# Different Similarity Measures

$$f^p, f^q : timeseries$$

▼ Manhattan

- Quantifies the absolute magnitude of the difference between time series
- Easy to calculate

$$D_{Man} = (\sum_{t=1}^{N} |f_t^p - f_t^q|)$$

- Implementation: scipy.spatial.distance.cityblock

▼ Euclidean

- Quantifies the Euclidean distance ofthe difference between time series
- Easy to calculate
- More sensitive to outliers, due to it's nonlinear character

$$D_E = (\sum_{t=1}^{N} |f_t^p - f_t^q|^2)^{\frac{1}{2}}$$

- Implementation: scipy.spatial.distance.euclidean

▼ Mahalanobis

- Quantifies the difference between time series but accounts for non-stationarity of variance and temporalcross-correlation

$$D_{Mah} = \sqrt{(f_t^p - f_t^q)^T \sum^{-1} (f_t^p - f_t^q)}$$

$$\sum : CovarianceMatrix$$

- Implementation: scipy.spatial.distance.mahalanobis

▼ Pearons's Correlation

- Quantifies the degree of linear relationship between time series

$$D_{CC} = \frac{\sum_{t=0}^{N-1}\left[\left(f_t^p - \bar{f}^p\right) * \left(f_{t-s}^q - \bar{f}^q\right)\right]}{\sqrt{\sum_{t=0}^{N-1}\left(f_t^p - \bar{f}^p\right)^2} * \sqrt{\sum_{t=0}^{N-1}\left(f_{t-s}^q - \bar{f}^q\right)^2}}$$

- Implementation: numpy.corrcoef

▼ Principal Component

- Quantifies the difference between times PCs that explain the majority of the variance
- Selecting m critical

$$D_{PCA} = \sqrt{\sum_{k=1}^{m} (PC_k^p - PC_k^q)^2}$$

- Implementation: sklearn.decomposition.PCA

▼ 3 Fourier Based Similarities

- Fourier Transformation of series f(t)

$$f(t) = A_0 + \sum_{k=1}^{N-1} A_k \cos(2\pi kt + \phi_k)$$

with

$$A_k = \sqrt{F_k^{c2} + F_k^{s2}}$$

and

$$\phi_k = \arctan\left(\frac{F_k^c}{F_k^s}\right)$$

## Quantifies the combined effect of FT amplitude and phase differences

$$D_{FFT} = \sqrt{\sum_{k=0}^{m}(A_k^p - A_k^q)^2 + \sum_{k=1}^{m}(\phi_k^p - \phi_k^q)^2}$$

## Quantifies shape similarity based onderived FT amplitude and phase differences

$$D_\xi = 3\sqrt{\sum_{k=1}^{m}\left(\alpha_k^{ref} - \alpha_k\right)^2} + \sqrt{\sum_{k=1}^{m}\left(\theta_k^{ref} - \theta_k\right)^2} \tag{9}$$

where $\alpha_k$ and $\theta_k$ are the relative amplitude and phase for each FT component:

$$\alpha_k = \frac{A_k}{A_1} \tag{10}$$

and

$$\theta_k = \left(\frac{A_k}{A_1}\right)_{ref}[2 + \cos(k\phi_1 - \phi_k)] \tag{11}$$

and α_k^ref and θ_k^ref are relative amplitude and phase for a reference class. D_ξ is zero if two time series, represented by their m Fourier components, have the same shape and increases as the differences between the shapes increases.

## Quantifies the FT of the difference between time series

$$
\begin{aligned}
A_k^{p-q} &= \sqrt{\left(F_k^c(p) - F_k^c(q)\right)^2 + \left(F_k^s(p) - F_k^s(q)\right)^2} \\
&= \sqrt{F_k^c(p-q)^2 + F_k^s(p-q)^2}
\end{aligned}
\tag{12}
$$

Based on the $A_k^{p-q}$ of the selected $m$ FT components, $D_{Fk}$ can be calculated:

$$
D_{Fk} = \sum_{k=0}^{N-1} w_k A_k^{p-q} \tag{13}
$$

where A_k^p−q is the F_k-distance between the time series of p and q respectively, and w_k is the weight of the kth frequency FT component. Modification of the weights allows to enhance (high w_k) or diminish(low w_k) the influence of each component on D_Fk and to accentuate specific components in the similarity measure. As a result, these weights will determine the sensitivity to time series differences. For example, when w_0=0 the D_Fk will be insensitive to amplitude translations as the offset value in the y-axis is neglected in the D_Fk calculation

- Implementation: numpy.fft

▼ Mutual Information

*Mutual Information.* Introduced by Shannon [26], the notion of *entropy* is a measure for the expected information from observing the value of a random variable $X$, noted as $H(X)$. The expected information for observed values of two random variables $X$ and $Y$ is the natural extension *joint entropy* $H(X, Y)$. This gives way to the notion of *Mutual Information*

$$
I(X;Y) = H(X) + H(Y) - H(X,Y), \tag{1}
$$

which describes the information shared between both variables. Using the definition of entropy for continuous random variables in Equation 1 yields the differential definition of MI

$$
I(X;Y) = \int_Y \int_X p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) dx\, dy \tag{2}
$$

where $p(x)$, $p(y)$ and $p(x,y)$ are the marginal and joint probability density functions of $X$ and $Y$, respectively [8]. Using the natural logarithm, MI is then measured in the *natural unit of information* (nat).

- Implementation: sklearn.feature_selection.mutual_info_classif