Summer Project: Megan Evans

Abstract
Over the summer of 2021, I worked with Mark and James Evans at Traxion-NDT to create MATLAB programs for pre and post-processing of finite element model data for a specialized FEM solver called POGO. This solver is specifically designed for wave propagation calculations utilizing multiple graphics cards to process the large number of calculations which are necessary. The main output from the post-processing is an animation of the wave propagating in the medium which will be used as a training tool by a colleague in the non-destructive testing industry.

The initial project brief was to produce a suite of programs in Matlab for this purpose. Most of these objectives were met but some of the more advanced post-processing functions proved to be more complex than originally thought.

Introduction
POGO is a finite element method (FEM) package that calculates the propagation of disturbances in solid materials. As well as this, it supports wave sources and detectors. The reason this is useful as one of the current ways of finding defects in materials is sending a wave packet through them via a piezoelectric transducer – i.e. something that converts electrical energy into work – and measuring the incoming signal from either the same transducer or a transducer at a different location. This signal can tell you whether the pulse has reflected off an area where the density changes. This has many commercial uses, one of which is finding cracking around bolt holes in rail without having to dismantle them.

The main difference between POGO and other FEM packages is primarily speed, since it uses the more powerful graphical processing unit rather than the central processing unit. However, a great deal of preparatory work needs to be done to create input files and set parameters before the power of POGO can be realised.

Methods
FEM is a powerful technique for numerically solving partial differential equations (PDEs). Before it can work, structures must first be divided into a mesh of lots of smaller elements with nodes at the corners. Then, a stiffness matrix is formed depending on the connectivity of the nodes. This is then inverted numerically to get the displacements from the forces. Often after that, a degree of interpolation is used to calculate the deflection of the lines connecting the nodes.

Like all methods that involve splitting a structure into smaller elements, the smaller the element size, the more accurate FEM is. However, adding more nodes increases computation time so the result is a trade-off. Since my pulses are primarily sinusoidal wave packets, elements must be smaller than the dominant wavelength – the suggested value is a size of $\lambda/20$. The FEM method requires three steps: Pre-processing, Solving and Post-processing. These are summarized in Figure 1 below.

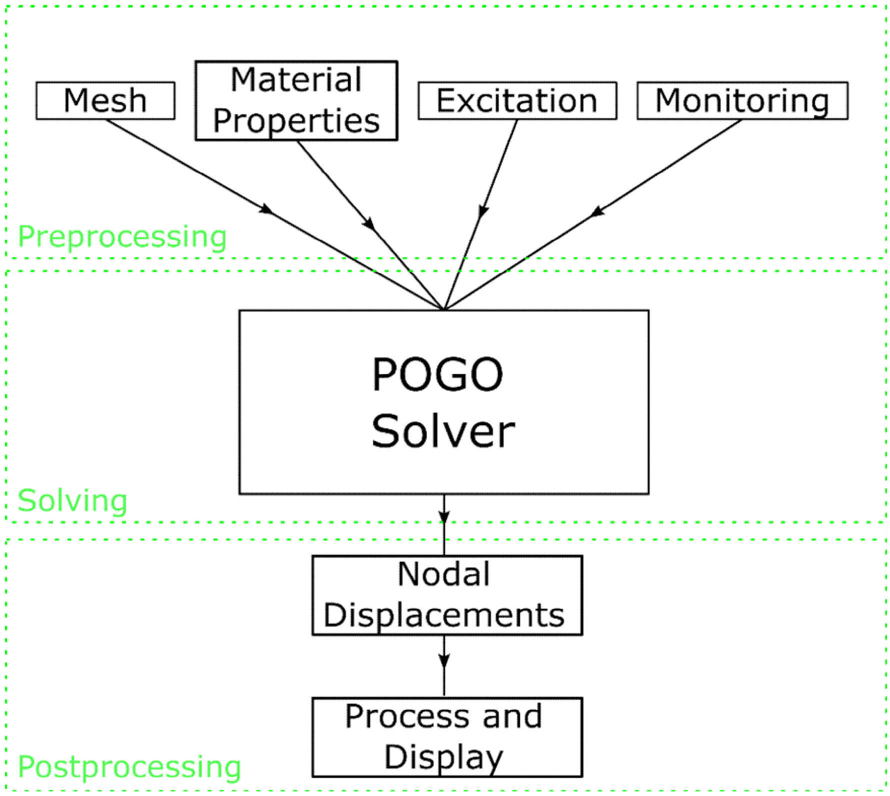


Figure 1. The FEM process

Pre-Processing

As can be seen in Figure 1, for the acoustic wave propagation problems I am primarily working on, the mesh, material properties, site and waveform of the excitation and the nodes monitoring the reflected pulse must be defined before POGO can solve it. I primarily focused on creating the mesh, leaving most the other problems to be solved by my colleagues.

The major meshing program, mesh2d, was created by Darren Engwarda. Since it was created to be used mostly for structural analysis programs, it had various quirks. For example, around sharp corners the mesh size was much finer, and although this is desired in structural analysis where we want to find stress concentrations in equilibrium, this is not wanted in wave propagation problems, where changing the element size in a single mesh will change the impedance. Therefore, the first task was to force the mesh to be uniform and changing the density of nodes along edges to have a degree of control of the element size.

The second major task was to write a software interface to import .dxf files generated by CAD software and classifying them so that objects with holes could be meshed. The DXF file format was chosen as it is a very commonly use file type for all CAD software which makes the import interface very flexible. Since the meshing algorithm took two main inputs – the nodes on the outside of the mesh and how they were connected to each other – the main challenge here was designating connectivity so that the outside edge and inside edge were not connected to each other. This could be extended to holes one element wide, which worked as cracks.

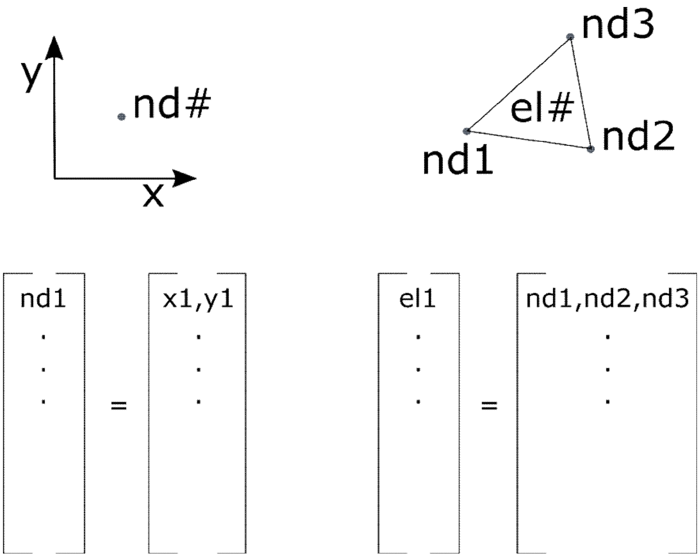


Figure 2. Mesh definition showing array of nodes and elements

One thing I attempted to do was to create cracks of zero size, since some defects would be very small compared to the wavelength. This involved creating a line of nodes in the mesh, duplicating the nodes along this line, and making sure the elements above the line were connected to the originals and the elements below the line were connected to the duplicates.

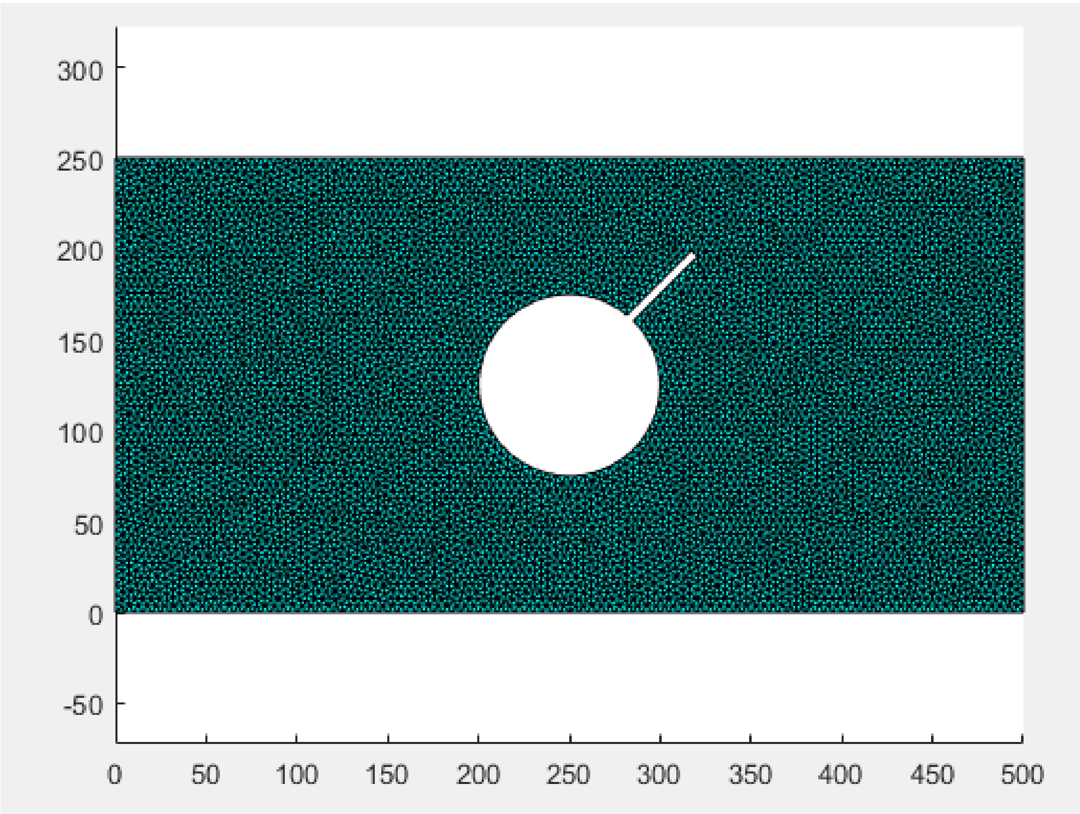


Figure 3. A mesh showing a hole and 'crack'

It helped that the mesh itself was saved as a struct containing the nodes and the elements, and the nodes in each element were designated by their index in nodes, see Figure 2. As a result, the duplicate nodes could be appended to the end of the matrix of nodes, and elements could be reconnected by changing index numbers in some of the nodes. However, it transpired that the main problem was creating one or more lines of nodes in an arbitrary mesh that may or may not contain holes. Just as with the creation of holes, the difficulty turned out to be in correctly labelling the connectivity of nodes. As of now, we have decided to assume that cracks are at least one element wide, see Figure 3 and see if any issues arise from that.

Post-Processing

I had a lot less time on post-processing and as a result was unable to do anything too intricate. Since the data was noisy, I removed some of the noise by setting a threshold – i.e. by refusing to plot any data with an amplitude that was below a certain value. Since most the noise was high-frequency, a more sophisticated way of removing that would have been a form of Fourier analysis.

Results

Wave behaviour

As seen in Figure 4, the wave behaved as expected. As the wave travelled through the material, it caused some dispersion, growing wider. The central part reflected directly back at the transducer since the wave was directed straight at the top of the hole. The wave curved slightly as it reflected, reflecting the hole's geometry. The edges of the wave did not reflect and were instead diffracted around the hole, creating a shadow in the wake of the hole. Although macroscopically this is the expected behaviour, it is worth verifying it properly at some later date.

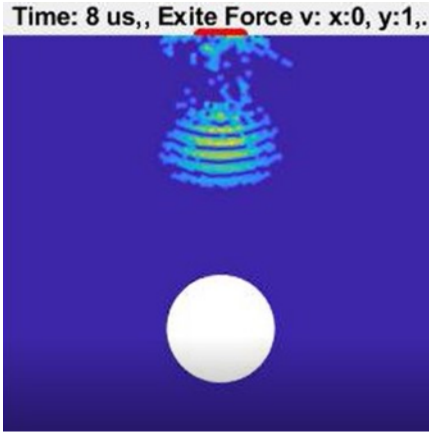
Noise

There was a great deal of high-frequency noise behind the wave that clouded signals. Although it was less bad for outgoing waves, it tended to interfere with incoming signals and thus produce a signal at the transducer nodes that was heavily distorted.

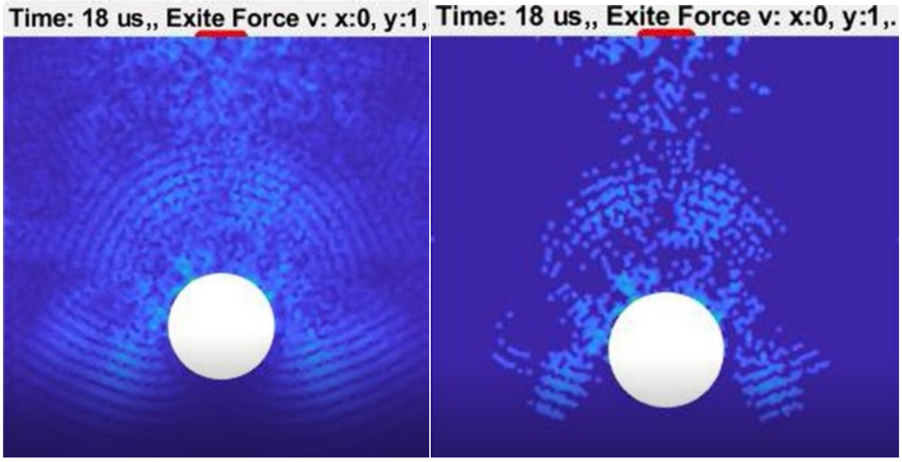
The manual cites one explanation of the noise – that the mesh size was too large. Although the mesh was approximately the right size compared to the dominant wavelength, in practice the actual wave packet was made from a windowed signal containing a spread of frequencies. Therefore, the mesh had to be the correct size to fit in the highest frequency in the signal. Another explanation of the noise is that although the material is isotropic, the mesh is not - elements look different from different directions. This could lead to reflections from circular waves that in practice would not happen. Both issues could be mitigated by reducing the element size. Since the noise was comparatively reduced when I decreased the frequency and thus increased the wavelength for the same mesh, this shows that I am on the right track. However, increasing the frequency by more meant that the size of the hole was on order with the wavelength and thus waves tended to diffract around it. Therefore, the mesh really did need to be finer. However, this put pressure on my computer, as detailed before.

Another cause of noise was edge effects from the transducer. Towards the centre of the transducer, the pulse is approximately flat but at the edges it looks more like a point source. This creates waves at the edges that interfere and reflect, creating noise. There was a great deal of noise like this when the transducer was made of comparatively few nodes. One way to mitigate that would be to make the transducer weaker at the edges than at the middle, and in practice most transducers are like that.

The primary way I removed noise is through adding a threshold that physically cut out most of the noise. Although that made the outgoing signal much more obvious, it cut out the returning signal and did not successfully remove all the noise.



Outgoing signal



Reflected signals with/without threshold applied

Figure 4. Screenshots from the animated display of wave propagation

Other ways of removing the noise involve Fourier methods, since the noise is a much higher frequency than wave or averaging the signal over two or three frames.

Computational Power

I did most the processing on my personal computer. Although POGO itself was stored on a remote desktop, and all the files I generated were stored on a separate drive, the calculations themselves were done on my computer. The things that tended to lock up my computer included meshing and creating the videos of the wave propagation. Therefore, I was forced to change the pre-processing programs that James had written so that my computer could run them. One change was saving and accessing meshes in a folder so I didn't have to generate a mesh more than once. Had I continued doing this project for longer, I would have cut the number of frames in the final output video by a factor of 2 or more, since the timesteps were very short.

Conclusion

In conclusion, the main objectives of the project have been met and there is now a suite of programs which enable CAD files (in DXF format) to be imported and meshed automatically. These files can include internal holes and crack as well as complex geometries. The meshing algorithm has been optimised for wave propagation calculations by forcing uniform mesh density throughout. Additionally, post processing functions have been written to allow visual representation of the propagating waves in the structure.

Additional work is required to extract more complex data from the results which include simulating what a second transducer would detect, rather than the wave field images.