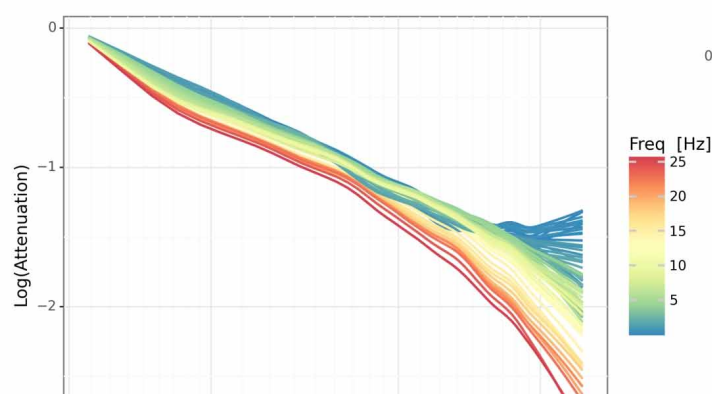
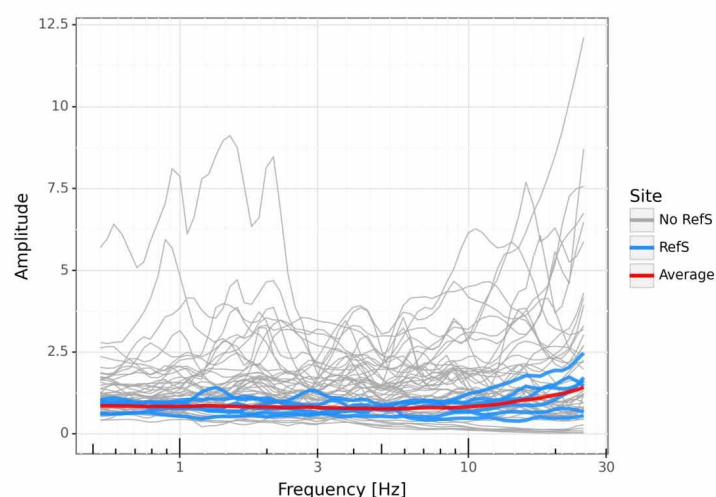
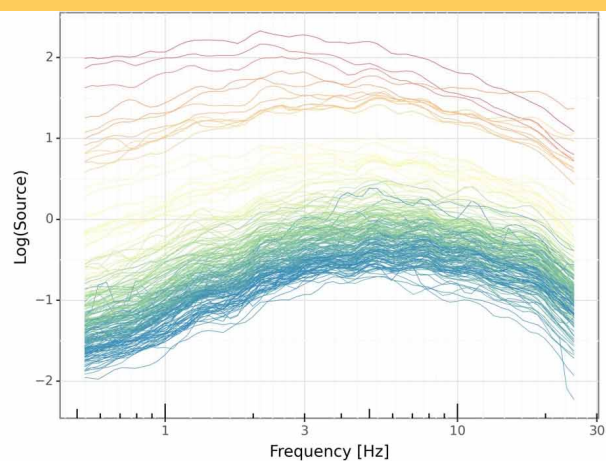


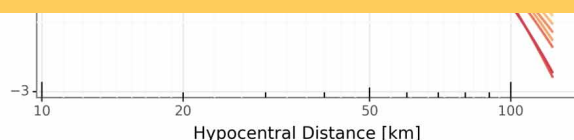


RAPPORTI TECNICI INGV

User Manual of GITpy: a Python-based
tool for the Generalized
Inversion Technique



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA



494

Direttore Responsabile

Daniela VERSACE

Editor in Chief

Milena MORETTI (editorinchief.collane-editoriali@ingv.it)

Editorial Board

Laura ALFONSI (laura.alfonsi@ingv.it)
Christian BIGNAMI (christian.bignami@ingv.it)
Simona CARANNANTE (simona.carannante@ingv.it)
Viviana CASTELLI (viviana.castelli@ingv.it)
Luca COCCHI (luca.cocchi@ingv.it)
Luigi CUCCI (luigi.cucci@ingv.it)
Lorenzo CUGLIARI (lorenzo.cugliari@ingv.it)
Alessia DI CAPRIO (alessia.dicaprio@ingv.it)
Roberto DI MARTINO (roberto.dimartino@ingv.it)
Domenico DI MAURO (domenico.dimauro@ingv.it)
Domenico DORONZO (domenico.doronzio@ingv.it)
Filippo GRECO (filippo.greco@ingv.it)
Alessandro IAROCCI (alessandro.iarocci@ingv.it)
Mario MATTIA (mario.mattia@ingv.it)
Daniele MELINI (daniele.melini@ingv.it)
Anna NARDI (anna.nardi@ingv.it)
Lucia NARDONE (lucia.nardone@ingv.it)
Marco OLIVIERI (marco.olivieri@ingv.it)
Pierangelo ROMANO (pierangelo.romano@ingv.it)
Maurizio SOLDANI (maurizio.soldani@ingv.it)
Sara STOPPONI (sara.stopponi@ingv.it)
Umberto TAMMARO (umberto.tammaro@ingv.it)
Andrea TERTULLIANI (andrea.tertulliani@ingv.it)
Stefano URBINI (stefano.urbini@ingv.it)

Ufficio Editoriale

Francesca DI STEFANO Coordinatore - Segreteria di Redazione

Produzione e grafica-redazionale

Barbara ANGIONI
Massimiliano CASCONI
Francesca DI STEFANO
Patrizia PANTANI

REGISTRAZIONE AL TRIBUNALE DI ROMA N.174 | 2014, 23 LUGLIO

© 2014 INGV Istituto Nazionale di Geofisica e Vulcanologia | Rappresentante legale: Carlo DOGLIONI

Sede: Via di Vigna Murata, 605 | Roma



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

RAPPORTI TECNICI INGV

User Manual of GITpy: a Python-based tool for the Generalized Inversion Technique

Maria D'Amico^{*,1}, Paola Morasca², Daniele Spallarossa³, Dino Bindi⁴, Matteo Picozzi⁵, Lorenzo Vitrano²

¹INGV | Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Catania - Osservatorio Etneo, Italy

²INGV | Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Milano, Italy

³Università di Genova, Dipartimento di Scienze della Terra dell'Ambiente e della Vita, Genova, Italy

⁴GFZ | German Research Centre for Geosciences, Potsdam, Germany

⁵OGS | Istituto Nazionale di Oceanografia e di Geofisica Sperimentale, Udine, Italy

**Corresponding author*

Accepted January 31, 2025 | *Accettato 31 gennaio 2025*

How to cite | *Come citare* D'Amico M., Morasca P., Spallarossa D., Bindi D., Picozzi M., Vitrano L., (2025). User Manual of GITpy: a Python-based tool for the Generalized Inversion Technique. Rapp. Tec. INGV, 494: 1-46, <https://doi.org/10.13127/rpt/494>

Cover Graphical outputs of GITpy: examples of source spectra, non-parametric attenuation curves at different frequencies, and transfer functions at various recording sites | *In copertina Output grafici di GITpy: esempi di spettri di sorgente, curve di attenuazione non parametrica a diverse frequenze e funzioni di trasferimento a vari siti di registrazione*

INDEX

Abstract	7
Introduction	7
1. Overview of the GIT technique	7
2. Installation	8
2.1 Instructions for Linux Users	8
2.1.1 Installing Python3 and required Python3 libraries (pip and venv)	9
2.1.2 Download the development version of GITpy	10
2.1.3 Creating a Virtual Environment	11
2.2 Instructions for Windows Users	11
3. Getting Started	14
3.1 GITeps.py	16
3.1.1 Configuration files for GITeps.py	16
3.1.2 Run Inversion module	19
3.1.3 Output of GITeps.py	20
3.2 CorrectFas.py	21
3.2.1 Configuration file for CorrectFas.py	21
3.2.2 Run CorrectFas.py	22
3.2.3 Output of CorrectFas.py	22
3.3 FitAttenuation.py	23
3.3.1 Configuration file for FitAttenuation.py	25
3.3.2 Run attenuation modeling module	27
3.3.3 Output of FitAttenuation.py	27
3.4 FitSource.py	28
3.4.1 Configuration file for module FitSource.py	31
3.4.2 Run source modeling module	33
3.4.3 Output of FitSource.py	34
3.4.3.1 Fit median source spectra derived from GITeps.py	34
3.4.3.2 Fit apparent source spectra derived from CorrectFas.py	35
4. Tutorial Guide for New Users	36
4.1 Example Dataset	36
4.2 Example Jupyter Python Notebooks	38
5. Conclusions	40
License	40
Citation	40
Getting Help	41
Disclaimer	41

Acknowledgements	41
Bibliography	41
Sitography	43

Abstract

GITpy is an open-source Python software package designed with an object-oriented architecture, implementing the Generalized Inversion Technique (GIT) to separate the source, propagation, and site effects of the ground motion from the Fourier Amplitude Spectrum (FAS). GITpy includes modules to perform both the non-parametric spectral decomposition and to fit the resulting source spectra and spectral attenuation with standard seismological models. Although the GIT is a well-established approach this tool offers a number of possibilities such as:

1. Streamline the modeling of attenuation and source properties by using configuration files or interactive workflows, enabling quick testing of various models;
2. Offer the flexibility to select from multiple levels of complexity for attenuation modeling, including options for geometrical spreading and anelastic attenuation;
3. Provide a choice of source spectrum modeling approaches, such as adopting either a homogeneous or heterogeneous crustal model or specifying the frequency range for model fitting;
4. Compute the apparent source spectra by removing from the input FAS the site effects and the non-parametric attenuation obtained from the inversion itself. The apparent source spectra can be modeled to derive key parameters useful for directivity studies, as well as for a rapid estimation of source parameters in case of a strong event;
5. Provide additional source parameters including the radiated energy, the apparent stress and the radiation efficiency together with seismic moment, corner frequency and stress drop. The additional parameters can be estimated for both the best-fit parametric source models and the non-parametric GIT results.

Here we present the documentation to help the Users to install and run the GITpy software package by means of an example dataset (GITpy_db) and Jupyter Notebooks tutorials.

Keywords Generalized Inversion Technique; Fourier Amplitude Spectra; Open-source software

Introduction

GITpy is an open-source software package developed in Python for applying the Generalized Inversion Technique (GIT) to datasets of Fourier Amplitude Spectra (FAS). It consists of four distinct modules to derive a suite of region-specific spectral parameters (i.e. source, attenuation, and site parameters), through an inversion scheme to separate the different contributions and applying subsequent modeling tools. GITpy is the result of a collaborative project between scientists from the Istituto Nazionale di Geofisica e Vulcanologia, Italy (INGV), German Research Centre for Geosciences, Germany (GFZ), University of Genoa, Italy (UNIGE), National Institute of Oceanography and Applied Geophysics (OGS).

1. Overview of the GIT technique

The GIT is a well-established method for separating source, propagation, and site terms from the FAS of recorded P- or S-waves [Andrews 1986; Castro et al., 1990; Boatwright et al., 1991; Oth et al., 2011; Bindi et al., 2020; 2023a; 2023b]. Two strategies can be used to isolate these terms: the parametric and non-parametric approaches. In the parametric approach [Drouet et al., 2008],

source and attenuation are fitted using standard seismological models, whereas the non-parametric approach [Oth et al., 2011] solves the equation system without imposing a-priori parametric models on the individual terms. The GITpy software implements the non-parametric approach. After the inversion process separates the source, propagation, and site contributions, the resulting attenuation and source non-parametric curves are fitted using standard seismological models to derive seismic parameters. This a posteriori fitting offers the advantage of allowing the testing of different parametric source models without the need for multiple inversions. GITpy includes two separate modules for modeling source and attenuation terms, providing users with the flexibility to make different choices. Future releases of GITpy may incorporate methods for alternative source and attenuation models. Further details are provided below.

2. Installation

This tutorial provides step-by-step instructions for installing the GITpy software, outlining its essential dependencies and how to manage them within a Python 3 virtual environment, applicable to both Linux and Windows platforms. GITpy has been tested on Linux (Linux Mint 20.1) and Windows (Windows 11 Pro). The initial step for both operating systems is to obtain the source code for a specific release of GITpy from the GitLab repository [1]. Table 1 lists the required dependencies.

Element	Version
Python	3.9.
numpy	1.24.4
scipy	1.13.0
pandas	2.2.2
plotnine	0.13.6
setuptools	69.5.1
pyyaml	6.0.1
obspy	1.4.1
openpyxl	3.1.2
lmfit	1.2.2

Table 1 GITpy requirements.

2.1 Instructions for Linux Users

The instructions in this tutorial assume that you are using a GNU/Linux distribution based on Ubuntu or Mint. For Debian and other distributions based on the apt package manager you may have to login as root instead of using *sudo*.

2.1.1 Installing Python3 and required Python3 libraries (pip and venv)

- Python3

Before you go any further, make sure you have *Python3* installed and that it is available from your command line. As it is usually already included by default on many GNU/Linux distributions, you can check this by running the following command at a shell prompt:

```
python3 --version
```

You should get some output with the exact version installed (e.g. v3.9). If you do not have Python3, you can install it using the following commands:

1. Update the package list using the following command:

```
sudo apt update
```

2. Use the following command to install Python3:

```
sudo apt install python3
```

- pip3

Next, make sure you have *pip3* available. Chances are it is already installed, you can check this by running the following command at a shell prompt:

```
pip3 --version
```

and you should get an output text with the exact version of the pip3 package installed on your system. If not, you can install pip3 as follows:

1. Update the package list using the following command (not needed if you just did it in the previous step):

```
sudo apt update
```

2. Use the following command to install pip3:

```
sudo apt install python3-pip
```

- venv

Now, make sure you have the Python3 *venv* module available, you can check this by running the following command at a shell prompt:

```
python3 -m venv
```

and you should get an output text with a brief usage synopsis. If not, you can install the *venv* module as follows:

1. Update the package list using the following command (not needed if you just did it in one of the previous steps):

```
sudo apt update
```

2. Use the following command to install the `venv` module for Python3:

```
sudo apt install python3-venv
```

2.1.2 Download the development version of GITpy

You can create a local copy of the latest development version of GITpy by cloning the `git` repository or download the package from the project main web page and extract it to any directory you like. If you choose to use `git`, you can check if it is already installed:

```
git --version
```

or install it the usual way:

```
sudo apt install git
```

From a terminal window, change to the local directory where you want to clone your repository and run:

```
git clone git@gitlab.rm.ingv.it:inversion/gitpy.git
```

or

```
git clone https://gitlab.rm.ingv.it/inversion/gitpy.git
```

If the clone was successful, a new sub-directory with the same name of the GitLab repository, appears on your local drive, in the same directory where you cloned your repository. To check how many branches of the GITpy development are available in the GitLab repository run:

```
git branch
```

a list like this will appear:

```
(gitpy) user@MyComputer:~/gitpy$ git branch
GITpy
* main
```

to switch to another branch use:

```
git checkout GITpy
followed by
git status
```

to verify that you are on the desired branch of GITpy development.

2.1.3 Creating a Virtual Environment

GITpy was developed using python libraries that are not included in the standard installation of Python 3.9 such as *numpy*, *scipy*, or *pandas* (see Table 1). To manage dependencies avoiding conflict due to different version installed on own Operating System you can create a “Python virtual environment” specifying the target directory (absolute or relative to the current directory) with the ground motion processing repository:

```
python3 -m venv /path/to/new/virtual/environment
```

For example:

```
python3 -m venv /path/to/gitpy/
```

To activate the Virtual Environment

```
cd /path/to/gitpy/
```

and run

```
source bin/activate
```

You should get a command line like this:

```
(gitpy) user@MyComputer:/gitpy$
```

At this point you are ready to install the GITpy requirements with pip3 only in the virtual environment:

```
pip3 install --upgrade pip
```

```
pip3 install obspy (this will also install numpy, matplotlib and scipy)
```

```
pip3 install pandas plotnine setuptools pyyaml openpyxl lmfit PyQt5 tornado
```

```
sudo apt install python3-tk
```

You should be now ready to use the GITpy software!

You can “exit” the virtual environment by deactivating it with the following command:

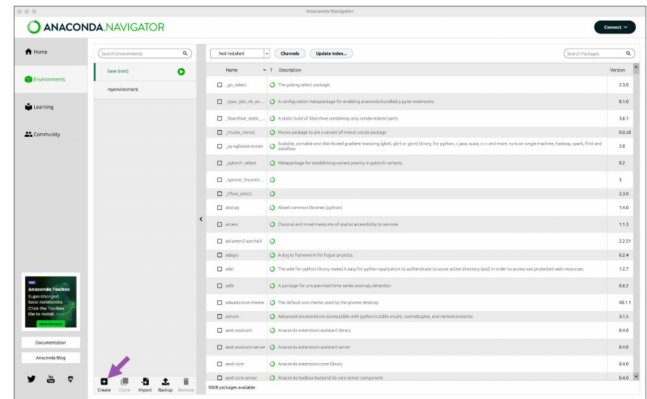
```
deactivate
```

2.2 Instructions for Windows Users

There are several ways to run Python scripts like GITpy on Windows 11, but the recommended procedure is to use Conda or Anaconda Navigator, to easily manage dependencies and create an isolated environment to install and run GITpy. In the following we describe the procedure for using Anaconda Navigator (Navigator enables you to use Conda to manage environments and packages, but in a graphical application instead of a command line interface).

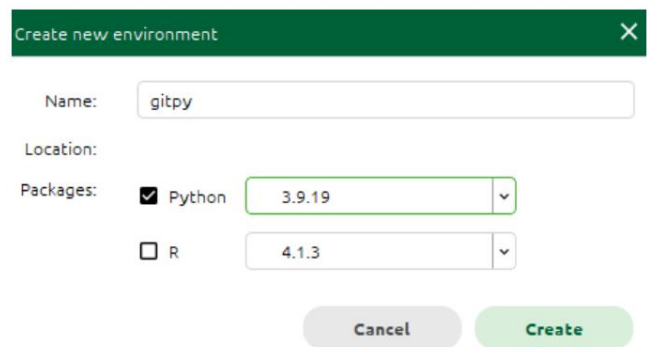
1. Install Anaconda by following the instructions provided here [2];
2. From the **Start** menu, search for “Anaconda Navigator” and click to open. The home page (on top left corner) shows the contents of the default environment “base (root)”. Now you need to create a new environment to install the GITpy code (Figure 1);
3. On the Environment page (on the left), click **Create** (Figure 1).

Figure 1 Screenshot of the Anaconda navigator.



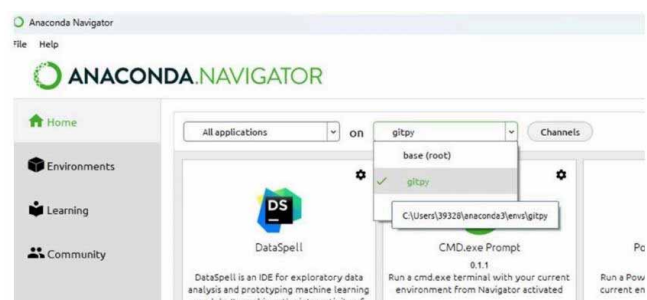
4. Give a name to the new environment (for example “gitpy”), click on Python and select version 3.9. Now you have a new environment “gitpy” with the correct Python version installed to run GITpy (Figure 2).

Figure 2 Example on how to create a new environment.



5. From the home page, switch to the new “gitpy” environment by clicking on its name (Figure 3).

Figure 3 Example of selection of the virtual environment created to run GITpy.



6. Go back to the **Environments** page and check that you have the “conda-forge” channel, if not you can add it:
 - click on the **Channels** button;
 - then click the **Add** button;
 - enter the channel URL [3];
 - press **Enter** on your keyboard;
 - click on the **Update channels** button.
7. From the Environments page check the installed packages versions (Figure 4). Make sure you have all the required packages and the correct versions, as listed in Requirements Table 1.

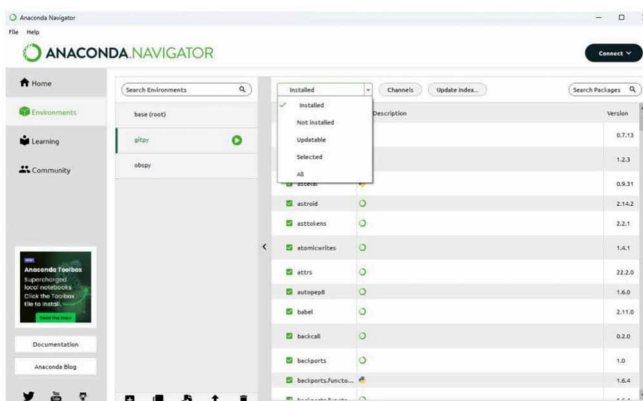


Figure 4 Example of Environment page of Anaconda navigator.

If some packages are not the correct version or they are missing at all, follow the instructions here described [4]

8. Go back to the Home page, you should now be in the new “gitpy” environment (check). Among the default applications there is “PowerShell Prompt “. Launch it and you are ready to run GITpy from the command line (Figure 5).

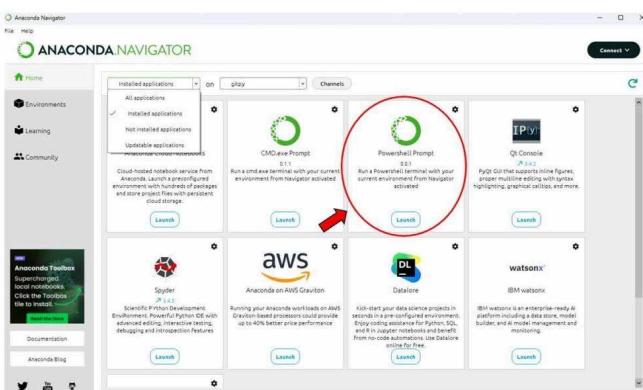


Figure 5 Example of the home page of the Anaconda navigator to select the Powershell Prompt.

9. Download GITpy from Gitlab repository at [5], save and unzip it in your work directory. Then from the PowerShell Prompt window go into your work directory to run the python codes (Figure 5).

3. Getting Started

GITpy is organized into four primary modules (Table 2), located within the “gitpy-main” code folder, along with a Python library called LibGit (Table 3). Users may interact with the main GITpy modules by modifying the configuration files. These files are structured in YAML format and contain all necessary configuration settings to run the various modules, along with optional input files. Users should not alter the structure of these files or the names of the key fields in the first column; only the values in the second column are intended to be modified. A detailed description of the input, execution, and output for the modules GITeps.py, CorrectFas.py, FitAttenuation.py, and FitSource.py is provided below.

Python modules	Description
GITeps.py	Separates the source, propagation and site contributions from the Fourier Amplitude Spectra (FAS) of the recorded S-waves; the equation system is solved without imposing any a-priori parametric models on the different terms
CorrectFas.py	Corrects the recorded spectra (FAS) for the attenuation and site effects, using the GIT spectral decomposition outputs, providing the apparent source spectra
FitAttenuation.py	Performs fitting of non-parametric spectral attenuation curves using standard seismological models in which the geometrical spreading and the anelastic attenuation contributions are isolated and described in terms of parametric spectral models depending on distance, quality factor, and kappa
FitSource.py	Performs fitting of source spectra in terms of seismic moment and corner frequency, and allows to derive other seismological parameters (e.g. stress drop, radiated energy)

Table 2 Summary of the four main GITpy modules.

Python modules	Description
Attenuation.py	Python class to manage input/output related to the attenuation
AttParametric.py	Python class to set the spectral model to fit the non-parametric attenuation curves
Brune.py	Python class to calculate source spectra using the ω^2 Brune's models

ConfGITeps.py	Python class to read the configuration file with parameters and paths to optional input files required to run the seismological data inversion (GITeps.py)
ConfAttFit.py	Python class to read the configuration file with parameters and paths to optional input files required to run the fitting of non-parametric spectral attenuation curves (FitAttenuation.py)
ConfCorrectFas.py	Python class to read the configuration file with paths to GIT spectral decomposition outputs required to run the apparent source spectra (CorrectFas.py)
ConfSourceFit.py	Python class to read the configuration file with parameters and paths to optional input files required to run the fitting of the source term (FitSource.py)
EvtGit.py	Python class to read/write event metadata related to source spectra
FASInv.py	Python class to to separates source, propagation, and site contributions from the Fourier Amplitude Spectra (FAS) of recorded S-waves; the equation system is solved without imposing any a-priori parametric models on the different terms
format_utils.py	Utility to manage the GITpy outputs
GeomSpread.py	Python class to fit the geometrical spreading
GitResults.py	Python class to manage outputs of the seismological data inversion
InputFas.py	Python class to read the FAS
QualityFactor.py	Python class to parametrize the frequency-dependent quality factor model
RegKappa.py	Python class to parametrize the exponential terms with kappa related to the anelastic attenuation model
Source.py	Python class to fit the non-parametric source spectra
Sources.py	Python class to manage input/output related to source spectra
StatGit.py	Python class to read/write station metadata
Stations.py	Python class to manage the site term
utilities.py	GITpy utilities

Table 3 GITpy Library.

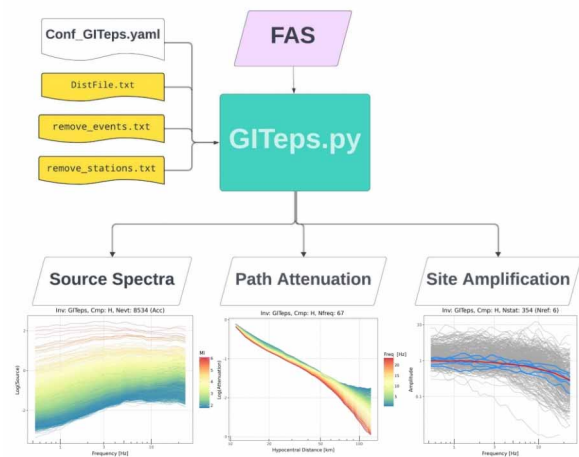
3.1 GITeps.py

The GITeps.py module implements the non-parametric GIT to derive source spectra, path attenuation, and site amplification factors (Figure 6). The GIT is based on the convolutional assumption that the FAS_{ij} of earthquake i recorded at stations j can be expressed as the product of three terms representing the source (S), propagation (P), and site (Z) contributions to the ground shaking in the spectral domain:

$$FAS_{ij}(R_{ij}, f) = S_i(f) * P(R_{ij}, f) * Z_j(f) \quad (1)$$

where R_{ij} is the hypocentral distance and f is the frequency. By analyzing each frequency separately, equation 1 produces an overdetermined linear system that can be solved in a least-squares sense. The linear system is characterized by two unresolved degrees of freedom, meaning the solution is not unique. To resolve the trade-offs between the three terms, two a priori constraints are necessary. A common choice is to constrain the average logarithm of all site amplifications to zero or to impose that the average amplification of a set of predefined reference stations (typically installed on bedrock) is zero. The second constraint typically involves the attenuation term, where a reference distance is defined at which the logarithm of attenuation equals zero, independent of frequency.

Figure 6 Flowchart of the GITpy module that separates the source, propagation and site contributions from the FAS of the recorded S-waves (GITeps.py).



3.1.1 Configuration files for GITeps.py

The main sections of the YAML configuration file (Conf_GITeps.yaml) are organized into blocks as follows: GLOBAL contains general settings; DIST BIN specifies information about distance discretization; SUBSET provides options for removing seismic events and stations from the original input dataset; CONSTRAINT DREF defines the characteristics of the reference distance; CONSTRAINTS SITES includes details on the constraints for reference sites; CONSTRAINT SMOOTH DIST describes the smoothing condition applied to attenuation functions; and LSQR INVERSION contains parameters related to the inversion process. A detailed description is provided in Table 4, with an example shown in Figure 7. The YAML configuration file also includes three fields (dist_file, event_remove, and station_remove) within the SUBSET block, which require the reading of external files containing information about the distance bins to be used during FAS inversion, as well as seismic events and stations to be excluded from the input ground motion dataset. It is important to note that these options should be used with caution, as they are primarily intended for removing outliers, not for generating modified datasets from the original data.

Compiling the FAS dataset for use with the GIT is a crucial step and should not be delegated to the inversion module. To solve the overdetermined linear system of equations (1) using least squares, the lsqr method from SciPy [6] is employed, which is specifically designed for solving sparse linear algebra problems.

```

1  GLOBAL:
2  job_name: "GIT CI"
3  fas_dir: "C:/Users/39328/INGV/PROGETTI/GIT_Python_in_locale/CI/FAS_Dataset/"
4  out_dir: "C:/Users/39328/INGV/PROGETTI/GIT_Python_in_locale/CI/test1/"
5  gui_env: ['Qt5Agg', 'TkAgg', 'nbAgg', 'WebAgg']
6  cmp: "H"
7  plot_graph: True
8  save_graph: True
9  mean: "VECTORIAL"
10 fmin: 0.5
11 fmax: 26
12 mmin: 1.5
13 mmax: 6.5
14
15 DIST BIN:
16 type_bin: LIN
17 dist_min: 5
18 dist_max: 125
19 dist_step: 2
20 nb_in_dist: 200
21 dist_file: "C:/Users/39328/INGV/PROGETTI/GIT_Python_in_locale/CI/DistFile.txt"
22
23 SUBSET:
24 type_sel: FILE
25 event_remove: "C:/Users/39328/INGV/PROGETTI/GIT_Python_in_locale/CI/remove_events.txt"
26 station_remove: "C:/Users/39328/INGV/PROGETTI/GIT_Python_in_locale/CI/remove_stations.txt"
27
28 CONSTRAINT DREF:
29 dref: 10
30 weight_dref: 5
31 vdref: 1
32
33 CONSTRAINTS SITES:
34 SITES: IT.LSS., IT.MNF., IT.SIO., IT.SNO., IT.SDM., IT.NRN..
35 weight_site: 5
36 Knot: 0.1
37 FileCrust: Na
38 Fk: 10
39
40 CONSTRAINTS SMOOTH DIST:
41 weight_smo: 25
42
43 LSQR INVERSION:
44 nrow_min: 100
45 damp: 1e-06
46 atol: 1e-08
47 btol: 1e-08
48 iter_lim: 1000

```

Figure 7 Example of configuration file for module GITeps.py.

Block	Field	Description
GLOBAL	job_name	The User can associate an arbitrary name to his analysis
	fas_dir	Path to Directory containing the FAS files
	out_dir	Output directory
	gui_env	List of interactive backend that can be used for displaying Matplotlib figures. Only ['Qt5Agg','TkAgg','nbAgg','WebAgg'] have been tested in GITpy. To run GITeps.py in Jupyter Notebook set gui_env: Na
	cmp	Component used for the inversion: "H": horizontal components; "Z": Vertical component
	plot_graph	"True": shows figures during the code execution; "False": avoids the figures visualization
	save_graph	"True" saves figures in the output directory; "False" does not save figures
	mean	Option "VECTORIAL" calculates the vectorial component on the two H components ($\sqrt{EW^2 + NS^2}$); option "GEOM" calculates the geometrical mean of the two H components
	fmin	Min frequency to consider for the analysis
	fmax	Max frequency to consider for the analysis
	mmin	Min magnitude to consider for the analysis
	mmax	Max magnitude to consider for the analysis

DIST BIN	type_bin	Option "LIN": Creates a distance vector composed by a linear distribution of values between "dist_min" and "dist_max" with a step as given in the variable "dist_step"; Option "LOG": Creates a distance vector composed by a logarithmic distribution of values between "dist_min" and "dist_max" for a given number of bins defined by the variable "nbin_dist"; Option "FILE": Uses an existing file with user defined distance vector whose name is set in the variable "dist_file"
	dist_min	Min distance for type_bin = LIN or LOG
	dist_max	Max distance for type_bin = LIN or LOG
	dist_step	Step in km between values (used for type_bin = LIN)
	nbin_dist	Number of bins (used for the case type_bin = LOG)
	dist_file	Distance filename including path with a user defined distance vector (see figure 8 C). used only when type_bin = FILE
SUBSET	type_sel	Option "FILE": to remove from inversion a list of events from a file indicated in "event_remove" and/or a list of stations from a file indicated in "station_remove". Option "NONE" to keep all data for the inversion (no files are required, "event_remove" and "station_remove" are ignored).
	event_remove	File name (including path) with a list of events to be excluded from the analysis (see figure 8 A); set to "NONE" to keep all events when "type_sel" is set to "FILE".
	station_remove	File name (including path) with a list of stations to be excluded from the analysis (see figure 8 B); set to "NONE" to keep all stations when "type_sel" is set to "FILE".
CONSTRAINT DREF	dref	Reference distance [km] for which attenuation functions at all frequencies is set to <i>vdref</i>
	weight_dref	Weight of attenuation constraint at reference distance <i>dref</i>
	vdref	Attenuation amplitude at all frequencies in correspondence of <i>dref</i>
CONSTRAINTS SITES	SITES	List of site names to be used as reference ground motion. Set this variable "ALL" to consider the average over all stations = 0
	weight_site	Weight for site condition
	Knot	High_freq decay kappa value [s]; set to <i>Na</i> to skip <i>Knot</i>
	Fk	Lower frequency bound [Hz] to use above which to estimate kappa; set to <i>Na</i> to skip <i>F_k</i>
CONSTRAINTS SMOOTH DIST	weight_smo	Weight for smoothing condition on attenuation functions, if dynamic smoothing, max. weight will be WSMOOTH, min can be set in functions themselves, currently on WSMOOTH/4
LSQR INVERSION	nrow_min	Minimum number of lines for each FAS file
	damp	Damping coefficient. The default is 0.
	atol	Stopping tolerances. <i>lsqr</i> continues iterations until a certain backward error estimate is smaller than some quantity depending on <i>atol</i> and <i>btol</i> .

	btol	Stopping tolerances. <i>lsqr</i> continues iterations until a certain backward error estimate is smaller than some quantity depending on atol and <i>btol</i> .
	iter_lim	Explicit limitation on number of iterations (for safety)

Table 4 Description of the key fields included in the configuration file of GITeps.py; for further details about the *lsqr* inversion see the *scipy* documentation [6].

Figure 8 shows three text files. File A, 'remove_events.txt', contains a header 'NUM;EVENTS' followed by four lines of event data. File B, 'remove_stations.txt', contains a header 'NUM;STATIONS' followed by two lines of station data. File C, 'DistFile.txt', contains a single column of distance values from 1 to 27.5 in increments of 1 or 1.5.

Figure 8 A) file “remove_events.txt” requires the first line including the variable names (NUM and EVENTS separated by “;”). The first column represents a progressive number, the second column shows the event ID. The end line should not be empty. B) file “remove_stations.txt” requires the first line including the variable names (NUM and STATIONS separated by “;”). The first column represents a progressive number, the second column shows the station ID (network_code.location_code.station_code). The end line should not be empty. C) The distance file is composed by a single column reporting the User defined distances in increasing order. All “.txt” file names can be arbitrarily modified by the User.

3.1.2 Run Inversion module

The User can launch the inversion by executing python from the command line as:

```
python $code_path/GITeps.py --config $your_path/conf_file_inversion.yaml --log logfile
```

Example:

```
python ./GITeps.py --config Conf_GITeps.yaml --log log_inversion.log
```

The description of the options useful to compose the command line is reported in Table 5.

Options	Description	Default	Required
config	configuration file in YAML format to set the Generalized Inversion of seismological records	na	True
log	log file in TXT format	na	True
v	Verbose mode (prints summary and other details)	True	

Table 5 List of options to run GITeps.py.

3.2 CorrectFas.py

The CorrectFas.py module provides the Fourier spectra of an earthquake as received by the recording stations around the source (apparent source spectra). The apparent source spectra are calculated by subtracting from the recorded FAS the non-parametric attenuation and the site amplification factors derived by GITeps.py module (Figure 11). The resulting apparent source spectra can be used as input for another GITpy module, FitSource.py, to analyze ground motion variability with respect to azimuthal and distance dependence. Additionally, CorrectFas.py can be employed for rapid estimation of source parameters following earthquakes that are not included in the ground motion dataset used to define the path attenuation and site amplification properties for a specific region.

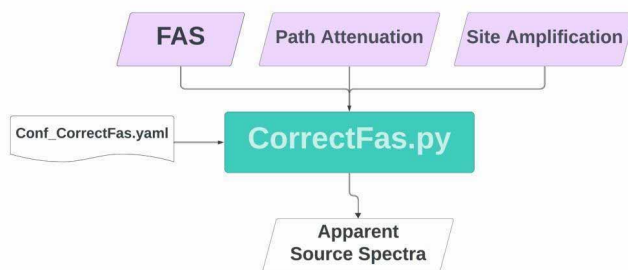


Figure 11 Flowchart of the GITpy module (CorrectFAS.py) that isolates the Fourier spectra radiated from the source and observed at each recording station.

3.2.1 Configuration file for CorrectFas.py

The YAML configuration file for module CorrectFas.py has only one block called GLOBAL that includes the general settings. Detailed descriptions are given in Table 6 and configuration file examples in Figure 12.

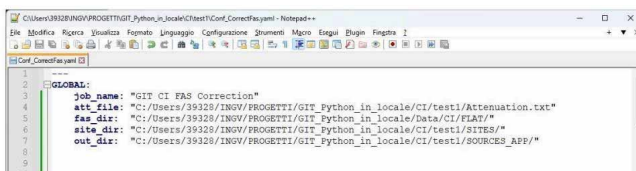


Figure 12 Example of configuration file for module CorrectFas.py

Block	Field	Description
GLOBAL	job_name	The User can associate an arbitrary name to his analysis
	att_file	File .txt containing the non-parametric attenuation curves (output of GITeps.py)
	fas_dir	Directory containing the FAS files
	site_dir	Directory containing the site amplification curves (output of GITeps.py)
	out_dir	Directory to save the CorrectFas.py outputs

Table 6 Description of the key fields included in the yaml configuration file of CorrectFas.py.

3.2.2 Run CorrectFas.py

The module `CorrectFas.py` correcting the recorded spectra (FAS) for the attenuation and site effects, using the GIT spectral decomposition outputs, provides the apparent source spectra. The command to execute this module is:

```
python $code_path/CorrectFas.py --config $your_path/Conf_CorrectFas.yaml --log
$your_path/logfile.log
```

Example:

```
python ./CorrectFas.py --config Conf_CorrectFas.yaml --log logCorrFAS.log
```

The description of the options useful to compose the command line to run `CorrectFas.py` is reported in Table 7.

Options	Description	Default	Required
config	configuration file in YAML format to execute the <code>CorrectFas.py</code> module	na	True
log	log file in TXT format	na	True
v	Verbose mode (prints summary and other details)	True	

Table 7 List of options to run `CorrectFas.py`.

3.2.3 Output of CorrectFas.py

The `CorrectFas.py` module generates two files for each event: `.app_source` and `.source` (Figure 13). These files are stored in the directory specified by the user in the configuration file (see the `out_dir` variable in Table 7 and the example in Figure 14). Both files have the same header, which includes general information such as the module name (`CorrectFas.py`), the component used for the analysis, the event ID, event magnitude from the FAS files, as well as latitude, longitude, and depth. The `.app_source` file (Figure 14A) contains all the apparent source spectra, one for each station. It is organized in matrix format, starting from line 2, where the column names are indicated. The first column, labeled “FDSN”, shows the station ID (`network_code.location_code.station_code`), the second column, “Dist”, indicates the hypocentral distance, and the third (“Az”) and fourth (“Bz”) columns represent the azimuth and back-azimuth, respectively. Starting from the fifth column, all frequencies are listed. To extract the spectrum for a specific station, the user should select the row corresponding to the station ID of interest. The amplitudes are expressed in cm/s. The `.source` file (Figure 14B) contains the median spectrum for the event and consists of four columns: frequency [Hz], log10 of the median amplitude, standard deviation, and the number of amplitudes used to compute the median value. Files `.source` can be distinguished from those produced by the `GITeps.py` module by examining the first field in the header of the first line.

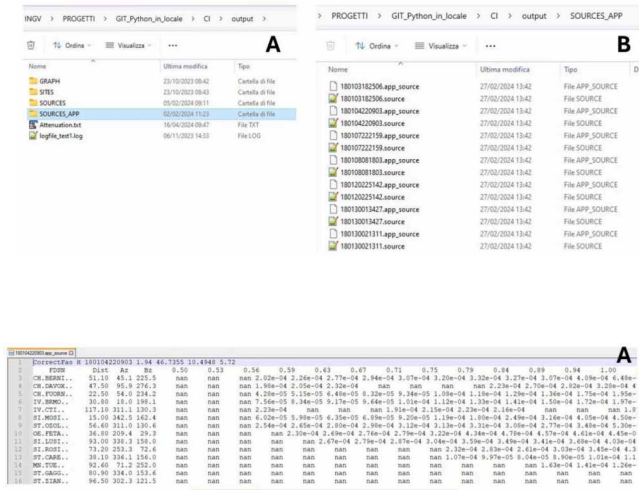


Figure 13 The output directory (in this example SOURCE_APP) contains two files for each event. The one with extension “app_source” and the other one with extension “source”.

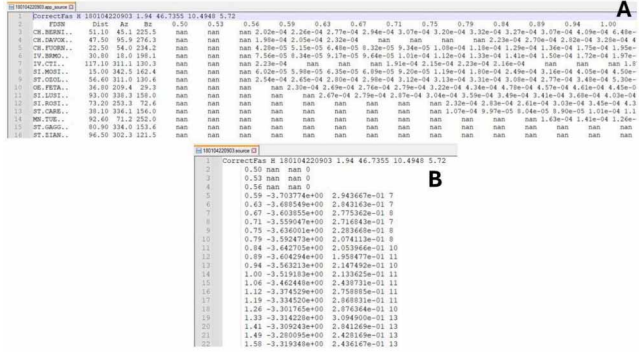


Figure 14 Example of CorrectFas.py output files for one event. A) File “app_source” includes the station source spectra for the event; B) File “source” is the median spectrum obtained from the apparent spectra.

3.3 FitAttenuation.py

FitAttenuation.py performs a parametric regression of GIT derived attenuation curves by using standard seismological models of geometrical spreading, anelastic attenuation, and high frequency decay (equation 2). The flowchart of FitAttenuation is shown in Figure 15.

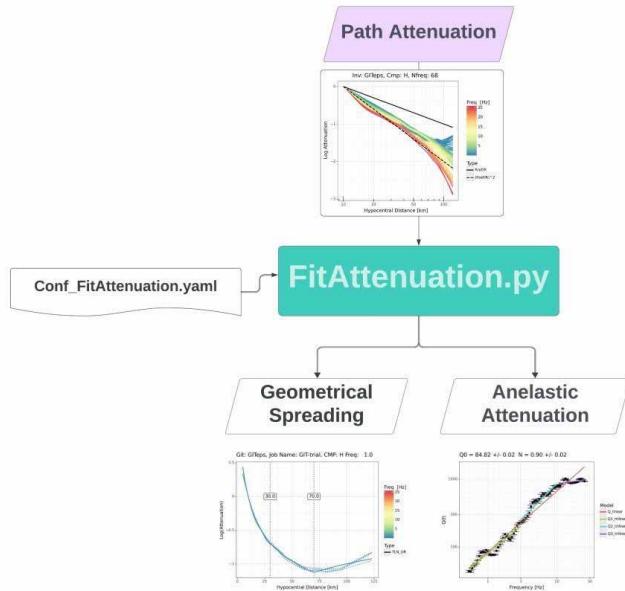


Figure 15 Flowchart of the GITpy module (FitAttenuation.py) which fits non-parametric spectral attenuation curves using standard seismological models. In this process, the geometrical spreading and anelastic attenuation contributions are isolated and described with parametric spectral models that depend on distance, quality factor, and kappa.

The FitAttenuation.py module fits the non-parametric attenuation curves using a model that incorporates a frequency-independent geometrical spreading term and a frequency-dependent anelastic attenuation term (equation 2):

$$\ln[P(R_{ij}, f)] = \ln[G(R)] - \frac{\pi f(R-R_0)}{Q(f)\beta} - \pi k(f - f_h) \quad (2)$$

where $G(R)$ is a piecewise function of the hypocentral distance R (equation 3), β is the shear wave velocity, R_0 is the reference distance, $Q(f) = Q_0 f^N$ is the quality factor, and k is the parameter describing the high-frequency attenuation above the hinge frequency f_h [Anderson and Hough, 1984].

The main challenge in reproducing the GIT-derived attenuation with equation (2) lies in the trade-off between $G(R)$ and $Q(f)$. FitAttenuation.py employs a two-step approach to model the non-parametric attenuation. In the first step, the geometrical spreading is determined by analyzing a set of non-parametric curves around 1 Hz (i.e., from 0.8 to 1.2 Hz) to minimize the effects of anelastic attenuation. In the second step, the attenuation is corrected for $G(R)$ to derive the k operator and the frequency-dependent quality factor.

Regarding the modeling of $G(R)$, the user can define up to three hinge distances (HD_1 , HD_2 , and HD_3), while the exponents of the model ($n1$, $n2$, $n3$, and $n4$) are determined through linear regression. The hinge distances can be defined after visually inspecting the non-parametric curves.

$$\ln[G(R)] = \begin{cases} \left(\frac{R_0}{R}\right)^{n1} & \text{for } R \leq HD_1 \\ \left(\frac{R_0}{HD_1}\right)^{n1} \left(\frac{HD_1}{R}\right)^{n2} & \text{for } HD_1 < R \leq HD_2 \\ \left(\frac{R_0}{HD_1}\right)^{n1} \left(\frac{HD_1}{HD_2}\right)^{n2} \left(\frac{HD_2}{R}\right)^{n3} & \text{for } HD_2 < R \leq HD_3 \\ \left(\frac{R_0}{HD_1}\right)^{n1} \left(\frac{HD_1}{HD_2}\right)^{n2} \left(\frac{HD_2}{HD_3}\right)^{n3} \left(\frac{HD_3}{R}\right)^{n4} & \text{for } R > HD_3 \end{cases} \quad (3)$$

Once the $G(R)$ has been calibrated, a preliminary fit of the $G(R)$ -corrected attenuation (in natural logarithm) is performed above a user-defined hinge frequency f_h , considering the linear term $\pi k(f - f_h)$. This step effectively removes the kappa model from the $G(R)$ -corrected attenuation curves, leaving the residual attenuation to represent only the contribution of the anelastic attenuation term (equation 4):

$$\ln[P(R, f)]_{corr} = C(R - R_0) = \frac{-\pi f(R - R_0)}{Q(f)\beta} \quad (4)$$

Finally, a least-squares regression is applied in the distance range where the $G(R)$ and k -corrected attenuation curves exhibit linear trends, allowing for the derivation of $Q(f)$. The frequency-dependent quality factor can be modeled either as a monotonically increasing function of the frequency $Q(f) = Q_0 f^N$, or as a piecewise linear function with user-defined hinge frequencies (HF_1 and HF_2) (equation 5).

$$Q(f) = \begin{cases} Q_{01} f^{N1} & \text{for } f < HF_1 \\ Q_{02} f^{N2} & \text{for } HF_1 \leq f < HF_2 \\ Q_{03} f^{N3} & \text{for } f \geq HF_2 \end{cases} \quad (5)$$

The drawback of this procedure is that the non-parametric curves around 1 Hz tend to maintain zero amplitude across the entire distance range due to the correction for geometrical spreading. As a result, the corresponding quality factors tend to infinity, and the $Q(f)$ trend deviates

significantly from linearity. To address this issue, one approach is to account for the trade-off between geometrical spreading and anelastic attenuation by also correcting the non-parametric attenuation for the anelastic attenuation evaluated around 1 Hz [Bindi and Kotha, 2020; Bindi et al., 2023c]. This correction can be applied in FitAttenuation.py by setting the configuration file parameters: `corr_fact_flag = 1` and `Q0_corr` to the assumed quality factor at 1 Hz.

3.3.1 Configuration file for FitAttenuation.py

The YAML configuration file for the FitAttenuation.py module consists of the following blocks: GLOBAL contains general settings; PLOT configures display and saving options for the figures; FREQUENCY GS defines options for selecting the frequency to fit the geometrical spreading; GEOMETRICAL_SPREADING provides alternatives to define a fitting model for the geometrical spreading; ANELASTIC_ATTENUATION allows the user to specify the characteristics of the anelastic attenuation model; KAPPA contains information about the high frequency attenuation. Detailed descriptions of each block are provided in Table 8, with an example shown in Figure 16.

Block	Field	Description
GLOBAL	att_dir	Directory of output results from GITeps.py
	att_file	File containing the non-parametric attenuation curves (output of GITeps.py)
	out_dir	Directory to save the FitAttenuation.py outputs
PLOT	plot_graph	"True": shows figures during the code execution; "False": avoids the figures visualization
	save_graph	"True" saves figures in the output directory; "False" does not save figures
FREQUENCY GS	f_step	The range of variability around the frequency used to calibrate the geometrical spreading (freq)
	freq	Reference frequency to calibrate the geometrical spreading. Commonly 1 Hz is assumed [Aki and Chouet, 1975]
GEOMETRICAL SPREADING	gs_model	Set the geometrical spreading model G(R) to use: Rref/R = 1, bilinear = 2, trilinear = 3, quadrilinear = 4
	HD1	First hinge distance (only for gs_model 2, 3 or 4)
	HD2	Second hinge distance (only for gs_model 3 or 4)
	HD3	Third hinge distance (only for gs_model 4)

ANELASTIC ATTENUATION	case	The User can select one of the following attenuation models by setting this variable as follows: 1: fit by considering a G(R) term + Q(f) term and NO kappa 2: fit by considering a G(R) term + Q(f) term and an UNKNOWN kappa 3: fit by considering a G(R) term + Q(f) term and a KNOWN kappa (being G(R) the geometrical spreading, Q(f) the frequency dependent quality factor and kappa the high-frequency attenuation term)
	corr_fact_flag	This flag allows the User to decide if correct G(R) with an exponential term accounting for the anelastic attenuation at 1Hz or not. 1: yes (default value) 0 : no
	Q_type	Set how to fit the frequency-dependent Q(f): 0: linear fitting 1: tri-linear fitting
	Q0_corr	Quality factor at 1 Hz (used if corr_fact_flag = 1)
	dist_min_Q	Minimum distance in km to fit Q(f)
	dist_max_Q	Maximum distance in km to fit Q(f)
	Vs	S-wave velocity [km/s]
	HF1	First hinge frequency (only for case Q_type = 1)
	HF2	Second hinge frequency (only for case Q_type = 1)
KAPPA	hingfreq	Hinge frequency [Hz] to fit/correct the kappa value (used only if case = 2 or 3)
	k_known	Kappa value if known (used only if case = 3)

Table 8 Description of the key fields included in the configuration file of FitAttenuation.py.

Figure 16 Example of configuration file for module FitAttenuation.py.

```

1 # Conf file for FitAttenuation.py
2
3 GLOBAL:
4   att_dir: "C:/Users/39328/INOV/PROGETTI/GIT_python_in_locale/CI/test1/"
5   att_file: "Attenuation.txt"
6   out_dir: "C:/Users/39328/INOV/PROGETTI/GIT_python_in_locale/CI/test1/FIT/AttMod/"
7
8 PLOT:
9   plot_graph: True
10  save_graph: True
11
12 FREQUENCY OR:
13   f_step: 0.2
14   f_max: 1 # choice of frequency to fit the geometrical spreading (among those in f_single). Default 1Hz
15
16 GEOMETRICAL SPREADING:
17   gs_model: 0 # first geom spreading: Rref/R = 1, bilinear = 2 trilinear = 3 quadrilinear = 4
18   R0: 20 # first hinge distance
19   R02: 0 # second hinge distance (only for gs_model 3 or 4)
20   R03: None # third hinge distance (only for gs_model 4)
21
22 ANELASTIC ATTENUATION:
23   case: 1 # 1: fit by considering a g(r) term + Q(f) term and NO kappa
24   # 2: fit by considering a g(r) term + Q(f) term and an UNKNOWN kappa
25   # 3: fit by considering a g(r) term + Q(f) term and a KNOWN kappa
26   corr_fact_flag: 1 # 0: you want to correct Q(R) with an exponential term accounting for the anelastic attenuation at 1Hz?
27   # (YES=1 - NO=0). Default : 1
28
29 Q_type: 0 # 0: fitting linear Q(f) 1: fitting tri-linear per Q(f)
30
31 Q0_corr: 100
32 dist_min_Q: 0
33 dist_max_Q: 70
34 Vs: 3.5
35
36 HF1: 1.0 # first hinge frequency (only for case Q_type: 1)
37 HF2: 10 # second hinge frequency (only for case Q_type: 1)
38
39 KAPPA:
40   hingfreq: 10
41   k_known: 0.004

```

3.3.2 Run attenuation modeling module

The User has the option to run FitAttenuation.py using an interactive mode or an automatic mode. The interactive mode is useful to find the best model by testing various options and immediately visualize the test results with the possibility to make many tests in a short time. The automatic procedure uses the information from the configuration file in case the User already knows the parameters he wants to use to model the non-parametric attenuation curves. Interactive mode is entered by executing the python code with --input followed by the non-parametric attenuation file (Attenuation.txt) without configuration file argument:

```
python $code_path/FitAttenuation.py --input $your_path/Attenuation.txt --log
$your_path/FitAttenuation.log
```

Automatic mode requires the configuration file:

```
python $code_path/FitAttenuation.py --config $your_path/ConfAttFIT.yaml --log
$your_path/FitAttenuation.log
```

Example:

```
python FitAttenuation.py --input GITOUT/Attenuation.txt --log FitAttenuation.log
(interactive mode)
python FitAttenuation.py --config Conf_FitAttenuation.yaml --log FitAttenuation.log
(automatic mode)
```

Options	Description	Default	Required
input	TXT file containing the non-parametric attenuation to be fitted	na	True (only for interactive mode)
config	Configuration file in YAML format to set the non-parametric curve fitting	na	True (only for automatic mode)
output	Output file format are: 'csv' or 'json'; to save data in both formats set 'all'	all	
log	Log file in TXT format	na	True
v	Verbose mode (prints summary and other details)	True	

Table 9 List of options to run FitAttenuation.py both in interactive and in automatic way.

3.3.3 Output of FitAttenuation.py

The outputs of the FitAttenuation.py module are saved in the directory specified in the configuration file (under the field "out_dir" in Table 8), or, if interactive execution is chosen, in the user-defined output directory. Figures are saved only if the "plot_graph" field in Table 8 is set to "true." In this case, three figures are generated (example shown in Figure 17). The numerical values of the attenuation model can be saved in two formats: "json" and/or "csv" (Figures 17 and 18).

For the csv format, the anelastic attenuation and geometrical spreading models are saved in two separate files (Figures 18A, 18B), with the parameters described in a third file named “dictionary.csv” (Figure 18D). If the “json” format is chosen, all the information is stored in a single file (Figure 18C). Finally, a log file is saved in the same directory as a “txt” file.

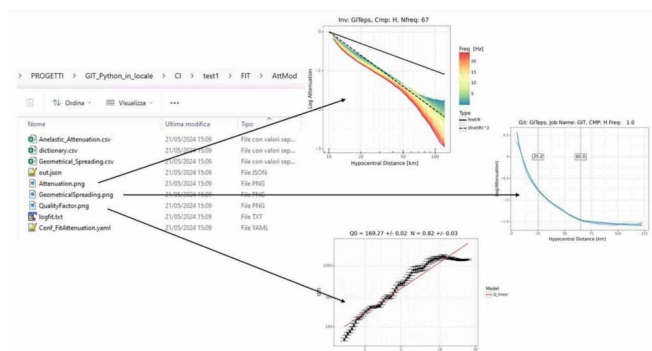


Figure 17 Output files of FitAttenuation.py.

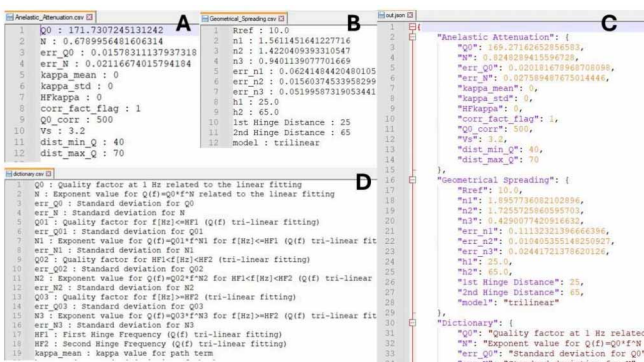


Figure 18 Structure of the FitAttenuation.py output files. A) Anelastic attenuation parameters; B) Geometrical spreading parameters; C) Anelastic attenuation and geometrical spreading parameters; D) glossary of each key field of the output files.

3.4 FitSource.py

FitSource.py describes the source spectra using standard seismological models that account for several source parameters, including seismic moment, corner frequency, stress drop, radiated energy, the ratio between observed and full energy, apparent stress, and radiation efficiency. The flowchart for FitSource.py is shown in Figure 19. The configuration file for FitSource.py includes settings for input/output, Brune’s model constraints, and source fitting parameters. It also supports three optional input files: one to constrain the seismic moment using reference catalogs, another to define the magnitude-dependent frequency ranges for fitting the source spectra, and a third to incorporate a 1D-layered crustal model instead of a homogeneous one.

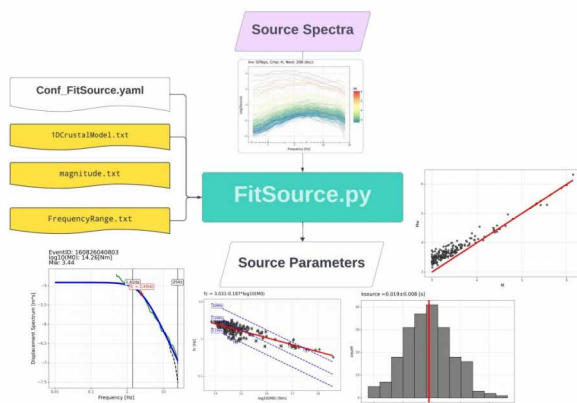


Figure 19 Flowchart of the GITpy module that performs fitting of source spectra in terms of seismic moment and corner frequency (FitSource.py), and allows to derive other seismological parameters (e.g. stress drop, radiated energy).

In the current version of GITpy, only Brune's ω^2 model [Brune, 1970, 1971] is implemented (equation 6) to fit the non-parametric source spectra derived from the GIT.

$$Si(f) = \frac{R_{\theta\phi}A}{4\pi\rho\beta^3R_0} M_{i0}(2\pi f)^2 \frac{1}{1 + \left(\frac{f}{f_{ic}}\right)^2} e^{-\pi\kappa_S(f - f_h)} \quad (6)$$

$R_{\theta\phi}$ represents the average S-wave radiation pattern [Boore and Boatwright 1984]; the parameter A accounts for the free-surface amplification factor. ρ and β are the assumed density [g/cm³] and the S-wave velocity [km s⁻¹] for the study region. R_0 indicates the reference distance [km] which is usually set to the lowest and well sampled bin of distance.

FitSource.py simultaneously fits the seismic moment M_{i0} [Nm] and the corner frequency f_{ic} [Hz] of the i -th earthquake. The process begins by correcting the acceleration source spectrum above the hinge frequency f_h to remove kappa effects. In the second step, the displacement source spectrum is fitted using Brune's model, with the seismic moment M_0 and corner frequency f_c constrained around the theoretical values defined by equations 7 [Kanamori, 1977] and 8 [Brune, 1970; 1971; Kramer, 1996], respectively.

$$M_0 = 10^{1.5 \cdot M_W + 16.1} \quad (7)$$

$$f_c = 4.9 \cdot 10^6 \cdot \beta \cdot \sqrt[3]{\frac{\Delta\sigma}{M_0}} \quad (8)$$

M_0 in equations 7 and 8 is in *dyne cm*, while $\Delta\sigma$ is fixed to 30 bars.

Fitting the source terms for large datasets of small-to-large seismic events recorded in regions with heterogeneous crustal properties is a complex challenge. For instance, limited bandwidth at low frequencies often prevents the observation of the corner frequencies of large earthquakes ($M > 5.5$) in their source spectra. To address this, we have implemented an option in FitSource.py that allows the seismic moment of large events to be fixed a priori using M_w estimates from independent sources. In this case, the only free parameter to be fitted in equation 6 is f_c . Additionally, the user can define the frequency bands used for fitting the source spectra, based on different magnitude intervals.

Once M_0 and f_c are obtained for each earthquake the Brune stress drop is calculated as follows:

$$\Delta\sigma = \frac{7M_0}{16r^3} \quad (9)$$

where r is the source radius of a circular fault model [Brune 1970; 1971]:

$$r = \frac{2.34 \beta}{2\pi f_c} \quad (10)$$

The apparent stress τ_a [N/m²], which is a measure of stress change at a seismic source is calculated as defined by Wyss [1970]:

$$\tau_a = \mu \frac{E_s}{M_o} \quad (11)$$

where μ [N/m²] is the shear modulus of rigidity of the source material and E_s [Nm] is the seismic energy radiated from the source.

In FitSource.py, we neglect the contribution from P-waves, which accounts for less than 5% of the total radiated energy. Seismic energy E_s is assessed from both theoretical and observed spectra. In the theoretical case, the seismic moment and corner frequency derived from the GIT are used in equation 12 [Izutani and Kanamori, 2001], with integration extending up to 10 times the corner frequency of each earthquake.

$$E_s = \frac{4\pi}{5Q\beta^5} \int_0^\infty \left| \frac{M_{of}}{1 + \left(\frac{f}{f_c}\right)^2} \right| df \quad (12)$$

In the second case, the radiated energy is estimated by equation 13:

$$E_s = \frac{16\pi Q\beta R_0^2}{5A^2 (R^{\theta\phi})^2} \int_{f_{min}}^{f_{max}} |S_{vel}|^2 df \quad (13)$$

where S_{vel} is the empirical velocity source spectrum at the reference distance R_0 . However, since the empirical source spectra provided by GIT are limited to a finite frequency range [f_{min} – f_{max}], the energy estimation can be affected by this bandwidth limitation. Given that more than 80% of the seismic energy may be radiated at frequencies above the corner frequency [Ide and Beroza, 2001], a correction for finite frequency bandwidth is applied to equation 13. This adjustment accounts for the energy radiated up to infinite frequency, as shown in equation 14:

$$\int_{f_{min}}^{f_{max}} |S_{vel}|^2 df = kv \int_{-\infty}^{+\infty} |S_{vel}|^2 df \quad (14)$$

The ratio between observed and full energy [Ide and Beroza, 2001; Hwang et al., 2022] is reported in equation 15:

$$kv = \frac{2}{\pi} \left[-\frac{\left(\frac{f_{max}}{f_c}\right)}{1 + \left(\frac{f_{max}}{f_c}\right)^2} + \frac{\left(\frac{f_{min}}{f_c}\right)}{1 + \left(\frac{f_{min}}{f_c}\right)^2} + \tan^{-1} \frac{f_{max}}{f_c} - \tan^{-1} \frac{f_{min}}{f_c} \right] \quad (15)$$

Finally, the radiation efficiency η_{sw} is computed as the ratio between apparent stress τ_a and static stress drop $\Delta\sigma$ [Savage and Wood, 1971; Beeler et al., 2003; 2012]:

$$\eta_{sw} = \frac{\tau_a}{\Delta\sigma} \quad (16)$$

3.4.1 Configuration file for module FitSource.py

The YAML configuration file for the FitSource.py module is structured into the following blocks: GLOBAL contains the general settings; CONSTRAINT includes constant values for calculations and information on managing reference moment magnitudes; CRUSTAL MODEL provides details on the crustal model to be used for source spectra estimation; SOURCEFITTING comprises the necessary parameters for source spectral fitting; KAPPA includes settings for kappa source fitting; PLOT allows configuration of figure display options; and SAVE contains information on how to save the figures. Detailed descriptions of each block are provided in Table 10, with example configurations shown in Figure 20. Figures 21 and 22 illustrate example files required for the configuration variables “magnitude_file,” “f_range_file,” and “crustal_model”. The “fixed_magnitude.txt” file (Figure 21) contains the moment magnitude values (M_w) for selected events in the dataset, as inferred from reference catalogs. It consists of three columns: the “Event_ID” column, which includes the event identifier; the “MW” column, which shows the independent moment magnitude value; and the “fixflag” column, which indicates how the M_w value is used within FitSource.py. Events with a fixflag value of 1 are used to constrain the source fit by fixing the seismic moment of each event to the M_w value (typically for large events, to mitigate the effects of limited bandwidth). Events with a fixflag value of 0 are used to anchor the seismic moments derived from the source spectral fit to a reference moment magnitude. In these cases, the average offset between GIT-derived seismic moments and the M_0 values inferred by reference M_w catalog is calculated. This offset is then used to shift all source spectra to correct for any bias with respect to the reference catalog.

Block	Field	Description
GLOBAL	job_name	Title for the analysis (not used at the moment but will be added to the plots)
	source_dir	Directory containing the non-parametric source spectra (output of GITeps.py) or directory containing the apparent spectra (output of CorrectFas.py)
	out_dir	Directory to save the FitSource.py outputs
CONSTRAINT	dref	Reference distance [km] for which attenuation function is set to vdref (Table 8). For coherence the User should set this value as in the inversion configuration file (Table 8)
	Rp	S-wave average radiation pattern
	Fs	Free-surface coefficient
	En	Coefficient that accounts for the separations of S-wave energy onto two horizontal components
	magnitude_file	File including a list of events with independent M_w . This file has 3 columns: the first is the event ID, the second the M_w value, the third is a flag: 1 means that this M_w is fixed for the event; 0 means that the event is one of those used to calculate the difference between the independent M_w and the calculated M_w . More details are in Figure 22
	AnchorageMw	The User can decide to anchor the estimated M_{ws} to the average of the difference with respect to an independent M_w catalog or not 0: no 1: yes

CRUSTALMODEL	type_model	Possible settings: HOMOGENEOUS: in this case variables "Vs" and "rho" are used for the source model. HETEROGENEOUS: in this case Vs and rho are set reading the file indicated in the variable "crustal_model"
	crustal_model	File name (including path) containing the used crustal model
	Vs	S-wave velocity [km/s] in the source region. For coherence the User should set this value as in the inversion configuration file (see Vs in Table 8)
	rho	Density in the source region [g/cm ³]
SOURCEFITTING	type	1 : simultaneous fit for M_0 and f_c
	sourcemodel	Option to choose the source model for future implementations. At the moment this parameter can be set as follows: 1: Brune's model
	f_range	Option to manage the range of frequency for spectrum fit: 1: $f_{\min}$ and $f_{\max}$ from file
	f_range_file	File name (with path) used when variable "f_range" is 1 (Figure 23)
	appflag	As FitSpectra.py can be used to fit both output of GITeps.py and CorrectFas.py it is important to set on which outputs to perform the fitting: 0:working on median source spectra (*.source); 1:working on apparent source spectra (*.app_source)
	list_eve	List of events to fit (used only for the case appflag=1)
KAPPA	hingfreq	Lower frequency bound above which to estimate kappa source
PLOT	param	Option to decide how create figures: #ACC : in acceleration (y-axis: m/s) #VEL : in velocity (y-axis: m) #DISP: in displacement (y-axis: m*s)
	plot_graph	Option to show figures on the screen : TRUE or FALSE
	list_eve	List of events for which to show plots on the screen
SAVE	save_graph	Option to save the figures: TRUE or FALSE
	list_eve	List of events for which to save plots

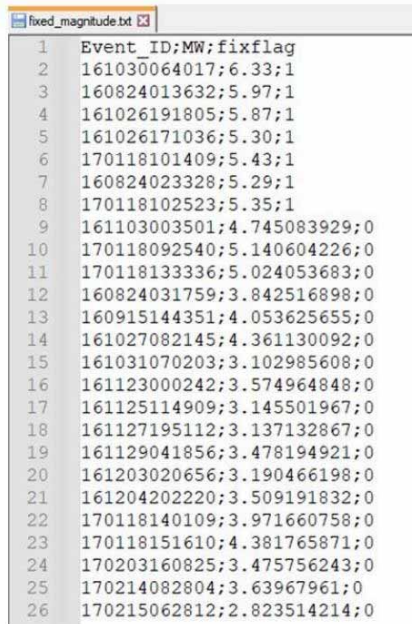
Table 10 Description of the key fields included in the configuration file of FitSource.py.

Figure 20 Example of configuration file for module FitSource.py.

```

1 GLOBAL:
2   job_name: "GIT CI"
3   source_dir: "C:/Users/39328/INOV/PROGETTI/GIT_Python_in_locale/test/500CES"
4   out_dir: "C:/Users/39328/INOV/PROGETTI/GIT_Python_in_locale/test/FIT/srcModel"
5
6 CONTOUR:
7   dwd: 10 #verify that the value is the name of the Attenuation fitting (VAME)
8   Rp: 0 # radiation pattern
9   Pz: 0 # wave direction
10  Bz: 0 # for vertical composition of 2 horiz. comp: 1/sgt(2); for rms spectra of 2 horiz. comp
11  magnitude_file: "C:/test/fitmed_magnitude.txt"
12  Anchoragep0: 1 # 1 yes / 0 no
13
14 CRUSTALMODEL:
15   type_model: HOMOGENEOUS #HOMOGENEOUS # HETEROGENEOUS, HOMOGENEOUS
16   crustal_model: "C:/Users/39328/INOV/PROGETTI/GIT_Python_in_locale/crustalmodel.txt"
17   Vs: 1 # verify that the value is the name of the Attenuation fitting (VAME)
18   rho: 1 # density g/cm3
19
20 SOURCEFITTING:
21   type: 1 # simultaneous fit for M0 and for 2: three steps from Bindi and Rota 2020
22   sourcemodel: 1 #1: Brune's model
23   f_range: 1 # min and max from table/ 2: this is automatically calculated and fixed=90 Hz
24   f_range_file: "C:/test/fit_f_range.txt"
25   appflag: 0 # 0: working on median source spectra (*.source); 1: working on apparent source spectra (*.app_source)
26   list_eve: [141030046017,140824013632,140824000409]
27
28 KAPPA:
29   hingfreq: 10
30
31 PLOT:
32   param: DISP #ACC,VEL,DISP
33   plot_graph: False
34   list_eve: [141030046017,140824013632,140824000409]
35
36 SAVE:
37   save_graph: True
38   list_eve: [141030046017,140824013632,140824000409]
39
40
41

```

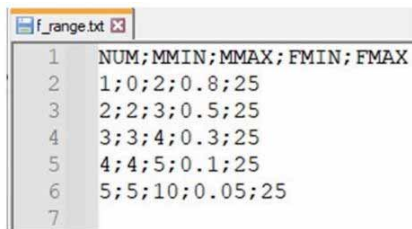


```

1 Event_ID;MW;fixflag
2 161030064017;6.33;1
3 160824013632;5.97;1
4 161026191805;5.87;1
5 161026171036;5.30;1
6 170118101409;5.43;1
7 160824023328;5.29;1
8 170118102523;5.35;1
9 161103003501;4.745083929;0
10 170118092540;5.140604226;0
11 170118133336;5.024053683;0
12 160824031759;3.842516898;0
13 160915144351;4.053625655;0
14 161027082145;4.361130092;0
15 161031070203;3.102985608;0
16 161123000242;3.574964848;0
17 161125114909;3.145501967;0
18 161127195112;3.137132867;0
19 161129041856;3.478194921;0
20 161203020656;3.190466198;0
21 161204202220;3.509191832;0
22 170118140109;3.971660758;0
23 170118151610;4.381765871;0
24 170203160825;3.475756243;0
25 170214082804;3.63967961;0
26 170215062812;2.823514214;0

```

Figure 21 File required by the yaml configuration file for FitSource.py (variable “magnitude_file”).



```

1 NUM;MMIN;MMAX; FMIN; FMAX
2 1;0;2;0.8;25
3 2;2;3;0.5;25
4 3;3;4;0.3;25
5 4;4;5;0.1;25
6 5;5;10;0.05;25
7

```

Figure 22 Example file “f_range.txt” for the variable “f_range_file” of the configuration file (Table 10).

3.4.2 Run source modeling module

FitSource.py can be executed to fit the median source spectra (GITeps.py output source files) or the apparent source spectra (output of CorrectFas.py). In both cases the code is launched in the same way, as the configuration file has the information on the spectra to be fitted (see Table 10 variable “appflag” and “sourc_dir”):

```
python $code_path/FitSource.py --config $your_path/Conf_FitSource.yaml --log
$your_path/logsource.log
```

Example:

```
python FitSource.py --config Conf_FitSource.yaml --log logsource.log
```

Options	Description	Default	Required
config	Configuration file in YAML format to set the source fitting	na	True
output	Output file format are: 'csv' and 'json'; to save data in both formats set 'all'	all	
log	Log file in TXT format	na	True
v	Verbose mode (prints summary and other details)	True	

Table 11 List of options to run the FitSource.py.

3.4.3 Output of FitSource.py

The FitSource.py outputs are different depending on whether the module is applied to median source spectra (files “.source”, output of GITeps.py) or to the apparent source spectra (files “.app_source”, output of CorrectFas.py).

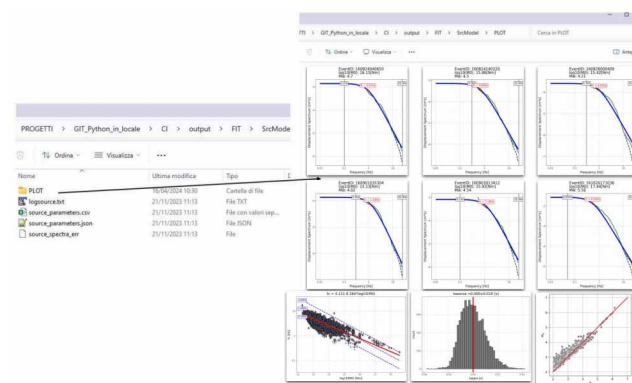
3.4.3.1 Fit median source spectra derived from GITeps.py

When FitSource.py is applied to the non-parametric source spectra generated by GITeps.py, the module fits the source spectra according to the specifications in the configuration file (Section 3.4.1). An example of the output files is shown in Figure 23. Based on the configuration file settings, users can choose to save the corresponding source spectra and models in acceleration, velocity, or displacement (refer to the PLOT and SAVE blocks in Table 11). Additionally, three figures summarizing the results for the entire dataset are produced (Figure 24). If the source fitting process fails to find a solution for a specific earthquake, the corresponding event ID is recorded in a file named “source_spectra_err,” allowing the user to easily identify and review the problematic event. Another output of the module is a table of source parameters, available in both “.csv” and/or “.json” formats (see Table 11). The parameters listed for each event are described in Table 12, with an example shown in Figure 25. Finally, a log file is generated at the end of the execution of FitSource.py.

Figure 23 Example of source_parameters.csv. MWcat = -9 means that the seismic event is not included in a reference moment magnitude catalog (see “magnitude_file” in Table 10).

	E	D	M	F	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
[Evel	M	MWcat	fmin [Hz]	fmax [Hz]	M0 [Dynes]	M0_err	f0 [Hz]	f0_err	M0_fit	M0_fit_err	sigma	sigma_err	sigma_fit	sigma_fit_err	log10(f0)	log10(f0_err)	log10(sigma)	log10(sigma_err)					
16100004017	6.14	0.05	25	5.94E+18	0	0.17478786	0.00017376	9999	0.02038308	0.00014201	1.42038308	0.00000000	14.18793276										
160804013632	6	5.97	25	1.14E+18	0	0.25383502	0.00780361	9999	0.023562234	0.001232785	4.78713523	0.00098629	13.9518434										
16100190803	5.98	5.97	25	0.64E+17	0	0.26234033	0.004071454	9999	0.02713508	0.001093031	3.74949024	0.00017962	13.5208031										
16100190807	5.83	9	25	3.51E+17	9.05E+16	0.30234488	0.002634566	-9997	0.00233374	0.00144503	5.80220966	1.30711139	13.1227769										
161002671036	5.36	5.3	25	1.12E+17	0	0.52303018	0.01203388	9999	0.024877991	0.001035862	4.10212602	0.02481612	12.5284741										
160804022036	5.42	5.28	25	1.39E+17	0	0.47026848	0.00668443	9999	0.02033395	0.001202638	2.35263264	0.0224877	12.3448489										
170110103409	5.42	5.43	25	1.76E+17	0	0.37111380	0.00611199	9999	0.02102701	0.000929597	2.36201048	0.0384385	12.4819418										
170110200540	5.23	5.14804236	0.85	25	6.19E+16	1.13E+16	0.16349623	0.00808792	-9992	0.020917076	0.00111868	3.14013122	0.58495424	12.1589392									
170110103252	5.33	5.35	25	1.33E+17	0	0.378021378	0.00606192	9999	0.016300121	0.00149589	2.181601138	0.03932614	12.3573308										
170110103254	5.3	9	25	9.49E+16	1.87E+16	0.18464891	0.001261453	-9999	0.016871843	0.000900733	3.542601727	0.039742134	12.3669008										
170110103256	5.34	5.024832683	0.85	25	4.81E+16	9.72046E+15	0.10911144	0.001752863	-3.885	0.018084884	0.000910185	2.842620807	0.34504603	11.9746454									
170110103153	5.18	8	25	8.45E+16	8.45E+16	0.43284824	0.00464233	-98033	0.01318162	0.00177779	3.47779387	0.36088626	12.14209158										
161102000001	4.54	4.74080839	0.1	25	2.81E+16	2.78797E+15	8.72588050	0.00178135	-0.9767	0.00881184	0.000104889	1.982103057	0.267799623	11.4640682									
160804103070	4.84	4.53798683	0.1	25	1.02E+16	5.63018E+14	0.09638325	0.04897383	-3640	0.002119632	0.001202938	1.884784046	0.179707174	11.15126033									

Figure 24 Output files of FitSource.py when applied to the median non-parametric source spectra derived from GITeps.py (*.source files).



Parameter	Description
Evel	Event Identification Number
M	Magnitude from FAS input files (the User should know what kind of magnitude he included in his input data files)
MWcat	Mw derived from reference catalog
fmin [Hz]	Min frequency used for the source spectra fitting
fmax [Hz]	Max frequency used for the source spectra fitting

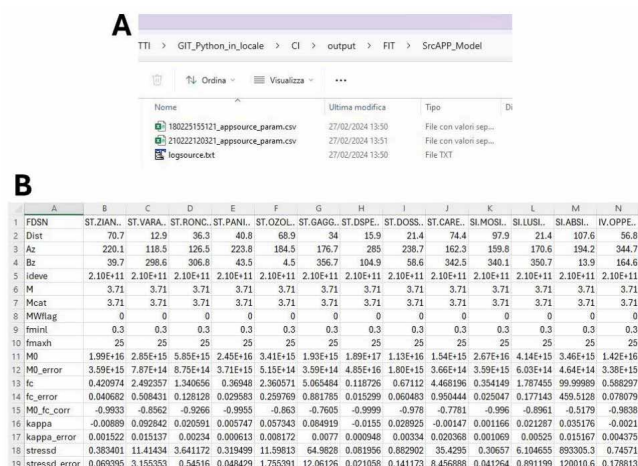
M0 [Nm]	Seismic moment [Nm]
M0_err	Standard error on seismic moment [Nm]
fc [Hz]	Corner frequency [Hz]
fc_error	Standard error on corner frequency [Hz]
M0_fc_corr	Correlation coefficient between M0 and fc
kappa [s]	Source kappa [s]
k_err	Standard deviation of the source kappa [s]
stress_drop [MPa]	Stress drop [MPa]
stress_drop_err	Error propagation on stress drop
log10_Er_mod [J]	log10 of radiated energy (from Brune's model)
AppStress [Pa]	Apparent stress by Wyss [1970] (using Er_mod)
log_10_Er_oss [J]	log10 of radiated energy [J] obtained from non-parametric spectrum
log_10_Er_oss_fit [J]	log10 of radiated energy [J] obtained from fitted spectrum in the range fmin and fmax
AppStress_oss [Pa]	Observed apparent stress by Wyss [1970] (using Er_oss)
log_10_Er_tot [J]	Total energy: Er_oss and extrapolated for low and high frequency [Ide and Beroza, 2001]
log_10_Er_tot_fmin_fmax_fit [J]	Total Energy: Er_oss_fit and extrapolated for low and high frequency [Ide and Beroza, 2001]
η SW	Savage-Wood seismic efficiency η SW (Apparent stress / stress drop)
KV	Ratio between the estimated energy and the total energy considering the entire frequency range [Ide and Beroza, 2001]
KV_fmin_fmax_fit	Ratio between the estimated energy and the total energy considering the frequency range [fmin,fmax] [Ide and Beroza, 2001]
Mw	Moment magnitude

Table 12 Source parameters saved in the output file “source_parameters.csv”.

3.4.3.2 Fit apparent source spectra derived from CorrectFas.py

When FitSource.py is applied to the apparent source spectra generated by CorrectFas.py, the module fits each apparent source spectrum (files with the “app_source” extension) according to the specifications in the configuration file (Table 7). Specifically, source fitting and the calculation of source parameters are performed for the events listed in the configuration file (under the SOURCEFITTING block, field “list_eve” in Table 10). For each event, a file named “*_appsource_param.csv” (where * corresponds to the event ID) is created (Figure 25A), containing the source parameters derived for each available station. An example is shown in Figure 25B, where a table of values is provided for a sample event. In the first column, the list of parameters is displayed, while from the second column onward, the corresponding values for each station are provided, with the station names indicated in the header row.

Figure 25 Output files of FitSource.py when applied to the apparent source spectra derived from CorrectFas.py (*.app_source files).



4. Tutorial Guide for New Users

The following is a step-by-step guide for a specific example taken from the Central Italy region.

4.1 Example Dataset

An example dataset (GITpy_db) is available for download from the GitLab README page [7] to test the GITpy package. GITpy_db consists of ~7,000 three-component accelerometric waveforms associated with 209 seismic events ($3.0 \leq M \leq 6.14$) recorded in Central Italy by 53 stations (Figure 26). The dataset includes the mainshocks from the most recent seismic sequences in the study area, such as the L'Aquila (2009) and Amatrice-Visso-Norcia (2016) events. GITpy_db is well-sampled in the near-field region Repi [20-40 km], and the average hypocentral depth is less than 10 km (Figure 27). However, the azimuthal coverage is poor, with most of the stations located in the NNW-SSE direction (Figures 26 and 27). It is important to note that GITpy_db does not fully capture the ground motion characteristics of Central Italy. It is primarily intended for testing the functionalities of GITpy before using more accurate datasets to derive region-specific source, attenuation, and site spectral parameters.

Figure 26 Left: Spatial distribution of events and stations included in the GITpy example dataset (GITpy_db); Right: magnitude-distance distribution in function of the number of waveforms.

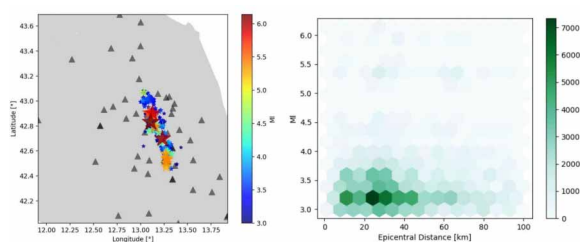
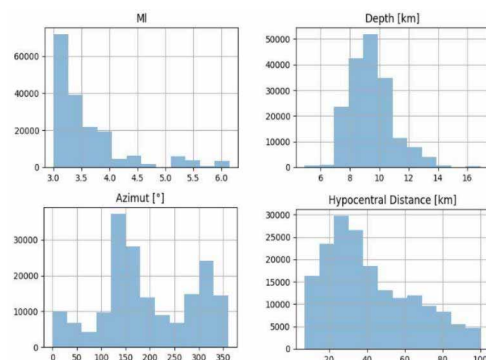


Figure 27 Top panel: Number of records according to events magnitude (left) and depth (right). Bottom panel: Number of records in function of recording stations azimuth (left) and hypocentral distance (right).



GITpy_db consists of a collection of flat files containing three-component FAS and related metadata for 69 events, analyzed in the frequency range 0.5 - 25.06 Hz. The FAS were computed from S-wave windows, which were selected based on a frequency-dependent threshold for the signal-to-noise ratio (SNR). For instance, SNR = 10 for frequencies between 0.2–0.4 Hz, SNR = 5 for 0.4–15 Hz, and SNR = 10 for frequencies above 15 Hz. The time windows used to calculate the spectral amplitudes were chosen based on the cumulative energy fraction: 90% for data recorded within a hypocentral distance of 25 km, 80% for distances between 25 and 50 km, and 70% for greater distances, with a minimum duration of 4 seconds. The FAS were then smoothed using the Konno and Ohmachi [1998] algorithm with a smoothing parameter of $b = 40$.

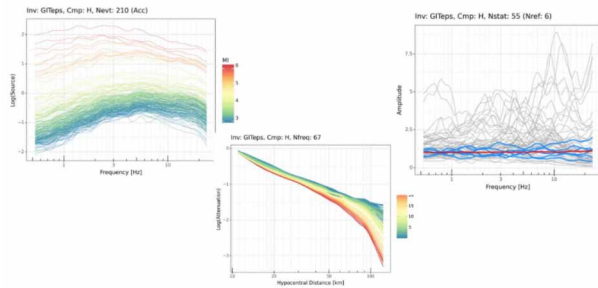


Figure 28 Graphical example of output by using the GITpy_db dataset.

An example of graphical output for source spectra, attenuation curves, and site amplification is shown in Figure 28. Each file in GITpy_d has the extension “.fas”. Below is an example (file FAS_001.00.fas in Figure 29):

Id	MI	EvLat	EvLon	EvtDpt	Ty	Stat	FUSM	CMP	MSA	StLat	StLon	StElev	Dip	Az	FAS_2	FAS_NS	FAS_EW	LenZ	LenNS	LenEW	Freq	TP	TS	ZE	ZS
1	050514014711	1.63	41.6007	14.9190	29.87	N	BSSO	IV.BSSO..	HH	1.41	41.5461	14.5938	0.0	30.9	43.0	241	2.954e-05	9.691e-05							
2	050514014711	1.63	41.6007	14.9190	29.87	N	TRIV	IV.TRIV..	HH	1.47	41.7666	14.5502	0.0	32.0	43.8	287	4.512e-05	1.850e-04							
3	050514014711	1.63	41.6007	14.9190	29.87	N	SACR	IV.SACR..	HH	1.51	41.3974	14.7057	0.0	36.1	46.9	209	3.307e-05	4.387e-05							
4	050514014711	1.63	41.6007	14.9190	29.87	N	MIDA	IV.MIDA..	HH	1.77	41.6419	14.2540	0.0	55.4	62.9	266	5.230e-05	3.195e-05							
5	050514014711	1.63	41.6007	14.9190	29.87	N	SOG	IV.SOG..	HH	1.45	41.3967	14.3792	0.0	55.5	63.1	234	1.902e-05	1.701e-05							
6	050514014711	1.63	41.6007	14.9190	29.87	N	VAGA	IV.VAGA..	HH	1.84	41.4154	14.2342	0.0	64.1	70.8	243	7.941e-06	7.572e-05							
7	051223042510	2.80	41.8915	15.2752	16.76	N	CIGN	IV.CIGN..	HH	2.15	41.6542	14.9050	0.0	40.5	43.8	229	6.651e-04	7.359e-04							
8	051223042510	2.80	41.8915	15.2752	16.76	N	FRES	IV.FRES..	HH	2.53	41.9735	14.6693	0.0	50.9	53.6	281	6.893e-04	3.207e-04							
9	051223042510	2.80	41.8915	15.2752	16.76	N	TRIV	IV.TRIV..	HH	2.62	41.7666	14.5502	0.0	61.6	63.9	257	1.475e-03	1.948e-03							
10	051223042510	2.80	41.8915	15.2752	16.76	N	BSSO	IV.BSSO..	HH	2.36	41.5461	14.5938	0.0	68.3	70.4	236	7.645e-05	3.157e-04							
11	051223042510	2.80	41.8915	15.2752	16.76	N	PSB1	IV.PSB1..	HH	2.72	41.2234	14.8108	0.0	83.7	85.4	208	3.797e-04	6.123e-04							
12	051223042510	2.80	41.8915	15.2752	16.76	N	MIDA	IV.MIDA..	HH	2.66	41.6419	14.2540	0.0	89.1	90.6	252	6.128e-04	5.749e-04							
13	051223042510	2.80	41.8915	15.2752	16.76	N	CAPE	IV.CAPE..	HH	3.16	41.0280	15.2366	0.0	96.0	97.5	182	1.864e-04	4.081e-04							
14	060430210757	2.99	41.4740	15.6668	15.21	N	MOCO	IV.MOCO..	HH	2.97	41.3700	15.1880	0.0	44.0	46.5	255	2.050e-03	2.738e-03							
15	060430210757	2.99	41.4740	15.6668	15.21	N	SOTA	IV.SOTA..	HH	2.99	41.1350	15.3650	0.0	45.3	47.8	214	1.860e-03	2.605e-03							
16	060430210757	2.99	41.4740	15.6668	15.21	N	PSB1	IV.PSB1..	HH	2.89	41.2234	14.8108	0.0	76.7	78.2	249	1.296e-03	1.511e-03							
17	060430210757	2.99	41.4740	15.6668	15.21	N	MRJC	IV.MRJC..	HH	3.36	40.7564	15.4889	0.0	81.1	82.5	191	7.769e-04	1.380e-03							
18	060430210757	2.99	41.4740	15.6668	15.21	N	SACR	IV.SACR..	HH	3.04	41.3974	14.7057	0.0	80.5	82.0	244	6.865e-04	2.953e-03							
19	060430210757	2.99	41.4740	15.6668	15.21	N	AMUR	IV.AMUR..	HH	3.06	40.9071	16.6041	0.0	100.6	101.7	128	7.841e-05	3.155e-04							
20	060430210757	2.99	41.4740	15.6668	15.21	N	CDRU	IV.CDRU..	HH	2.97	40.4896	15.3046	0.0	113.6	114.6	196	3.263e-04	2.509e-04							
21	060806193331	3.86	41.8838	15.4517	18.25	N	SOBT	IV.SOBT..	HH	4.11	41.7546	15.7437	0.0	28.1	33.5	121	4.316e-02	3.996e-02							
22	060806193331	3.86	41.8838	15.4517	18.25	N	MSAG	IV.MSAG..	HH	4.29	41.7120	15.5096	0.0	42.5	46.2	117	2.774e-02	4.031e-02							
23	060806193331	3.86	41.8838	15.4517	18.25	N	CIGN	IV.CIGN..	HH	3.53	41.6542	14.9050	0.0	52.0	55.1	241	1.119e-02	6.960e-03							
24	060806193331	3.86	41.8838	15.4517	18.25	N	MOCO	IV.MOCO..	HH	3.64	41.3700	15.1580	0.0	62.1	64.7	203	9.545e-03	6.046e-03							
25	060806193331	3.86	41.8838	15.4517	18.25	N	TRIV	IV.TRIV..	HH	3.70	41.7666	14.5502	0.0	75.5	78.0	260	1.914e-02	1.860e-02							
26	060806193331	3.86	41.8838	15.4517	18.25	N	BSSO	IV.BSSO..	HH	3.50	41.5461	14.5938	0.0	80.5	82.5	242	2.859e-03	3.422e-03							
27	060806193331	3.86	41.8838	15.4517	18.25	N	SACR	IV.SACR..	HH	3.42	41.3974	14.7057	0.0	82.2	84.2	229	2.458e-03	3.708e-03							
28	060806193331	3.86	41.8838	15.4517	18.25	N	SOTA	IV.SOTA..	HH	4.05	41.1350	15.3650	0.0	83.5	85.5	195	4.405e-03	4.929e-03							

Figure 29 Example of FAS file (file with extension “.fas”).

The User can create its own input files with an arbitrary number of columns to include various information (as in the example file above). GITpy will read only mandatory fields, the other columns will be ignored (Table 13).

Field	Description	Format	Mandatory
Id	Identification number of the event	string	yes
MI	Local Magnitude	float	yes
EvLat	Event latitude [°]	float	yes
EvLon	Event longitude [°]	float	yes
EvtDpt	Event depth [Km]	float	yes

Ty			
Stat			
FDSN	Code to identify the seismic station. The identifier is composed of the FDSN network_code, the location_code, and the station_code separated by a	char	yes
CMP	Identifier code composed of two letters: the first indicates the general sampling rate and response band of the data source; the second one specifies the family to which the sensor belongs.	char	yes
MSta			
StLat	Station latitude	float	yes
StLon	Station longitude	float	yes
StElev	Station elevation [m]	float	yes
Depi	Epicentral distance [km]	float	yes
Dipo	Hypocentral distance [km]	float	yes
Az	Event to station azimuth [°]	float	yes
FAS_Z	Fourier amplitude on vertical component [cm/s]	float	yes
FAS_NS	Fourier amplitude on NS component [cm/s]	float	yes
FAS_EW	Fourier amplitude on EW component [cm/s]	float	yes
LenZ			
LenNS			
LenEW			
Freq	Frequency of reference for the FAS measurements [Hz]	float	yes
TP			
TS			
ZE			
ZS			

Table 13 Description of the fields included in the GITpy_db FAS files (*.fas). The mandatory fields are required to run GITpy, while the metadata related to the optional ones (in yellow) contribute to qualify the dataset. Note that in GITpy_db we provided the local magnitude M_L . In a further release of GITpy we will implement the possibility to consider different types of magnitude.

4.2 Example Jupyter Python Notebooks

In this section we provide some Jupyter Notebooks tutorials to guide the User on how to run the four main python modules of GITpy (Table 2). Make sure you have Jupyter installed on your laptop. If not, you can install Jupyter in the gitpy python environment as follows if you are a LINUX User:

```
pip3 install notebook
```

To run the notebook:

```
jupyter notebook
```

On Windows, the user can launch Jupyter through the Anaconda shortcut. In both cases, this will open a new tab in the default web browser, which should resemble the screenshot in Figure 30. The Jupyter Notebooks (.ipynb files) from the GITpy project serve as tutorials to guide new users in running the code, displaying results, and reading explanations for each block of the source code. Examples of Notebooks for the GITeps.py (Figure 31), FitAttenuation.py (Figure 32), and FitSource.py (Figure 33) modules are provided below. These notebooks allow users to execute the three modules step-by-step, following the comments and instructions in each code block. This approach is particularly useful for familiarizing oneself with the software package before applying it to relevant case studies. Users should ensure that they correctly edit the paths and filenames of the input files, making sure they align with the directory structure of their operating system.

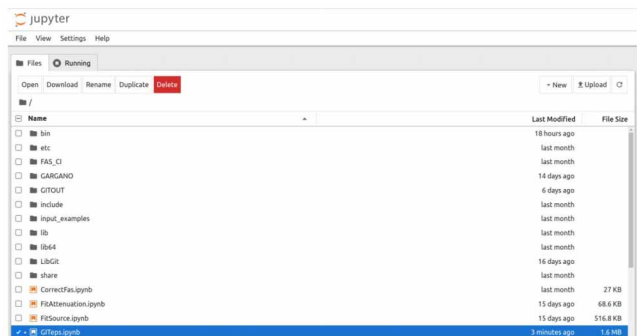


Figure 30 Example of dashboard, specifically designed for managing Jupyter Notebooks (.ipynb files).

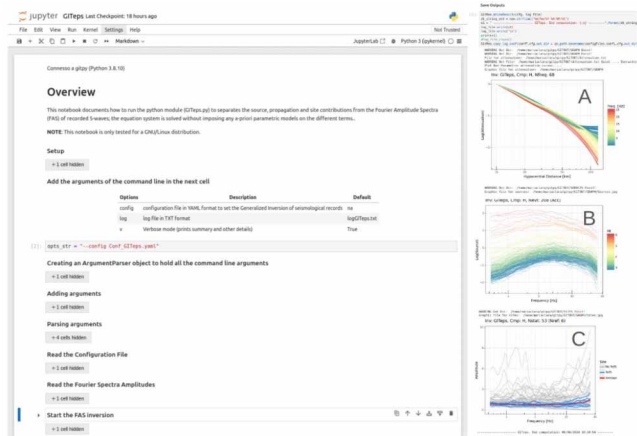


Figure 31 Example of Jupyter Notebook to run the GITpy inversion module (GITeps.py) to separate the FAS into the following three terms: A) path attenuation; B) Source spectra; C) site amplification. The blue lines in the plot of the frequency dependent site amplification factors indicate the reference ground motion used to constrain the solution of the overdetermined linear system (eq. 1), whereas the red lines indicate the median reference site seismic response.

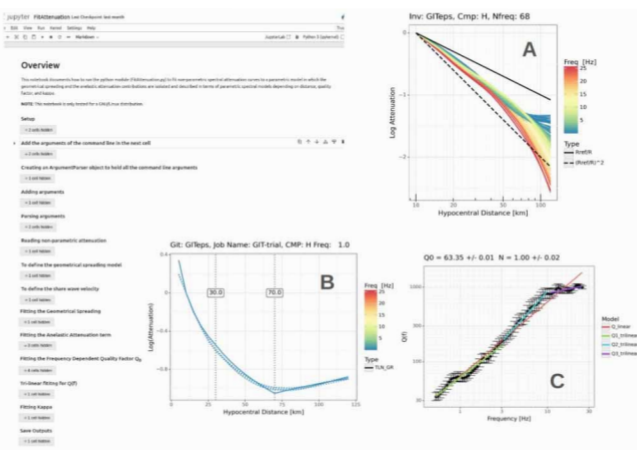
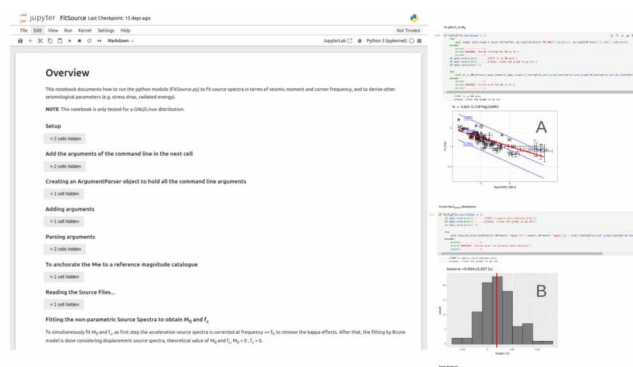


Figure 32 Example of Jupyter Notebook to run the GITpy module (FitAttenuation.py) developed for the physical parametrization of the non-parametric attenuation curves (A) in terms of: geometrical spreading (B) and frequency dependent quality factor (C).

Figure 33 Example of Jupyter Notebook to run the GITpy module (FitSource.py) developed for the physical parametrization of the source spectra in terms of: seismic moment M_0 and corner frequency f_c (A) and frequency k_s (B).



5. Conclusions

Although the generalized inversion of FAS for earthquakes is a well-established and widely used methodology in the scientific community, there is a lack of open-source software that implements this technique. GITpy is the first open-source tool designed to apply the generalized inversion technique to isolate and model the contributions of source, propagation, and site effects on ground shaking within the spectral domain. Specifically, GITpy is an object-oriented Python software package comprising four primary modules that calculate: 1) non-parametric source, path, and site terms derived from S-wave FAS; 2) apparent source spectra; 3) geometrical spreading, anelastic attenuation, and source-to-site kappa effects; and 4) source parameters inferred from seismological models and earthquake scaling laws.

The version presented in this technical report enables users to manage various levels of complexity in parameterizing geometrical spreading and anelastic attenuation. Although the current implementation is limited to Brune's standard model for describing source spectra in terms of moment magnitude, corner frequency, and stress drop, the software also includes the estimation of additional parameters (i.e., radiated energy, the ratio of observed to full energy, apparent stress, and radiation efficiency). Furthermore, a specific tool is provided to compute the source spectra as observed at each station, correcting the empirical FAS for site amplification and non-parametric attenuation. These apparent source spectra are particularly valuable for investigating spatial variability in ground motion (e.g., directivity effects) and for rapid estimation of source parameters after the occurrence of a significant earthquake.

Finally, the modular and flexible architecture of GITpy ensures that future software releases can easily incorporate more advanced models, further expanding its functionality.

License

GITpy is licensed under the GNU Lesser General Public License (LGPL) v3.0.

Citation

Permission to use, copy or reproduce parts of the GITpy code and/or related outputs is granted, provided that GITpy is properly referenced as:

D'Amico M., Morasca P., Spallarossa D., (2024). *GITpy, a python framework for the Generalized Inversion of Fourier Amplitude Spectra and the physical parametrization of Attenuation and Source Spectra*, Istituto Nazionale di Geofisica e Vulcanologia and Genova University Software Release. <https://gitlab.rm.ingv.it/inversion/gitpy/-/releases>

Getting Help

If you need help, please browse and search this documentation. To stay updated on the latest code developments and updates, please refer to the GitLab project <https://gitlab.rm.ingv.it/inversion/gitpy> and the related README. Additionally, you can submit questions, bug reports, and feature requests on the issue tracker on GitLab (<https://gitlab.rm.ingv.it/inversion/gitpy/-/issues>). All feedback and suggestions at maria.damico@ingv.it or paola.morasca@ingv.it are welcome.

Disclaimer

This software is preliminary or provisional and is subject to revision.

GITpy is provided on an As Is, with all faults, and As Available basis, without any warranties of any kind. We specifically do not warrant that GITpy is complete, accurate, reliable, or secure in any way, suitable for, or compatible with any of your contemplated activities, devices, operating systems, browsers, software and tools, or that their operation will be free of any viruses, bugs or program limitations.

The use of GITpy for the inversion of seismic records and the parametrization of related spectra are devoted to qualified users. No warranty, implicit or explicit, is attached to GITpy data and metadata. Every risk due to the improper use of data or related information is assumed by the User.

Acknowledgements

GITpy has been developed by the INGV and the Genova University. Thanks to Francesca Pacor for suggesting the idea of developing GITpy and to Laura Cataldi for testing the software package.

References

- Aki K. and Chouet B., (1975). *Origin of Coda Waves: Source, Attenuation and Scattering Effects*. Journal of Geophysical Research, 80, 3322-3342, <https://doi.org/10.1029/jb080i023p03322>
- Anderson J.G. and Hough S.E., (1984). *A Model for the Shape of the Fourier Amplitude Spectrum of Acceleration at High Frequencies*. Bulletin of the Seismological Society of America, 74, 5, 1969–93, <https://doi.org/10.1785/BSSA0740051969>
- Andrews D. J., (1986). *Objective determination of source parameters and similarity of earthquakes of different size*, in Earthquake Source Mechanics, S. Das, J. Boatwright, and C. H. Scholz (Editors). Geophysical Monograph Series, 259–267. <https://doi.org/10.1029/GM037p0259>
- Beeler N. M., Kilgore B., McGarr A., Fletcher J., Evans J. and Baker S. R., (2012). *Observed source parameters for dynamic rupture with non-uniform initial stress and relatively high fracture energy*. Journal of Structural Geology, 38, 77–89, <https://doi.org/10.1016/j.jsg.2011.11.013>
- Beeler N.M., Wong T.F. and Hickman S.H., (2003). *On the expected relationships among apparent stress, static stress drop, effective shear fracture energy, and efficiency*. Bulletin of the Seismological Society of America, 93, 1381–1389, <https://doi.org/10.1785/0120020162>
- Bindi D., Spallarossa D., Picozzi M. and Morasca P., (2020). *Reliability of Source Parameters for Small Events in Central Italy: Insights from Spectral Decomposition Analysis Applied to Both Synthetic and Real Data*. Bulletin of the Seismological Society of America, 110, 6, 3139–3157, <https://doi.org/10.1785/0120200126>

- Bindi D., Spallarossa D., Picozzi M., Oth A., Morasca P. and Mayeda K., (2023a). *The Community Stress-Drop Validation Study—Part I: Source, Propagation, and Site Decomposition of Fourier Spectra*. *Seismological Research Letters*, 2023, 94, 4, 1980–1991. <https://doi.org/10.1785/0220230019>
- Bindi D., Spallarossa D., Picozzi M., Oth A., Morasca P. and Mayeda K., (2023b). *The Community Stress-Drop Validation Study—Part II: Uncertainties of the Source Parameters and Stress Drop Analysis*, *Seismological Research Letters*, 2023, 94, 4, 1992–2002, <https://doi.org/10.1785/0220230020>
- Bindi D. and Kotha S., (2020). *Spectral decomposition of the Engineering Strong Motion (ESM) flat file: regional attenuation, source scaling and Arias stress drop*, *Bulletin of Earthquake Engineering*, 1-26, <https://doi.org/10.1007/s10518-020-00796-1>
- Bindi D., Zaccarelli R., Razafindrakoto H.N.T, Yen M.H. and Cotton F., (2023c). *Empirical shaking scenarios for Europe: a feasibility study*. *Geophysical Journal International*, 232, 2, 990–1005, <https://doi.org/10.1093/gji/ggac382>
- Boatwright J., Fletcher J.B. and Fumal T.E., (1991). *A general inversion scheme for source, site, and propagation characteristics using multiple recorded sets of moderate-sized earthquakes*. *Bull. Seismol. Soc. Am.*, 81, 5, 1754–1782, <https://doi.org/10.1785/BSSA0810051754>
- Boore D.M. and Boatwright J., (1984). *Average body-wave radiation coefficients*. *Bull. Seism. Soc. Am.*, 74, 2035-2039. <https://doi.org/10.1785/BSSA0740051615>
- Brune J.N., (1970). *Tectonic stress and the spectra of seismic shear waves from earthquakes*. *J. Geophys. Res.*, 75, 4997–5009, <https://doi.org/10.1029/JB075i026p04997>
- Brune J.N., (1971). *Correction to Tectonic Stress and the Spectra of Seismic Shear Waves from Earthquakes*. *Journal of Geophysical Research.*, 76, 5002, <http://dx.doi.org/10.1029/JB076i020p05002>
- Castro R.R., Anderson J. G. and Singh S. K., (1990). *Site response, attenuation and source spectra of S waves along the Guerrero, Mexico, subduction zone*. *Bull. Seismol. Soc. Am.*, 80, 1481–1503, <https://doi.org/10.1785/BSSA08006A1481>
- Drouet S., Chevrot S., Cotton F. and Souriau A., (2008). *Simultaneous inversion of source spectra, attenuation parameters, and site responses: Application to the data of the French accelerometric network*. *Bull. Seismol. Soc. Am.*, 98, 198–219, <https://doi.org/10.1785/0120060215>
- Kanamori H., (1977). *The energy release in great earthquakes*. *Journal of geophysical research.*, 82, 2981-2987. <https://doi.org/10.1029/JB082i020p02981>
- Kramer S. L., (1996). *Geotechnical Earthquake Engineering*. Prentice Hall, Upper Saddle River, New Jersey, 653 pp. <https://istasazeh-co.com/wp-content/uploads/2022/05/Geotechnical-Earthquake-Engineering.pdf>
- Konno K. and Ohmachi T., (1998). *Ground-motion characteristics estimated from spectral ratio between horizontal and vertical components of microtremor*. *Bulletin of the Seismological Society of America*, 88, 1, 228-241, <https://doi.org/10.1785/BSSA0880010228>
- Hwang R.D., Huang Y.L., Chang W.Y., Chang, Wen-Yen, Lin C., Lin C., Wang S., Chan J.B., Chang J.P. and Lin T.W., (2022). *Radiated seismic energy from the 2021 ML 5.8 and ML 6.2 Shoufeng (Hualien), Taiwan, earthquakes and their aftershocks*, *TAO* 33, 19, <https://doi.org/10.1007/s44195-022-00020-4>
- Ide S. and Beroza G. C., (2001). *Does apparent stress vary with earthquake size?* *Geophysical Research Letters*, 28, 17, 3349 - 3352, <https://doi.org/10.1029/2001GL013106>
- Izutani Y. and Kanamori H., (2001). *Scale-dependence of seismic energy-to-moment ratio for strike-slip earthquakes in Japan*. *Geophys. Res. Lett.*, 28, 4007-4010, <https://doi.org/10.1029/2001GL013402>
- Oth A., Bindi D., Parolai S. and Di Giacomo D., (2011). *Spectral analysis of K-NET and KiK-net data in Japan, Part II: On attenuation characteristics, source spectra, and site response of borehole and surface stations*. *Bull. Seismol. Soc. Am.*, 101, 2, 667–687. <https://doi.org/10.1785/0120100135>

- Savage J. C. and Wood M. D., (1971). *The relation between apparent stress and stress drop*. Bull. Seism. Soc. Am., 61,5, 1381–1386. <https://doi.org/10.1785/BSSA0610051381>
- Wyss M., (1970). *Apparent Stresses of Earthquakes on Ridges compared to Apparent Stresses of Earthquakes in Trenches*. Geophysical Journal International., 19, 5, 479–484, <https://doi.org/10.1111/j.1365-246X.1970.tb00153.x>.

Sitography

- [1] <https://gitlab.rm.ingv.it/inversion/gitpy/-/releases>
- [2] <https://docs.anaconda.com/free/anaconda/install/windows/>
- [3] <https://conda.anaconda.org/conda-forge/>
- [4] <https://docs.anaconda.com/free/navigator/tutorials/manage-packages/>
- [5] https://gitlab.rm.ingv.it/inversion/gitpy/-/tree/GITpy?ref_type=heads
- [6] <https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.lsqr.html>
- [7] https://gitlab.rm.ingv.it/inversion/gitpy/-/blob/GITpy/README.md?ref_type=heads

QUADERNI di GEOFISICA

ISSN 1590-2595

<https://istituto.ingv.it/it/le-collane-editoriali-ingv/quaderni-di-geofisica.html/>

I QUADERNI DI GEOFISICA (QUAD. GEOFIS.) accolgono lavori, sia in italiano che in inglese, che diano particolare risalto alla pubblicazione di dati, misure, osservazioni e loro elaborazioni anche preliminari che necessitano di rapida diffusione nella comunità scientifica nazionale ed internazionale. Per questo scopo la pubblicazione on-line è particolarmente utile e fornisce accesso immediato a tutti i possibili utenti. Un Editorial Board multidisciplinare ed un accurato processo di peer-review garantiscono i requisiti di qualità per la pubblicazione dei contributi. I QUADERNI DI GEOFISICA sono presenti in "Emerging Sources Citation Index" di Clarivate Analytics, e in "Open Access Journals" di Scopus.

QUADERNI DI GEOFISICA (QUAD. GEOFIS.) welcome contributions, in Italian and/or in English, with special emphasis on preliminary elaborations of data, measures, and observations that need rapid and widespread diffusion in the scientific community. The on-line publication is particularly useful for this purpose, and a multidisciplinary Editorial Board with an accurate peer-review process provides the quality standard for the publication of the manuscripts. QUADERNI DI GEOFISICA are present in "Emerging Sources Citation Index" of Clarivate Analytics, and in "Open Access Journals" of Scopus.

RAPPORTI TECNICI INGV

ISSN 2039-7941

<https://istituto.ingv.it/it/le-collane-editoriali-ingv/rapporti-tecnici-ingv.html/>

I RAPPORTI TECNICI INGV (RAPP. TEC. INGV) pubblicano contributi, sia in italiano che in inglese, di tipo tecnologico come manuali, software, applicazioni ed innovazioni di strumentazioni, tecniche di raccolta dati di rilevante interesse tecnico-scientifico. I RAPPORTI TECNICI INGV sono pubblicati esclusivamente on-line per garantire agli autori rapidità di diffusione e agli utenti accesso immediato ai dati pubblicati. Un Editorial Board multidisciplinare ed un accurato processo di peer-review garantiscono i requisiti di qualità per la pubblicazione dei contributi.

RAPPORTI TECNICI INGV (RAPP. TEC. INGV) publish technological contributions (in Italian and/or in English) such as manuals, software, applications and implementations of instruments, and techniques of data collection. RAPPORTI TECNICI INGV are published online to guarantee celerity of diffusion and a prompt access to published data. A multidisciplinary Editorial Board and an accurate peer-review process provide the quality standard for the publication of the contributions.

MISCELLANEA INGV

ISSN 2039-6651

https://istituto.ingv.it/it/le-collane-editoriali-ingv/miscellanea-ingv.html

MISCELLANEA INGV (MISC. INGV) favorisce la pubblicazione di contributi scientifici riguardanti le attività svolte dall'INGV. In particolare, MISCELLANEA INGV raccoglie reports di progetti scientifici, proceedings di convegni, manuali, monografie di rilevante interesse, raccolte di articoli, ecc. La pubblicazione è esclusivamente on-line, completamente gratuita e garantisce tempi rapidi e grande diffusione sul web. L'Editorial Board INGV, grazie al suo carattere multidisciplinare, assicura i requisiti di qualità per la pubblicazione dei contributi sottomessi.

MISCELLANEA INGV (MISC. INGV) favours the publication of scientific contributions regarding the main activities carried out at INGV. In particular, MISCELLANEA INGV gathers reports of scientific projects, proceedings of meetings, manuals, relevant monographs, collections of articles etc. The journal is published online to guarantee celerity of diffusion on the internet. A multidisciplinary Editorial Board and an accurate peer-review process provide the quality standard for the publication of the contributions.

Coordinamento editoriale

Francesca DI STEFANO
Istituto Nazionale di Geofisica e Vulcanologia

Progetto grafico

Barbara ANGIONI
Istituto Nazionale di Geofisica e Vulcanologia

Impaginazione

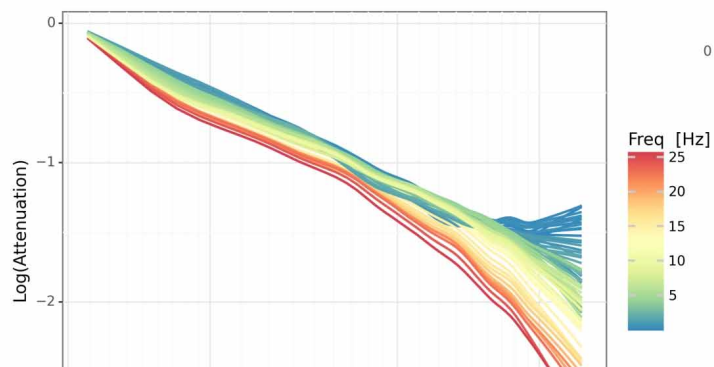
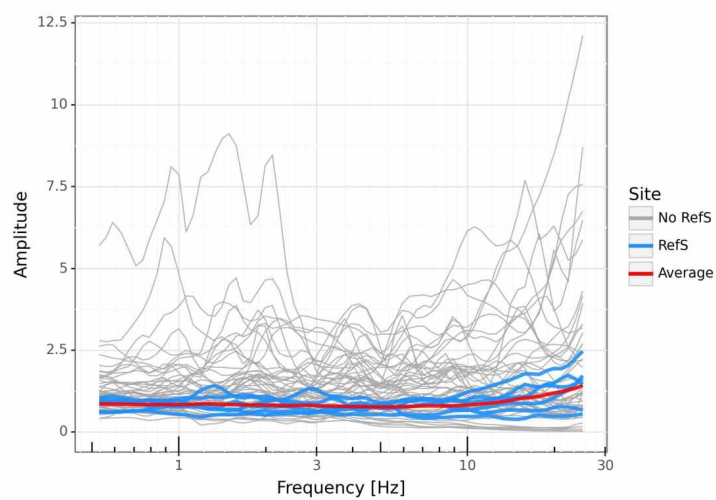
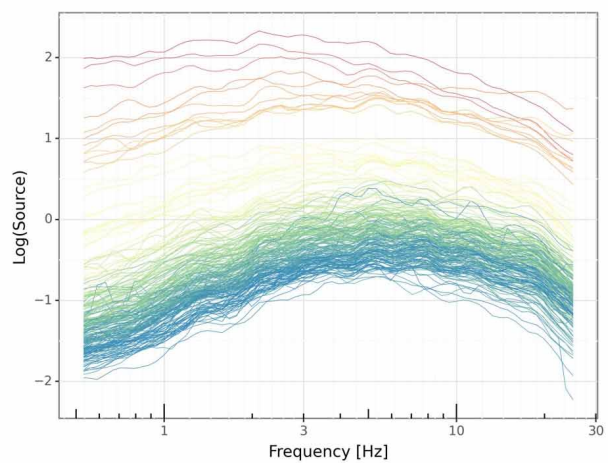
Barbara ANGIONI
Patrizia PANTANI
Massimiliano CASCONI
Istituto Nazionale di Geofisica e Vulcanologia

©2025

Istituto Nazionale di Geofisica e Vulcanologia
Via di Vigna Murata, 605
00143 Roma
tel. +39 06518601
www.ingv.it



Creative Commons Attribution 4.0 International (CC BY 4.0)



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

