Center for Mind and Brain
University of California, Davis

The ERP Boot Camp

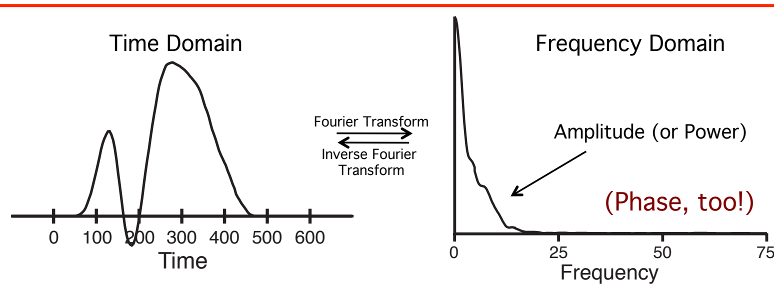
The Title of this Lecture is Secret

(Time-Domain Signals, Filtering, and Linear Systems Analysis)

Filtering Overview

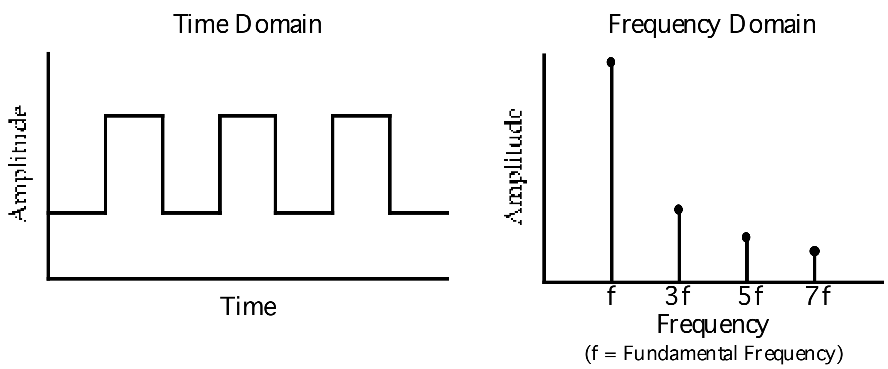
- Filter- To remove some components of an input and pass others
  - We will concentrate in Finite Impulse Response (FIR) filters
- Different approaches to filtering
  - Hardware filters
  - Filtering by conversion to frequency domain
  - Filtering by computing weighted average of adjacent points
  - Filtering by convolving with impulse response function
- These are all mathematically equivalent
  - Relatively simple relationships between them
- By understanding these relationships, you will have a much deeper understanding of the nature of ERPs
  - But this stuff can blow your mind...

Fourier Analysis

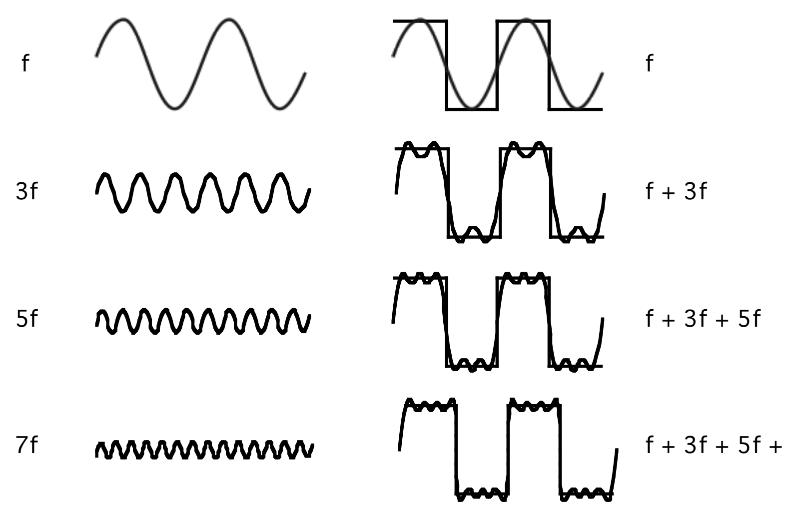


- Any waveform is equivalent to the sum of a set of sine waves of different frequencies, amplitudes and phases
  - Time domain: Amplitude as a function of time
  - Frequency domain: Amplitude and phase as a function of frequency
- Fourier transform converts time domain to frequency domain
- Inverse Fourier transform converts frequency domain to time domain

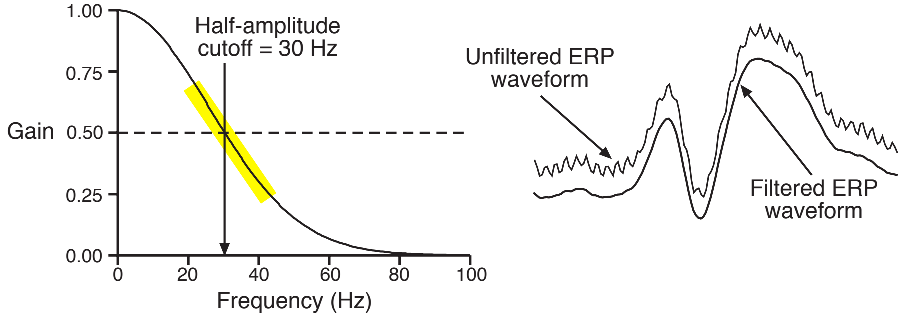
Fourier Analysis



Fourier Analysis



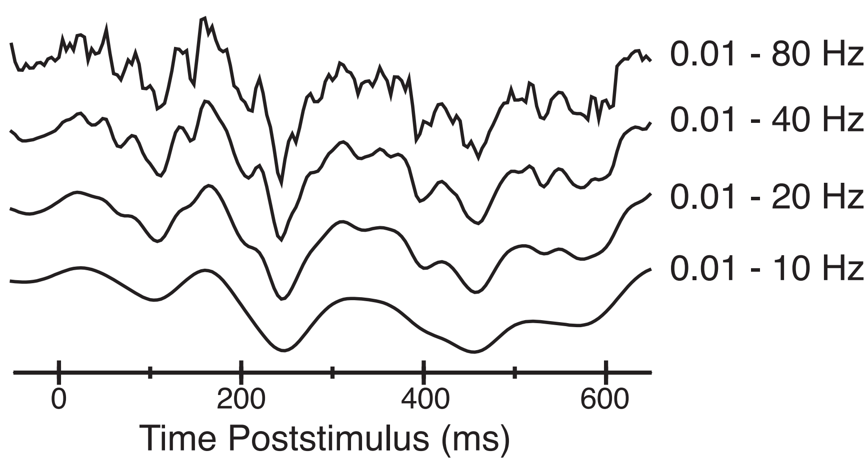
Example Low-Pass Filter



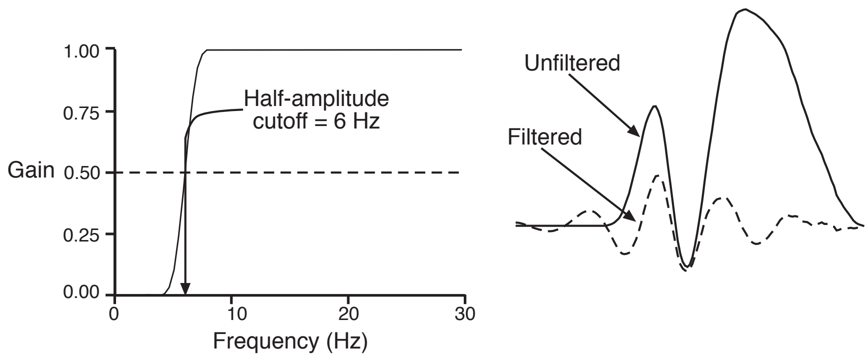
Low-pass filters pass low frequencies, attenuate high frequencies

Rolloff (slope): How steeply gain changes as frequency increases
Measured in dB/octave
6 dB = 50% drop in amplitude; 3 dB = 50% drop in power
1 octave = doubling of frequency

Variations in Cutoff Frequency

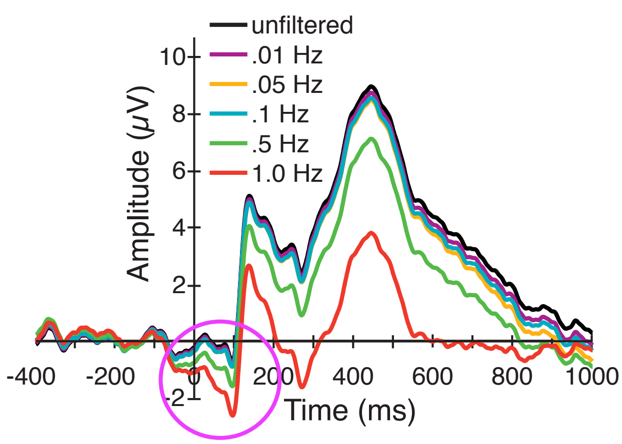


Example High-Pass Filter



High-pass filters pass high frequencies, attenuate low frequencies

Effect of Cutoff Frequency



High-pass filters cause significant amplitude reduction in slow components (and distortion of fast components) when the cutoff exceeds ~0.1 Hz

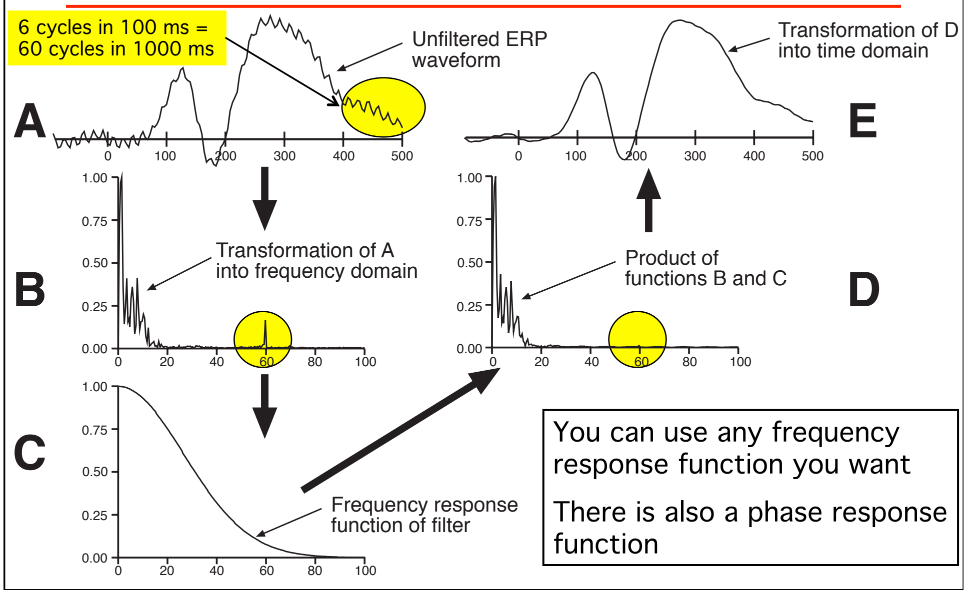
Why are filters necessary?

- Nyquist Theorem
  - Need to make sure we don't have frequencies >= 1/2 the sampling rate
- Noise reduction
  - Low-pass filters for muscle noise
  - High-pass filters for skin potentials
  - Notch filters for 50/60-Hz line noise
- But filters distort your data, so they should be used sparingly
- Hansen's axiom: There is no substitute for clean data
- Luck's complaint: ERPs are not actually the sum of a set of infinite-duration sine waves
  - So don't pretend they are

Fundamental Principle

- Precision (spread) in frequency domain is inversely related to precision (spread) in time domain
  - What is the time domain representation of an infinitesimally narrow spike in frequency domain?
  - What is the frequency domain representation of an instantaneous impulse in the time domain?
- The more you filter, the more temporal precision you lose
- The sharper your filter rolloffs, the more temporal precision you lose
- The loss of temporal precision can create artifacts that will lead to incorrect conclusions

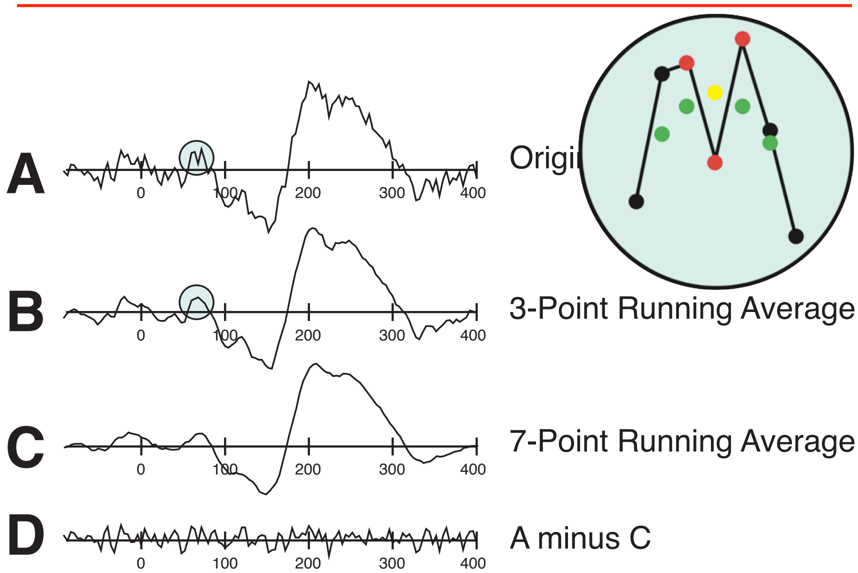
Filtering in Frequency Domain



You can use any frequency response function you want

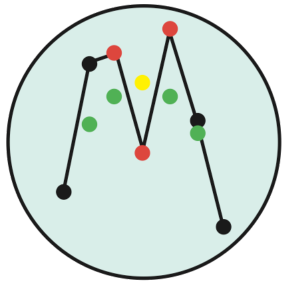
There is also a phase response function

Filtering in Time Domain



Running-Average Filter

(AKA Boxcar Filter)



$$fERP_i = \frac{1}{p}\sum_{j=-n}^{n} ERP_{i+j} = \sum_{j=-n}^{n} \frac{1}{p} ERP_{i+j}$$

where:

$fERP_i$ is the filtered ERP waveform at time i

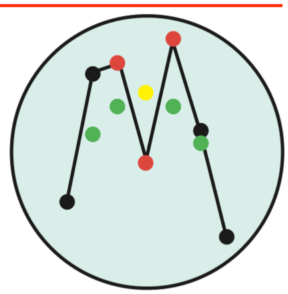
ERP_i is the unfiltered ERP waveform at time i

n is the number of points on each side of the current time point

p is the total number of points to be averaged together ($2n+1$)

Note: Increasing p produces both greater filtering and a reduction in temporal precision

General Time-Domain Filter



$$fERP_i = \sum_{j=-n}^{n} W_j ERP_{i+j}$$

Instead of taking the average of the p points, we use a weighted average, giving nearby points greater weight

W is the weighting function

Example: $W_{-1} = .25$, $W_0 = .50$, $W_{+1} = .25$

Gaussian function is common

Running-average filter uses equal weights (.33, .33, .33)

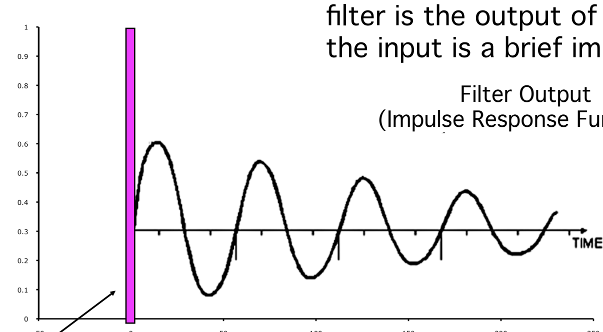
Impulse-Response Function

- Up to this point, we have been thinking about time-domain filtering from the point of view of calculating the filtered value at a given point in time
  - Filtered value at time t = weighted average of unfiltered values at surrounding time points
- We can also think of filtering from the point of view of how the unfiltered value at time t influences the whole set of filtered time points
  - Key: Filters are linear (for FIR filters)
  - If we see the filter's output for a single point, we can predict it's output for the whole waveform

Impulse-Response Function (IRF)

The impulse-response function of a filter is the output of the filter when the input is a brief impulse

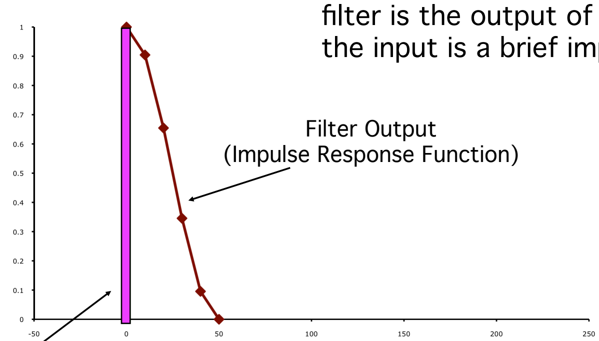
Filter Output
(Impulse Response Function)



Filter Input
(Impulse)

Impulse-Response Function (IRF)

The impulse-response function of a filter is the output of the filter when the input is a brief impulse

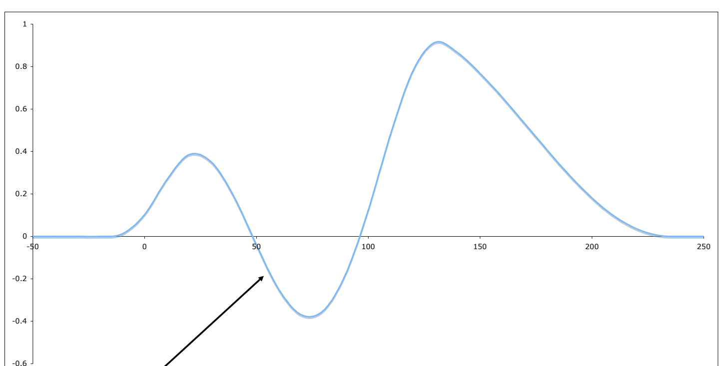


Filter Output
(Impulse Response Function)

Filter Input
(Impulse)

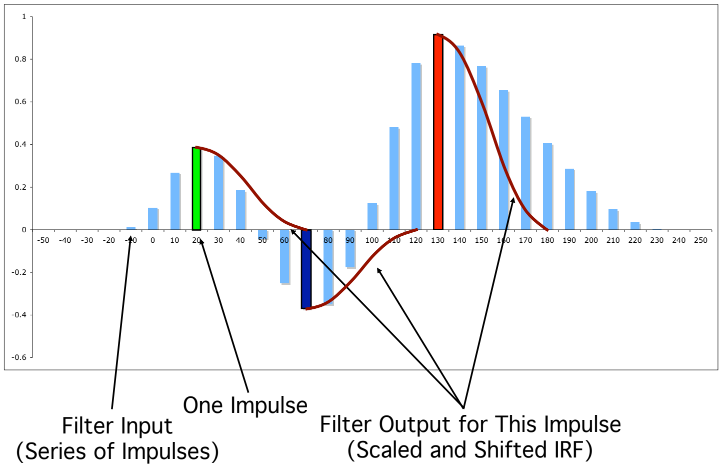
IRF amplitude is proportional to impulse amplitude
(IRF area = impulse height)
IRF duration is constant

Impulse-Response Function (IRF)



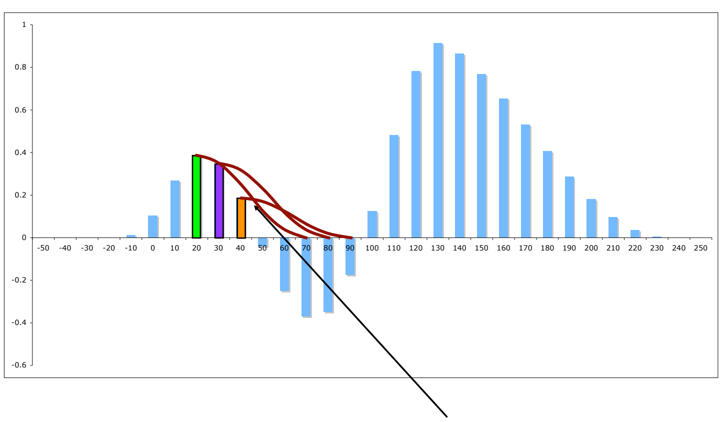
Filter Input
(Continuous ERP Waveform)

Impulse-Response Function (IRF)



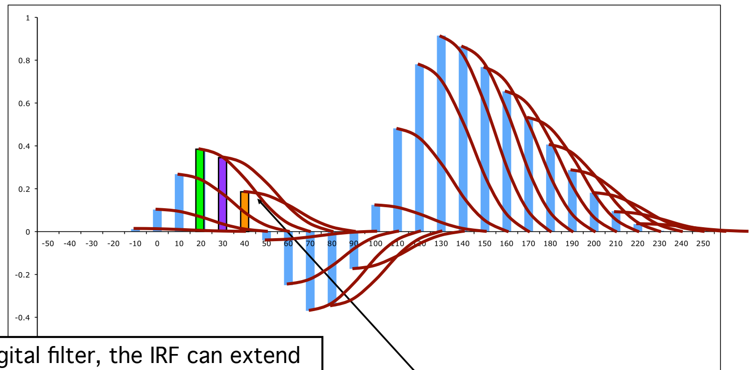
Each IRF is not actually as tall as the corresponding impulse (IRF area = impulse height)

Impulse-Response Function (IRF)



Each IRF is not actually as tall as the corresponding impulse (IRF area = impulse height)

Impulse-Response Function (IRF)



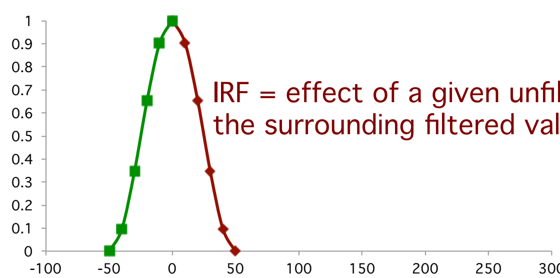
With a digital filter, the IRF can extend both forward and backward in time

These scaled responses simply sum together
(This is what it means when we say that the filter is "linear")

Each IRF is not actually as tall as the corresponding impulse
(IRF area = impulse height)

IRF vs. Weighting Function

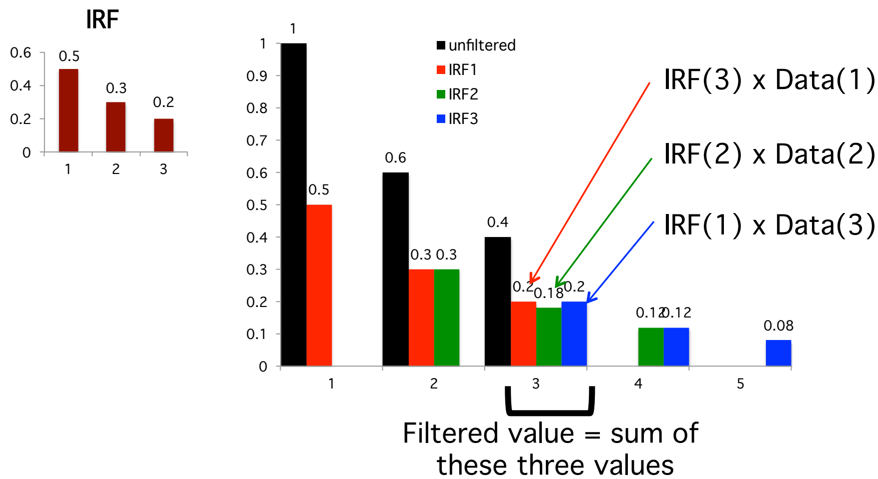
Weighting function = IRF reflected about time zero



IRF = effect of a given unfiltered value on the surrounding filtered values

Weighting function = effect of the surrounding unfiltered values on a given filtered value

IRF vs. Weighting Function



Convolution

$$fERP_i = \sum_{j=-n}^{n} W_j ERP_{i+j}$$

$$fERP_i = \sum_{j=-n}^{n} IRF_j ERP_{i-j}$$

IRF_j = impulse response function at time j
= weighting function at time -j

This is the “convolution” of IRF and ERP:

$$fERP = IRF * ERP$$

<u>Relation to filtering in the frequency domain:</u>
Convolution in the time domain =
Multiplication in the frequency domain

Convolution & Multiplication

Time Domain Representation

Unfiltered ERP Waveform * **Impulse Response Function** = **Filtered ERP Waveform**

Fourier Transform | Inverse Fourier Transform

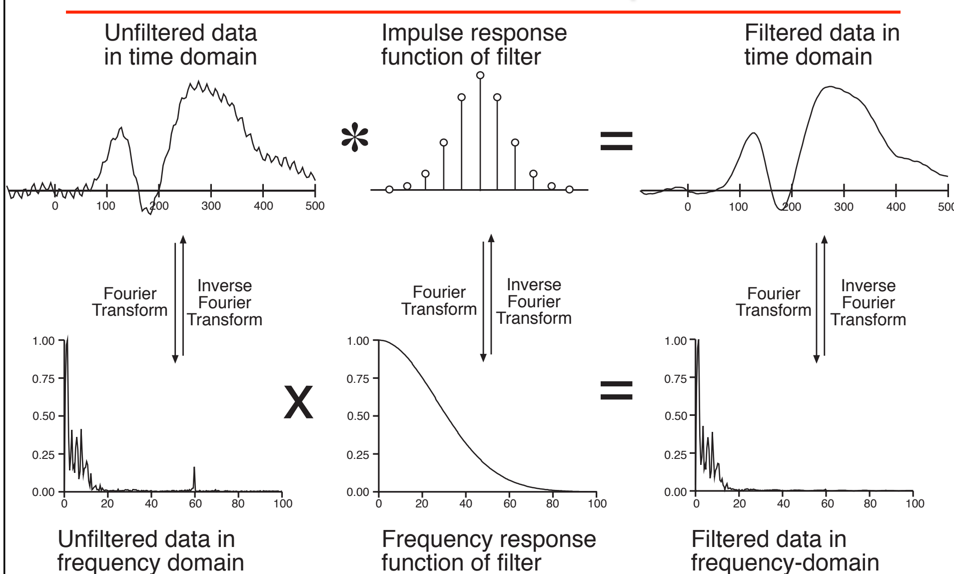
Fourier Transform | Inverse Fourier Transform

Fourier Transform | Inverse Fourier Transform

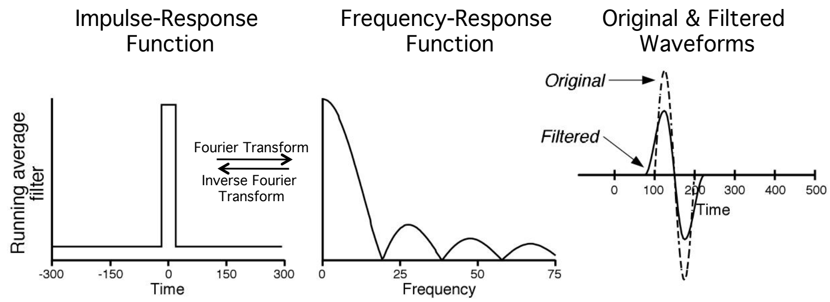
Unfiltered ERP Frequency Spectrum x **Transfer Function** = **Filtered ERP Frequency Spectrum**

Frequency Domain Representation

Convolution & Multiplication



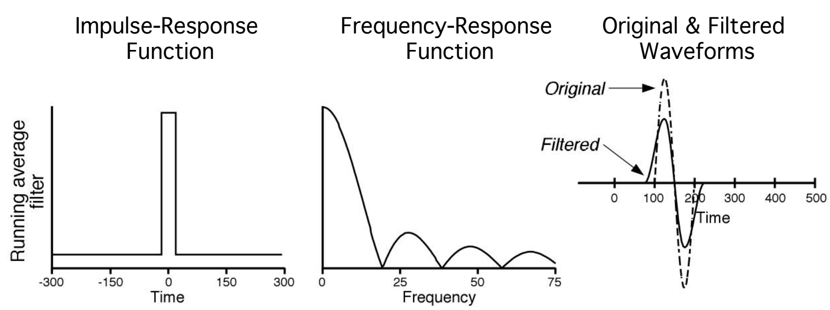
Examples of Low-Pass Filters



Sudden transitions of impulse-response function lead to "side lobes" in frequency-response function

This can be used to have a zero point at a particular frequency (e.g., 60 Hz)

Examples of Low-Pass Filters

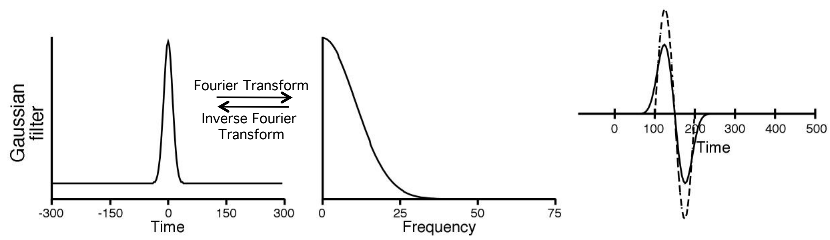


Sudden transitions of impulse-response function lead to "side lobes" in frequency-response function

This can be used to have a zero point at a particular frequency

Examples of Low-Pass Filters

Impulse-Response Function | Frequency-Response Function | Original & Filtered Waveforms

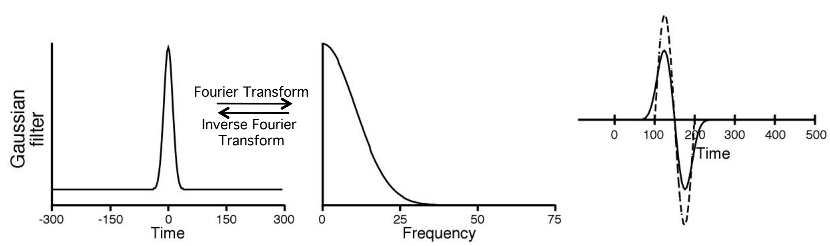


Gradual changes in Gaussian impulse-response function lead to smooth, monotonic rolloff

Gaussian is optimal trade-off between precision in the time and frequency domains

Examples of Low-Pass Filters

Impulse-Response Function | Frequency-Response Function | Original & Filtered Waveforms



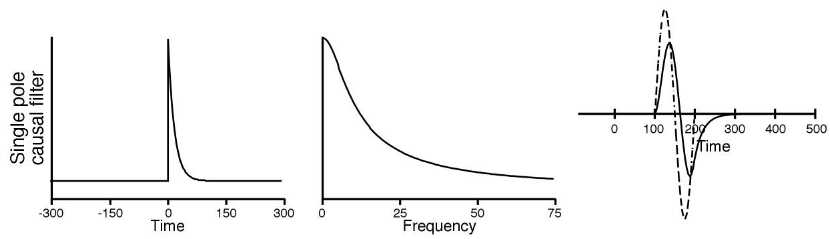
Gradual changes in Gaussian impulse-response function lead to smooth cutoff

Boxcar Gaussian

Gaussian is optimal trade-off between precision in the time and frequency domains

Examples of Low-Pass Filters

Impulse-Response Function | Frequency-Response Function | Original & Filtered Waveforms



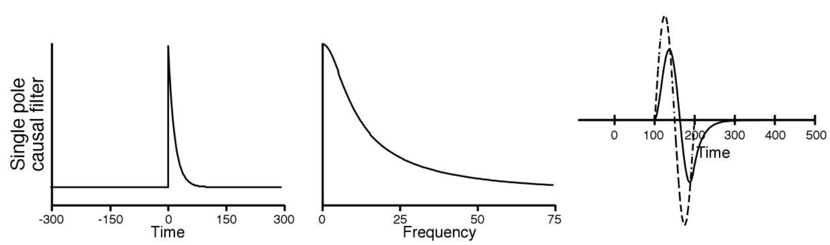
This is a “causal” filter: It’s impulse-response function is zero prior to time zero. Most digital filters are “non-causal.”

Analog filters in EEG amplifiers are causal

Causal filters tend to increase ERP latencies (phase shift)

Examples of Low-Pass Filters

Impulse-Response Function | Frequency-Response Function | Original & Filtered Waveforms

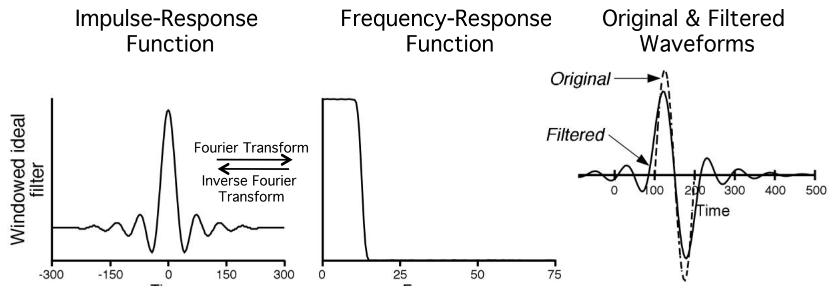


This is a “causal” filter: It’s impulse-response function is zero prior to time zero. Most digital filters are “non-causal.”

Analog filters in EEG amplifiers are causal

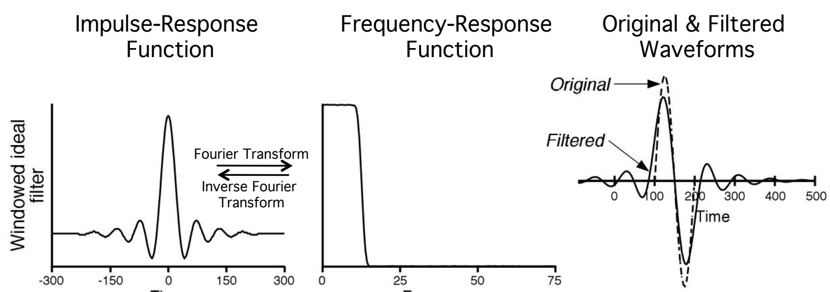
Causal filters tend to increase ERP latencies (phase shift)

Examples of Low-Pass Filters



This frequency-response function has a very steep rolloff but produces extreme time-domain distortions

Examples of Low-Pass Filters



This frequency-response function has a very steep cut-off but produces a large time-domain distortions

Properties of Convolution

Associative & Distributive Properties:

$$A * (B * C) = (A * B) * C$$

$$A * (B + C) = (A * B) + (A * C)$$

Filtering Twice:

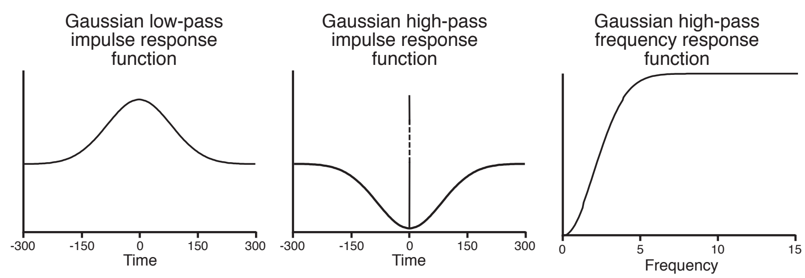
$$(ERP * IRF_1) * IRF_2 = ERP * (IRF_1 * IRF_2)$$

This gives you a wider impulse response function

What does this do to the frequency response function?

Answer: The two frequency response functions are multiplied by each other to create the combined frequency response function

High-Pass Filters



High-pass filtering involves subtracting the low frequencies from the unfiltered ERP

Consequently, the IRF of a high-pass filter is an inverted version of a low-pass filter

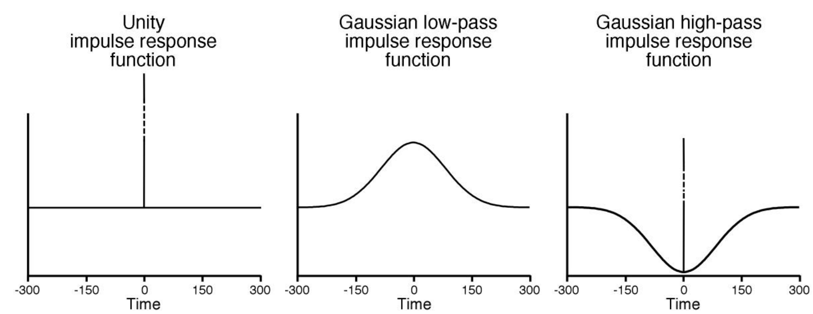
High-Pass Filters

$ERP_H = ERP - (IRF_L * ERP)$

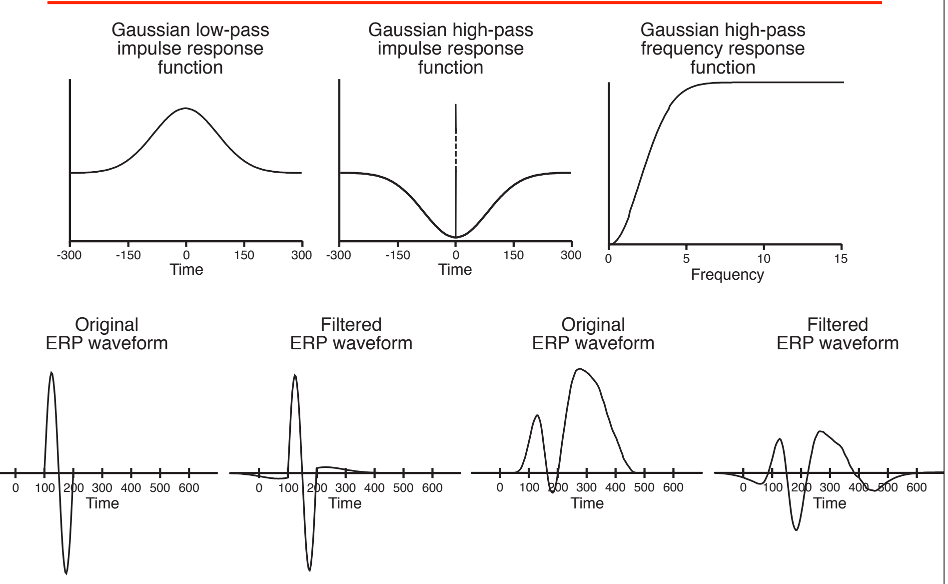
$= (IRF_U * ERP) - (IRF_L * ERP)$ [because ERP = IRF_U * ERP]

$= (IRF_U - IRF_L) * ERP$ [because of the distributive property]

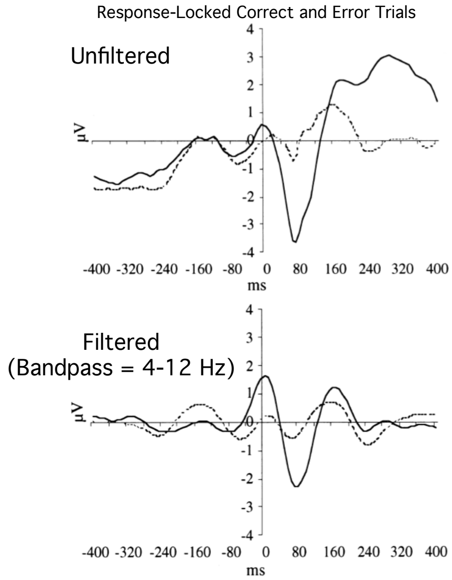
$= IRF_H * ERP$, where $IRF_H = IRF_U - IRF_L$



High-Pass Filters



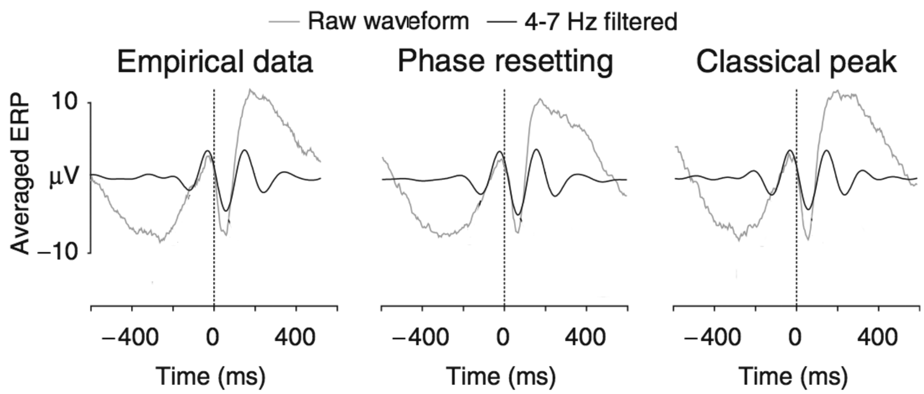
Are Filter Artifacts a Real Problem?



Luu & Tucker (2001)

"By filtering out the large slow waves of the event-related potential, we found that the error-related negativity (Ne/ERN) arises from a midline frontal oscillation..."

Are Filter Artifacts a Real Problem?



Extreme filtering produces an oscillating output, regardless of whether the input contains true oscillations

Yeung et al. (2007)

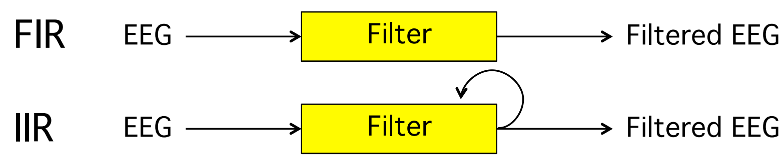
Recommendations- Online Filtering

- Filter as little as possible online
  - You can't "unfilter" filtered data
- Low-pass filter at 1/3–1/4 times the sampling rate
- Notch filters are usually OK if needed
- If you have 24+ bits
  - Record at DC
  - Record ~20 s of blank at beginning and end of each trial block
- If you have <24 bits
  - High-pass filter cutoff between .01 and .1 Hz to avoid saturating the amplifier/ADC
    - .01 Hz for ideal conditions
    - 0.1 Hz for less-than-ideal conditions

Recommendations- Offline Filtering

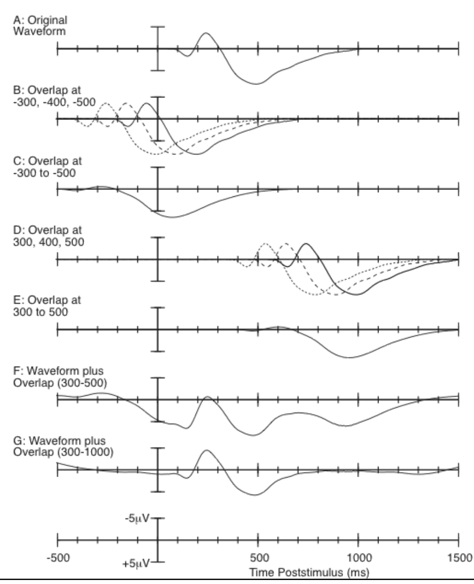
- Low-pass filter cutoff at 20–40 Hz to reduce noise during plotting or when measuring peak amplitudes or latencies
- High-pass filter cutoff at 0.01–0.1 Hz if you recorded at DC
  - 0.1 Hz usually gives best statistical power with minimal distortion
  - Avoid strong high-pass filters unless absolutely necessary
  - Do not use >0.1 Hz unless you really know what you're doing
- Use Gaussian impulse-response functions
  - If not, at least know what the impulse response function is
  - You can find out by filtering a brief impulse
- If in doubt, try filtering a fake waveform to see what kinds of distortion are produced by the filter
- Apply high-pass filter to EEG, not ERP (avoid edge effects)
- Apply low-pass filter to ERP when plotting or measuring nonlinear features (e.g., peaks), but not when measuring mean amplitude
- You may want to filter prior to artifact rejection if this helps you to identify real artifacts

Infinite Impulse Response Filters



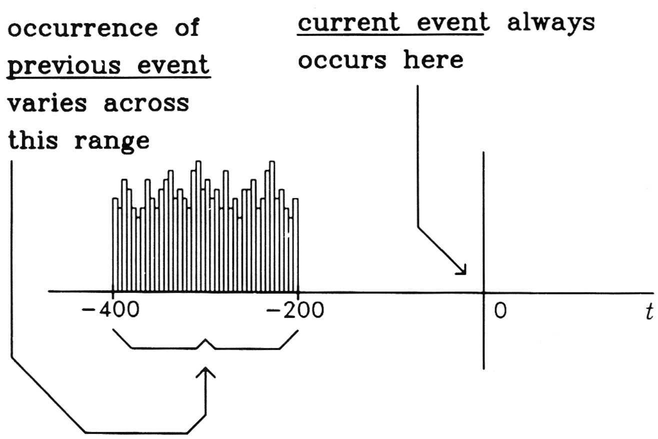
- The good:
  - IIR can achieve a sharper rolloff with fewer data points
- The bad:
  - Can be unstable (but typically OK for noncausal filters)
  - Harder to predict the effects by looking at IRF
  - Anything called “infinite” is scary
- My experience with Butterworth IIR filters:
  - No worse than FIR filters in terms of filter artifacts

Overlap



Overlap can be modeled as a convolution of the ERP waveform and the distribution of SOAs

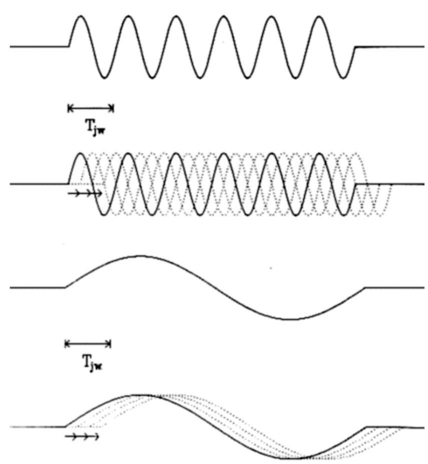
Overlap



Woldorff (1993)

Overlap for current event is equal to the ERP waveform shifted in time and scaled by the frequency of occurrence of each SOA (convolution of ERP waveform and distribution of SOAs)

Overlap

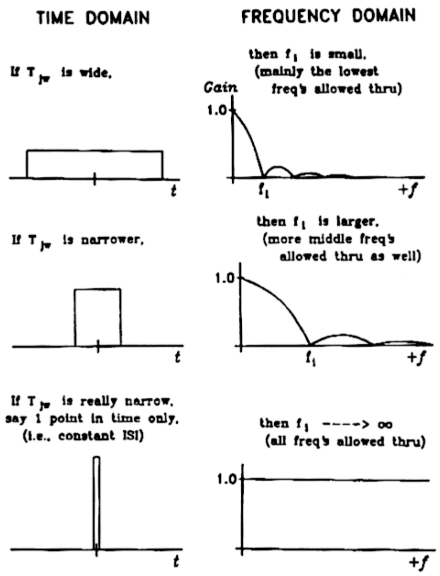


Woldorff (1993)

High frequencies tend to be filtered out well by a modest jittering of the SOA; low frequencies are not filtered very well

ERP Boot Camp

Overlap



A jittered SOA is like a boxcar low-pass filter

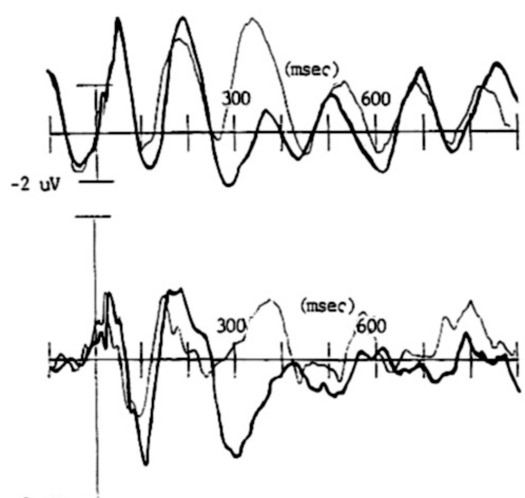
The wider the SOA jitter, the lower is the half-amplitude cutoff of the filtering effect of the jitter

Woldorff (1993)

Overlap

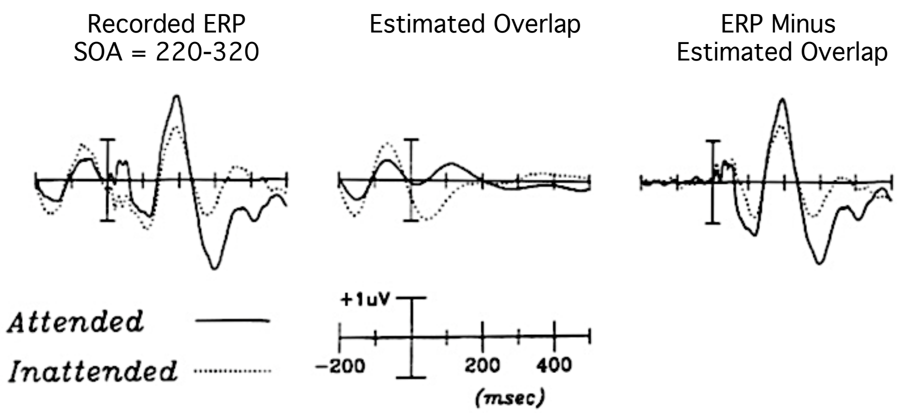
SOA = 130-190

SOA = 125-275



Woldorff (1993)

Overlap – ADJAR Filter



Woldorff (1993)