Creating an FTAN map

This file shows two ways of creating an FTAN map and digitising the curve. The default approach (1) is to digitise a matplotlib instance. However, in some cases it us useful to revist an FTAN map and re-digitise, and we provide that option as a Figure loading example (2).

In either case, the first three clicks are expected to be the origin, the end of the x-axis, and then the end of the yaxis. To reiterate:

Click 1 = origin
Click 2 = end of x-axis
Click 3 = end of y-axis
Make sure that the pixel coordinates are being displayed when clicking. If not, you may not be in a valid area of the plot. If the first three clicks aren't in this order, weird things can and will happen during the linear tranformation to real coordinates.

To begin with, we import the libraries, and then our file containing our waveform (active trace started at trigger shot, or alternatively cross-correlation function).

In [ ]:

```
from FTANos import *

#Input the filename of the seismic data
sfile = "example_trace.sac"
```

Next we need to set the parameters that will define the FTAN map. Typically these involve setting an upper and lower frequency (or period) on the x-axis, and an upper or lower velocity on y-axis. Note we focus on Rayleigh wave group velocities in these examples.

In [ ]:

```
freq1 = 10
freq2 = 500
vel1 = 100
vel2 = 1400
```

Note that the FTAN initialisation can be done with the default parameters (see code base for list). In that case, we just initialise the class with the following:

In [ ]:

```
outfile = FTANos(filename=sfile)
# all the parameters listed can be changed according to the survey or set
```

Alternatively we can initialise the FTANos structure with our own parameters like this:

In [ ]:

```
outfile = FTANos(filename=sfile, fre1=freq1, fre2=freq2,vel1=vel1,vel2=vel
```

See the main codebase for full descriptions. The frequencies and velocities define the bounds of the FTAN map.

alpha is the gaussian kernel half-width for FTAN plotting.

ftan_sc is a scaling term for plotting (from aftan). It is generally 60 but can be varied for different data noise levels.

dist is distance between source and receiver

Plotting the FTAN map

The following has two ways of digitising the FTAN map. The first creates the plot and digitises in one step.

The second uses an existing FTAN map (here called "example_trace.sac.png"), and digitises that. This still assumes that the plot follows the frequency and velocity bounds defined in the previous step.

Option (1): create map and digitise directly in matplotlib:

In [ ]:
```
outfile.plot_FTAN()
```

Note this uses matplotlib's ginput function, and thus works best outside of a Jupyter notebook environment (where all matplotlib objects are rendered as png images). The command is best run from a terminal of API like Spyder.

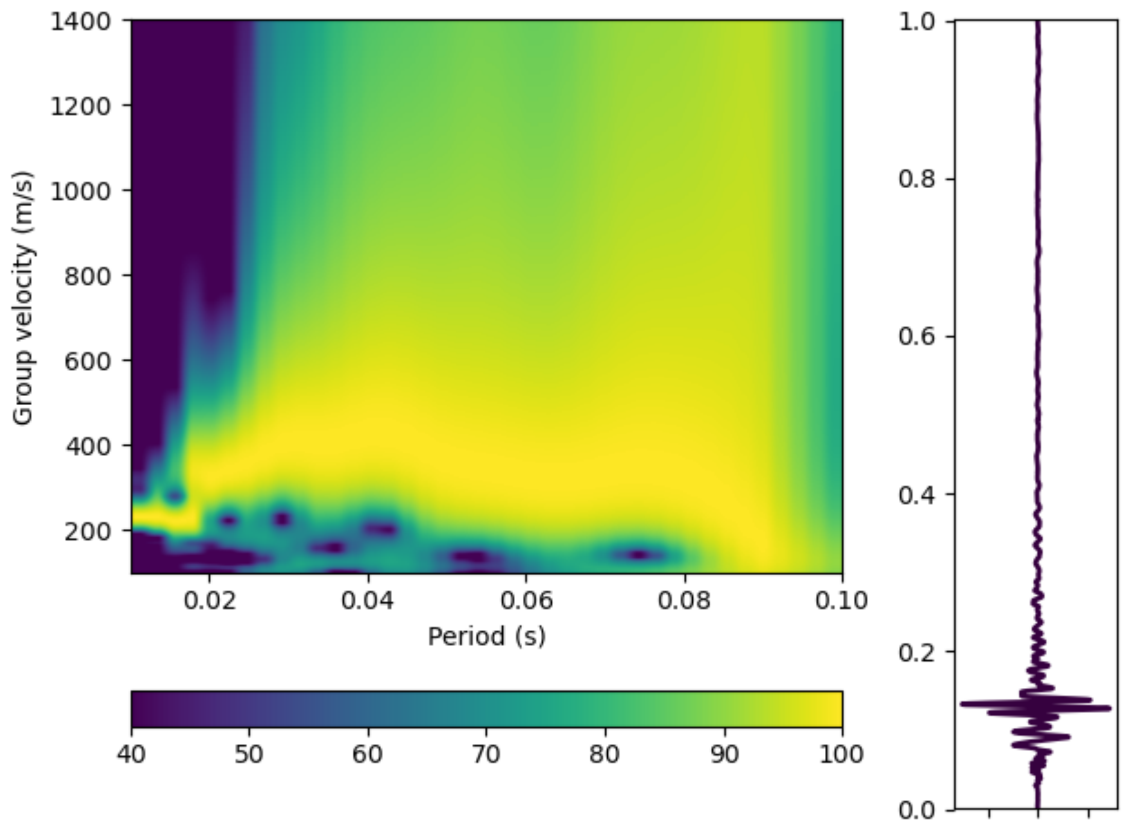
Options (2): Import existing FTAN image file to digitise:

In [ ]:
```
outfile.plot_digitise_file(im_file="example_trace.sac.png")
```

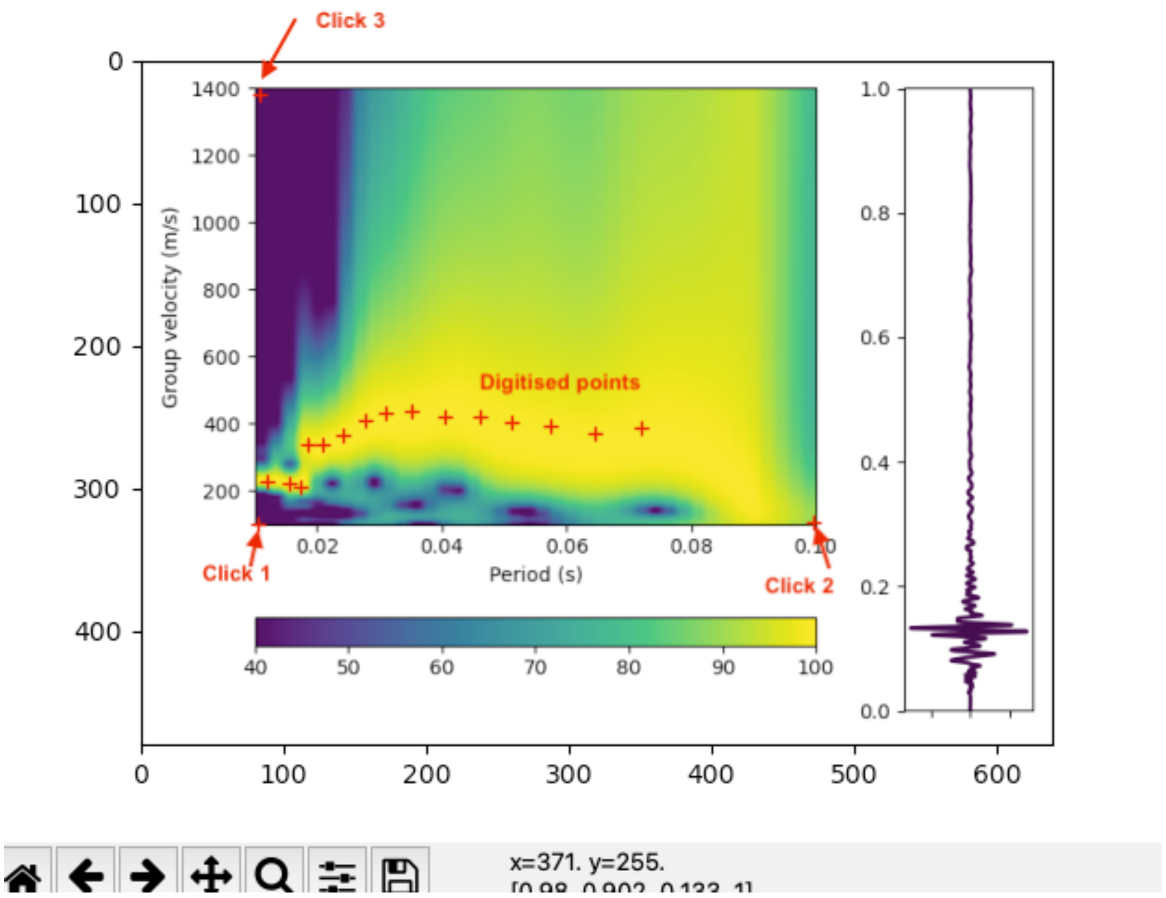
In either case, an output dispersion will be made, called "example_trace.sac.disp", in a (periods (s), group velocity (m/s)) format.

Outputs

Below is an example of a produced FTAN map from the previous commands.



An example window from the digitising step is shown below. The first three clicks at the origin, x axis maxima, and y axis maxima, are shown. Subsequent points are treated as data. Pressing escape exits the program and saves the points in a file called (here) "example_trace.sac.disp", where "example_trace.sac" was the original data file.



In [ ]:

```

```