

# Digital filters

Eugeniy E. Mikhailov

The College of William & Mary



Lecture 27

# DFT filters (repeat)

Once you get a signal you can filter unwanted components out of it. The recipe is the following

- sample the signal
- calculate FT (fft)
- have a look at the spectrum and decide which components are unwanted
- apply filter which attenuate unwanted frequency component (remember that if you attenuate the component of the frequency  $f$  by  $g_f$  you need to attenuate the component at  $-f$  by  $g_f^*$ ).
- calculate inverse FT (ifft) of the filtered spectrum
- repeat if needed

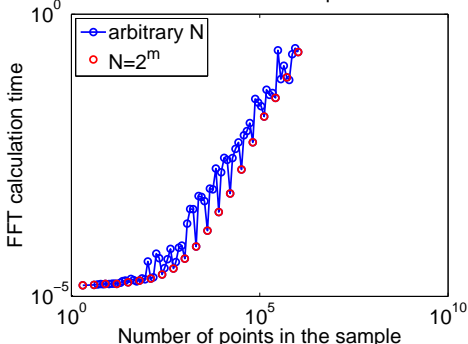
$$y_{filtered}(t) = \mathcal{F}^{-1} [\mathcal{F}(y(t))G(f)] = \mathcal{F}^{-1} [Y(f)G(f)]$$

# Speed of FFT

- The main work horse of the DFT filters is FFT algorithm
- it is handy to know its performance behavior

$$y_k = \frac{1}{N} \sum_{n=0}^{N-1} c_n \exp(i \frac{2\pi(k-1)n}{N}) \quad \text{inverse Fourier transform}$$

FFT calculation time vs number of points in the sample

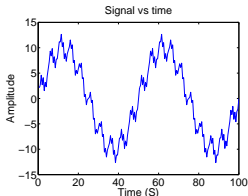


- General DFT scales  $\sim N^2$  ( $N$  coefficient each involving sum of  $N$ )
- FFT scales  $\sim N \log_2 N$  which is great speed up

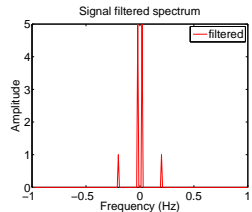
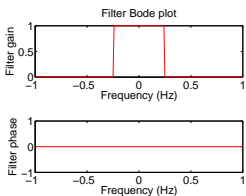
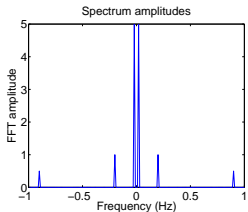
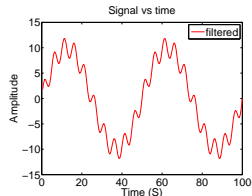
The fastest calculation time for

$$N = 2^m$$

# Brick wall low-pass filter



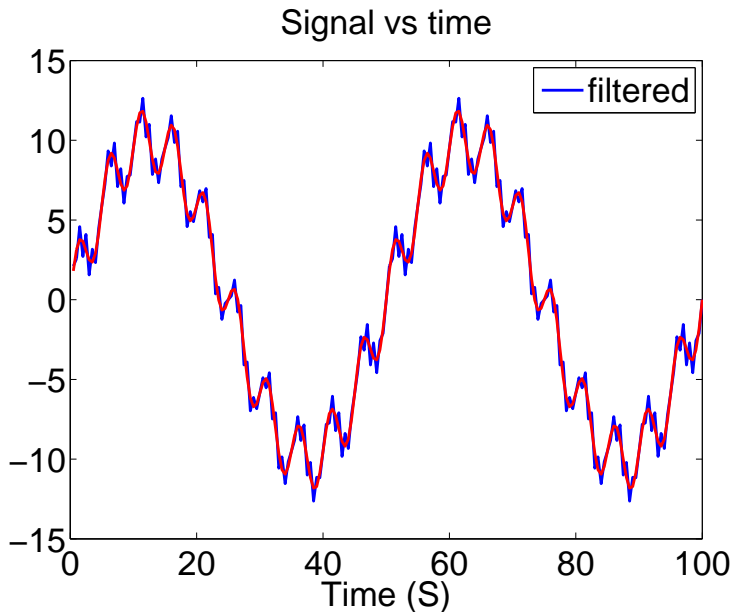
$$\text{Filter gain function} \\ G(f) = \begin{cases} 1, & |f| \leq f_{cutoff} \\ 0, & |f| > f_{cutoff} \end{cases}$$



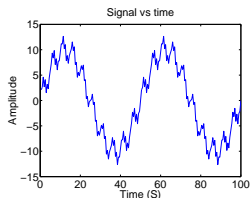
$$y_{filtered}(t) = \mathcal{F}^{-1} [\mathcal{F}(y(t))G(f)] = \mathcal{F}^{-1} [Y(f)G(f)]$$

```
freq=fourier_frequencies(SampleRate, N);  
G=ones(N,1); G( abs(freq) > Fcutoff, 1) = 0;  
y_filtered = ifft( fft( y ) * G )
```

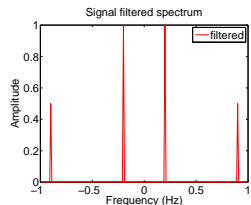
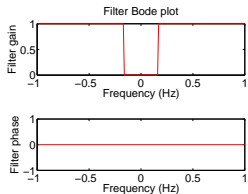
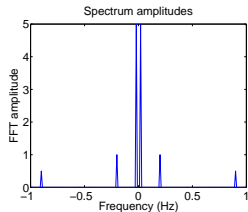
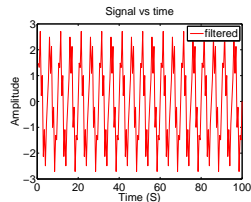
# Brick wall low-pass filter (continued)



# Brick wall high-pass filter



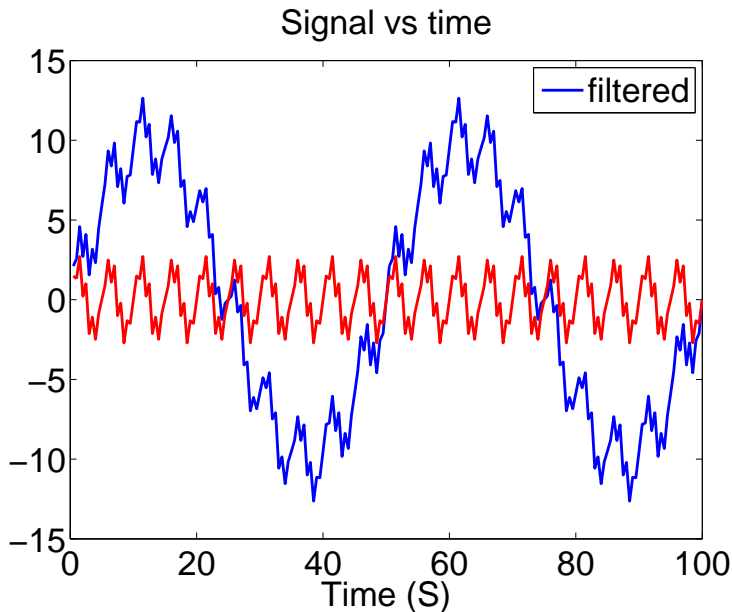
$$\text{Filter gain function} \\ G(f) = \begin{cases} 1, & |f| \geq f_{cutoff} \\ 0, & |f| < f_{cutoff} \end{cases}$$



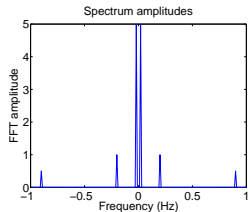
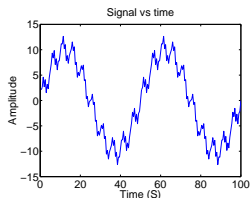
$$y_{filtered}(t) = \mathcal{F}^{-1} [\mathcal{F}(y(t))G(f)] = \mathcal{F}^{-1} [Y(f)G(f)]$$

```
freq=fourier_frequencies(SampleRate, N);  
G=ones(N,1); G( abs(freq) < Fcutoff, 1) = 0;  
y_filtered = ifft( fft( y ) * G )
```

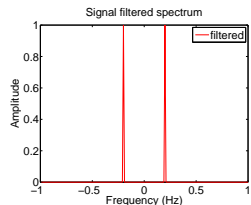
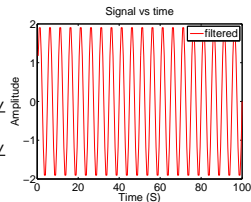
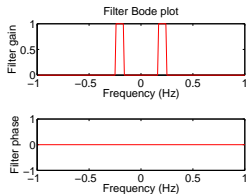
# Brick wall high-pass filter (continued)



# Brick wall band-pass filter



$$\text{Filter gain function} \\ G(f) = \begin{cases} 1, & ||f| - f_c| \leq \frac{f_{bw}}{2} \\ 0, & ||f| - f_c| > \frac{f_{bw}}{2} \end{cases}$$

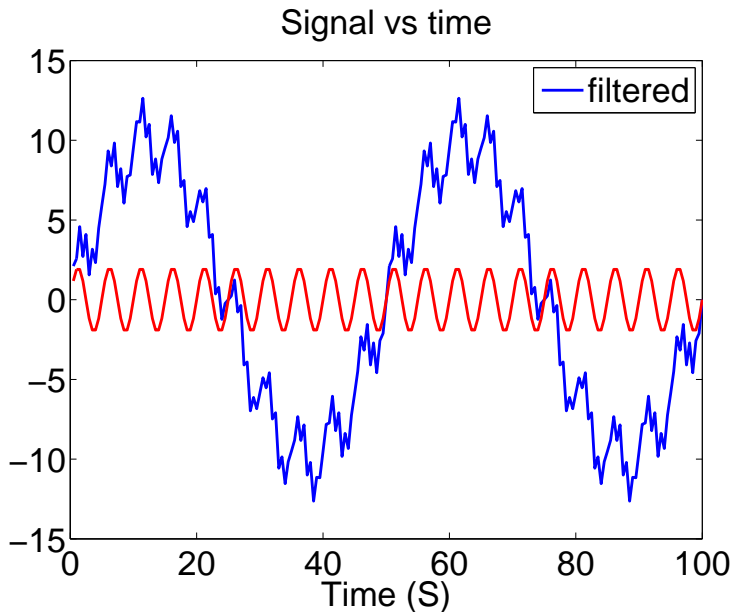


$$y_{filtered}(t) = \mathcal{F}^{-1} [\mathcal{F}(y(t))G(f)] = \mathcal{F}^{-1} [Y(f)G(f)]$$

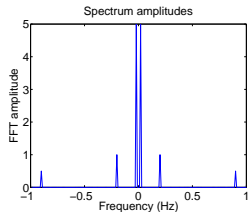
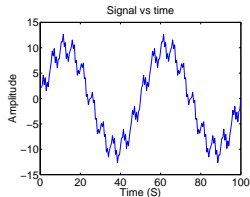
```
freq=fourier_frequencies(SampleRate, N);  
G=ones(N,1); G( abs(abs(freq)-Fcenter) > BW/2, 1)=0;  
y_filtered = ifft( fft( y ) * G )
```



# Brick wall band-pass filter (continued)

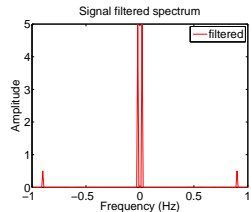
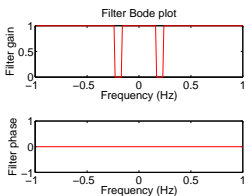
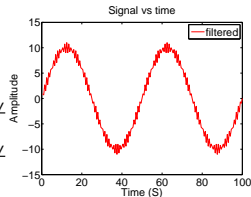


# Brick wall band-stop filter



Filter gain function

$$G(f) = \begin{cases} 0, & ||f| - f_c| \leq \frac{f_{bw}}{2} \\ 1, & ||f| - f_c| > \frac{f_{bw}}{2} \end{cases}$$

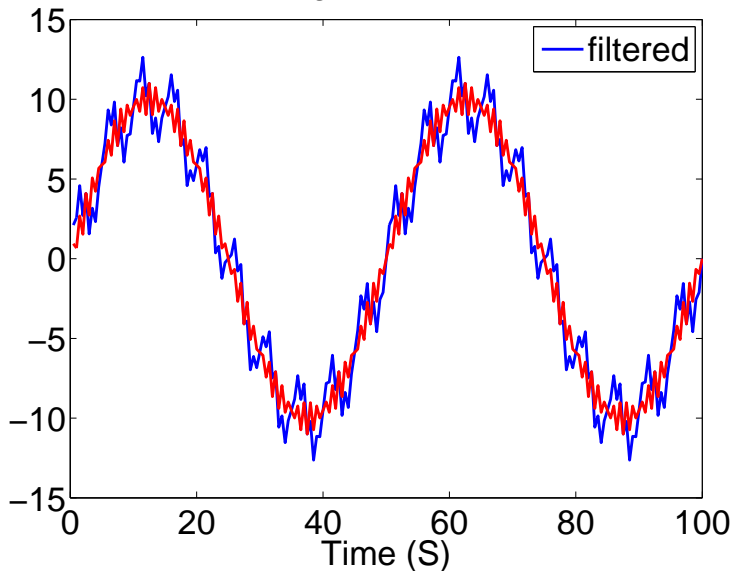


$$y_{filtered}(t) = \mathcal{F}^{-1} [\mathcal{F}(y(t))G(f)] = \mathcal{F}^{-1} [Y(f)G(f)]$$

```
freq=fourier_frequencies(SampleRate, N);  
G=zeros(N,1); G( abs(abs(freq)-Fcenter) > BW/2, 1)=1;  
y_filtered = ifft( fft( y ) * G )
```

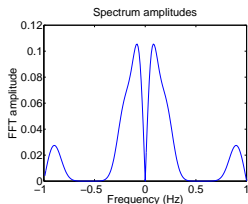
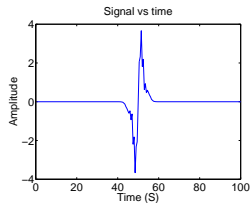
# Brick wall band-cut filter (continued)

Signal vs time



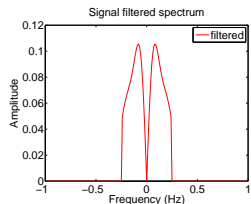
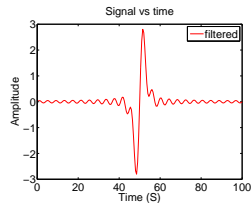
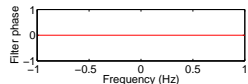
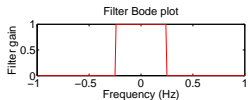
# Brick wall filters artifacts

Sharp features in Fourier spectrum produce ring-down like signals

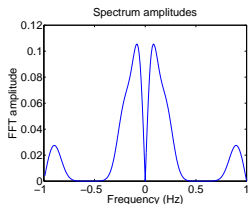
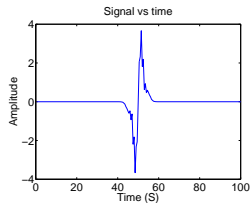


Filter gain function

$$G(f) = \begin{cases} 1, & |f| \leq f_{cutoff} \\ 0, & |f| > f_{cutoff} \end{cases}$$

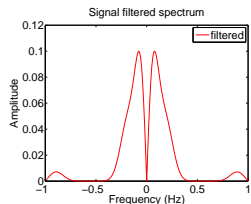
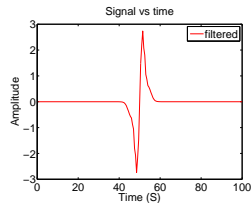
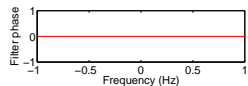
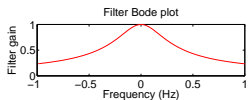


Sharp features in Fourier spectrum produce ring-down like signals



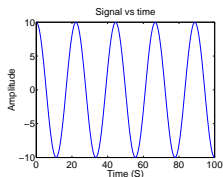
Filter gain function

$$G(f) = \left| \frac{1}{1 + if/f_{cutoff}} \right|$$



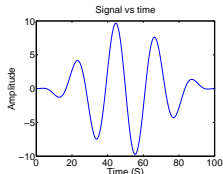
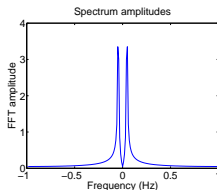
# Windowing artefact

Similarly sharp features in time lead to broadening of the spectrum



Implicitly assumed  
Rectangular window

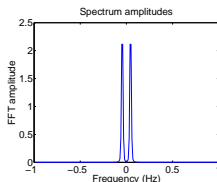
$$w_n = 1$$



$$y_{windowed_n} = y_n w_n$$

Hann window

$$w_n = \frac{1}{2} \left( 1 - \cos\left(2\pi \frac{n-1}{N-1}\right) \right)$$



- Note: spectral resolution  $\sim 1/T_{window}$ .

Search for other windowing functions: Hamming, Tukey, Cosine, Lanczos, Triangulars, Gaussians, Bartlett-Hann, Blackmans, Kaisers.