# Digital filters

Eugeniy E. Mikhailov

The College of William & Mary

Lecture 25

Notes



## DFT filters (repeat)

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

- sample the signal
- calculate DFT (use Matlab `fft`)
- have a look at the spectrum and decide which frequencies are unwanted
- apply a filter which attenuates unwanted frequencies amplitudes
  - If you attenuate the component of the frequency $f$ by $g_f$, you need to attenuate the component at $-f$ by $g_f^*$. Otherwise, the inverse Fourier transform will have non zero imaginary part.
- calculate inverse DFT (`ifft`) of the filtered spectrum
- repeat if needed

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

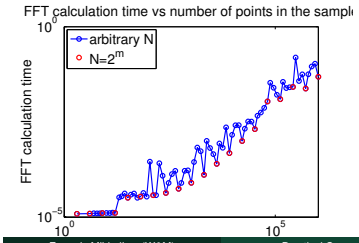
Notes



## Speed of FFT

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

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




- A naive DFT implementation scales $\sim N^2$ ($N$ coefficient each involving sum of $N$)
- The FFT scales $\sim N \log_2 N$. This is a fantastic speed up

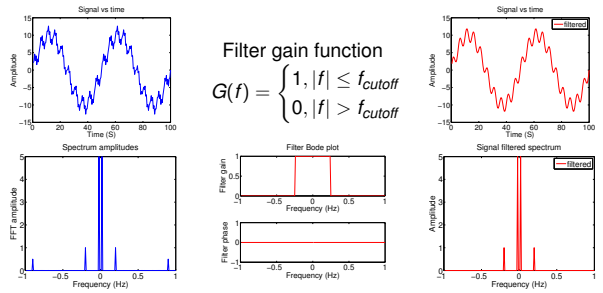
The fastest calculation time for

$$N = 2^m$$

Notes



## Brick wall low-pass filter

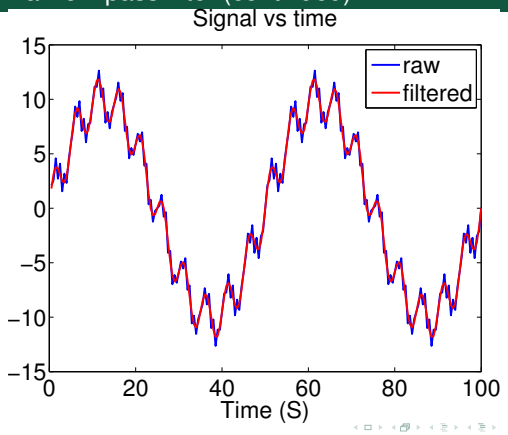



$$y_{filtered}(t) = \mathcal{F}^{-1}\left[\mathcal{F}(y(t))G(f)\right] = \mathcal{F}^{-1}\left[Y(f)G(f)\right]$$

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

Notes

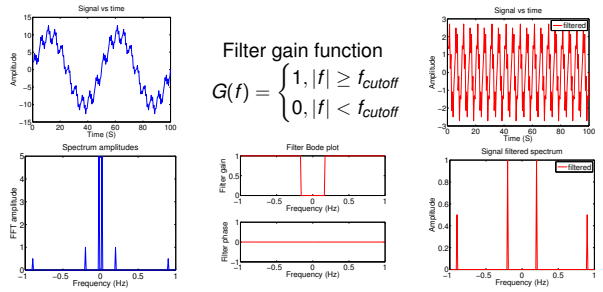
## Brick wall low-pass filter (continued)






Notes

## Brick wall high-pass filter




Filter gain function

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

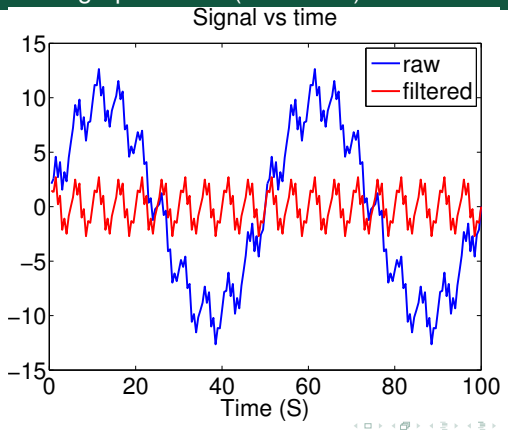
$$y_{filtered}(t) = \mathcal{F}^{-1}\left[\mathcal{F}(y(t))G(f)\right] = \mathcal{F}^{-1}\left[Y(f)G(f)\right]$$

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



Notes

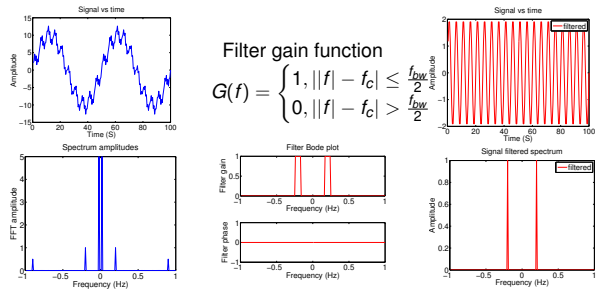
## Brick wall high-pass filter (continued)






Notes

## Brick wall band-pass filter




Filter gain function

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

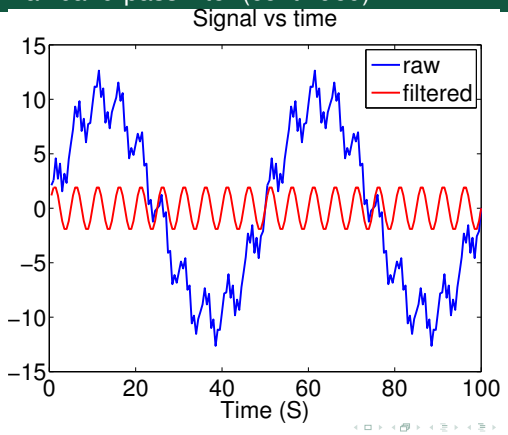
$$y_{filtered}(t) = \mathcal{F}^{-1}\left[\mathcal{F}(y(t))G(f)\right] = \mathcal{F}^{-1}\left[Y(f)G(f)\right]$$

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



Notes

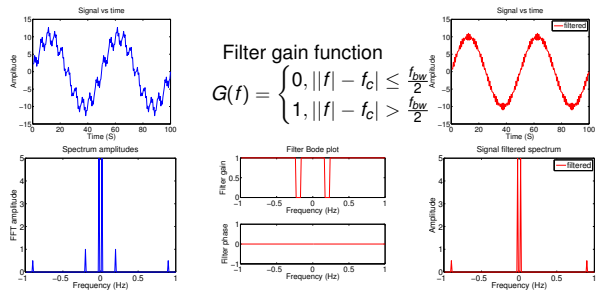
## Brick wall band-pass filter (continued)






Notes

## Brick wall band-stop filter




Filter gain function

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

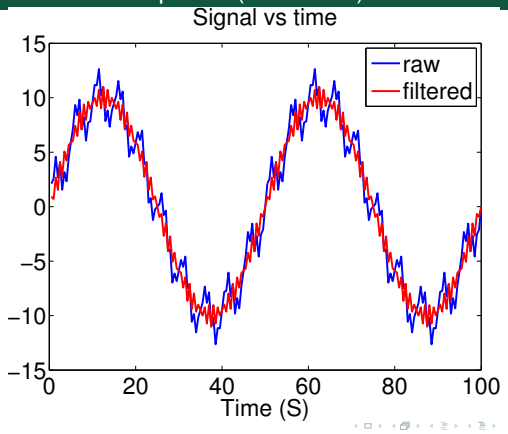
$$y_{filtered}(t) = \mathcal{F}^{-1}\left[\mathcal{F}(y(t))G(f)\right] = \mathcal{F}^{-1}\left[Y(f)G(f)\right]$$

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



Notes

## Brick wall band-stop filter (continued)

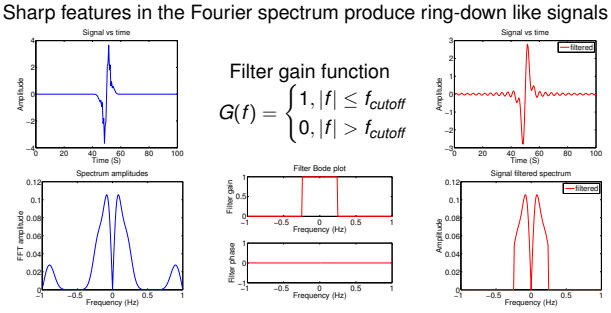





Notes

## Brick wall filters artifacts

Sharp features in the Fourier spectrum produce ring-down like signals




Filter gain function

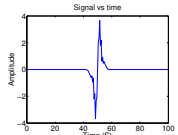
$$G(f) = \begin{cases} 1, |f| \leq f_{cutoff} \\ 0, |f| > f_{cutoff} \end{cases}$$



Notes

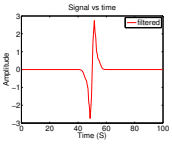
## Low pass smoothed

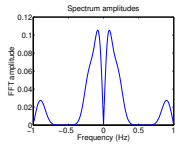
Sharp features in the Fourier spectrum produce ring-down like signals

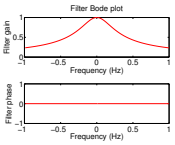


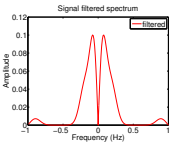

Filter gain function

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



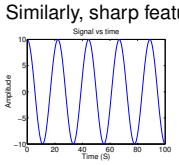





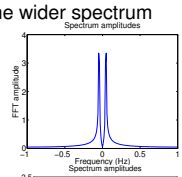

## Windowing artifacts

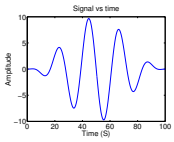
Similarly, sharp features in the time domain take the wider spectrum




Implicitly assumed
Rectangular window

$$w_n = 1$$

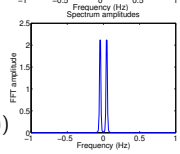



$$y_{windowed_n} = y_n w_n$$

The Hann window

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



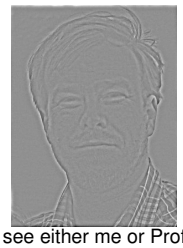

- Note: spectral resolution $\sim 1/T_{window}$.

Search for other windowing functions: Hamming, Tukey, Cosine, Lanczos, Triangulars, Gaussians, Bartlett-Hann, Blackmans, Kaisers. They all decrease a signal at the beginning and at the end to zero.



## Other DFT applications

Fun one: merging of two dimensional high- and low-pass image filters



Depending on distance to the image, you should see either me or Prof. Novikova in the middle.
To see the other person in the image, either step aside or decrease zoom till you do not see details on the right most image.
If you can take off your glasses, the illusion is stronger.



Notes

Notes

Notes

Notes