SourceSpec Documentation

Release 1.5+0.gb875ed0.dirty

Claudio Satriano

Aug 01, 2022

Contents

SourceSpec is a collection of command line programs (written in Python) to determine earthquake source parameters (seismic moment M_0, corner frequency f_c) and the inelastic attenuation term (t^*), from the modeling of waveform spectra.

Other parameters (source radius r_0, stress drop $\Delta\sigma$) are computed from the inverted ones. The quality factor Q is determined from t^*.

As a bonus, local magnitude M_l is computed as well.

SourceSpec is composed of the following programs:

- `source_spec`: inverts the S-wave displacement spectra from station recordings of a single event.
- `ssp_residuals`: computes station residuals from `source_spec` output.
- `source_model`: direct spectral modelling.

Contents:

CHAPTER 1

SourceSpec

Earthquake source parameters from inversion of S-wave spectra.

1.1 Overview

`source_spec` inverts the S-wave displacement spectra from station recordings of a single event.

1.2 Spectral model

The Fourier spectrum of the S-wave displacement in far field can be modelled as the product of a source term (Brune model) and a propagation term (geometric and anelastic attenuation of body waves):

$$S(f) = \frac{1}{r} \times \frac{2R_{\Theta\Phi}}{4\pi \rho_h^{1/2} \rho_r^{1/2} \beta_h^{5/2} \beta_r^{1/2}} \times M_O \times \frac{1}{1 + \left(\frac{f}{f_c}\right)^2} \times \exp\left(\frac{-\pi r f}{Q_O V_S}\right)$$

where r is the hypocentral distance; $R_{\Theta\Phi}$ is the radiation pattern coefficient for S-waves; ρ_h and ρ_r are the medium densities at the hypocenter and at the receiver, respectively; β_h and β_r are the S-wave velocities at the hypocenter and at the receiver, respectively; M_O is the seismic moment; f is the frequency; f_c is the corner frequency; V_S is the average S-wave velocity along the wave propagation path; Q_O is the quality factor.

In `source_spec`, the observed spectra $S(f)$ are converted in moment magnitude M_w.

The first step is to multiply the spectrum for the hypocentral distance and convert them to seismic moment units:

$$M(f) \equiv r \times \frac{4\pi \rho_h^{1/2} \rho_r^{1/2} \beta_h^{5/2} \beta_r^{1/2}}{2R_{\Theta\Phi}} \times S(f) = M_O \times \frac{1}{1 + \left(\frac{f}{f_c}\right)^2} \times \exp\left(\frac{-\pi r f}{Q_O V_S}\right)$$

Then the spectrum is converted in unities of magnitude (the $Y_{data}(f)$ vector used in the inversion):

$$Y_{data}(f) \equiv \frac{2}{3} \times (\log_{10} M(f) - 9.1)$$

The data vector is compared to the teoretical model:

$$Y_{data}(f) = \frac{2}{3}\left[\log_{10}\left(M_O \times \frac{1}{1+\left(\frac{f}{f_c}\right)^2} \times \exp\left(\frac{-\pi r f}{Q_O V_S}\right)\right) - 9.1\right] =$$

$$= \frac{2}{3}(\log_{10} M_0 - 9.1) + \frac{2}{3}\left[\log_{10}\left(\frac{1}{1+\left(\frac{f}{f_c}\right)^2}\right) + \log_{10}\left(\exp\left(\frac{-\pi r f}{Q_O V_S}\right)\right)\right]$$

Finally coming to the following model used for the inversion:

$$Y_{data}(f) = M_w + \frac{2}{3}\left[-\log_{10}\left(1+\left(\frac{f}{f_c}\right)^2\right) - \pi f t^* \log_{10} e\right]$$

Where $M_w \equiv \frac{2}{3}(\log_{10} M_0 - 9.1)$ and $t^* \equiv \frac{r}{Q_O V_S}$.

The parameters to determine are M_w, f_c and t^*.

CHAPTER 2

Configuration File

Configuration file (default name: `source_spec.conf`) is a plain text file with keys and values in the form `key = value`. Comment lines start with `#`.

Some fields are comma-separated lists: even if only one element is specified, a comma is always required (e.g., `ignore_stations = STA01,`).

Here is the default config file, generated through `source_spec -S`:

```
# Config file for source_spec

# GENERAL PARAMETERS --------
# Print debug information
DEBUG = False

# Show interactive plots (slower)
PLOT_SHOW = False
# Save plots to disk
PLOT_SAVE = True
# Plot file format: 'png', 'pdf' or 'pdf_multipage'
PLOT_SAVE_FORMAT = png

# Channel naming for mis-oriented channels (vertical, horiz1, horiz2):
# Example:
#   mis_oriented_channels = Z,1,2
mis_oriented_channels = None

# Option to specify non standard instrument codes (e.g., "L" for accelerometer)
instrument_code_acceleration = None
instrument_code_velocity = None

# For more complex network.station.location.channel (SCNL) naming scenarios,
# you can provide a file, in json format, with traceid (SCNL) mapping
traceid_mapping_file = None

# List of traceids to ignore.
```

(continues on next page)

(continued from previous page)

```
# Use network.station.location.channel; wildcards are accepted
# Example:
#   ignore_traceids = FR.CIEL.*.*, AM.RA0D3.00.*
ignore_traceids = None

# List of traceids to use.
# Use network.station.location.channel; wildcards are accepted
# Example:
#   use_traceids = FR.CIEL.*.*, AM.RA0D3.00.*
use_traceids = None

# Maximum epicentral distance (km) to process a trace
max_epi_dist = None

# Optionally, it is possible to read event information and traces
# from a pickle file.
# When using a pickle catalog, you must use the "--evid" command line
# option to select an event from the catalog and you can use the "--station"
# option to select a station.
pickle_catalog = None
# If you have custom classes in your pickle catalog which are not in your
# system path, specify here the path to python files containing
# class definitions.
pickle_classpath = None

# Directory or file for station metadata.
# It can be one ore more files in one of the following formats:
#  StationXML, dataless SEED, SEED RESP
# Note that SEED RESP does not contain station coordinates, which should
# therefore be in the trace header (traces in SAC format)
station_metadata = None

# Alternatively, a directory with PAZ files can be specified:
# Note that PAZ files do not contain station coordinates, which should
# therefore be in the trace header (traces in SAC format)
paz = None

# It is also possible to provide a constant sensitivity (i.e., flat sensor
# response curve) as a numerical value or a combination of SAC header fields
# (in this case, traces must be in SAC format).
# This parameter overrides the response curve computed from metadata or from
# PAZ files. Leave it to None to compute sensor response from metadata or PAZ.
# Examples:
#  sensitivity = 1
#  sensitivity = 1e3
#  sensitivity = resp0
#  sensitivity = resp1*resp2
#  sensitivity = user3/user2
sensitivity = None

# Database file for storing output parameters (optional):
database_file = None

# Correct_instrumental_reponse (optional, default=True):
#   'True', 'False' or 'sensitivity only'
# If 'sensitivity only', traces are not fully deconvolved
# for the instrumental response: only the
```

(continues on next page)

(continued from previous page)

```
# sensitivity is corrected (faster, especially
# on a large number of traces).
correct_instrumental_response = True

# Trace units.
# Leave it to 'auto' to let the code decide, based on instrument type.
# Manually set it to 'disp', 'vel' or 'acc' if you have already preprocessed
# the traces.
trace_units = auto
# -------- GENERAL PARAMETERS


# TIME WINDOW PARAMETERS --------
# P and S wave velocity (in km/s) for travel time calculation
# (if None, the global velocity model 'iasp91' is used)
vp_tt = None
vs_tt = None
# As an alternative, a directory containing NonLinLoc travel time grids
# can be specified.
# Note that reading NonLinLoc grids takes time. For simple 1D models, you
# can speed up considerably the process using a generic station
# named "DEFAULT". The coordinates of this default station are not important,
# since they will be superseded by each station's coordinates.
NLL_time_dir = None

# Arrival tolerances (in seconds) to accept a manual P or S pick
p_arrival_tolerance = 4.0
s_arrival_tolerance = 4.0

# Start time (in seconds) of the noise window, respect to the P arrival time
pre_p_time = 6.0

# Start time (in seconds) of the S-wave window, respect to the S arrival time
pre_s_time = 1.0

# Length (in seconds) for both noise and S-wave windows
win_length = 5.0
# -------- TIME WINDOWS PARAMTERS


# SPECTRUM PARAMETERS --------
# Wave type to analyse: 'S', 'SH' or 'SV'
# If 'SH' or 'SV' are selected, traces are rotated in the radial-transverse
# system. Transverse component is used for 'SH', radial (and vertical)
# components are used for 'SV'
wave_type = S

# Integrate in time domain (default: integration in spectral domain)
time_domain_int = False

# Ignore vertical components
ignore_vertical = False

# Taper half width: between 0 (no taper) and 0.5
taper_halfwidth = 0.05

# Spectral window length (seconds)
```

(continues on next page)

(continued from previous page)

```
# Signal is tapered, and then zero padded to
# this window length, so that the spectral
# sampling is fixed to 1/spectral_win_length.
# Comment out (or set to None) to use
# S-wave window as spectral window length.
spectral_win_length = None

# Spectral smoothing window width in frequency decades
# (i.e., log10 frequency scale).
# Example:
#  spectral_smooth_width_decades=1 means a width of 1 decade
#  (generally, too large, producing a spectrum which is too smooth).
#  spectrum(f0) is smoothed using values between f1 and f2, so that
#  log10(f1)=log10(f0)-0.5 and log10(f2)=log10(f0)+0.5
#    i.e.,
#  f1=f0/(10^0.5) and f2=f0*(10^0.5)
#    or,
#  f2/f1=10 (1 decade width)
# Default value of 0.2 is generally a good choice
spectral_smooth_width_decades = 0.2

# Residuals file path
# (a pickle file with the mean residuals per station,
# used for station correction):
residuals_filepath = None

# Band-pass frequencies for accelerometers and velocimeters (Hz).
# Use bp_freqmin_STATION and bp_freqmax_STATION to provide
# filter frequencies for a specific STATION code.
# TODO: calculate from sampling rate?
bp_freqmin_acc    = 1.0
bp_freqmax_acc    = 50.0
bp_freqmin_shortp = 1.0
bp_freqmax_shortp = 40.0
bp_freqmin_broadb = 0.5
bp_freqmax_broadb = 40.0

# Spectral windowing frequencies for accelerometers and velocimeters (Hz)
# (spectra will be cut between these two frequencies)
# Use freq1_STATION and freq2_STATION to provide
# windowing frequencies for a specific STATION code.
freq1_acc     = 1.0
freq2_acc     = 30.0
freq1_shortp  = 1.0
freq2_shortp  = 30.0
freq1_broadb  = 0.5
freq2_broadb  = 30.0
# -------- SPECTRUM PARAMETERS


# SIGNAL/NOISE PARAMETERS --------
# Minimum rms (in trace units before instrument corrections)
# to consider a trace as noise
rmsmin = 0.

# Time domain S/N ratio min
sn_min = 0
```

(continues on next page)

(continued from previous page)

```
# Maximum accepted percentage of clipped data respect to the total trace lenght
clip_max_percent = 5.0

# Maximum gap length for the whole trace, in seconds
gap_max = None
# Maximum overlap length for the whole trace, in seconds
overlap_max = None

# Sspectral S/N ratio min, below which a spectrum will be skipped
spectral_sn_min = 0
# Frequency range (Hz) to compute the spectral S/N ratio
# (comment out or use None to indicate the whole frequency range)
spectral_sn_freq_range = None
# -------- SIGNAL/NOISE PARAMETERS


# INVERSION PARAMETERS --------
# P and S wave velocity close to the source (km/s)
vp = 5.5
vs = 3.2
# As an alternative, a directory containing a NonLinLoc model can be specified
NLL_model_dir = None
# Density (kg/m3):
rho = 2500
# S-wave average radiation pattern coefficient:
rps = 0.62
# Radiation pattern from focal mechanism, if available
rps_from_focal_mechanism = False

# Weighting type: 'noise' or 'frequency'
weighting = noise
# Parameters for 'frequency' weighting (ignored for 'noise' weighting):
#   weight for f<=f_weight (Hz)
#   1      for f> f_weight (Hz)
f_weight = 7.
weight = 10.

# Initial value for t_star (seconds)
t_star_0 = 0.045
# Try to invert for t_star_0.
# If the inverted t_star_0 is non-positive, then fixed t_star_0 will be used
invert_t_star_0 = False
# Allowed variability around inverted t_star_0 in the main inversion
# (expressed as a fraction of t_star_0, between 0 and 1).
# If the inverted t_star_0 is non-positive, then t_star_min_max is used
# (see below).
t_star_0_variability = 0.1
# Allowed variability around Mw_0 during the main inversion,
# in units of magnitude. Mw bounds will be:
#   Mw_0 - Mw_0_variability, Mw_0 + Mw_0_variability
Mw_0_variability = 0.1
# Inversion algorithm:
# TNC: truncated Newton algorithm (with bounds)
# LM: Levenberg-Marquardt algorithm
# (warning: Trust Region Reflective algorithm will be used instead if
#  bounds are provided)
```

(continues on next page)

(continued from previous page)

```
# BH: basin-hopping algorithm
# GS: grid search
# IS: importance sampling of misfit grid, using k-d tree
inv_algorithm = TNC
# Parameter bounds:
# specify bounds as a list, ex.:
#   fc_min_max = 0.1, 40
# (comment out or use None to indicate no bound)
# If not specified, fc bounds will be autoset
fc_min_max = None
# t_star_min_max does not superseed t_star_0_variability
t_star_min_max = None
# optional : Qo bounds (converted into t_star bounds in the code).
# (comment out or use None to indicate no bound)
# Note: if you want to explore negative t_star values, you have to specify
# -Qo_min, Qo_min. This beacause t_star is proportional to 1/Qo.
# Example, for searching only positive t_star values:
#   Qo_min_max = 10, 1000
# If you want to search also negative t_star values:
#   Qo_min_max = -10, 10
Qo_min_max = None
# -------- INVERSION PARAMETERS

# POST-INVERSION PARAMETERS --------
# Post-inversion bounds: use this bounds to reject certain inversion
# results, per station.
# Sometimes it is better to be more permissive with inversion parameters and
# reject "bad" solutions after the inversion, rather than forcing the
# inversion to converge within strict bounds.
# fc bounds, in Hz
pi_fc_min_max = None
# t_star bounds, in s
pi_t_star_min_max = None
# Brune stress drop bounds, in MPa
pi_bsd_min_max = None
# Maximum acceptable misfit between inverted and observed spectrum
pi_misfit_max = None
# -------- POST-INVERSION PARAMETERS


# RADIATED-ENERGY PARAMETERS --------
# Maximum frequency (Hz) to measure radiated energy Er
# (above this frequency, the finite-band correction
# of Di Bona & Rovelli, 1988, will be applied)
max_freq_Er = None
# -------- RADIATED-ENERGY PARAMETERS


# LOCAL MAGNITUDE PARAMETERS --------
compute_local_magnitude = False
# Local magnitude parameters:
#   ml = log10(A) + a * log10(R/100) + b * (R-100) + c
# where A is the maximum W-A amplitude (in mm)
# and R is the hypocentral distance (in km)
# Default values (for California) are:
#   a = 1., b = 0.00301, c = 3.
a = 1.
```

(continues on next page)

(continued from previous page)

```
b = 0.00301
c = 3.
# Band-pass filtering frequencies (Hz) for local magnitude
ml_bp_freqmin = 0.1
ml_bp_freqmax = 20.0
# -------- LOCAL MAGNITUDE PARAMETERS


# AVERAGES PARAMETERS --------
# Reject outliers before averaging, using the IQR method.
# IQR is the interquartile range Q3-Q1, where Q1 is the 25% percentile
# and Q3 is the 75% percentile.
# Values that are smaller than (Q1 - nIQR*IQR) or larger than (Q3 + nIQR*IQR)
# will be rejected as outliers.
# Set nIQR to None to disable outlier rejection.
# Note: this parameter also controls the position of "whiskers" on the source
# parameter box plots.
nIQR = 1.5
# -------- AVERAGES PARAMETERS


# PLOT PARAMETERS --------
# Plots an extra synthetic spectrum with no attenuation
plot_spectra_no_attenuation = False
# Plots an extra synthetic spectrum with no fc
plot_spectra_no_fc = False
# Max number of rows in plots
plot_spectra_maxrows = 3
plot_traces_maxrows = 3
# Plot ignored traces (low S/N)
plot_traces_ignored = True
# Plot ignored spectra (low S/N)
plot_spectra_ignored = True
# Plot station map
plot_station_map = False
# Plot station names on map
plot_station_names_on_map = False
# Text size for station names
plot_station_text_size = 8
# Coastline resolution
# Use None to let the code autoset the coastline resolution.
# Otherwhise choose one of: 'full', 'high', 'intermediate', 'low' or 'crude'
plot_coastline_resolution = None
# Zoom level for map tiles
# Use None to let the code autoset the zoom level
# Otherwhise choose an integer beteen 1 (minimum zoom) and 18 (maximum zoom)
# Note: for zoom levels larger than 11, some map tiles could be missing
plot_map_tiles_zoom_level = None
# -------- PLOT PARAMETERS


# HTML REPORT --------
# Generate an HTML page summarizing the results of this run
html_report = False
# -------- HTML REPORT
```

CHAPTER 3

SourceSpec API

SourceSpec has a modular structure. Each module corresponds to a specific function or class of functions.

SourceSpec modules are presented below, following the logical order on which they're used within `source_spec.py`.

3.1 ssp_setup

Setup functions for sourcespec.

copyright 2012 Claudio Satriano <satriano@ipgp.fr>

2013-2014 Claudio Satriano <satriano@ipgp.fr>, Emanuela Matrullo <matrullo@geologie.ens.fr>, Agnes Chounet <chounet@ipgp.fr>

2015-2022 Claudio Satriano <satriano@ipgp.fr>

license CeCILL Free Software License Agreement v2.1 (http://www.cecill.info/licences.en.html)

`ssp_setup.`**`configure`**`(`*options*, *progname*`)`
Parse command line arguments and read config file.

Returns a `Config` object.

`ssp_setup.`**`dprint`**`(`*string*`)`
Print debug information.

`ssp_setup.`**`move_outdir`**`(`*config*`)`
Move outdir to a new dir named from evid.

`ssp_setup.`**`remove_old_outdir`**`(`*config*`)`
Try to remove the old outdir.

`ssp_setup.`**`save_config`**`(`*config*`)`
Save config file to output dir.

`ssp_setup.`**`setup_logging`**(*config*, *basename=None*, *progname='source_spec'*)

Set up the logging infrastructure.

This function is typically called twice: the first time without basename and a second time with a basename (typically the eventid). When called the second time, the previous logfile is renamed using the given basename.

3.2 ssp_read_traces

Read traces in multiple formats of data and metadata.

`ssp_read_traces.`**`read_traces`**(*config*)

Read traces, store waveforms and metadata.

3.3 ssp_process_traces

Trace processing for sourcespec.

`ssp_process_traces.`**`process_traces`**(*config*, *st*)

Remove mean, deconvolve and ignore unwanted components.

3.4 ssp_build_spectra

Build spectral objects.

`ssp_build_spectra.`**`build_spectra`**(*config*, *st*)

Build spectra and the spec_st object.

Computes S-wave (displacement) spectra from accelerometers and velocimeters, uncorrected for attenuation, corrected for instrumental constants, normalized by hypocentral distance.

3.5 ssp_plot_traces

Trace plotting routine.

`ssp_plot_traces.plot_traces` (*config*, *st*, *spec_st=None*, *ncols=4*, *block=True*, *async_plotter=None*)

Plot traces in the original instrument unit (velocity or acceleration).

Display to screen and/or save to file.

3.6 ssp_local_magnitude

Local magnitude calculation for sourcespec.

`ssp_local_magnitude.local_magnitude` (*config*, *st*, *proc_st*, *sourcepar*)

Compute local magnitude from max absolute W-A amplitude.

3.7 ssp_inversion

3.8 ssp_spectral_model

Spectral model and objective function.

`ssp_spectral_model.objective_func` (*xdata*, *ydata*, *weight*)

Objective function generator for bounded inversion.

`ssp_spectral_model.spectral_model` (*freq*, *Mw*, *fc*, *t_star*, *alpha=1.0*)

Spectral model.

$$Y_{data} = M_w + \frac{2}{3}\left[-\log_{10}\left(1+\left(\frac{f}{f_c}\right)^2\right) - \pi\, f t^* \log_{10} e\right]$$

see *source_spec* for a detailed derivation of this model.

3.9 ssp_residuals

Spectral residual routine for sourcespec.

copyright

2013-2014 Claudio Satriano <satriano@ipgp.fr>, Agnes Chounet <chounet@ipgp.fr>

2015-2022 Claudio Satriano <satriano@ipgp.fr>

license CeCILL Free Software License Agreement v2.1 (http://www.cecill.info/licences.en.html)

`ssp_residuals.`**`spectral_residuals`**(*config*, *spec_st*, *sourcepar*)

Compute spectral residuals with respect to an average spectral model.

Saves a stream of residuals to disk using pickle.

3.10 ssp_correction

Spectral station correction calculated from ssp_residuals.

copyright

2013-2014 Claudio Satriano <satriano@ipgp.fr>, Agnes Chounet <chounet@ipgp.fr>

2015-2022 Claudio Satriano <satriano@ipgp.fr>

license CeCILL Free Software License Agreement v2.1 (http://www.cecill.info/licences.en.html)

`ssp_correction.`**`station_correction`**(*spec_st*, *config*)

Correct spectra using station-average residuals.

Residuals are obtained from a previous run.

3.11 ssp_output

3.12 ssp_plot_spectra

Spectral plotting routine.

copyright 2012 Claudio Satriano <satriano@ipgp.fr>

2013-2014 Claudio Satriano <satriano@ipgp.fr>, Emanuela Matrullo <matrullo@geologie.ens.fr>

2015-2022 Claudio Satriano <satriano@ipgp.fr>

license CeCILL Free Software License Agreement v2.1 (http://www.cecill.info/licences.en.html)

`ssp_plot_spectra.`**`plot_spectra`**(*config*, *spec_st*, *specnoise_st=None*, *ncols=4*, *stack_plots=False*, *plot_type='regular'*, *async_plotter=None*)

Plot spectra for signal and noise.

Display to screen and/or save to file.

3.13 ssp_util

Utility functions for sourcespec.

`ssp_util.`**`bsd`**(*Mo_in_N_m*, *ra_in_m*)
Compute Brune stress drop in MPa.

Madariaga (2009), doi:10.1007/978-1-4419-7695-6_22, eq. 27

`ssp_util.`**`get_vel`**(*lon*, *lat*, *depth*, *wave*, *config*)
Get velocity at a given point from NonLinLoc grid.

Fall back to config.vp or config.vs if no grid is defined.

`ssp_util.`**`hypo_dist`**(*trace*)
Compute hypocentral and epicentral distance (in km) for a trace.

`ssp_util.`**`mag_to_moment`**(*magnitude*)
Convert magnitude to moment.

`ssp_util.`**`moment_to_mag`**(*moment*)
Convert moment to magnitude.

`ssp_util.`**`quality_factor`**(*hyp_dist_in_km*, *vs_in_km_per_s*, *t_star_in_s*)
Compute quality factor from t_star, distance and vs.

`ssp_util.`**`select_trace`**(*stream*, *traceid*, *instrtype*)
Select trace from stream using traceid and instrument type.

`ssp_util.`**`source_radius`**(*fc_in_hz*, *vs_in_m_per_s*)
Compute source radius in meters.

Madariaga (2009), doi:10.1007/978-1-4419-7695-6_22, eq. 31

3.14 spectrum

A Spectrum() class defined as a modification of the ObsPy class Trace().

Provides the high-level function do_spectrum() and the low-level funciton do_fft().

`spectrum.`**`do_fft`**(*signal*, *delta*)
Compute the complex Fourier transform of a signal.

`spectrum.`**`do_spectrum`**(*trace*)
Compute the spectrum of an ObsPy Trace object.

3.15 config

Config class for sourcespec.

license CeCILL Free Software License Agreement v2.1 (http://www.cecill.info/licences.en.html)

class `config.Config`

Config class for sourcespec.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

c

s

Index