**Cheat sheet for running multiplet detector**

**26 July 2013 – Kate Allstadt**

**Disclaimer:**

-The detector is currently set up to detect repeaters that repeat at least once each day. The codes would have to be modified to detect less frequent repeaters (and you’d probably only want to compare the detections with really large STA/LTA or else your computer might burst into flames).

-The codes are also set up to build repeater families one month at a time, they can be combined later if families cross between months, or you can modify MULTDETECTlongterm to run for a longer than one month at a time, but it might freeze your computer if you make the time period too long with too many events to compare.

-The codes I’m giving you are basic, there are more complicated methods I’m using that help eliminate garbage detections (saves processing time) but they require more work to implement and it’s probably better to start simpler with a new dataset.

-Make sure you have a lot of room on your computer because all the data for events is saved locally and it adds up fast. There are files you can delete as you go along if you’re happy with results of each step (I do this automatically sometimes but these codes don’t do it).

**To run:**

*Edit MULTDETECTlongterm.m*

-edit the addpaths at the top of the file for your computer (obviously)

-make sure defaultOptStruc.m is in the folder you’re running from or on your path (I like to keep it in the same folder, not buried in the codes, because it is something you edit for each run)

-If you have the parallel computing toolbox, uncomment ‘matlabpool(#)’ before each loop to use it on, uncomment matlabpool close after each loop, and change ‘for’ to ‘parfor’ for each loop. If you have a lot of processors, this can speed things up severalfold.

*Edit defaultOptStruc.m*

-This is where you define your STA/LTA times and ratios, add any filtering you want and how much data to extract before and after each detection etc. etc., it is well commented

-you can rename this to save a separate one for each run you do (so you know what parameters you used) but you need to update the name in MULTDETECTlongterm if you do

-just leave opt.maxNumClust, opt.tbin, opt.test, opt.level as they are

*Run MULTDETECTlongterm for each station to detect multiplets*

What it does:

1. Goes through each month, one day at a time (getPicks2) that loads data in a bit at a time, uses STA/LTA to find detections, and when it does, it cuts the data out and saves it all in a separate ARRAYPICKS file for each day (splits into 2 files if there are too many events)
2. Then it reads in the ARRAYPICKS files and separates all the events into families based on criteria you set in defaultOptStruc. The remaining events are designated as ‘orphans.’ ARRAYPICKS files can be deleted after this point if you are happy with the results in CLUSTERPICKS because CLUSTERPICKS contains all the same data just organized differently. This is in a separate loop from the getPicks2 step because it is more memory intensive and you can’t have as many parallel loops at the same time for this one.
3. Then families detected each day within each month are combined (getRaClusters). Note that at this point, if a family crosses into a separate month, it is saved as a separate family in a new file for that next month. You can delete CLUSTERPICKS after this point if you are happy.

Postprocessing:

1. Run fixMulttemplALL to review events in each month file, (edit the things at the top and a path in here to correspond to the folder containing subdirectories for each month), this saves MULTTEMPL\_GOOD.mat. You then browse through a stack and a subset of detections and decide whether to keep it (press enter) or throw it out (press x then enter).
2. Run multtimeline to make a timeline plot of each month
3. There is a stack of each family saved in the structure of MULTTEMPL\_GOOD, as well as the data for each individual event (multtempl.multiplets), and the stack is saved as multtempl.stack. Use coralPlot to plot things. coralFilter to filter etc. etc.
4. coralRaStack makes stacks using the median, coralRaStackNoZero.m makes stacks using the median that ignores flatlined data.
5. browseMults and browseMultsMany are like fixMulttemplALL, they let you look through the events one by one, but don’t alter the data (and aren’t well commented, they are written as scripts right now).
6. multPerDay.m and multPerHr.m output vectors that quantify the number of multiplets per day or per hour for the given multiplet set (input multtempl.multiplets and a minimum size).

THESE NEXT CODES ARE REALLY NEW AND A MESS AND PROBABLY AREN’T COMMENTED WELL, ASK FOR HELP WHEN YOU GET TO THIS POINT

1. Use ExtractTemplates.m to combine sets across months or even years. You have to edit a lot of stuff in here to get it to run and it’s a mess, ask me for help when you want to do this.
2. Run TEMPLATEDETECTlongterm.m with the results of ExtractTemplates.m to pull out any missed occurrences of each multiplet set
3. Run reconcileMTcatalogs.m to combine the multiplet detection and template detection catalogs for each station separately
4. Run combineCatalogs.m to figure out which events in each separate stations catalog correspond to events in other stations catalogs (this is only written for 2 stations at present, I have another code somewhere that can do more, let me know if you want me to find it)
5. Run WorkWithBigSets.m to pull out data from a bunch of stations for whatever sets you want to work with and make stacks to get locations etc.