CogSci109 Lecture 6

Wed, Oct. 10, 2007

Fourier transforms, Low-pass filtering, High-pass filtering, two filters and their code

Outline for today

- Announcements
- Review of last time
- A bit more about linearity vs. nonlinearity and why this is an important point in modeling
- A bit more about Fourier analysis, frequency response, and how to do them in matlab
- Low-Pass filtering your data
  - **Moving average**
  - **Recursive low pass filtering**
- High-Pass filtering your data
  - **How to derive a high pass filter from a low-pass filter**

Announcements

- Recordings are up
- OCE accounts for remote login
- Homework assigned Friday
  - **I want to go through more info first, and give you a reading or two which will help you with your assignment**
  - **Reading will be assigned tonight late, please at least look over it before starting the assignment on Friday**

Last time -

- We went over several examples of discretization, sampling and aliasing
- Mentioned fourier transforms and frequency analysis
    - **We'll need that for our discussion of filtering today**

More on linearity vs. nonlinearity

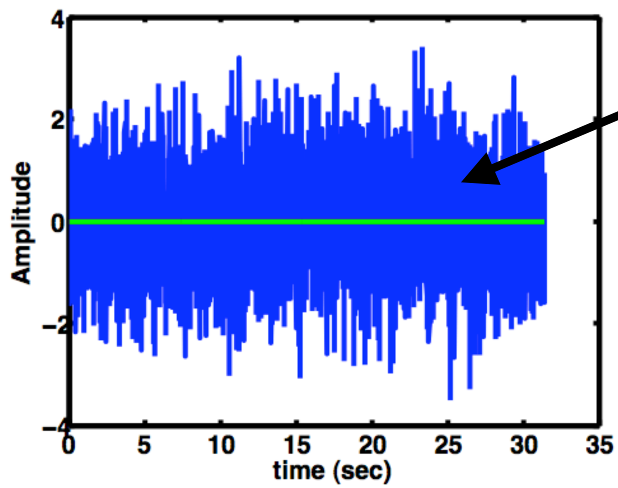
- Power
  - **A linear system is a system whose dependent variables are related to its independent variables by a power of one**
- Linear systems have these particular properties (and they are very favorable)
  - **Additivity** $T[x_1(n) + x_2(n)] = T[x_1(n)] + T[x_2(n)]$
  - **homogeneous** $T[cx(n)] = cT[x(n)]$
- Linear differential equations are more well-understood than nonlinear differential equations

Fourier transforms

- Frequency domain example : Musical note vs. the sound
  - **More parsimonious to describe a song in terms of its notes than time domain signal (when creating a ‘model’ for a song which can be communicated)**
- Reading will cover the mathematical details and be more in depth
  - **I will provide example code for performing Fourier analysis, and computing a Frequency response in Matlab**
- We’ll come back to fourier transforms when creating basic models, and analyzing the properties of the filters we discuss today

We return to noisy data which we want to 'clean up'

- We do this by removing undesired components of the signal
- One way to do this is *averaging* out the noise
- If it's Gaussian and additive…

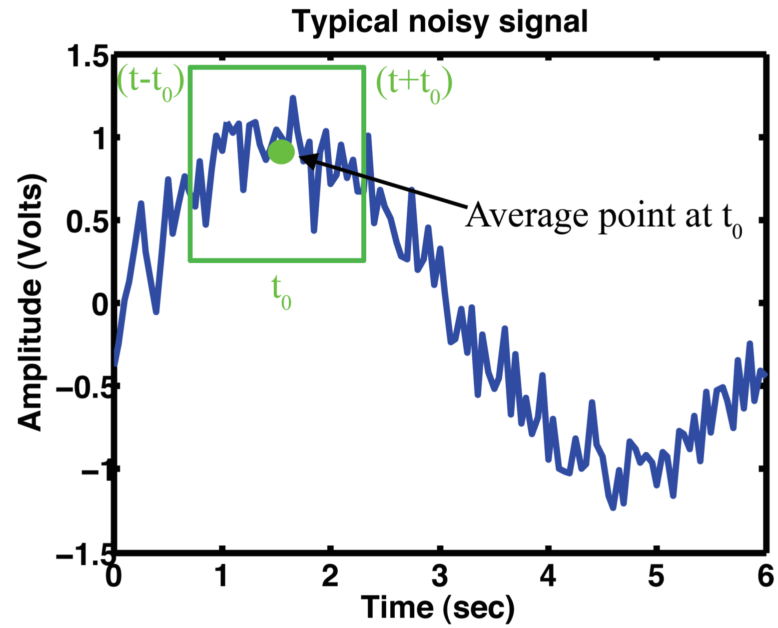


This is gaussian noise, and the average of this is approximately the green line, 0

$-5 + 5 = 0$

How to do it

- Decide on a 'window' of data to average over, which is narrower than the fastest component to your changing signal
- Sum up over that window of points and divide by the number of points (average)



Continuous form

$$x_f(t) = \int_{t-t_0}^{t+t_0} x(\tau)d\tau$$

Discrete form

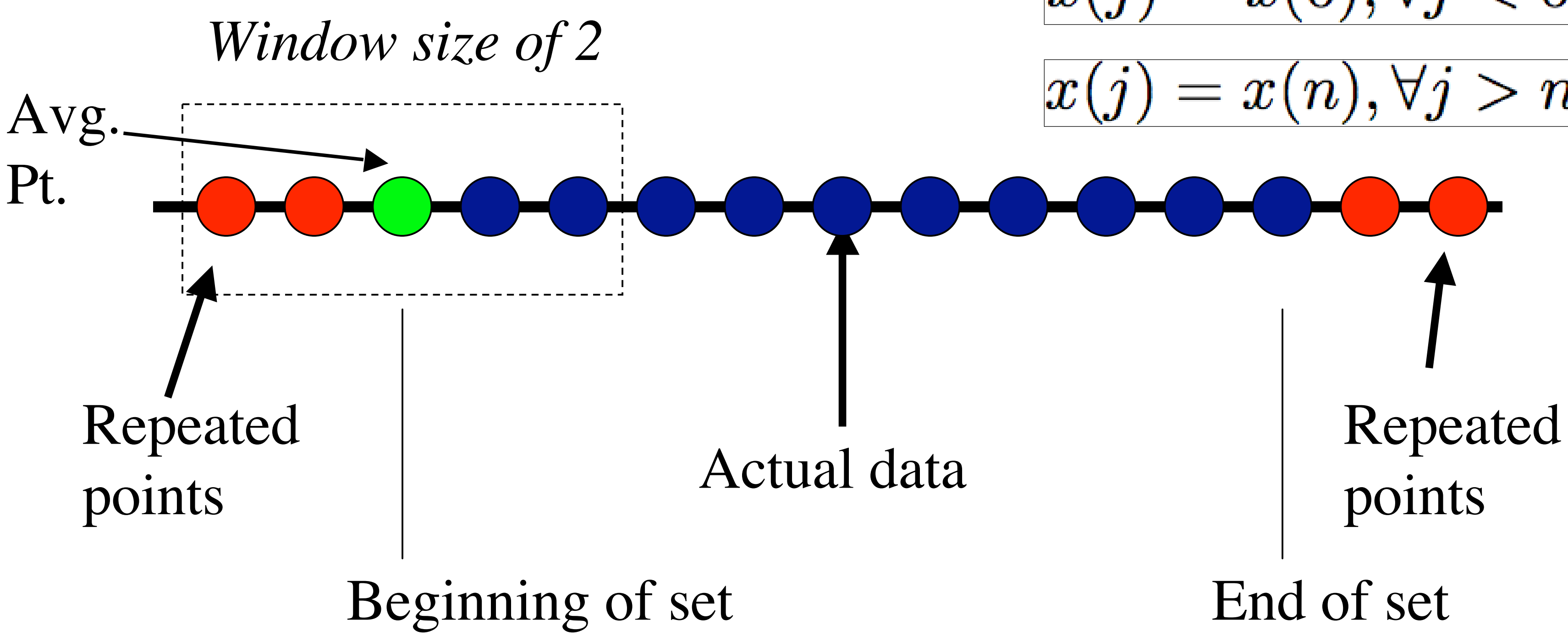
$$x_f(i) = \frac{1}{2k+1}\sum_{j=i-k}^{i+k} x(j)$$

A few details

- What about at the ends of the data where we don't have information before (at the beginning of the data set) or after (at the end of the data set)?
  - **Copy the first or last point and repeat as necessary**

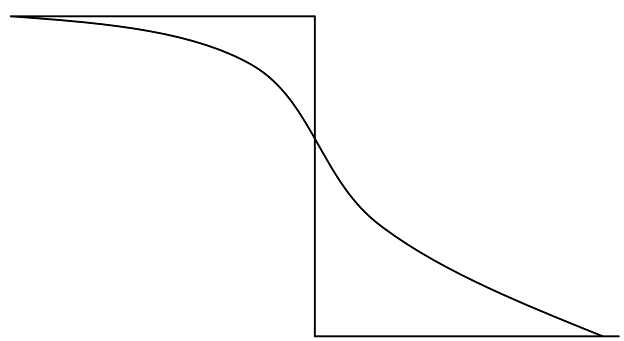
$$x(j) = x(0), \forall j < 0$$

$$x(j) = x(n), \forall j > n$$



Disadvantages…

- Need to have all data in memory already, so it isn't an 'online' filter
- Causality
  - **If we care about an exact event timing, this is a poor filter to use:**



Signal anticipates changes!

Solution…

- Recursive filter
  - **Simple to implement**
  - **Solves causality problems**