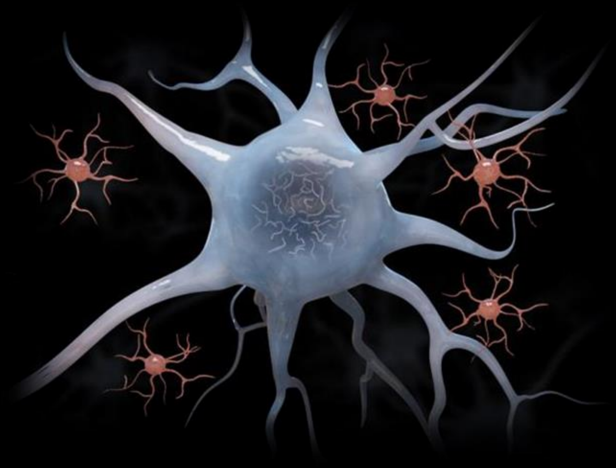
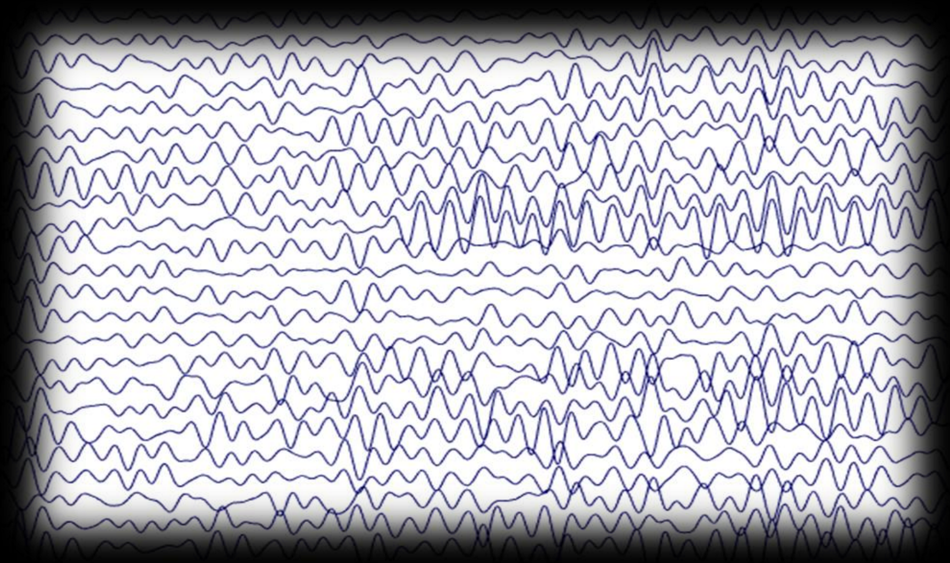
Visual explanation of some digital processing and statistical methods for neurophysiological data

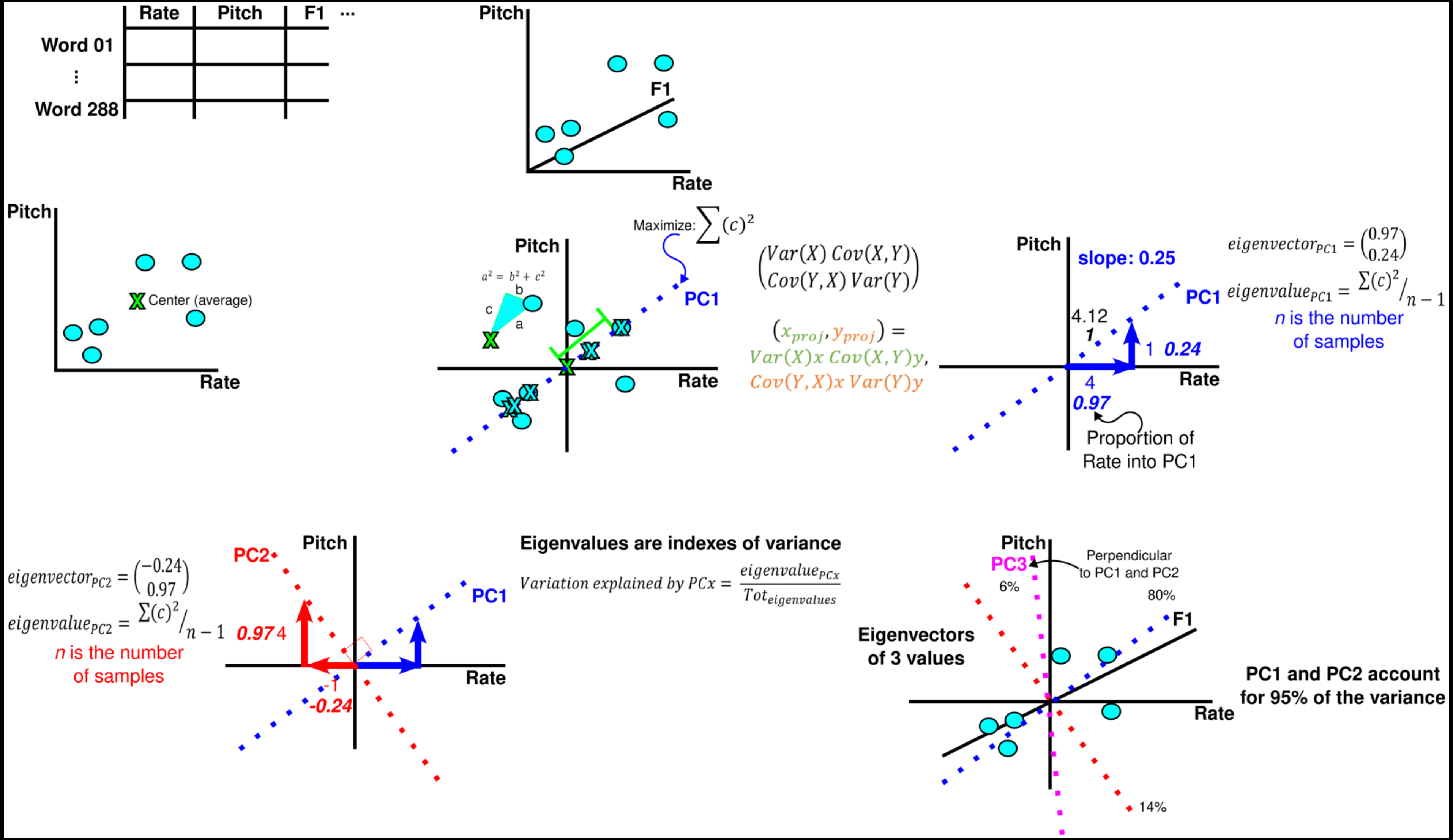


By

Mathilde Marie Duville, PhD



Principal Component Analysis (PCA)

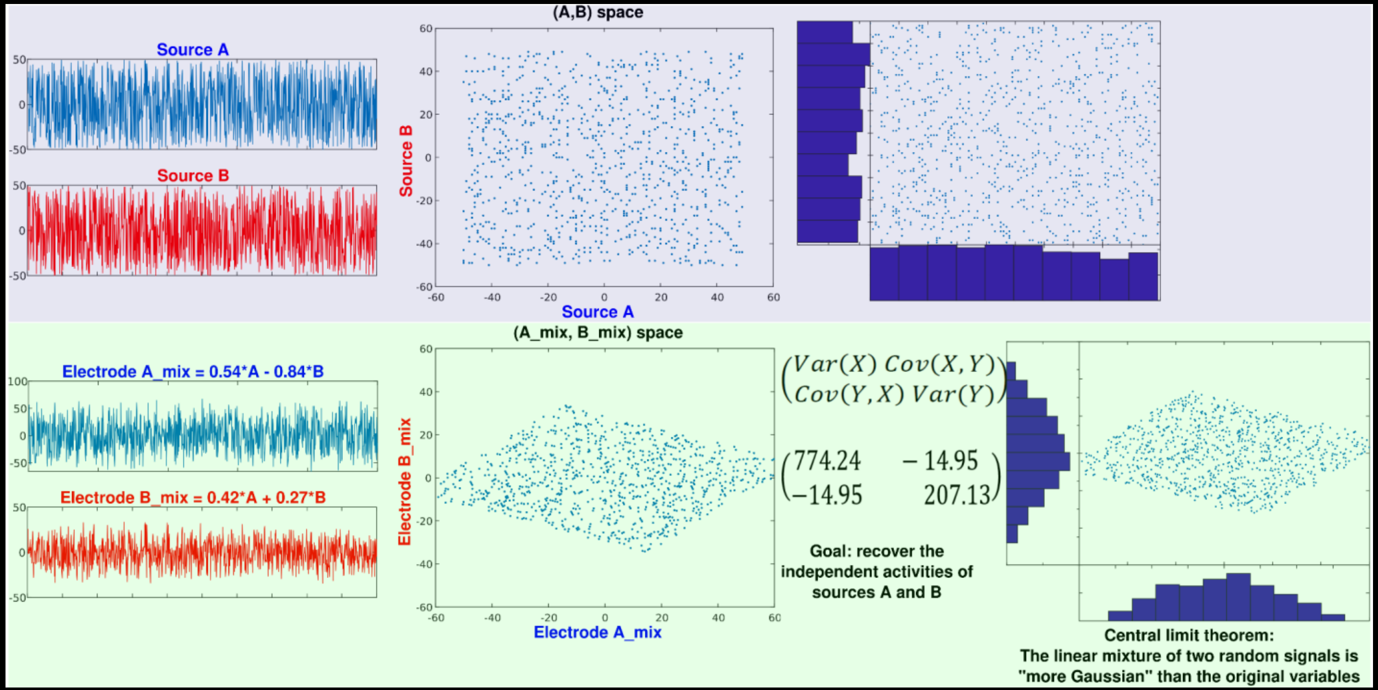


$$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n-1}$$

where x_i is the observation of value i,
$\bar{x}$ is the mean over all observations,
n is the number of observations.

1

Independent Component Analysis (ICA)

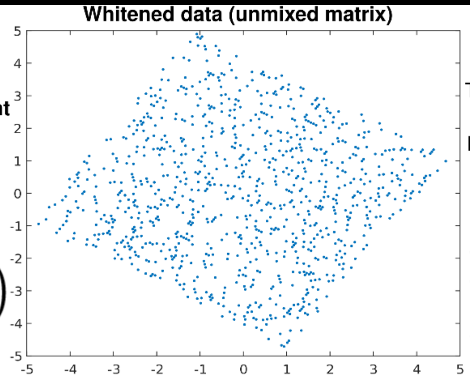


https://arnauddelorme.com/ica_for_dummies/

Independent Component Analysis (ICA)

Linear change of coordinate of the mixed data
The variance of both axis is equal
The covariance is null: signals are now independent

$$\begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}$$



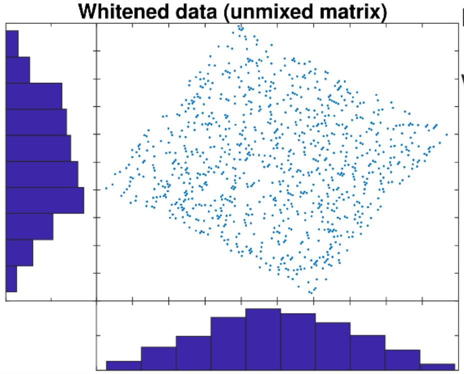
Here is an example with 2 dimensions
In practice we have n_electrodes dimensions
ICA would rotate the n_electrodes dimensions after whitening
The goal is to **minimize the Gaussian distribution of the projection on all axis**

ICA components are the matrix used to rotate the data from the initial space to the ICA space

$$S = WX$$

where S is the activity of the sources (components x time),
W is the unmixing matrix (components x electrodes),
X is the EEG data (electrode x time)

The projections on both axis follow a Gaussian distribution



Projection of components to the electrode space:

$$X = W^{-1}S$$

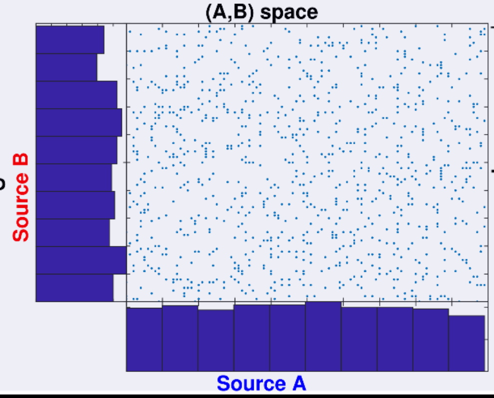
where W^{-1} is the inverse matrix of W (electrodes x components)

=

$$X(elec, time) = \sum_{k=1}^{nb\_comp} W^{-1}(elec,k)\, S(k,time)$$

Example:
Electrode 1, sample 1 (2 components):
(W-1(Elec1/comp1)*S(comp1/sample1))+
(W-1(Elec1/comp2)*S(comp2/sample1))

Rotation back onto the (A,B) space



To isolate the EEG activity corresponding to a component:

$$X_i = W_i^{-1} S_i$$

X_i is the EEG activity of corresponding to component i (electrodes x time)
W_i^{-1} is the ICA mixing coefficient of component i (electrodes x 1)
S_i is the source activity of component i (1 x time)

To remove the activity of component i (i.e., X_i), we can subtract (X_i) to X.

https://arnauddelorme.com/ica_for_dummies/

Wavelet-ICA, ICLabel, and MARA (Multiple Artifact Rejection Algorithm)

w-ICA

In practice, the activity of source *i* may still contain a mixture of artifact and neuronal activities.

$$S_i(t) = a(t) + n(t)$$

where $a(t)$ is the artifact activity through samples (t), and $n(t)$ is then neuronal activity.

We can estimate the properties of signals $a(t)$ and $n(t)$ by Wavelet decomposition to ensure frequency precision in time. Constant-source artifact data are usually of high amplitude and localized in time and frequency.

w-ICA:

1. ICA decomposition to obtain the source activity (S) of N components and the unmixing matrix (W).
2. Wavelet transformation to obtain wavelet coefficients (time-frequency decomposition).
3. Threshold the wavelet coefficients to set at 0 those higher than the threshold.
4. Inverse wavelet transform to recover only the neuronal information of ICA components $(n_i(t))$.
5. Compose wICA-corrected EEG: $\tilde{X} = W^{-1}[n_1(t), \ldots n_N(t)]$

Threshold definition:

$$Threshold = \frac{median[abs(D)]}{0.6745}\sqrt{2\log(T)}$$

where D are the wavelet coefficients and T is the length of the ICA components.

ICLabel

IC components classifier

1. Training set: 6352 EEG recordings (203 307 IC components)

2. Scalp topography (unmixing matrix), PSD, autocorrelation function, equivalent current dipoles model fits

3. ICLabel website: collaborative labeling (constant updating)

4. Crowd labeling: estimation of a single “true label” given redundant labels provided by various labelers

5. Expert labeling for the validation set (more accurate reference for final accuracy)

6. Supervised learning: deep neural network (~85% accuracy)

7. Output: probability (%) for each class
Brain, eye, muscle, heart, line noise, channel noise, other

MARA

IC components classifier

NOT updated (2014)

1. EEG data:
- Driving simulation: 43 sets of 10 min
- Audiovisual story listening: 71 sets of 3.77 min
- Auditory speech listening: 18 sets of 30 min
- Motor imagery: 8000 sets of 8 seconds

2. Current density, max-min in time series,
local skewness over 15s windows,
average power of alpha (8-13Hz),

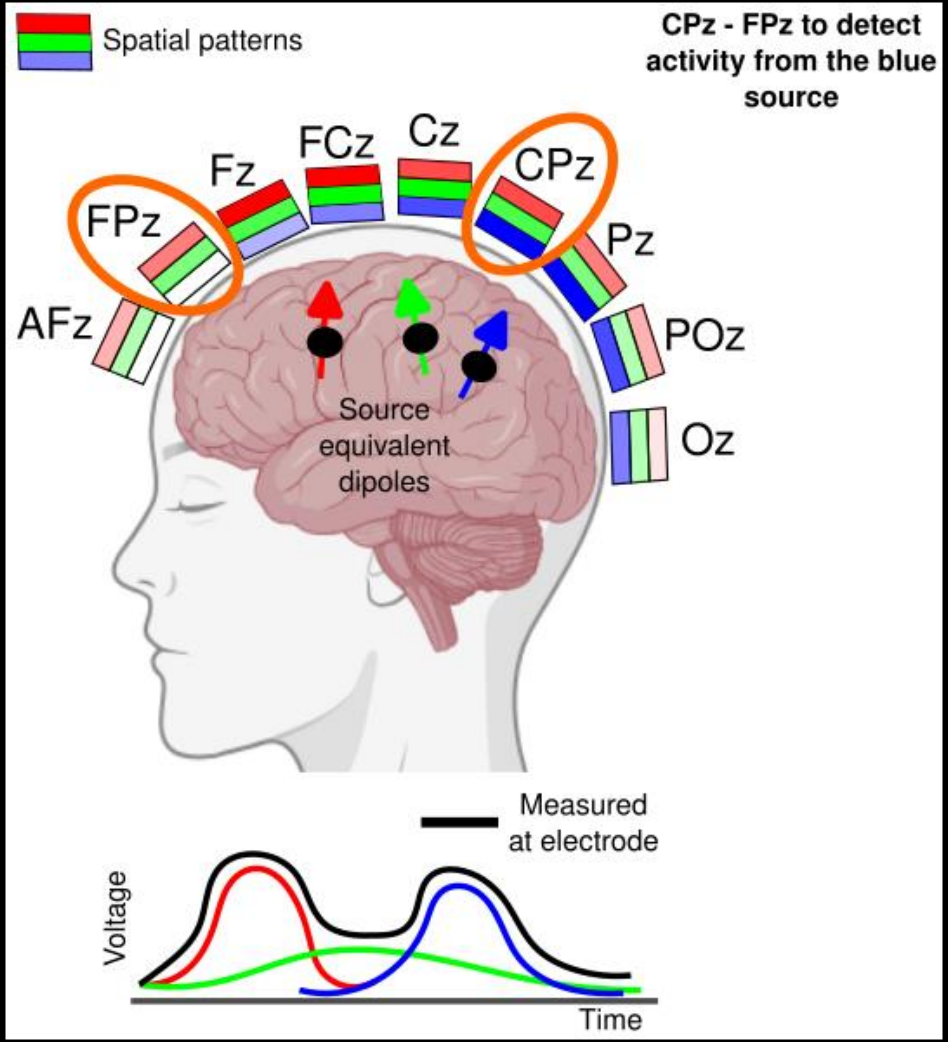
3. Expert labeling

4. Supervised learning: binary linear classifier
Training on a dataset, validation on the others

5. Output: artifact/neuronal (~70% accuracy)

2-4

Single-Trial Event-Related Potentials: Linear Discriminant Analysis Beamformer



An ERP per trial (utterance)
No average (like in the traditional method)
Allows the posterior validation of the coherence between words of same emotion

Estimates of the sources are formed by applying a **spatial filter** to the data, which is a **linear combination of sensor data.**

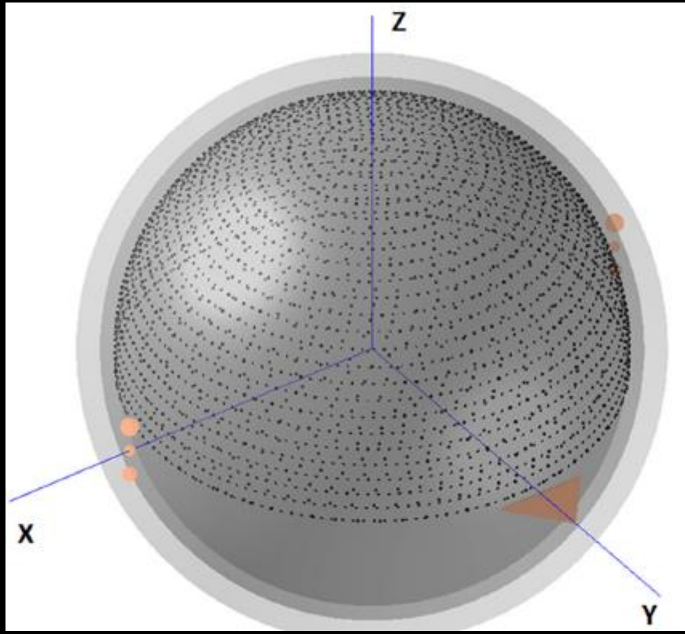
$$x(t) = P\,S(t) + \varepsilon = (P_1\,P_2 \dots P_M)(S_1(t)\,S_2(t) \dots S_M(t)) + \varepsilon$$

where $x(t)$ are EEG data across time t,
P is the spatial pattern, M are brain regions, where P is a N (channels) x M matrix that specify **how the activity of the sources propagates towards the sensors**,
$S(t)$ is the source activity in M brain regions, where $S(t) = W^T x(t)$, with W^T the **spatial filter (inverse model)**
ε is the error term (noise and artifacts)

5

Reference Electrode Standardization Technique (REST) re-referenciation

Forward model: how the activity of a neuronal source propagates to the sensors, **independently of the reference**
The temporal evolution of the reference signal should be similar to the one of neighboring channels (similar source, temporal dependency). The potential at infinity is ideal since it is the farthest from neural sources.S



Standardized head model to compute G

$$V_{REST} = G * \boxed{S}$$

where V_{REST} is the EEG activity re-referenced to REST (n sensors*k samples),
G is the lead field matrix (n*y*x*z or n*m neural sources). Reflects the geometrical and conductive properties of the head.
S is the equivalent current source density (source model; m*k or y*x*z*k)

For an earlobe online reference:

$$V_e = V_{REST} - v_e = GS - g_e S = (G - g_e)\, S = G_e S \longrightarrow G_e = G - g_e$$

where V_e is the EEG activity referenced to the earlobe, v_e is the row vector in V_{REST} corresponding to the reference electrode, g_e is the row vector (1 x m) corresponding to the reference electrode in G, and G_e is the lead field matrix with the earlobe as reference.

However, the reference does not influence the source localization, therefore:

$$\boxed{\hat{S}} = G_e^{+} V_e = G_{AR}^{+} V_{AR}$$

where $\hat{S}$ is the estimate of reconstructed equivalent sources, G_e^{+} and G_{AR}^{+} are the Moore-Penrose generalized inverses of matrices G_e and G_{AR} (lead field matrix with the average reference), and V_{AR} is the EEG activity average referenced.

6-7

Artifact Subspace Reconstruction (ASR)

1. Extract the reference data (clean)

I. Compute Root Mean Square (RMS) values of each channel over 1s windows:

$$RMS = \sqrt{\frac{1}{n}\sum_i x_i^2}$$

where i is the index of every sample x, and n is the total number of samples.

II. Compute the Z-score across all windows:

$$Z = \frac{x-\mu}{\sigma}$$

where μ is the average and σ is the standard deviation.

III. Identify clean windows: Z-score between -3.5 and 5.5 and concatenate the windows to obtain the reference data (X_r).

2. Determine thresholds to identify artifacts components.

I. Apply a IIR filter to X_r to suppress frequency bands typically associated with brain oscillations, obtaining $\widetilde{X_r}$.

II. Compute:

a. the mixing matrix M_r, that is the square root of the of $Cov(\widetilde{X_r})$

b. the eigenvalue decomposition of M_r to obtain the eigenvectors matrix V_r, and eigenvalues vector D_r. Each column in V_r is the eigenvector corresponding to the eigenvalue in D_r.

c. Projection into the Principal Component (PC) space $\widetilde{Y_r} = V_r^T\,\widetilde{X_r}$. Compute the mean μ_i and standard deviation σ_i of the RMS values across all 0.5s windows of $\widetilde{Y_r}$ for each component i.

d. **Define the rejection threshold:** $\Gamma_i = \mu_i + k\,\sigma_i$, **where k is defined by the user.**

3. Reject artifacts component and reconstruct the data.

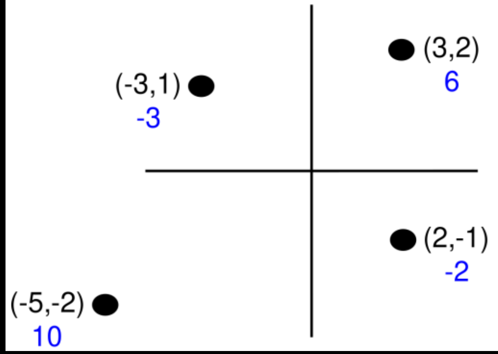
I. Compute the eigenvalue decomposition to the covariance matrix taken across the channels of the IIR-filtered **uncleaned data** ($Cov(\widetilde{X_t}) = V_t D_t V_t^T$) along a sliding window of size 0.5s and step size of 0.25s.

II. For each window, identify whether the j^{th} PC $(V_t)_j$ of variance $(D_t)_j$ is larger than the rejection threshold Γ_i projected from V_r onto V_t: $(D_t)_j > \sum_i(\Gamma_i (V_r)_i^T (V_t)_j)^2$. If this inequality holds, then the value of the j^{th} PC are replaced by zeros: $(V_t^T M_r)_{trunc}$.

III. Cleaned data may be reconstructed using the following equation:

$$(X_t)_{clean} = M_r\,(V_t^T M_r)^{+}_{trunc}\,V_t^T X_t$$

where + represents the Moore-Penrose generalized inverse matrix



Product of coordinates

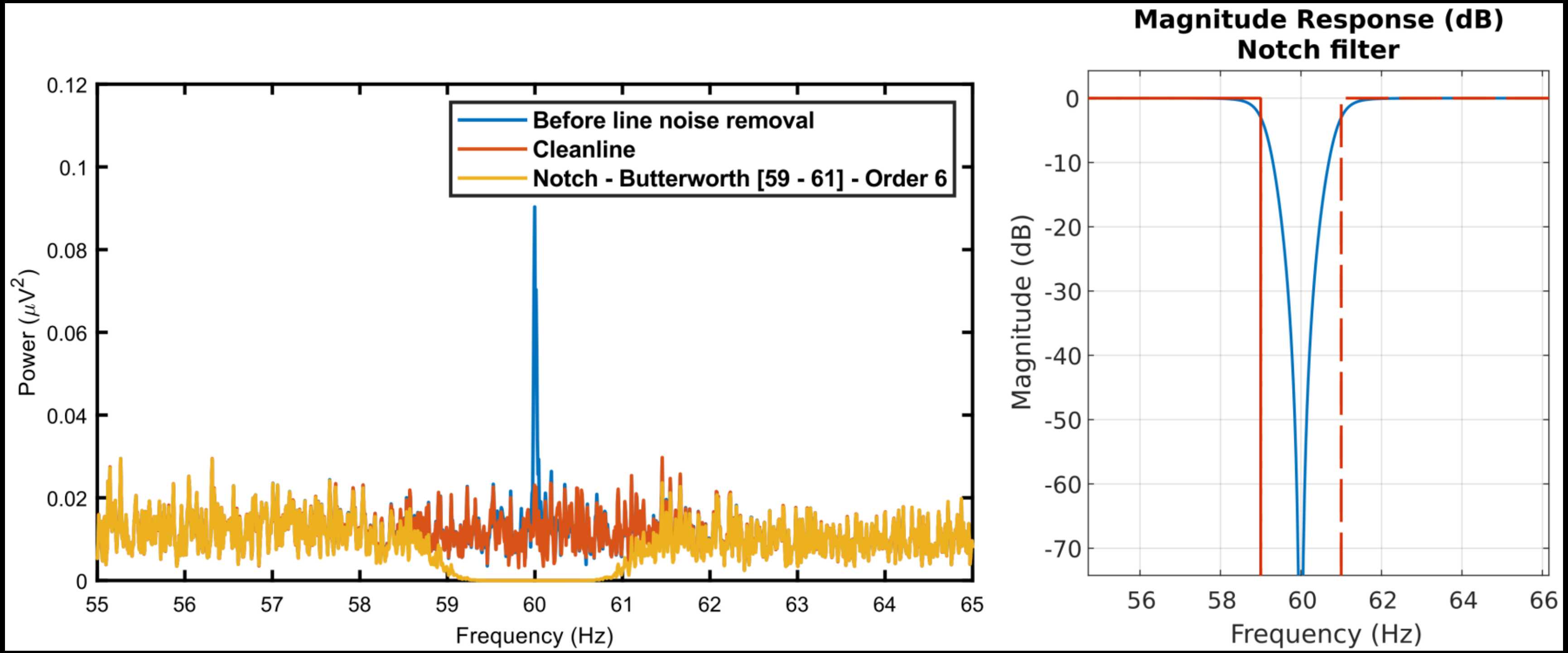
covariance = (-3+6+10-2)/4 = 3.75

covariance matrix

$$\begin{pmatrix} Var(x) & Cov(x,y) \\ Cov(y,x) & Var(y) \end{pmatrix}$$

8

Line noise removal



Cleanline:

- Frequency domain (multitaper FFT)
- Regression of the complex spectrum of a sinusoidal signal (e.g., 60 Hz) onto the spectrum of the original data. The regression coefficient is a complex number representing the phase and amplitude of the deterministic sinusoid (i.e., how much the original data can explain the 60 Hz sinusoid). **From this, a time-domain representation of the sinusoid may be constructed and subtracted from the data to remove the line.**
- Determine the statistical significance of a non-zero regression coefficient within a range of frequencies near 60 Hz to remove the more accurate line noise (maximize the test statistic).
- **The temporal and spectral smoothing from multitapers avoids the “band-hole” effect of notch filtering.**

9
https://sccn.ucsd.edu/wiki/Plugin_list_all
https://www.nitrc.org/projects/cleanline

Superfast spherical spline interpolation

$$\begin{cases} G_{good}\, C + C_0 = V_{good} \\ \sum_{i=1}^{N-k} C_i = 0 \\ V_{SSI}^{k_l} = \sum_{i=1}^{N-k} (C\, G_{sph}^{k_l}) + c_0 \end{cases}$$

where V_{good} is the activity of the good channels of size $(N\ channels - k\ bad\ channels) * T\ samples$, G_{good} are the Legendre polynomials of the cosine of the angle between the good channels projected into a unit sphere (size: $(N-k)*(N-k)$), C of size $(N-k)*T$ is an unknown term that need to be solved, C_0 of size $(N-k)*T$ is the constant of V_{good} defined by:

$C_0 = \begin{bmatrix} 1 \\ . \\ . \\ . \\ 1 \end{bmatrix} c_0$ where c_0 is the average vector $(1*T)$ of V_{good} across channels, $V_{SSI}^{k_l}$ is a vector of size $1*T$ that is the activity of the k_lth interpolated channel, and $G_{sph}^{k_l}$ are the Legendre polynomials of the cosine of the angle between the k_lth bad channels and the good channels projected into a unit sphere (size: $(N-k)*T$).

Parameters *m* and *n* of spherical splines

Spherical splines assume the potential at any point to be:

$$V(\vec{r}) = C_0 + \sum_{l=1}^{N_e} C_i g_m(\cos(\vec{r}, \vec{r_l}))$$

Where $\vec{r_l}$ is the location of a sensor with $l = 1, \dots, N_e$, $\vec{r}$ denote the position of an arbitrary point, C_i is the constant fit to the data, c_0 is the average of V_{good} across channels, $g_m(x)$ is the function of the cosine of the angle between the interpolation point $(\vec{r_l})$ and an electrode location $(\vec{r})$, given by:

$$g_m(x) = \frac{1}{4\pi} \sum_{n=1}^{\infty} \frac{2n+1}{(n+(n+1))^m} P_n(x)$$

where $P_n(x)$ is the Legendre polynomial of order n.

Legendre polynomials are differential equations defined between -1 and 1 by:

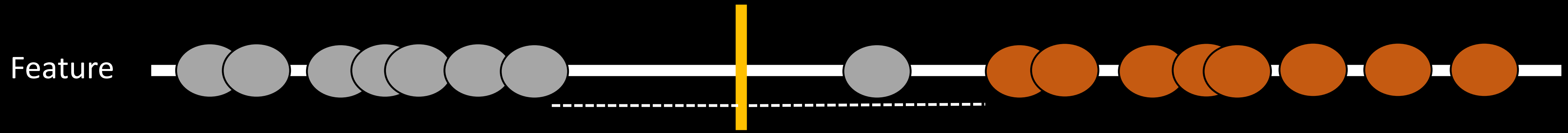
$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) + nP_{n-1}(x)$$

By increasing *m, gm(x)* converges more rapidly, leading to a smoother interpolation curve

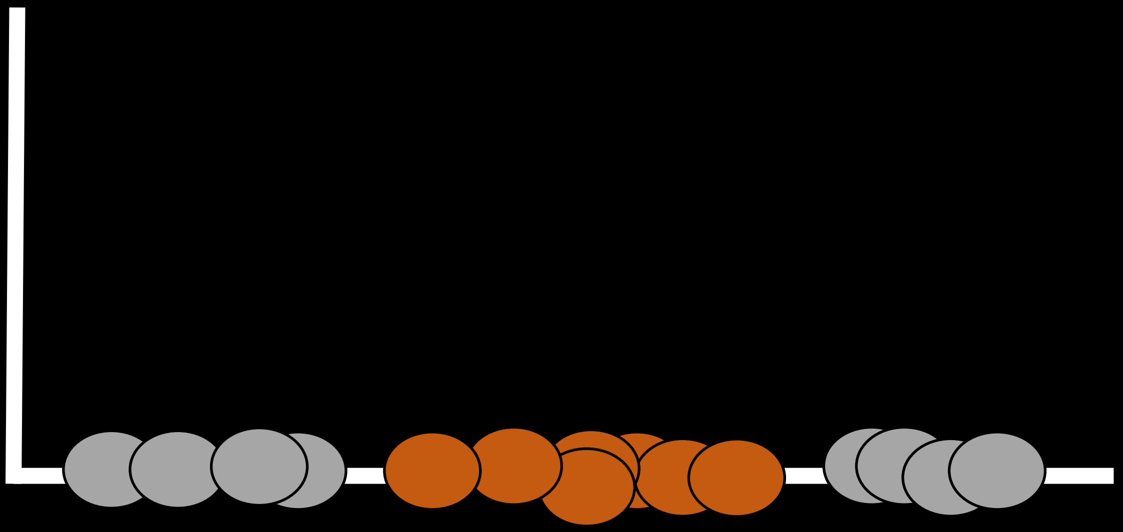
n are the terms of the polynomials

10

Support Vector Machines (SVM)



Soft Margin: high bias/low variance/allows outliers

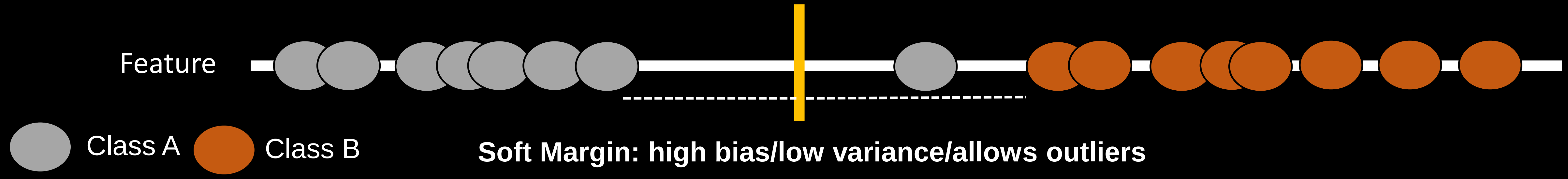


Rescaling [0;1]

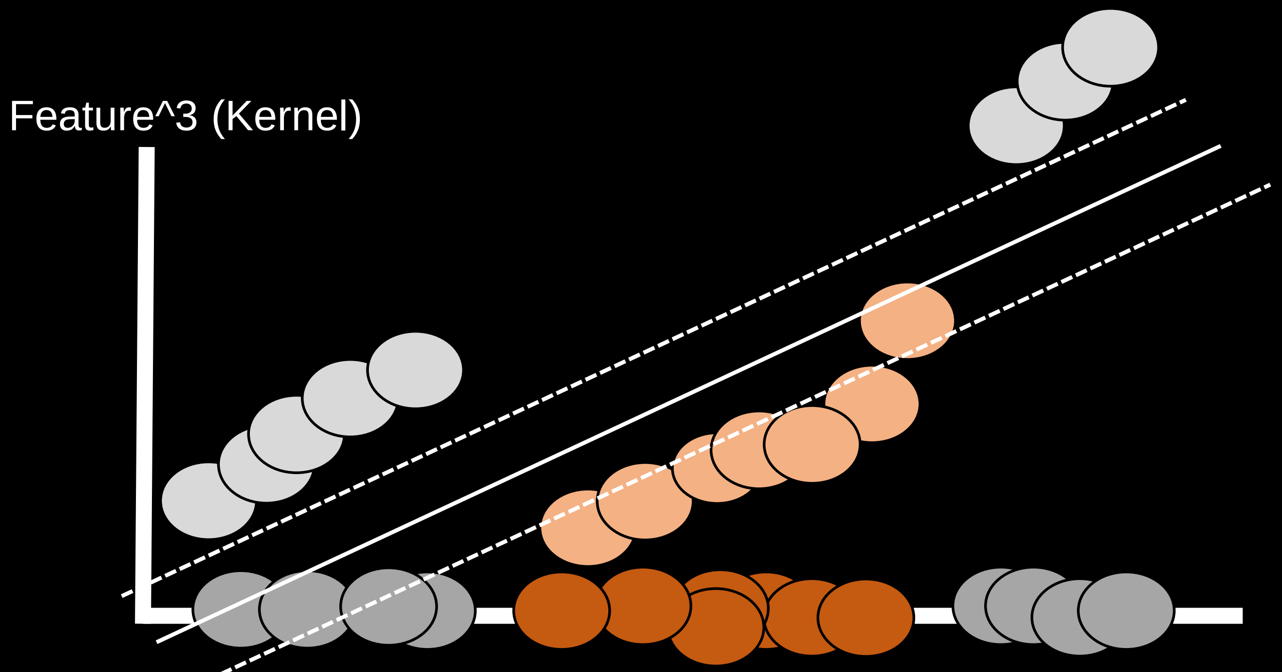
$$x' = a + \frac{(x - min(x))(b - a)}{\max(x) - \min(x)}$$

x' is in range [a;b]

Support Vector Machines (SVM)



Soft Margin: high bias/low variance/allows outliers



Rescaling [0;1]

$$x' = a + \frac{(x - min(x))(b - a)}{\max(x) - \min(x)}$$

x' is in range [a;b]

Supervised learning performance

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

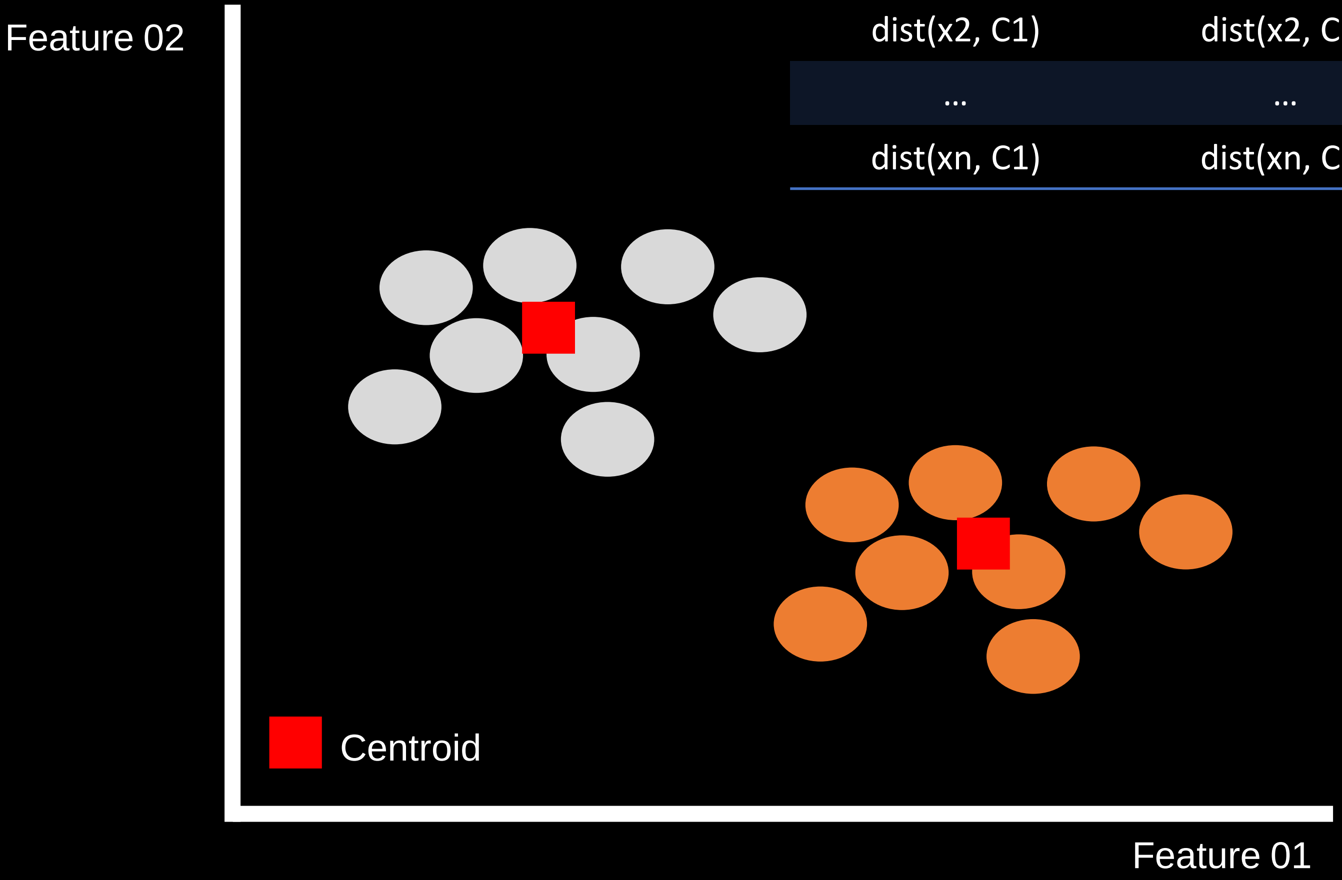
$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - measure = \frac{2\ X\ Precision\ X\ Recall}{Precision + Recall}$$

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

K-means clustering



C1	C2
dist(x1, C1)	dist(x1, C2)
dist(x2, C1)	dist(x2, C2)
…	…
dist(xn, C1)	dist(xn, C2)

$$Dist(x_1, x_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

i: index of feature
x: sample
n: number of features

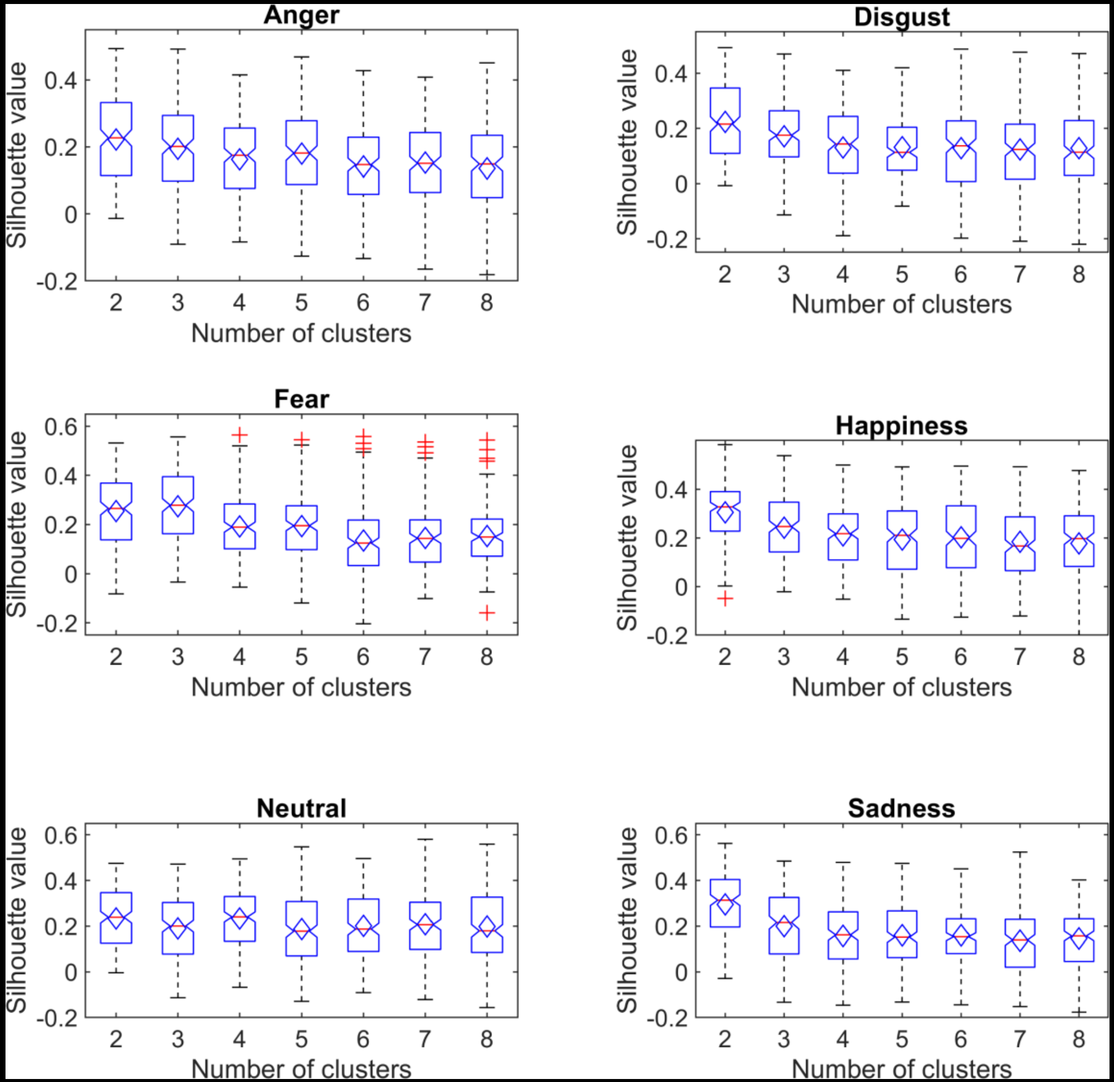
1. Randomly initialize the position of the centroids
2. Compute the distance matrix
3. Assign each point to the closest centroid → assign to a cluster
4. Update the centroids to the mean of each cluster
5. Repeat 2→4 until the centroids no longer move

$$SSE = \sum_{i=1}^{n} (x_i - C_j)^2$$

Sum of squared error → minimized
i: sample
j: assigned centroid
n: number of samples

https://www.mathworks.com/help/stats/kmeans.html#bueftl4-1

Silhouette values



$$S_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

S_i is the silhouette value of the *i*th point

a_i is the average distance from the *i*th point to the other points in the same cluster as *i*,

b_i is the minimum average distance from the *i*th point to points in a different cluster, minimized over the clusters.

The silhouette values range from –1 to 1. A high silhouette value indicates that the point is well matched to its own cluster, and poorly matched to other clusters.

If most points have a high silhouette value, then the clustering solution is appropriate.

If many points have a low or negative silhouette value, then the clustering solution might have too many or too few clusters.

https://www.mathworks.com/help/stats/silhouette.html

Multiband parametric equalization

Linear combination of a notch and a peak filter:

$$H_{eq}(z) = G_0 H_{notch}(z) + G H_{peak}(z)$$

where $H_{eq}(z)$ is the magnitude response of the equalizer,
$H_{notch}(z)$ and $H_{peak}(z)$ are the transfer function of the notch and the peak filters, respectively,
G_0 and G are their gains.

$$\omega_0 = \frac{2\pi f_0}{f_s}$$

$$\Delta\omega = \frac{2\pi \Delta f}{f_s}$$

where $\Delta\omega$ is the bandwidth of the magnitude response (digital), ω_0 is the center frequency (digital), f_s is the sampling frequency, f_0 is the center frequency (analogue), and Δf is f_0's bandwidth (analogue).

$$\omega_0 = \sqrt{\omega_1 \, \omega_2}$$

$$\Delta\omega = \omega_2 - \omega_1$$

where ω_2 and ω_1 are the edges of the magnitude response, $\omega_2 > \omega_1$
The center frequency is the geometric mean of ω_1 and ω_2

Notch filter (transfer function):

$$H_{notch}(z) = b \, \frac{1 - 2\cos\omega_0 z^{-1} + z^{-2}}{1 - 2b\cos\omega_0 z^{-1} + (2b-1)\, z^{-2}}$$

where b is the gain, given by:

$$b = \frac{1}{1+\beta}$$

with:

$$\beta = \frac{\sqrt{1 - G_B^2}}{G_B} \tan\frac{\Delta\omega}{2}$$

Where G_B is the gain at the cutoff frequencies ω_1 and ω_2

Peak filter:

$$H_{peak}(z) = 1 - b \, \frac{1 - z^{-2}}{1 - 2b\cos\omega_0 z^{-1} + (2b-1)\, z^{-2}}$$

where b is the gain, given by:

$$b = \frac{1}{1+\beta}$$

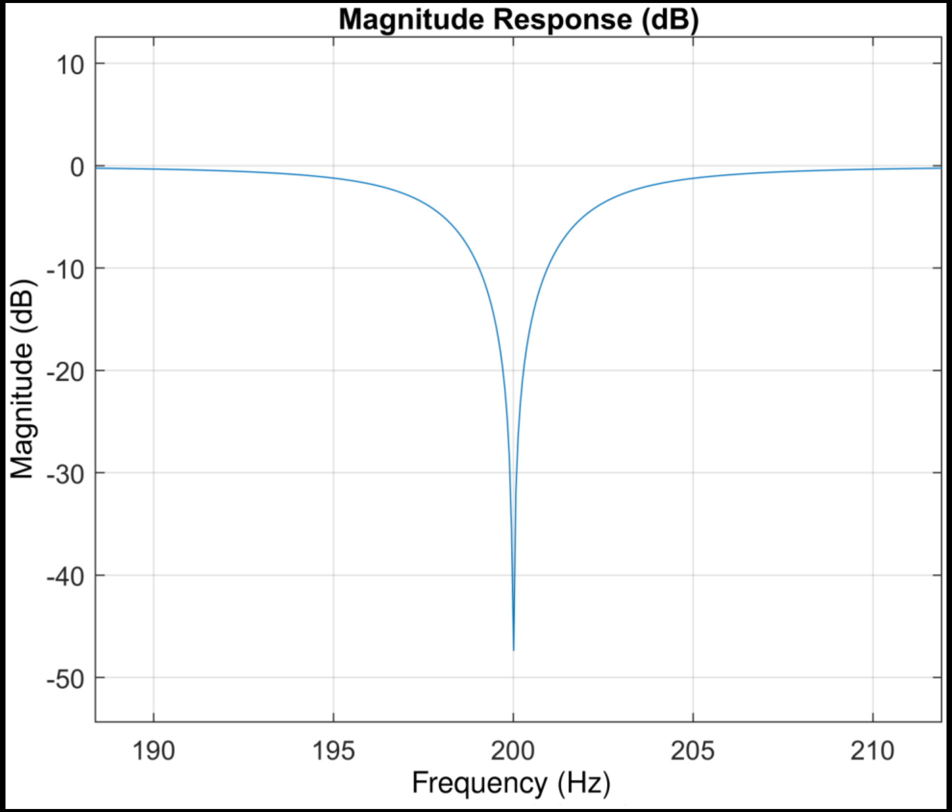
with:

$$\beta = \frac{G_B}{\sqrt{1 - G_B^2}} \tan\frac{\Delta\omega}{2}$$

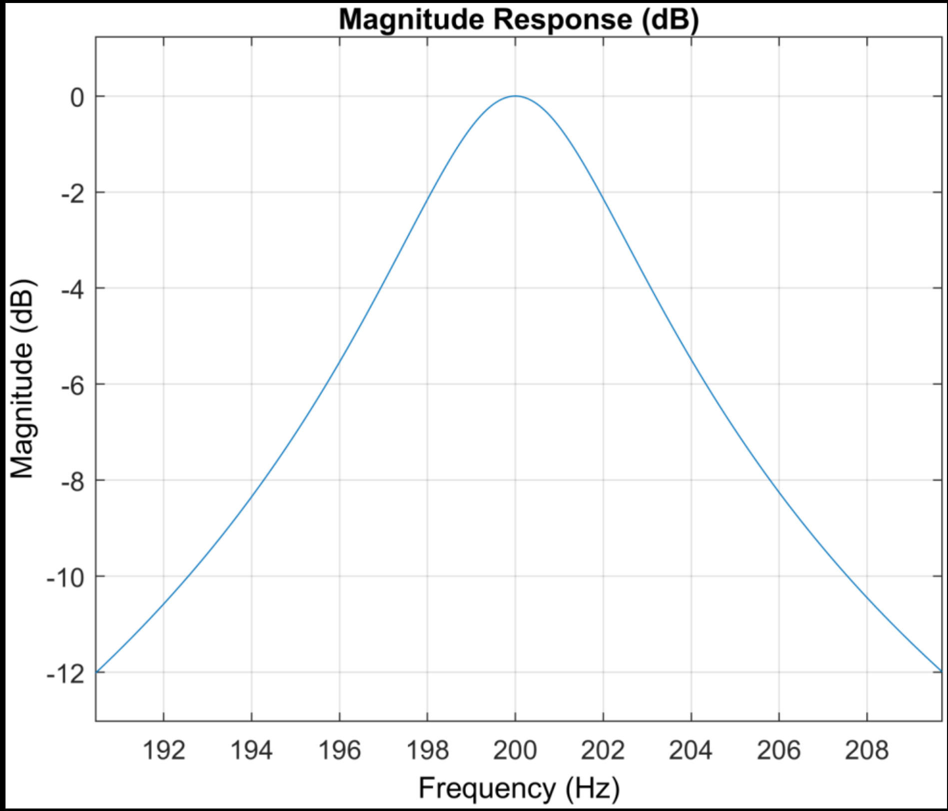
Where G_B is the gain at the cutoff frequencies ω_1 and ω_2

Multiband parametric equalization

Notch filter



Peak filter



Multiband parametric equalization

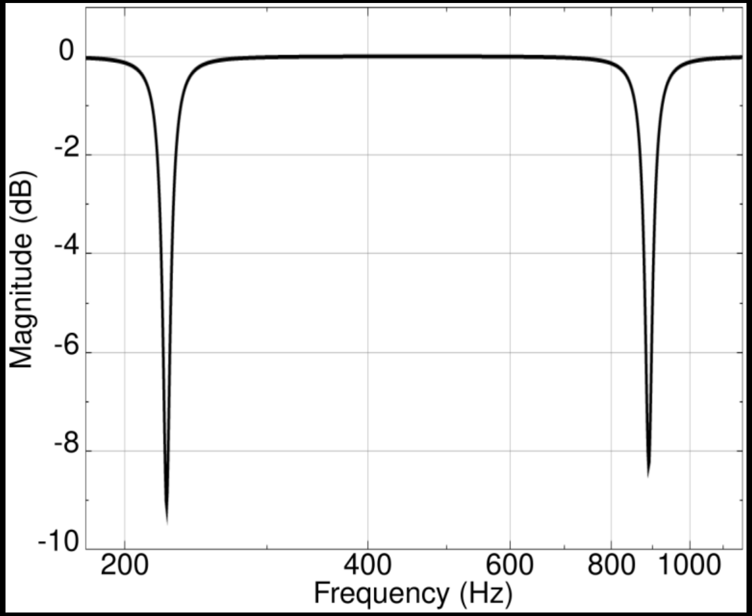
A multiband parametric equalizer acts as filter which parameters are **the center frequencies** to edit, **their bandwidth**, and their intensity **gain**. **The bandwidth parameter is defined by the quality factor**, also known as Q, and was computed as detailed in equation.

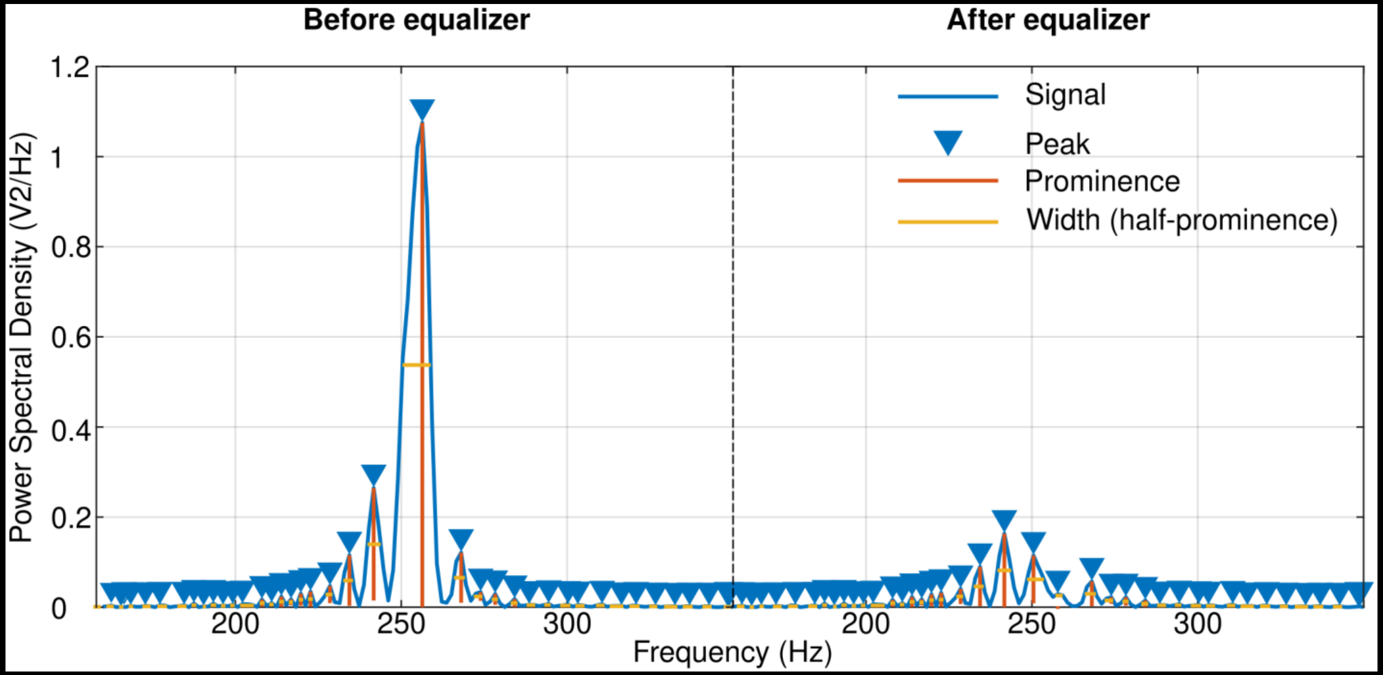
$$Q = \frac{\emptyset}{B}$$

where $\emptyset$ is the center frequency of the harmonic to be edited (Hz), and B is the bandwidth of interest.

Bandwidth: harmonics' BW

Distance between the points where the descending signal intercepts a horizontal reference line. This reference was positioned beneath the peak at a vertical distance equal to half the peak prominence

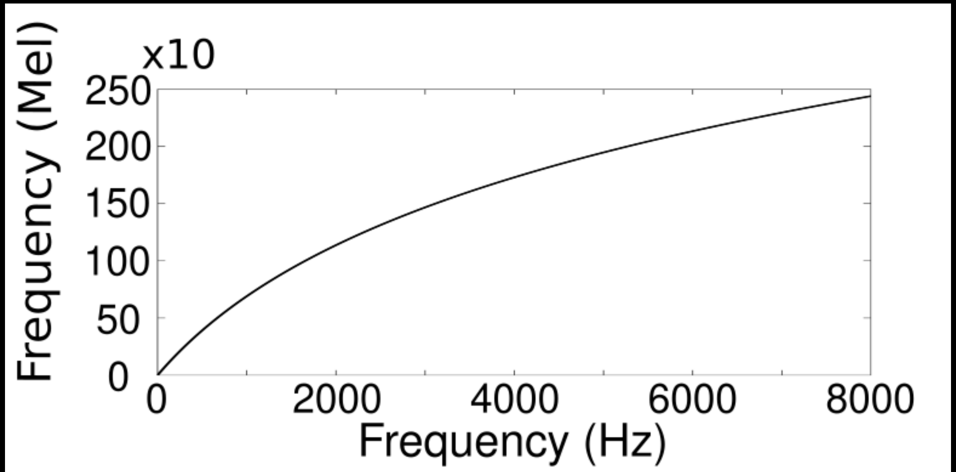




https://www.mathworks.com/help/audio/ref/multibandparametriceq-system-object.html

Mel Frequency Cepstral Coefficients (MFCC)

Reflect the processing of the human hearing system
Better precision within low frequencies



Pre-emphasis → Widowing → DFT → Triangular filters Mel → Energy (log) → DCT

Pre-emphasis

Increases the amplitude of high frequency with respect to low frequencies
Highlights the formants.

Widowing

~20 ms – Hanning – 50% overlap
Approximates stationarity and non-randomness

DFT

Power spectral density (PSD)

Triangular filters Mel

Spaced out according to the Mel scale
Higher precision at lower frequencies

Energy (log)

log(x^2)
Sum within the range of every filter

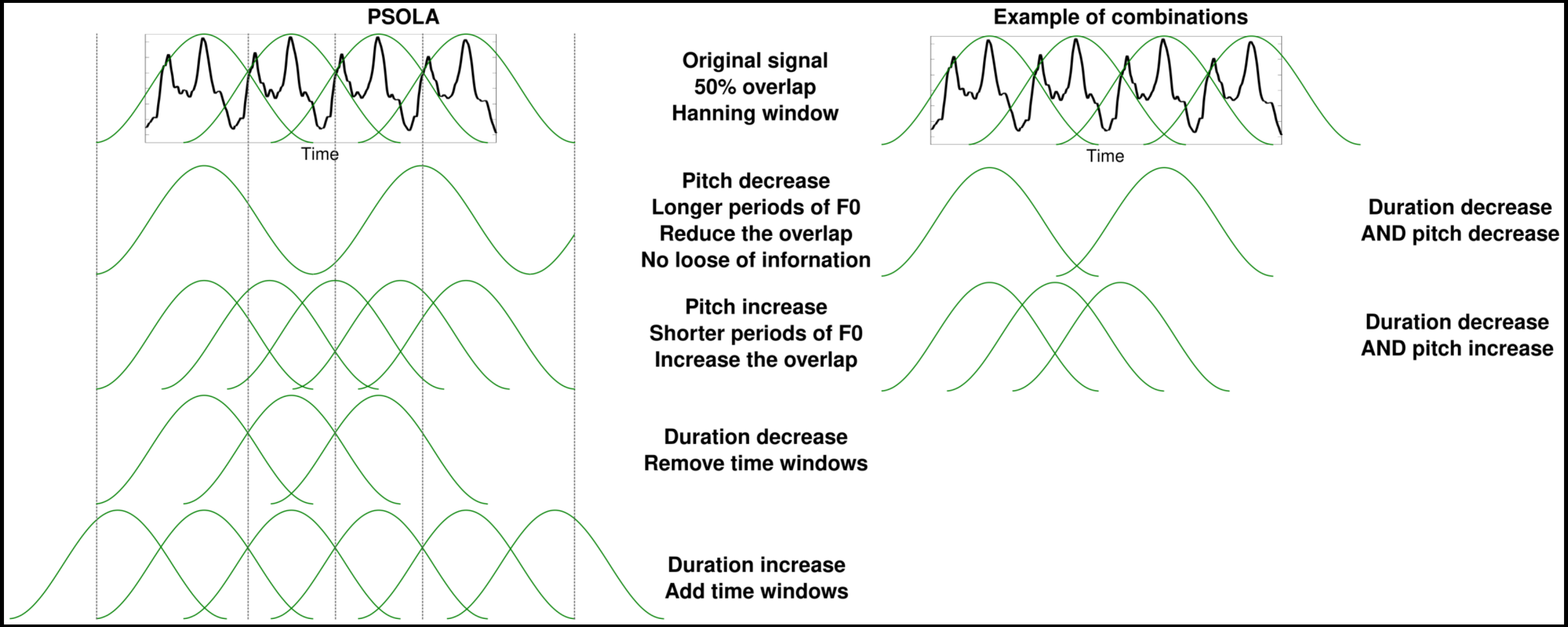
DCT

Discrete Cosine Transform

Reduces dependency between coefficients

https://www.mathworks.com/help/audio/ref/mfcc.html#mw_42368d66-4f36-4220-a679-1ab088533c79

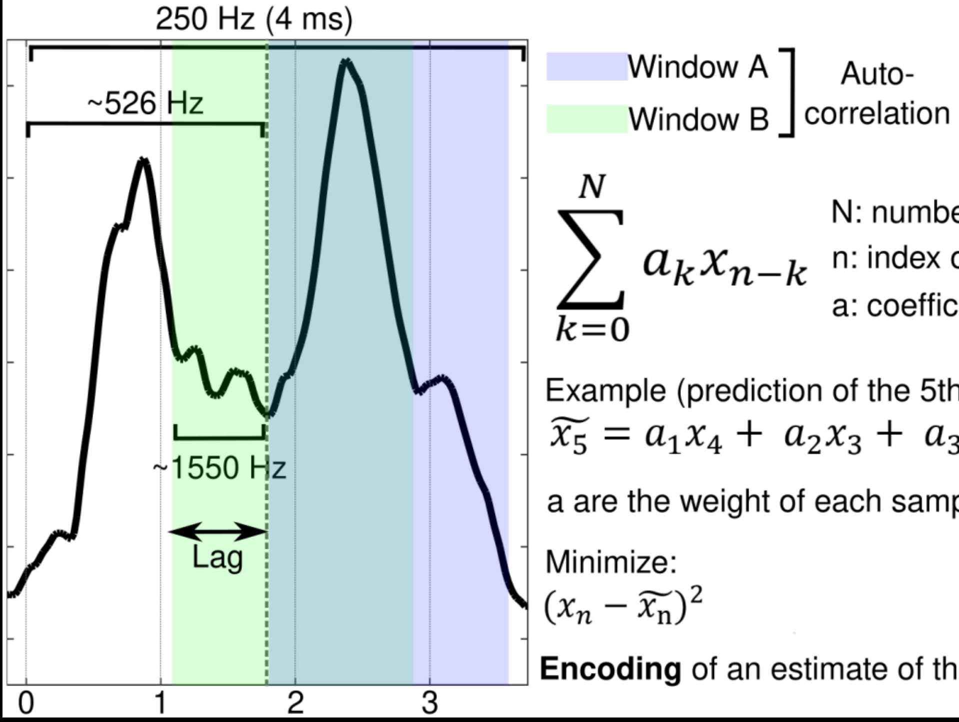
Pitch Synchronous Overlap and Add (PSOLA)



13

Linear Predictive Coding (LPC)

Linear predictive coding



Define 2 time windows of high autocorrelation
When the lag is exactly the time period of a component

$$\sum_{k=0}^{N} a_k x_{n-k}$$

N: number of samples
n: index of sample x
a: coefficients

Define a correlation function
Approximate of the vocal tract filter function

Example (prediction of the 5th sample):

$$\widetilde{x_5} = a_1 x_4 + a_2 x_3 + a_3 x_2 + a_4 x_1$$

a are the weight of each sample to predict the other

Minimize:

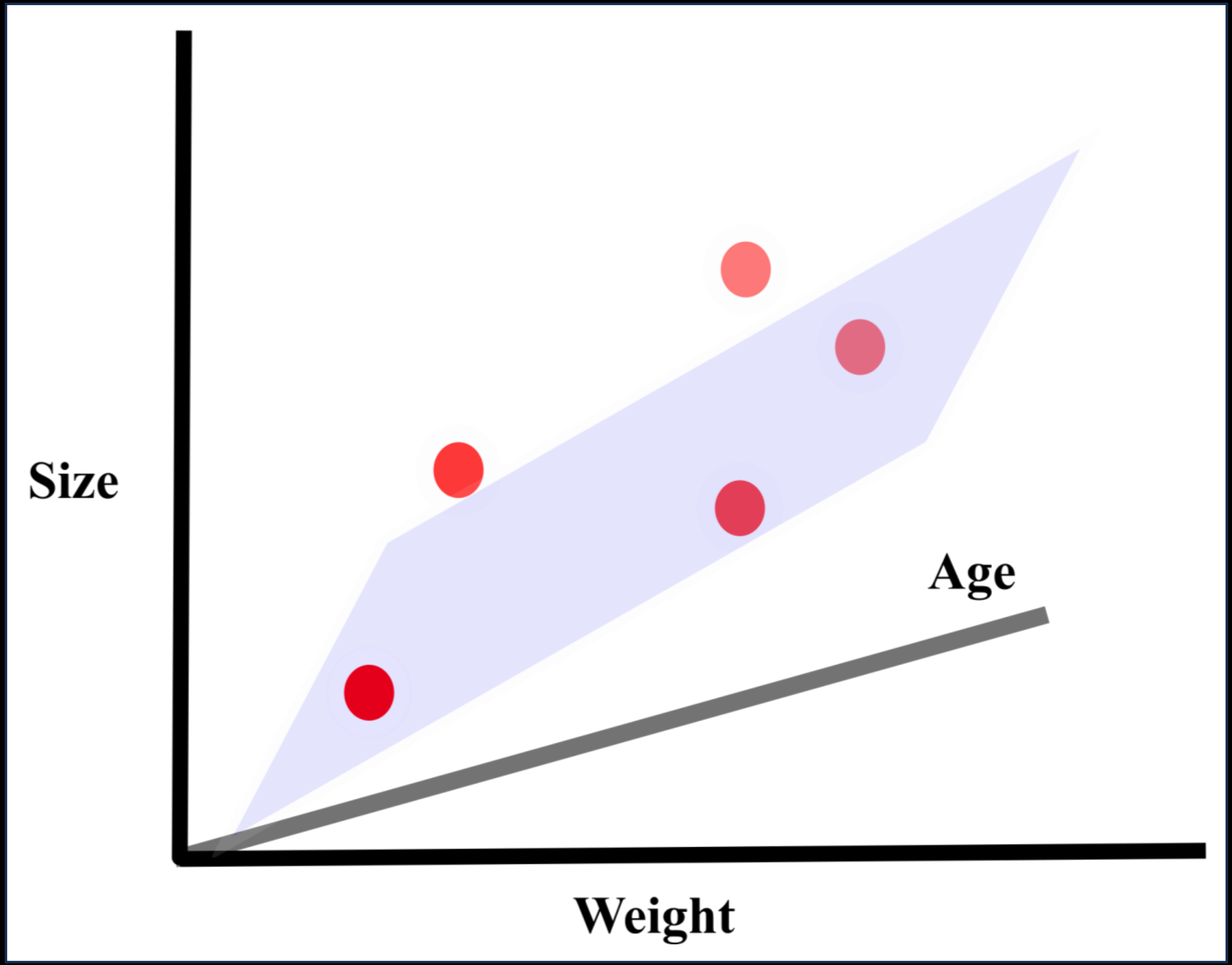
$$(x_n - \widetilde{x_\mathrm{n}})^2$$

Encoding of an estimate of the original signal

14

Multiple Linear Regression

Size = y-intercept + slope1*Weight + slope2*Age + ε



Find the plane:

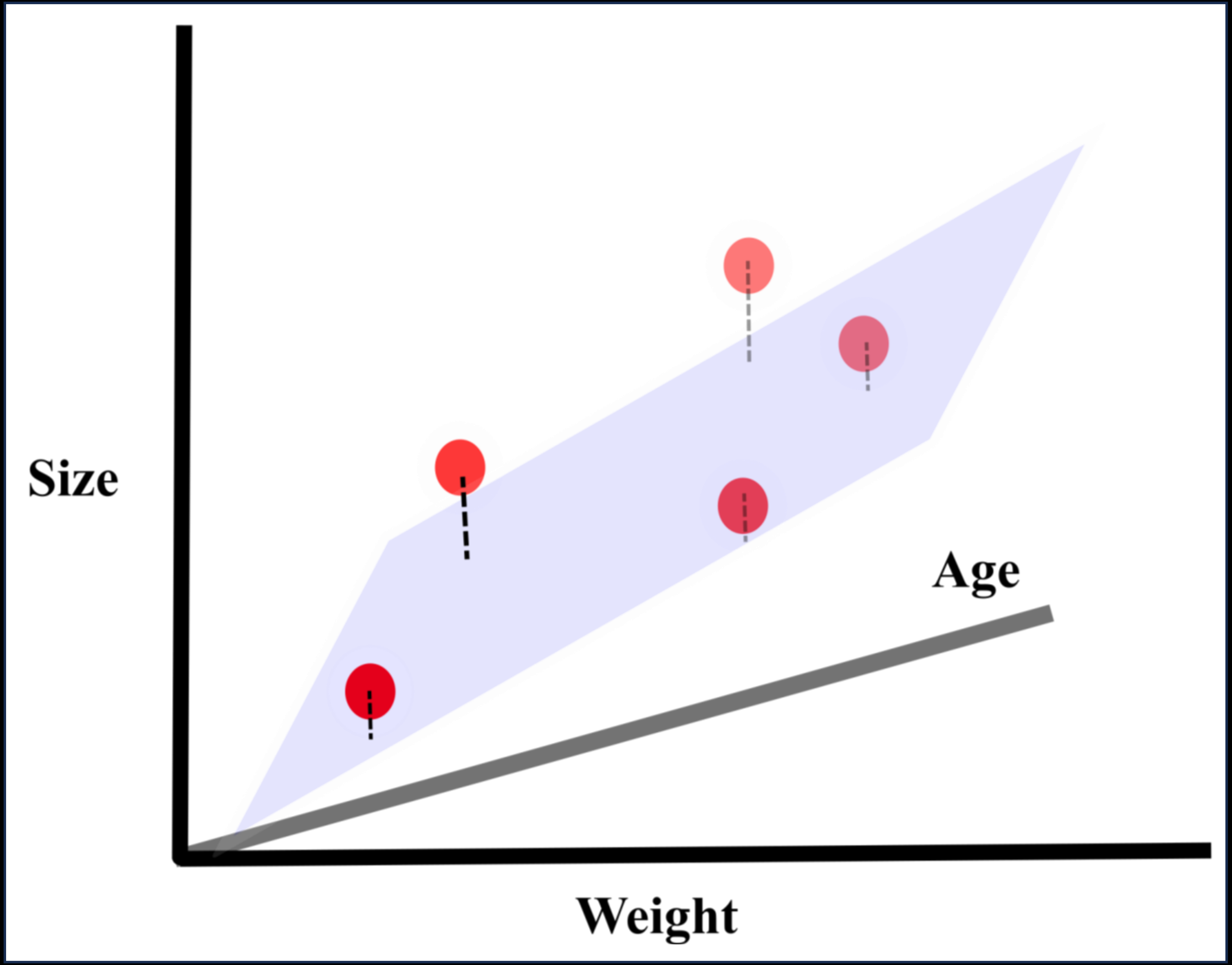
$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

y_i : response (dependent variable)
i : observation index
x_{ip}: observed value
β_p: model parameters (slopes)
ε_i: residual ($y_i - \hat{y}_i$)

15

Multiple Linear Regression

Size = y-intercept + slope1*Weight + slope2*Age + ε



That minimizes:

$$\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

N: Total number of observations

15

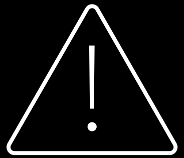
Multicollinearity

The relation for one independent variable with the dependent one is estimated when all other independent variables are constant

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

Tease apart the individual effects of each independent variable on the dependent variable

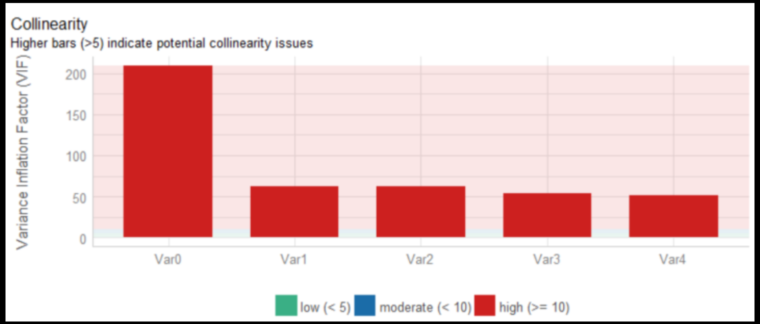
If independent variables are related, it is impossible to hold one variable constant while modulating the other



15

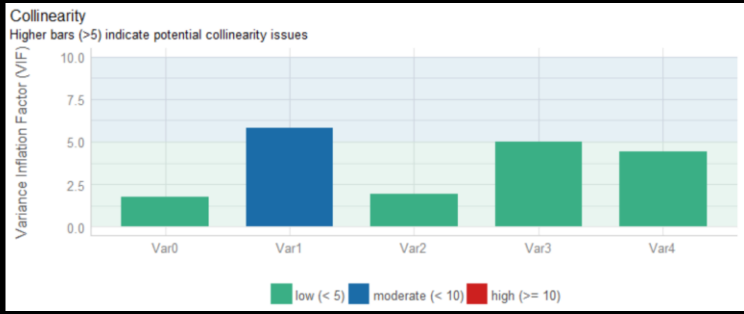
Multicollinearity

Multicollinearity inflates the variance of the affected variables



Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	138119	65960	2.094	0.104
Var0	9612	8628	1.114	0.328
Var1	-3787	6267	-0.604	0.578
Var2	-9336	7364	-1.268	0.274
Var3	1140	2322	0.491	0.649
Var4	15350	21721	0.707	0.519



Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	7850.0	26640.1	0.295	0.7829	
Var0	1762.7	2692.9	0.655	0.5485	
Var1	20287.0	9147.8	2.218	0.0908	.
Var2	3609.4	1606.6	2.247	0.0880	.
Var3	982.8	962.5	1.021	0.3649	
Var4	9257.2	7695.6	-1.203	0.2953	

Variance Inflation Factor

Create auxiliary regressions for every independent variable
For instance:

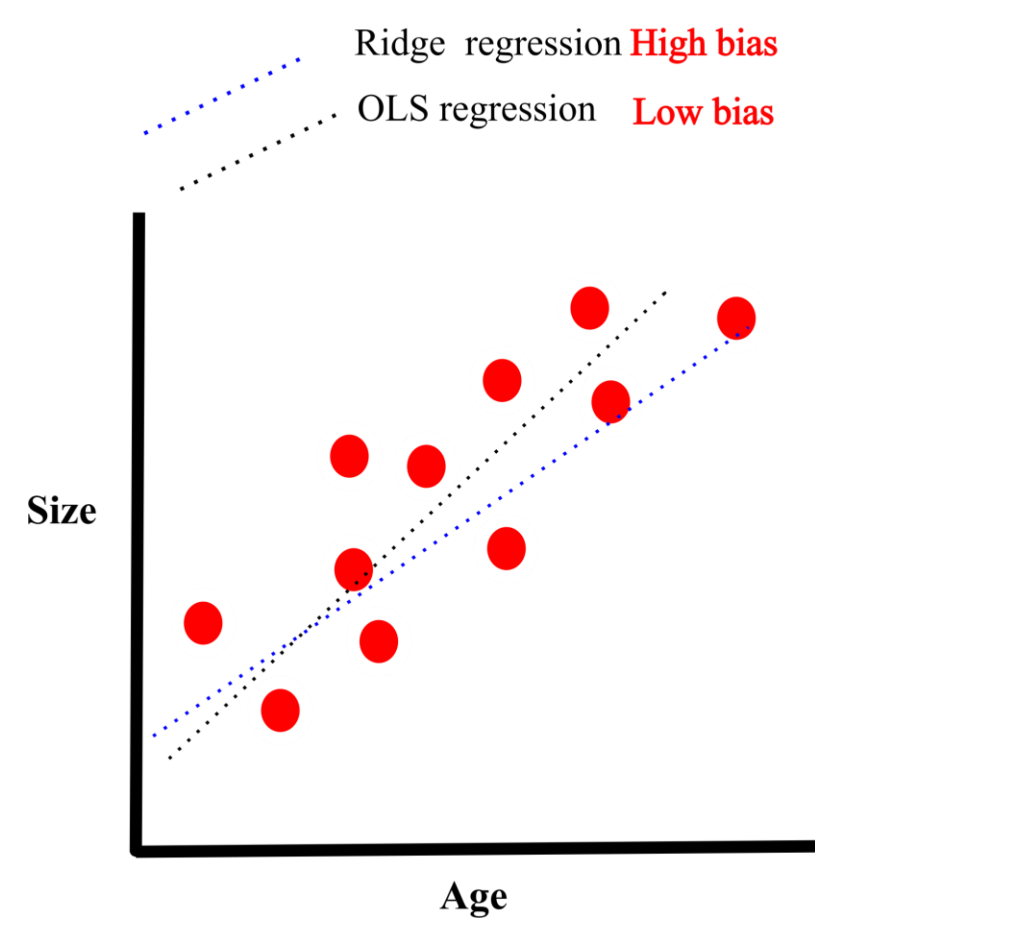
$$X_1 = \beta'_0 + \beta'_2 X_2 + \beta'_3 X_3 + \beta'_4 X_4 + \beta'_5 X_5 + \varepsilon'$$

How much of X_1 is being explained by the other independent variables

$$VIF_1 = {}^{1}/_{1 - R_1^2}$$

Ridge Regression

Higher bias is added to independent variables that explain least the dependent variable	Adding different amount of bias reduces the multicollinearity between variables



Ridge regression minimizes:

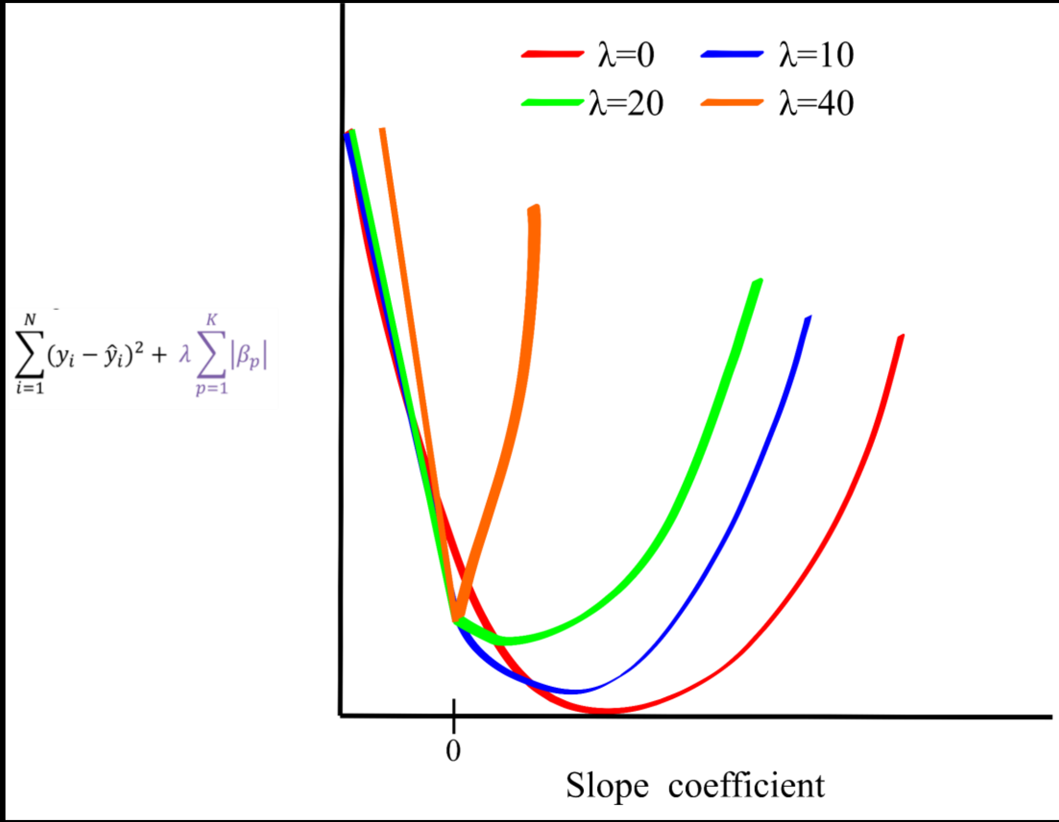
$$\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \lambda \sum_{p=1}^{K} \beta_p^2$$

i : observation index
y_i : response (dependent variable)
$\hat{y}_i$: estimated response
β_p: model parameters (slopes)
N: total number of observations
K = total number of parameters
λ: penalty parameter ($[0 \; +\infty]$)

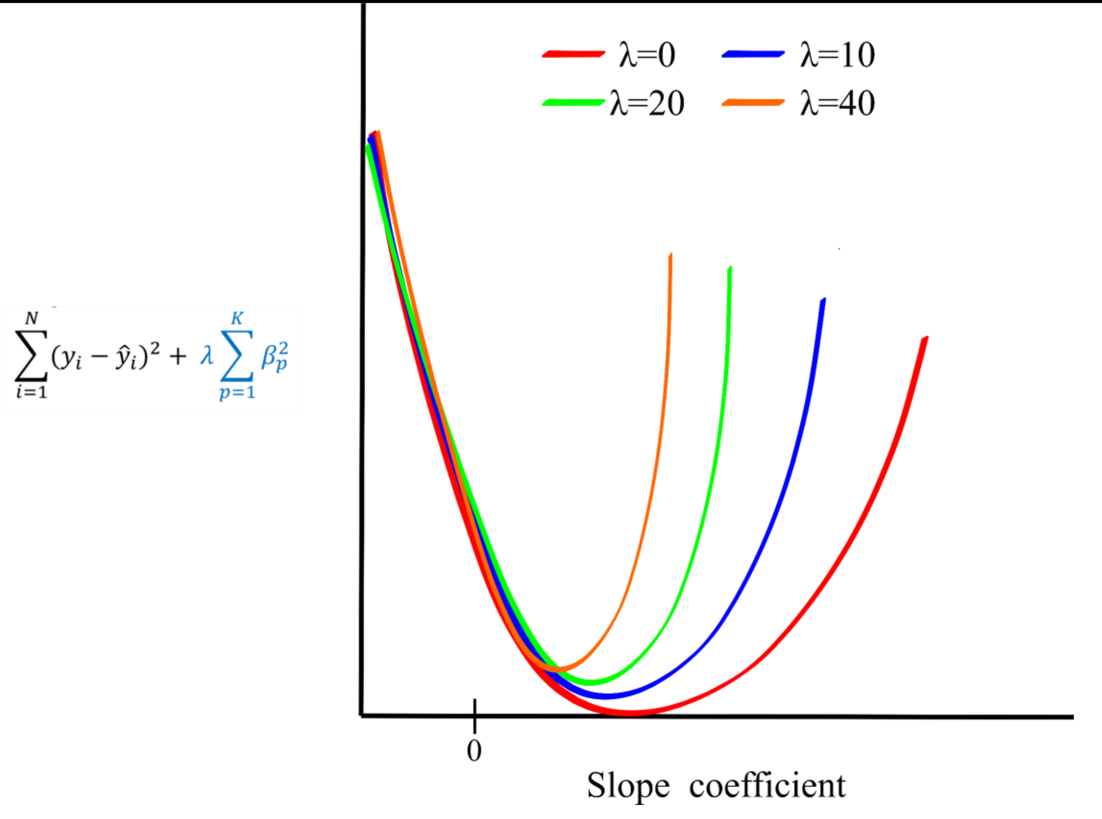
Ridge versus Lasso Regression

With Lasso, parameters of variables that predict less the dependent variables are cancelled out (slope=0)

Lasso regression



Ridge regression

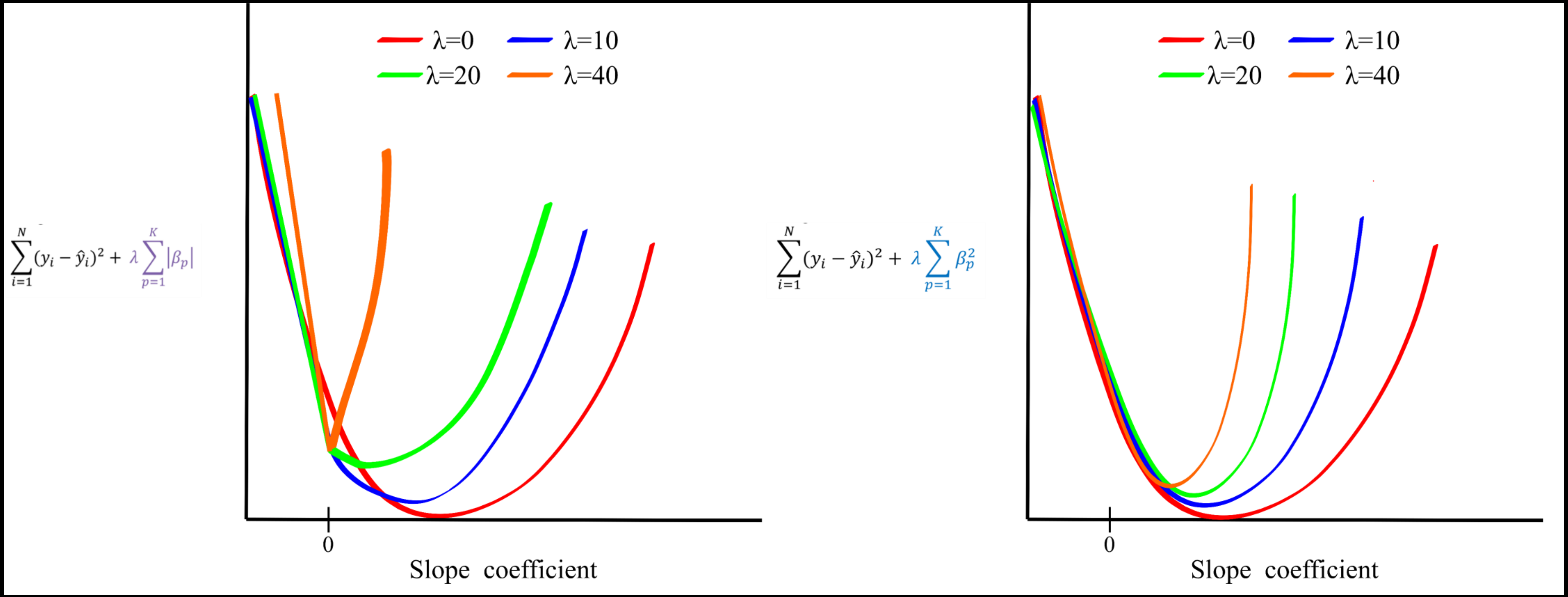


Ridge versus Lasso Regression

With Lasso, parameters of variables that predict less the dependent variables are cancelled out (slope=0)	Note on other methods: components decomposition methods (e.g., PCA) result in variables that are linear combinations of the original variables and are difficult to interpret

Lasso regression

Ridge regression



Smoothing Splines

Find a regression function *g* that minimizes:

Integral: sum over its entire range
Measure of the total change of g

$$\underbrace{\sum_{i=1}^{n}(y_i - g(x_i))^2}_{\text{Loss function}} + \underbrace{\lambda \int g''(t)^2 dt}_{\substack{\text{Penalty term} \\ \text{Penalizes the variability of } g}}$$

where λ is a positive tunning parameter, g is the smoothing spline, and g'' is its second derivative.
If g is smooth, g'' will have small values.
If $\lambda = 0$, the penalty term has no effect, and g exactly interpolates the data (more weight to minimize the difference between the data and the prediction by the curve).
If $\lambda \to \infty$, g is perfectly smooth (a straight line; more weight to minimize the variability of the prediction by the curve).

The second derivative reflects the amount by which the slope is changing

It is a measure of smoothness: it is large in absolute value if g(x) is very wiggly, and it is close to zero otherwise

Cluster-based permutations

Fp1	1.08	1.6	0.9	1.8	2.1	0.6	0.7
Fp2	0.5	2.7	2.3	2.1	2.7	3.8	2.68
AF3	1.8	0.9	2.1	0.8	2.9	3.6	4
AF4	2.1	1.05	0.3	1.5	3.9	2.99	3.7
F7	2.9	2.3	1.6	4.04	2.6	2.65	2.7
F3	0.3	1.9	1.2	1.4	4	2.73	3.88

Time (samples)

$p<0.05$

cluster

Significant comparison that is not within any cluster

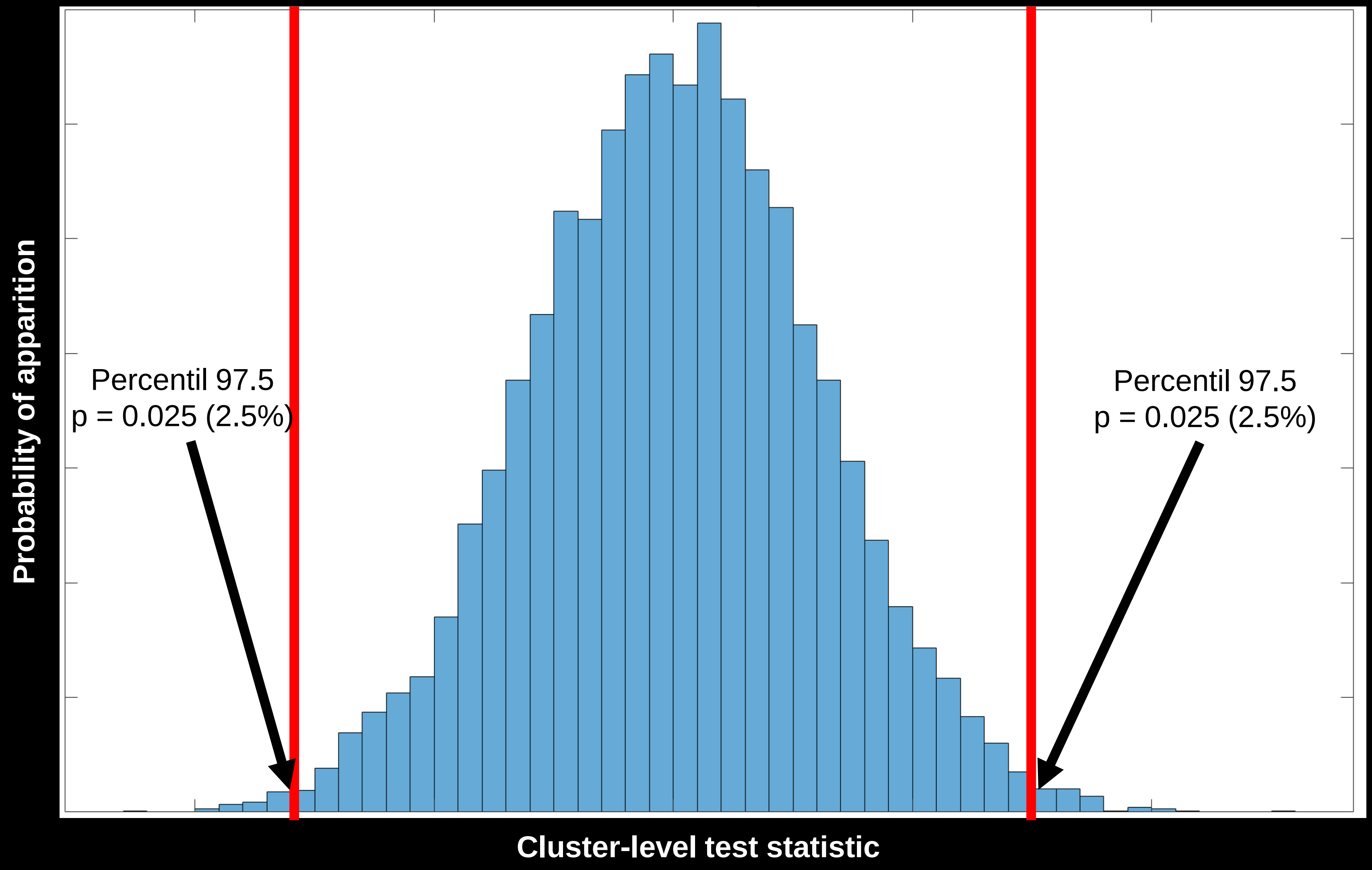
Cluster-level statistic: sum of statistics within every cluster

Cluster-based permutations

Monte-Carlo estimate with **1000 random permutations** under the null hypothesis of data exchangeability

For every iteration, the cluster-formation stage was repeated, and **the maximum cluster-level statistic** was stored to **build a distribution**.

Cluster-based permutations



The cluster-level statistic defined from the **observed** data was further **located on the empirical distribution**.

Because **a single value** (the cluster-level statistic) whose probability under the null hypothesis was considered instead of the 3-dimensional matrix, **multiple-comparison issues are exempted**.

16

Bayesian statistics

$$\underbrace{\left(\frac{P(H_1)}{P(H_0)}\right)}_{\text{Prior odds}}\underbrace{\left(\frac{P(D|H_1)}{P(D|H_0)}\right)}_{\text{Bayes factor }(BF_{10})} = \underbrace{\frac{P(H_1|D)}{P(H_0|D)}}_{\text{Posterior odds}}$$

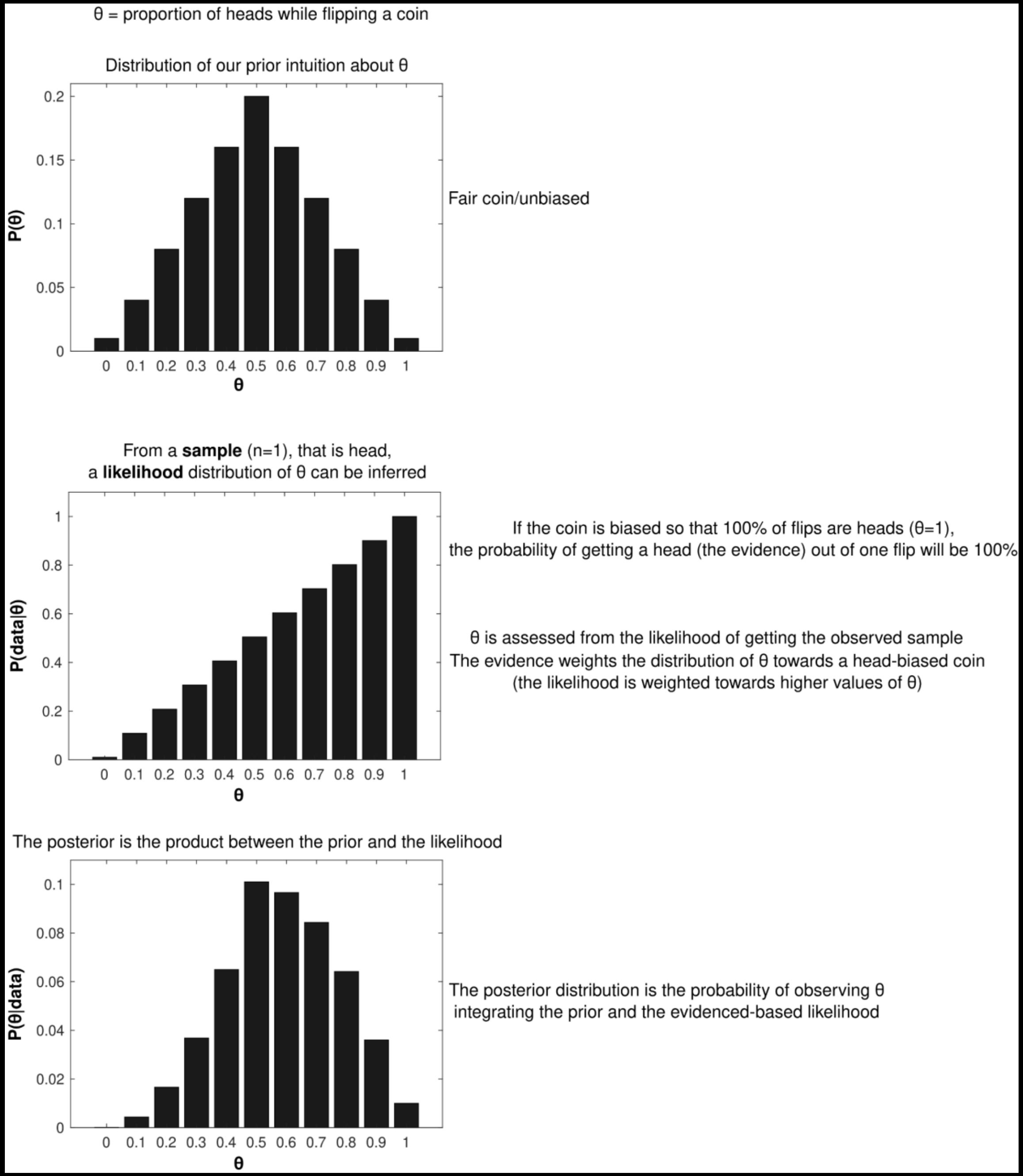
where the **prior** odds are the relative plausibility of the alternative hypothesis **before seeing the data**,
the Bayes factor is the **evidence provided by the data**,
the **posterior odds** are the relative plausibility of the alternative hypothesis **after seeing the data**,
D is the data under study

Updating in beliefs according to the data

$$\underbrace{P(\theta|D)}_{\text{Posterior}} = \frac{\overbrace{P(D|\theta)}^{\text{Likelihood}}\overbrace{P(\theta)}^{\text{Prior}}}{P(D)}$$

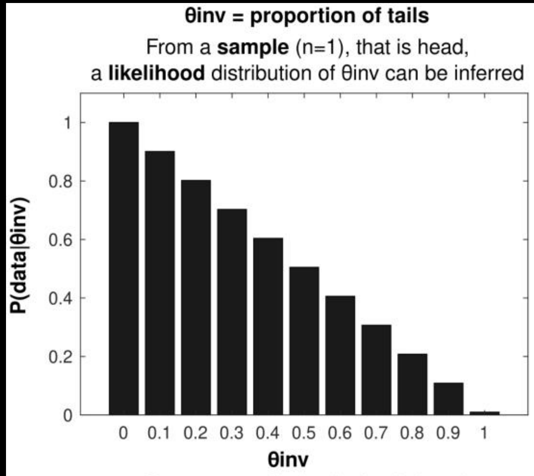
where θ is the parameter under study (e.g., proportion of heads/tails, proportion of different emotions)
D is the sample (data under study)
The likelihood is the distribution possible sample outcomes for every possible value of θ.
The posterior distribution is proportional to the likelihood times the prior distribution.
Updating the prior distribution with information from the sample.
Precision: weighting the information from the prior relative to the information from the sample through the likelihood.

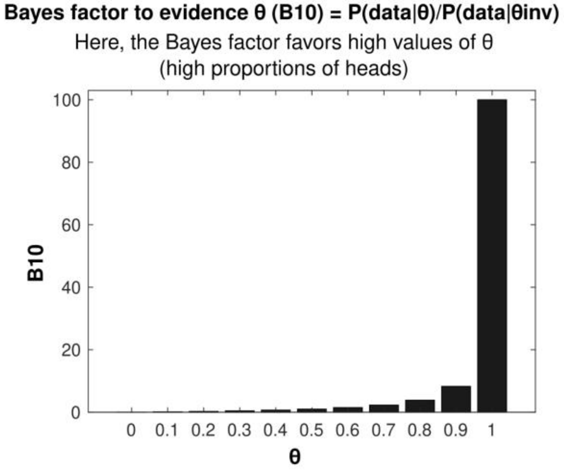
The Bayes factor reflects the likelihood

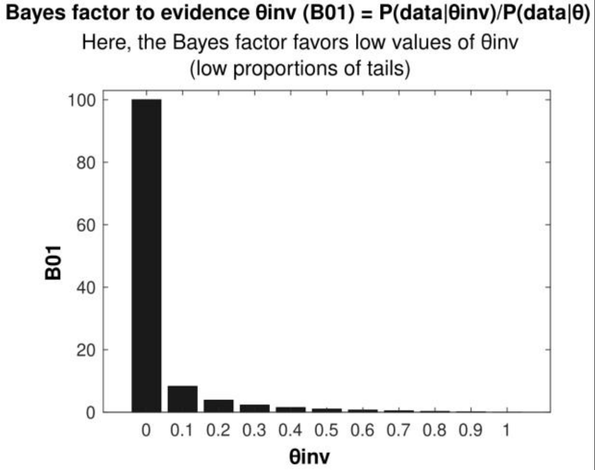


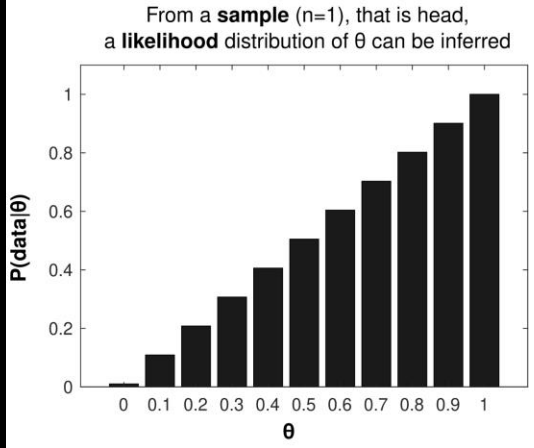
17

Bayesian statistics









B_{01}	Interpretation
1/30 – 1/100	Very strong evidence for the alternative hypothesis (H1)
1/10 – 1/30	Strong evidence for the alternative hypothesis (H1)
1/3 – 1/10	Moderate evidence for the alternative hypothesis (H1)
1 – 1/3	Anecdotal evidence for the alternative hypothesis (H1)
1	No evidence
1 - 3	Anecdotal evidence for the alternative hypothesis (H0)
3 - 10	Moderate evidence for the alternative hypothesis (H0)
10 - 30	Strong evidence for the alternative hypothesis (H0)
30 - 100	Very strong evidence for the alternative hypothesis (H0)

Example1:

The hypothesis H1 could be that the coin is highly biased towards heads (θ=1; high B10)

H0 would be that the coin is highly biased towards tails (θinv=1; low B01)

Example2:

H1: the coin is unbiased (θ=0.5; low B10)

H0: the coin is biased (θinv≠0.5, high B01)

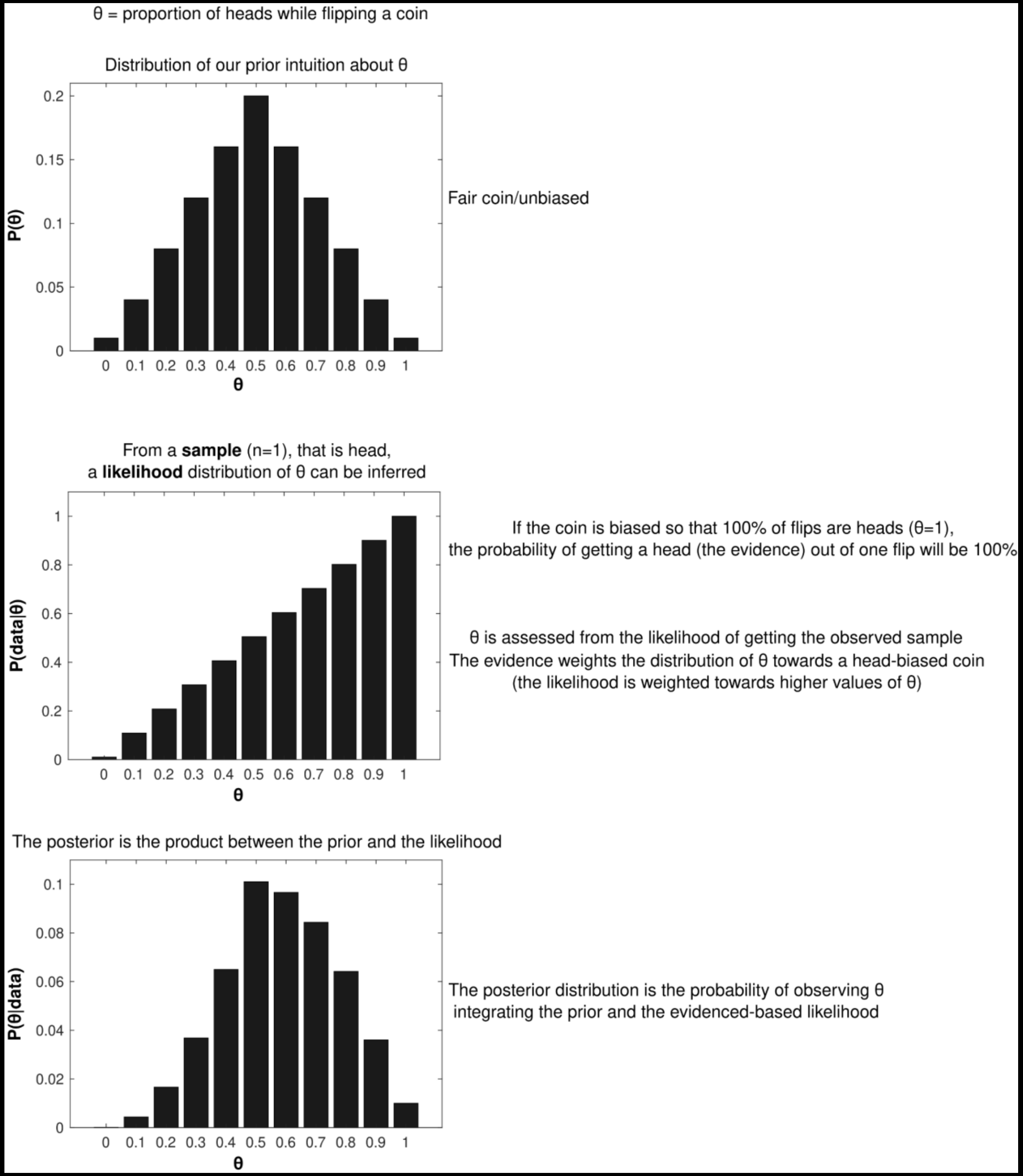
17

Bayesian statistics

About the human perception:

If the variability of the sensory evidence (data, D) is lower than the expected variability (priors), the weight towards the sensory evidence will be low (likelihood distribution; precision) and the perception (posterior) will be biased towards the priors (previously acquired knowledge).

In autism: atypical adaptation of precision (likelihood distribution) towards overweighting the sensory evidence and biasing both the perception and the priors towards fine-grained information (constant high prediction error).



17

Entropy and Mutual Information

Mutual Information (MI)

- MI could be used to compute the tracking of speech by neuronal response or to assess brain-brain couplings
- Both linear and non-linear dependencies
- All types of synchronizations : **phase-phase**, phase-frequency, **phase-amplitude**, amplitude-frequency, amplitude-amplitude, and frequency-frequency

Quantification of the amount of shared information between two time series based on Shannon entropy

- Higher entropy, higher number of possible configurations, higher amount of information
- Bins the time series as to create a histogram
- Expressed in bits for which one bit represents the uncertainty of an equally probable binary variable (e.g., a fair coin).

$$H(X) = -\sum_{i=1}^{N} p(X_i) log_2\, p(X_i)$$

where $p(X_i)$ is the probability of observing the ith value of the bin series signal X, and N is the number of bins. $p(X_i)$ may be calculated by dividing the bin count by the sum of all bin counts.

Entropy and Mutual Information

- Relies on probabilities / Independent of temporal and amplitude modulations and scaling

Number of bins

- Sensitive to the number of bins
- Freedman-Diaconis rule

$$nbins = \left\lceil \frac{\max(X) - \min(X)}{2Q_X n^{-1/3}} \right\rceil$$

where Q is the interquartile range of time series distribution X, n is the total number of samples. $\lceil x \rceil$ indicates ceiling

- For MI: average number of bins between both time series

Entropy and Mutual Information

- Relies on probabilities / Independent of temporal and amplitude modulations and scaling

Number of bins

- Sensitive to the number of bins
- Freedman-Diaconis rule

$$nbins = \left\lceil \frac{\max(X) - \min(X)}{2Q_X n^{-1/3}} \right\rceil$$

where Q is the interquartile range of time series distribution X, n is the total number of samples. $\lceil x \rceil$ indicates ceiling

- For MI: average number of bins between both time series

Joint entropy

- Amount of information available by both variables

$$H(X,Y) = -\sum_{j=1}^{M}\sum_{i=1}^{N} p(X_i, Y_j) log_2\, p(X_i, Y_j)$$

where $p(X_i, Y_j)$ is the probability of observing both ith and jth values of the bin series signals X and Y, N is the number of bins used for X, and M is the number of bins used for Y.

Entropy and Mutual Information

- Relies on probabilities / Independent of temporal and amplitude modulations and scaling

Number of bins

- Sensitive to the number of bins
- Freedman-Diaconis rule

$$nbins = \left\lceil \frac{\max(X) - \min(X)}{2Q_X n^{-1/3}} \right\rceil$$

where Q is the interquartile range of time series distribution X, n is the total number of samples. $\lceil x \rceil$ indicates ceiling

- For MI: average number of bins between both time series

Joint entropy

- Amount of information available by both variables

$$H(X,Y) = -\sum_{j=1}^{M}\sum_{i=1}^{N} p(X_i, Y_j) log_2\, p(X_i, Y_j)$$

where $p(X_i, Y_j)$ is the probability of observing both i^th and j^th values of the bin series signals X and Y, N is the number of bins used for X, and M is the number of bins used for Y.

Mutual Information (MI)

$$\boxed{MI(X,Y) = H(X) + H(Y) - H(X,Y)}$$

Entropy and Mutual Information

Number of samples

- A low number of samples : overestimation of entropy
- Factor of inflation

$$\Delta MI = \frac{(nbins-1)^2}{2nln2}$$

where $nbins$ is the number of bins used to compute MI, n is the number of samples of time series, and $ln2$ is the natural logarithm of 2.

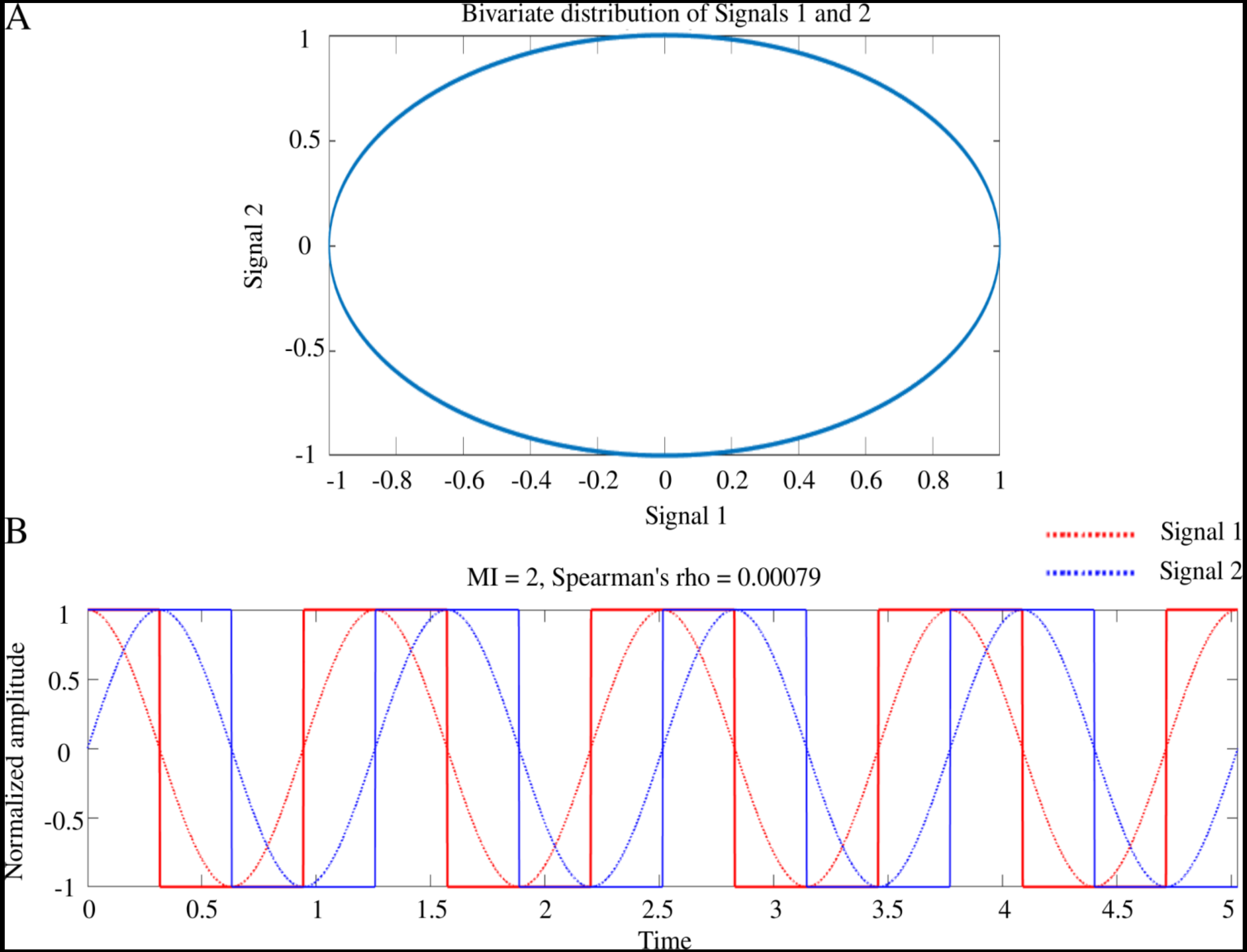
Compute MI, and then subtract the factor of inflation from the results to correct for the bias originated from sample size.

Advantages

- Independent of linearity or non-linearity of the dependency
- Assess dependencies between two time series with different distributions

Drawbacks

Entropy and Mutual Information



Entropy and Mutual Information

Number of samples

- A low number of samples : overestimation of entropy
- Factor of inflation

$$\Delta MI = \frac{(nbins-1)^2}{2nln2}$$

where $nbins$ is the number of bins used to compute MI, n is the number of samples of time series, and $ln2$ is the natural logarithm of 2.

Compute MI, and then subtract the factor of inflation from the results to correct for the bias originated from sample size.

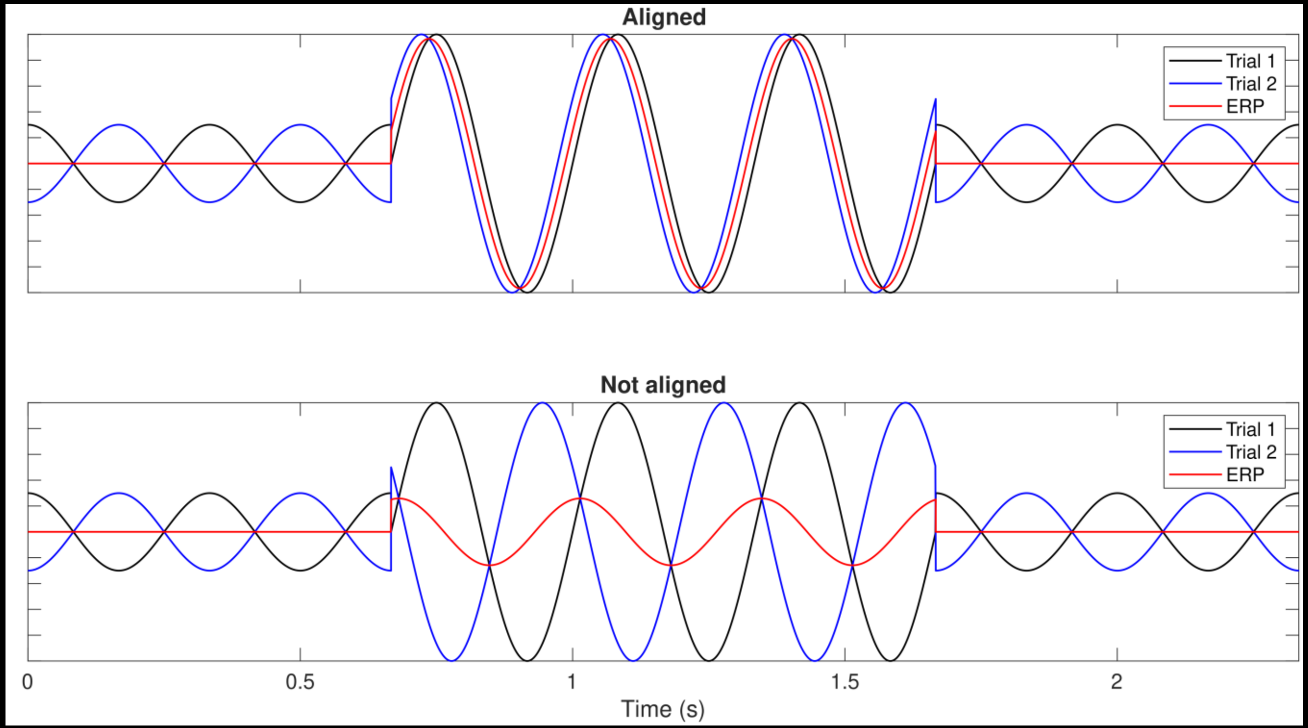
Advantages

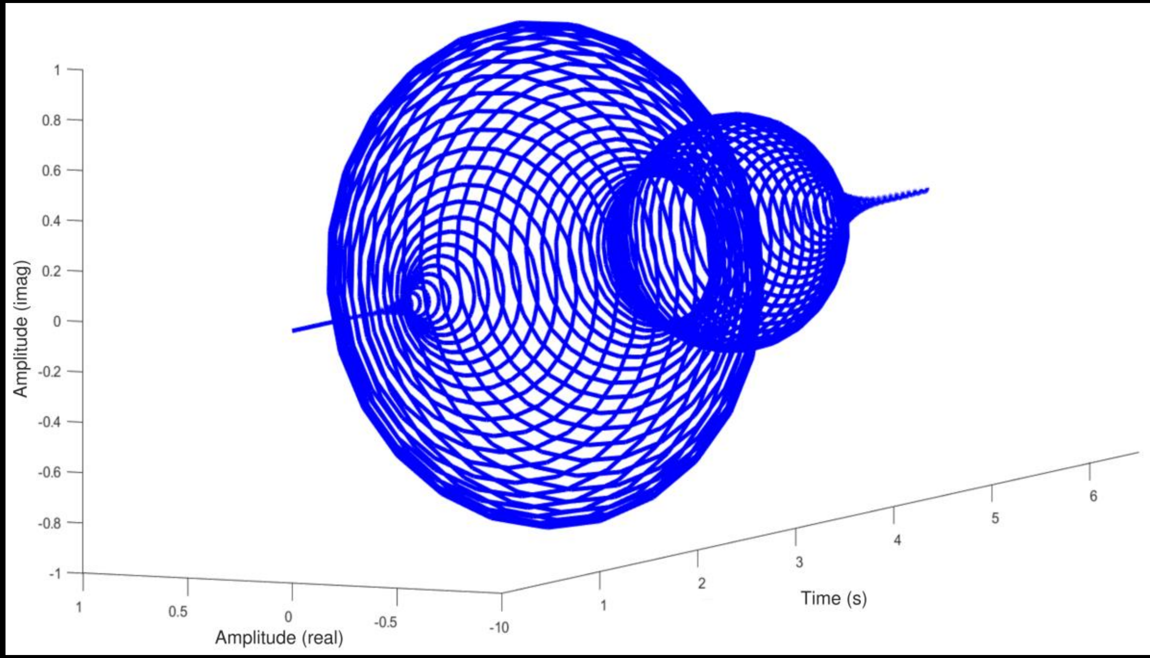
- Independent of linearity or non-linearity of the dependency
- Assess dependencies between two time series with different distributions

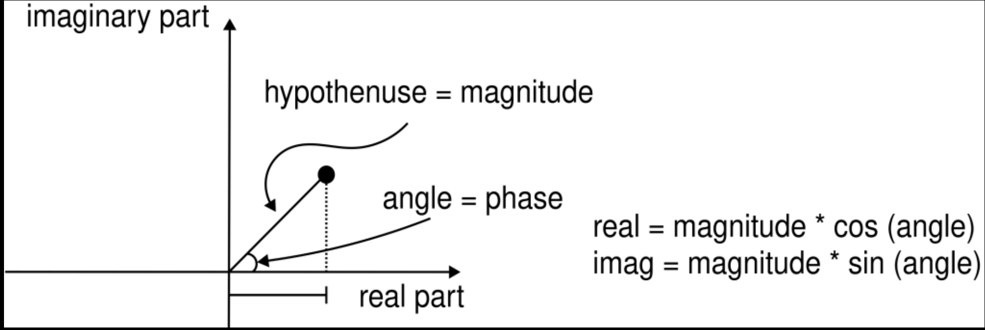
Drawbacks

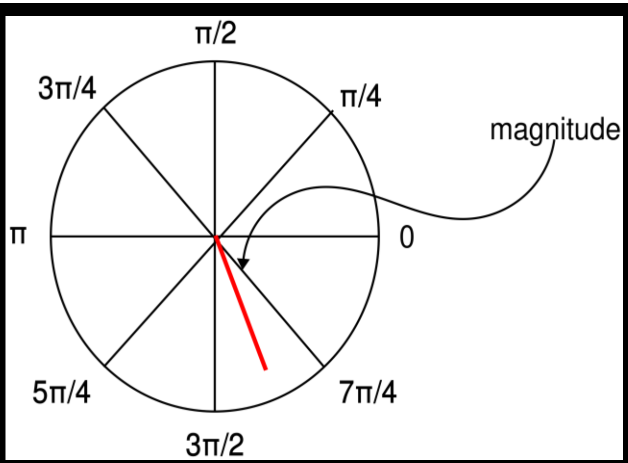
- Sensitive to noise
  - ✓ High levels of noise : underestimation of MI
  - ✓ Requires clean data

Inter-Trial Phase Coherence (ITC or ITPC)









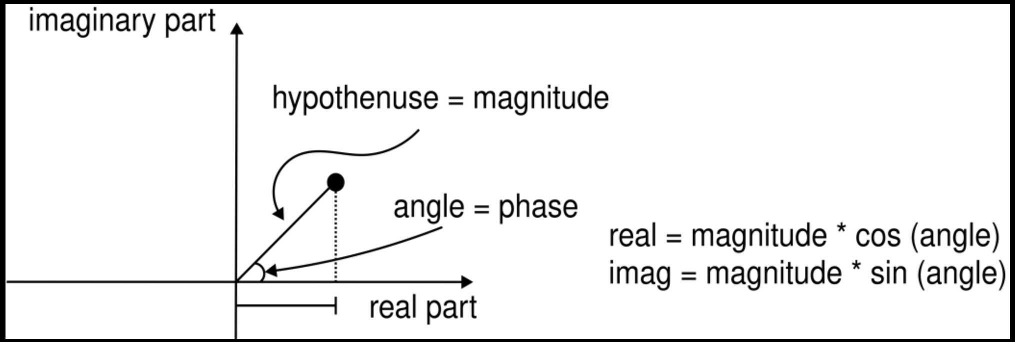
18

Inter-Trial Phase Coherence (ITC or ITPC)

$$a + ib = M[\cos(\theta) + i\sin(\theta)]$$

$$Me^{i\theta} = M[\cos(\theta) + i\sin(\theta)]$$ Euler's formula

where $e \sim 2.72$ т



$$ITPC = \left| mean_{trials}\left({}^{M[\cos(\theta) + i\sin(\theta)]}/_{M} \right) \right|$$ Only phase information

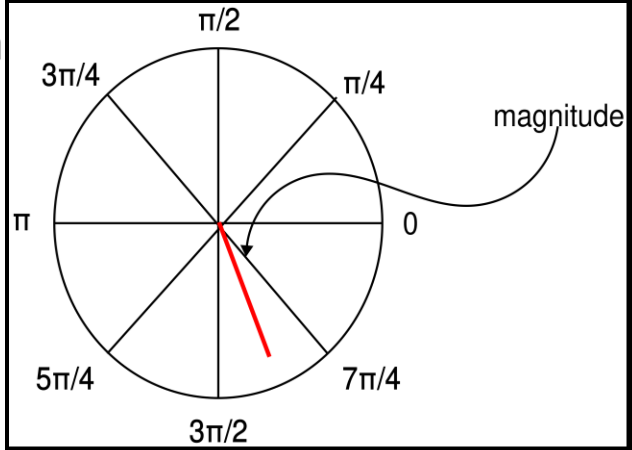
If $\theta_1 = \pi, \cos(\theta_1) + i\sin(\theta_1) = -1$

If $\theta_2 = 0, \cos(\theta_2) + i\sin(\theta_2) = 1$

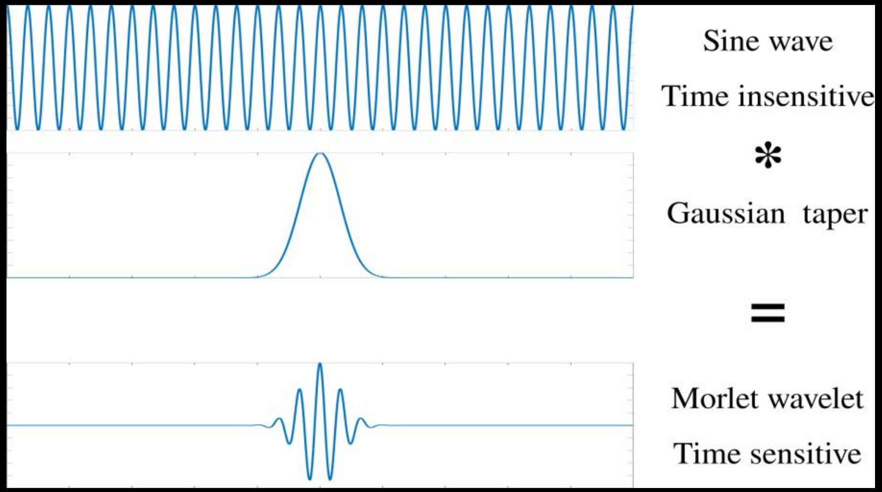
$ITPC = \left| {}^{-1+1}/_{2} \right| = 0$

If $\theta_1 = \theta_2 = 0, \cos(\theta_{1\ or\ 2}) + i\sin(\theta_{1\ or\ 2}) = 1$

$ITPC = \left| {}^{1+1}/_{2} \right| = 1$

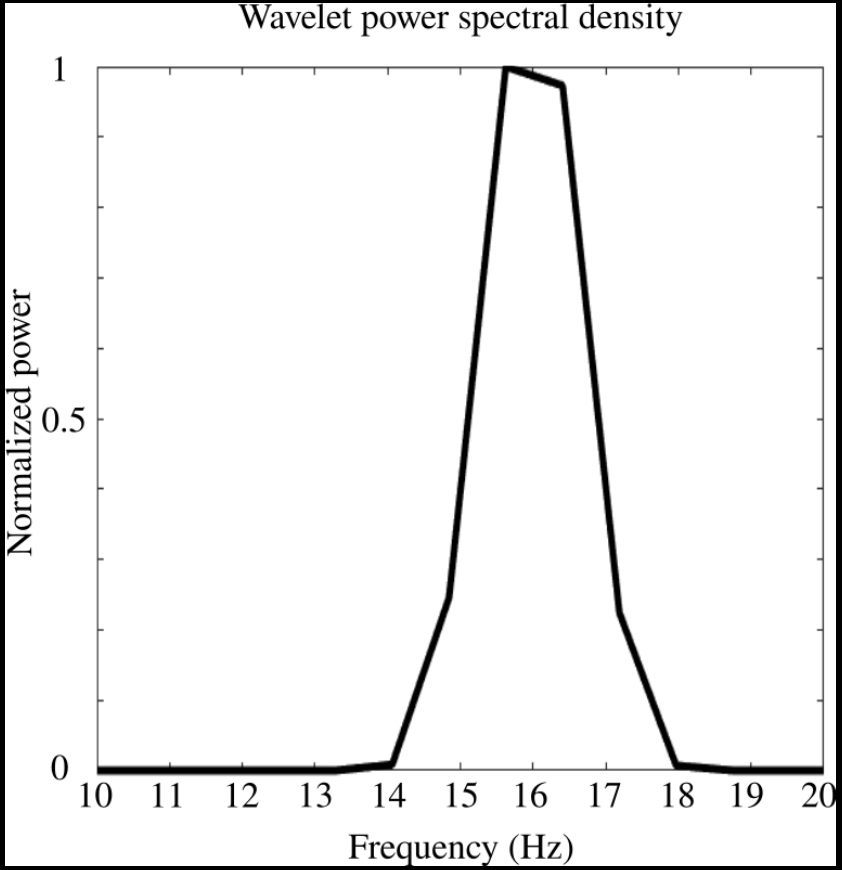


Inter-Trial Phase Coherence (ITC or ITPC)

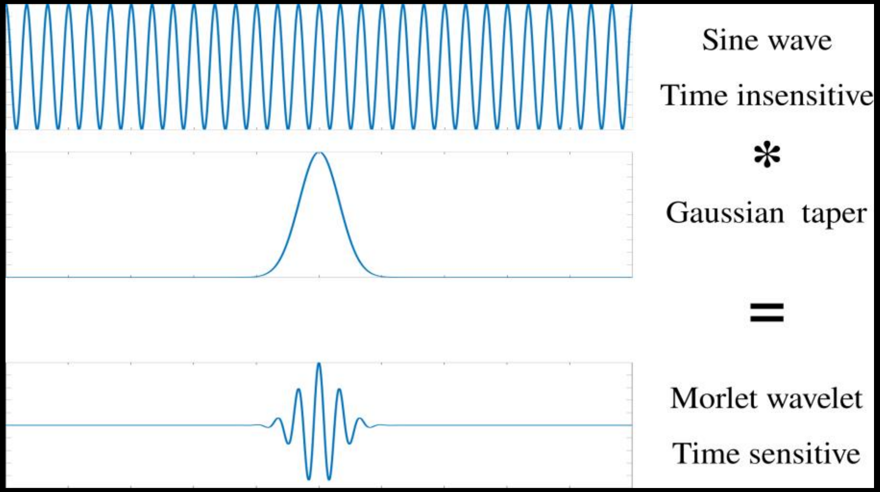


Wavelet peak frequency = sine frequency
Wavelet contains a range of frequencies

Inter-Trial Phase Coherence (ITC or ITPC)

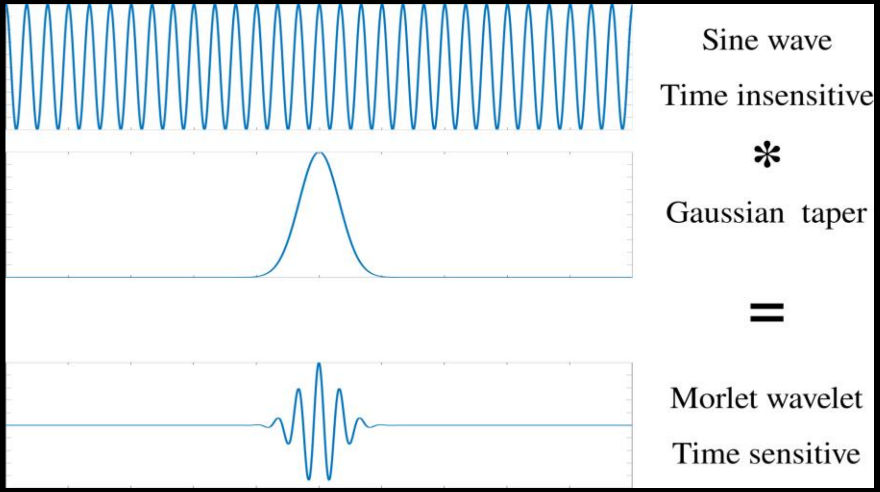


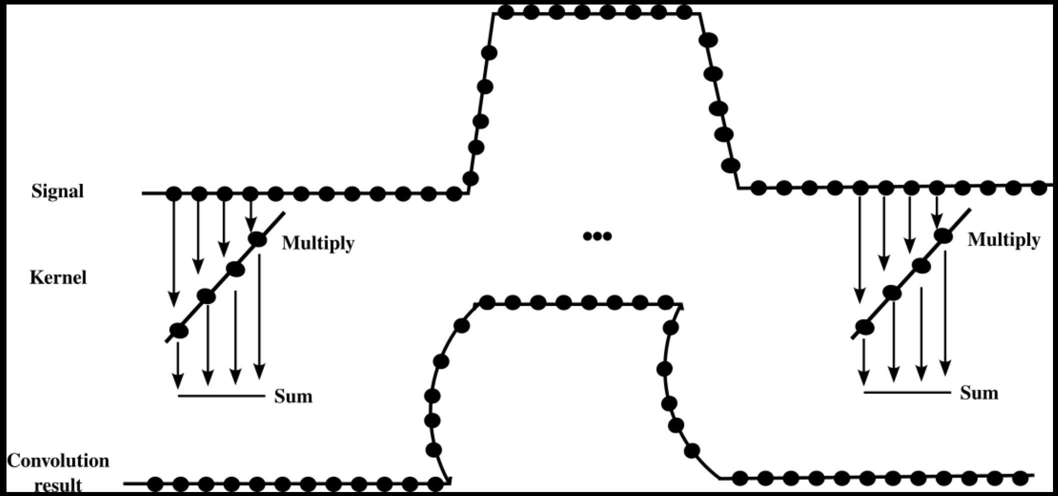
Inter-Trial Phase Coherence (ITC or ITPC)



Wavelet peak frequency = sine frequency
Wavelet contains a range of frequencies

Inter-Trial Phase Coherence (ITC or ITPC)

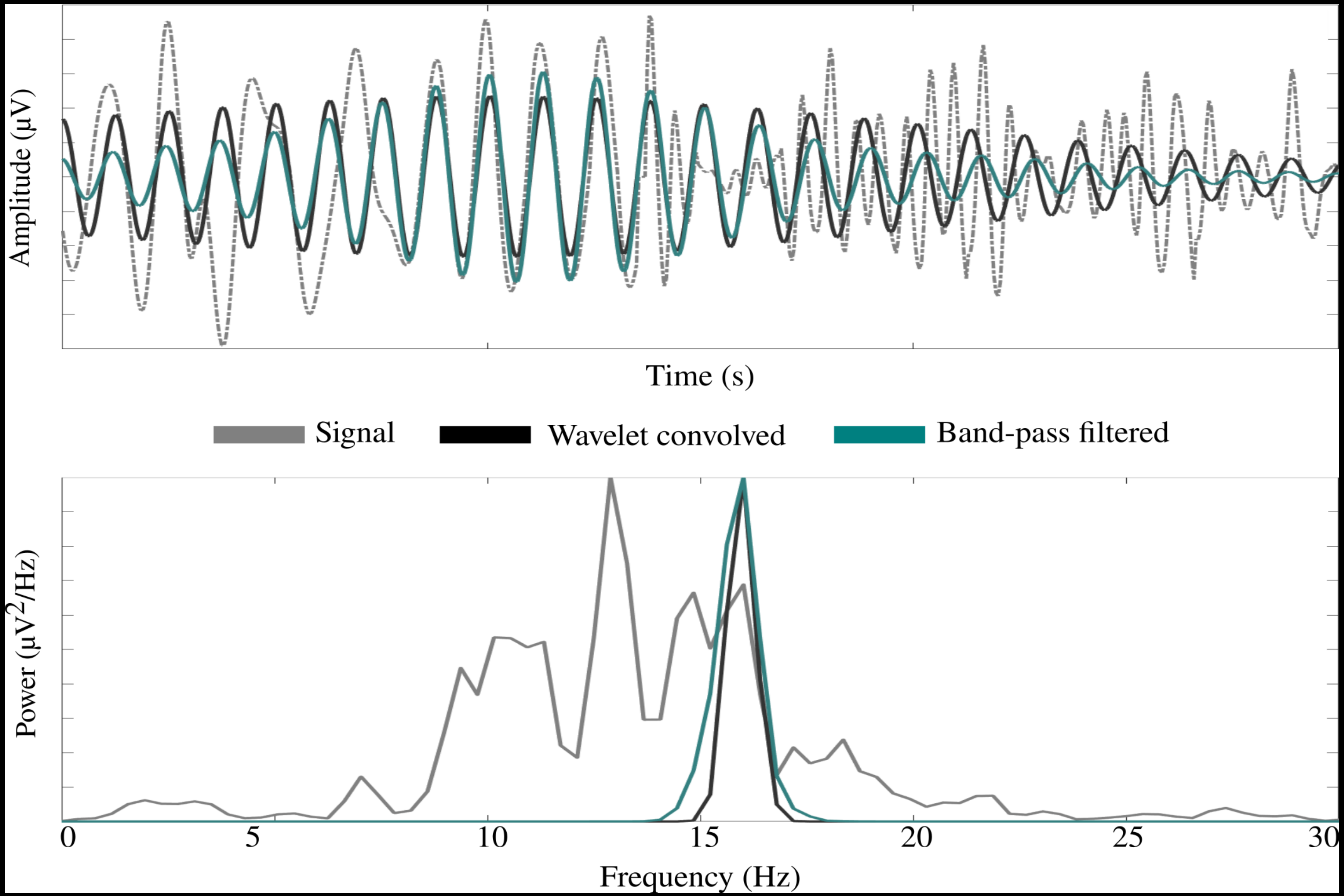




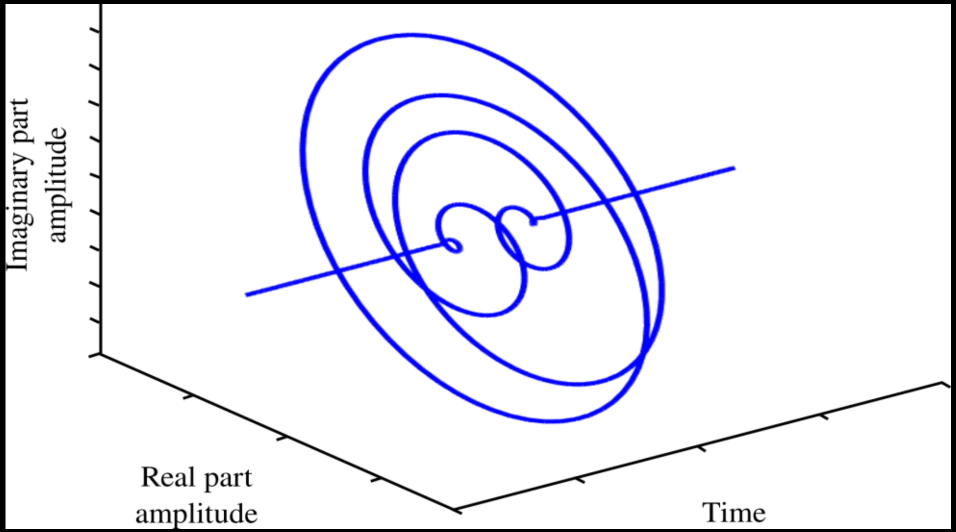
Wavelet peak frequency = sine frequency
Wavelet contains a range of frequencies

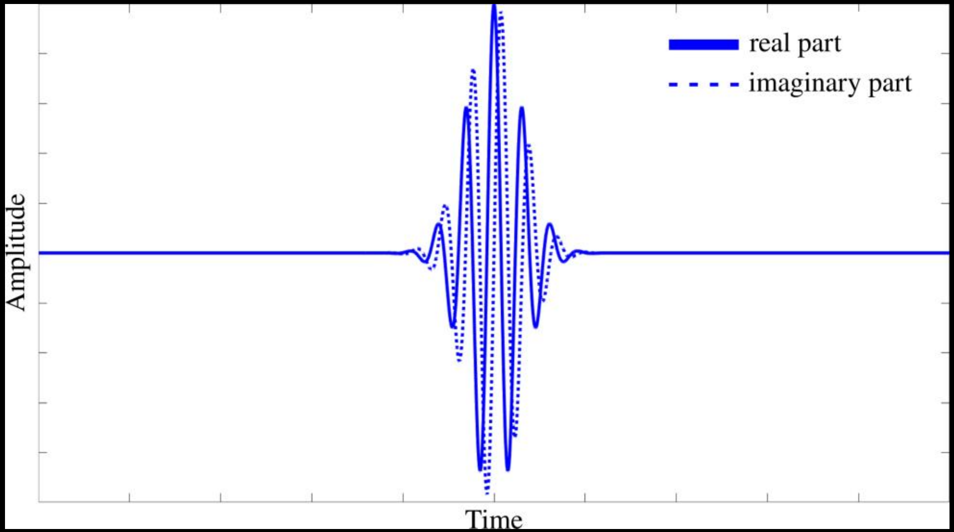
Mapping between a kernel and a signal. The signal is weighted through time by kernel properties
Frequency spectra are multiplied

Inter-Trial Phase Coherence (ITC or ITPC)

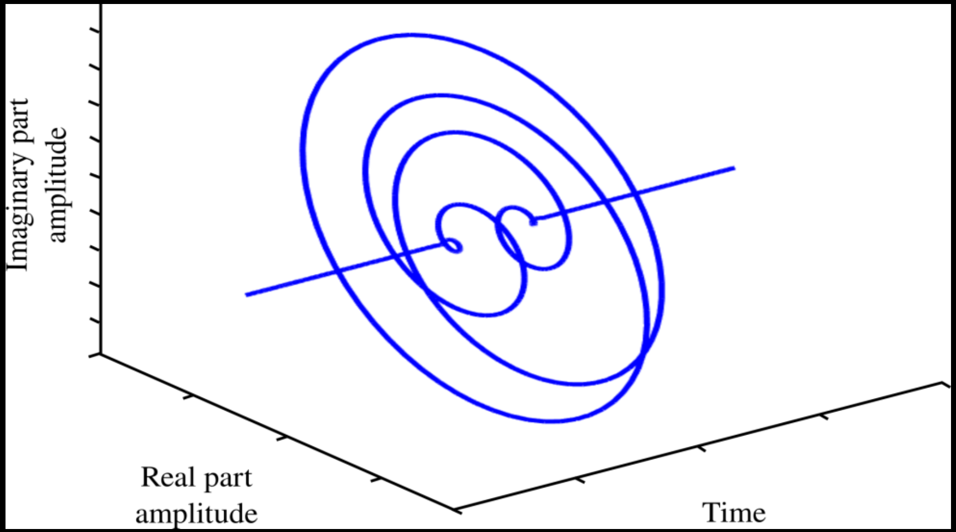


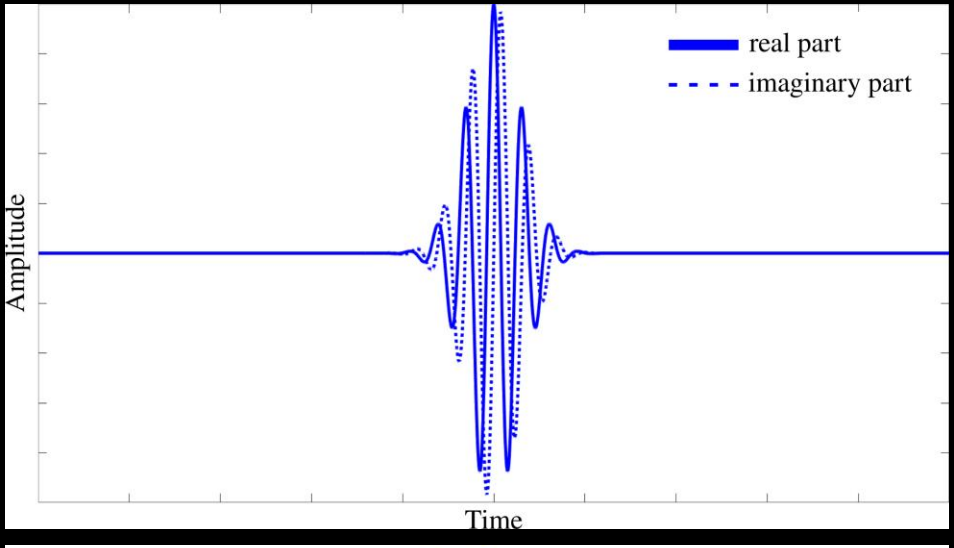
Morlet wavelet: phase and amplitude

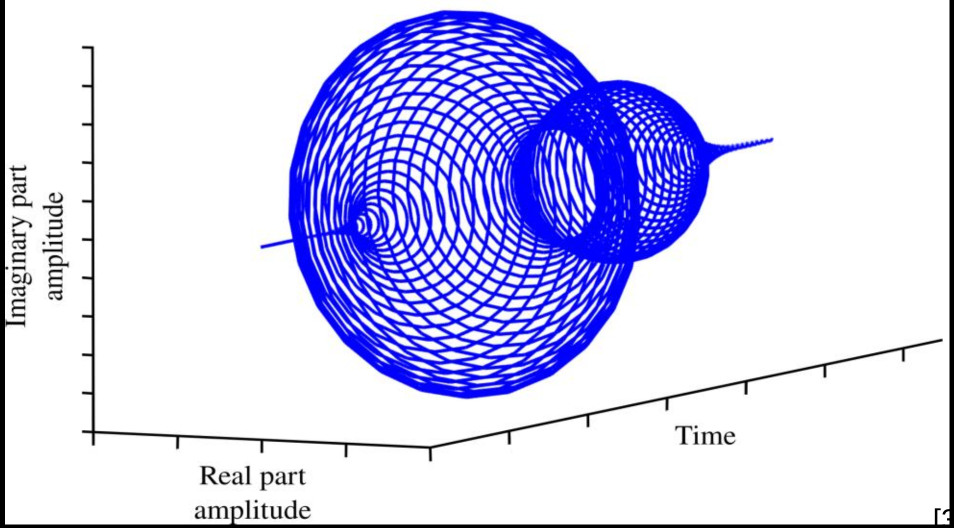




Morlet wavelet: phase and amplitude

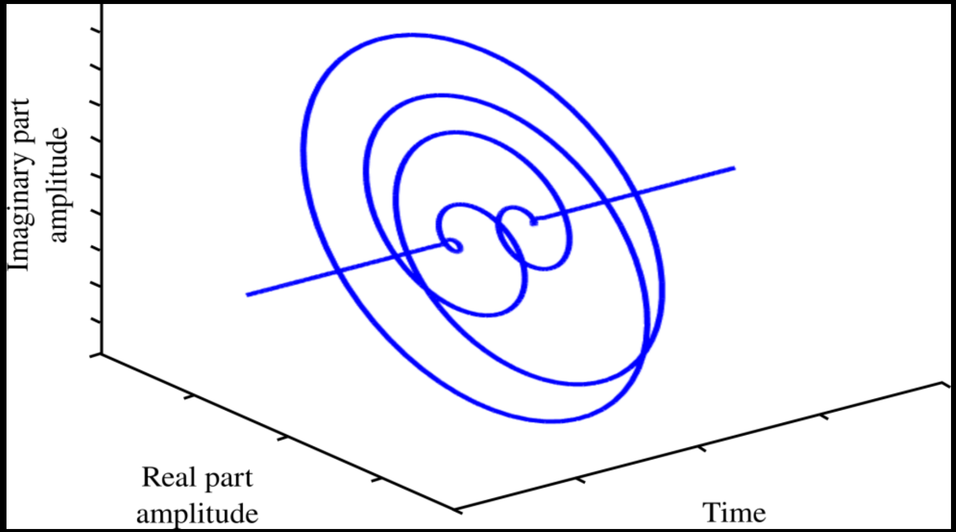


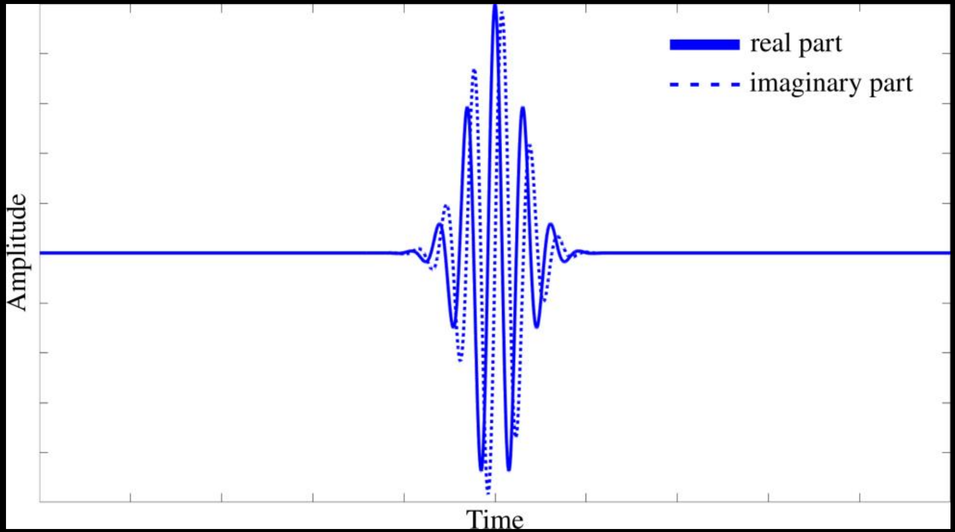


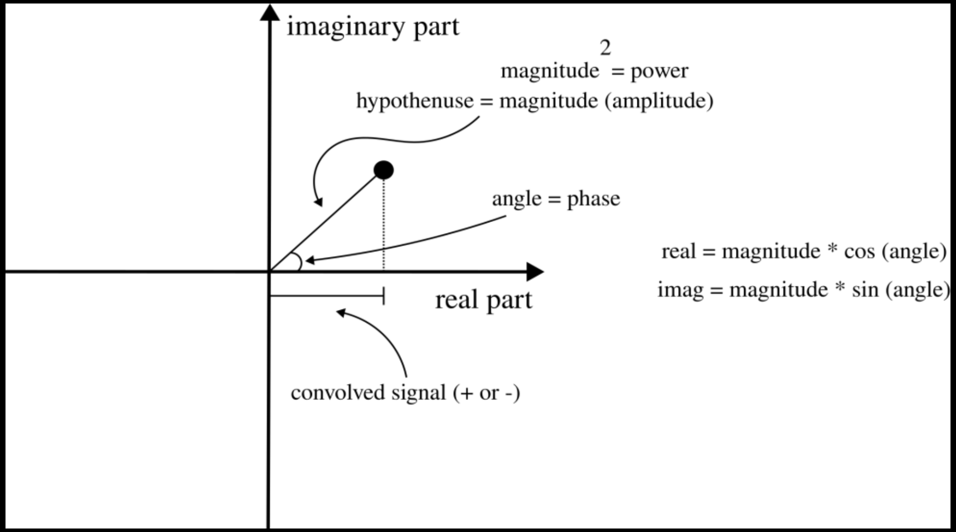


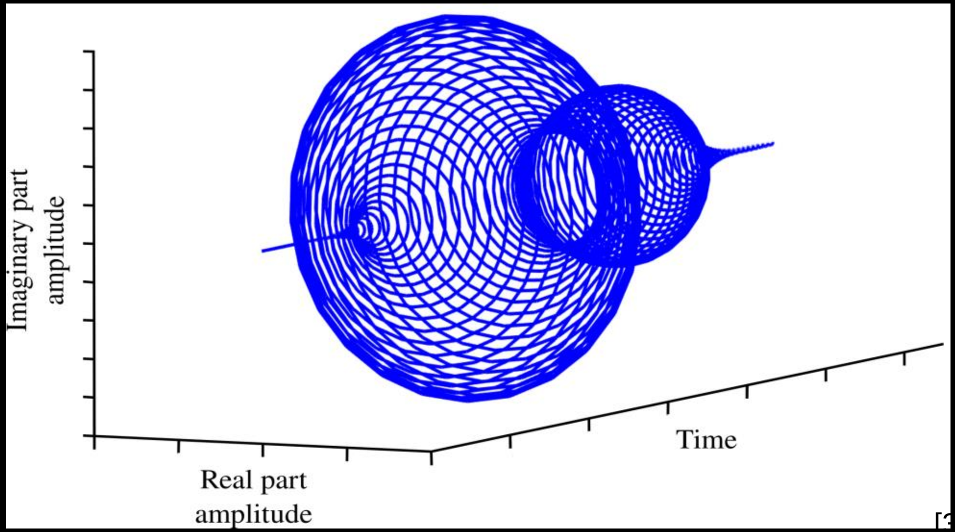
[3

Morlet wavelet: phase and amplitude

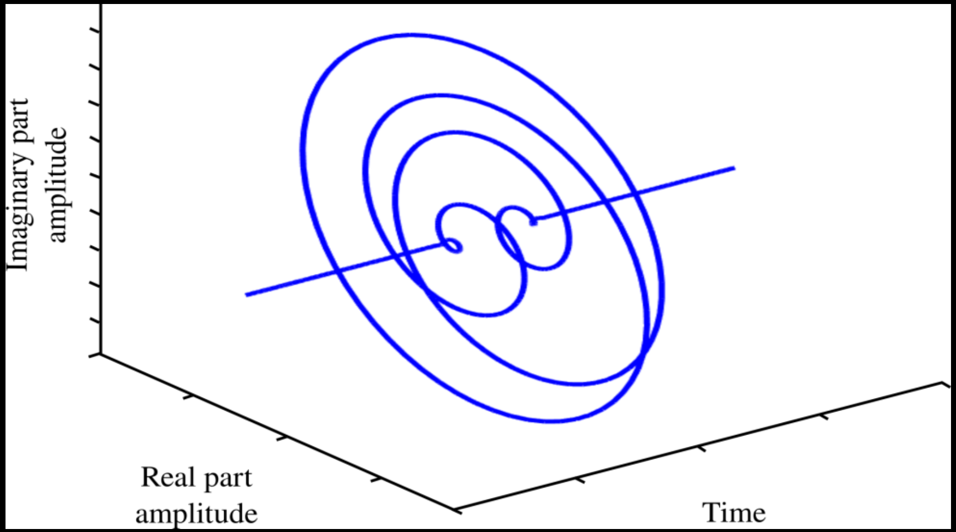


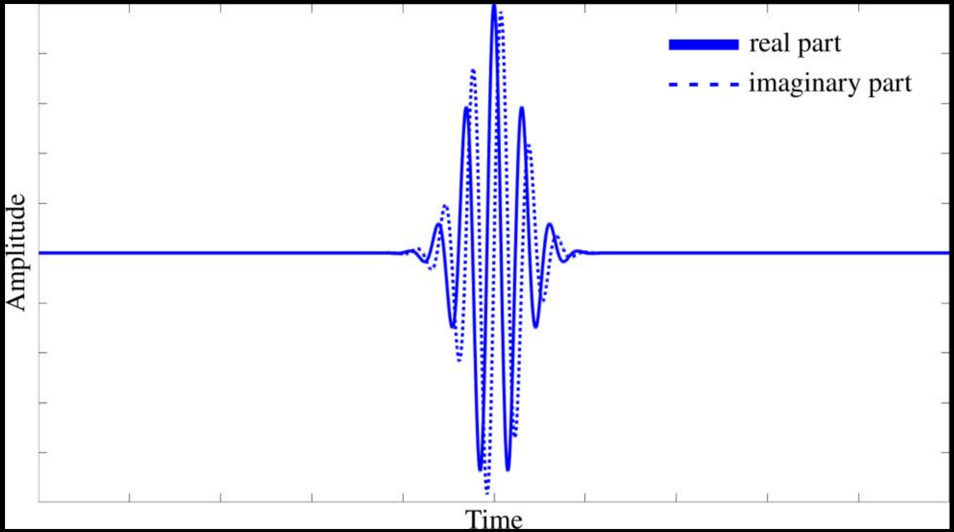


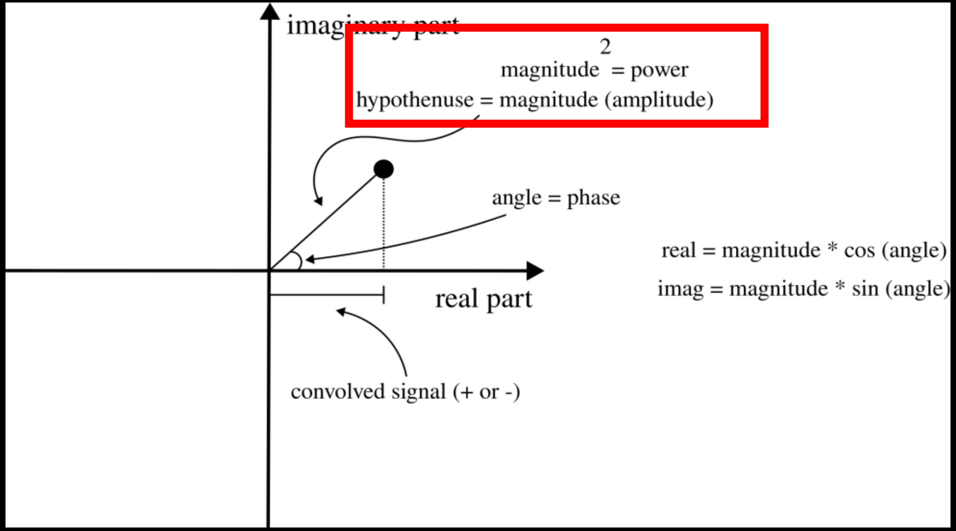


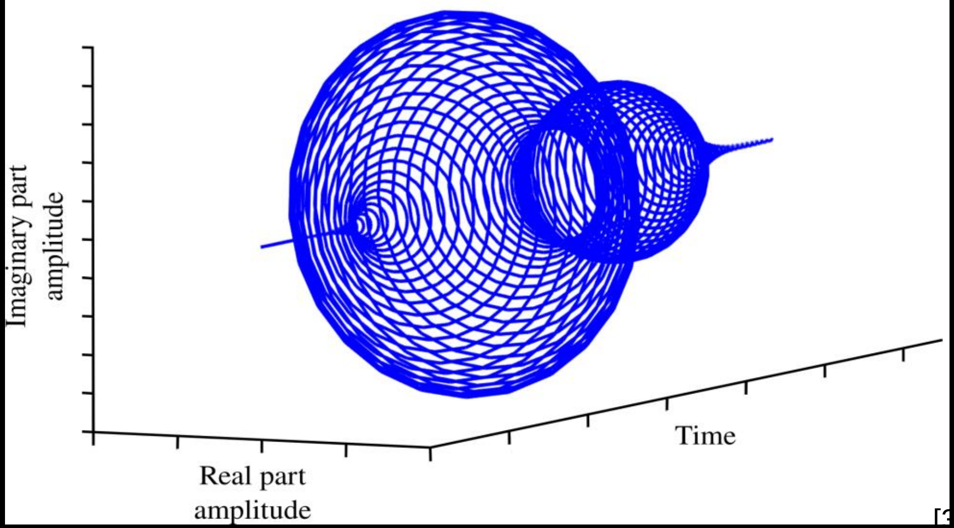


Morlet wavelet: phase and amplitude

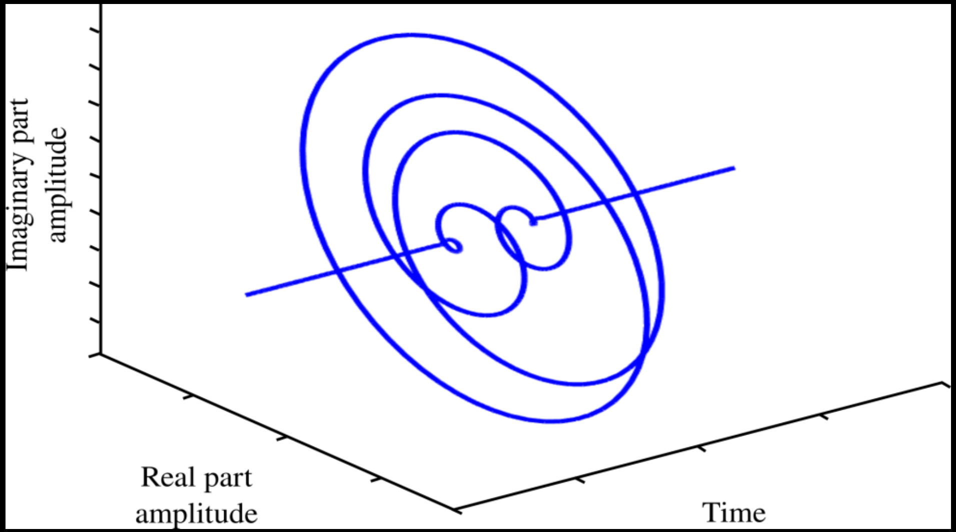


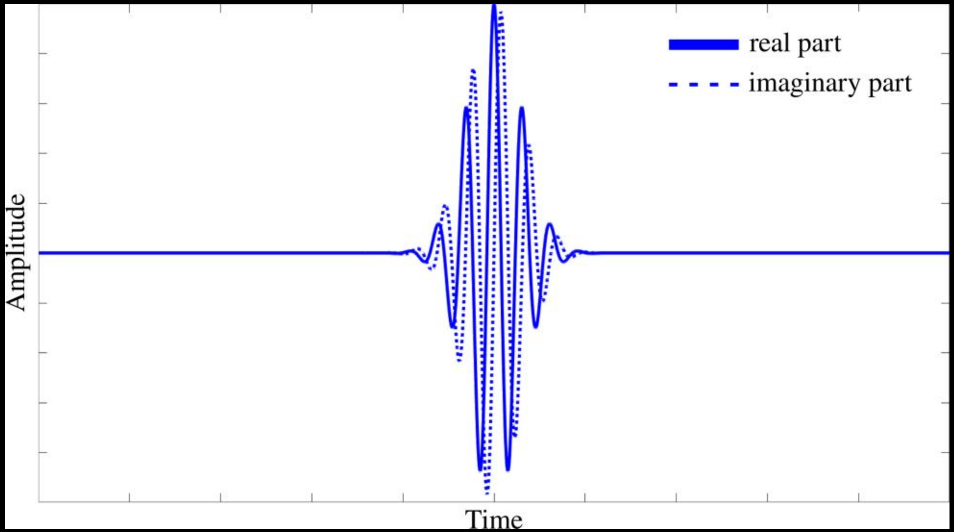


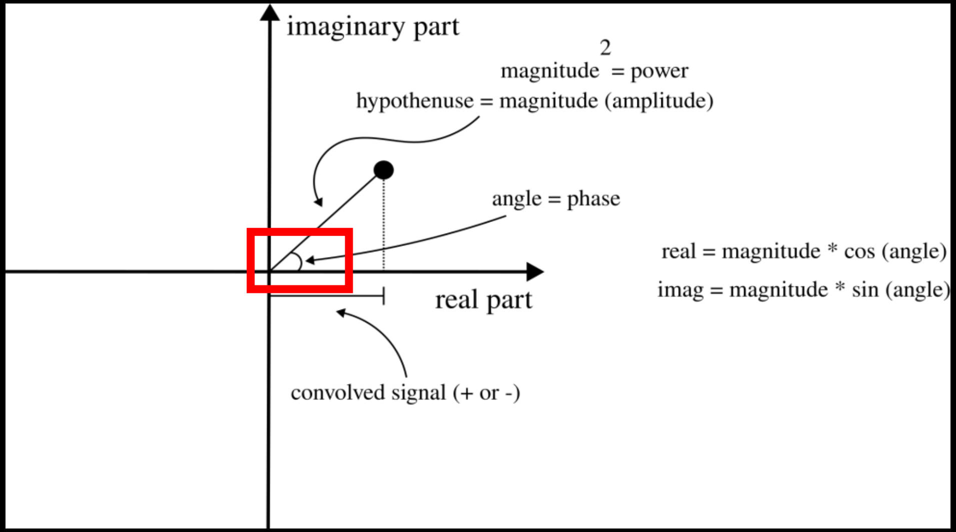


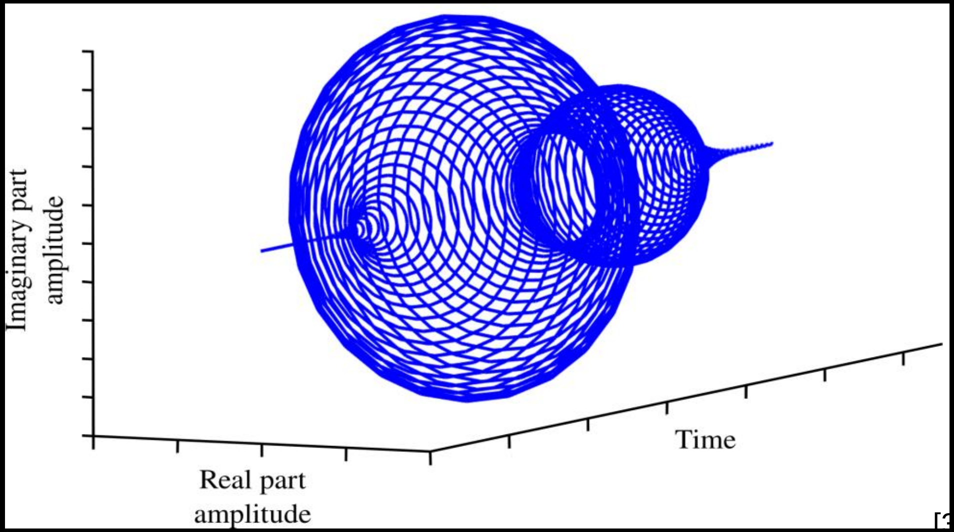


Morlet wavelet: phase and amplitude

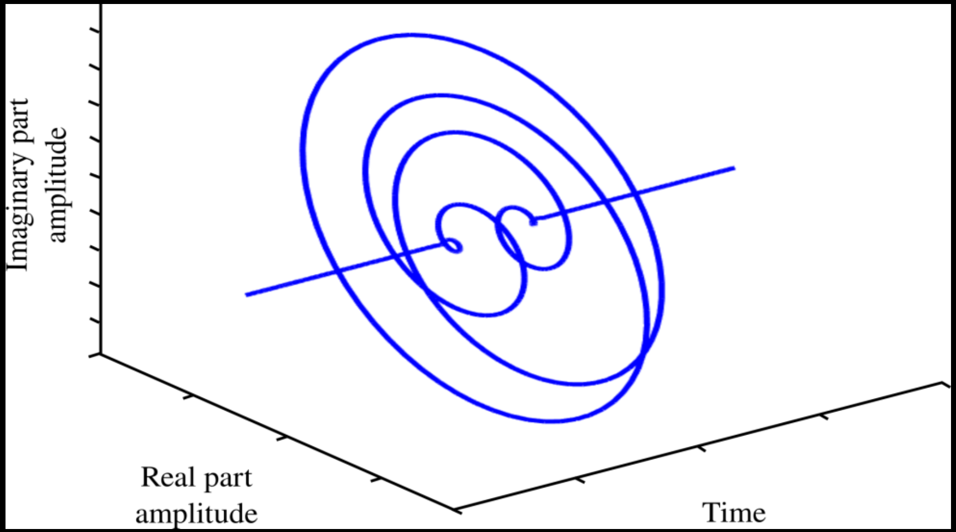


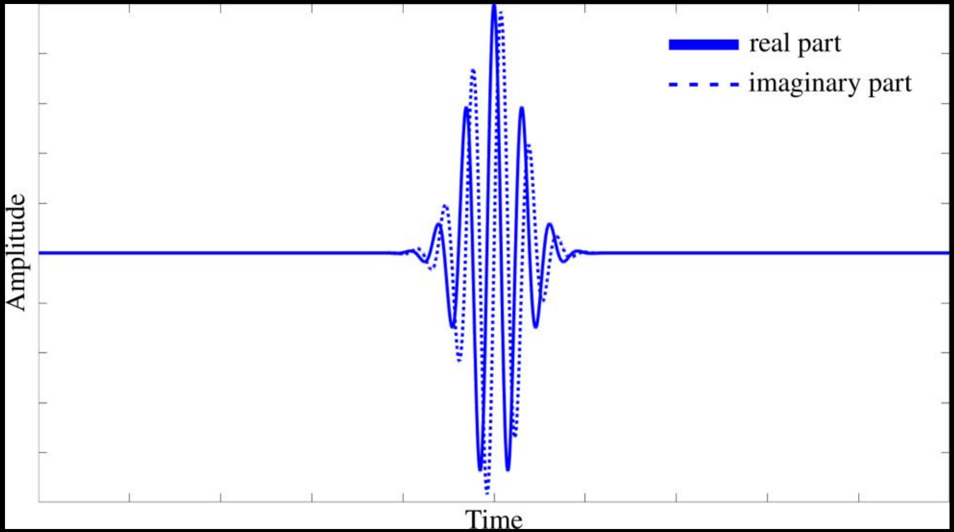


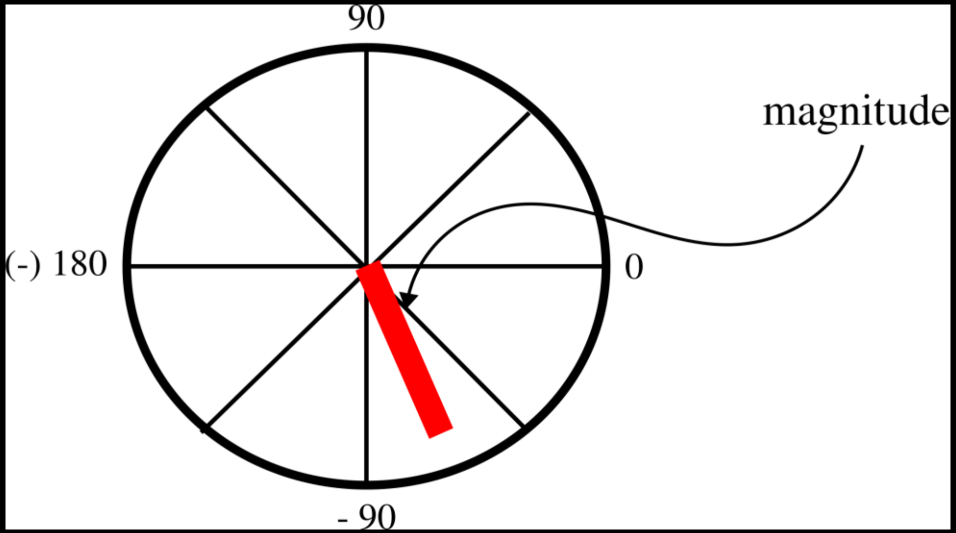


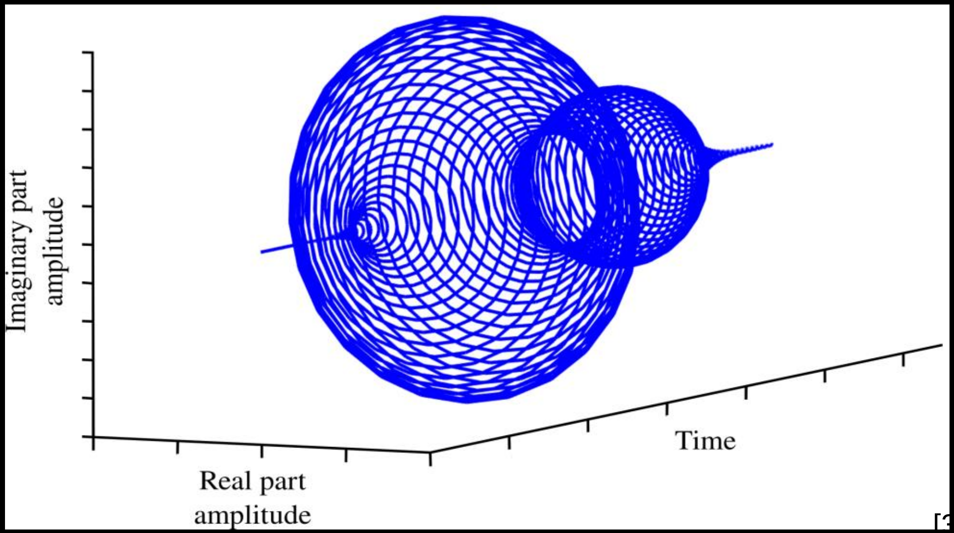


Morlet wavelet: phase and amplitude

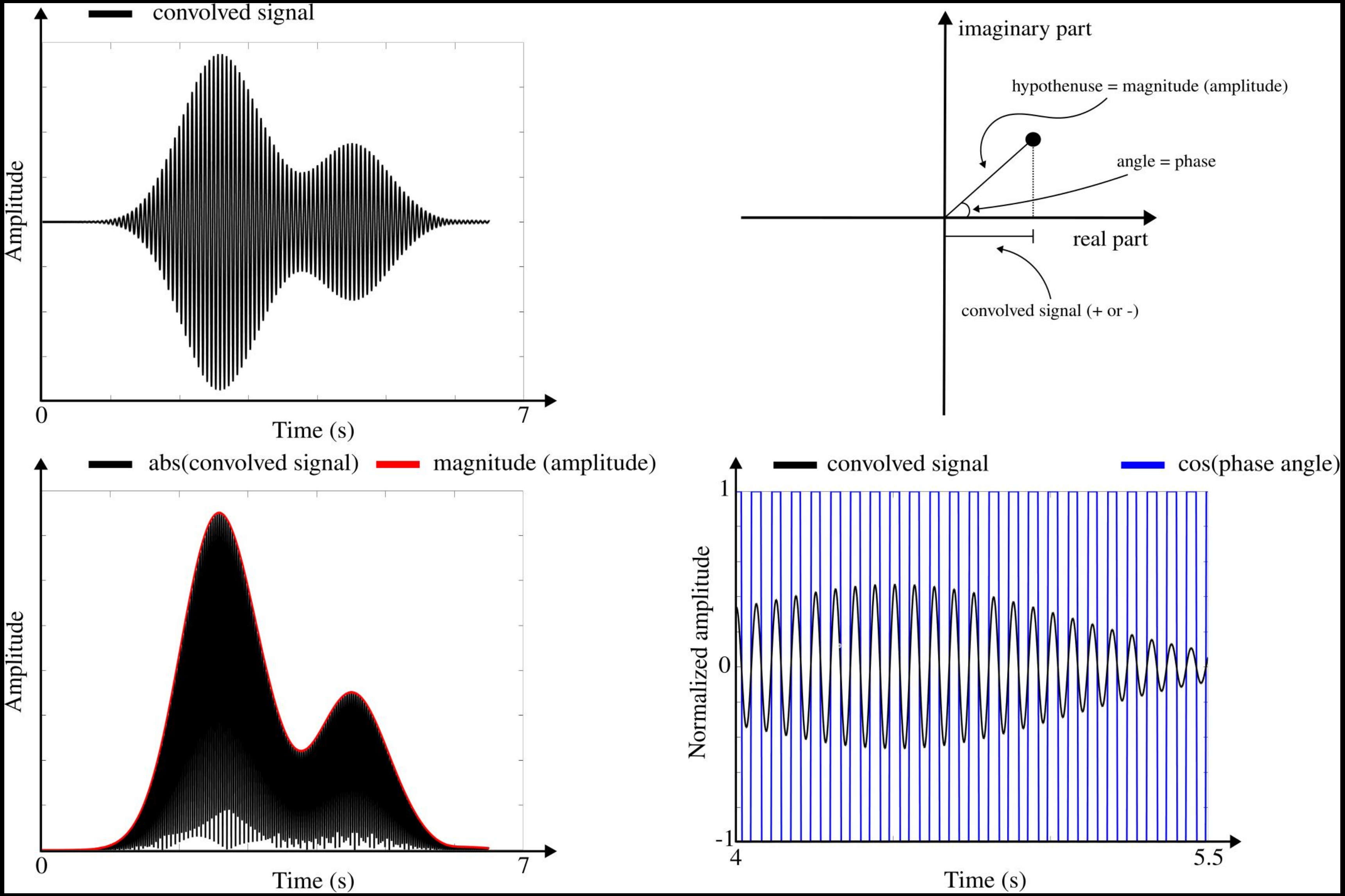




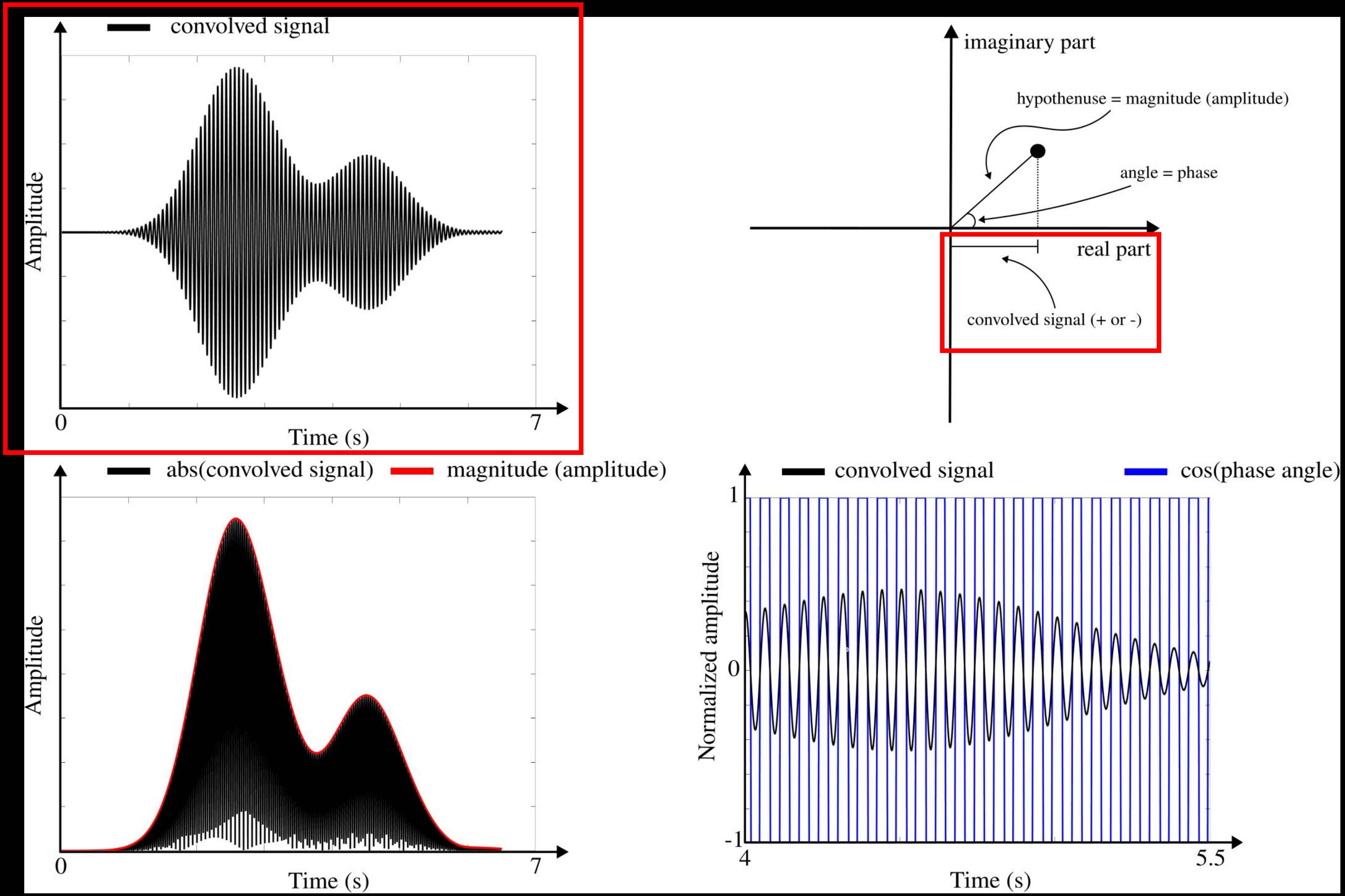




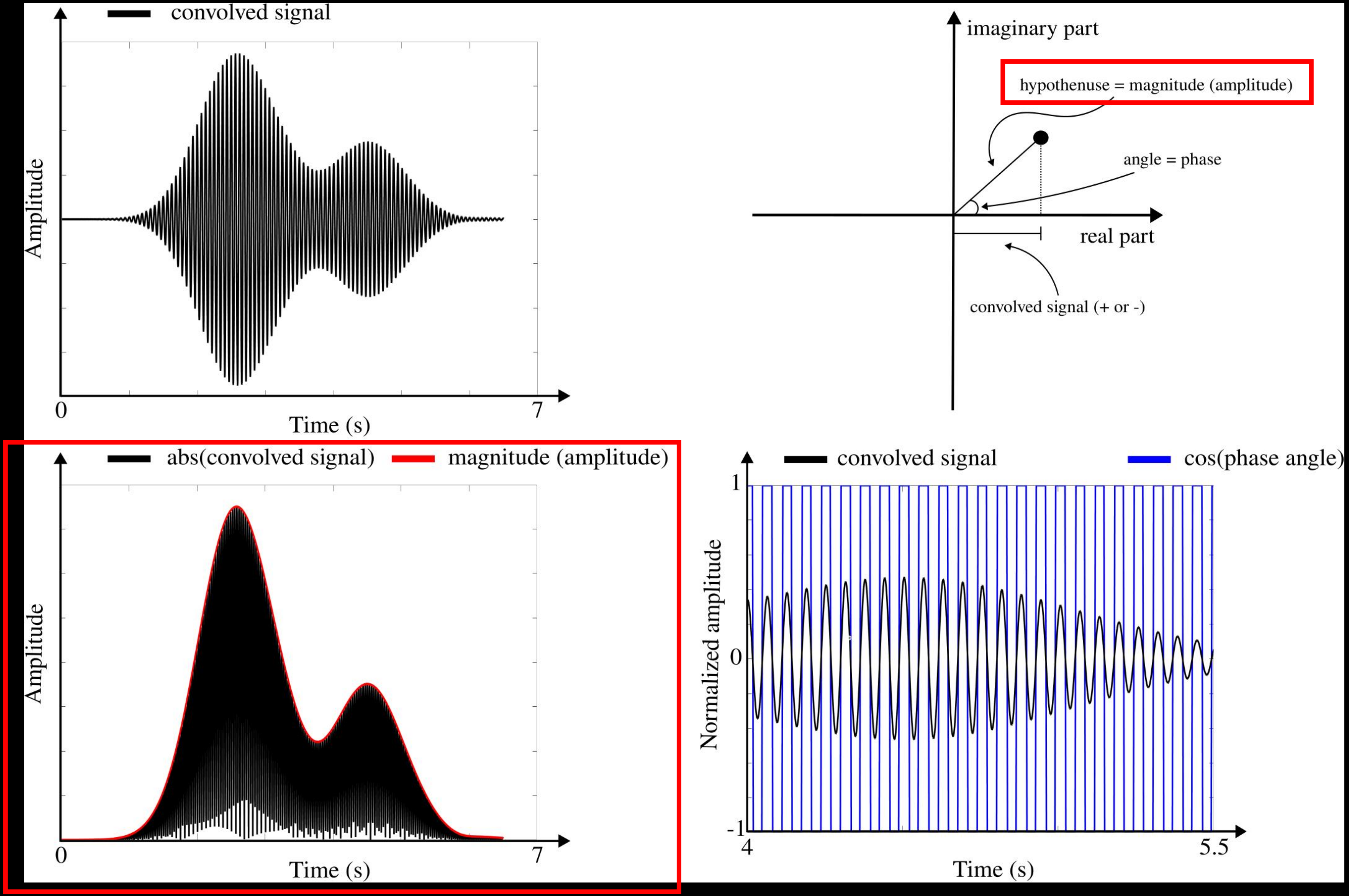
Morlet wavelet: phase and amplitude



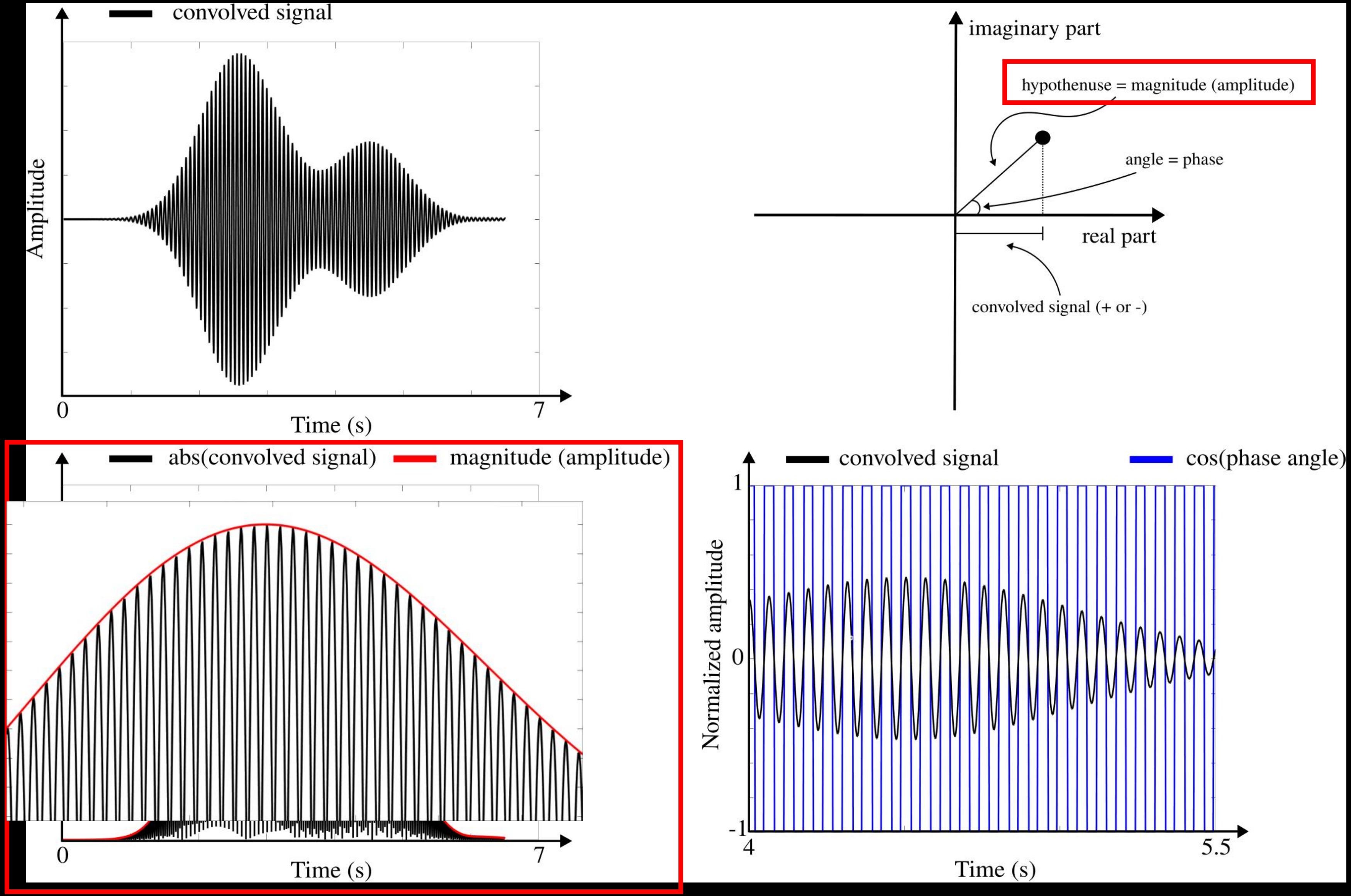
Morlet wavelet: phase and amplitude



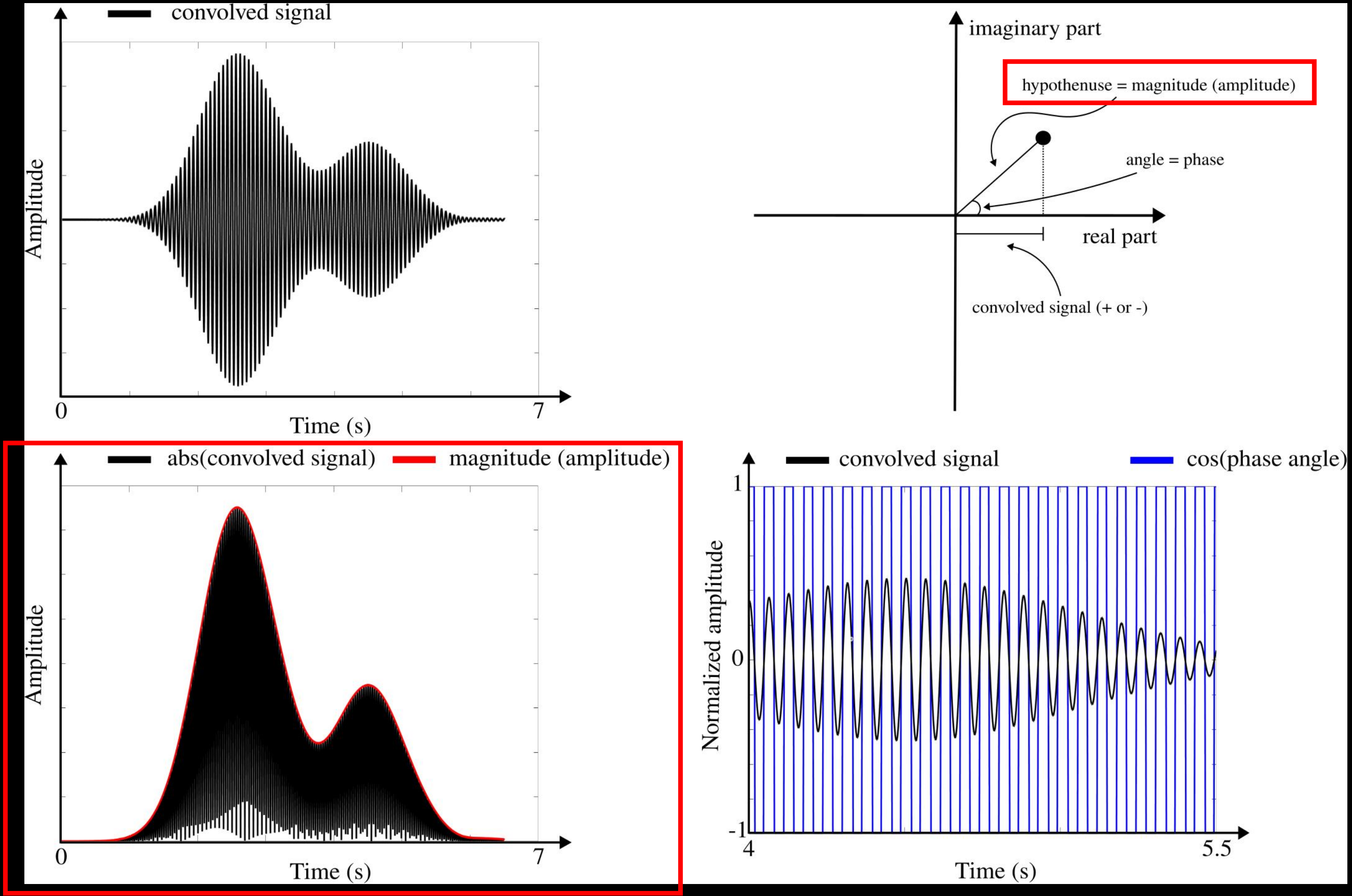
Morlet wavelet: phase and amplitude



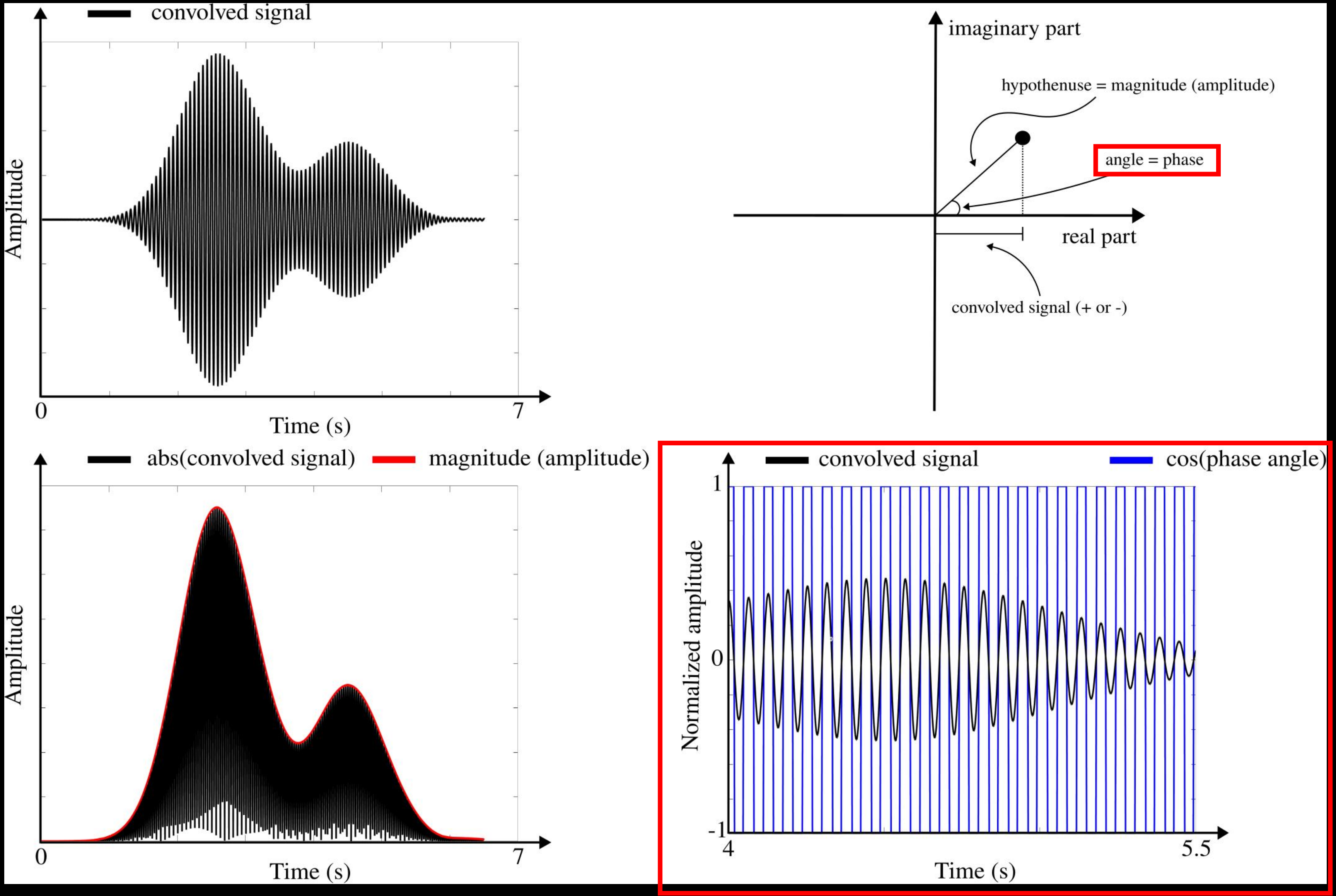
Morlet wavelet: phase and amplitude



Morlet wavelet: phase and amplitude

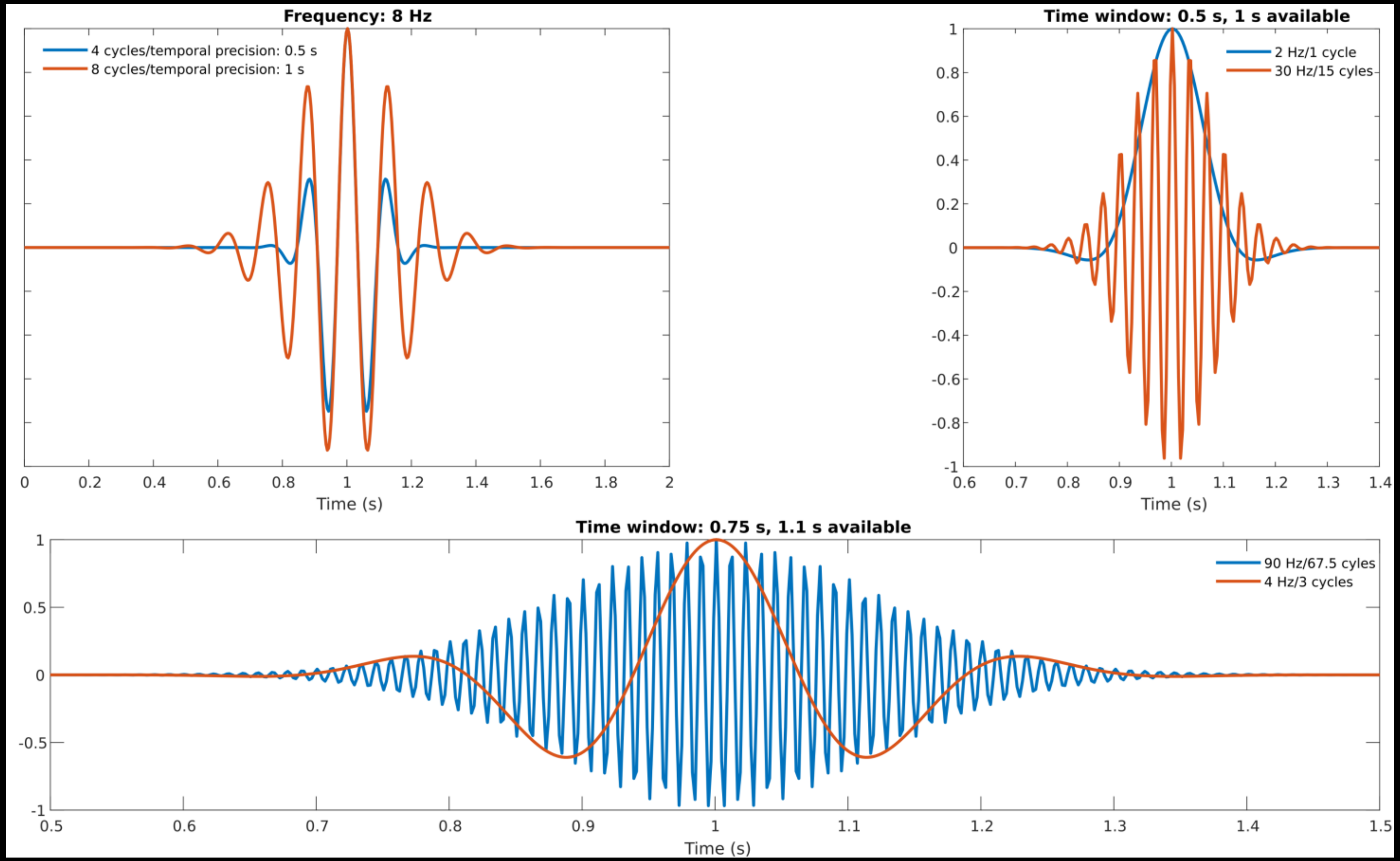


Morlet wavelet: phase and amplitude



Morlet wavelet: parameters

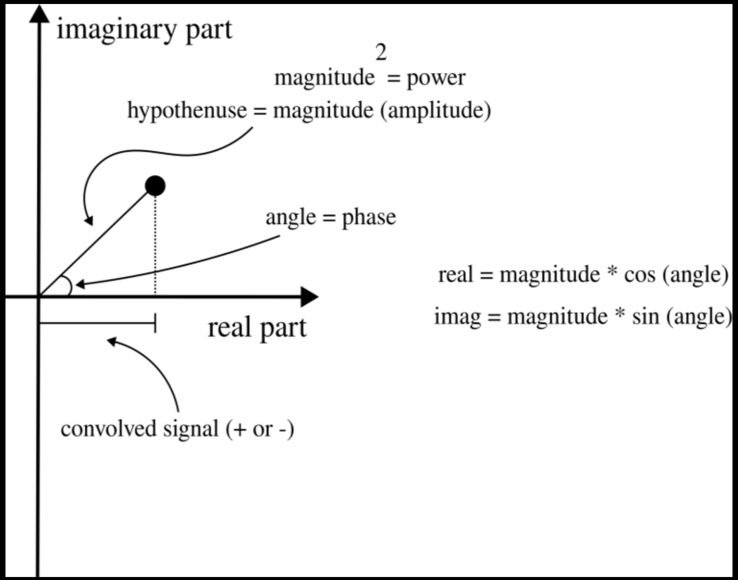
- **Lowest frequency (the signal must last for at least 2 cycles of the lowest frequency)**
- **Highest frequency (no more than the Nyquist frequency)**
- **How long should the wavelet be? (time)**
  - ❑ Lowest frequencies must taper zero
- **How many cycles?**
  - ❑ Defines time/frequency precision
  - ❑ Temporal precision: (1/freq)*nb_cycles



18

Filter-Hilbert: basic concepts

Hilbert Transform: extract the imaginary part of a real-valued signal (here: EEG or speech)



- Computes Fourier Transform
- Multiply coefficients by *i*
  - ✓ M * cos (angle) → *i* M * cos (angle)

- 0 > Positive frequencies < Nyquist
- Negative frequencies > Nyquist (mirror)

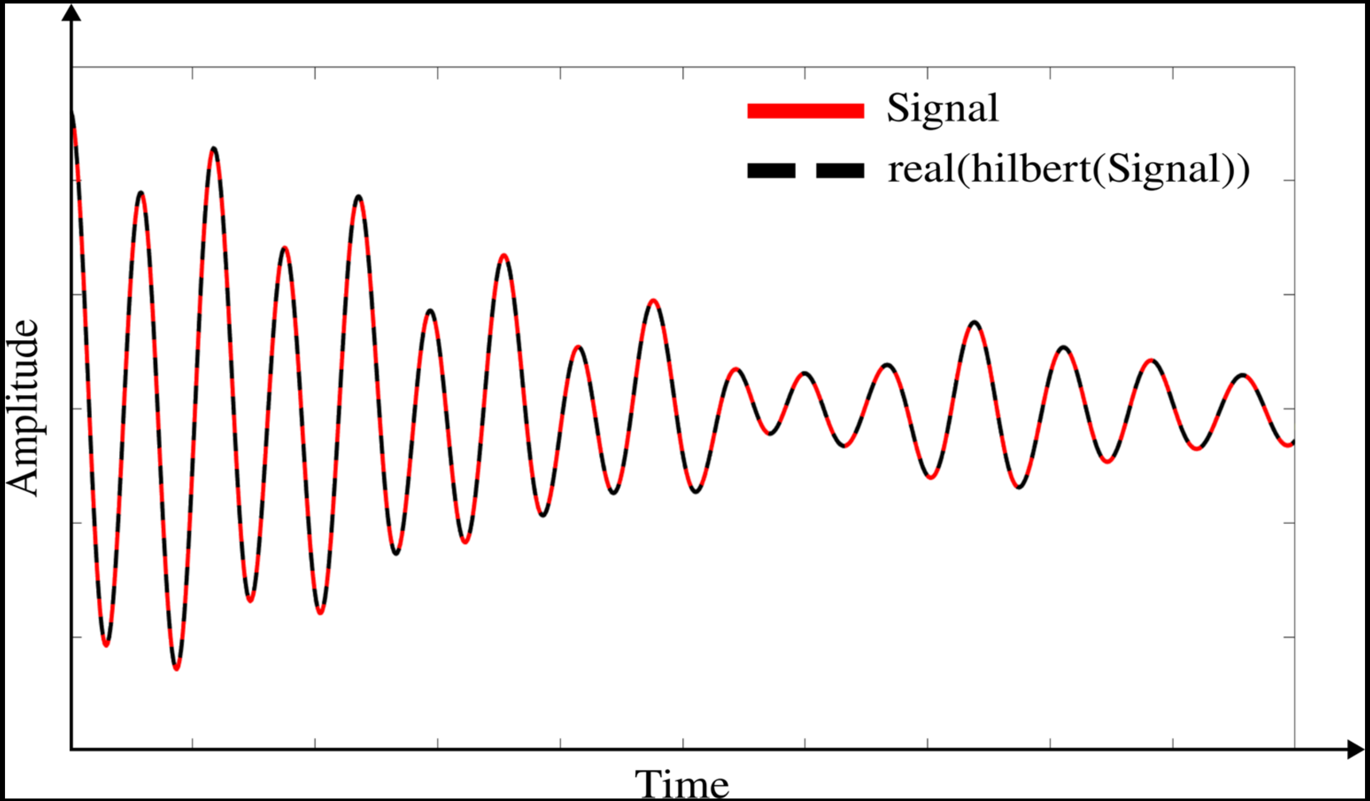
- Converts *i* M * cos (angle) to *i* M * sin (angle): difference between cos and sin is 90° (or pi/2)
  - ✓ Rotate positive frequencies 90° (1/4 cycle) counterclockwise (multiply by -*i*)
  - ✓ Rotate negative frequencies 90° (1/4 cycle) clockwise (multiply by *i*)

- Adds rotated coefficients to the original coefficients (from step 1)
  - ✓ Positive coefficients double (*i*M sin(angle) * (-*i*) → *i**(-*i*) = 1)
  - ✓ Negative coefficients become 0 (does not mirror anymore) (→ *i**(*i*) = -1)

- Computes the Inverse Fourier Transform
  - ✓ Outputs the analytical signal that is complex because the complex part of positive and negative frequencies do not cancel each other

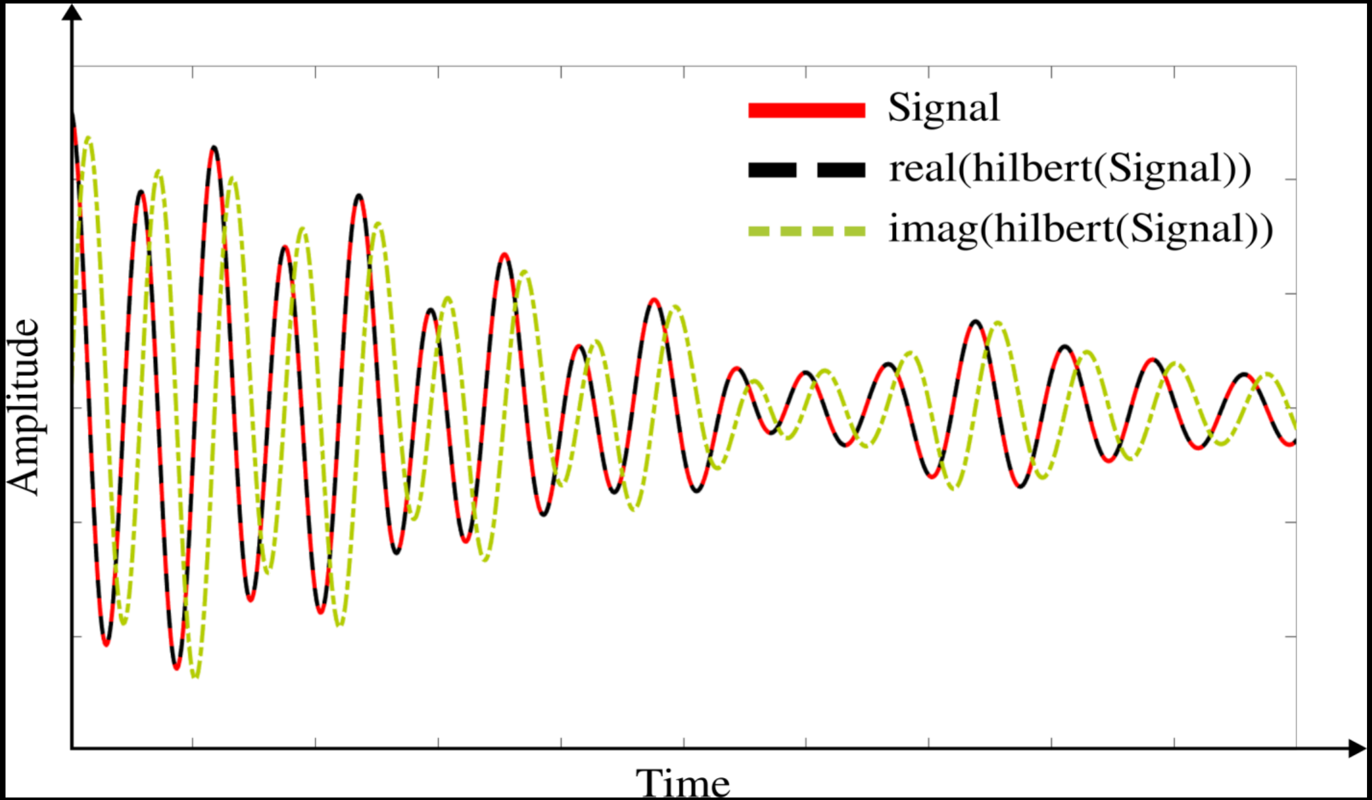
Filter-Hilbert: basic concepts

Hilbert Transform: extract the imaginary part of a real-valued signal (here: EEG or speech)

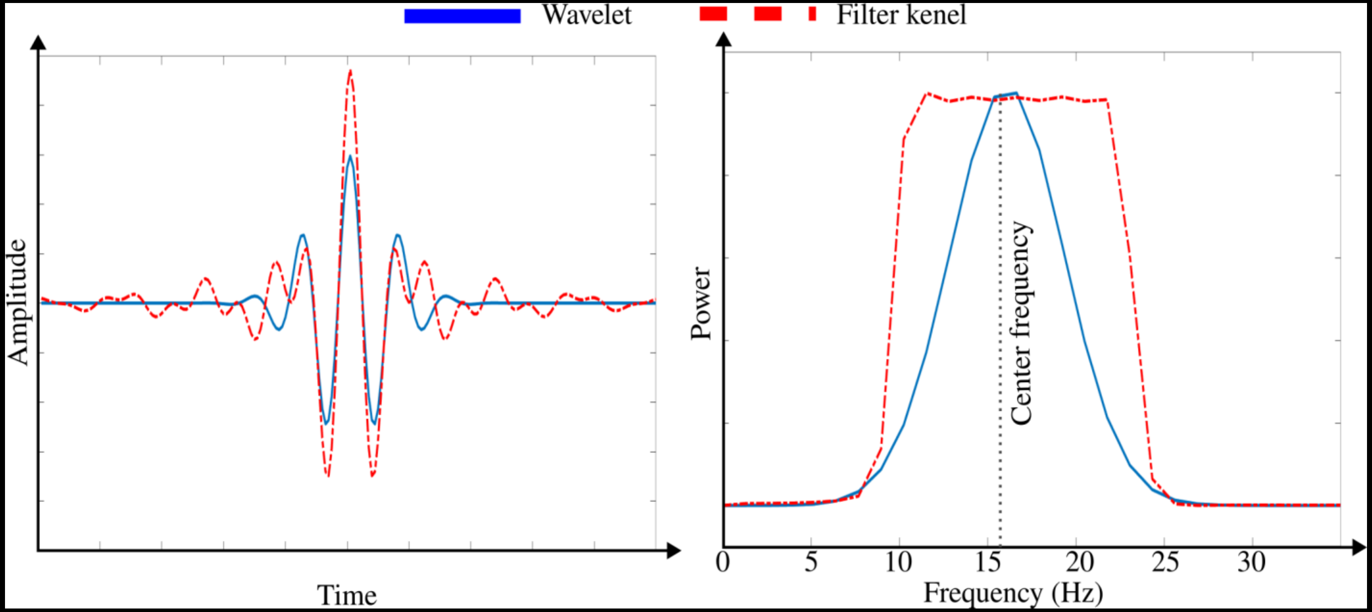


Filter-Hilbert: basic concepts

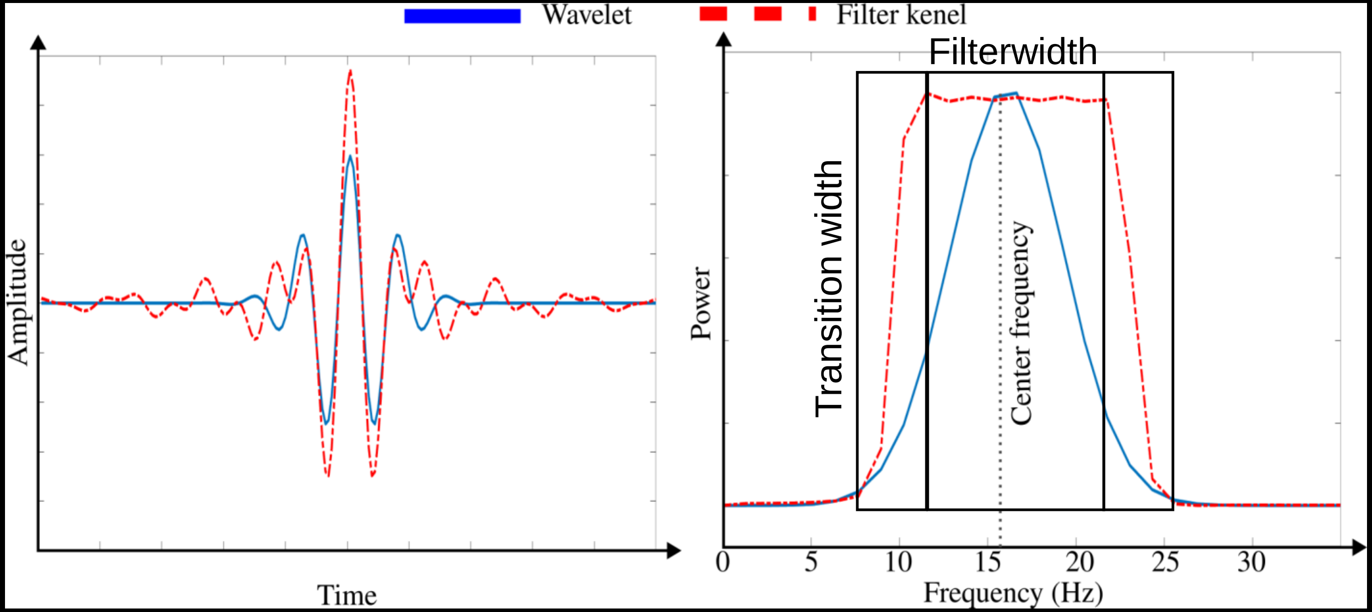
Hilbert Transform: extract the imaginary part of a real-valued signal (here: EEG or speech)



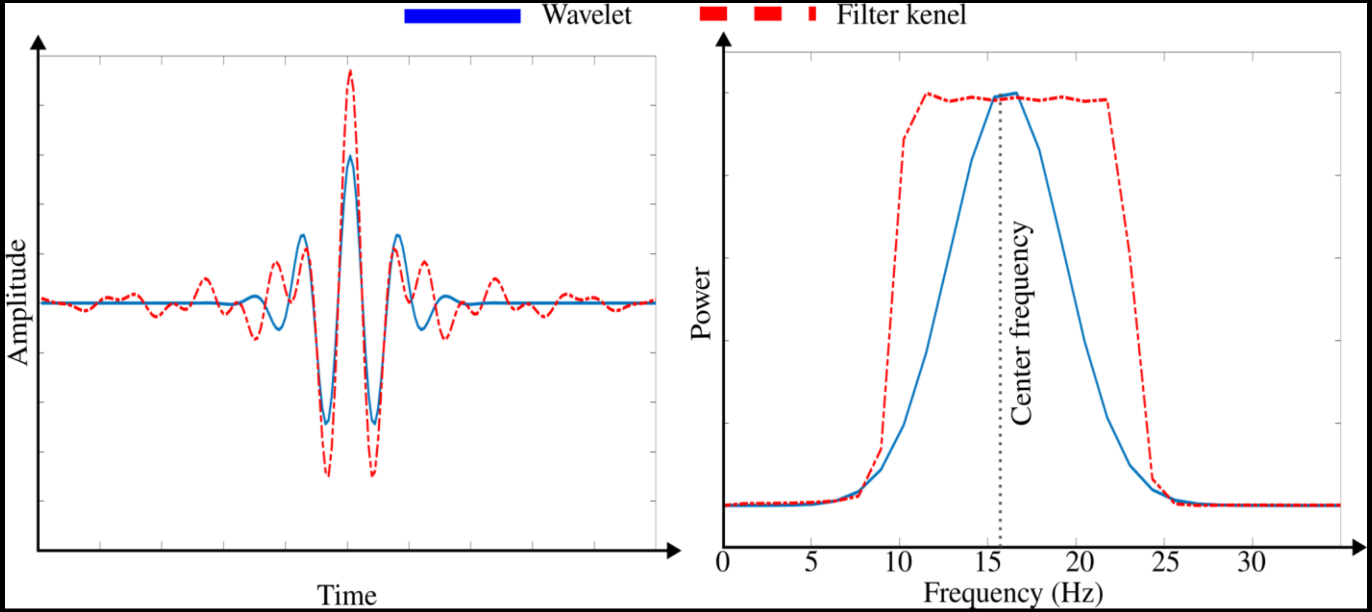
Filter-Hilbert: filtering before Hilbert transform



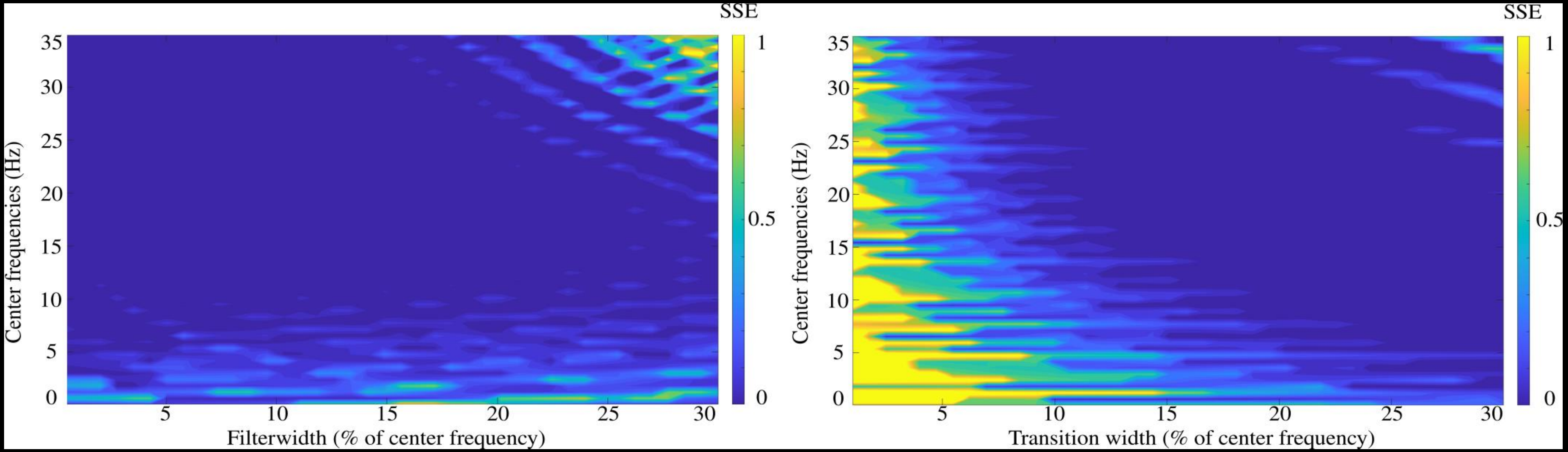
Filter-Hilbert: filtering before Hilbert transform



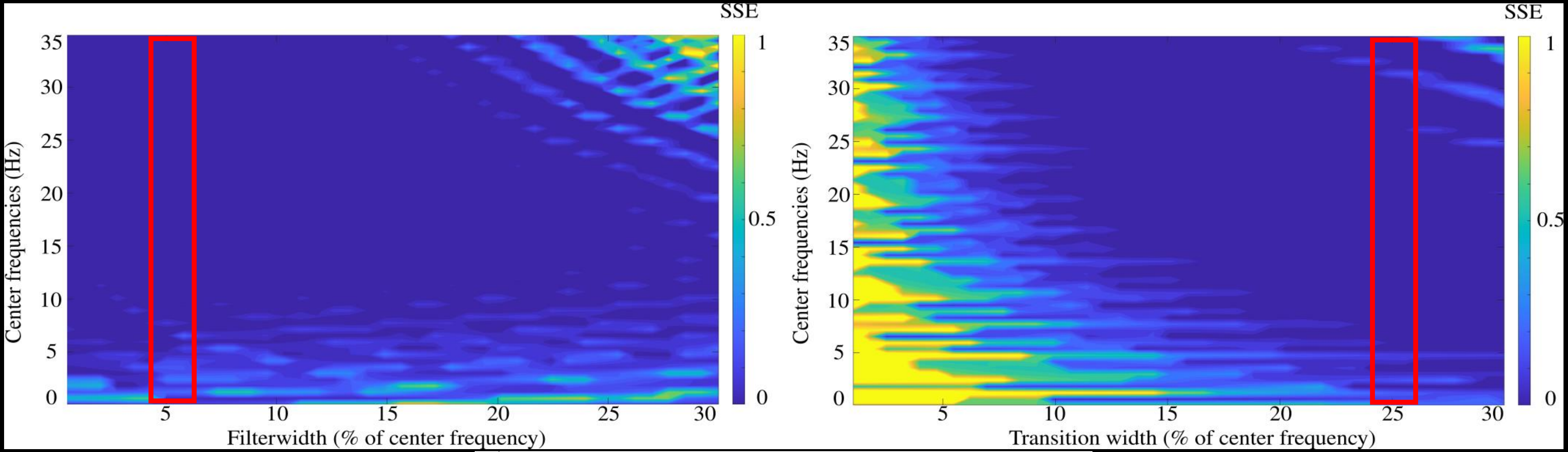
Filter-Hilbert: filtering before Hilbert transform



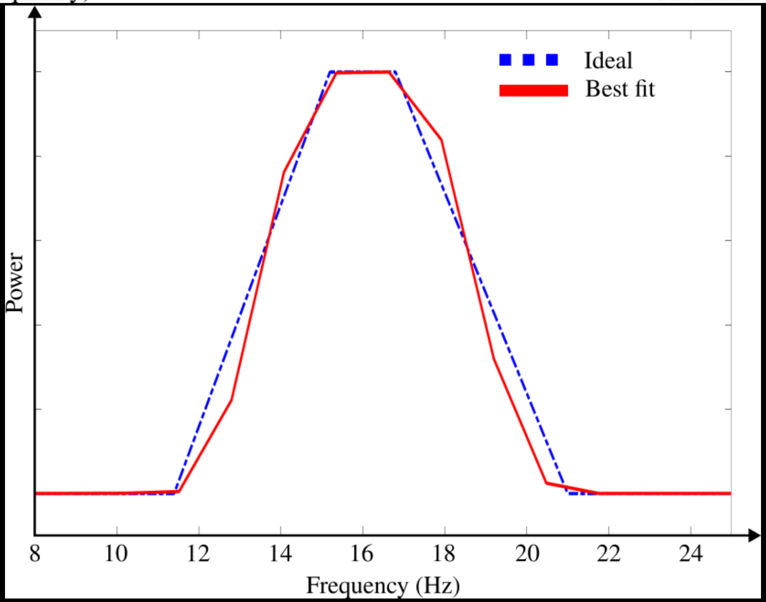
Filter-Hilbert: filtering before Hilbert transform



Filter-Hilbert: filtering before Hilbert transform



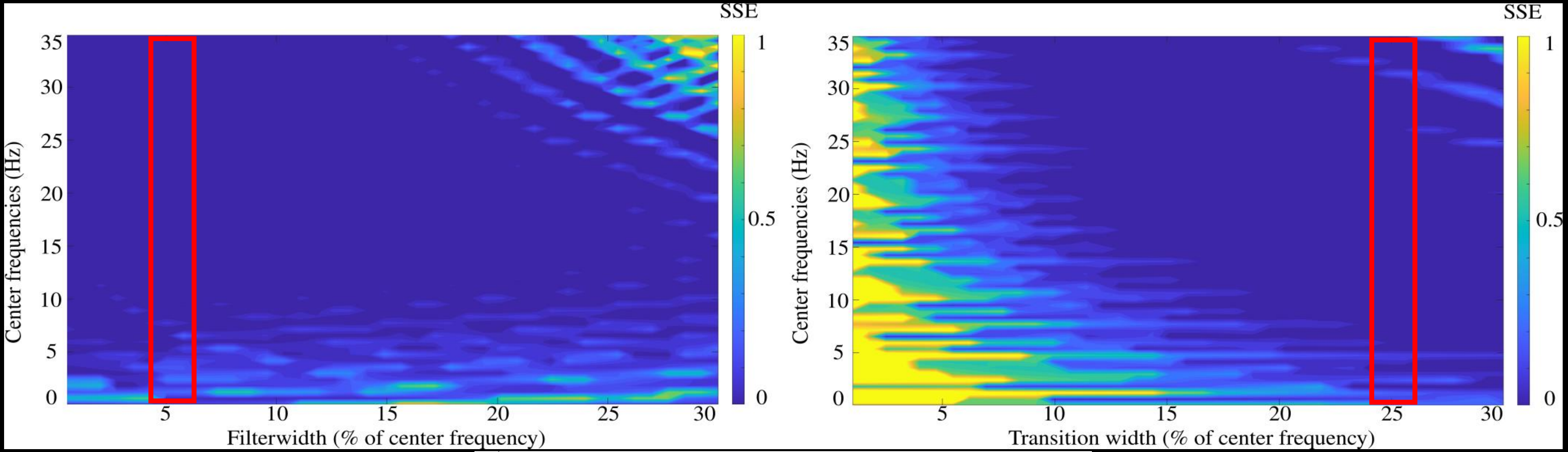
SSE=6.3e-4



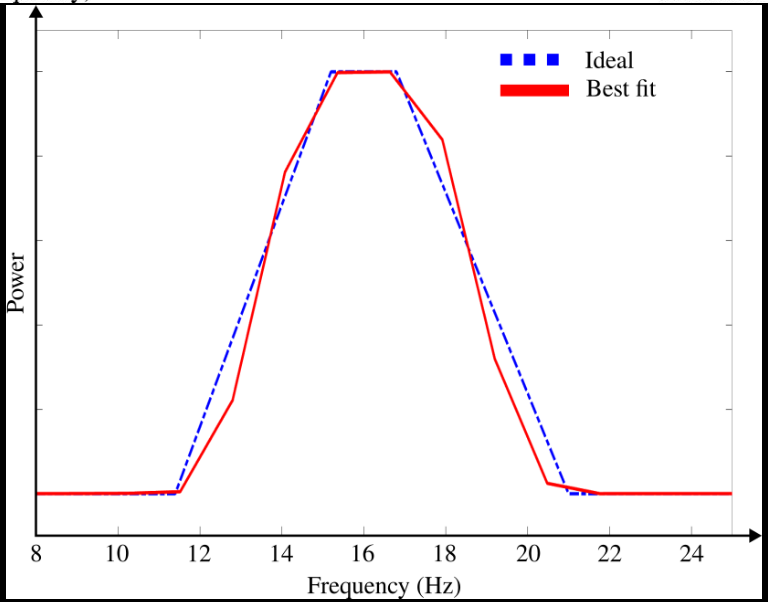
$$SSE = \sum_{i=1}^{n} (ideal_i - actual_i)^2$$

Where *SSE* is the sum of squared error,
n is the number of frequencies in the ideal filter,
Ideal and *actual* refer to the power spectra

Filter-Hilbert: filtering before Hilbert transform



SSE=6.3e-4



$$SSE = \sum_{i=1}^{n} (ideal_i - actual_i)^2$$

SSE:

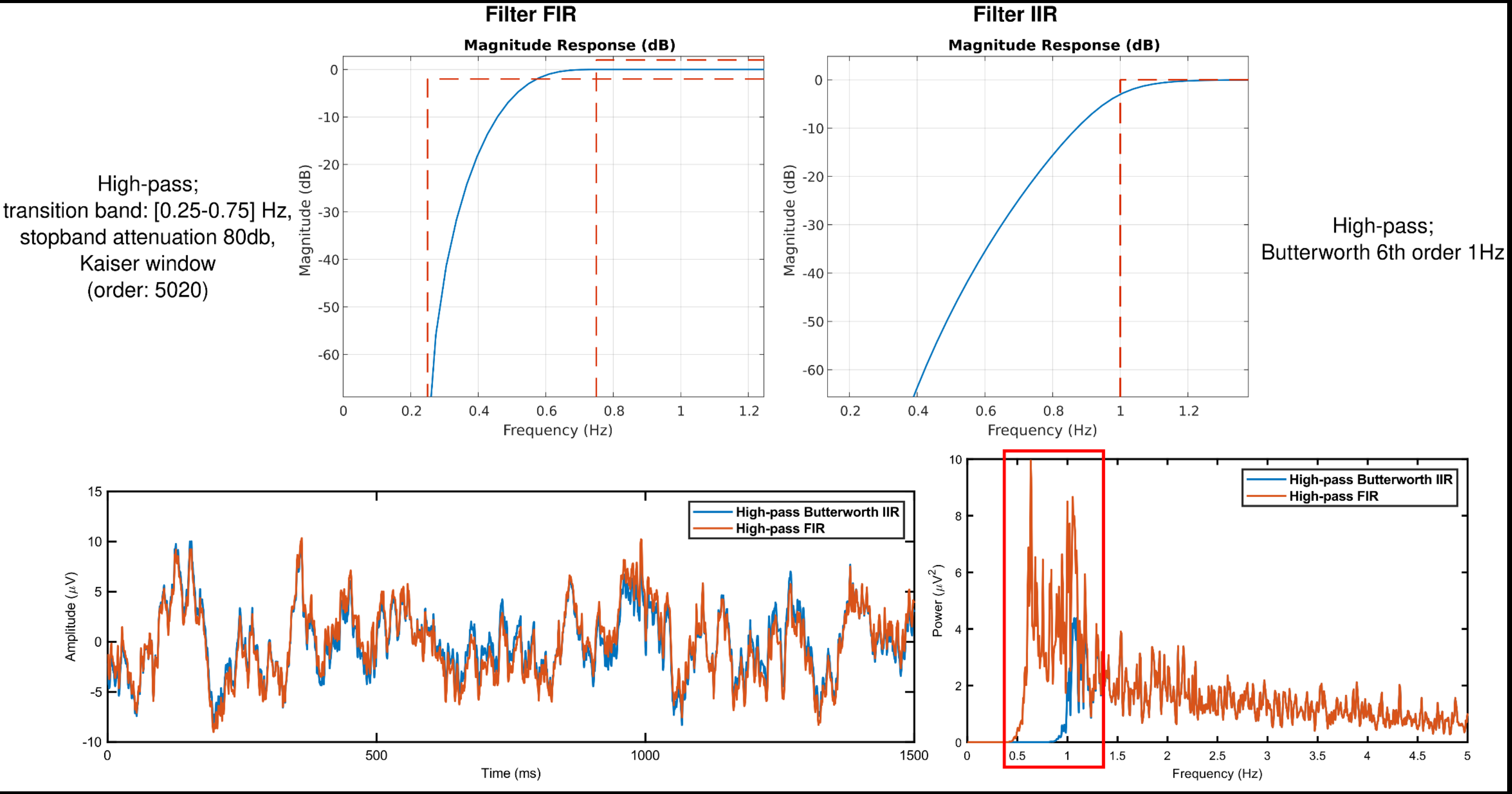
- Approximates 0
- Lower than 1

Short-term Fourier decomposition

Fast Fourier Transform on brief segments of the data (convolution by sine wave)

- Taper every short segments by a window to reduce edges artifacts (e.g., Hanning; must taper to 0).
- Take the average of several frequency bins surrounding each requested frequency to increase the signal-to-noise ratio (in case of better frequency resolution than needed).
- Shorter time segments → better temporal precision at the expanse of spectral precision, and vice versa.
  - The larger the time segment, the more frequencies can be extracted, and thus the greater the frequency precision (the number of frequencies returned by the FFT is equal to $N/2 + 1$).
  - Duration of at least 2 cycles of the lowest frequency
- Loss of signal due to tapering → overlap of time segments to countereffect (increase computation).
- STF outputs the phase parameter for each time window and frequency (phase value when t=0) instead of the phase vector through time.

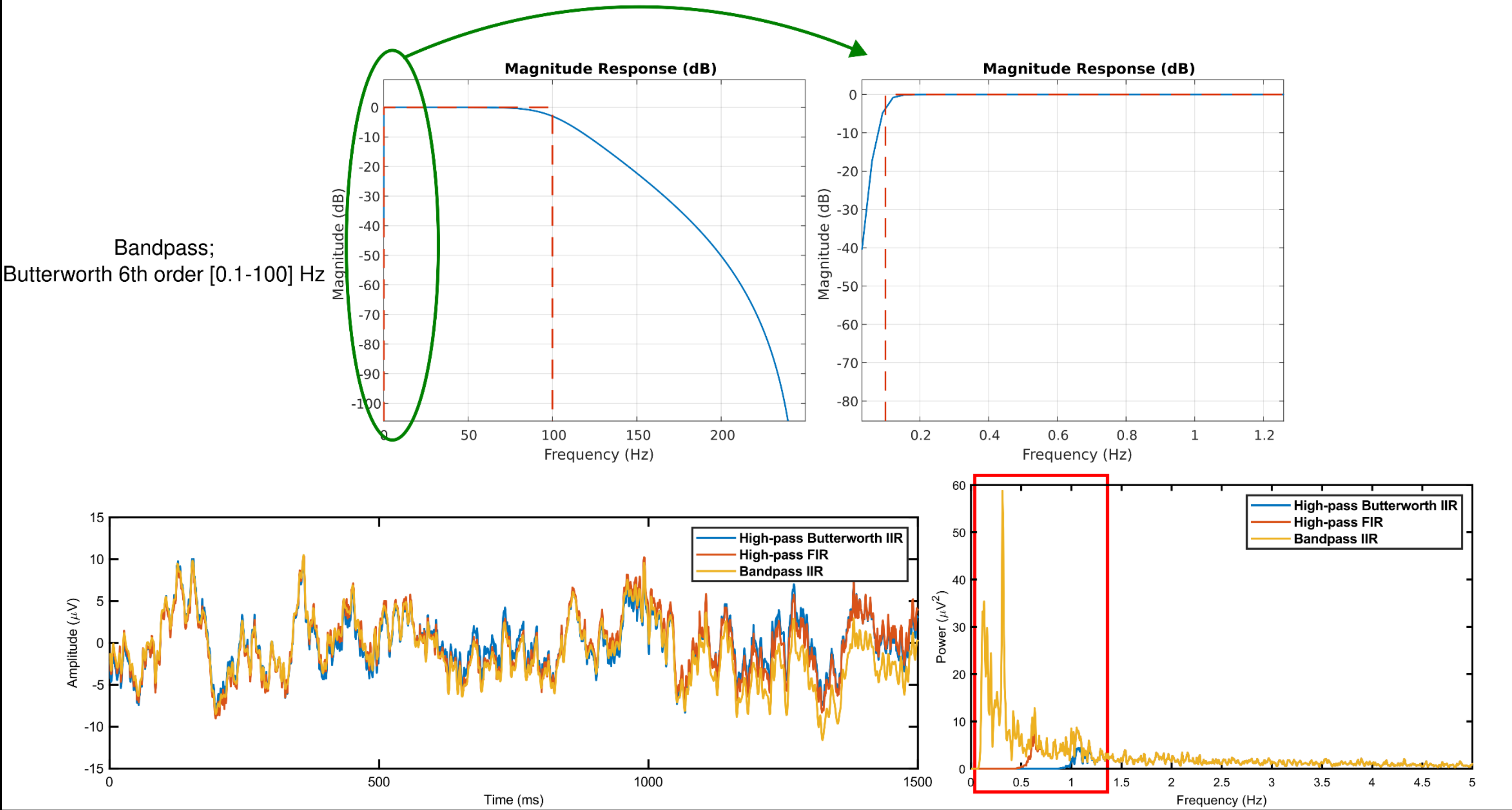
Filters



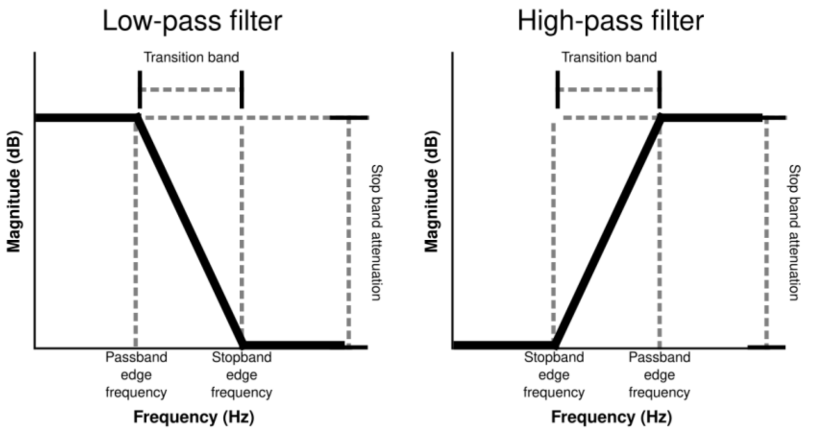
FIR: Finite Impulse Response
IIR: Infinite Impulse Response

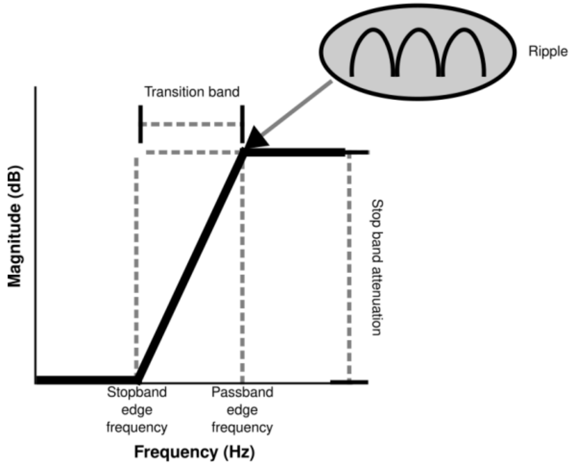
Filters

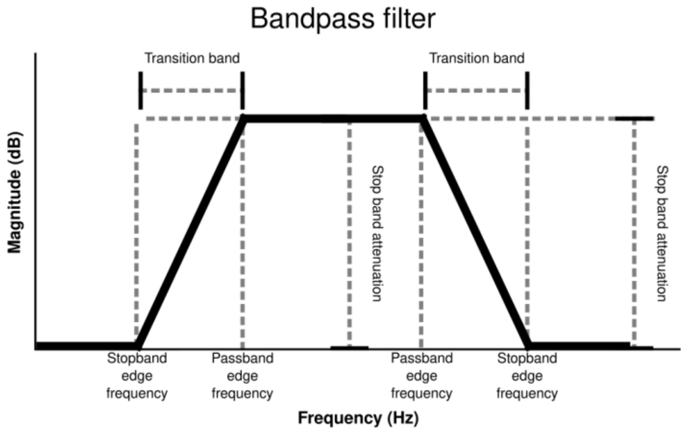
FIR: Finite Impulse Response
IIR: Infinite Impulse Response



Filters



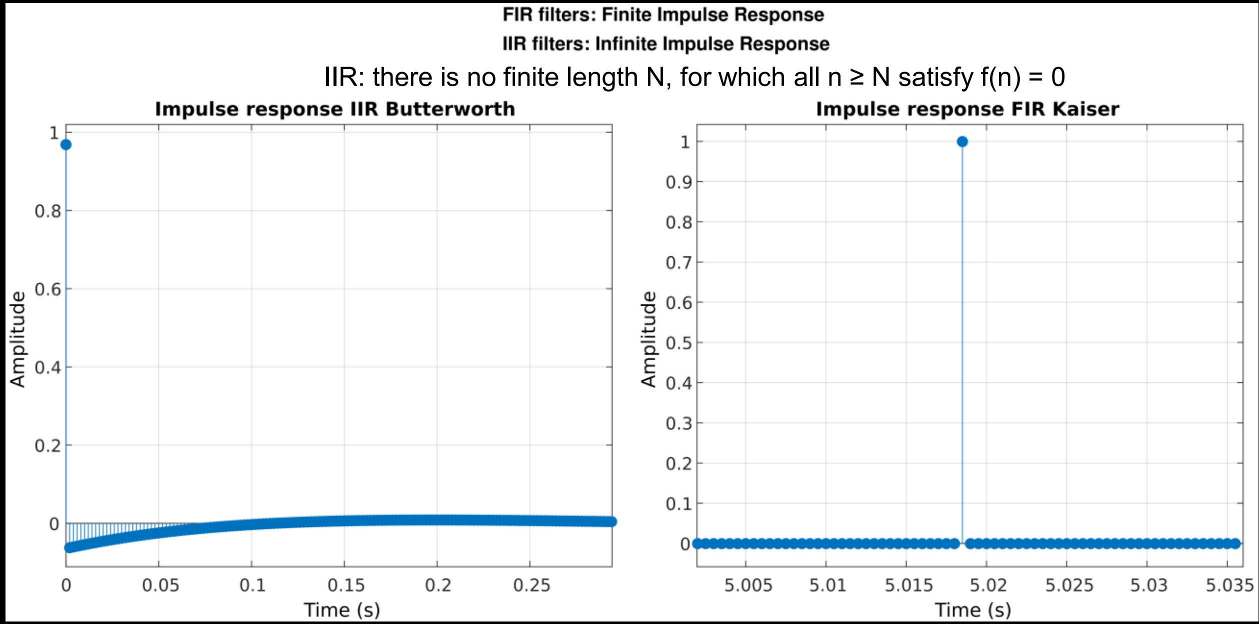




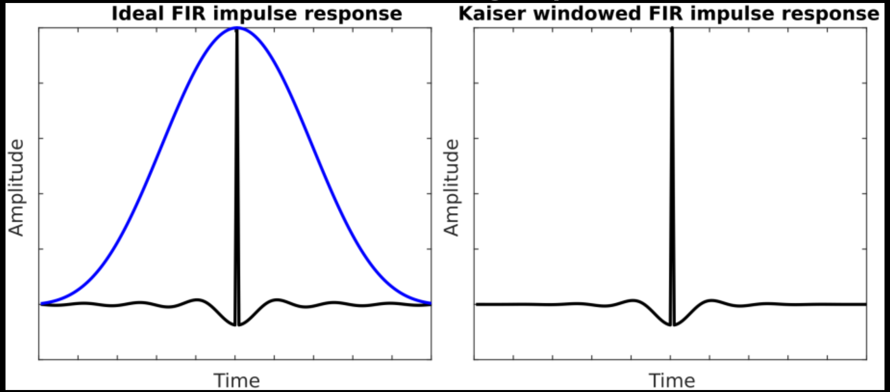
IIR Filter	Ripple in passband	Ripple in stopband
Butterworth	No	No
Chebyshev type I	Yes	No
Chebyshev type II	No	Yes
Elliptic	Yes	Yes

Filters

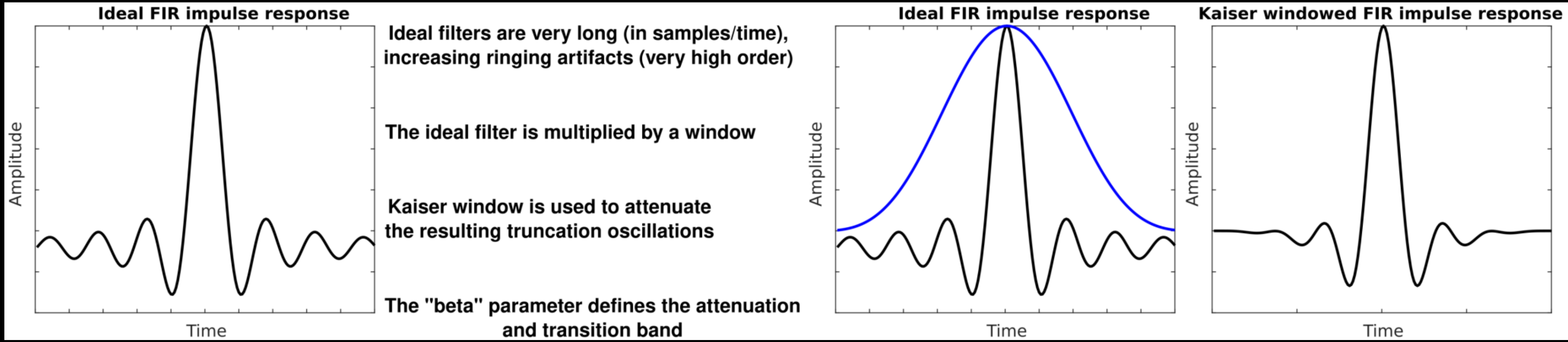
FIR: for low frequencies, need for very high-order filters (longer/nb cycles of freq.)
Reached for much lower order by IIR filters



High-pass

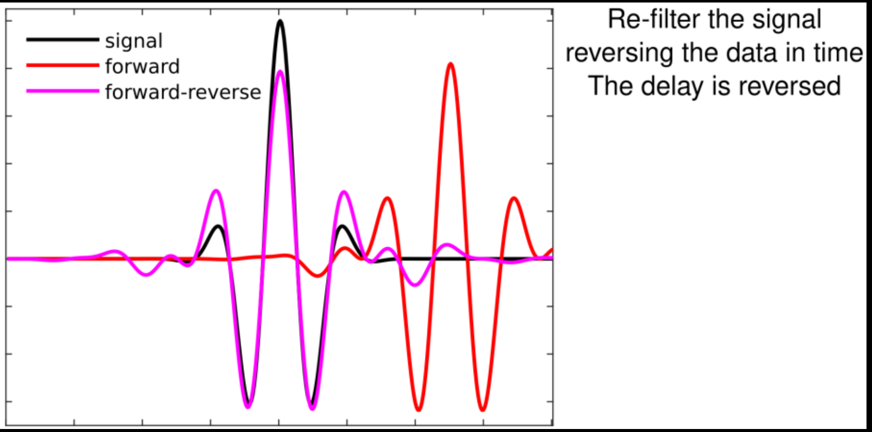


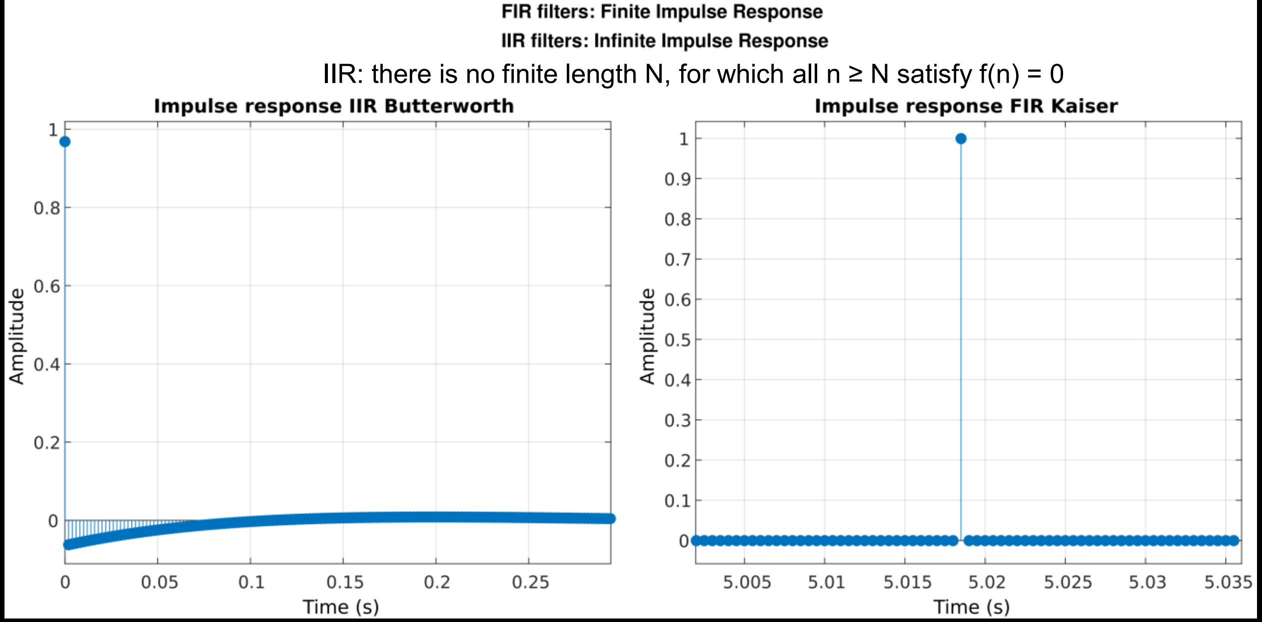
Low-Pass



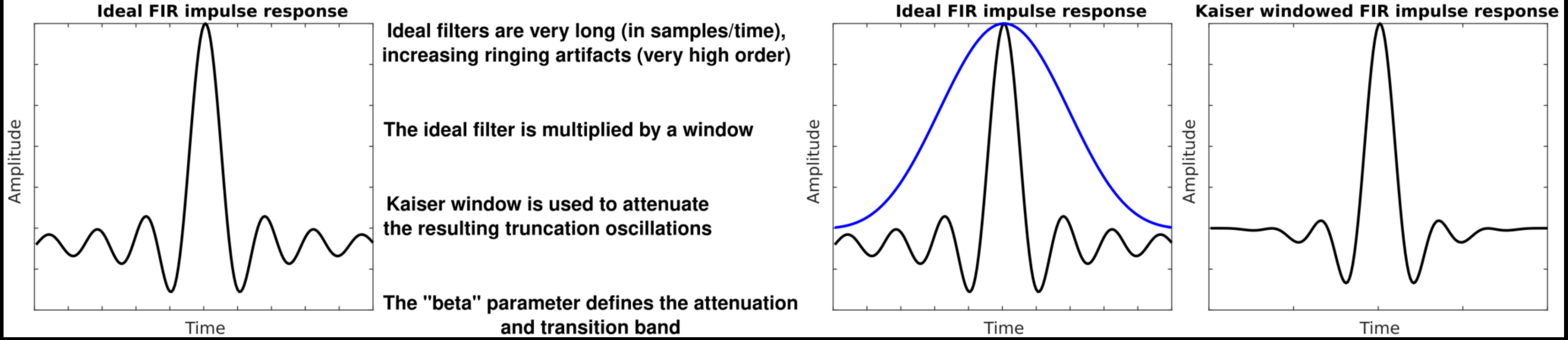
Filters

FIR: for low frequencies, need for very high-order filters (longer/nb cycles of freq.)
Reached for much lower order by IIR filters





Low-Pass



References

1. Greenacre, M. *et al.* Principal component analysis. *Nat Rev Methods Primers* **2**, 100 (2022).

2. Yasoda, K., Ponmagal, R. S., Bhuvaneshwari, K. S. & Venkatachalam, K. Automatic detection and classification of EEG artifacts using fuzzy kernel SVM and wavelet ICA (WICA). *Soft Comput* **24**, 16011–16019 (2020).

3. Pion-Tonachini, L., Kreutz-Delgado, K. & Makeig, S. ICLabel: An automated electroencephalographic independent component classifier, dataset, and website. *NeuroImage* **198**, 181–197 (2019).

4. Winkler, I. *et al.* Robust artifactual independent component classification for BCI practitioners. *J. Neural Eng.* **11**, 035013 (2014).

5. Treder, M. S., Porbadnigk, A. K., Shahbazi Avarvand, F., Müller, K.-R. & Blankertz, B. The LDA beamformer: Optimal estimation of ERP source time series using linear discriminant analysis. *NeuroImage* **129**, 279–291 (2016).

6. Yao, D. *et al.* Which Reference Should We Use for EEG and ERP practice? *Brain Topogr* **32**, 530–549 (2019).

7. Dong, L. *et al.* MATLAB Toolboxes for Reference Electrode Standardization Technique (REST) of Scalp EEG. *Front. Neurosci.* **11**, 601 (2017).

8. Chang, C.-Y., Hsu, S.-H., Pion-Tonachini, L. & Jung, T.-P. Evaluation of Artifact Subspace Reconstruction for Automatic Artifact Components Removal in Multi-Channel EEG Recordings. *IEEE Trans. Biomed. Eng.* **67**, 1114–1121 (2020).

9. *Neuroergonomics: Principles and Practice*. (Springer International Publishing, Cham, 2020). doi:10.1007/978-3-030-34784-0.

10. Perrin, F., Pernier, J., Bertrand, O. & Echallier, J. F. Spherical splines for scalp potential and current density mapping. *Electroencephalography and Clinical Neurophysiology* **72**, 184–187 (1989).

11. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L. & Lopez, A. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing* **408**, 189–215 (2020).

12. Christensen, M. G. *Introduction to Audio Processing*. (Springer International Publishing, Cham, 2019). doi:10.1007/978-3-030-11781-8.

13. Moulines, E. & Charpentier, F. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication* **9**, 453–467 (1990).

14. Johnson, K. *Acoustic and Auditory Phonetics*. (Blackwell Pub, Malden, Mass, 2003).

15. James, G., Witten, D., Hastie, T. & Tibshirani, R. *An Introduction to Statistical Learning : With Applications in R*. (2021).

16. Sassenhagen, J. & Draschkow, D. Cluster-based permutation tests of MEG/EEG data do not establish significance of effect latency or location. *Psychophysiology* **56**, e13335 (2019).

17. Dienes, Z. Using Bayes to get the most out of non-significant results. *Front. Psychol.* **5**, (2014).

18. Cohen, M. X. *Analyzing Neural Time Series Data: Theory and Practice*. (The MIT Press, Cambridge, Massachusetts, 2014).

19. Orfanidis, S. J. *Introduction to Signal Processing*. (Prentice Hall, Upper Saddle River, N.J, 2010).