Denoising:

* **Running mean time series** - this filter is good for a signal with normal distributed noise:



* **Gaussian smoothing time series** – like mean smoothing but a bit more smooth product:



W is the full-width at half maximum. Higher w means wider window

* TKEO:



In this way we can **highlight peaks and anomalies in data** which is quite noisy but also has pulses or peaks

* **Find median to remove spike noise – good for spikes only**:

First sort the vector, then find the middle value which is the median.

If we have an even number of numbers in the vector, we average both middle values after sorting.

In both we pick a threshold for what the highest (or lowest depends on the peak) we approve and then replace it with the median. One good way to choose threshold is by making a histogram of all the points and see where the gaussian bell ends

* **Linear detrending:**

Just using detrend to detrend a linear trend in data (after visualizing it)

* **Remove nonlinear trends from data** (remove fluctuations or drifts in data):

It is like linear detrending but now we must use baes criteria to determine what’s the order of the polynom we are fitting to. Bayes information criterion:



It evaluates the fit of a model to the data. Epsilon is the distance between the modeled polynom and the actual data. N is the number of data points k is the number of parameters (how many orders we have in the polynom) y with hat is the predicted data and y without is the real data. Then we plot b against polynomial order and pick the **lowest b. we can also find lowest b by looking for min in b vector**

* Averaging multipile repetitions in order to get better SNR. Simple average but we need to know when it happens or if it is a recuring event in time (מחזורי).
* Remove artifact via least-squares and template matching. Good if we have a second channel that records the artifacts. The least square algorithm:



Where beta is the regression weights X is the design matrix, y is the looked on data, r is the residual and X times beta is the predicted data. This all is real matrixes. X matrix is two column vector the first column is all 1’s and the second is the times series of the artifact vector

Fourier transform:

* **Nyquist** is half the sampling rate.
* We should always remember to **factor out the DC frequency**. We can do that in two ways:
* One is detrending. But then a real trend might be missed so it doesn’t always fit.
* Subtract the mean value from the data (mean centering). But then the first point is not real but stems from a trend if it is present
* **Welch's method:** after ‘cutting’ the signal into pieces apply it to each part independently. We should always apply hann window on every piece to minimize edge effects.
* Welch’s method is not always optimal if the frequencies are changing a lot. Then we would like to use wavelets
* **We can see time and frequency domain at the same time on a spectrogram. Its frequency as function of time and it gives a heat map representing the amplitude.**

Digital Filtering:

* **Procedure for filtering data:**

1. Define frequency domain shape and cut-offs.
2. Generate filter kernel
3. Evaluate kernel and its power spectrum
4. Apply filter kernel to data

* **FIR vs IIR:**

|  |  |  |
| --- | --- | --- |
| Name | FIR | IIR |
| Full name | Finite Impulse Response | Infinite Impulse Response |
| Kernel length | Long | Short |
| Speed | Slower | Fast |
| Stability | High | Data-dependent |
| Mechanism | Multiply data with kernel | Multiply data with data |

* **FIR kernels with firls: finite impulse response Least-squares linear-phase kernels:**

With filters we should always stay with simple window on the frequency domain. The kernel goes from zero (DC) to Nyquist they are always normalized by nyquist frequency. the gain goes until one (the amplitude of the window). The window of band pass filter goes to a maximum between the two frequencies we want to pass through it. We don’t want the edges of the kernel on the freq domain because it requires too much energy – this is the transition zone. The order (number of points in the kernel on the time domain) is usually making the kernel better with higher numbers but there is a limit to it where after the limit it will be worse and also take longer to compute.