Manual for Adaptive Stacking Program tcas

by Nick Rawlinson and Brian Kennett

Research School of Earth Sciences, Australian National University, Canberra ACT 0200

1 Introduction

This document describes how to compile and run the adaptive stacking code tcas, and includes detailed descriptions of the input and output file formats. tcas is a FORTRAN77 code, and encapsulates the adaptive stacking method described in the paper:

- Rawlinson, N. and Kennett, B. L. N. 2003. Rapid estimation of relative and absolute delay times across a network by adaptive stacking, *Geophys. J. Int.*, **Submitted.**

Please note that since tcas is freeware, the authors cannot be held accountable for any bugs or other shortcomings in the code. However, if you do encounter any problems, please email a report to **nick@rses.anu.edu.au**.

2 Brief synopsis of files contained in the distribution

- **tcas.f:** FORTRAN77 code for adaptive stacking. Although the code makes a few departures from what is strictly regarded as FORTRAN77, it should compile without difficulty on most modern fortran compilers such as g77.
- **tcas.cmd:** Input command file for tcas.f. This file provides the name of the input data file, the norm used for the misfit measure, the error limits, the picking error coefficient etc.
- **Makefile:** The Makefile used to compile tcas. Since the complete code is contained in a single file, the Makefile is simple, and the compilation could be achieved by just using a command line like `f77 -o tcas tcas.f`.

- **ts1101559.aq:** Example input data file in AQ format. An AQ file is an ASCII text file with a very simple format (described later). Most people who intend to use tcas will have access to code that will read standard data file formats (like SEED, SAC etc), so extracting relatively small portions of traces in AQ format should be quite simple.
- **asi1101559.aq:** AQ file that is essentially the same as ts1101559.aq except that the linear stack and quadratic stack have been added to the original traces.
- **asf1101559.aq:** The final AQ file that results from adaptive stacking. All traces should be well aligned, and the linear stack should be a good representation of the "true" noise-free trace across the array.
- **ts1101559.ttr:** Output data file containing the magnitude of traveltime residuals and associated error estimates.
- **aqplot.f:** FORTRAN77 program for plotting the contents of an AQ file. This program requires PGPLOT, a Fortran graphics subroutine library, to be installed. PGPLOT can be obtained for free from **http://www.astro.caltech.edu/~tjp/pgplot/**. However, note that since AQ is such a simple data format, it shouldn't take much effort to plot these files using your favourite graphics program e.g. GMT.
- **aqplot.in:** Input parameter file for the program aqplot.
- **asiplot.ps:** PostScipt plot of the AQ file asi1101559.aq, made using the program aqplot.
- **asfplot.ps:** PostScipt plot of the AQ file asf1101559.aq, made using the program aqplot.

3 Compiling and running the code

The source code of tcas is FORTRAN77, and is simple to compile. A Makefile has been provided - if you wish to use it, just replace g77 with your favourite Fortran compiler (if necessary) and type make at the command line prompt. Otherwise, you can just type in something like `f77 -o tcas tcas.f`.

To test that tcas works, try it out with the example file provided i.e., ts1101559.aq. You won't need to edit the default input file, because it is already set up for these data. Note that the

name of the file is just a concatenation of the survey name (TIGGER Shortperiod or ts), and the time of the event (Julian day 110, hour 15, minute 59), and that it corresponds to Figure 2 in the GJI paper referenced at the beginning of the manual.

To run the program, simple type tcas. With the default example, you should get the following output on the screen:

```
pstakn = 1.76544201
pstakn = 1.76426828
pstakn = 1.76422405
pstakn = 1.76422405
pstakn = 1.76422405
```

pstakn is the value of the trace misfit measure (the difference between the stacked trace and each station trace). In this case, five iterations of the adaptive stacking procedure have been applied, but convergence has been achieved after just two iterations. Note that in our experience, most of the misfit reduction occurs in the first iteration. For residuals larger than those encountered in TIGGER, this may not be the case.

When tcas is executed, it creates three output files in addition to the screen output. In the example above, these files are asi1101559.aq, asf1101559.aq and ts1101559.ttr. It is important to note that these file names are automatically generated from the input file name. The part of the code that does this is below:

```
c
c   Construct initial and final stack output
c   filenames.
c
aqini="asi"//efile(3:9)//".aq"
aqfini="asf"//efile(3:9)//".aq"
c
c   Construct file name for traveltime
c   residual output.
c
ttres="ts"//efile(3:9)//".ttr"
```

In other words, the code takes letters 3-9 of the input data file ts1101559.ttr and appends it to asi, asf and ts, before adding the appropriate file extension. It is relatively simple to alter this structure in the code if it doesn't suit your purposes.

The output AQ files can be plotted using the supplied program aqplot, which is described in more detail below.

4 Format of input files

4.1 tcas command file

tcas.cmd is the command or input parameter file for the program tcas. In the example distributed with the code, the file contains the following:

```
5                       number of iterations
3.0                     index for phase stack
1.15                    coefficient for pick error
25.0    150.0           min, max error limits (ms)
12.0    10.0            stack window (start,length)
ts1101559.aq            event file
-1.0                    min diff time
1.0                     max diff time
```

The comments following the parameters make this file somewhat self-explanatory, but further information will be provided. The `number of iterations` refers to the number of times the adaptive stacking loop will be run. The `index for phase stack` controls the norm used to measure the difference between the stacked trace and each station trace. The above example uses an L_3 norm. The `coefficient for pick error` is the value of ϵ referred to in the GJI paper. The `min, max error limits` constrain the error between the two given values; these can be set to zero (`min`) and some very large value (`max`) if you want to allow the error to be completely unconstrained.

The `stack window (start, length)` specifies the start time and length of the stack window. The start time is measured from the time given in the header of the AQ file.

This window should span the length of the wavelet you wish to use for adaptive stacking. If you look at the file asiplot.ps, you will see that the start time (12 s) and length (10 s) allows the stack window to capture the main P pulse. The `event file` is the data file in AQ format that has been approximated aligned using model predictions. The `min diff time` and `max diff time` refer to the limits of allowable time shifts used to find the minimum of the difference between each stacked trace and the reference trace. In the teleseismic case, ±(1.0 - 2.0) s should be sufficient.

4.2 AQ input file

This is the input data file in AQ format, and is called ts1101559.aq for the example provided. This is a simple ASCII text file comprising a main header followed by trace time series with local headers. The first ten lines of ts1101559.aq contains:

```
72
-16.3800    173.2600    33.00
2002    4    20
16    6    27.7905    389.9905
0.0500    P
1    640    -3.0557    tsa1
3158.69019 3149.93091 3085.04834 2993.58228 2862.65332 2677.26245
2428.67432 2098.42114 1682.52197 1203.59558 687.729797 164.642014
-321.779907 -730.800842 -1031.34155 -1188.25146 -1175.22717 -999.087585
-694.296631 -304.492126 126.761864 559.635559 959.327881 1297.48315
```

`72` is the number of stations in the array. `-16.3800 173.2600 33.00` is the latitude, longitude and depth (in km) of the event. `2002 4 20` is the year, month and day of the trace start time; `16 6 27.7905 389.9905` is the hour, minute and seconds of the trace start time, followed by the number of seconds between the event start time and the trace start time; and `0.0500 P` is the sample interval (in seconds) and phase type (for teleseismic events, this could be P, PcP, PP etc.). These values comprise the main header of the AQ file. Note that tcas only makes use of the number of stations in the array (`72`) and the sample interval (`0.05`), so in principal the other entries could be dummy values if necessary. The reason we include

them is that they make the output files somewhat self-contained, and the estimation of absolute traveltimes easier to work out.

The next line, `1 640 -3.0557 tsa1` is the local header for the first of the 72 traces (in this case). The first entry is a scaling factor for the trace. In general, set this to `1` if the trace exists and you want it to be used in the adaptive stacking routine. Otherwise, set it to `0` (e.g. if the station didn't record data or you want to ignore the trace because it contains no signal). The next entry (`640`) is the number of points in the time series that comprises the first trace. This is followed by the model prediction time shift or moveout correction (`-3.0557` s) that approximately aligns the trace with a reference station. In this example, *ak135* predictions for P are used, and station `tsi3` is used as the reference trace. Note that the moveout correction for station `tsi3` is not exactly zero, because we also include a small time shift to account for the fact that the first sample of each trace does not start at exactly the same time (but within a sample interval of each other). The final entry (`tsa1`) is the station name. The next 640 entries (we only show 24 above) are the values of the trace time series.

The Fortran code that is used to read in data in AQ format is:

```
c
integer nsta,iyr,imon,iday,ihr
integer imn,w(MAXST),npts(MAXST)
real evlat,evlon,evdep,sec,ttto
real sampin,tshift(MAXST),zu(MAXST,MAXD)
character*8 akph, csta(MAXST)
c
read(EVT,*) nsta
read(EVT,*) evlat, evlon, evdep
read(EVT,*) iyr,imon,iday
read(EVT,*) ihr,imin,sec,ttto
read(EVT,*) sampin,akph
DO i=1,nsta
    read(EVT,*) w(i),npts(i),tshift(i),csta(i)
    read(EVT,*) (zu(i,j),j=1,npts(i))
ENDDO
```

where EVT is the Unit associated with the AQ input file, MAXST is the maximum number of stations and MAXD is the maximum number of points in the time series.

Obviously, in order to use tcas, you will need to write a program to convert data from your own format (e.g. SAC) to AQ format, but this shouldn't be difficult. You will also need some means of approximately aligning traces with model predictions. A means of doing this with *ak135* can be found at the web site from which you obtained tcas. These predictions can also be used to help determine the start time and length of the trace segment to be saved in AQ format.

5 Format of output files

5.1 AQ output files

tcas writes output to two AQ files, asi*.aq and asf*.aq. For the given example, these files are asi1101559.aq and asf1101559.aq, and respectively show the initial alignment of traces and the final alignment of traces. asi*.aq is essentially the same as the AQ input file, except that two traces have been added: the linear stack (labelled `zssl`) and the quadratic stack (labelled `zscp`). In the example, these traces are numbered 73 and 74. The AQ file of the final stack (asf1101559.aq) should show a good alignment of all traces and a more defined quadratic stack. Both files asi1101559.aq and asf1101559.aq are plotted in the GJI paper as Figure 2a and 2b respectively.

5.2 Traveltime residual and error estimates

The file containing traveltime residual and error information has an extension *.ttr; in the example provided, this file is called ts1101559.ttr. The first 12 lines of this file are shown below:

```
72
-16.3800     173.2600     33.000
2002     4     20
16     6     27.7905
389.9905
0.0500
P
```

```
5
1 tsa1 0.0500 0.0410 -3.0557 1
2 tsb1 0.1500 0.0250 -2.5817 1
3 tsc1 0.2000 0.0250 -2.0693 1
4 tsd1 0.2000 0.0250 -1.2506 1
```

The first few lines of this file are similar to the input AQ file. `72` is the number of stations; `-16.3800 173.2600 33.000` is the latitude, longitude and depth of the event; `2002 4 20` is the year, month and day of the start time of the trace; `16 6 27.7905` is the hour, minute and seconds of the start time of the trace; `389.9905` is the time interval between the event time and the trace start time; `0.0500` is the sample interval and `P` is the phase type. The last value in the header (`5`) is the number of adaptive stacking iterations used to achieve alignment.

The list that follows is station specific information; in our example there are 72 entries, which correspond to the number of stations in the array (we only show the first four above). The first column is just the station number; the second column is the station name; the third column is the time shift required to achieve alignment (i.e. the *ak135* residuals in our example); the fourth column are the error estimates associated with the residuals; and the fifth column contains the time shifts required to obtain the initial approximate alignment (*ak135* moveout corrections in the example). The last column indicates whether the station has been used in the adaptive stacking process. When this value is 1, then it has been used; when it is 0, it has been ignored, in which case the residual is set by default to zero and should not be used. This value simply comes from the first entry in the trace header (the line above each data block) of the AQ file.

6 Plotting AQ files with aqplot

The simple format of AQ files as described above means that they should be relatively straight forward to plot with your favourite graphics program. However, we supply a simple plotting program, called aqplot for your convenience. The only drawback of this program is that you need access to the PGPLOT graphics subroutine library, but this can be obtained for free at:

- http://www.astro.caltech.edu/~tjp/pgplot/

Once you have installed PGPLOT, you can compile aqplot with a command like:

- `f77 -o aqplot aqplot.f -lpgplot -lX11`

To run the program, just type `aqplot` at the command line. The default output file name is pgplot.* (e.g. pgplot.ps if you opt for postscript output).

The input command file for aqplot is called aqplot.in and for the example provided contains the following:

```
asf1101559.aq          c:  Name of .aq file for plotting
/vps                   c:  Output device (/ps,/vps,etc)
5.0 10.0               c:  Width, height of plot in inches (8x11.3=a4)
1.0                    c:  Character height
1                      c:  Line thickness
0                      c:  Include auto-label (0=no,1=yes)
```

The entries are quite self explanatory. The output device depends on the device drivers that you choose to enable when you compile PGPLOT. For a full install, most common graphics formats are allowed. `/ps` refers to landscape postscript while `/vps` refers to portrait postscript. To get a full list of the output options, use a question mark `?` as the output device. The auto-label option generates a label at the top of the plot based on the header information in the AQ data file.