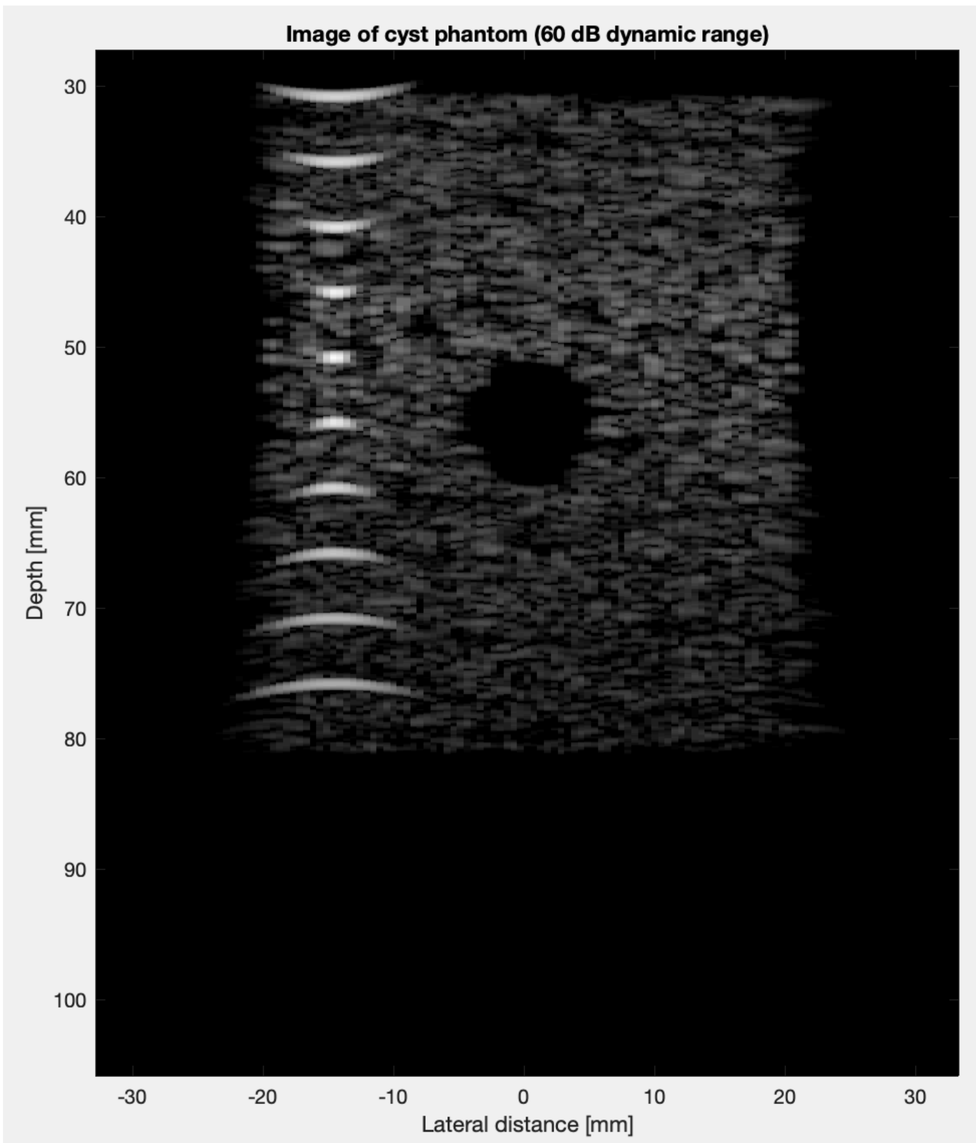
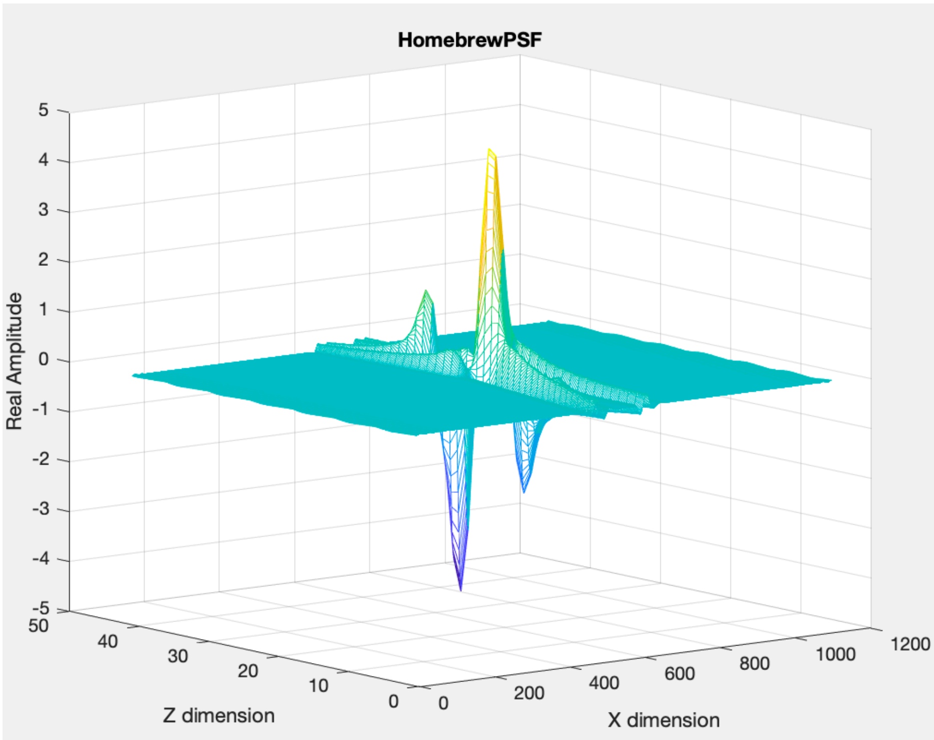
BME 8730
Fall 2019

Jensen's FIELD II speckle

1. Set up a speckle image using FIELDII. Use the transducer parameters from LinearArrayExample62. Use cyst_phantom per the script. (20)



2. Now use our “homemade” code to replicate the simulation of a Point Spread Function (“wave”). (20)

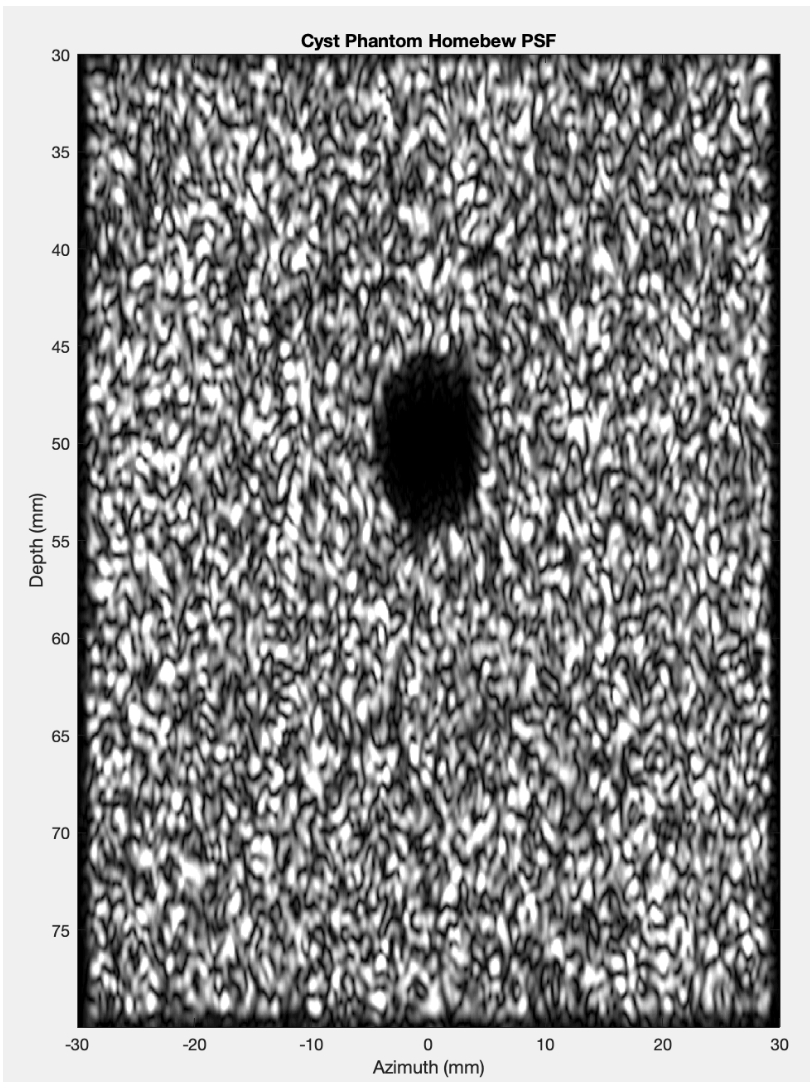



```
%% Parameters
z_foc=50e-3;              %  Range direction focal distance
fc=3e6;                   %  Transducer center frequency [Hz]
c=1540;                   %  Speed of sound [m/s]
lambda=c/fc;              %  Wave length [m]
width=lambda;             %  Width of element
element_height=5/1000;    %  Height of element [m]
num_elems=64;             %  Number of array elements
kerf=width/20;            %  Spacing between elements
txl=kerf+width;           %  Space between signals
fieldxs=(-30e-3:txl/10:30e-3);  % X points in field
fieldzs=(30e-3:txl/10:80e-3);   % Z points in field
fs=fc/128;                % Define a sampling frequency (not critical)
f=(fs:fs:10*fc);          % Define an adequate frequency range (improve resolution in time domain)
w=2*pi*f;                 % Angular frequency radians
ns=length(f);             % Number of samples
bw=80;                    % Fractional bandwidth as percent
sig=bw*fc/100;            % Width of Gausian
tstep=1./max(f);          % Time steps after using Inverst FFT
t=(1:ns).*tstep;          % Define time axis
cystc=[0,50].*1e-3;       % Cyst center point x,z
cystr=5e-3;               % Cyst Radius
%% Deffinition of time delays
elsxpos=(-(txl*num_elems/2):txl:(txl*num_elems/2)); % X positions of transducer
trans_tdels=(sqrt((elsxpos.^2)+(z_foc.^2))./c);      % Time for signal from transducer to reach focus
trans_tdels=trans_tdels-max(trans_tdels);            % Built in transducer focusing delay

tdels=zeros(length(elsxpos),length(fieldxs));        % Delay matrix initialization
% calculation of time delay from each element to each x position at focus depth
for m=1:length(elsxpos)
    for i=1:length(fieldxs)
            tdels(m,i)=sqrt(((fieldxs(i)-elsxpos(m)).^2)+(z_foc.^2))./c;
    end
end
tdels=tdels-trans_tdels'; % apply built in delays to delay matrix
%% Gauss pulse claculation and summation
% Generate Gaussian pulse (frequency domain)
gauss_pulse=exp(-pi*((f-fc)/sig).^2);
% initialize matrix of pulses
gauss_pulses=zeros(length(elsxpos),length(gauss_pulse),length(fieldxs));
% initialize matrix of pulses in time domain
gauss_t=gauss_pulses;
```

```
% Calculate pulses
for i=1:length(fieldxs)
        for m=1:length(elsxpos)
            % Calculate a pulse from each element given time delays
            gauss_pulses(m,:,i)=gauss_pulse.*exp(-j.*w.*tdels(m,i));
            % Inverse forier into time domain
            gauss_t(m,:,i)=ifft(gauss_pulses(m,:,i));
        end
end
% Sum pulses from each transducer element at each position
gauss_t=squeeze(sum(gauss_t));
%% Field deffinition
% Make field of random points
pointvals=rand(length(fieldxs),length(fieldzs))-.5;
% Define cyst shadow
for i=1:length(fieldxs)
    for k=1:length(fieldzs)
        if (sqrt(((fieldzs(k)-cystc(2)).^2)+((fieldxs(i)-cystc(1)).^2)))<=cystr
            pointvals(i,k)=0;
        end
    end
end
%% Convolution of PSF with field
PSF=gauss_t(1031-20:1031+20,:); % restrict PSF to times relivant to Z range
kvs=conv2(pointvals,PSF);       % convolution
kvsr=squeeze(abs(hilbert(real(kvs))));  % Envelope the result of convolution
kvsr=kvsr./max(max(kvsr));              % Normalize
kvsr=permute(kvsr,[2 1]);               % Re-orient
%% speckle SNR calculation
A=kvsr(600:800,800:1100);    % A region of speckle
SNR=mean(A)/std(A)           % Signal to noise calculation
%% Figures
% PSF
figure
mesh(real(PSF))
xlabel('X dimension')
ylabel('Z dimension')
zlabel('Real Amplitude')
title('HomebrewPSF')
% Cyst shadow
figure
image(fieldxs*1000,fieldzs*1000,(127.*kvsr(550:1500,:)./max(max(kvsr))))
colormap(gray)
xlabel('Azimuth (mm)')
ylabel('Depth (mm)')
title('Cyst Phantom Homebew PSF')
```

3. Use convolution of the “homemade” PSF with the phantom and verify that a similar result is obtained. Find the cyst_phantom m-file and replicate it as a 2D regular spaced array (i.e. matching the X-Z resolution of the PSF) and insert the required modifications – i.e. 100X increased amplitude point targets with respect to speckle random scatterer level and a cyst region with no scatterers. Note that you may want to down sample your PSF if it is excessively fine in either dimension. However, please verify that your PSF is not excessively down sampled either by explaining carefully why you believe your PSF is properly sampled or by showing results with, 2X finer sampling in X and Z, and that the result is similar (not identical because it cannot be). Verify that the Std Dev/Mean is 1.9 in a uniform speckle region. You need to be sure you are analyzing the enveloped data but not the logarithmically compressed (video) data. (20)

Done with same code as in 2

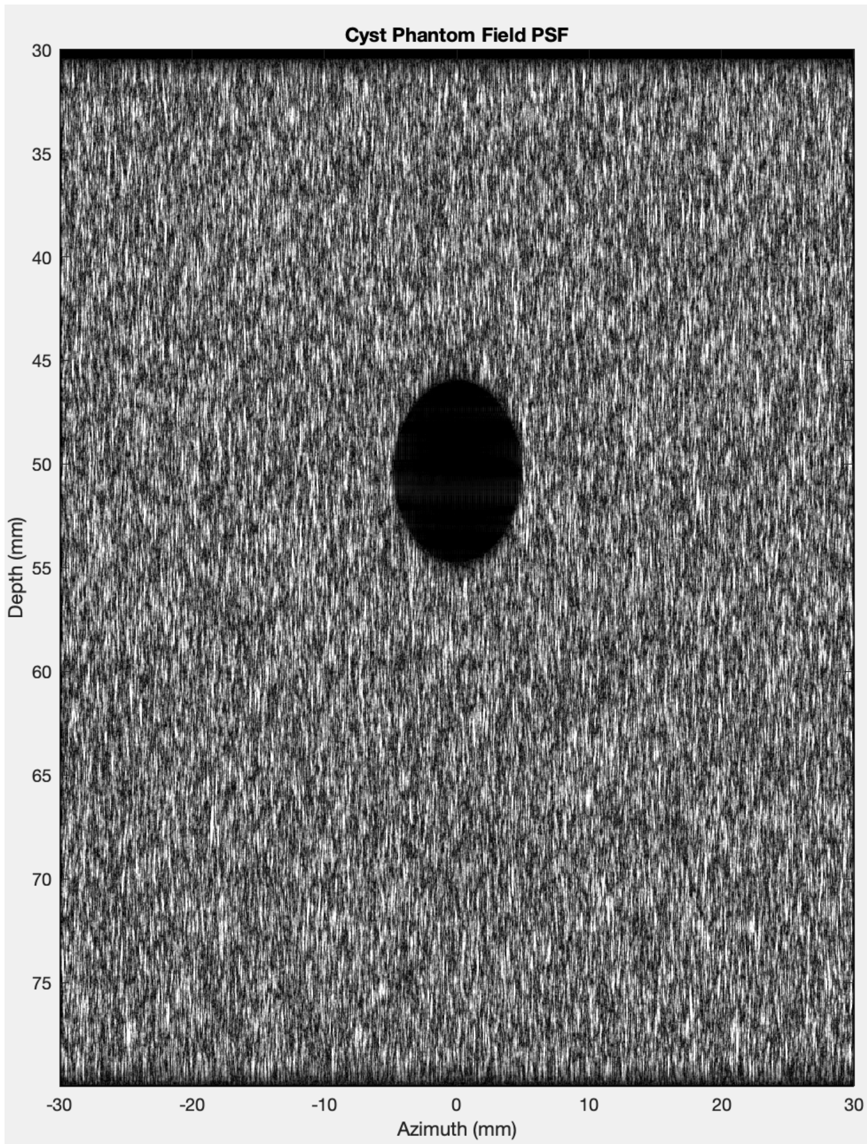



```
SNR =

    1.9372
```

4. An intermediate approach is to take the PSF from FIELD and do a convolution with a speckle generating phantom target. Do this. Again, be sure that your sampling is appropriate – i.e. alter either your FIELD sampling or phantom sampling or both so that they are matched. (20)

Code on next page

```
SNR =

    1.9684
```



```matlab
%% Field call
path(path,'/Users/samuelsklar/Documents/Classwork/Ultrasound/Field_II_ver_3_24_mac');
field_init(-1);
%% Variables
N_elements=64;
z_foc=50e-3;              %  Range direction focal distance
fc=3e6;                   %  Transducer center frequency [Hz]
c=1540;                   %  Speed of sound [m/s]
lambda=c/fc;              %  Wave length [m]
width=lambda;             %  Width of element
element_height=5/1000;    %  Height of element [m]
num_elems=64;             %  Number of array elements
kerf=width/20;            %  Spacing between elements
txl=kerf+width;           %  Space between signals
height=5/1000;            %  hight
fieldxs=(-30e-3:txl/10:30e-3);  % X points in field
fieldzs=(30e-3:txl/10:80e-3);   % Z points in field
z=(50:txl*1000:60).*1e-3;       % Z set for PSF calc
x=(-1:txl*1000:1).*1e-3;        % X set for PSF calc
cystc=[0,50].*1e-3;       % Cyst center point x,z
cystr=5e-3;               % Cyst Radius
focus=[0,0,50e-3];        % Focus point
%% arrange points for PSF calc for field
zpointvals=zeros(length(z)*length(x),1);
xpointvals=zpointvals;
ypointvals=zpointvals;
for i=1:length(x)
    zpointvals((length(z)*(i-1))+1:length(z)*i)=z;
    xpointvals((length(z)*(i-1))+1:length(z)*i)=x(i);
end
points=[xpointvals,ypointvals,xpointvals];
%% Field call and pressure calc
Th = xdc_linear_array (N_elements, width, height, kerf, 1, 1, focus);
[PSF,t]=calc_hp (Th,points);
PSF=PSF./max(max(PSF));
%% Field deffinition
% Make field of random points
pointvals=rand(length(fieldxs),length(fieldzs))-.5;
% Define cyst shadow
for i=1:length(fieldxs)
    for k=1:length(fieldzs)
        if (sqrt(((fieldzs(k)-cystc(2)).^2)+((fieldxs(i)-cystc(1)).^2))<=cystr
            pointvals(i,k)=0;
        end
    end
end
%% Convolution of PSF with field
kvs=conv2(pointvals,PSF);                   % convolution
kvsr=squeeze(abs(hilbert(kvs)));  % Envelope the result of convolution
kvsr=kvsr./max(max(kvsr));                  % Normalize
kvsr=permute(kvsr,[2 1]);                   % Re-orient
%% Signal to noise calculation
A=kvsr(600:800,200:400); % speckle region
SNR=mean(A)/std(A)       % SNR calculation
%% Figures
% PSF
figure
mesh(PSF)
xlabel('X dimension')
ylabel('Z dimension')
zlabel('Amplitude')
title('Field PSF')
% Cyst shadow
figure
image(fieldxs*1000,fieldzs*1000,(127.*kvsr(30:985,1:1120)./max(max(kvsr))))
colormap(gray)
xlabel('Azimuth (mm)')
ylabel('Depth (mm)')
title('Cyst Phantom Field PSF')
```

5. Repeat 1 using dynamic receive focusing. Verify that point target resolution is improved in the near field in the presence of dynamic receive focusing. Look up the field m-file for dynamic receive and use reasonable inputs. (These are derivative from xdc_focus – requiring direction angle instead of focal point) (20)

<u>You may not get exactly 1.9 for your SD/Mean but I do expect it to lie in range 1.8 to 2.0.</u>

Changed line 63 in example 2 from xdc_focus to xdc_dynamic_focus to incorporate the dynamic receiver. The dynamic focus figure is on the right and the image from problem 1 is on the left. The image produced with dynamic focusing has much less blurring of the points in the near field (top), but perhaps more in the far field (bottom).