Table of Contents

.... 1
Creating FIDs 1
Generate spectra 2
Adding Noise 3
Apodization with 5Hz decaying exponential filter & SNR 4
SNR calculation 5 Hz 5
Apodization with 20Hz decaying exponential filter & SNR 6
SNR calculation 20 Hz 6
ppm referencing 7

```
% MRS/MRSI Processing Lab
% Demo #1 -Single voxel MRS, FID --> spectrum
%
% By the end of this portion of the lab you should be familiar with
 the
% following preprocessing steps associated with single voxel MRS
 recon:
%  (1) generation of a spectrum from an FID
%  (2) the effects of apodization
%  (3) the effects of zero-filling
%  (4) conversion between time, frequency, and ppm
%
%
 -------------------------------------------------------------------------


clear all;
close all;

%==============================================================
```

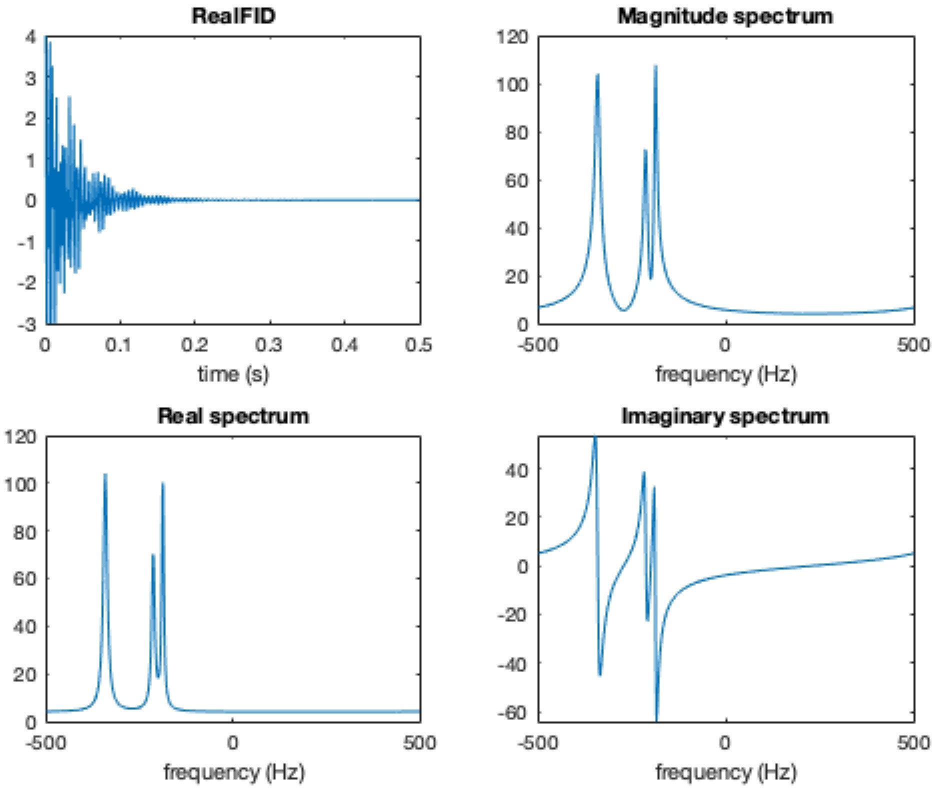
Creating FIDs

```
%set up parameters for FID
dwt=.001;  %dwell time in seconds
n=512;  %datapoints
t=0:dwt:dwt*(n-1); %time of fid signal
%t=dwt:dwt:dwt*(n);

%amplitude of each peak
c_a=2;
c_b=2;
c_c=4;


%linewidths in Hz
k_a=30;
```

```
k_b=20;
k_c=40;


%peak location in Hz
%w_a=0;
w_a=(3.0-4.67)*42.7*3;
w_b=(3.2-4.67)*42.7*3;
w_c=(2-4.67)*42.7*3;

%generate FID
fid = (c_a*exp(-k_a*t).*exp(2*pi*j*w_a*t))+(c_b*exp(-
k_b*t).*exp(2*pi*j*w_b*t)+(c_c*exp(-k_c*t).*exp(2*pi*j*w_c*t)));

% 1) The sweepwidth of the spectrum is 1000 Hz
% 2) A larger linewidth gives a lower peak height due to faster T2
% relaxation, which will affect SNR calculation.
```

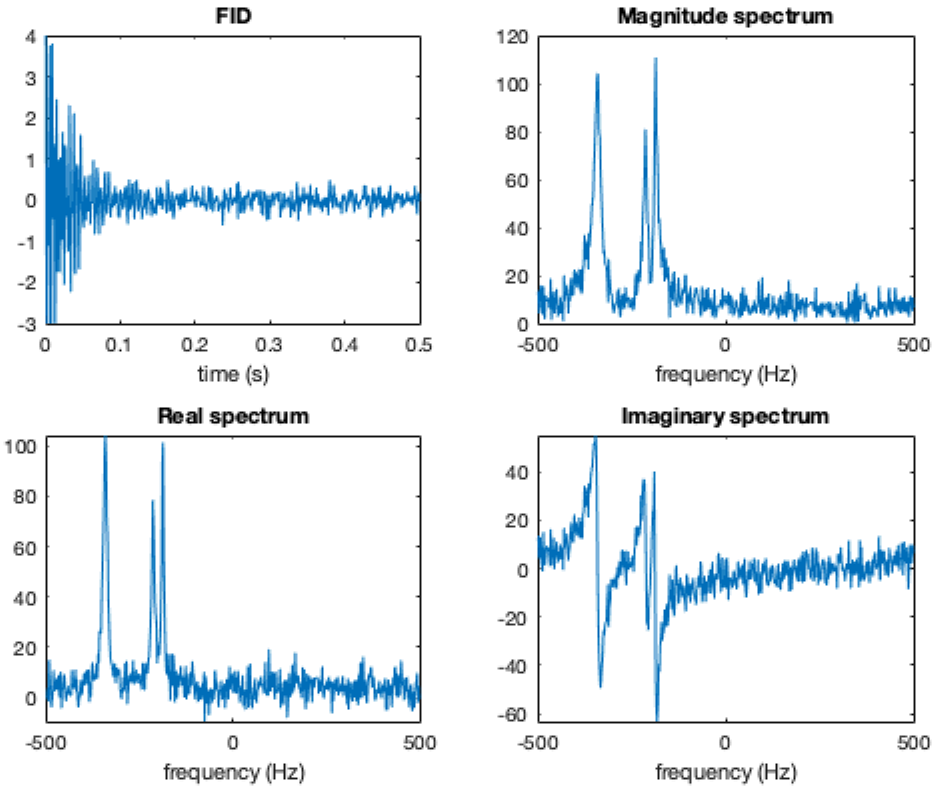
Generate spectra

```
spectrum=fftshift(fft(fid));
w=(-1/(2*dwt)+(1/(dwt*n))):1/(dwt*n):(1/(2*dwt));

%plot result
figure(1);
subplot(2,2,1); plot(t,real(fid)); xlabel('time (s)');
 title('Real{FID}'); axis([0 .5 -3 4]);
subplot(2,2,2); plot(w,abs(spectrum)); xlabel('frequency (Hz)');
 title('Magnitude spectrum');
subplot(2,2,3); plot(w,real(spectrum)); xlabel('frequency (Hz)');
 title('Real spectrum');
subplot(2,2,4); plot(w,imag(spectrum)); xlabel('frequency (Hz)');
 title('Imaginary spectrum');
```



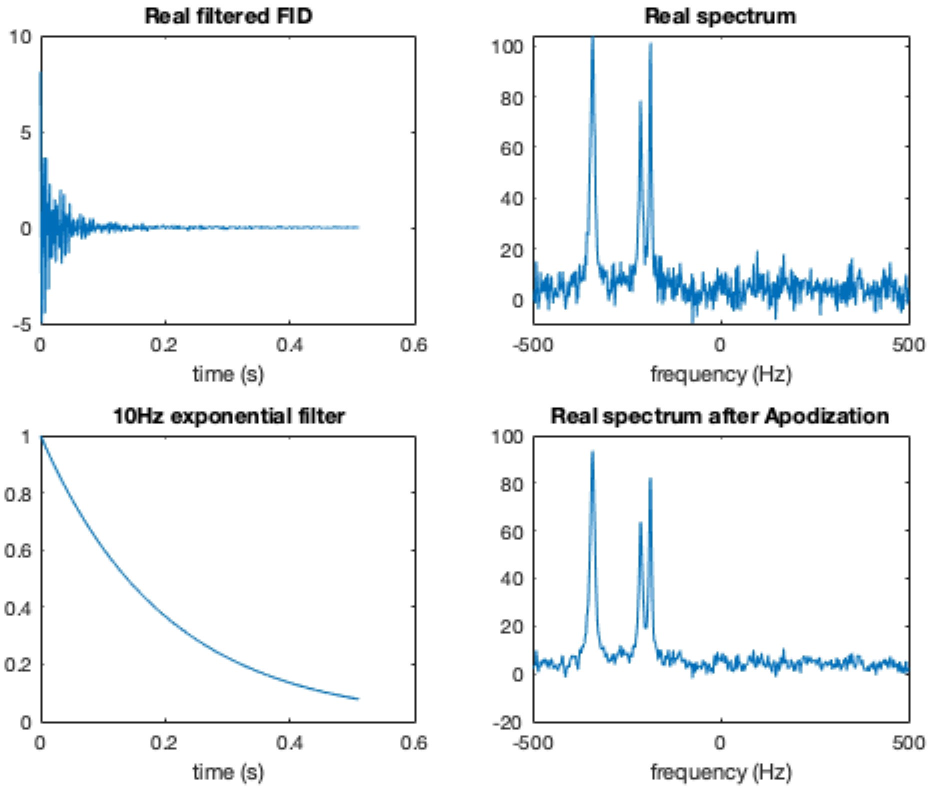
Adding Noise

```
origFid = fid;

%Generate values from a normal distribution with mean 0 and std 0.2.
noise =  2/10 .*randn(1,n) + 2/10 .*randn(1,n) * 1i;

fid=noise + fid;
spectrum = fftshift(fft(fid));


%plot result w/noise
figure(2);
subplot(2,2,1); plot(t,real(fid)); xlabel('time (s)'); title('FID');
 axis([0 .5 -3 4]);
subplot(2,2,2); plot(w,abs(spectrum)); xlabel('frequency (Hz)');
 title('Magnitude spectrum');
subplot(2,2,3); plot(w,real(spectrum)); xlabel('frequency (Hz)');
 title('Real spectrum');
subplot(2,2,4); plot(w,imag(spectrum)); xlabel('frequency (Hz)');
 title('Imaginary spectrum');
```



Apodization with 5Hz decaying exponential filter & SNR

```
filter_freq=-5;
fid_ap=exp(filter_freq*t).*fid;
spectrum_ap=fftshift(fft(fid_ap));


%plot result
figure(3);
subplot(2,2,1); plot(t,real(fid_ap)); xlabel('time (s)'); title('Real
 filtered FID');
subplot(2,2,2); plot(w,real(spectrum)); xlabel('frequency (Hz)');
 title('Real spectrum');
subplot(2,2,3); plot(t,exp(filter_freq*t));xlabel('time (s)');
 title('10Hz exponential filter');
subplot(2,2,4); plot(w,real(spectrum_ap)); xlabel('frequency (Hz)');
 title('Real spectrum after Apodization');
```



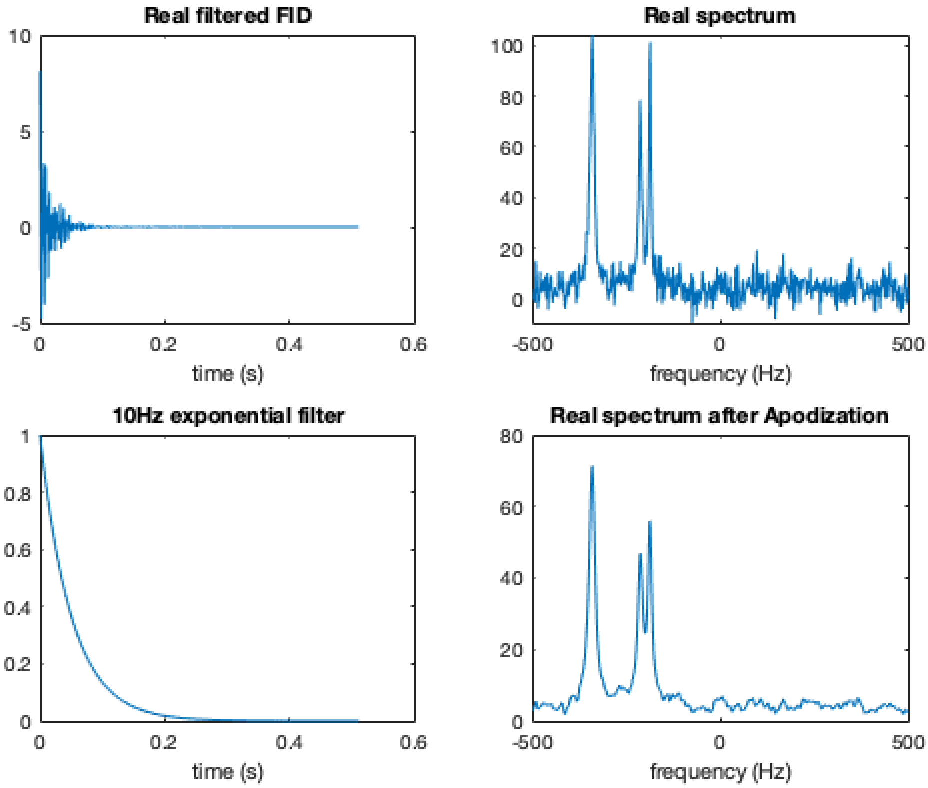
SNR calculation 5 Hz

```
peaks_5Hz=maxk(real(spectrum_ap), 3);
SNR_c = peaks_5Hz(1)/std(real(noise))
SNR_a = peaks_5Hz(2)/std(real(noise))
SNR_b =peaks_5Hz(3)/std(real(noise))
```

```
SNR_c =

  464.1541


SNR_a =

  445.4376


SNR_b =

  420.6824
```

Apodization with 20Hz decaying exponential filter & SNR

```
filter_freq=-20;
fid_ap=exp(filter_freq*t).*fid;
spectrum_ap=fftshift(fft(fid_ap));


%plot result
figure(4);
subplot(2,2,1); plot(t,real(fid_ap)); xlabel('time (s)'); title('Real
 filtered FID');
subplot(2,2,2); plot(w,real(spectrum)); xlabel('frequency (Hz)');
 title('Real spectrum');
subplot(2,2,3); plot(t,exp(filter_freq*t));xlabel('time (s)');
 title('10Hz exponential filter');
subplot(2,2,4); plot(w,real(spectrum_ap)); xlabel('frequency (Hz)');
 title('Real spectrum after Apodization');
```



SNR calculation 20 Hz

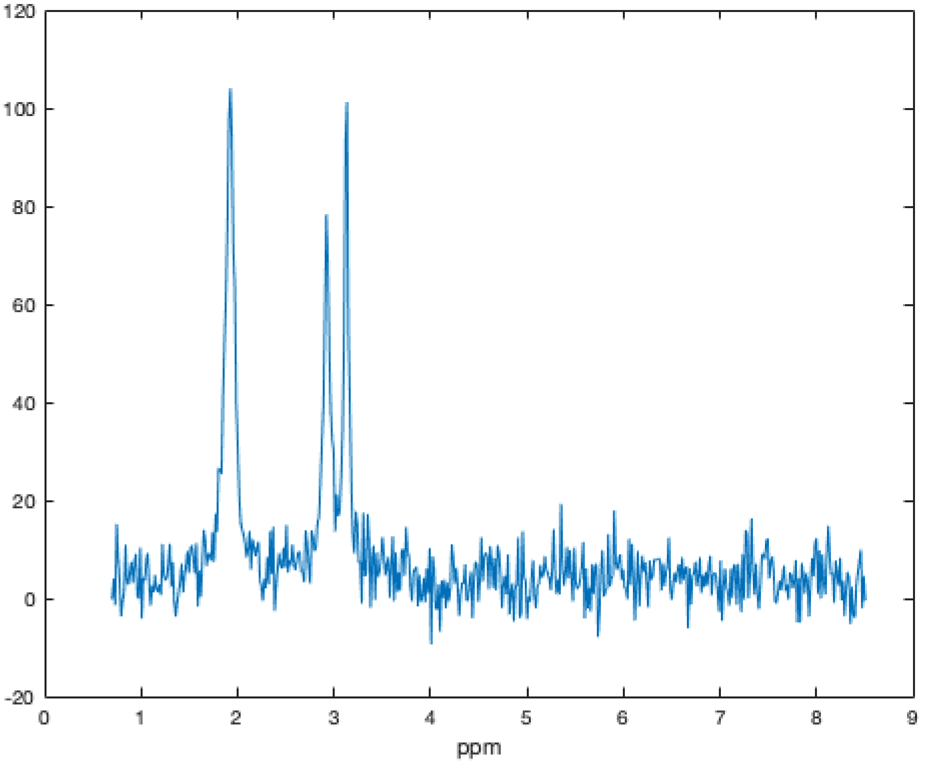
```
peaks_20Hz=maxk(real(spectrum_ap), 3);
SNR_c = peaks_20Hz(1)/std(real(noise))
SNR_a = peaks_20Hz(2)/std(real(noise))
```

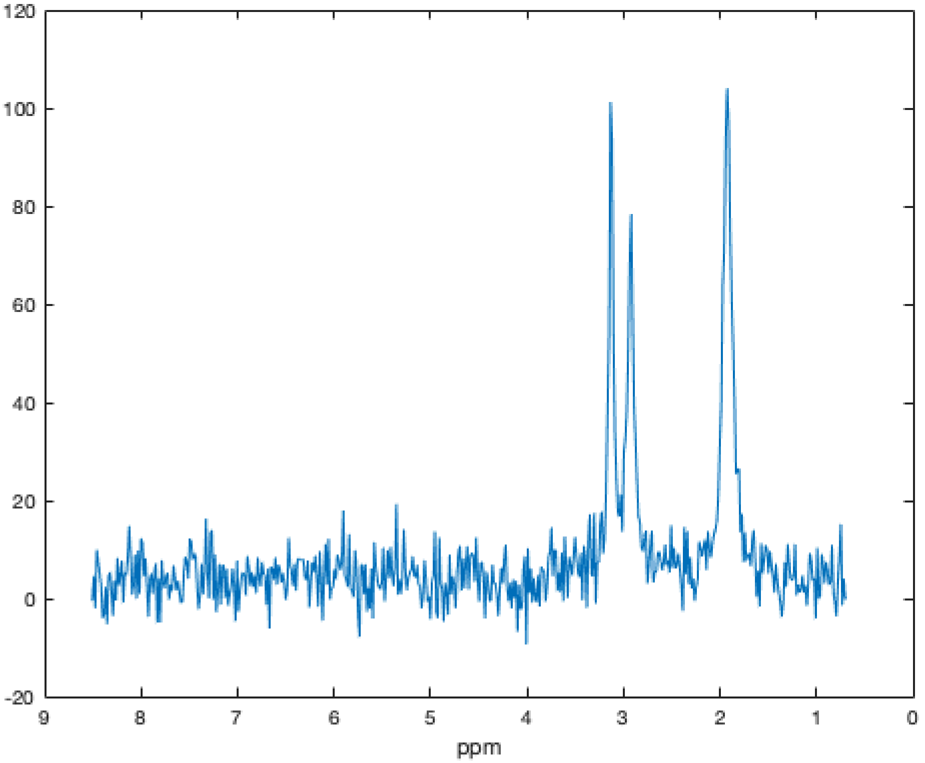
```
SNR_b =peaks_20Hz(3)/std(real(noise))
% By visual assessment, the noise is much lower so the SNR would be
 better
% with 20 Hz apodization compared to 5 Hz.
```

```
SNR_c =

  354.4984


SNR_a =

  345.9459


SNR_b =

  336.5643
```

ppm referencing

```
% ppm = delta freq (in Hz)/operating frequency (in MHz)
% operating frequency = gamma * B0, where gamma = 42.58 for proton
% assume data is acquired with center frequency on water

fc = 127.763229; %assume 3T MHz centfreq;
water_ppm = 4.6;
ppm_ref = water_ppm;
ppm_range = 1000/fc;

% creating ppm reference axis for plotting in ppm standard
freq_res = 0.0153*fc; %frequency resolution in Hz
ppm_res = freq_res/fc; %frequency resolution in ppm
ppm_min =    water_ppm - (ppm_range/2)      ;    %ppm_min - range goes
 from ppm_min to water to
ppm_max =   water_ppm  + (ppm_range/2)      ;   %ppm_max;
ppm_axis = ppm_min:ppm_res:ppm_max;

figure(5); plot(ppm_axis, real(spectrum)); xlabel('ppm');

%inverse the ppm direction
figure(6);
h1 = axes;
plot(ppm_axis, real(spectrum)); xlabel('ppm');
set(h1, 'Xdir', 'reverse')
```


120
100
80
60
40
20
0
-20
0
1
2
3
4
5
6
7
8
9
ppm


120
100
80
60
40
20
0
-20
9
8
7
6
5
4
3
2
1
0
ppm

Published with MATLAB® R2018b