# Exercise 3: Array Design

Matthew Chandler

## Abstract

An ultrasonic phased array was designed using MATLAB v9.9.0 for the purpose of inspecting a simulated water tank containing a series of defects. In achieving this design, the beam profile from a linear array was predicted and optimised. The full matrix of time-domain ultrasonic signals was then simulated for the phased array inspecting a sample with known defect locations. These time-traces were then focussed using the total focussing method (TFM), to inspect the defects in the sample.

## Introduction and background

Ultrasonic phased arrays have several advantages over monolithic transducers when inspecting a component for defects, including flexibility (capable of performing many functions), reliability (electronic scanning over a region of interest as opposed to mechanical scanning) and range of application: many functions commonly performed with an array are not possible to achieve with a single transducer, e.g. setting a variable focal length during or post scan. As a result, they are a staple of ultrasonic non-destructive testing (NDT). Of particular interest to the work described in this report is the modelling of ultrasonic phased arrays in a simulated environment rather than the physical implementation and experimentation: models provide evidence that the physics of the system to be inspected is well understood, and provides a much cheaper and more accessible means for evaluating the performance of an array than physical prototyping.

In 2D, the acoustic pressure field at coordinates due to a transducer can be found by dividing the source into elements and adding up the individual contributions [1]. It is given by equation

where is the distance from the th element in the probe to the coordinates defined as at time , and are the wavenumber and frequency of the wave respectively and is a complex number expressing the amplitude and phase of the source. This model assumes that the length of the transducer in the out-of-plane direction is much longer than the length in the plane. A phased array can be modelled using the same equation, modelling each individual transducer within the array as an element.

Common methods of inspection with arrays include plane, focussed and sector scans [2], which delays the signal from each element such that the shape of the ultrasonic beam generated is changed. If the complete set of time-domain signals is collected independently from every pair of transmitter-receivers in the array (the collection of which is called full-matrix capture, FMC), any of these inspection methods can be used in post-processing to produce an image, , given by

where is the time-trace recorded by the th element in the array on the th transmitting element (i.e. the element in the FMC data), is the amplitude applied to the time-trace obtained by the th receiving element from the signal broadcast by the th transmitting element, and is the delay applied to the th transmitting or th receiving element. Both are dependent on the method of inspection.

The total focussing method (TFM) is an imaging method applied in post-processing after collection of FMC data, where the beam is focussed at every point in the image. This is achieved by finding the time taken for the wave to travel from the th transmitter to the point in the image and then to the th receiver,

where is the speed of the wave in the medium. From equation 2, this value is used to lookup the amplitude of the signal in the th time-trace. The resulting value in the image at point is then just the sum of these amplitudes for all elements in the FMC data.

It is of note that as TFM occurs entirely in post-processing, it is not required that the data is obtained experimentally: the FMC data can be obtained from a simulation. The work described here uses a ray-tracing method [3] to obtain the path of the wave from transducer to scatterer, and simulated the FMC data using a linear system approach [4]. As the waves of interest pass through one medium only, the ray is a simple line-segment drawn from the transducer element to the scatterer.

## Linear beam profile

The inspection target to be studied in this report is a calibration sample for an ultrasonic phased array designed to operate into a human body. The sample consists of a series of point targets suspended in a water tank. Using this sample, the properties of the phased array are to be designed such that the point targets are well resolved.

Initially, the beam profile from a linear array into water was predicted using a 2D MATLAB program. This was done using equation 1, where the amplitude was defined as for the th array element. In this definition, is the time taken for the wave to travel from the th element to the focal point, is the angle that the ray makes with the probe element normal, and is the directivity function of the array element, defined by equation

where is the array element width, with pitch , where is the separation between adjacent elements. The resulting profiles for a range of transducer frequencies and element pitch values are shown in fig 1.

The resulting beam profiles indicate that over the range examined, the focus matches with the focal point fairly well, particularly for figs 1(c, e-i). In particular, it can be inferred that the transducer centre frequency influences the lateral width of the focus, and the pitch influences the length of the focus when number of elements and element separation are constant. Additionally, the effect of the total number of elements used was studied as centre frequency was varied: the resulting beam profiles are shown in fig 2. This was done by fixing the total array length as constant , and varying both the pitch and number of elements in the array. In these plots, it appears as though it is the overall length of the array that influences the shape of the focus, with the frequency controlling the size, rather than the pitch which controls the length of the focus.

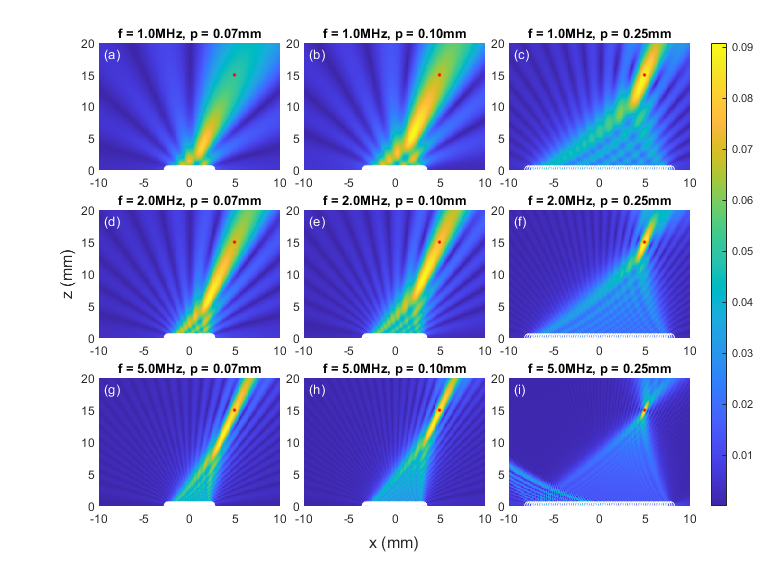


Fig. 1 (a-i) Beam profiles for transducer frequencies and pitch values . Array element locations are shown as white circles, and the beam focal point is shown as a red circle. Number of elements in all cases is ; element separation is . Note that a colourbar has not been included as the absolute signal is not important, and individual subfigures are plotted on independent colour limits.

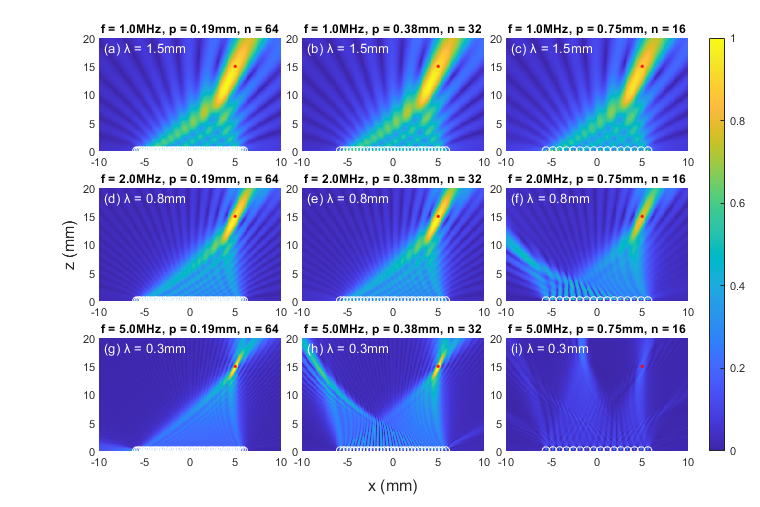
****

Fig. 2 (a-i) Beam profiles for transducer frequencies , pitch values and . Array element locations are shown as white circles, and the beam focal point is shown as a red circle. Element separation in all cases is . Note that the total length of the array in all cases is . Note that each subfigure is plotted independently of each other.

In figs 2(f-i), there are grating lobes present away from the focal point. In the literature [2, 5], it is discussed that grating lobes are suppressed when the spacing between array elements (i.e. the pitch) is less than half the wavelength of the probe. The beam profiles shown in fig 2 display a good agreement with this condition, as it is satisfied in all cases where grating lobes are not present. In particular, fig 2g shows a very small grating lobe for and small , and it can be seen that the pitch is only slightly larger than half of the wavelength of the wave. Fig 2i shows the most significant grating lobes, where , the most significant deviation from the condition.

To resolve the defects in the calibration sample to a high resolution, figs 1-2 suggested that a high frequency and number of elements should be used, with pitch set to half the wavelength. The specification of the array

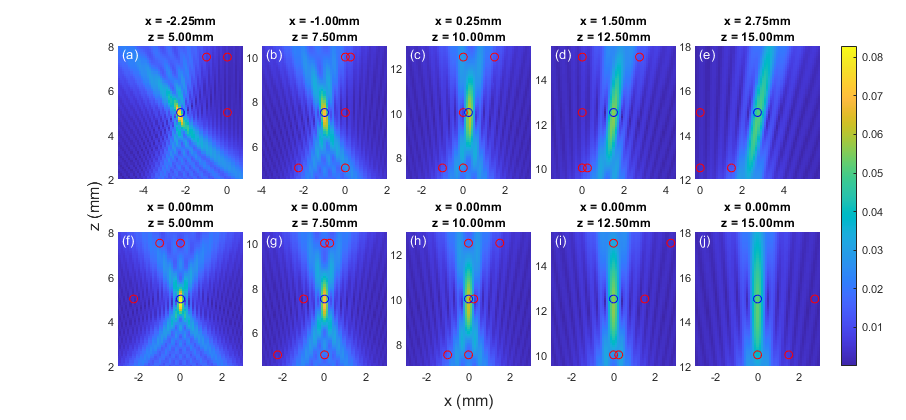


Fig. 3 Focal points set at each of the defect locations in the calibration sample, with transducer frequency , pitch , total elements and element separation . Note that the full region of interest has not been plotted, instead the focus has been magnified to inspect how well each point is resolved. The subfigures are each plotted on a common colourbar scale.

required no more than elements, with the gap between elements being at least . The smallest separation between adjacent scatterers in the medium is in the -direction, and in the - direction. Taking the largest frequency examined , in order to achieve as fine a resolution as possible, the beam profiles with focal points set to the scatterer locations and these array specifications are plotted in fig 3. It can be seen in this figure that the points which are the least likely to be resolved individually are the points in fig 3c and 3h, and the points in fig 3i and 3j. This is because the focus is much longer in the -direction than in the -direction, and becomes longer as the focal point moves away from the array in the -direction. In subsequent analysis, these points will be studied when examining the array’s performance.

## Time-trace simulation and Imaging algorithms

## References

1. Wilcox, P., “Ultrasonic Arrays I,” *Ultrasonic NDT*, 31 Mar 2021, University of Bristol. Lecture.
2. Holmes, C., Drinkwater, B. W. and Wilcox, P., “Post-processing of the full matrix of ultrasonic transmit-receive array data for non-destructive evaluation,” *NDT&E Intl.*, **38**, pp. 701-711, 2005.
3. Budyn, N. S., “Imaging and defect characterisation using multi-view ultrasonic data in nondestructive evaluation,” EngD thesis, University of Bristol, pp. 23-40, 2020 .
4. Velichko, A., “The Frequency Domain,” *Ultrasonic NDT*, 29 Mar 2021, University of Bristol. Lecture.
5. Pompei, F. and Wooh, S-C., “Phased array element shapes for suppressing grating lobes,” *J. Acoust. Soc. Am.* **111**(5), pp.2040-2048, 2002.

## Appendix A: Matlab code (Linear beam profile)

clear; %clear all variables from memory

close all; %close all windows

clc; %clear command window

%% Plotting parameters

n = 3;

m = 3;

t = tiledlayout(m,n);

subfig\_label = 'abcdefghijklmnopqrstuvwxyz';

kk = 1;

%% Inputs

centre\_frequency = [1e6, 2e6, 5e6];

el\_pitch = [.1875e-3, .375e-3, .75e-3];

el\_separation = .05e-3;

num\_els = [64, 32, 16];

velocity\_L = 1500;

backwall\_distance = 20e-3;

grid\_pts = 201;

focal\_pt\_x = 5e-3;

focal\_pt\_z = 15e-3;

for ii = 1:m

for jj = 1:n

%% Parameters

wavelength = velocity\_L / centre\_frequency(ii);

k = 2 \* pi / wavelength;

omega = 2 \* pi \* centre\_frequency(ii);

el\_width = el\_pitch(jj) - el\_separation;

% Generate grid.

x = linspace(-backwall\_distance/2, backwall\_distance/2, grid\_pts);

z = linspace(0, backwall\_distance, grid\_pts);

dx = x(2) - x(1);

dz = z(2) - z(1);

[X, Z] = meshgrid(x, z);

% Get focal\_pt indices

focal\_pt\_ii = round(focal\_pt\_x / dx) + round(grid\_pts / 2);

focal\_pt\_kk = round(focal\_pt\_z / dz);

% Mid-coordinates of each element.

source\_x\_positions = linspace(0, el\_pitch(jj)\*(num\_els(jj)-1), num\_els(jj));

source\_x\_positions = source\_x\_positions - mean(source\_x\_positions);

% Adjust X, Z, source\_x\_positions for vectorisation.

X = repmat(X, 1, 1, length(source\_x\_positions));

Z = repmat(Z, 1, 1, length(source\_x\_positions));

source\_positions = reshape(source\_x\_positions, 1, 1, length(source\_x\_positions));

%% Beam Profile of linear array

% Calculate distance from source to point.

r\_j = sqrt((X - source\_positions).^2 + (Z - .01e-3).^2);

% Calculate angles made from element to pixel location.

phi = acos(Z ./ r\_j);

directivity\_f = el\_width \* sinc(k \* el\_width / (2 \* pi) \* sin(phi));

% Array time delays. Get the list of distances to this point for all els.

d\_j = squeeze(r\_j(focal\_pt\_kk, focal\_pt\_ii, :));

t\_j = (d\_j - d\_j(round(num\_els(jj)/2))) / velocity\_L;

% Weight-and-phase delay.

B\_j = repmat(reshape((exp(- 1i \* omega \* t\_j)), 1, 1, num\_els(jj)), length(z), length(x), 1);

% Calculate pressure field. Sum over elements.

p = sum( ...

1./sqrt(r\_j) .\* exp(1i\*(k \* r\_j - omega \* 0)) .\* directivity\_f .\* B\_j, ...

3 ...

);

nexttile;

hold on

box on

imagesc(x\*10^3, z\*10^3, abs(p))

scatter(source\_x\_positions\*10^3, zeros(num\_els(jj),1), 'wo')

scatter(focal\_pt\_x\*10^3, focal\_pt\_z\*10^3, 'r.')

text(-9.5, 18.5, sprintf('(%s) λ = %.1fmm', subfig\_label(kk), wavelength\*10^3), 'Color', 'white')

kk = kk+1;

title(sprintf('f = %2.1fMHz, p = %3.2fmm, n = %d', centre\_frequency(ii)\*10^-6, el\_pitch(jj)\*10^3, num\_els(jj)))

end

end

t.XLabel.String = 'x (mm)';

t.YLabel.String = 'z (mm)';

t.TileSpacing = 'tight';

cb = colorbar;

cb.Layout.Tile = 'east';

## Appendix B: Matlab code ()