

Python Scripts for NIED continuous waveform data requesting and processing

- Author: Dongdong Tian @ USTC
- Update: 2014-12-27

This is a collection of scripts to request, download and process continuous waveform data available from [NIED Hi-net](#) website.

It does not come with any warranties, nor is it guaranteed to work on your computer. The user assumes full responsibility for the use of all scripts. The author is **NOT** responsible for any damage that may follow from correct *or* incorrect use of these scripts.

Dependency

- Python 3.4 (Not work under Python 2; Not Tested under Python 3.3)
- Python third-party modules
 - [requests](#)
 - [clint](#)
 - [docopt](#)
- Hinet [win32tools](#): `catwin32` and `win2sac_32`

How to get

If you use git, just clone this repo to your work directory:

```
git clone https://github.com/seisman/HinetScripts.git
```

You can update this repo using:

```
git pull
```

If you do not use git, just click the “Download ZIP” button on the right.

Before you use it

1. Register on the [NIED Hi-net](#) website, so you can access to NIED waveform data;

2. Download [win32tools](#) and compile them, make sure binary `catwin32` and `win2sac_32` are in your PATH;
3. Request, download and process data manually at least one time, make sure that you know the whole procedures and limitations of NIED website;
4. Modify configure file `Hinet.cfg` to your needs:
 - **User & Password**
 - **Net** : Network code to request waveform data as default
 - **Maxspan**: Maximum record length allowed for one web request
 - **catwin32**: Path to `catwin32` supplied by win32tools
4. Run `HinetDoctor.py` to check your configure file;

What is network code?

Each network is represented by a network code. For example, Hi-net network has a code of '0101', while V-net '0105'. You can see the full code list by run `python HinetContRequest.py -h`.

What is Maxspan? And how to choose it?

NIED Hi-net website set a limitation of data size in one request:

1. Record Length < 60 min
2. Number of channels * Record Length <= 12000 min

Just take Hi-net as example, Hi-net network has about 800 station and 24000 channels. According to the limitations, the record length should be no more than 5 minutes long in one web request. So the Maxspan, allowed maximum record length, should be no more than 5 for Hi-net network with all stations selected.

The request script `HinetContRequest.py` helps you break through the limitation. Using this script, you can request datas with a much longer record length, this script will split the request into multiple sub-request, each has a record length no more than **Maxspan** minutes.

Scripts

HinetDoctor.py

`HinetDoctor.py` helps you check your configure file, you should run it when you modify `Hinet.cfg`.

1. Is username and password correct?
2. Has Hi-net website been updated?
3. Is catwin32 command in you path and executable?
4. How many station are selected for Hi-net and F-net?
5. Is Maxspan in allowed range?

HinetContRequest.py

HinetContRequest.py is used to request and download data from Hi-net server.

Usage

```
$ python HinetContRequest.py -h
Request continuous waveform data from NIED Hi-net.
```

Usage:

```
HinetContRequest.py <year> <month> <day> <hour> <min> <span> [options]
HinetContRequest.py -h
```

Options:

```
-h, --help          Show this help.
-c CODE --code=CODE  Select code for organization and network.
-m SPAN --maxspan=SPAN Max time span for sub-requests
-d DIR --directory=DIR Output directory. Default: current directory.
-o FILE --output=FILE Output filename.
                        Default: CODE_YYYYMMDDHHMM_SPAN.cnt
-t FILE --ctable=FILE Channel table filename. Default: CODE_YYYYMMDD.ch
```

Examples

1. Request data of Hi-net start from 2010-10-01T15:00:00 (JST) with duration of 20 minutes

```
python HinetContRequest.py 2010 10 01 15 00 20
```

2. Request data of F-net start from 2010-10-01T15:00:00 (JST) with duration of 20 minutes

```
python HinetContRequest.py 2010 10 01 15 00 20 -c 0103
```

3. Request data of Hi-net, with customized output directory and filename

```
python HinetContRequest.py 2010 10 01 15 00 20 -d aaa -o aaa.cnt -t aaa.ch
```

4. Request data of Hi-net, use default filename and customized output directory. (**Highly Recommended**)

```
python HinetContRequest.py 2010 10 01 15 00 20 -d 201010010600
```

If you run `HinetContRequest.py` in the highly recommender way, you will get a directory `201010010600` with two file inside: `0101_201010011500_20.cnt` and `0101_20101001.ch`.

```
|-- 201010010600
   |-- 0101_201010011500_20.cnt
   `-- 0101_20101001.ch
```

rdhinet.py

`rdhinet.py` is used to extract SAC files from WIN32 file.

Usage

Extract SAC data files from NIED Hi-net WIN32 files

Usage:

```
rdhinet.py DIRNAME [-C <comps>] [-D <outdir>] [-S <suffix>] [-P <procs>]
rdhinet.py -h
```

Options:

```
-h          Show this help.
-C <comps>  Components to extract, delimited using commas.
             Aavailable components are U, N, E, X, Y et. al.
             Default to extract all components.
-D <outdir> Output directory for SAC files.
-S <suffix> Suffix of output SAC files. Default: no suffix.
-P <procs>  Parallel using multiple processes.
             Set number of CPUs to <procs> if <procs> equals 0. [default: 0]
```

Examples

1. Extract all channels

```
python rdhinet.py 201010010600
```

2. Extract NEU components with suffix 'SAC'

```
python rdhinet.py 201010010600 -C U,N,E -S SAC
```

In most cases, what you need is only `-C` option.

If you run `python rdhinet.py 201010010600 -C U`, you will get SAC files looks like `N.FRNH.U` under directory `201010010600`.

ch2pz

`ch2pz.py` is used to extract SAC PZ files from Channel Table file.

Usage

```
$ python ch2pz.py -h
```

Convert NIED Hi-net Channel Table file to SAC PZ files

Usage:

```
ch2pz.py DIRNAME [-C <comps>] [-D <outdir>] [-S <suffix>]
```

Options:

<code>-C <comps></code>	Channel Components to convert. Choose from U,N,E,X,Y et. al. Default to convert all components.
<code>-D <outdir></code>	Output directory of SAC PZ files. Use the directory of Channel Table file as default.
<code>-S <suffix></code>	Suffix for SAC PZ files. [default: SAC_PZ]

Examples

1. Extract all channels

```
python ch2pz.py 201010010600
```

2. Extract NEU components

```
python ch2pz.py 201010010600 -C U,N,E
```

In most cases, what you need is only `-C` option.

If you run `python ch2pz.py 201010010600 -C U`, you will get SAC PoleZero files looks like `N.FRNH.U.SAC_PZ` under directory `201010010600`.