Ba*It*: an Iterative Baer picker
User Manual
`v2.1.6`

Matteo Bagagli – ETH, Zurich

March 26, 2019

Contents

1 Introduction

Picking algorithm have always been a foundamental part in the seismological routine workflows of observatories , and more often they have been such a painful step in the workflow for tuning and theri semi–automatic procedure as normally a cross-check by human eye is required. This steps, though, cannot be completely erased at the moment and neither this program show

2 Installation

The code has been developed in Python–3 and is distributed under the *GPL-v3* license guidelines. A number of dependecies are requested and are listed in the `requirements.txt`. In order to avoid major issues and libraries conflicts it's strongly recommended to use a *virtual environment*; for this task you could either use conda or pip–virtualenv as your preference. To semplify this passage, a file is provided under the `config` folder (`bait_env.yml`), that you could use to generate the environment by typing:

```
$ cd /where/the/package/is
$ conda env create -f ./config/bait_env.yml
```

Strictly speaking, the installation part is trivial: once the environment has been set-up, all you need to do is to open a terminal and type:

```
$ cd /where/the/package/is
$ pip intall .
```

Once the package is successfully installed, you could verify the software integrity by tiping:

```
$ cd /where/the/package/is
$ pytest
```

Once all the test are passed, you could start to use the software without any problems.

3 The code

The package here presented is composed by 3 different modules:

- `bait.py`: this module contains all the main **Ba***It* class and can be considered the *core* of this package.
- `bait_customtests.py`: this module contains all the test-functions aimed at the pick validation and could be expanded with customizable test by the user (see Sect.4).
- `bait_plots.py`: this module contains the necessary function to plot the waveforms and relative **Ba***It* charachteristic functions and picks.
- `bait_errors.py`: in this modules are contained various type of errors raised by the software. This module can be expanded with additional errors if the user require so (see Sect.4).

3.1 Main flow

The workflow of the picker is schematized in Fig.**??**. At the actual stage the **Ba***It* picking algorithm is able to detect properly the P-first arrival times and could detect sometimes the secondary as well. For each waveform the user can decide to apply a processing step (higly recommended before picking stage) or turn it off. This picking approach consist in apply 2 two different parameters setup of the same picker: if the first call doesn't return a pick, the waveform is discarded and no picks are found, while instead a pick is found an evaluation on the pick begins. If the evaluation is positive the pick is accepted as a first arrival and the program switch to the next waveform. If the picks is negatively evaluated (rejected) a secondary setup on the trimmed trace is called (*auxiliary picker*). From this point, if a pick is found it will be evaluated and eventually accepted, otherwise the waveform will be discarded and no first arrival will be assigned. If, the secondary picks are rejected, then the loop of trimming–picking–evaluating will continue until the maximum iteration number is reached.

$$CF_i = \left| \frac{2 \cdot (wave_i - min(wave))}{max(wave) - min(wave)} - 1 \right| \tag{1}$$

3.2 Picks evaluation stage

The evaluation stage is done over a carachteristic function (CF) that The evaluation part is composed of 3 steps. The first one is to create a charachteristic function of the waveform as described in Eq.1 over which some user–customized tests will be performed to legitimate the pick. The two tests performed to avoid mispicks in this dataset are represented by the two conditions described in Eq.2 and Eq.3.

$$max(W_1) \geqslant PAR_1 \tag{2}$$

$$mean(W_n) \geqslant PAR_2 \cdot mean(W_0) \tag{3}$$

4 Customization

This software has been developed by an user that doesn't like the frustation of every one user to one other user still. With this general idea in mind, the user

5 Examples