can define the period t_0 as the time required for the phasor to complete one rotation over the unit circle, and the frequency $f_0 = t_0^{-1}$ as the rotations per unit of time. Then, at time t, the phasor defines a series component with amplitude

$$s_t = \underbrace{\alpha}_{amplitude\ scaling} \cos(\overbrace{2\pi f_0 t}^{angle\ \theta_{0,t}} \underbrace{+\phi}_{shifting}).$$

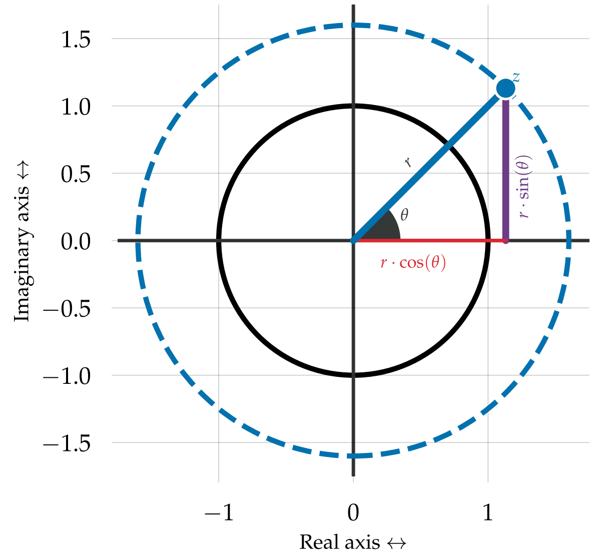
The shift ϕ indicates the shift of the sinusoid itself with respect to $t = 0$. A nice interactive visualization here.

Sinusoids and the complex unit circle

UNIVERSITÀ DI PISA

By Euler, we can map complex numbers to the complex unit circle.

$$e^{-i\theta} = \cos(\theta) + i\sin(\theta).$$

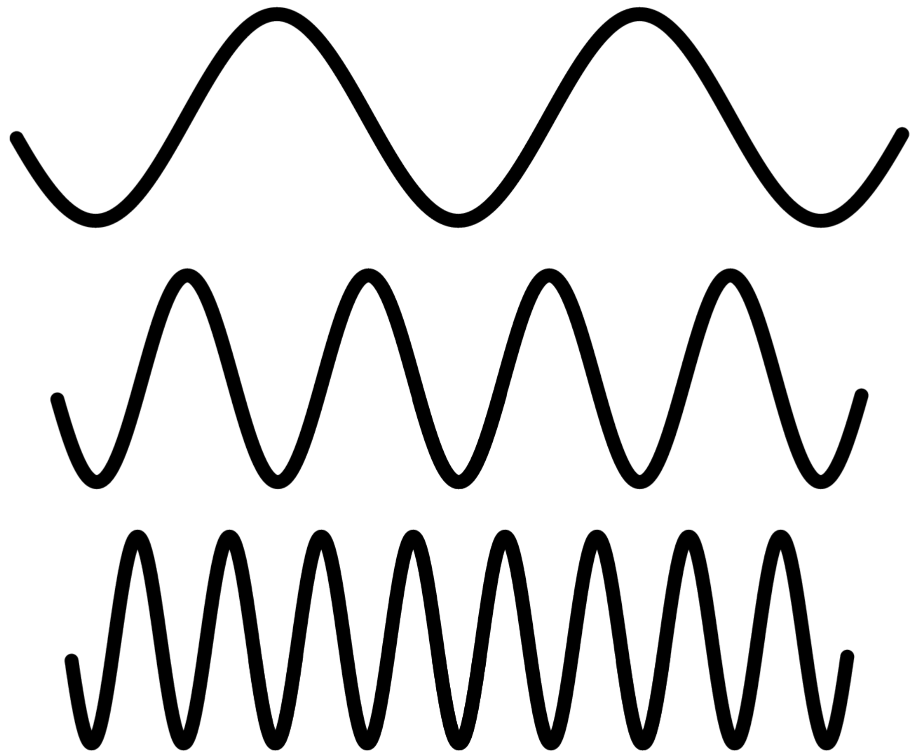


The complex unit circle and an imaginary number $z = e^{-i\theta}$. As in linear algebra, we can define a vector $(z_{\mathbb{R}}, z_{\mathbb{C}})$ through the standard basis, in this case $(1, 0)$, $(0, i)$.

Fourier, and the frequency domain

Università di Pisa

Phasors of different frequencies are the building block of the Fourier representation of the signal: we will map signals to a *frequency domain* populated by sinusoids of different frequencies.



Sinusoid signals of different frequencies define a basis in the frequency domain. An interactive visualization can be found here.

Fourier, and the coefficients

Having defined a basis, we now need to learn the coefficients α. Coefficients ought to measure the presence of a frequency, and its scaling. Thus, $\alpha \in \mathbb{R}^{+n}$, and $\alpha_j = 0$ for frequencies not present in s. Inner products, e.g., dot products, satisfy both conditions: the coefficients will be given by the inner product of basis signals and the signal s:

$$\alpha_j = s \cdot \psi_j.$$

A small caveat: orthogonality... again?

Signals may be out orthogonally out of phase, thus inducing null products, leading to misses on the basis. To tackle this, we use the orthogonal components of the basis: the $\sin$ component of the phasor!

$$s_t = \sum_{\rho=1}^{n} \alpha_\rho (\cos(\theta_{\rho,t}) + i \sin(\theta_{\rho,t}))$$

Remember: by definition, inner product satisfy $a \cdot b = 0$ for a, b orthogonal!

Discrete-Time Fourier Transform (DTFT)

Signals may be out orthogonally out of phase, thus inducing null products, leading to misses on the basis. To tackle this, we use the orthogonal components of the basis: the $\sin$ component of the phasor!

$$s_t = \sum_{\rho=1}^{n} \alpha_\rho (\cos(\theta_{\rho,t}) + i \sin(\theta_{\rho,t}))$$

Finally, we leverage Euler, and have

$$s_j = \sum_{\rho=1}^{n} \alpha_\rho e^{\theta_{\rho,t}} = \sum_{\rho=1}^{n} \alpha_\rho e^{-2\pi f_\rho t}$$

Remember: by definition, inner product satisfy $a \cdot b = 0$ for a, b orthogonal!

Discrete-Time Fourier Transform

Università di Pisa

- Quick: $\mathcal{O}(n \log n)$
- Decomposition in separate and different signals

- Decomposition defined exclusively for sinusoidal series
- Decomposition of *periodic* series
- Decomposition exclusively in terms of frequency, not time

Wavelets

Wavelet tackle several weaknesses of Fourier Transforms.

Fourier Transform: ψ_κ

- Decomposition defined exclusively for sinusoidal series
- Decomposition of *periodic* series
- Decomposition exclusively in terms of frequency, not time

vs

Wavelets: $\psi_{\kappa,\tau}$

- Flexible decomposition defined by *mother wavelets*
- Decomposition of arbitrary series
- Wavelets parameterized in frequency **and** time

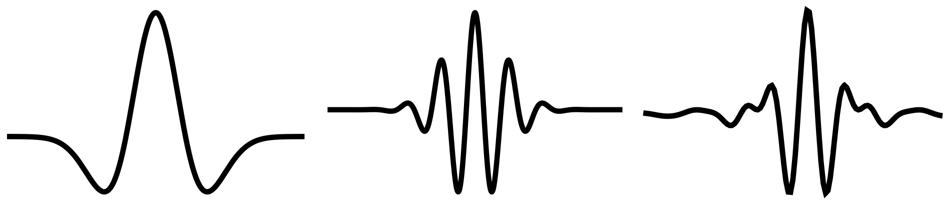
Wavelets

UNIVERSITÀ DI PISA

Wavelets (little waves) aim to replace sinusoidal phasors, and are both more general, and flexible enough for domain-specific application. A wavelet $\psi_{\kappa,\tau} : \mathbb{R} \to \mathbb{R}$ is a function s.t.

- $\int_{-\infty}^{+\infty} \psi_{\kappa,\tau}(t)\, dt = 0$ *(zero mean)*
- $\int_{-\infty}^{+\infty} \psi_{\kappa,\tau}(t)^2\, dt \neq \infty$ *(finite energy)* or *(compact support)*

Finite energy makes it so a wavelet, unlike a sinusoidal function, is bounded: thus, by construction, wavelets **can be localized in time**!


Three wavelets: Mexican Hat, Morlet, Meyer.

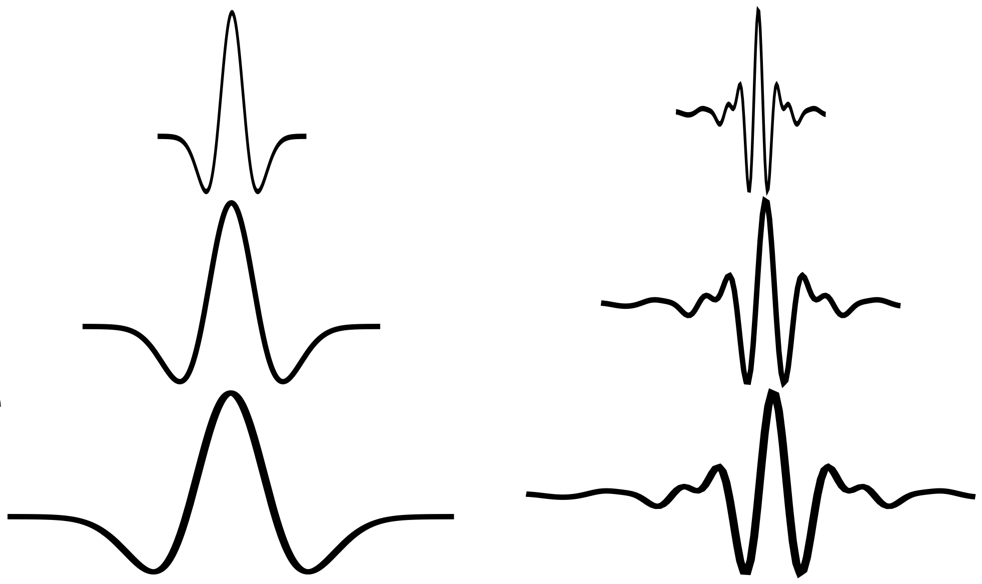
Surfing the series with wavelets

Localization is innate in wavelets by their own definition. A *mother wavelet* $\psi_{K,T}$ defines a family of *daughter* wavelets defined by

- A frequency κ: shrink or stretch the daughter wavelet
- A shift τ: pushes or pulls the daughter wavelet across time.

We define a daughter wavelet $\psi_{\kappa,\tau}$ as

$$\psi_{\kappa,\tau}(t) = \frac{1}{\sqrt{\kappa}}\psi(\frac{t-\tau}{\kappa}).$$


Stretched wavelets: Mexican Hat wavelet (left column), and Meyer wavelet (right column).

Wavelets Transforms

Moving from sinuoidal phasors to wavelets is seamless in the formulation

$$\underbrace{\sum_{\rho=1}^{n} \alpha_\rho \psi_\kappa(t)}_{Discrete\ Fourier} \rightarrow \underbrace{\sum_{t=1}^{n} \alpha_\rho \psi_{\kappa,\tau}(t)}_{Wavelet\ Transform} \ .$$

Wavelets are convoluted across the series, producing a list of coefficients. κ and τ are dyadic, i.e., they are taken as powers of 2: $\tau = 2^{-i}, \tau = k2^{-i}$.

From representations to motifs

Time series: motifs

Motifs tie strongly with subseries extraction for discretization: if representation algorithms extracted to *represent*, and thus describe, motif extraction algorithms instead extract subseries to *represent*, and thus *discriminate*. In other words, a *motif* is a subseries characteristic of a set of series. We can search motifs following the two views:

- descriptive: a motif is a reoccurring subseries in the set S
- discriminative: a motif is a reoccurring subseries in the set S... and not reoccurring in another set $S^{\neq}$. Also called a *shapelet*

Searching for Motifs

The first step is already solved! We know how to extract a set of subseries $S_{\subset}$ from series representation. We need to quantify the descriptive and discriminative power of candidate motifs.

Descriptive power. Distance of $s^i_{\subset}$ with respect to all possible subseries of s^j yields distances $D_{i,S} = \{d^1_{i,j}, \dots, d^k_{i,j}\}$. Descriptive power given by lower distances, e.g., $\min D$

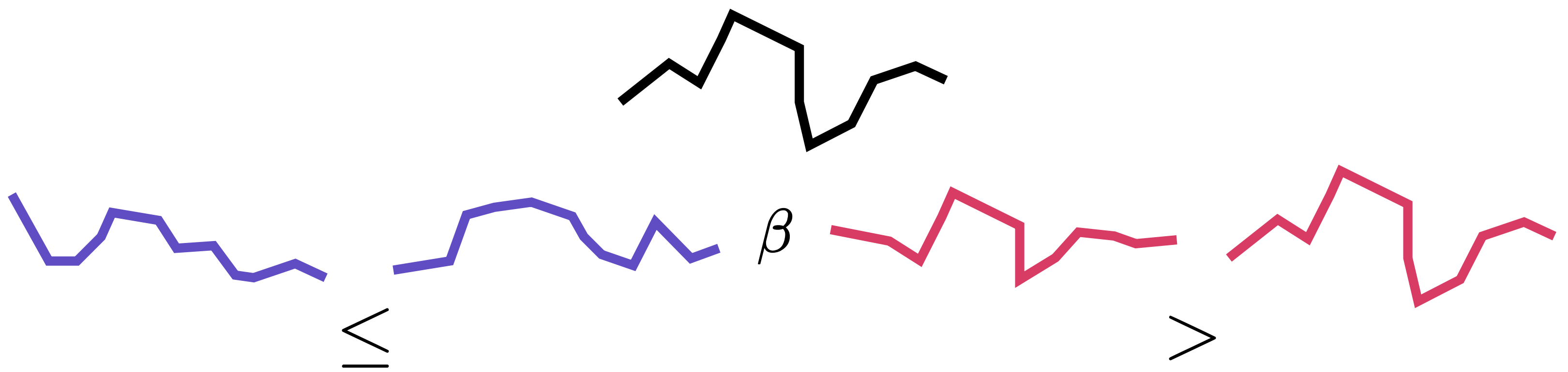
Discriminative power. Comparison of descriptive power with respect to S and $S^{\neq}$, e.g.,

$$\frac{\min D_{i,S}}{\min D_{i,S^{\neq}}}$$

From motifs to shapelets

Università di Pisa

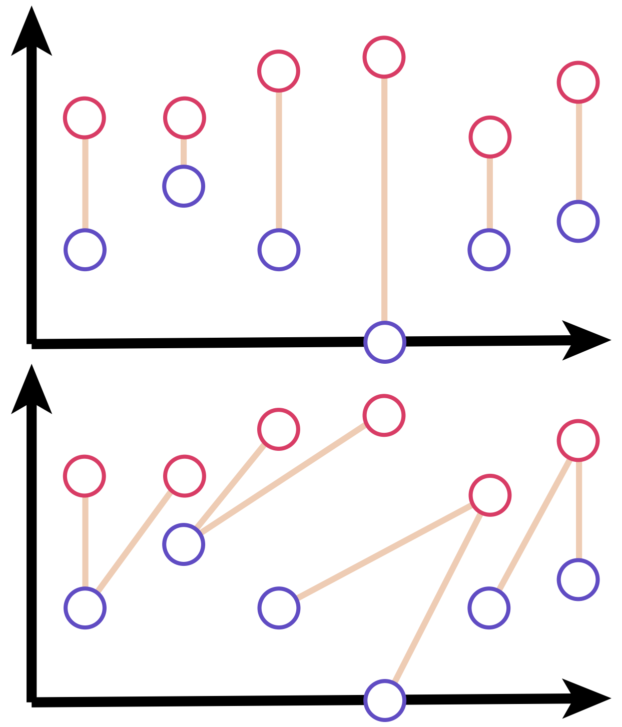
Since motifs are supposed to discriminate, why not directly measure their discriminative power? Partition subseries in $S \cup S^{\neq}$ according to their distance from a given candidate $s^i_{\subset}$ and a threshold β, obtaining two sets $S^{\leftarrow}_{\beta}, S^{\rightarrow}_{\beta}$. Then, compute a discrimination measure, e.g., entropy, information gain, etc., on the two sets $S^{\leftarrow}_{\beta}, S^{\rightarrow}_{\beta}$. The larger the measure, the higher the discriminative power!



A shapelet (in black), a threshold β, and the two sets $S^{\leftarrow}_{\beta}, S^{\rightarrow}_{\beta}$ (left and right, in blue and red, respectively).

Time series: alignment

An alignment $A = [(i, j)]^{\max n,m}$ of two time series l, u with components $[l_1, \dots, l_n], [u_1, \dots, u_m]$ is an assignment of each component l_i of l to a component u_j of u. An alignment A induces an alignment *cost* C_A quantifying how unaligned the two series are.



Two possible alignments (in beige) of an *upper* series s^1 (in red), and a *lower* series s^2 (in blue).

We are going to assume equal time sampling for both series.

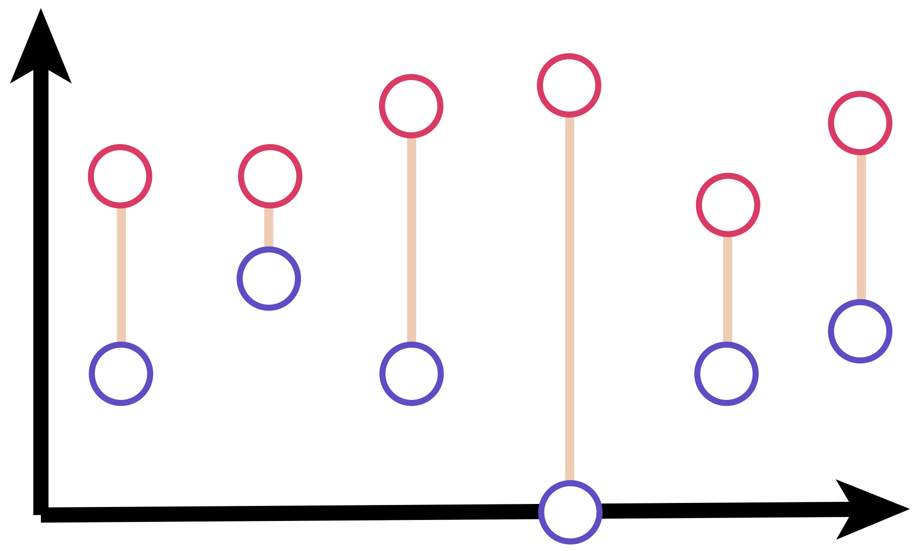
Alignment: local cost

Alignment costs $C.$ are based on two separate costs:

- *local* (component-wise, or point-wise) alignment cost $c^A_{i,j}$: defines the cost of aligning l_i with l_j. *How much do I pay for this alignment?*
- *match* alignment cost $c^\Sigma_{i,j}$: defines the cost of foregoing matching i with j, in favor of a lower-cost $i' \neq i$. *How much would I pay for another alignment?*

Alignment algorithms look to minimize a combined cost of the two.

Alignment: straight match



A straight match alignment.

Alignment: straight match

UNIVERSITÀ DI PISA

Trivial alignment: each point l_i is assigned to l_i. The alignment induces a pairwise alignment cost of $c^a_{i,i} =|| l_i - u_i ||^p$. The alignment cost C_A is given by the sum of the pairwise alignments:

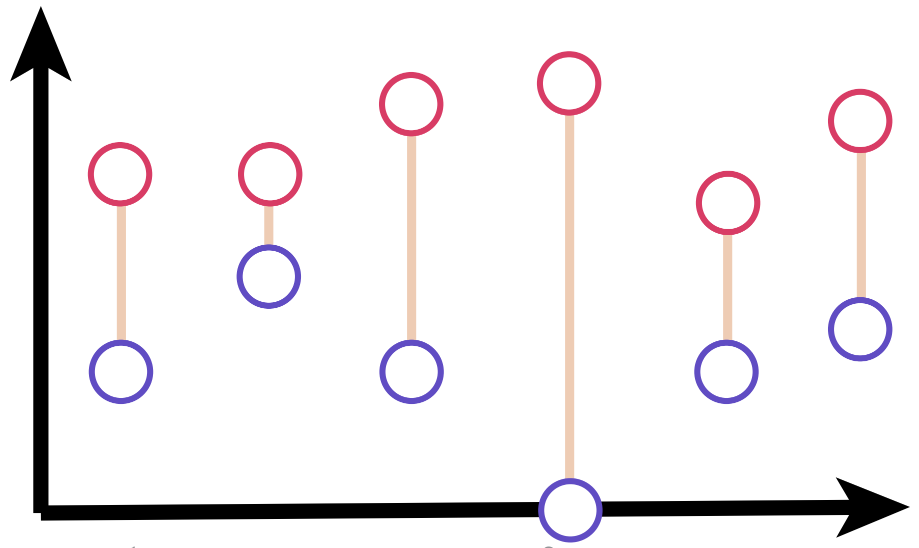
$$C_A = \sum_{i,i} c^a_{i,i} = \sum_i || l_i - u_i ||^p.$$

Since i is always aligned with j, it follows that $c^\Sigma_{i,j} = 0$. This produces an alignment $A = [(l_i, u_i)]^K$, only applicable for $m = n$.

Applicable to series with the same sampling rate!

Alignment: straight match

Università di Pisa



Two possible alignments of an *upper* series s^1, and a *lower* series s^2.

- Simple definition
- Minimizes norm cost
- Quick: $\mathcal{O}(n)$

- Only applicable to equal-length series
- Assumes alignment

Alignment: Shifted match

Università di Pisa

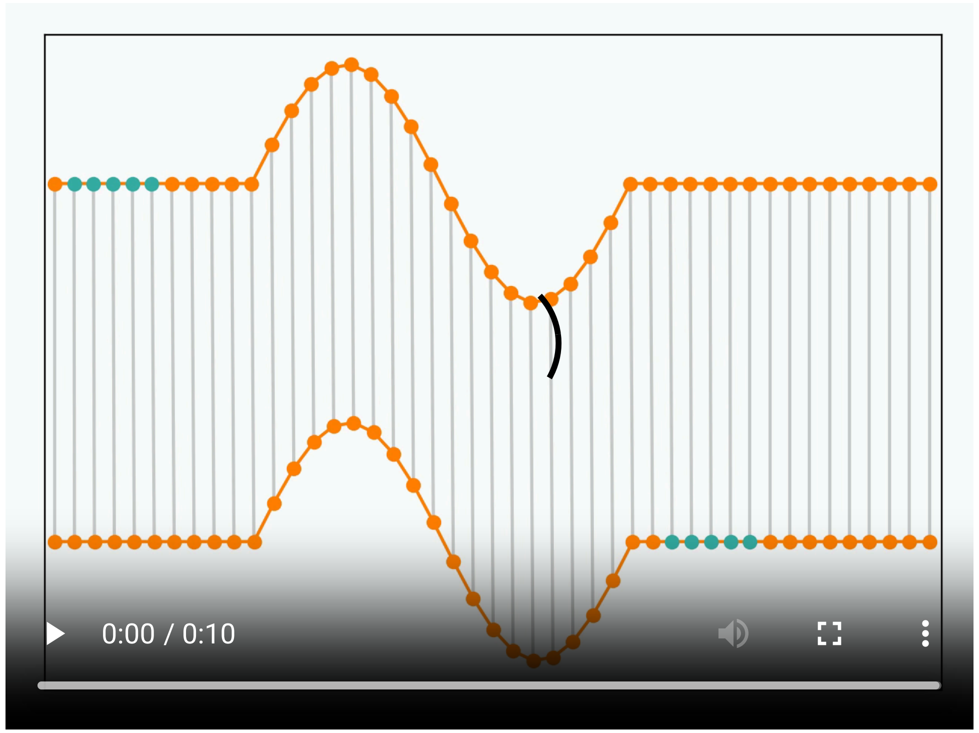
Not all time series are already aligned! Given a component l_i, we need to *look for* an aligning u_j, under a *shift assumption*.

Shifted alignment.
Assuming some subseries of either l, u are shifted, under a shifted alignment:

- **Warp, Replication.** Each component l_i can be assigned to any component u_j
- **Planarity.** Assignments are monotonic: a successive point cannot be assigned backwards: $\forall i, j. \, (l_i, u_j) \in A \implies (l_{i+1}, u_{j-1}) \notin A$

Alignment: Shifted match

If properly performed, a shifted match minimizes norm cost, and emulates a straight alignment... with additional aligned components.



A shifted match alignment as a straight match alignment.

Shifting as Warping

Warp, Replication. Each component l_i can be assigned to any component u_j.

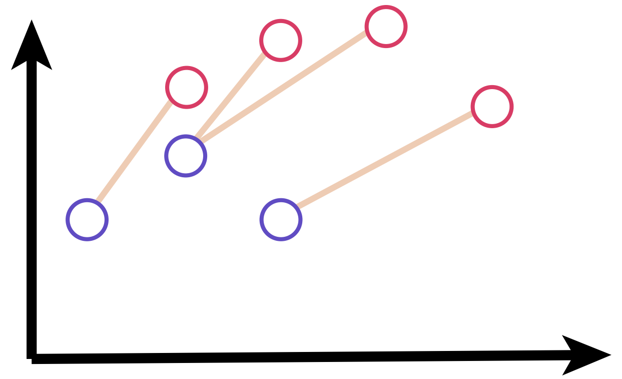
By *warping*, we replicate a component, warping it also further in the series. This allows us to replicate components, and emulate a straight match.

- *no warp*: the assignment (l_i, u_j) is attempted as is
- *lower warp*: the assignment (l_i, u_j) is attempted on (u_{i+1}, u_j), replicating the component on the lower series u
- *upper warp*: the assignment (l_i, u_j) is attempted on (l_i, u_{j+1}), replicating the component on the upper series v

Dynamic Time Warping (DTW)

DTW divides the alignment problem in two steps:

1. compute a cumulative minimal alignment cost matrix C^Σ, defining the minimal alignment cost of every pair of components l_i, u_j
2. search an alignment A on C^Σ, minimizing the alignment cost



$$C^\Sigma \begin{bmatrix} 0.495 & 0.613 & 0.429 & 0.618 \\ 0.524 & 0.350 & 0.536 & 0.885 \\ 0.458 & 0.214 & 0.511 & 0.194 \end{bmatrix}$$

A DTW alignment (top), and a cumulative alignment cost matrix C^Σ (bottom).
The (i, j) component holds the cumulative cost of aligning the l_i, u_j.

Σ stands for sum, unlike the alphabet used in SAX representation.

Warping in Dynamic Time Warping

C^{Σ} computes a *minimal* and *cumulative*: $c^{\Sigma}_{i,j}$ may be $\neq 0$!

- base case: the alignment (l_1, u_1) has minimal cost. This is trivially true, since there are no accumulated costs, i.e., $C^{\Sigma}_{1,1} = || \ l_1 - u_1 \ ||^p$
- inductive case: i, j fall within the three warping categories, thus it must be one of three cases
  - *(no warp)* i, j: the accumulated cost is $c^{\Sigma}_{i,j} = C^{\Sigma}_{i-1,j-1}$
  - *(lower warp)* i has warped: the accumulated cost is $c^{\Sigma}_{i,j} = C^{\Sigma}_{i-1,j}$
  - *(upper warp)* j has warped: the accumulated cost is $c^{\Sigma}_{i,j} = C^{\Sigma}_{i,j-1}$

Computing warp cost

Entries in C^Σ are computed as

$$\begin{aligned} C^\Sigma_{i,j} &= c^a_{i,j} + c^\Sigma_{i,j} \\ &= \underbrace{\| \, l_i - u_j \, \|^p}_{alignment\ cost} + \min\{\underbrace{C^\Sigma_{i-1,j-1}}_{no\ warp}, \underbrace{C^\Sigma_{i,j-1}}_{lower\ warp}, \underbrace{C^\Sigma_{i-1,j}}_{upper\ warp}\} \end{aligned}$$

$$C^\Sigma \begin{bmatrix} 0.495 & 0.613 & 0.429 & 0.618 \\ 0.524 & 0.350 & 0.536 & 0.885 \\ 0.458 & 0.214 & 0.511 & 0.194 \end{bmatrix}$$

$$C^\leftarrow = \begin{bmatrix} - & \leftarrow & \leftarrow & \leftarrow \\ \uparrow & \nwarrow & \uparrow & \leftarrow \\ \uparrow & \uparrow & \leftarrow & \leftarrow \end{bmatrix}$$

Directions of minimal accumulated cost $C^\leftarrow$ over C^Σ: entries indicate which alignment choice has produced the minimal cost.

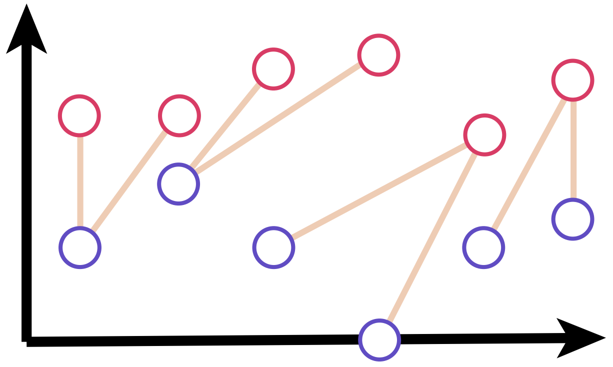
Searching for the lowest cost alignment

As a matrix of minimal accumulated alignment cost, we know that for each component i, j in C^{Σ}, we have, by construction, the minimal cost to align up to i, j.
Thus, we can simply start from the last alignment, and follow $C^{\leftarrow}$ backwards for the warps of minimal cost!

$$C^{\leftarrow} = \begin{bmatrix} - & \leftarrow & \leftarrow & \leftarrow \\ \uparrow & \nwarrow & \uparrow & \leftarrow \\ \uparrow & \uparrow & \leftarrow & \leftarrow \end{bmatrix}, A = [(1,1), (2,2), (3,2), (3,3), (3,4)]$$

Directions of minimal accumulated cost $C^{\leftarrow}$ over C^{Σ}: the red path from the last entry indicates induces the alignment of minimal cost. The alignment A is given by the indices of the path.

Dynamic Time Warping


A Dynamic Time Warping alignment.

- Minimizes norm cost on shifted series

- High cost of $\mathcal{O}(mn)$

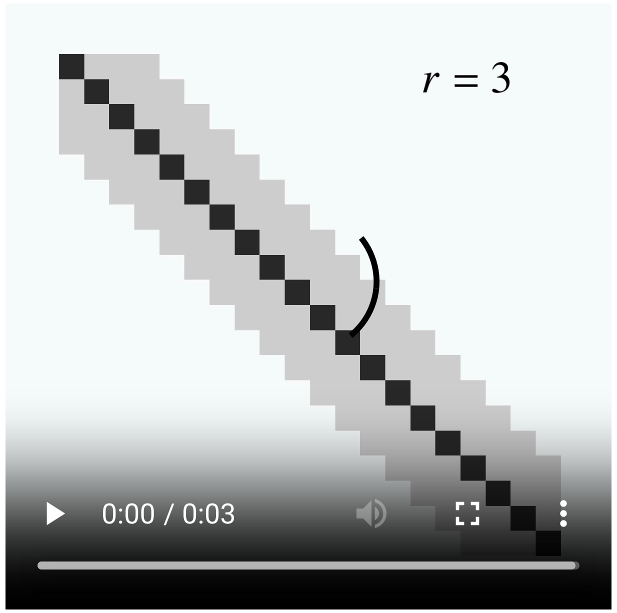
Constraining alignments

Within C^{Σ}, we have several possible movements:

- alongside a row (i is constant): indicate a sequence of upper warps, as we are aligning one component of l with consecutive components of u
- alongside a column (j is constant): indicate a sequence of lower warps, as we are aligning one component of u with consecutive components of l
- diagonally (neither i nor j is constant): indicate a no-warp alignment

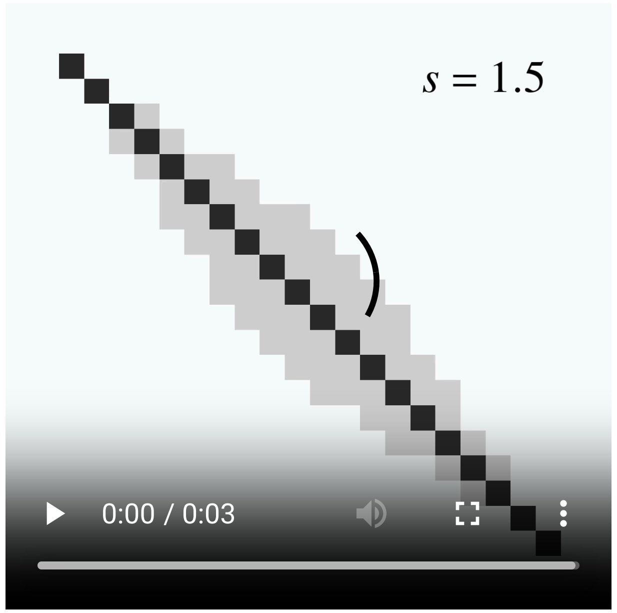
Sakoe-Chiba search

The optimal search we have highlighted can move almost arbitrarily over C^{Σ}, inducing arbitrarily large row- and column-segments, and thus arbitrarily large warps.
Sakoe-Chiba search instead constraints the alignment A to be such that $\forall i, j \in A. \mid i - j \mid < \gamma$ by introducing a search radius γ, and binding the maximum warp size of either series.



Itakura parallelogram

The Itakura constraint instead binds the slope of segments, effectively binding consecutive warps.
For any two $(i, j), (k, l) \in A.\, k \geq i, l \geq j.\ \frac{l-j}{k-i} < \alpha$.



From alignment to similarity

Time series: similarity

Similarity	Similar if...	Measure	Sensitive to
Straight match	Similar values	Norm (Euclidean)	Time shifts
Dynamic Time Warping	Similar (interleaved) shapes	Norm	
Autocovariance	Seasonal	Inner product	Amplitude shifts
Statistical	Similar statistics	Mean, Variance, etc.	

References

Topic	Reference
Fourier Transform	Digital Signals Theory e-book, Chapters 1, 2, 4, 5, 9
Wavelet Transform	Data-driven science and engineering. By S. L. Brunton, J. N. Kutz, 2nd edition. 2.5
Shapelets	Time series shapelets: a new primitive for data mining
Dynamic Time Warping	An introduction to Dynamic Time Warping

Credits for some images and animations to Romain Tavenard.