Modules and functions of the MTpy package – overview

Kent Inverarity, Jared Peacock, Lars Krieger*

2012

*University of Adelaide – School of Earth and Environmental Sciences – Mawson Building – Adelaide

Introduction

This is a collection of all *MTpy* subpackages, modules, functions, and scripts.

July 10, 2015, LK

Contents

1 Structure

1.1 overview

MTpy is organised in subpackages. They are separated by the main topic of the contained modules and functions.

Besides these subpackages, there is an extra folder, which contains available documentation.

1.2 subpackages

The existing subpackes are

- *core*
  These modules deal with the elementary steps of general MT processing steps (mttools) and the attributes of the impedance tensor (Z).
- *imaging*
  Modules and functions, which deal with the general visualisation within the analysis of MT data. More specific plotting routines are around in other subpackages (e.g. processing).
- *modeling*
  Wrapper and auxilliary functions for the handling of external standard MT inversion and modelling tools: WingLink, Occam, Ws3Dinv, ModEM,...
- *processing*
  Wrapper and functions for the processing of MT data. Mainly concentrating on BIRRP.
- *utils*
  Tools and scripts for helping with small or very specific tasks. For instance conversions of units, data formats, or modeltypes. To be included later: GUIs for running BIRRP, OCCAM, Ws3Dinv.

After the installation of MTpy, the respective subpackages can be imported with
`import mtpy.<subpackagename>`.
For the direct import of a module type
`import mtpy.<subpackagename.<modulename>` instead.

1.3 command line scripts

Command line scripts are directly callable from the terminal (provided Python ios installed). Arguments are following the scriptname, all separated by just a blank space.

From within an IPython session, these scripts can be called as if the call was from the terminal by putting a preceeding `run` in front of the call.

2 core

2.1 z

Module for dealing with the MT impedance tensor **Z**. The main class (Z) inherits a class that handles data files (Edi). Attributes of **Z** can be determined, visualised, and converted.

2.1.1 Edi (Class)

A class for reading EDI files. Additionally, their contents can be changed and the files can be rewritten.

readEDI (Method)

Read in Edi files.

rewriteedi (Method)

Write out Edi files.

2.1.2 Z (Class)

Defining an impedance tensor object. This can then be analysed, transformed, visualised. For instance, REsistivity-Phase-Plots can be generated.

getInvariants (Method)

.

getPhaseTensor (Method)

.

getTipper (Method)

.

removeDistortion (Method)

.

removeStaticShift (Method)

.

getResPhase (Method)

.

getResTensor (Method)

.

plotResPhase (Method)

.

plotPTAll (Method)

.

plotTipper (Method)

.

2.1.3 PhaseTensor (Class)

Generates a phase tensor object from given $\mathbf{Z}$, following Caldwell et al. [2004].

2.1.4 ResPhase (Class)

ResPhase is a resistivity and phase class...

2.1.5 Tipper (Class)

Generates a Tipper element from tipper vector and given rotation.

2.1.6 Zinvariants (Class)

Class for holding just the invariants of $\mathbf{Z}$

2.1.7 ResistivityTensor (Class)

gets components of the resistivity tensor defined by O'reilly [1978] and Weckmann et al. [2003]

2.1.8 PhaseTensorResidual (Class)

calculates the tensor residual as defined by $\Delta\Phi_{1,2} := \hat{\mathbf{I}} - \Phi_1^{-1}\Phi_2$

2.1.9 ResistivityTensorResidual (Class)

calculate the resistivity residual between two tensors $\Delta\Phi_{1,2} := \hat{\mathbf{I}} - \Phi_1^{-1}\Phi_2$????

2.2 mttools

Module for helping with all general MT analysis steps

2.2.1 convertIpB (Function)

Convert the magnetic field from counts to units of $\mu V/nT$.

2.2.2 convertE (Function)

Convert the electric field from counts to units of $\mu V/m$.

2.2.3 rotateE (Function)

Rotate the electric field to geographic north

2.2.4 rotateB (Function)

Rotate the magnetic field such that B_x is pointing to magnetic north and B_y to geomagnetic east.

2.2.5 padzeros (Function)

return a function that is padded with zeros to the next power of 2

2.2.6 filter (Function)

apply a sinc filter of width w to the function f by multipling in the frequency domain.

2.2.7 dctrend (Function)

remove a DC trend

2.2.8 normalizeL2 (Function)

return the function f normalized by the L2 norm:$\frac{f(x)}{\sqrt{(\sum_i |x_i|^2)}}$.

2.2.9 decimatef (Function)

decimate a function by the factor m. First an 8th order Cheybechev type I filter with a cuttoff frequency of $0.8/m$ is applied in both directions to minimize any phase distortion and remove any aliasing.

2.2.10 openMTfile (Function)

Open an MT raw data file, convert counts to units and return an 1D-array.

2.2.11 convertCounts2Units (Function)

Conversion counts -¿ units ...???

2.2.12 combineFewFiles (Function)

Combine files in a directory path (dirpath) that have a given start and end time in the form of (HHMMSS). It looks for files at cachelength then decimates the data.

2.2.13 combineFiles (Function)

Combine raw data files from different days into one file.

2.2.14 adaptiveNotchFilter (Function)

apply a notch filter to an array by finding the nearest peak around the supplied notch locations. The filter is a zero-phase Chebyshev type 1 bandstop filter with minimal ripples.

2.2.15 removePeriodicNoise (Function)

take average a window of length noise period and average the signal for as many windows that can fit within the data. This averaged window is convolved with a series of delta functions at each window location to create a noise time series. This is then subtracted from the data to get a 'noise free' time series.

2.2.16 imp2resphase (Function)

convert impedances z[4,3,len(freq)] to resistivities (ohm-m) and phases (deg) as well as the errors of each. Note the phase is calculated using arctan2 putting the phase in the correct quadrant. The xy component is placed into the positive quadrant by adding 180 deg.

Does NOT take impedance objects though!!

2.2.17 sigfigs (Function)

return a string with the proper amount of significant digits for the input number

2.2.18 writeedi (Function)

write an .edi file for one station

2.2.19 rewriteedi (Function)

rewrite an edifile for one station, say if it needs to be rotated or distortion removed.

2.2.20 getnum (Function)

get number from string list and put it into an array ??????

2.2.21 readedi (Function)

read in an edi file written in a format given by `http://www.dias.ie/mtnet/docs/ediformat.txt`. Returns: lat,lon,frequency,Z[zreal+i*zimag],Zvar,tipper,tippervar (if applicable)

2.2.22 combineEdifiles (Function)

combine edifile1 with edifile2 according to nread1 (integer of number of frequencies to read in from edifile1.Index is from the start, ie [0:nread1]) and nread2 (integer of number of frequencies to read in from edifile2. Index is from end of frequencies, ie [-nread2:]). It will combine frequencies, impedance and tipper.

Note nread1 is from the start for edifile1 and nread2 is from end for edifile2.

2.2.23 sil (Function)

split a single line written in an .ini file for burpinterface and return the list of strings.

2.2.24 readStationInfo (Function)

read in a .txt (tab delimited) or .csv(comma delimited)file that has the following information: station name, latitude, longitude, elevation, date collected, components measured (a number: 4 for E_x, E_y, B_x, B_y, 5 for E_x, E_y, B_x, B_y, B_z, 6 for $E_x, E_y, B_x, B_y, B_z, TP$, magnetic type (BB or LP), dipole lengths (m), data logger gain (very low=2.5,low=1,high=.1),interface box gain (10 or 100), B_z correction for longperiod data and the B_x, B_y, B_z components as they were measured in the field. Use headers: station, lat, lon, elev, date, mcomps, magtype, ex, ey, dlgain, igain, lpbzcor, magori. Returns a list of the dictionaries with found information.

2.2.25 getStationInfo (Function)

returns info for the nominated station from the stationinfofile as a dictionary with key words in the hdrinfo.

2.2.26 convertfiles (Function)

convert data of counts from data logger to units....again????

2.2.27 removeStaticShift (Function)

remove static shift by calculating the median of respones of nearby stations, within a given radius. If the ratio of the station response to the median on either side is out of 1±tolerance then the impedance tensor for that electric component will be corrected for static shift.

2.2.28 makeDayFoldersFromObservatoryData (Function)

take observatory data and put it into day folders for processing.

2.2.29 makeKML (Function)

make a kml file for Google Earth to plot station locations relatively quickly

2.2.30 getPeriods (Function)

Plots periods for all stations in edipath and the plot is interactive, just click on the period you want to select and it will appear in the console, it will also be saved to lp.plst. To sort this list type

```
lp.plst.sort()
```

The x's mark a conformation that the station contains that period. So when looking for the best periods to invert for look for a dense line of x's

3 imaging

3.1 mtplottools

A collection of functions taht enable the plotting and/or visualisation of data for all MT analysis steps.

3.1.1 plotcoh (Function)

plot coherence output from birrp_bbconvert. If you want to save the plot do so using the interactive gui.

3.1.2 plotResPhase (Function)

plot the apparent resistivity and phase for TE and TM modes from a .dat file produced by writedat. If you want to save the plot use the save button on top left.

3.1.3 resPhasePlots (Function)

plot multiple responses given full path filenames. Can input key word list dictionary if parameters for each plot are different or just single values.

3.1.4 plotTSplotTipper (Function)

plot the real and imaginary induction arrow from EDI file.
ToDo !!! void function !!!

3.1.5 plotTS (Function)

plot timeseries that have been combined using combineFewFiles. combinefilelst is the output list of filenames

3.1.6 plotPTpseudoSection (Function)

plot a pseudo section of phase tensor ellipses given a list of full path filenames.

3.1.7 plotRTpseudoSection (Function)

plot a pseudo section of resistivity tensor ellipses given a list of full path filenames. (Weckmann et al. 2002)

3.1.8 plotPTMaps (Function)

Plots phase tensor ellipses in map view from a list of edifiles with full path.

3.1.9 plotResPhasePseudoSection (Function)

plots a pseudo section from a list of edifiles with full path with descending frequency or ascending period on the y axis and relative position on the x.

3.1.10 plotRTMaps (Function)

plots phase tensor ellipses in map view from a list of edifiles with full path.

3.1.11 plotRoseStrikeAngles (Function)

plots the strike angle as determined by phase tensor azimuth (Caldwell et al. [2004]) and invariants of the impedance tensor (Weaver et al. [2003]).

The data is split into decades where the histogram for each is plotted in the form of a rose diagram with a range of 0 to 180 degrees. Where 0 is North and 90 is East. The median angle of the period band is set in polar diagram. The top row is the strike estimated from the invariants of the impedance tensor. The bottom row is the azimuth estimated from the phase tensor. If tipper is 'y' then the 3rd row is the strike determined from the tipper, which is orthogonal to the induction arrow direction.

3.2 ptplots

Ploitting functions and routines in connection to the phase tensor

3.2.1 ptplots (Class)

Generate a PhaseTensorPlot object. Various visualisations can be done from this.

Plots are:

- plotPhaseTensor(save='n',fmt='pdf',fignum=1,thetar=0)
  plots the phase tensor as ellipses with long axis directed towards electrical strike and short axis directed across magnetic strike. Spaces the ellipses across the period axis.
- plotStrikeAngle(save='n',fmt='pdf',fignum=1,thetar=0)
  -plots the electric strike angle as a function of period with error bars.
- plotMinMaxPhase(save='n',fmt='pdf',fignum=1,thetar=0)
  plots the min and max phase angle as a function of period with error bars.
- plotAzimuth(save='n',fmt='pdf',fignum=1,thetar=0)
  plots the azimuth angle as a function of period with error bars.
- plotSkew(save='n',fmt='pdf',fignum=1,thetar=0)
  plots the skew angle as a function of period with error bars.
- plotElliticity(save='n',fmt='pdf',fignum=1,thetar=0)
  plots the ellipticity as a function of period with error bars.
- plotAll(save='n',fmt='pdf',fignum=1,thetar=0)
  plots phase tensor, strike angle, min and max phase angle, azimuth, skew, and ellipticity as subplots on one plot.

You can save the plots by making save='y' or a path you input. fmt is the fmt you want to save the figure as. Can be: pdf,eps,ps,png or svg. Fignum is the figure number in the event you want to plot multipl plots."""

plotPhaseTensor (Method)

plot phase tensor ellipses

plotStrikeangle (Method)

Plot the strike angle as calculated from the invariants

plotMinMaxPhase (Method)

Plot the minimum and maximum phase of phase tensor

plotAzimuth (Method)

plot the azimuth of maximum phase

plotSkew (Method)

Plot the skew angle

plotEllipticity (Method)

Plot the ellipticity as $\frac{\Phi_{max}-\Phi_{min}}{\Phi_{max}+\Phi_{min}}$

plotAll (Method)

plot phase tensor, strike angle, min and max phase angle, azimuth, skew, and ellipticity as subplots on one plot

plotResPhase (Method)

plot the resistivity and phase for all impedance tensor polarizations. 2 plots, one containing xy, yx- polarizations, the other xx, yy

4 modeling

4.1 modemtools

Collection of functions for dealing with ModEM. So far there are conversions from Ws3Dinv models and data into ModEM3D model and data. Furthermore, WingLink-generated model meshes can be converted.

4.1.1 winglinkmesh2modelfile (Function)

Convert a WingLink mesh geometry into ModEM modelfile (so far only homogeneous halfspace model supported)

4.1.2 latlon2xy (Function)

UNUSED

4.1.3 edis2datafile (Function)

Generate ModEM datafile from a combination of a WingLink model .out-file and a list of EDI files

4.1.4 generate_edilist (Function)

browse a folder for existing EDI files. Sensitive to file suffix 'EDI',/'edi'

4.1.5 winglink2modem (Function)

Conversion of WingLink output files into ModEM input. Wrapper fro 2 functions, which deal with the model- and data-file respectively.

4.1.6 wsinv2modem_data (Function)

Convert an existing input data file from Weerachai's wsinv style into Egbert's ModEM type

4.1.7 wsinv2modem_model (Function)

Convert an existing input model file from Weerachai's wsinv style into Egbert's ModEM type

4.1.8 plotmodel3d (Function)

ToDo !!! void function !!!

4.1.9 getmeshblockcoordinates (Function)

Read a ModEM-style model file and return a list of 3 lists, which again contain the X/Y/Z coordinate of a mesh block

4.2 occamtools

Tools for OCCAM handling

4.2.1 Occam1D (Class)

==========
This class will deal with everything occam 1D
==========

make1DdataFile (Method)

write a data file for Occam1D

make1DModelFile (Method)

Makes a 1D model file for Occam1D.

make1DInputFile (Method)

Make a 1D input file for Occam 1D

read1DModelFile (Method)

read in model 1D file

read1DInputFile (Method)

reads in a 1D input file

read1DdataFile (Method)

reads a 1D data file

read1DIterFile (Method)

read an 1D iteration file

read1DRespFile (Method)

read response file

plot1D (Method)

Plots the results of a 1D inversion. The left plot is the response and the right hand plot is the model as a function of depth.

plotL2Curve (Method)

Plot the L curve for RMS vs Iteration and RMS vs Roughness.

4.2.2 getdatetime (Function)

calls current time: `time.asctime(time.gmtime())`

4.2.3 makestartfiles (Function)

returns start files (MeshF,ModF,SF)

4.2.4 writemeshfile (Function)

.

4.2.5 writemodelfile (Function)

.

4.2.6 writestartupfile (Function)

.

4.2.7 read_datafile (Function)

#RELYING ON A CONSTANT FORMAT, ACCESSING THE PARTS BY COUNTING OF LINES!!!:

4.2.8 get_model_setup (Function)

??

4.2.9 blocks_elements_setup (Function)

??

4.2.10 OccamPointPicker (Class)

This class helps the user interactively pick points to mask and add error bars.
Usage:
———-
To mask just a single point right click over the point and a gray point will appear indicating it has been masked

To mask both the apparent resistivity and phase left click over the point. Gray points will appear over both the apparent resistivity and phase. Sometimes the points don't exactly matchup, haven't quite worked that bug out yet, but not to worry it picks out the correct points

To add error bars to a point click the middle or scroll bar button. This only adds error bars to the point and does not reduce them so start out with reasonable errorbars. You can change the increment that the error bars are increased with reserrinc and phaseerrinc.

inAxes (Method)

gets the axes that the mouse is currently in.

inFigure (Method)

gets the figure number that the mouse is in

on_close (Method)

close the figure with a 'q' key event and disconnect the event ids

4.2.11 Occam2DData (Class)

Occam2DData covers all aspects of dealing with data for an Occam 2D inversion using the code of Constable et al. [1987] and deGroot-Hedlin and Constable [1990] from Scripps avaliable at `http://marineemlab.ucsd.edu/Projects/Occam/2DMT/index.html`

make2DdataFile (Method)

Make a data file that Occam can read. At the moment the inversion line is the best fit line through all the stations used for the inversion.

read2DdataFile (Method)

read2DdataFile will read in data from a 2D occam data file. Only supports the first 6 data types of occam2D

rewrite2DdataFile (Method)

rewrite2DDataFile will rewrite an existing data file so you can redefine some of the parameters, such as rotation angle, or errors for the different components or only invert for one mode or add one or add tipper or remove tipper.

plotMaskPoints (Method)

An interactive plotting tool to mask points an add errorbars

maskPoints (Method)

maskPoints will take in points found from plotMaskPoints and rewrite the data file to nameRW.dat. **Be sure to run plotMaskPoints first**

read2DRespFile (Method)

read2DRespFile will read in a response file and combine the data with info from the data file.

plot2DResponses (Method)

plotResponse will plot the responses modeled from winglink against the observed data.

plotPseudoSection (Method)

plots a pseudo section of the data and response if input.

plotAllResponses (Method)

Plot all the responses of occam inversion from data file. This assumes the response curves are in the same folder as the datafile.

4.2.12 Occam2DModel (Class)

This class deals with the model side of Occam inversions, including plotting the model, the L-curve, depth profiles. It will also be able to build a mesh and regularization grid at some point.

It inherits Occam2DData and the data can be extracted from the method get2DData(). After this call you can use all the methods of Occam2DData, such as plotting the model responses and pseudo sections.

read2DIter (Method)

read an iteration file and combine that info from the datafn and return a dictionary of variables.

read2DInmodel (Method)

read an INMODEL file for occam 2D

read2DMesh (Method)

reads an Occam 2D mesh file

get2DData (Method)

get data from data file using the inherited :func:'read2DdataFile'

get2DModel (Method)

create an array based on the FE mesh and fill the values found from the regularization grid. This way the array can be manipulated as a 2D object and plotted as an image or a mesh.

plot2DModel (Method)

plot the model output by occam in the iteration file.

plotL2Curve (Method)

plot the RMS vs iteration number for the given inversion folder and roughness vs iteration number

plotDepthModel (Method)

Plots a depth section profile for a given set of stations.

4.2.13 compare2DIter (Function)

take the difference between two iteration and make a difference iter file

4.3 winglinktools

Collection of tools for dealing with output from Windows WingLink software

4.3.1 readOutputFile (Function)

read an output file from winglink and output data in the form of a dictionary.

4.3.2 plotResponses (Function)

plot the responses modeled from winglink against the observed data.

4.3.3 readModelFile (Function)

read the XYZ .txt-file output by Winglink.
(profile directions restricted to NS or EW !!!)

4.3.4 readWLOutFile (Function)

read .out file from winglink

4.3.5 readSitesFile (Function)

read sites file output from winglink

4.3.6 readSitesFile2 (Function)

read sites file output from winglink - alternative
more consistent output!

4.3.7 getXY (Function)

get x (e-w) and y (n-s) position of station and put in middle of cell
connection x-East and y-North here !!!! (ToDo change this)

4.3.8 getmeshblockcoordinates (Function)

(slightly different from ModEM mesh block coordinates!!) return a list of 3 lists, which again contain the X/Y/Z coordinate of a mesh block

Orientation is X-North, Y-East, Z-Down. Horizontal origin is in the center of the mesh, Indexing starts at the lower left (SouthWest) corner

4.4 ws3dtools

Handling Weerachai's Ws3Dinv code

4.4.1 ListPeriods (Class)

class to pick periods

connect (Method)

?

onclick (Method)

?

disconnect (Method)

?

4.4.2 writeWSDataFile (Function)

writes a data file for WSINV3D from winglink outputs

4.4.3 writeInit3DFile (Function)

Makes an init3d file for WSINV3D

4.4.4 writeStartupFile (Function)

makes a startup file for WSINV3D

4.4.5 readDataFile (Function)

read in Ws3Dinv data file

4.4.6 plotDataResPhase (Function)

plot responses from the data file and if there is a response file

4.4.7 plotTensorMaps (Function)

plot phase tensor maps for data and or response, each figure is of a different period. If response is input a third column is added which is the residual phase tensor showing where the model is not fitting the data well. The data are plotted in km in units of ohm-m.

4.4.8 readModelFile (Function)

read in a Ws3Dinv model file

5 processing

5.1 birrptools

Tools for dealing with the (Fortran) processing software BIRRP

5.1.1 finishBeep (Function)

!!! void function !!! (commented, since Windows based tool)

5.1.2 readProDict (Function)

read in the information from processing infofile and output as a list of dictionaries.

5.1.3 scriptfilePrep (Function)

calculate parameters, combine files and convert if need be.

5.1.4 writeScriptfile (Function)

will write a script file for BIRRP using info in processingdict which is a dictionary

5.1.5 callBIRRP (Function)

call BIRRP from a command window and run the script file provided

5.1.6 convertBIRRPoutputs (Function)

take the outputs of BIRRP and manipulate them into .edi, .dat, .coh, .imp files as well as generate plots of apparent resistivity and phase and coherence.

5.1.7 plotBFfiles (Function)

plot the apparent resisitivty and phase calculated from edifile as 2 seperate plots for the different polarizations. It will plot the coherency between components and if input will plot the time series.

5.1.8 writeLogfile (Function)

write a log file of how a station was processed

5.1.9 runBIRRPpp (Function)

processes a station from start to finish on seperate processors

5.1.10 sigfigs (Function)

return a string with the proper amount of significant digits for the input number, can be str or float.

5.1.11 sil (Function)

split a single line written in an .ini file for burpinterface and return the list of strings.

5.1.12 read2c2 (Function)

read in a .2c2 file output from BIRRP and return a list containing [period],[freq],[coh],[zcoh]. Note if any of the coherences are negative a value of 0 will be given to them.

5.1.13 writecoh (Function)

write a coherence file using the BIRRP outputs residing in the dirpath folder. The output file is tab delimited and if any values are negative they are put to 0. Each value has 7 significant digits. Returns written to filename.

5.1.14 readcoh (Function)

read a coherence file writen by writecoh. The output is period,frequency,coh1,zcoh1,coh2,zcoh2,coh3,zcoh3

5.1.15 bbcalfunc (Function)

generate a function fitting the calibration data in bbfile to the frequencies in nfreqlst using a cubic interpolation algorithm. The output is the real and imaginary functions as a function of the nfreqlst in units of microV/nT

5.1.16 readj (Function)

read in the .j file output by BIRRP, which is better than reading in the .irj.rf files

5.1.17 readrf (Function)

read in .irj.rf file output by BIRRP that reside in the folder nominated by dirpath. Returns: period, freq, z[ijreal,ijimag,ijvar] as array. If any of the numbers are NaN or +Inf they are set to zero

5.1.18 bbconvz (Function)

convert the .rf output files of BIRRP into correct units using broadband calibrations given by file bbcalfile that has format log(freq),breal,bimag. dlgain is the data logger gain (verylow= 2.5,low=1,high=.1) The output is period,freq,z[[zijr,ziji,zije]]. Note that it assumes the efields are already converted.

5.1.19 lpconvz (Function)

Convert the magnetic field from counts to units of microV/nT. bfield is a list of numbers. dlain is amount of gain applied by data logger(verylow=2.5,low=1, high=.1)

5.1.20 writeimp (Function)

writes a tab delimited .imp file of converted impedances from.rf outputs of BIRRP and calibrates using file bbfile and dlgain which is the data logger gain (verylow=2.5,low=1,high=.1). Returns written to filename.

5.1.21 readimp (Function)

read in the impedances from a .imp file written by writeimp. the output is ofil,period,z[zr,zi]

5.1.22 writedat (Function)

write a .dat (resistivity and phase) file from .rf output files from birrp after converting the broadband magnetic channels. dirpath is where the .rf files reside, bb file is where the conversion file resides and df is the sampling frequency. Returns written to filename.

5.1.23 readdat (Function)

read in a .dat file written by writedat and output ofil, period,[resistivity,resistivityerr],[phase,phaseerr]

5.1.24 writeedi (Function)

write an .edi file for a station processed by BIRRP given the station info file, station and bbfile if applicable. Returns the full path to the .edi file

5.1.25 readini (Function)

read in an inifile and return a dictionary of initial parameters for the burpinterface program.

5.1.26 writeini (Function)

write an .ini file from the to the filepath as station.ini. The argsdict must be len 40 or will not write, the variables should be: dict[defdirpath,station,magtype,lpyn,eyn,mcomps,magdec, df,cacherate, dlength,dlgain,egain,lpbzcor,bbcal,magori,birrploc,ilev,nout, ninp,tbw,nfft,nsctmax,uin,ainuin,c2t nlev,nar,imode,jmode,nfil,complstr,thetae,thetab,thetaf].

5.2 runbirrpsinglestation (Script)

Program to process a single station using information from files.

If you use the optimized birrp.exe and you get an error saying: didn't find all the .bf files, something a miss. Try running using the old version. Something in the optimization compiler changes something in the variable useage in the Fortran program.

At the end I've added a plot section, if you want to save the plots change the n to y, and hopefully that will work, not sure about windows 7. If that doesn't work you can leave it at n and save the plot when it comes up by clicking on the save icon on the plot and save manually.

It the rotation angles are not correct or you want to change birrp parameters change it in the processinginfofile and rerun the program.

5.3 striketools

Tools for STRIKE software (?) McNeice, G. W.and Jones, A. G., 2001, Multisite, multifrequency tensor decomposition of magnetotelluric data: *Geophysics*, **66**, pg. 158–173.

All functionality is within one class

5.3.1 Strike (Class)

Strike class ... ???

readDCMP (Method)

read in the output file .dcmp

plotHistogram (Method)

plot the histogram of the different angles

plotAngles (Method)

plot the angles vs. log(period)

plotResPhase (Method)

plot the regional resistivity and phase

5.4 tftools

Collection of functions for Time-Frequency-Analysis of MT (raw-)data

5.4.1 padzeros (Function)

return a function that is padded with zeros to the next power of 2

5.4.2 sfilter (Function)

apply a sinc filter of width w to the function f by multipling in the frequency domain

5.4.3 dctrend (Function)

remove a dc trend from the function

5.4.4 normalizeL2 (Function)

normalizeL2(f) will return the function f normalized by the L2 norm

5.4.5 decimatef (Function)

Will decimate a function by the factor m. First an 8th order Cheybechev type I filter with a cuttoff frequency of .8/m is applied in both directions to minimize any phase distortion and remove any aliasing. Note decimation values above 10 will typically result in bad coefficients, therefore if you decimation is more than 10 just repeat the decimation until the desired decimation is reached.

5.4.6 dwindow (Function)

Calculates the derivative of the given window

5.4.7 gausswin (Function)

gausswin will compute a gaussian window of length winlen with a variance

5.4.8 wvdas (Function)

compute the analytic signal for WVVD as defined by J. M. O' Toole, M. Mesbah, and B. Boashash, (2008), "A New Discrete Analytic Signal for Reducing Aliasing in the Discrete Wigner-Ville Distribution", IEEE Trans. on Signal Processing,

5.4.9 stft (Function)

calculate the spectrogam of the given function by calculating the fft of a window of length nh at each time instance with an interval of tstep. The frequency resolution is nfbins Can compute the cross STFT by inputting fx as [fx1,fx2]

5.4.10 reassignedstft (Function)

compute the reassigned spectrogram by estimating the center of gravity of the signal and condensing dispersed energy back to that location.

5.4.11 wvd (Function)

calculate the Wigner-Ville distribution for a function f. Can compute the cross spectra by inputting fx as [fx1,fx2]

5.4.12 spwvd (Function)

calculate the smoothed pseudo Wigner-Ville distribution for a function fx. smoothed with Gaussians windows to get best localization.

5.4.13 robustwvd (Function)

calculate the smoothed pseudo Wigner-Ville distribution for a function fx. smoothed with Gaussians windows to get best localization.

5.4.14 specwv (Function)

calculate the Wigner-Ville distribution mulitplied by the STFT windowed by the common gaussian window h for a function f.

5.4.15 modifiedb (Function)

calculate the modified b distribution as defined by $\cosh(n)^{-}2$ beta for a function fx.

5.4.16 robuststftMedian (Function)

output an array of the time-frequency robust spectrogram calculated using the vector median simplification.

5.4.17 robuststftL (Function)

output an array of the time-frequency robust spectrogram by estimating the vector median and summing terms estimated by alpha coefficients.

5.4.18 smethod (Function)

calculate the smethod by estimating the STFT first and computing the WV of window length L in the frequency domain. For larger L more of WV estimation, if L=0 get back STFT

5.4.19 robustSmethod (Function)

computes the robust Smethod via the robust spectrogram.

5.4.20 reassignedSmethod (Function)

calulate the reassigned S-method as described by Djurovic[1999] by using the spectrogram to estimate the reassignment

5.4.21 plottf (Function)

plot a calculated tfarray

5.4.22 plotAll (Function)

plot a calculated tfarray with limits corresponding to tlst and flst.

5.4.23 stfbss (Function)

estimates sources using a blind source algorithm based on spatial time-frequency distributions. At the moment this algorithm uses the SPWVD to estimate TF distributions.

6 utils

6.1 combineedis (Script)

combine two edi files into one a number of files

6.2 csvutm (Script)

Convert and add columns for different coordinate systems to a CSV file.

This script requires pyproj installed. If you have a CSV file with two columns containing x and y coordinates in some coordinate system (the "from" system), this script will add another two columns with the coordinates transformed into another system (the "to" system). The CSV file must have a header row, and the column names should be specified on the command line as well.

6.2.1 csvutm (Function)

...

6.2.2 get_parser (Function)

...

6.3 edidms2deg (Script)

??????????

6.3.1 dms2deg (Function)

...

6.3.2 deg2dms (Function)

...

6.3.3 check_format (Function)

...

6.3.4 edidms2deg (Function)

converts the LAT/LON information within given EDI files from/to the degree/deg:min:sec format.

Call it with (a list of) files as arguments. Wildcards are allowed. The repective lines within the files are converted and the files written to the current working directory. The names are not changed, so existing files get overwritten!!

6.4 latlongutmconversion (Script)

Lat Long - UTM, UTM - Lat Long conversions

6.4.1 LLtoUTM (Function)

converts lat/long to UTM coords. Equations from USGS Bulletin 1532 East Longitudes are positive, West longitudes are negative. North latitudes are positive, South latitudes are negative Lat and Long are in decimal degrees Written by Chuck Gantz- chuck.gantz@globalstar.com

6.4.2 UTMtoLL (Function)

converts UTM coords to lat/long. Equations from USGS Bulletin 1532 East Longitudes are positive, West longitudes are negative. North latitudes are positive, South latitudes are negative Lat and Long are in decimal degrees. Written by Chuck Gantz- chuck.gantz@globalstar.com Converted to Python by Russ Nelson ¡nelson@crynwr.com¿

6.5 modem2vtk3d

Build VTK files for visualising ModEM3D modelfile and station locations

Orientation convention:

coordinate system NED is used! First component is positive from South to North, second component is positve from West to East, third component is positive Downwards

6.5.1 model2vtkgrid (Function)

Convert ModEM output files (model and responses) into 3D VTK resistivity grid

6.5.2 data2vtkstationsgrid (Function)

Convert ModEM data file into 2D VTK station set (unstructured grid)

6.6 pakasc2TSasc (Script)

Convert the generic ascii tables generated by Pak2Asc (from e-logger data) into 5-column data asciis (time in epochs and 4 channels)

6.6.1 pakascii2TSascii (Function)

the conversion function, called by a wrapper

6.7 runparalanamt (Script)

process a single station using information from files.

If you use the optimized birrp.exe and you get an error saying: didn't find all the .bf files, something a miss. Try running using the old version. Something in the optimization compiler changes something in the variable useage in the Fortran program.

There is a beep when it finishes and can be loud depending on the computer volume.

At the end I've added a plot section, if you want to save the plots change the n to y, and hopefully that will work, not sure about windows 7. If that doesn't work you can leave it at n and save the plot when it comes up by clicking on the save icon on the plot and save manually.

It the rotation angles are not correct or you want to change birrp parameters change it in the processinginfofile and rerun the program.

Good luck

Jared Peacock 2011

6.8 ws2para (Script)

Convert ws3Dinv output files (model and responses) into 3D VTK resistivity grid and unstructured VTKGrid containing station locations.