Filter Effects & Artefacts

This is an interactive notebook about filters and the consequences you need to be aware of if you use filters. Before we come to the interactive part (the plots and the options in the sidebar), we briefly discuss some general knowledge about filtering in EEG.

Why do we filter?

Short & Simple: Because of Noise. Many if not all EEG signals would be nearly to impossible to analyse without filtering. And yes: One man's noise is another man's signal.
For now just consider for example high-frequency from the power line as noise.

What are the problems?

Filtering can distort the original signal. The result are so called filter effects or artefacts.
Those can be e.g. ...

- a smoother or different amplitude of the signal
- a phase shift or time distortion of the signal

The use of filters when you are unaware of the resulting consequences therefore lay the foundation for wrong interpretations!

To further take a look into the possible artefacts we have to consider the different possible filter methods. Often the artefacts are strongly related to the used method. By knowing the resulting artefacts from different filter methods you can avoid certain kind of artefacts by choosing a corresponding filter method.

FIR vs. IIR

To understand the difference between FIR & IIR filters take a look at the written out abbreviations:

- FIR := Finite Impulse Response
- IIR := Infinite Impulse Response

We have two open case:

- What is an impulse response?
- When is it called finite / infinite?

Let's start with the first one! An impulse response is the response of a filter to a unit impulse (signal that is 1 at the onset and 0 everywhere else). The response in the fourier domain is called frequency response. Easy! Onto the second one...

A Finite IR Filter has an finite impulse response which is after a finite time t zero. FIR Filters are also called linear phase filters. This means that all frequencies are shifted by the same value and no phase distortion takes place.

A Infinite IR Filter has no finite impulse response. So theoretically the impulse response based on the parameter choice can get an infinite number of non-zero values.

Causal vs. Acausal

Causal

Filter uses only the past and present => can only result in effects and artefacts after the onset

Acausal

Filter uses the past, present & future => can result in effects and artefacts before the onset. In practice this is achieved by filtering twice. Once with the original signal, once with the signal reversed (corresponds to backwards filtering).

Filter Methods

In this notebook four filter methods are used. Here are some characteristics listed...

1. FIR causal
   - linear phase
   - causal => no effect before onset
2. FIR acausal
   - linear phase
   - acausal => effects before onset
3. Butterworth of order 4 (IIR)
   - causal
   - phase distortion possible
   - flat pass band but broad transition band
4. Chebyshev of order 4 with 1 ripple (IIR)
   - causal
   - phase distortion possible
   - steeper transition band, but ripples in the pass band

Filter Type

The filter types are named straight forward. Once you've heard them you understand them.

Lowpass

A **lowpass filter** let's the low frequencies pass. It is used to zero out frequencies above a certain threshold / cutoff.

Highpass

A **highpass filter** is the inverse of the lowpass filter. It let's the frequencies above a certain threshold pass.

Bandpass

A **bandpass filter** is a combination of a lowpass and a highpass filter. It let's frequencies in a by parameter defined range pass. This range is called the passband. This is achieved by applying the low and highpass filter sequentially.

Bandstop

Also often called Notch. A bandpass filter is also a combination of a lowpass and a highpass filter. You can imagine it as the inverted bandpass filter. The passband of the bandpass filter is now a stopband and blocks the corresponding frequencies. To get the bandstop filter a lowpass and bandpass is applied separately to the original signal, afterwards the signal is combined.

More Background Information

If you want to get a more detailed background in filters, this MNE_Tutorial as well as this Paper is a great resource!

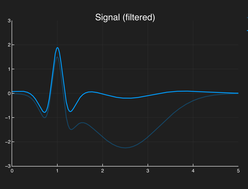
Example Configurations

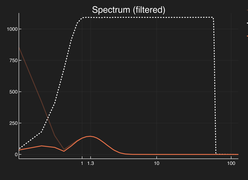
Here are some example configuration which produce some interesting structures...
Hint: Start by comparing the impulse response of the unit impulse to get an better understanding. of the filters.

5 rows × 7 columns

	signal String	freq String	ftype String	fmethod String	low_cutoff String	high_cutoff String	notes String
1	Unit Impulse	–	Lowpass	FIR causal	10 Hz	–	=> causal
2	Unit Impulse	–	Lowpass	FIR acausal	10 Hz	–	=> acausal
3	Unit Impulse	–	Lowpass	Butterworth	10 Hz	–	=> causal
4	Unit Impulse	–	Lowpass	Chebychev1	10 Hz	–	=> causal
5	ERP	–	Highpass	FIR	–	1 Hz	–

Interactive Plots





For saving to png with the Plotly backend PlotlyBase has to be installed.

Interactive Sliders

Here are all interactive bits of the notebook at one place.
Feel free to change them!

Signal

ERP
Unit Impulse
Boxplot
Sinusoidal

Noise 0.0

Line noise (50 Hz)
Shift at t=2

Filter Type & Method

Lowpass
Highpass
Bandpass
Bandstop

FIR causal
FIR acausal
Butterworth
Chebychevi

Filter Parameters

Change the low cutoff 0.5
Change the high cutoff 60.0

Cluster Permutation Test

No matter in which field you work, if you are in research, you can't avoid statistics.
Even if you would like.

This interactive tutorial has the goal to give you an insight into cluster permutation tests. Afterwards should be clear why cluster permutation is used and how it works! This journey is separated into five steps.

Why do we need cluster permutation tests? To answer this question we need to know more about the setup / scenario we are in.

Assume we have recorded EEG data over multiple electrodes and we want to analyse it. This leads to the unavoidable problem of performing statistical tests on all of these electrode and timepoint combinations.

Isn't possible to use a multiple t-test? No! Already with a small number of timepoints and electrodes the probability of a false positive would be close to 100%. This is accumulation of type I errors is called the multiple comparison problem.

The Multiple Comparison Problem

The more hypotheses one tests on a data set, the higher the probability is that one of them will be incorrectly accepted as true.

Therfore other possibilities need to be look at! Cluster Permutation Tests for example :)

Interactive Sliders

Here are all interactive bits of the notebook at one place.
Feel free to change them!

Data, pinknoise & observed clustermass

Change effect size 0.2

Change the level of pink noise 0.04

Permutation of data

Change the permutation step 1

Step 0: Data, pinknoise & clustermass

In praxis the procedure would be the following (simplyfied):

1. Collect EEG data (time x sensor) from multiple subjects (and for each condition).
2. Extract ERPs per subject and condition.

But since we are here on theoretical terrain, things are a bit tidier :)
Instead of going through hours of hours of EEG data collection, we simulate the EEG data or more specific the ERPs for the conditions.

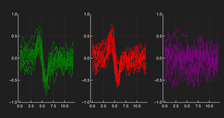
In our case we have ERPs of two conditions (A and B). Feel free to change the effect size and noise with the sliders below!

Change effect size 0.2

Change the level of pink noise 0.04

Step 1: Calculate the difference

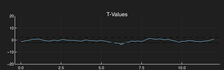
We calculate the difference between the two conditions A and B (a within subject comparison). Thus, we get difference values for each subject over time (purple).



Step 2: Clusters over time

As briefly mentioned in the intro, we do not want to do a statistical test for each time-point individually, because we would need to correct for multiple comparison for all timepoints.

Instead we use a trick:
Let's define clusters by an arbitrary threshold and test whether these clusters are larger clusters that occur by chance. The arbitrary threshold we use is at p=0.05 (which directly corresponds to two t-values when given the number of subjects, e.g. for 15 subjects the t-values corresponding to p=0.05 are 2.14 and -2.14).



$$\text{Observed Clustermass} = 19.724354208386163$$

The light grey filled patch is a cluster (if no cluster is visible, increase the effect size.) The observed clustermass is the sum of the t-values over the interval of the grey filled patch.

Note: It is possible that not only one but multiple cluster of different sizes could have formed. In this case we choose the biggest cluster to calculate the observed clustersize!

Note: As a statistic we could use the *number* of samples the cluster extends, the summed t-value, or many other statistics. We use cluster-mass, which is as described above the sum of the t-values.

Side note: t-values and variance

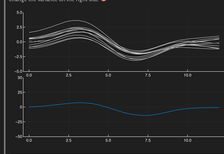
Not familiar with t-values? Show more ...

First things first: t-values aren't the only possible choice. As a test statistic we have multiple options to choose from. One for example would be the mean. We prefer the t-statistic because of its characteristic that it punishes high variance between subjects. Try it out by changing the slider below!

Definition of t-value:

$$t = \frac{\bar{x}}{\frac{s}{\sqrt{n}}}$$

Change the variance on the right side 0



Take Away !

The **greater** the variance, the **smaller** the t-value!

Step 3: Permutation of data

We now want to estimate how big clusters would be, if there would be no differences between the conditions A and B. This would mean that the clusters formed just by chance (this is our H0 distribution of cluster sizes). To do this, we shuffle the condition label for each subject.

The idea is that if there is no difference between the conditions, the labels are meaningless and therefore shuffeling them would yield similar results as before.

Note that they are similar, not identical. We thus try to estimate how big the variability of these similar results are, and whether our observed value falls into the variability, or whether i is special.

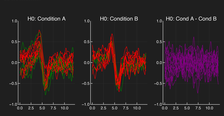
The next three plots visualize this procedure.

- the first three subplots show the permutated trials of H_0 for both conditions and the difference
- the second graph shows the t-values
- the last plot is the histogram of clustermasses

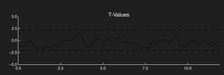
By interacting with the corresponding buttons below you can see how the histogram of the clustermasses is created. Try it out!

Change the permutation step 1

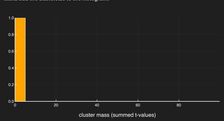
So we shuffle...



...and calculate the clustersize...



...and add the clusterize to the histogram.

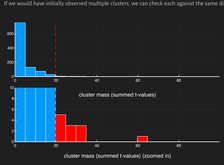


Tip for the computation: Note that we actually do not need to go back to the two conditions between each permutation, but we could just flip (multiply by -1) randomly every subject-difference curve (purple colored lines).

Step 4: Check the tail !

We now check whether our observed cluster mass (Step 2) is greater than 95% of what we would expect by chance (Step 3). The exact value gives us the p-value of the cluster, the probability that cluster-mass with the observed (or more extreme) size would have occured when there was no actually difference between the conditions.

If we would have initially observed multiple clusters, we can check each against the same distribution.



$$p = 0.012$$

Change the effect size and noise and see how it affects our decision on the H_0.

Reject H0 !

Because **p < 0.05** we reject H_0.
This means that the observed clustermass is **unlikely** to come from random chance alone.

Some more notes...

Important!

The statement you can make (if p<0.05):
There is a significant difference between condition A and condition B (optional: in clustersize)

The statistical statement is about the cluster, not about the individual vowels comprising the cluster.

The statements you can not make:
- **We observed a significant difference between robins and pigeons**
- **The first significant cluster (p=0.04) was found at occipital electrodes**

Deconvolution (Overlap-Correction)

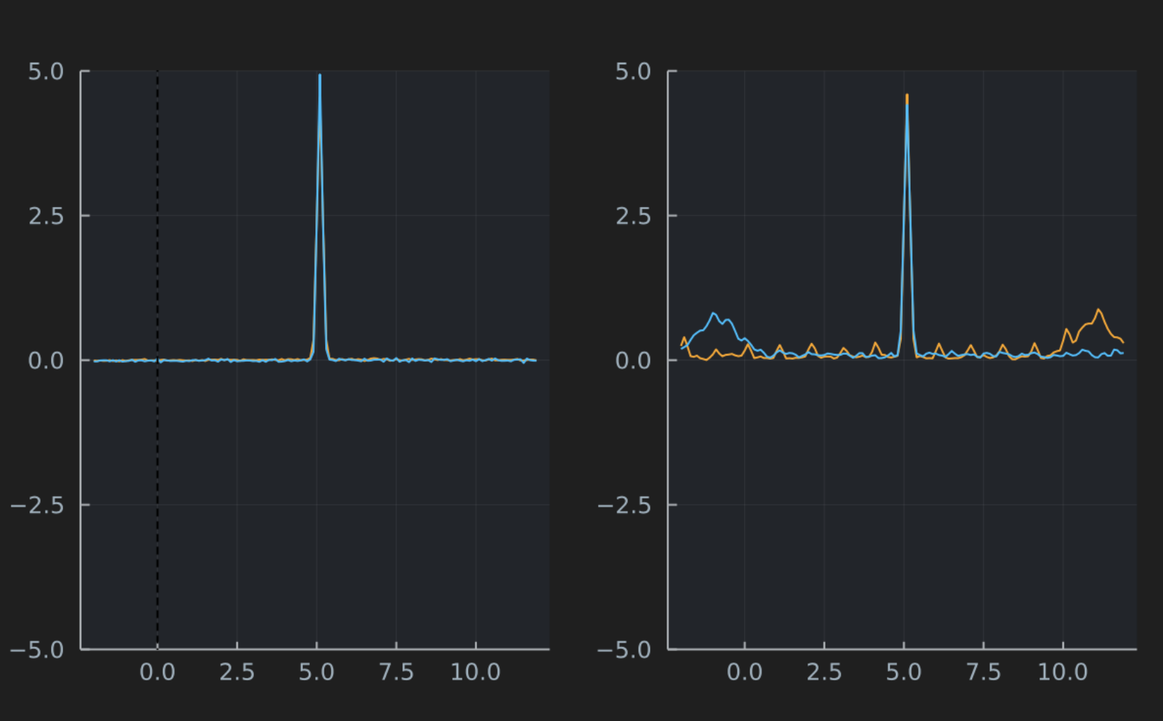
Let's get right into it - you are probably asking yourself, what is this about? Do i need to know this? Here is a small motivation...

In state-of-the-art EEG research the experiment setup is highly controlled and simplified in a way to avoid overlaps of stimuli responses. This means often only a single stimulation per trial is presented. But the more complex and realistic the research topic, the more complex gets the experiment design. Up to a point were it isn't anymore possible to avoid overlaps. Examples for this are experiments were eye tracking and EEG is combined or the tracking of EEG data in a free environment. However also classic EEG research experiments often contain overlapping responses as simple as a manual button presses or involuntary microsaccades.

In this case adequately modeling of those overlaps, different to simple averaging is required! If not further considered, this can lead to wrong interpretations and false reasoning.

This notebook should give an intuition **why** the approach of deconvolution should further be investigated and **what** the consequences of not accounting for overlaps are!

Deconvolution (Left) vs. Without-Deconvolution (Right)



```
Resolving package versions...
No Changes to `~/.julia/environments/v1.7/Project.toml`
No Changes to `~/.julia/environments/v1.7/Manifest.toml`
```

```
For saving to png with the Plotly backend PlotlyBase has to be installed.
```

Interactive Sliders

Here are all interactive bits of the notebook at one place.
Feel free to change them!

Stimuli Response
Choose the response function for stimulus A: Function1

$$ERP_A(t) = -5(t-5)e^{-0.5(t-5)^2}$$

Choose the response function for stimulus B: Function2

$$ERP_B(t) = 2.5e^{-(t-5)^2}$$

Event onsets
Change deviation: σ_1 = 0.1
Change mean:μ_1 = 0.0

Linear Deconvolution
Change noise: σ = 0.0

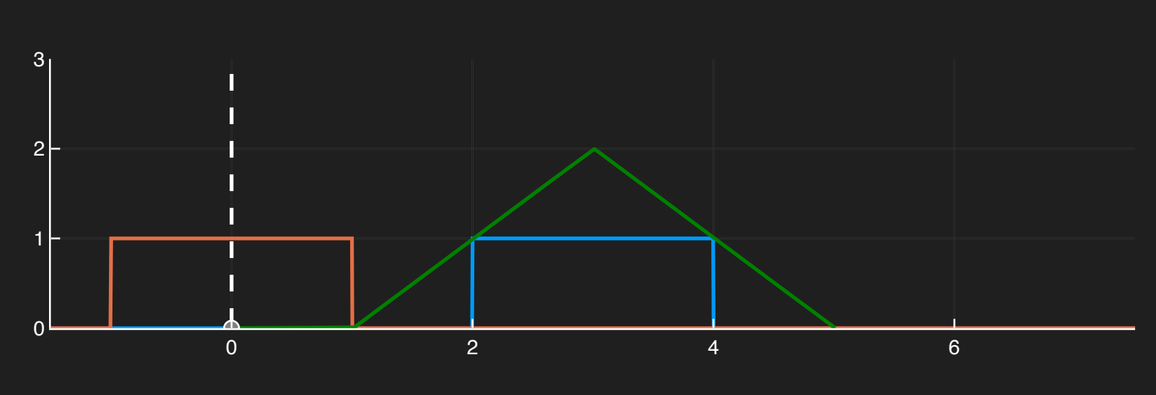
Intuition to Convolution

Before we start digging deeper into the features and characteristic of deconvolution we should first briefly take a closer look at the topic of convolution.

In math convolution is a operation which takes two functions as input and outputs a third function. Formal, deconvolution of the functions f and g is written as $(f*g)$. But what does this new output function describe?

For this a closer look to the process of convolution helps:
Convolving two functions is often described as sliding one function over another. Try sliding the orange function over the blue by changing the value of the slider below.

Change the position of the orange function -1.0



Consider the following setup:
We want to simulate the measured EEG signal of an experiment. In this experiment we have two different stimuli. Each stimuli evokes a different response. Assume we already know this specific response to each stimuli. Additonal we know from the experiment setup at which timepoint each stimulus occurred.

Response to Stimuli / Kernel

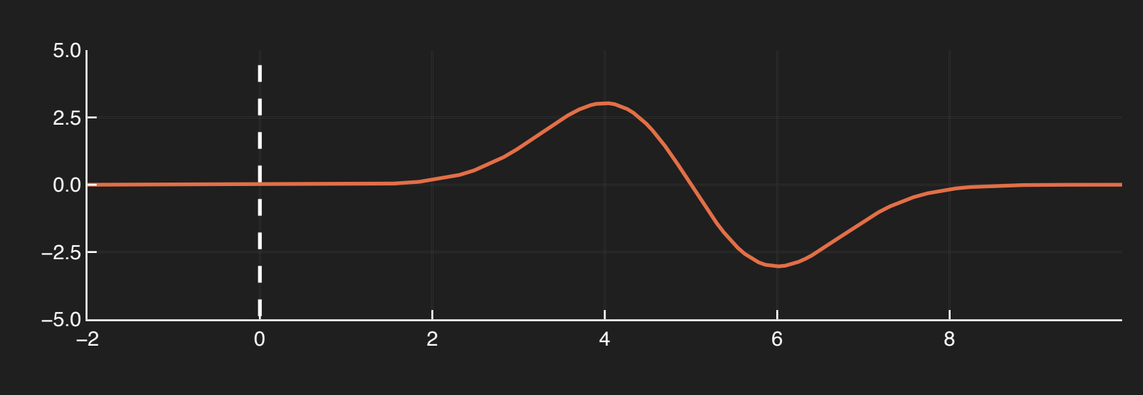
As described above, we need for our simulation of the EEG signal, the isolated response of each stimuli. In the context of convolution this is often called kernel. The following figures show the kernel A (orange) and B (green).

The **response to stimuli A** is modelled by the function $ERP_A(t)$

Choose the response function for stimulus A: Function1

Change the value of b 5

$$ERP_A(t) = -5(t-5)e^{-0.5(t-5)^2}$$

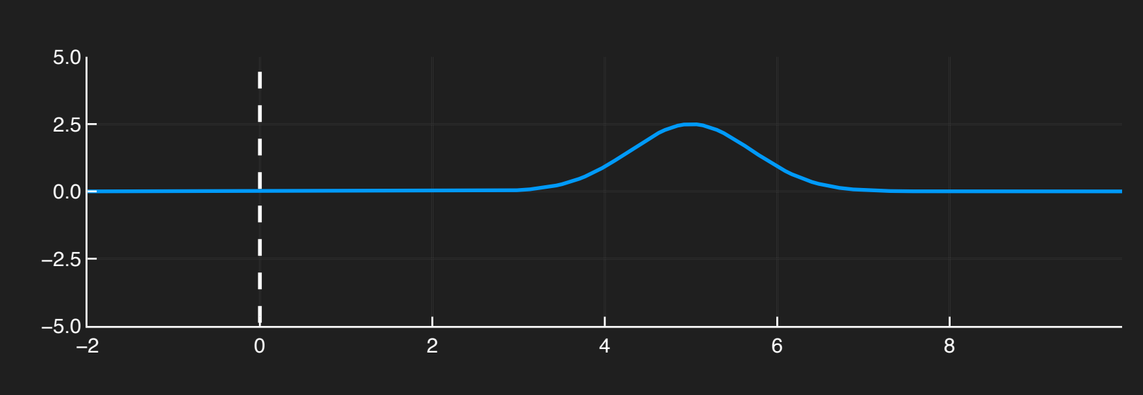


Analogous to this the **response to stimuli B** is modelled by $ERP_B(t)$.

Choose the response function for stimulus B: Function2

Change the value of d 5

$$ERP_B(t) = 2.5e^{-(t-5)^2}$$



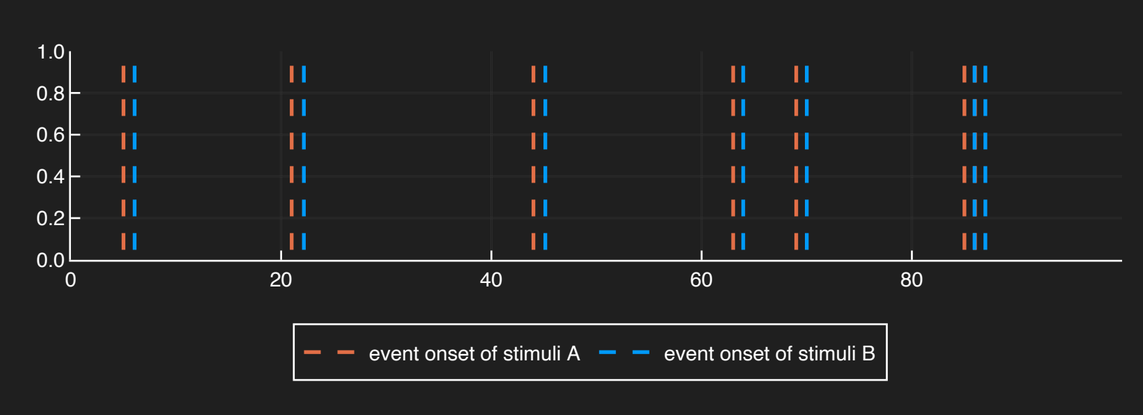
Event onsets

The next figure shows the **event onsets**. They are part of the experiment design. Normally in research those event onsets are distributed in such a way that overlapping responses are avoided at all costs. But this is not always possible given the experiment or research. To later dicuss the problems of not using overlap-correction, we will not avoid overlapping responses in our simulated EEG data. More: We will enforce them to happen at some level.

Since we want to create a simulated EEG signal we simply choose 300 random values between 1 and 6000 for each stimuli. The event onsets are visualized in the figure below. The orange vertical line corresponds to the event onsets of stimuli A. The green line to the event onsets of stimuli B.

Change deviation: σ_1 = 0.1

Change mean:μ_1 = 0.0



Convolution of kernels with event onsets

In our process to simulate the continuous EEG signal, we rely on multiple assumptions. One main assumption is that signals within the brain add up linear. Based on this, we can describe the continuous EEG Signal at each timepoint t as following:

$$EEG(t) = \sum_{i=1}^{n_A} ERP_A(t - eventOnsetA_i) + \sum_{i=1}^{n_B} ERP_B(t - eventOnsetB_i)$$

Is this a convolution?

Yes indeed. By replacing the event onsets with a vector g with zeros everywhere and 1 at the event onsets, we can reformulate the equation from above:

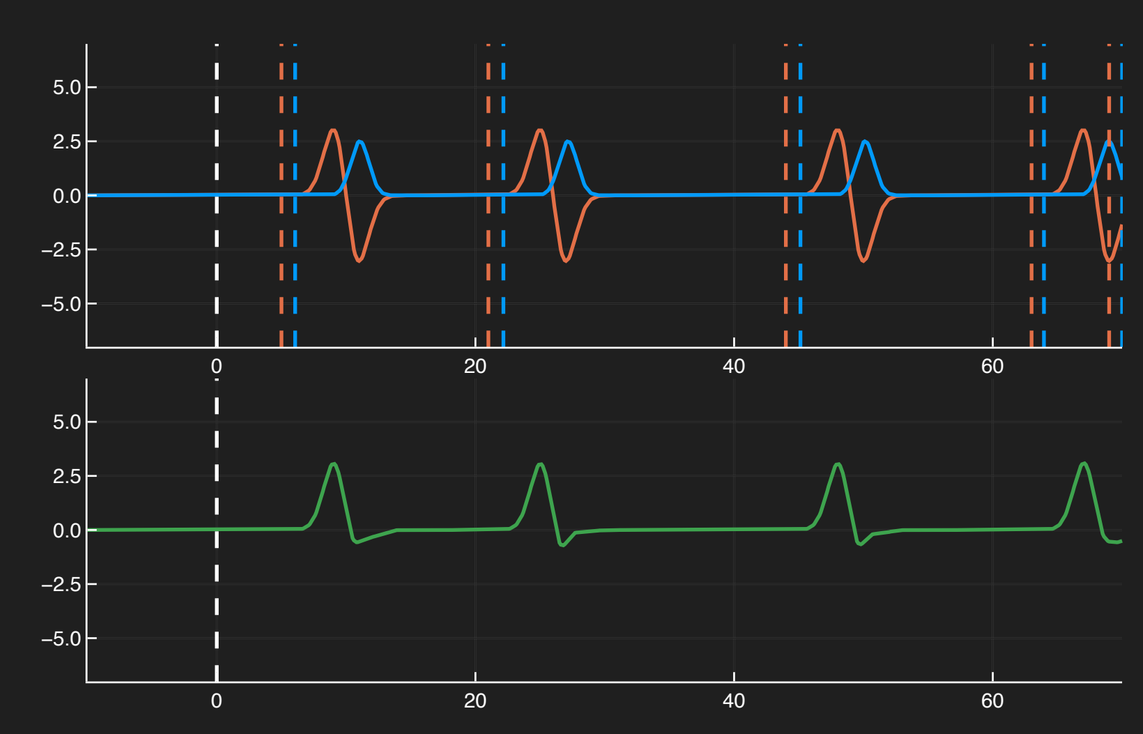
$$EEG(t) = g_A * ERP_A + g_B * ERP_B$$

Take Away!
This is a sum of two convolutions!

Simulated EEG signal

How does this sum of convolutions look like? Take a look at the next figure!

The first graph shows the signals of the convolution of the event onsets with the respective kernel of the stimulus. This results in a signal for each stimulus. The orange signal belongs to stimuli A, the blue to stimuli B. The vertical lines show the event onsets in the respective color.
The green graph below shows the overall signal. This results from adding up the orange and blue signal at each timepoint.



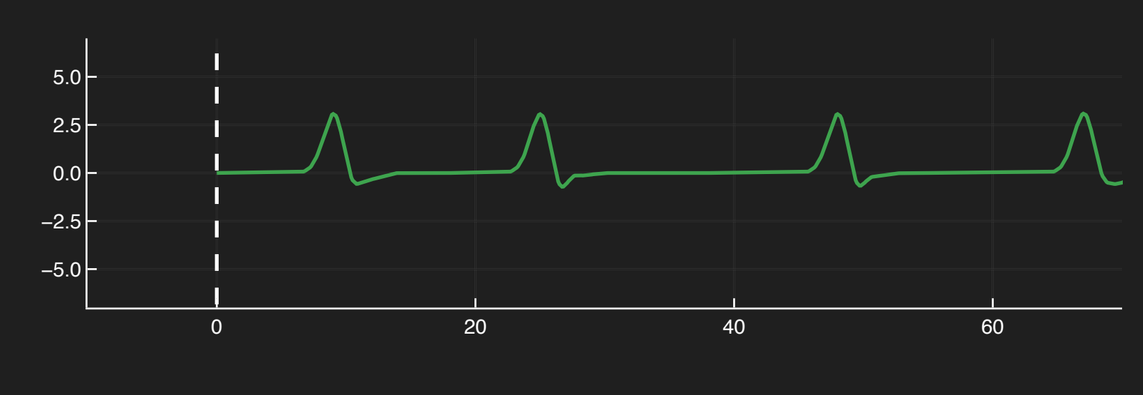
Linear Deconvolution with Unfold

From here on, lets assume we **measured** the **green signal** from the above in our **eeg experiment**. From our experiment we additional know at which time each respective stimulus was presented (event onsets). Based on this we try to recover the underlying ERP for each stimulus. In our case stimuli A and stimuli B. This is the inverse operation of the above performed convolution.

For illustration purpose we introduce noise to the data. The level of noise is contolled by the variable σ. The greater σ, the more noise is added to the original data.

Feel free to adjust the noise, and see how the quality of the results change.

Change noise: σ = 0.0



$$EEG(t) = g_A * ERP_A + g_B * ERP_B$$

By taking a closer look at the formula this means the following:

- We know the measured EEG Signal at each timepoint ($EEG(t)$)
- We know the event onsets for each stimuli (g_A, g_B)
- We want to recover the isolated response function to each stimuli ERP_A and ERP_B

To achieve this, **Linear Modeling** comes to our rescue! Why? We can use LMs because of two reasons / assumptions:

1. The **overlap** for every event onset is always **slightly different**. This makes it possible to disentangle the two separate responses.
2. We assume that two in time following events don't influence each other within the brain. More specific: The first event **does not influence** the **processing** of the second event.

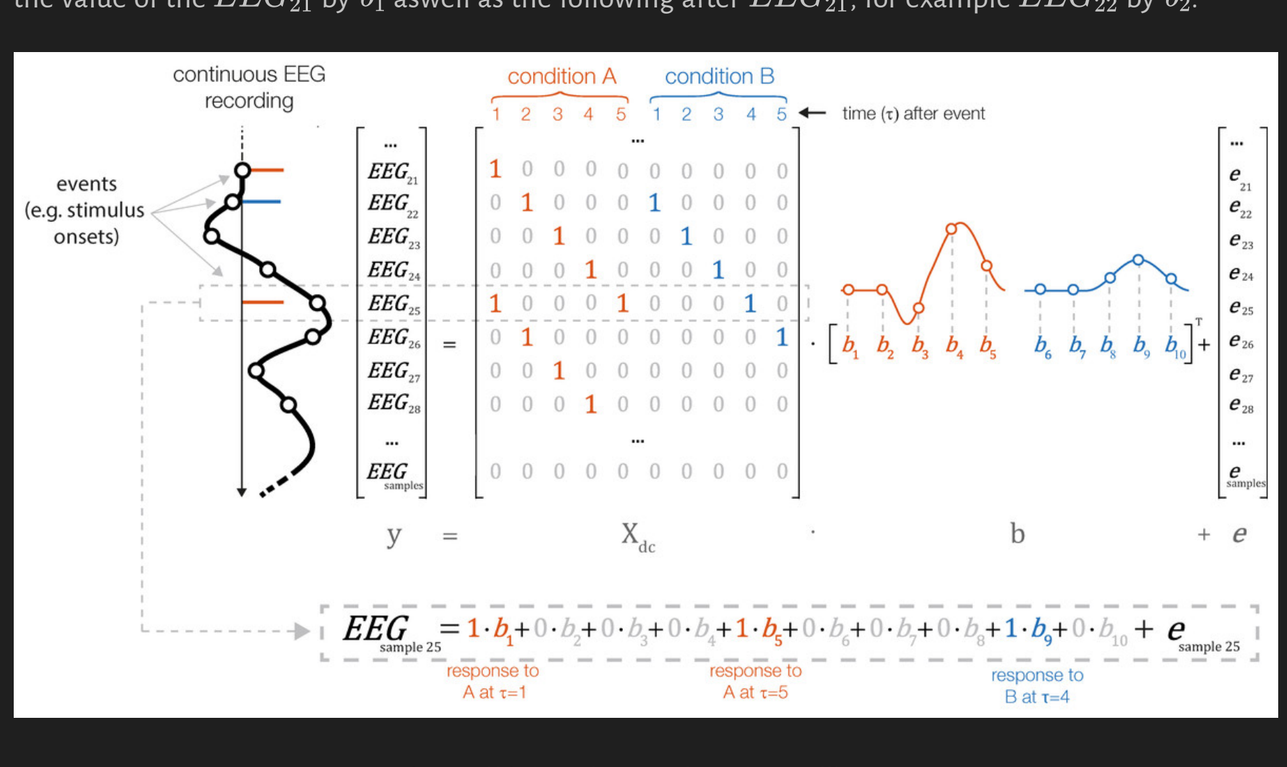
Key Idea
Each **timepoint / observed sample** of the EEG signal can be modelled as a **linear combination** of the (possibly) **overlapping responses / kernels**.

This key idea is visualized in the following figure.

On the left side the figure shows the continuous EEG recording together with the event onsets. It is splitted into distinctive samples for each timepoint and forms the vector y.

X_{dc} is the timeexpanded designmatrix. The timeexpansion is indicated by the diagonal columns with the value one over time. Time in this case means timesteps after the specific event onset (τ).
To get the in the figure stated equation the designmatrix is then multplied by the vector b. This gives us an equation which we can optimize for the best fitting betas.

Intuitively this is what we would expect. The response to the event onset at timepoint t=21 influences the value of the EEG_{21} by b_1 aswell as the following after EEG_{21}, for example EEG_{22} by b_2.



https://doi.org/10.7717/peerj.7838/fig-2

Take a closer look at the example for EEG_{25}.

The equation for EEG_{25} includes 3 weights:

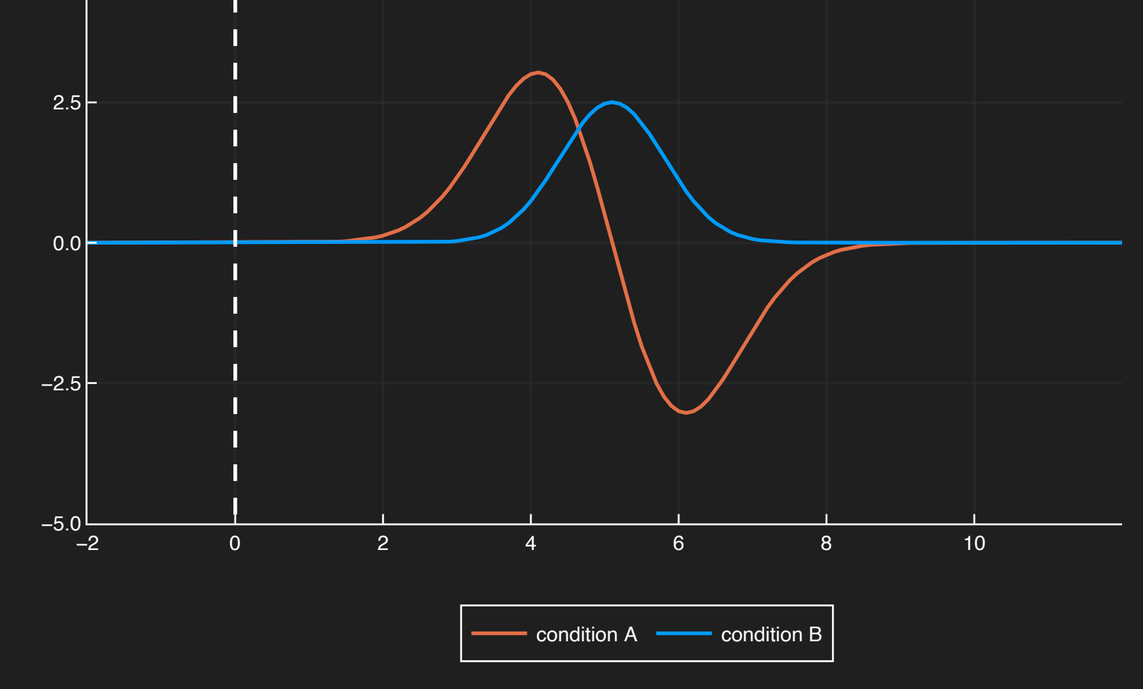
- β_1 : The response to stimuli A at the local time τ=1
- β_5 : The response to stimuli A, 5 seconds after the event onset (τ=5)
- β_9 : The response to stimuli B at the local time τ=4

This tells us that each of those weights accounts to some extend to the value of EEG_{25}. Those multiple equations (each for one timepoint) with a slightly different linear combinations of weights allow us to find the best fitting β's to model our data.

As additional variable we introduce the window size. Feel free to change the upper bound and inspect the results graph.

Change window size τ = (-2.0, 12)

Plotting the results



Take Away !
Deconvolution recovers the respective kernels in this case regardless of overlaps!

Question ?
Does the same apply for the widly used mass-univariate approach?

Deconvolution vs. Mass-Univariate

This plot shows the comparison between the **deconvolution with overlap correction** (left) and the **mass-univariate approach** (right).

As a reminder: Feel free to play around with the interaactive parts (at the right side) and compare the results for differnt functions and values.