ChaRM python package

Version 0.1.0

Mohammad Maysami

Seismic Laboratory for Imaging and Modeling (SLIM)
Department of Earth and Ocean Sciences(EOSC)
University of British Columbia (UBC)

December 2007

Description

Charm is a python package to implement recent studies on ChaRM project (Characterization of reflectors and modeling). It is implementing the new trace-based algorithm we proposed in our recent abstract (Maysami and Herrmann, 2007 [1])

License

You may use this code only under the conditions and terms of the license contained in the file *LICENSE* provided with this source code. If you do not agree to these terms you may not use this software.

Prerequisites

- Python-2.4 or newer *(http://www.python.org)*
- NumPy-1.0 or newer *(http://numpy.scipy.org)*
- SciPy-0.5.1 or newer *(http://www.scipy.org)*
- Matplotlib-0.90.1 or newer *(http://matplotlib.sourceforge.net)*
- MADAGASCAR-2817 *(SVN developer tree at http://rsf.sourceforge.net)*

  Note: Compile with API=python

  Note: It is optional and only required to run SConstruct script.

Installation

There are two easy installation method for this package. Considering permissions limits for non-administrator users, the first one is advised since it use this package in the place you have it copied.

- Just put the package folder (Charm) into your `$PYTHONPATH`. For example, if you are using *BASH* and `$Charm` is the path to your Charm folder(excluding the folder itself), then type:

[1] M. Maysami and F.J. Herrmann, Seismic reflector characterization by a multiscale detection-estimation method, accepted for EAGE 69th Conference & Exhibition, London.

```
export PYTHONPATH=$PYTHONPATH:$Charm
```

For *CShell* type:

```
setenv PYTHONPATH $PYTHONPATH:$Charm
```

- Alternatively, you may use setup.py to install the package in your default python path location, enter this command:

  ```
  python setup.py install
  ```

  To install it into a specific folder type this, where /path/to/mypython is a folder which is in your `$PYTHONPATH` environment.

- Set environment variables(paths) if they are different than default values. These variables are *CHARM_Data,CHARM_Results,CHARM_pydata,CHARM_Demos*. Otherwise it will be set to defaults which are folders with the same name (Data, Results, and etc.) in `$Charm` as mentioned above.

  For example, if you are using *BASH* and `$DataPATH` is the path where you have input rsf data,then type:

  ```
  export CHARM_Data=$DataPATH
  ```

  For *CShell* type:

  ```
  setenv CHARM_Data $DataPATH
  ```

- One of the global variables, `_df_input`, in `__init__.py` points to default input rsf file for some of the functions of this package. For your convenience, it is better to be set to a sample rsf file in order to skip declaring it in input argument

Structure

Note that there are many parameters in this method that give more control over characterization process. However, to make it simple I have only included key parameters in function calls and the rest are set in the codes. So, if you need to have more control over settings,l you can access them inside `Main.py` file header.

Folders:

The package consists of some folders as below:

Core It holds main modules of package. Each module has special purposed functions inside. Here is a short list of exiting files in *Core* folder and their functionality.

- `__init__.py` : Initializes the import command in python
- `Misc.py` : Miscellaneous functions for use in other files.
- `API.py` : Python interface to RSF for exchanging data.
- `Synthesize.py` : Module for generating synthetic data using fractional splines or Gaussian manifolds.
- `Cwt.py` : Module for continuous wavelet transform and other wavelet analysis.
- `Window.py` : Module for different window functions to be used in segmentation of events in seismic trace.
- `Manifold.py` : Module for generating Gaussian manifolds and handling them.
- `Steps.py` : Includes functions to do different steps of characterization as detection, segmentation, and estimation.
- `Main.py` : Includes characterization function which implements our new method for input seismic trace.
- `Show.py` : Carries functions for plotting results.

Data This folder is default folder for searching input files unless `CHARM_Data` is set as explained in installation section.

Results This folder is default folder for storing outputs of characterization unless `CHARM_Results` is set as explained in installation section.

Doc It carries documentation files for the package.

pydata This is where generated python data files by package are stored for future use unless `CHARM_pydata` is set as explained in installation section.

Demos Any saved figure through using this package will sits in this folder unless `CHARM_Demos` is set as explained in installation section.It also contains a simple demo to show how the main steps are used.

Files:

Charm folder also contains some other files as below:

`__init__.py` initializes the import command in python and sets global variables for the package.

`setup.py` helps user to install package as explained before.

`README` quick guide user

`LICENSE` license agreement information

`GNU` GNU general public license

`SConstruct` *scons* script which runs `sfchar.py`. Input and output arguments has to be set as explained in the files comments.

`sfchar.py` a wrapper for function *char*(responsible for the analysis of seismic traces for characterization and extracting attributes) in `Main.py` file to be called with standard I/O in Unix terminal as explained later in *Using package* section. This will implement the method on 2-D seismic sections as input.

`sfshow.py` responsible for plotting *rsf* files from Unix terminal with python imaging library. It is able to plot either ordinary 2-D *rsf* files or the 3-D *rsf* files generated by `sfchar.py`. See the file header for more details about syntax and arguments.

Key variables in Python environment

This section explains important variables that are passed in the output dictionary of characterization function (*char* in `Main.py`) which is responsible for full analysis of seismic traces. Remind that this function is also able to generate synthetic data. For more information check `Main.py` file. The variables dealing with trace and generated trace with estimation are as below:

`N` : length of trace

`trace` : seismic trace as a row vector

`trace_det` : trace formed by superposition of windowed events

`trace_est` : trace formed by superposition of estimations of each event

`err` : array consisting of estimation errors for each event.

`events` : a matrix whose rows are events in seismic trace.

`events_det`: a matrix whose rows are segmented events of seismic trace.

`masks` : a matrix whose rows has corresponding mask used to segment each event.

Attribute matrices:

Variables with the form of attrib_xxx are matrices of attributes of events found in the seismic trace. Each row of matrix has attributes (location,scale, singularity order, and instantaneous phase) of one events in trace.

`attrib_org` : original attributes for synthetic trace.

`attrib_det` : attributes after detection step.

`attrib_est` : attributes after estimation step.

Variables with the form of attrib_vect are matrices, where number of columns is equal to length of seismic trace. The rows show actual values of amplitude, scale, singularity order, and phase components at location of events and set to null elsewhere.

`attrib_vect` : Original attributes in case of synthetic data.

`attrib_vect_est` : Estimated attributes.

Python data files (pyd extension):

These are generated python data files by package(*char* function) stored for future reload.

`synt_model.pyd` : Contains a dictionary of parameters for synthetic seismic data.

`real_model.pyd` : Contains real seismic trace.

`waves .pyd` : Contains a dictionary of source wavelet and CWT parameters.

`attrib_det.pyd` : Contains matrix of attributes after detection step.

`attrib_est.pyd` : Contains matrix of attributes after estimation step.

`events_det .pyd` : A matrix of traces, each containing one windowed event.

`events_est.pyd` : A matrix of traces, each containing one estimated event.

`trace_est.pyd` : Contains superposition of estimated events which should be similar to actual trace.

For more information about these data files refer to `pydataFiles.txt`

How to use package

Python:

Go into your python interpreter. Type "import Charm" at your python command prompt. As a demo, you may run following command in python after importing the package.

```
Charm.char(type='new',user=1)
```

You may also try more detailed things while using *detect, segment, and BFGSestimate* functions from the package. All these functions are well-documented and easy to follow. To get help for any function you can use *help(name of function)*.

Unix terminal with standard input and output:

Use following commands in terminal to analyze seismic data and show results.

```
./sfchar.py <input.rsf >output.rsf args=...

./sfshow.py <output.rsf args=...
```

Arguments are optional and will be set to default if not provided. For more details about arguments check the scripts header for documentation.

Mohammad Maysami
http://www.eos.ubc.ca/∼mmaysami
Seismic Laboratory for Imaging and Modeling (SLIM)
University of British Columbia (UBC)