# Exercise 2: Simulation of ultrasound field using Field II

The purposes of this exercise is to learn how to:

- Set up the simulation environment and model a transducer in Field II
  - Single element transducer
  - Array transducer
- Adjust the accuracy of your simulation
- Calculate the spatial impulse response
- Calculate the pressure field from your transmitting transducer
  - In a single point
  - Along a line
  - In a plane
- Add apodization.
- Calculate the sensitivity of your receiving transducer
- Add dynamic receive focusing
- Calculate the pulse-echo response/sensitivity of your transducer
- Investigate grating lobes
- Add expanding receive aperture
- Investigate the effects of parallel beamforming on the pulse-echo response

Field II can be downloaded from:
http://server.electro.dtu.dk/personal/jaj/field/

## *2.1 Set up the simulation environment and an single element transducer*

1. Create rectangular aperture using xdc_linear_array(). Create one aperture for transmit and one for receive. They should be identical.
   a. Set parameters: no_elements=1, width=18.5mm, height=13mm, kerf=0, no_sub_x=1, no_sub_y=1, focus=[0,0,60].
   b. Plot the aperture using show_xdc_geir(yourEmitAperture, 1)
   c. Set excitation pulse using function xdc_excitation() for both transmit and receive aperture:
      Center frequency: 2.5MHz, pulse length: 1.5 periods.
      Plot excitation pulse.
   d. Set transmit and receive transducer impulse response using xdc_impulse().
      Plot impulse response versus time.
      Plot impulse response in frequency domain using freqz().
      Set fractional bandwidth to 0.6 using the following code:

```
t_ir = -2/f0:1/fs:2/f0;
Bw = 0.6;
impulse_response=gauspuls(t,f0,Bw);
```

2. Define a measurement point using a vector of length 3, containing the x, y and z coordinates of your measurement point. Set this point to x=30mm, y=10mm, z=50mm and calculate the spatial impulse response at this point. Use calc_h().
   Plot the resulting spatial impulse response versus time.
3. Increase no_sub_x and no_sub_y to 30.
   Set sampling frequency to 400 MHz using set_sampling(fs).
   What happens? Considering which part of the aperture that are contributing at which

times, is this a more plausible result? Why? It will be useful to plot your measurement point in the same plot as in 1b.

4. Repeat 2 and 3 for point x=0mm, y=0mm, z=60mm.

## *2.2 Expand to array transducer*

1. Create array transducer using xdc_focused_array(). Hint: Element width equals *pitch-kerf*.
2. Set parameters: no_elements=64, pitch=0.29mm, elementHeight=13mm, kerf=0.020mm, Rfocus=60mm, no_sub_x=5, no_sub_y=15, focus=[0,0,60]. Set sampling frequency to 100 MHz using set_sampling(fs). Plot aperture using show_xdc_geir(yourAperture, 1). It should be approximately 18.5mm in the azimuth-direction (along x-axis).
3. Calculate spatial impulse response at point x=0mm, y=0mm, z=60mm. Use calc_h(). Considering which part of the aperture that are contributing at which times, is this a plausible result?
4. Calculate spatial impulse response at point x=20mm, y=0mm, z=60mm. Use calc_h(). Does this seem as a plausible result? Actually, yes, the zeros in between appear since there are no overlap between the spatial impulseresponses from each individual element. This can be seen by setting the number of elements to 1 and measure the length of the spatial impulse response.
5. Calculate spatial impulse response at point x=20mm, y=10mm, z=60mm. Use calc_h(). Compare to 4. This time, the spatial impulse response from each individual element is much longer and overlaps with the responses from neighbouring elements. This results in a more even total spatial impulse response.

## *2.3 Measure spatial impulse response and pressure along a line*

1. Use same transducer as in Exercise 2.2, however, this time increase the number of measurement points to 101. Create a measurement_points matrix where each row contains the x, y and z coordinate of a measurement point. Distribute measurement points on a line extending from x=-20mm to x=20mm.
2. Calculate the spatial impulse response on this line at depths 20mm, 60 mm and 90 mm.
   Plot the spatial impulse responses versus time along y-axis and azimuth position (x) along x-axis.
3. Calculate the pressure in a line from x=-20mm to x=20mm at depths 20mm, 60 mm and 90mm.
   Plot the pressure versus time along y-axis and azimuth position (x) along x-axis.
   To the sides of the transmit direction the pulse is splitting in two. This is referred to as edge waves. What is the origin of these?
4. Use the data from 2.3.3 to calculate the beam profile at depths 20mm, 60mm and 90mm. Beamprofiles are calculated as the RMS (Root Mean Square) value of the signal in time.
   Normalize to maximum value and plot beamprofile vs azimuth position.
5. From the Fresnel and Fraunhofer –approximations it can be shown that the pressure field at focal range is proportional to the Fourier transform of the aperture. For a rectangular aperture of size *aTx* and focus depth *R*, frequency *f0* and speed of sound *c*, the Fourier transform becomes sinc(x*aTx*f0/(R*c)). Plot the absolute value of this sinc in the same plot as the beamprofile for 60mm depth.
   How does it match the simulated beam profile? Can you explain the differences?

## *2.4 Calculate pressure along a radial line*

1. Place 100 measurement points on a line starting in 5 mm depth extending to 150mm depth.
2. Set focalDepth to 60mm.  
   Calculate the transmit pressure field.  
   Normalize to maximum value.  
   Plot versus depth.  
   At which range does the maximum value occur? Does it correspond with the focal range? Why, why not?
3. An equation for the depth of focus, the region around the focal depth where the amplitude is above half maximum is 7.2 * lambda * (focalDepth/apertureSize)^2. Lambda is the wave length of the center frequency and equals c/f0. How does this formula match the simulation? Hint: Use the data cursor tool in the figure window to measure.
4. Change focus to 30mm range (Keep Rfocus at 60 mm). Repeat exercise 2.4.2. Is the maximum at the same location compared to the focal range as in 2.4.2? Why, why not?
5. Increase both Rfocus and focus gradually to move maximum further out. What is the farthest maximum you can achieve? How does this match the equation for diffraction focus Rdf = aTx^2/(4*lambda)?
6. Reduce number of elements to 32. How does this affect the range maximum? Why?

## *2.5 Calculate pressure field in the whole XZ-plane*

1. Create a 81x59 (Nx, Nz) grid of measurement points covering the XZ-plane from x=-15mm to x=+15mm and z=5mm to z=150mm.  
   Hint: use [X,Z] = meshgrid(x,z), where x and z are vectors.  
   Further use measurement_points = [X(:), zeros(length(X(:),1),Z(:)].  
   (plot points together with aperture to ensure that they are in the correct position.)
2. Calculate pressure in the XZ-plane using the following procedure:
   a. Calculate the transmit pressure in all measurement points. If memory problems, reduce fs to 50MHz or reduce number of measurement points.  
   b. Calculate RMS value of pressure at each point. This will result in a vector with length equal to the number of measurement points.  
   c. Normalize to maximum value.  
   d. Reshape to a matrix of dimensions Nz times Nx. Use *BPmatrix = reshape(RmsValues, Nz, Nx)*.  
   e. Display resulting pressure with azimuth along x-axis and range along y-axis. Use log compression when displaying pressure.  
   Use *pcolor(xAxis, zAxis, 20*log10(BPmatrix))* and *shading interp*.  
   Use *colorbar* to display what pressure the color coding represents.  
   Use *caxis([-30 0])* to set the dynamic range to 30 dB.  
   f. The plot in 2.5.2.e shows the pressure field normalized to global maximum. Sometimes it is more desirable to investigate the evolution of the beam width vs depth. To see this, the pressure should be normalized for each depth range. Create an additional plot where the pressure field is normalized at each depth. To do this use *BPmatrix=BPmatrix./repmat(max(BPmatrix')', 1,Nx);*  
   Use *caxis([-30 0])* to set the dynamic range to 30 dB.

## *2.6 Adding apodization*

1. Continue on the script from 2.5. But add a section after setting the excitation and impulse response where you define your transmit apodization. Create a vector, *txApodWeights*, containing the weights of all your transducer elements. Use Hanning weighting (*hanning()* in Matlab). The Field II apodization is set using the function *xdc_apodization(yourEmitAperture, 0, txApodWeights).*
2. Calculate transmit field both with global and depth normalization.
   Use *caxis([-30 0])* to set the dynamic range to 20 dB.
   Describe the difference. What has happened to beam width, side lobe level and depth of focus? Is it all good? What happens to the SNR?
3. Repeat 2.6.2, but this time use *tukeywin(nTransmitElements, 0.3)* to create a cosine-tapered window. Describe the difference from using Hanning- and rectangular-apodization. What about the SNR?
   This might be a good middle way if you want to slightly improve contrast resolution (i.e. reduce side lobes) without sacrificing too much resolution and SNR/penetration.
4. If available, start Window Design & Analysis Tool in Matlab. Since there is a Fourier relation between your aperture function and your field at focal range, this can be used to evaluate different apodization functions regarding beam width vs sidelobe level.

## *2.7 Receive sensitivity and dynamic focusing*

1. Up till now we have concentrated on the transmitted pressure field. The receive "field" can be calculated in the same way, only that the plots then show sensitivity instead of pressure. I.e. it shows from which spatial positions the receiver array is sensitive.
   a. Continuing on the script from 2.6, add a receive aperture in exactly the same way as the transmit aperture by using *xdc_focused_array()*. Same parameters.
   b. Set the impulse response to the same as the transmit aperture using *xdc_impulse()*. Notice that no exitation is set for this aperture. This off course is because it is only receiving.
2. Calculate the receive sensitivity in the XZ-plane using the same parameters as in 2.5. Due to reciprocity, the receive sensitivity pattern will be quite similar to the transmit pressure field. The only difference, is that the receive sensitivity is only a convolution between receive spatial impulse response and transducer impulse response, while the transmit has an additional convolution with the excitation pulse.
3. The advantage on the receiving side is that one can do dynamic focusing. I.e. the receive focus can be set to follow the transmitted pulse as it propagates away from the transducer.
   Add dynamic receive focusing. This is done by inserting the following code line before calculating the sensitivity: *xdc_dynamic_focus(yourReceiveAperture, 0, 0, 0).*
   Where the last three parameters are:
   Time from which dynamic focus should be switched on,
   receive beam angle in XZ-plane,
   receive beam angle in YZ-plane.
   What is the effect of dynamic focusing on resolution, contrast and depth of focus?
4. Add receive apodization in the same way as in 2.6.1. Try rectangular-, Hanning-, and Tukeywin(…,0.3). What happens to the depth of focus compared to using static focusing with and without apodization?

## 2.8. Pulse-echo response

1. Transmitting a sound pulse illuminates a certain area/volume with ultrasound. The receive array has a certain sensitivity pattern, telling us the spatial positions it is sensitive to ultrasound. The transmit field and receive sensitivity combined gives us the pulse-echo responsen (also called two-way response). This gives us the "true" sensitivity map of our beam, where the echos most likely are originating from. Mathematically it is given by the convolution between five transfer functions: excitation, transmit transducer impulse response, transmit spatial impulse response, receive spatial impulse response, receive transducer impulse response.

   Calculate the pulse-echo response using the function *calc_hhp(yourTransmitAperture, yourReceiveAperture, yourMeasurementPoints)*. What is the approximate level of the first side lobe around focal depth? Compare with side lobe level for transmit or receive only. How does it compare with the first side lobe of a sinc^2? What aperture function/apodization will give a sinc^2 at focal range?
   Answer: A triangular function. I.e. the pulse-echo aperture function will be a triangular function, or more correctly, the convolution between the transmit and receive aperture. I.e. the pulse-echo aperture function will have a width corresponding to the sum of the transmit and receive aperture.

   Further, this will give us an estimate of the -6dB pulse-echo beam width to be (-6dB power that is, alt. -3dB amplitude) *focalDepth/(transmitApertureSize+receiveApertureSize).* Compare this formula to the simulations. Decrease the lateral extent of your measurement point to x=-5mm to x=5mm and set the dynamic range to 6dB to more accurately evaluate the 6dB beamwidth vs depth. I.e. *caxis([-6 0]).* Most likely, you'll find that the formula is a bit optimistic (approx 10-15% to narrow).

   Since the dynamic range of your pulse echo response is doubled, you should increase the dynamic range to 50 dB ( using *caxis([-50 0])* ) when examining the pulse-echo response in general.

## 2.9 Grating lobes

1. Using the script from 2.8., set the steering angle of your transmit and receive beam to -35 degrees. The focus can either be changed by changing the initial *focus* parameter, or it can be set to a new value using for example *xdc_focus(yourAperture, 0, newFocusPoint).*
   a. Add a parameter in the setup-section for transmit and receive angle and use the latter approach to set the transmit focus before calculating the responses. Your new transmit focus will be *focalRange*[sin(steeringAngle*pi/180), 0, cos(steeringAngle *pi/180)];*
   b. On receive, using dynamic receive focusing, use *xdc_dynamic_focus(yourReceiveAperture, 0, receiveAngle, 0);* This will result in dynamic focusing along the specified angle in the XZ-plane.
   c. Redefine your measurement points to a 161x30 (Nx, Nz) grid of measurement points covering the XZ-plane from x=-30mm to x=+30mm and z=5mm to z=80mm
2. Calculate both transmit, receive and pulse-echo field. Set *axis equal tight* to get proper geometrical relations. Consider the globally normalized plot only. The depth normalized plot will not be suitable.

3. Increase the pitch (distance between the transducer elements) to lambda. Decrease the number of elements to 32 (This will result in an approximately equal transducer size).
    a. Repeat exercise 2.9.2. What happens?
    b. Compare the grating lobe direction to the theoretical: *gratingLobeAngle=asin(lambda/pitch+sin(steeringAngle));*
    c. Evaluating the two way response, what will the influence of this grating lobe be on the image quality?

Afternoon exercises:
If you are experiencing memory errors in Matlab, either reduce the number of measurement points or reduce the sampling frequency (minimum to 50MHz, do not reduce further).

## *2.10 Expanding aperture*

The lateral resolution in the image is given by *Fnumber*lambda*, where *Fnumber* is *focalDepth/apertureSize*. Using dynamic focusing on receive, focalDepth is changed constantly as the transmit pulse propagates. This results in a constantly changing resolution as well. Expanding aperture is a technique where the aperture is changed to keep the *Fnumber* constant, hence the lateral resolution constant. This means using a small aperture when focusing in the near field, and increasing the aperture until reaching its physical size. This technique is implemented using apodization.

Transducer parameters: no_elements=64, pitch=0.29mm, elementHeight=13mm, kerf=0.020mm, Rfocus=60mm, no_sub_x=5, no_sub_y=15, focus=[0,0,60].

Measurement points: 161x30 (Nx, Nz) grid of measurement points covering the XZ-plane from x=-10mm to x=+10mm and z=5mm to z=80mm.

Your goal is to keep a receive **F-number of 2.5** as long as possible (limited by aperture physical size)

1. Define a time vector of length *Nfocalzones* and an apodization matrix of size *Nfocalzones x Nelements*, where each row corresponds to a time instance in the time vector after which the apodization is valid. This is explained on page 16 in the Field II user manual. The time vector is typically calculated by starting out with a focal depths vector defining the ranges for each apodization function. Use 300 focal depths from range 0 to 20 cm. The aperture size is reduced by adding zeros at each side of the apodization vectors. Use rectangular/flat weighting of the active elements at each depth.
2. Calculate the receive sensitivity in the XZ-plane. Set dynamic range to 35 dB using *caxis([-35 0])*. At which range does the beam width remain relatively constant? Compare to the case without expanding receive aperture.
3. Can you figure out a smarter way of apodizing your receive aperture that would have the identical beam width as in 2, but better suppression of the sidelobes (especially in the near field)? You are not required to implement this, but please explain the concept if you can figure it out.
4. Calculate the pulse-echo response in the XZ-plane. Compare with fixed receive aperture width.

## 2.11 Multi Line Acquisition

The frame rate of ultrasound is limited by the speed of sound. Your transmitted pulse must propagate to your maximum imaging depth and back (echoes) before the next transmit pulse can be transmitted. Multi Line Acquisition, MLA, is a technique for improving the frame rate by beamforming several receive directions for each transmitted pulse. To do this, the receive beam must be steered off the centre axis of the transmit beam as illustrated in the figure below.

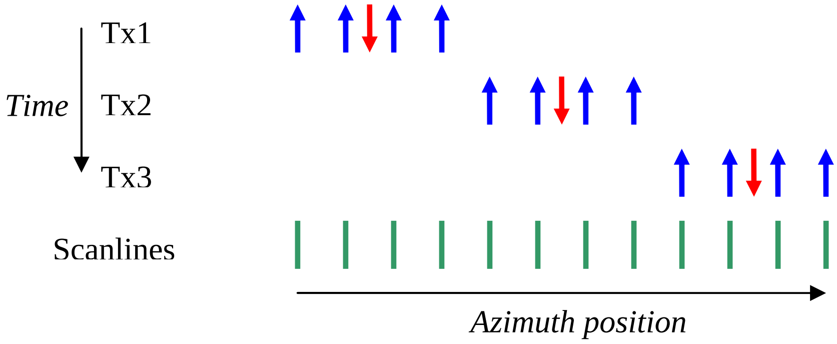

***Figure 1**. Illustration of 4MLA. That is four parallel receive beams for each transmit beam. The up-arrows indicate receive beam position, the down-arrows transmit direction. The positions of the resulting scan lines are indicated in the bottom of the figure.*

Transducer parameters: no_elements=64, pitch=0.29mm, elementHeight=13mm, kerf=0.020mm, Rfocus=60mm, no_sub_x=5, no_sub_y=15, focus=[0,0,60].

Measurement points: 161x30 (Nx, Nz) grid of measurement points covering the XZ-plane from x=-10mm to x=+10mm and z=5mm to z=80mm.

You are going to simulate the pulse-echo response of the 4MLA case illustrated in Figure 1. That is identical transmit direction and four different receive directions. Assuming that your transmit beams are spaced according to Nyquist: *dThetaTx=lambda/aTx*, the offset of the outermost receive beam compared to transmit direction (in radians) should be: *thetaReceive1=-dThetaTx*3/8,*

1. First, simulate the reference cases where the transmit and receive direction are identical (*-dThetaTx*3/8)*. Simulate transmit pressure field, receive sensitivity and the pulse-echo response in the XZ-plane. NB! Do not use *axis equal* when plotting the field. Use *axis normal*. That will make it easier to see the effects in the next part.
2. Set transmit angle to 0 degrees, but keep the receive angle of *-dThetaTx*3/8*. Simulate transmit pressure field, receive sensitivity and the pulse-echo response in the XZ-plane. Compare to the reference case in a. What is the difference and how does it evolve with depth? Does the lateral maximum of the pulse-echo response at focal range coincide with the maximum in the reference case? By looking at the transmit and receive fields, can you explain why this happens?
3. Repeat 2 for transmit angle 0 and receive angle *+dThetaTx*3/8*. That is the outermost receive direction at the other side of the transmit beam. What is the difference between the pulse-echo response of the two outermost receive directions?