# An introduction to SeisTomoPy

S. Durand, R. Abreu & C. Thomas

# Contents


**1 What is SeisTomoPy** **3**

**2 Requirements** **5**

**3 Installation** **6**

**4 Reported bugs** **6**

**5 How to use SeisTomoPy class** **6**
- 5.1 Cross sections . . . 7
- 5.2 Maps . . . 8
- 5.3 Spectrum . . . 10
- 5.4 Correlations . . . 13
- 5.5 Paths . . . 18
- 5.6 Travel times . . . 21

**6 How to use SeisTomoPy GUI** **23**
- 6.1 Cross sections . . . 24
- 6.2 Maps . . . 26
- 6.3 Spectrum . . . 27
- 6.4 Correlations . . . 28
- 6.5 Paths . . . 29
- 6.6 Travel times . . . 31

**7 Jupyter notebooks** **32**

**8 Model and parameter options** **32**

**9 Creating a new tomographic model file** **33**

**10 Changing plate boundary model and hot spots and earthquake catalogs** **33**
- 10.1 Plate boundary model . . . 34
- 10.2 Hot spot catalog . . . 34
- 10.3 Earthquake catalog . . . 34

**11 Input file formats for Spectrum, Correlations and Paths** **35**

**12 Future developments** **36**

**13 Reference** **37**

# 1 What is SeisTomoPy

SeisTomoPy is a Python tool that facilitates the use of a suite of tomographic models available to the public, with a single program. SeisTomoPy provides six tools that allow to visualize tomographic models, compare them and extract information for further scientific purposes. The tool comes with a graphical interface with intuitive buttons and simple parameters but the same information can also be gained by using the Python class that can be run routinely in Python scripts. SeisTomoPy is suited for global and spherical tomographic models and is provided with a default list of recent tomographic models. However, the user can also upload additional models if desired.

SeisTomoPy includes several recent tomographic models that are available to the community. In Table 1 we summarize the characteristics of the different models included in the first version of SeisTomoPy. For detailed information on each model the reader is referred to the original publication.

Associated to this first library of models, we provide six tools that will aid extracting information about each of these models:

- Cross section for generating cross sections and extracting values of $V_s$, $V_p$ and density as input for AxiSEM to generate synthetic seismograms,
- Map for producing global tomographic maps at a given depth,
- Spectrum for computing the amplitude spectrum at any depth,
- Correlation for computing the correlation at any depth between the different models,
- Path for plotting wave paths on the top of tomographic models,
- Travel Time for computing travel times of body waves for the different tomographic models.

SeisTomoPy has been developed for saving the generated results from each of the functions in output files or figures that can be used for further scientific studies.

It is also possible to upload a model if desired. A package is provided in order to create new model files. Our choice was to develop models in spherical harmonics which:

- 1) enables to store the models quite efficiently, and avoids to deal with huge model files,
- 2) is an efficient way to compute any section, map, spectrum, etc, in a reasonable amount of time.

This means that certain tomographic models are not suitable to be included in the tool, in particular those with irregular parametrization and regional models.

Table 1: Summary of the characteristics of every tomographic model included in SeisTomoPy. Abbreviations in the table: SW - Surface waves, BW - body waves, NM - normal modes, T - period, Sph. Harm. - spherical harmonics, $V_s, V_p, V_{sh}, V_{sv}$ - shear velocity, compressional velocity, horizontal shear velocity, vertical shear velocity, $\nu_\alpha, \nu_\rho$ - scaling factors such that $d\ln(V_p) = \nu_\alpha d\ln(V_s)$ and $d\ln(\rho) = \nu_\rho d\ln(V_s)$.

| Model | Data | Inverted parameter | Lateral param. | Radial param. | Theory used | Reference model | Scaling laws | Crustal model | Global or upper mantle |
|---|---|---|---|---|---|---|---|---|---|
| S40RTS [Ritsema et al., 2011] | SW phase velocities<br>BW travel times<br>NM splitting coefficients | $V_s$ | Sph. Harm. up to degree 40 | 21 cubic spline functions | Ray theory | PREM iso. | $\nu_\alpha$ varies from 2 (surf) to 3 (CMB)<br>$\nu_\rho = 0.3$ | Corrected CRUST2.0 [Bassin et al., 2000] | Global |
| S362WMANI+M [Moulik & Ekström, 2014] | SW phase velocities<br>Long period BW waveforms<br>BW travel times<br>NM splitting coefficients | $V_{sh}$, $V_{sv}$ | Spherical splines | 16 cubic splines discontinuous across 650 km | Ray theory | STW105 [Kustowski et al., 2008] | $\nu_\alpha = 0.55$<br>$\nu_\rho = 0$ | Corrected CRUST2.0 | Global |
| SEMUCB-WM1 [French & Romanowicz, 2014] | SW wavforms (T > 60 s)<br>BW waveforms<br>(T > 36 s and T > 32 s) | $V_{sh}$, $V_{sv}$ | Spherical splines | 20 cubic splines | 3-D synthetics and NACT | their own model [French & Romanowicz, 2014] | $\nu_\alpha = 0.5$<br>$\nu_\rho = 0$ | Corrected smoothed version CRUST2.0 | Global |
| SGLOBE-rani [Chang et al., 2015] | SW phase velocities<br>BW travel times | $V_{sh}$, $V_{sv}$ | Sph. Harm. up to degree 35 | 20 cubic splines | Ray theory | PREM aniso. | $\nu_\alpha = 0.5$<br>$\nu_\rho = 0.4$ | Corrected CRUST2.0 | Global |
| SEISGLOB1 [Durand et al., 2016] | SW phase velocities<br>NM splitting and coupling coefficients | $V_s$ | Sph. Harm. up to degree 20 | 21 cubic spline functions | Ray theory | PREM iso. | $\nu_\alpha = 0.55$<br>$\nu_\rho = 0.2$ | Inverted | Global |
| SP12RTS [Koelemeijer et al., 2016] | SW phase velocities<br>BW travel times<br>NM splitting coefficients | $V_s$, $V_p$ | Sph. Harm. up to degree 12 | 21 cubic spline functions | Ray theory | PREM iso. | $\nu_\rho = 0.3$ | Corrected CRUST2.0 | Global |
| SEISGLOB2 [Durand et al., 2017b] | SW phase velocities<br>NM splitting coefficients<br>NM splitting and coupling coefficients | $V_s$ | Sph. Harm. up to degree 40 | 21 cubic spline functions | Ray theory | PREM iso. | $\nu_\alpha = 0.55$<br>$\nu_\rho = 0.2$ | Corrected CRUST2.0 | Global |
| 3D2016-09S [?] | SW phase velocities | $V_{sv}$, $V_{sh}$ | Correlation length 200 km down to 750 km increases to 800 km down to 1000 km | Correlation length 50 km | Path average | PREM aniso. | | Corrected 3SMAC [Nataf & Ricard, 1993] | Upper mantle |

## 2 Requirements

SeisTomoPy is a Python software that can run on any version of Python from 2.6 to 3.7. However, we highly recommend to install Python 3.7, since Python 2.7 will be maintained until 1[rst] January 2020 only. You can download anaconda using this link:
https://www.anaconda.com/download/#macos

Please check that at the end of the installation of anaconda there is a `~/.bash_profile` file that looks like:

```
# added by Anaconda3 2018.12 installer
# >>> conda init >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$(CONDA_REPORT_ERRORS=false '/Users/YOURNAME/anaconda3/bin/conda'
 shell.bash hook 2> /dev/null)"
if [ $ -eq 0 ]; then?
    \eval "$__conda_setup"
else
    if [ -f "/Users/YOURNAME/anaconda3/etc/profile.d/conda.sh" ]; then
        . "/Users/YOURNAME/anaconda3/etc/profile.d/conda.sh"
        CONDA_CHANGEPS1=false conda activate base
    else
        \export PATH="/Users/YOURNAME/anaconda3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda init <<<
```

SeisTomoPy has a number of dependencies listed below.

- gfortran : GNU Fortran (MacPorts gcc48 4.8.5_0) 4.8.5
- Python 2.7, 3.5, 3.6, 3.7
- iPython 7.2.0
- matplotlib 3.0.2
- numpy 1.15.4
- obspy 1.1.0
- pyqt 5.9.2
- scipy 1.1.0
- basemap 1.2.0

- pyproj 1.9.6
- proj4 5.0.2

For installing the Python dependencies, please run :
`conda install -c conda-forge obspy h5py basemap pyqt pip pyqtgraph`

If there is any problem with the compilation of fortran source files, we recommand to install fortran using:
https://gcc.gnu.org/wiki/GFortranBinaries

# 3 Installation

Clone the git repository and install in an editable fashion:

```
$ git clone https://github.com/stephaniedurand/SeisTomoPy_V3.git
$ cd SeisTomoPy_V3
$ pip install -v -e .
```

**This will install the package on your computer and it will create a directory `SeisTomoPy_files` in your home directory that should not be modified or removed for any reason, otherwise, the package won't work.**

**It is possible to uninstall SeisTomoPy from anywhere on your computer with the command**

```
$ pip uninstall SeisTomoPy
```

# 4 Reported bugs

There is currently a problem with the projection in Maps when you want to change the center longitude. You can only use values from -180 to 0 degrees otherwise the projection fails. We are currently working on fixing this issue in the next version of SeisTomoPy.

# 5 How to use SeisTomoPy class

Launch iPython. The first thing to do is to import SeisTomoPy so that you will be able to use the various SeisTomoPy functions.

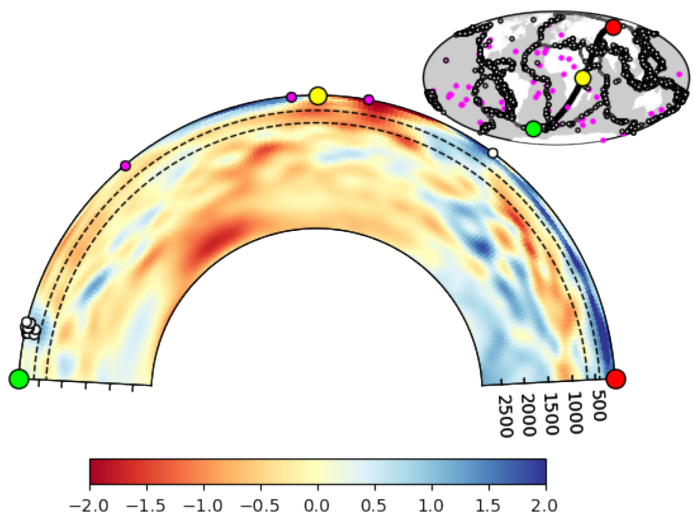
```python
import SeisTomoPy
```

## 5.1 Cross sections

The user can generate cross sections through $V_p$, $V_s$ or $\rho$ variations anywhere on Earth in some default tomographic models provided with the tool using `SeisTomoPy.cross_section_plot`. This generates directly the plot of the cross section. However, the user could also be interested in getting the values of the cross section and then perform some further calculations. This can also be achieved running `SeisTomoPy.cross_section`. The latter function returns as outputs the matrix containing the cross section ($Z$) as well as the angle ($th$) and radius vectors ($r$) that the user would need if he desires to perform further calculations with this cross section.

Below is an example of how to get a cross section beneath Africa using both class. Pink circles show hot spot locations, white circles denote earthquake locations and green, red and yellow circles the starting, ending and mid-point, respectively, along the profile.

```python
# Setting parameters
# Model to be plotted
# Choose between:
# SEISGLOB2, S40RTS, SEMUCBWM1, S362WMANIM, SEISGLOB1, SP12RTS, SGLOBE, 3D2016, MYMODEL
model = "SEISGLOB2"
# Parameter to be plotted
# Choose between: VS, VP, RHO
para = "VS"
# Latitude of the starting point of the cross section
elat = -60
# Longitude of the starting point of the cross section
elon = -49
# Latitude of the ending point of the cross section
slat = 60
# Longitude of the ending point of the cross section
slon = 119
# Depth of the cross section
depth = 2890
# Spherical harmonic degrees to be used
NSmax = 40
# Maximal velocity perturbations for the colorbar
Vmax = 2

# Running cross_section_plot
SeisTomoPy.cross_section_plot(model,para,elat,elon,slat,slon,depth,NSmax,Vmax)

# Running cross_section
Z, th, r = SeisTomoPy.cross_section(model,para,elat,elon,slat,slon,depth,NSmax)
```

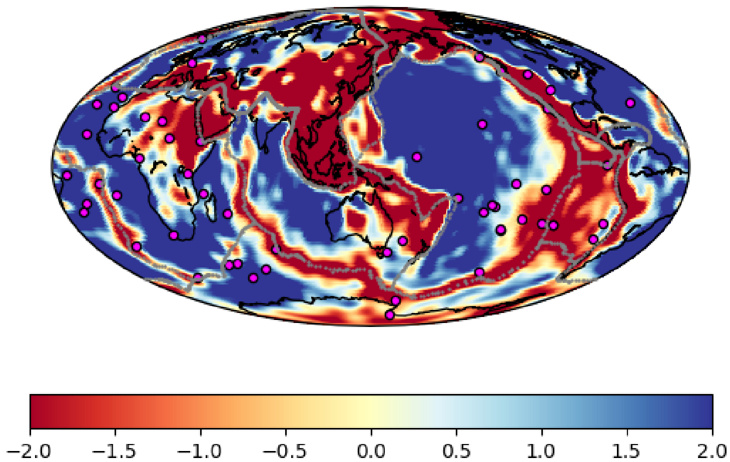


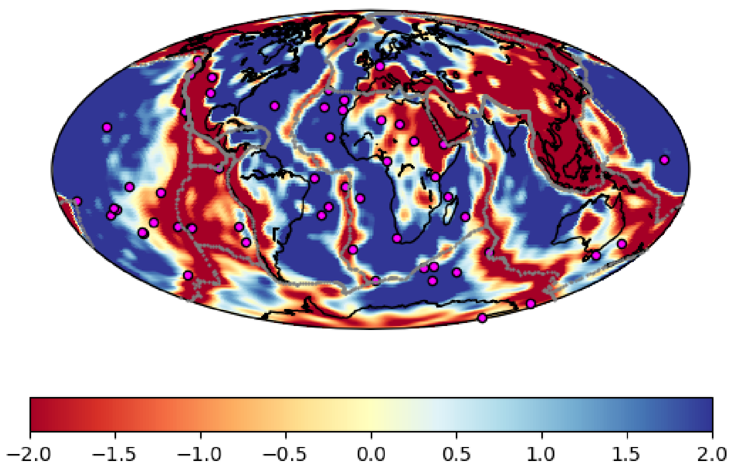

## 5.2 Maps

The user can create maps at a given depth for the whole globe using `SeisTomoPy.tomomap_plot`. This generates directly the plot of the map. It is also possible to obtain the values of the map and then perform some further calculations using `SeisTomoPy.tomomap`

Below is an example of how to obtain a map for the tomographic model 3D2016-S09 at 50 km depth using both class. Pink circles show hotspot locations and gray lines the plate boundaries.

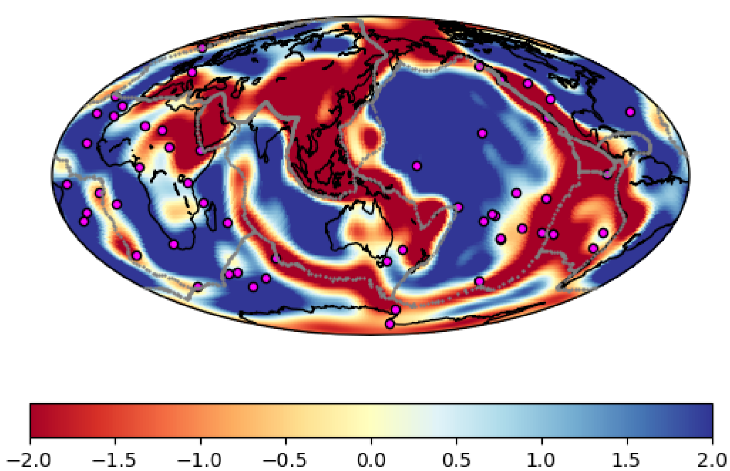
```
# Setting parameters
# Model to be plotted
model = "3D2016"
# Parameter to be plotted
para = "VS"
# Depth of the map to be plotted
depth = 50
# Spherical harmonic degrees to be used
NSmax = 60
# Central longitude
# default value 140 degrees
lon0 = 140
# Maximal velocity perturbations for the colorbar
Vmax = 2

# Running tomomap_plot
SeisTomoPy.tomomap_plot(model,para,depth,NSmax,Vmax,lon0)

# Changing the central longitude
lon0 = 0
SeisTomoPy.tomomap_plot(model,para,depth,NSmax,Vmax,lon0)

# Running tomomap
Z, lat, lon = SeisTomoPy.tomomap(model,para,depth,NSmax)
```




It is possible to change $NSmax$ to a value below 60 in order to filter the model.

```
# Filtering the model
NSmax = 18
SeisTomoPy.tomomap_plot(model,para,depth,NSmax,Vmax)
```



**These first tools can also be used with an uploaded tomographic model (see section 7 for details). The user would then refer to the new model by setting:**

```
model = "MYMODEL"
```

## 5.3 Spectrum

The user can compute the amplitude spectrum, $S(X, z, l)$, for a given model at a given depth, $z$, for a given parameter, $X$, and for a certain spherical harmonic degree, $l$, using `SeisTomoPy.spectrum` and `SeisTomoPy.spectrum_fromfile`. The first one computes the spectrum for the models included by default in SeisTomoPy while the second one enables the user to compute the spectrum in another model as long as the correct input file is uploaded. Please refer to section 11 for details about these files.

The spectrum is defined by:

$$S(X, z, l) = \sqrt{\frac{1}{4\pi} \sum_{m=-l}^{l} \left(\frac{\delta X}{X}(z)\right)_{lm} \left(\frac{\delta X}{X}(z)\right)^{\star}_{lm}}, \tag{1}$$

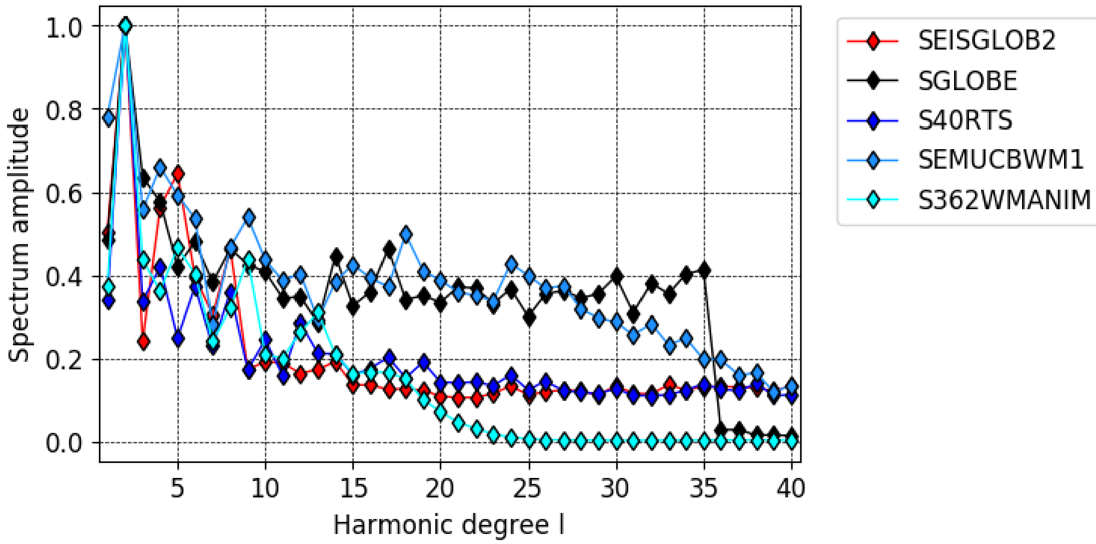
where the exponent "$\star$" denotes the complex conjugate and $\left(\frac{\delta X}{X}(z)\right)_{lm}$ the spherical harmonic coefficients of the considered parameter $X$ for a given tomographic model at depth $z$. $X$ stands for $V_s$, $V_p$ or density.

Below is first an example of obtained spectrum for various models at 520 km depth, for parameter $V_s$ and up to spherical harmonic degree 40. We then show how to get the spectrum for another model not included in SeisTomoPy.

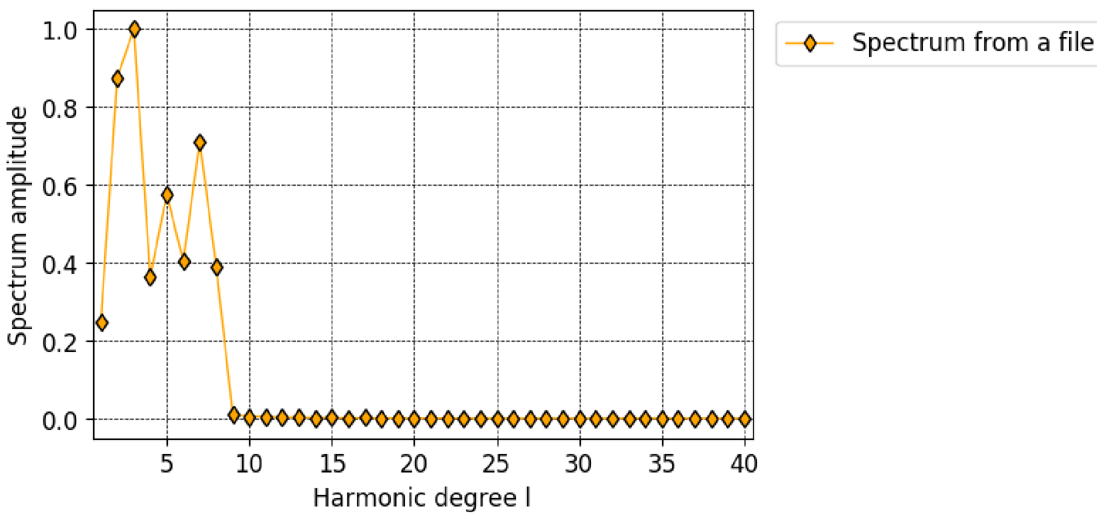
```
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print(" Example of spectrum computed in the default tomographic models ")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

# Setting parameters
# Models to be used for computing the spectrum
model1 = "SEISGLOB2"
model2 = "SGLOBE"
model3 = "S40RTS"
model4 = "SEMUCBWM1"
model5 = "S362WMANIM"
# Depth at which the spectrum is computed
depth = 520
# Parameter used for computing the spectrum
para = "VS"
# Maximum spherical harmonic degree up to which the spectrum is computed
NSmax = 40

# Running spectrum
sp1 = SeisTomoPy.spectrum(model1,para,depth,NSmax)
sp2 = SeisTomoPy.spectrum(model2,para,depth,NSmax)
sp3 = SeisTomoPy.spectrum(model3,para,depth,NSmax)
sp4 = SeisTomoPy.spectrum(model4,para,depth,NSmax)
sp5 = SeisTomoPy.spectrum(model5,para,depth,NSmax)
```

```
# Plotting the results
plt.plot(sp1[:,0], sp1[:,1]/np.amax(sp1[:,1]), linewidth=1.0,color="red", ...
        marker="d",markeredgecolor="k", label=model1)
plt.plot(sp2[:,0], sp2[:,1]/np.amax(sp2[:,1]), linewidth=1.0,color="k", ...
        marker="d",markeredgecolor="k", label=model2)
plt.plot(sp3[:,0], sp3[:,1]/np.amax(sp3[:,1]), linewidth=1.0,color="blue", ...
        marker="d",markeredgecolor="k", label=model3)
plt.plot(sp4[:,0], sp4[:,1]/np.amax(sp4[:,1]), linewidth=1.0,color="dodgerblue",...
        marker="d",markeredgecolor="k", label=model4)
plt.plot(sp5[:,0], sp5[:,1]/np.amax(sp5[:,1]), linewidth=1.0,color="cyan",...
        marker="d",markeredgecolor="k", label=model5)
plt.legend(bbox_to_anchor=(1.45, 1))
plt.xlabel("Harmonic degree l")
plt.ylabel("Spectrum amplitude")
plt.xlim([0.5, NSmax+0.5])
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.show()

print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print("  Example of spectrum computed in another model")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

# Running spectrum_fromfile
filename = "SeisTomoPy_notebook/files/input_file_spectrum.xyz"
sp_fromfile = SeisTomoPy.spectrum_fromfile(filename,NSmax)

# Plotting the results
plt.plot(sp_fromfile[:,0], sp_fromfile[:,1]/np.amax(sp_fromfile[:,1]), ...
         linewidth=1.0,color="orange",marker="d",markeredgecolor="k", ...
        label="Spectrum from a file")
plt.legend(bbox_to_anchor=(1.55, 1))
plt.xlabel("Harmonic degree l")
plt.ylabel("Spectrum amplitude")
plt.xlim([0.5, NSmax+0.5])
plt.grid(color="k", linestyle="--",linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.show()
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   Example of spectrum computed in the default tomographic models
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```




```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    Example of spectrum computed in another model
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```




It is possible to run the spectrum calculations routinely, at various depths for instance. The example below can take some time to run, it can be reduced by changing

$ depths = np.arange(100,2900,100)

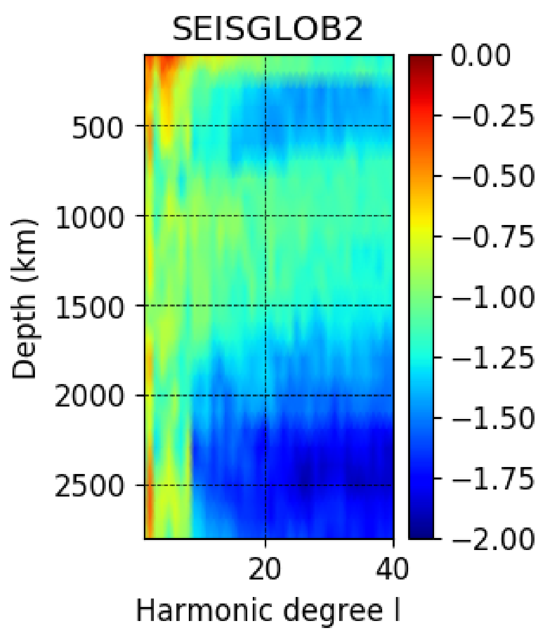
to

$ depths = np.arange(100,1100,100)

for instance.

```
# Setting parameters
# Model to be used for computing the spectrum
model = "SEISGLOB2"
# Depth range at which the spectrum is computed
depths = np.arange(100,2900,100)
```

```
# Parameter used for computing the spectrum
para = "VS"
# Maximum spherical harmonic degree up to which the spectrum is computed
NSmax = 40

deg = np.arange(1,41,1)
sp1 = np.zeros([len(deg),len(depths)])

# Running spectrum routinely at various depths
for i in range(len(depths)):
    sp11 = SeisTomoPy.spectrum(model,para,depths[i],NSmax)
    for k in range(len(deg)):
sp1[k,i]=sp11[k,1]

# Plotting the result
X, Y = np.meshgrid(deg, depths)
plt.subplots_adjust(bottom=0.2, right=0.5, left=0.2, top=0.9)
plt.pcolormesh(X, Y, (np.log10(np.transpose(sp1))), ...
   shading="gouraud", cmap="jet", vmin=-2, vmax=0)
plt.gca().invert_yaxis()
plt.ylabel("Depth (km)")
plt.xlabel("Harmonic degree l")
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.title("SEISGLOB2")
plt.colorbar()
plt.show()
```




## 5.4 Correlations

The user can compute the correlation between two tomographic models, (1) and (2), using `SeisTomoPy.correlation`, `SeisTomoPy.correlation_fromfile` and

SeisTomoPy.correlation_fromfile2. The first one computes the correlation between two models chosen from the included ones by default in SeisTomoPy, while the second and third ones enable the user to compute the correlation between either any model with one of the default ones or between any other models that the user wishes to use. In order to use the two last functions the user thus must provide the required input file. Please refer to section 11 for details about these files.

The correlation can be carried out for any parameter $X_1$ of model (1) and $X_2$ of model (2) and for the same depth between the two models ($z_1 = z_2$) or it could be for different depths ($z_1 \neq z_2$). Correlations are computed at a given spherical harmonic degree, $l$, following:

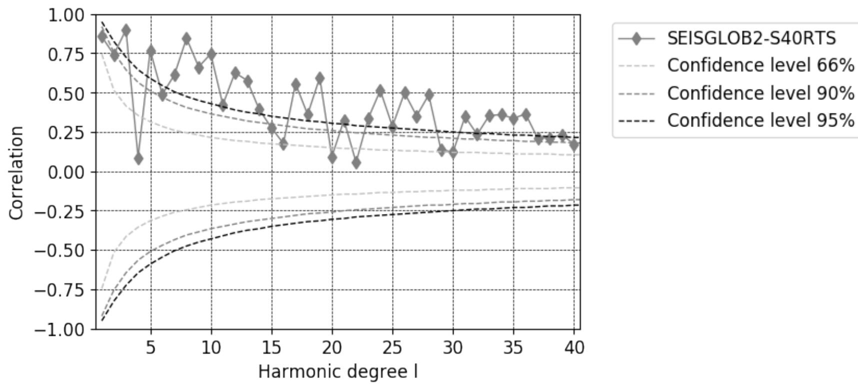
$$C(X_1, z_1, X_2, z_2, l) = \frac{\sum_{m=-l}^{l} \left(\frac{\delta X_1}{X_1}(z_1)\right)_{lm} \left(\frac{\delta X_2}{X_2}(z_2)\right)_{lm}^{\star}}{\sqrt{\sum_{m=-l}^{l} \left(\frac{\delta X_1}{X_1}(z_1)\right)_{lm} \left(\frac{\delta X_1}{X_1}(z_1)\right)_{lm}^{\star} \sum_{m=-l}^{l} \left(\frac{\delta X_2}{X_2}(z_2)\right)_{lm} \left(\frac{\delta X_2}{X_2}(z_2)\right)_{lm}^{\star}}}. \tag{2}$$

Below is an example of the correlation computed between SEISGLOB2 and S40RTS at 520 km depth, for parameter $V_s$ and up to spherical harmonic degree 40. We then show how to find the correlation for any model not included in SeisTomoPy.

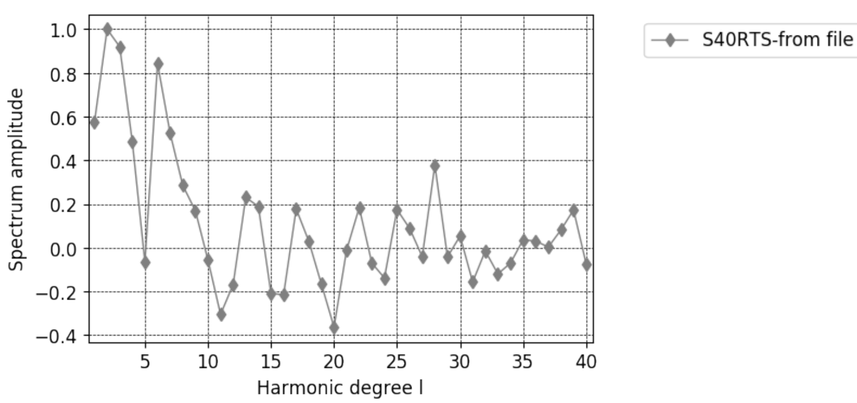
```
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print("Example of correlation computed betwenn the default tomographic models")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

# Setting parameters
# Models to be used for computing the correlation
model1 = "SEISGLOB2"
model2 = "S40RTS"
# Depth at which the correlation is computed
depth = 520
# Parameters used for computing the correlation
para1 = "VS"
para2 = "VS"
# Maximum spherical harmonic degree up to which the correlation is computed
NSmax = 40

# Running correlation
corr = SeisTomoPy.correlation(model1,model2,depth,depth,para1,para2,NSmax)

plt.plot(corr[:,0],corr[:,1],linewidth=1.0,color="grey",marker="d", ...
        label=model1 + "-" + model2)
conf66 = np.loadtxt("conf66.dat")
conf90 = np.loadtxt("conf90.dat")
conf95 = np.loadtxt("conf95.dat")
plt.plot(conf66[:,0], conf66[:,1], linewidth=1.0,color="silver", ls="--", ...
        label="Confidence level 66%")
plt.plot(conf90[:,0], conf90[:,1], linewidth=1.0,color="gray", ls="--", ...
label="Confidence level 90%")
plt.plot(conf95[:,0], conf95[:,1], linewidth=1.0,color="black", ls="--", ...
        label="Confidence level 95%")
```

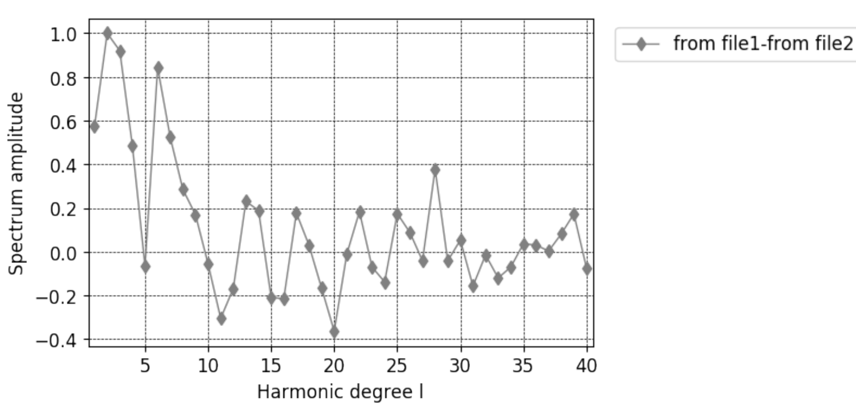
```python
plt.plot(conf66[:,0], -conf66[:,1], linewidth=1.0,color="silver", ls="--")
plt.plot(conf90[:,0], -conf90[:,1], linewidth=1.0,color="gray", ls="--")
plt.plot(conf95[:,0], -conf95[:,1], linewidth=1.0,color="black", ls="--")
plt.xlabel("Harmonic degree l")
plt.ylabel("Correlation")
plt.xlim([0.5, NSmax+0.5])
plt.ylim([-1, 1])
plt.legend(bbox_to_anchor=(1.6, 1))
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.show()

print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print("   Example of correlation using other models   ")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
# Setting new parameters
# Models to be used for computing the correlation
model1 = "S40RTS"
# Depth at which the correlation is computed
depth1 = 2800
# Parameters used for computing the correlation
para1 = "VS"
# Maximum spherical harmonic degree up to which the correlation is computed
NSmax = 40

# Running correlation_fromfile
filename = "SeisTomoPy_notebook/files/input_file_spectrum.xyz"
corr_fromfile = SeisTomoPy.correlation_fromfile(model1,depth1,para1,filename,NSmax)

# Plotting the results
plt.plot(corr_fromfile[:,0], corr_fromfile[:,1]/np.amax(corr_fromfile[:,1]), ...
        linewidth=1.0,color="grey",marker="d",label=model1 + "-from file")
plt.legend(bbox_to_anchor=(1.55, 1))
plt.xlabel("Harmonic degree l")
plt.ylabel("Spectrum amplitude")
plt.xlim([0.5, NSmax+0.5])
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.show()

# Running correlation_fromfile2
filename1 = "SeisTomoPy_notebook/files/input_file_spectrum.xyz"
filename2 = "SeisTomoPy_notebook/files/input_file_spectrum2.xyz"
corr_fromfile2 = SeisTomoPy.correlation_fromfile2(filename1,filename2,NSmax)

# Plotting the results
plt.plot(corr_fromfile2[:,0], corr_fromfile2[:,1]/np.amax(corr_fromfile2[:,1]), ...
        linewidth=1.0,color="grey",marker="d",label="from file1-from file2")
plt.legend(bbox_to_anchor=(1.55, 1))
plt.xlabel("Harmonic degree l")
plt.ylabel("Spectrum amplitude")
plt.xlim([0.5, NSmax+0.5])
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
```

```
plt.show()

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Example of correlation computed between the default tomographic models
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```




```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   Example of correlation using other models
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

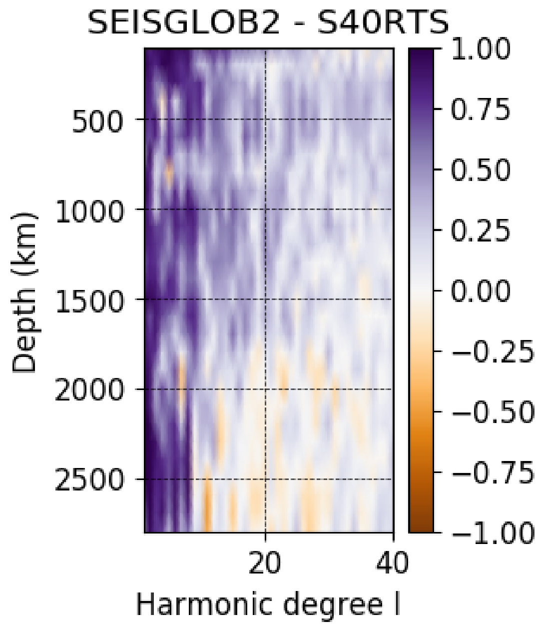



It is possible to run the correlation calculations routinely at various depths. The example below can take some time to run. One can reduce this time changing

$ depths = np.arange(100,2900,100)
to
$ depths = np.arange(100,1100,100).

```
# Setting parameters
# Models to be used for computing the correlation
model1 = "SEISGLOB2"
model2 = "S40RTS"
# Depth range at which the correlation is computed
depths = np.arange(100,2900,100)
# Parameters used for computing the correlation
para1 = "VS"
para2 = "VS"
# Maximum spherical harmonic degree up to which the correlation is computed
NSmax = 40

deg = np.arange(1,41,1)
corr1 = np.zeros([len(deg),len(depths)])

# Running correlation routinely at various depths
for i in range(len(depths)):
    corr11 = SeisTomoPy.correlation(model1,model2,depths[i],depths[i],para1,para2,NSmax)
    corr1[:,i] = corr11[:,1]

# Plotting the results
X, Y = np.meshgrid(deg, depths)
plt.subplots_adjust(bottom=0.2, right=0.5, left=0.2, top=0.9)
plt.pcolormesh(X, Y, ((np.transpose(corr1))), ...
   shading="gouraud", cmap="PuOr", vmin=-1, vmax=1)
plt.gca().invert_yaxis()
plt.ylabel("Depth (km)")
plt.xlabel("Harmonic degree l")
plt.title("SEISGLOB2 - S40RTS")
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.colorbar()
plt.show()
```




### 5.5 Paths

The user may want to check which seismic structures of the mantle are sampled by seismic waves. To do so the user can use `SeisTomoPy.path_plot` and `SeisTomoPy.path_plot_fromfile` to display seismic wave paths on top of cross sections made in the desired tomographic model coming from either one of the default ones included in SeisTomoPy or that the user provides. The user must also choose in which 1-D model the paths are calculated. It can be one of the default ones or the user can upload its one model. Please refer to section 11 for details about these models and files. Finally, the user must give files with the location of the stations and earthquakes. These files have two columns, angle in degrees from the start of the cross-section and depth in km (see section 6.5).

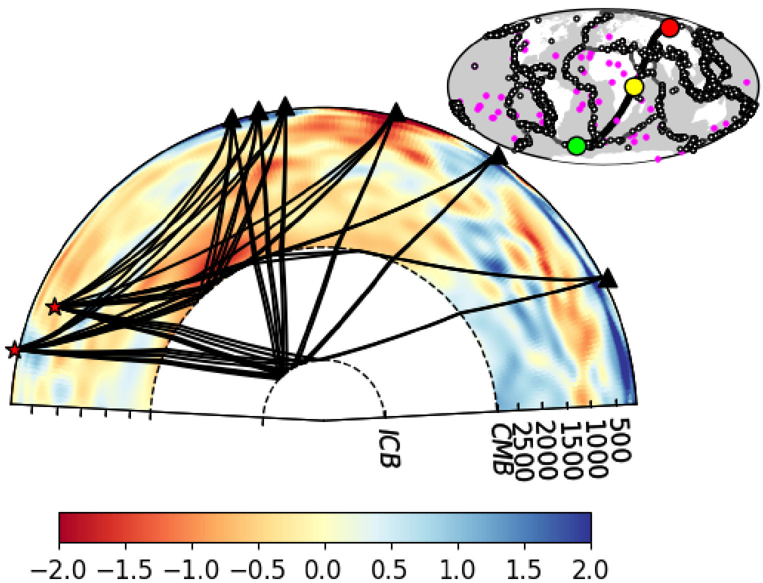
Below is an example with tomographic model SEISGLOB2, parameter $V_s$ and seismic phases S, ScS, PKP, PKiKP and Sdiff. If it happens that the seismic phase does not exist for the distance range between the earthquake and the station, then it will simply be ignored.

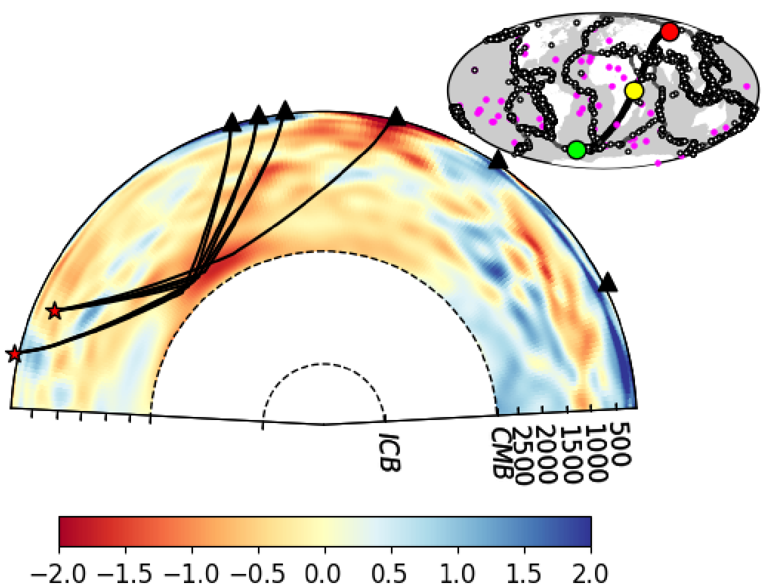
```
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print("Two examples of path plots using the default tomographic models")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
# Setting parameters
# Model to be plotted
model = "SEISGLOB2"
# Parameter to be plotted
para = "VS"
# Latitude of the starting point of the cross section
elat = -60
# Longitude of the starting point of the cross section
elon = -49
# Latitude of the ending point of the cross section
slat = 60
# Longitude of the ending point of the cross section
```

```python
slon = 119
# Maximal velocity perturbations for the colorbar
Vmax = 2
# List of seismic phases
phlist = "S ScS PKP PKiKP Sdiff"
#  1-D model to compute the paths
#  Can be one of the default models
model1d = "prem"
# Position of stations and event
EVT = np.loadtxt("SeisTomoPy_notebook/files/event.xy")
STA = np.loadtxt("SeisTomoPy_notebook/files/station.xy")

# Running path_plot
SeisTomoPy.path_plot(model,para,Vmax,elat,elon,slat,slon,EVT,STA,phlist,model1d)

# 1-D model to compute the paths
# The user can give the path to a model file
model1d = "SeisTomoPy_notebook/files/premDs.nd"
# List of seismic phases
phlist = "Sv2491S"

# Running path_plot
SeisTomoPy.path_plot(model,para,Vmax,elat,elon,slat,slon,EVT,STA,phlist,model1d)

print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")
print("  An example of path plot using a model file ")
print("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

# Setting new parameters
# Latitude of the starting point of the cross section
elat = -60
# Longitude of the starting point of the cross section
elon = -49
# Latitude of the ending point of the cross section
slat = 60
# Longitude of the ending point of the cross section
slon = 130
# Parameter to be plotted
para = "VS"
# Angular size of the cross section
width = 180
# Sampling rate in theta direction
width_step = 1
# Theta vector
th = np.arange(90-(width/2), 90+(width/2)+width_step, width_step)
# Starting and ending radius of the cross section
r_min = 3481
r_max = 6281
# Sampling rate in radius direction
r_step = 50
# Radius vector
r = np.arange(r_min, r_max+r_step, r_step)
# List of seismic phases
phlist = "Pdiff"
```

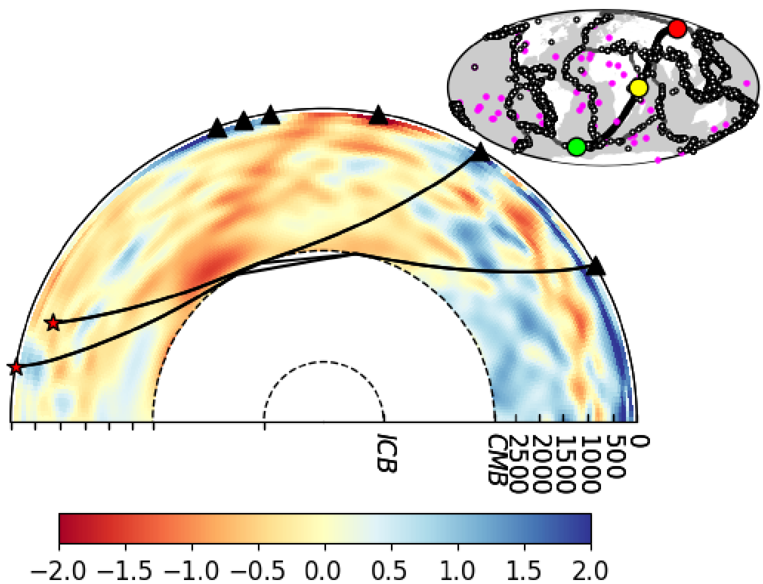
```
#  1-D model to compute the paths
model1d = "prem"

# Running path_plot_from_file
filename = "SeisTomoPy_notebook/files/input_file_path.xyz"
SeisTomoPy.path_plot_fromfile(filename,th,r,para,elat,elon,slat,slon,Vmax,EVT,STA,phlist,model1d)
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Two examples of path plots using the default tomographic models
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```




```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  An example of path plot using a model file
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```



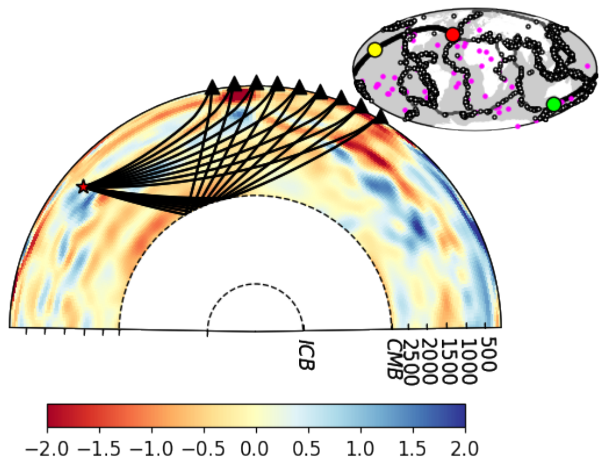

## 5.6 Travel times

The user can compute the travel time delays with respect to a reference model through a given tomographic model, $\delta t^{3D}_{TomoPy}$, for any given seismic phases and for any combinations of source and receiver provided by the user using `SeisTomoPy.get_travel_time`.

In the example below, we first plot the paths to see which part of the mantle the seismic waves are sampling. Then, we provide an example of delays computed in model SEISGLOB2 for S and ScS seismic phases in the same region.

```
# Setting parameters
# Model to be plotted
model = "SEISGLOB2"
# Parameter to be plotted
para = "VS"
# Latitude of the starting point of the cross section
elat =   -43
# Longitude of the starting point of the cross section
elon = 140
# Latitude of the ending point of the cross section
slat = 44
# Longitude of the ending point of the cross section
slon =  -42
# Maximal velocity perturbations for the color bar
Vmax = 2
# Depth of the cross section
depth = 2890
# List of seismic phases
list = "S ScS"
#  1-D model to compute the paths
model1d = "prem"
# Position of stations and event
EVT = np.loadtxt("SeisTomoPy_notebook/files/event_time.xy")
```

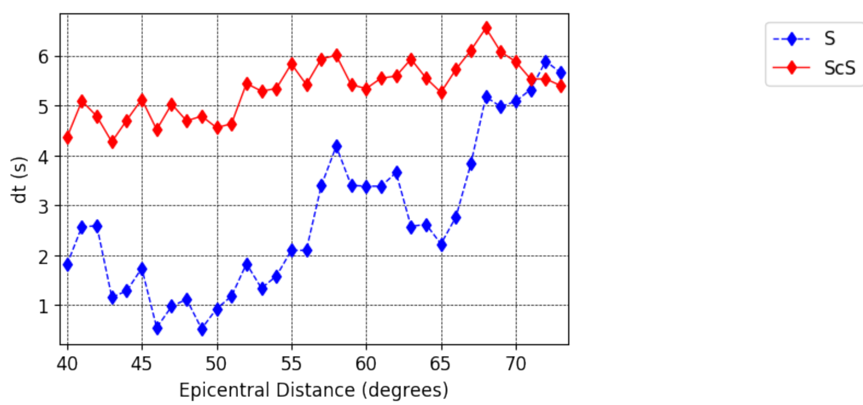
```
STA = np.loadtxt("SeisTomoPy_notebook/files/station_time.xy")


# Running path_plot
SeisTomoPy.path_plot(model,para,Vmax,elat,elon,slat,slon,EVT,STA,list,model1d)
```




```
# Setting parameters
# Model to be plotted
model = "SEISGLOB2"
# Latitude of the event
elat = -21
# Longitude of the event
elon = -179
# Depth of the event
edepth = 610
# Position of stations
STA = np.loadtxt("SeisTomoPy_notebook/files/lat_lon_ttstation.txt")

tt2D = np.zeros(len(STA))
dtt2D = np.zeros(len(STA))
ttREF = np.zeros(len(STA))
ttTHEOR = np.zeros(len(STA))
dist = np.arange(40,74,1)
degmin = 40
degmax = 73

# Running TimePy routinely
for k in range(len(STA)):
    # List of seismic phases
    List = ["S"]
    tt2D[k],  dtt2D[k], ttREF[k], ttTHEOR[k], phase_name = ...
  SeisTomoPy.get_travel_time(model,elat,elon,edepth,STA[k,0],STA[k,1],List)
    file_str = str(k) + " " +  "S " + str(tt2D[k]) + " " + str(tt2D[k]+ttREF ...
  [k]) + " " + str(dtt2D[k]) + "\n"
    print(file_str)
plt.plot(dist,tt2D,marker="d",linewidth=1.0,color="blue", ls="--",label = "S")

for k in range(len(STA)):
```

```
    # List of seismic phases
    List = ["ScS"]
    tt2D[k], dtt2D[k], ttREF[k], ttTHEOR[k], phase_name = ...
  SeisTomoPy.get_travel_time(model,elat,elon,edepth,STA[k,0],STA[k,1],List)
    file_str = str(k) + " " + "ScS " + str(tt2D[k]) + " " + str(tt2D[k]+ttREF ...
  [k]) + " " + str(dtt2D[k]) + "\n"
    print(file_str)

# Plotting the results
plt.plot(dist,tt2D,marker="d",linewidth=1.0,color="red", ls="-",label = "ScS")
plt.xlabel("Epicentral Distance (degrees)")
plt.ylabel("dt (s)")
plt.xlim([degmin-0.5, degmax+0.5])
plt.grid(color="k", linestyle="--", linewidth=0.5)
plt.rcParams.update({"font.size": 12})
plt.legend(bbox_to_anchor=(1.6, 1))
plt.show()
```

```
0 S 1.71 724.6 0.16

1 S 2.46 738.1 0.16

2 S 2.49 763.9 0.16

3 S 1.11 775.4 0.16

4 S 1.23 788.1 0.16

5 S 1.7 801.2 0.18

...
```




## 6 How to use SeisTomoPy GUI

It is possible to use SeisTomoPy tools via the GUI graphical interface. To do so the user has to launch the GUI interface (see Figure 1). With the top bar it is possible to navigate through the different tools. Each of them offer the possibility to save the plots as well as some output files.

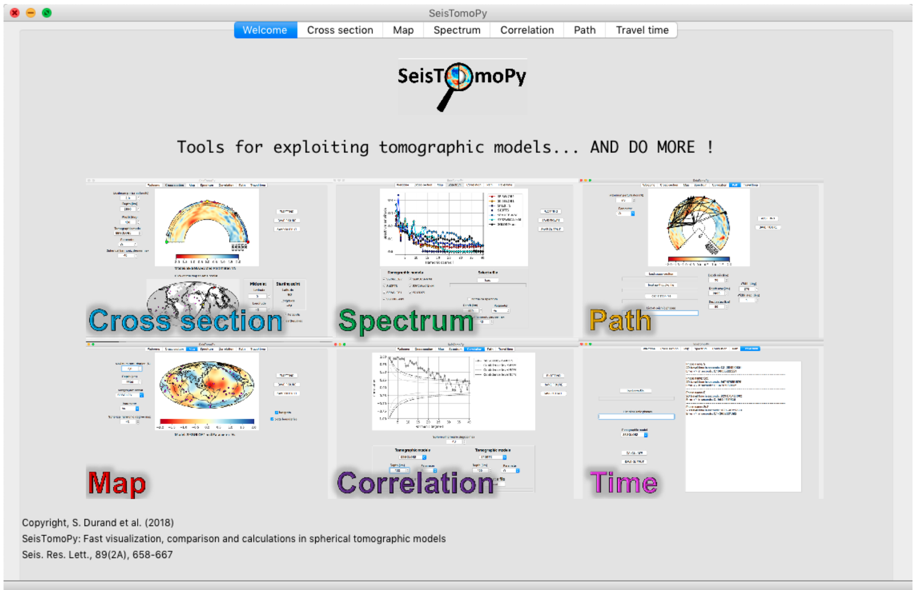



Figure 1: Welcoming screen of SeisTomoPy GUI interface.

The user must run:

```
ipython -m SeisTomPy.gui
```

## 6.1 Cross sections

Go on the tab Cross section. A screen with a map at the bottom and an empty space at the top appears. The parameters are already set to default values so that clicking on the button PLOTTING (top right button) computes a cross section.

Various options are available. First, it is possible to choose the location of the profile. To do so:

- 1) Click on the map at the bottom which will move the profile. This changes the mid-point of the profile (yellow dot). One can also manually enter the values of the latitude and longitude of the mid-point.
- 2) Change the azimuth of the profile.

Other parameters that can be changed include:

- 3) The width of the profile.
- 4) The maximum depth down to which the cross section will be plotted.
- 5) The tomographic model and parameter to be plotted.
- 6) The spherical harmonic degree up to which the model will be plotted. It can be at most 60.

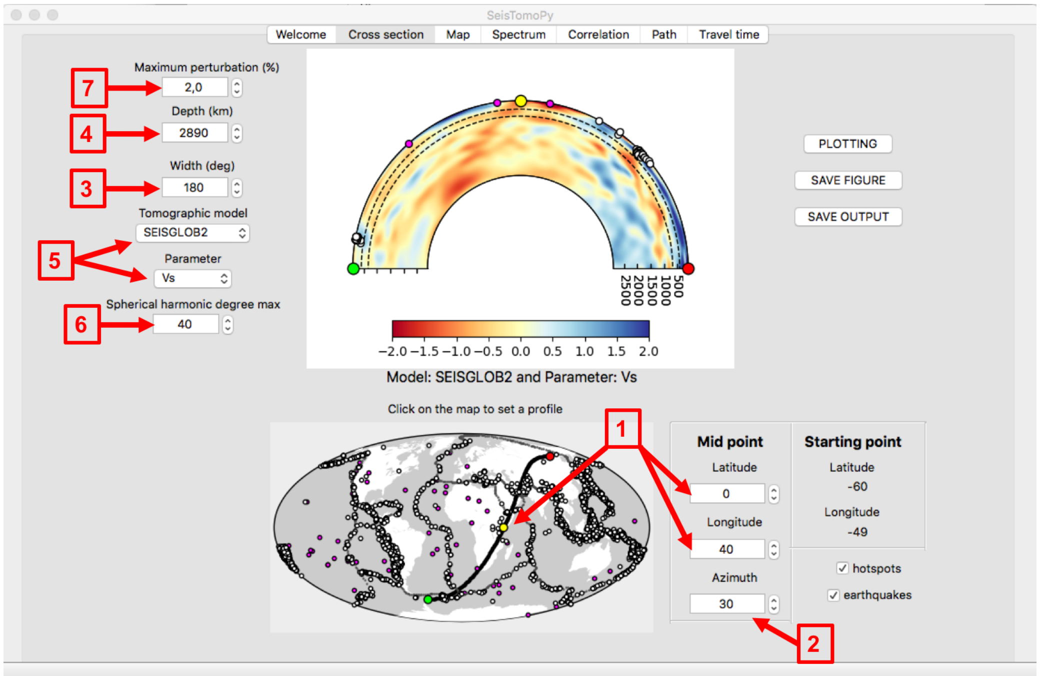



Figure 2: Summary of the different options in Cross section.

- 7) The maximum values of the velocity perturbations used for the color bar.

All these actions are summarized on Figure 2.

One can save the plots by clicking on SAVE FIGURE (top right button). This will produce two pdf files:

- `map_MODEL_PARA_LAT_LON_AZI.pdf`
- `crossec_MODEL_PARA_LAT_LON_AZI.pdf`

MODEL, PARA, LAT, LON and AZI, respectively, correspond to the tomographic model, parameter, latitude and longitude of the mid-point and azimuth used for generating the profile.

Some useful output files can also be saved by clicking on SAVE OUTPUT. A folder entitled `output_crossection_LAT_LON_AZI` will appear at the desired location. It contains 3 files:

- `output_cross_MODEL.out`: file containing all the values to reproduce the cross section. There are 6 columns: latitude, longitude, radius (km), $d\ln(V_p)$ (%), $d\ln(V_s)$ (%), $d\ln(\rho)$ (%).
- `MODEL_input_AxiSEM.sph`: input file for running synthetic seismograms with AxiSEM. **It is important to note that these files give the velocity perturbations with respect to the reference model used in every tomographic inversion so that if the user provides this file to AxiSEM, the good reference model file should also be provided. They are provided in** `Taup_models/AXISEM_REF`.

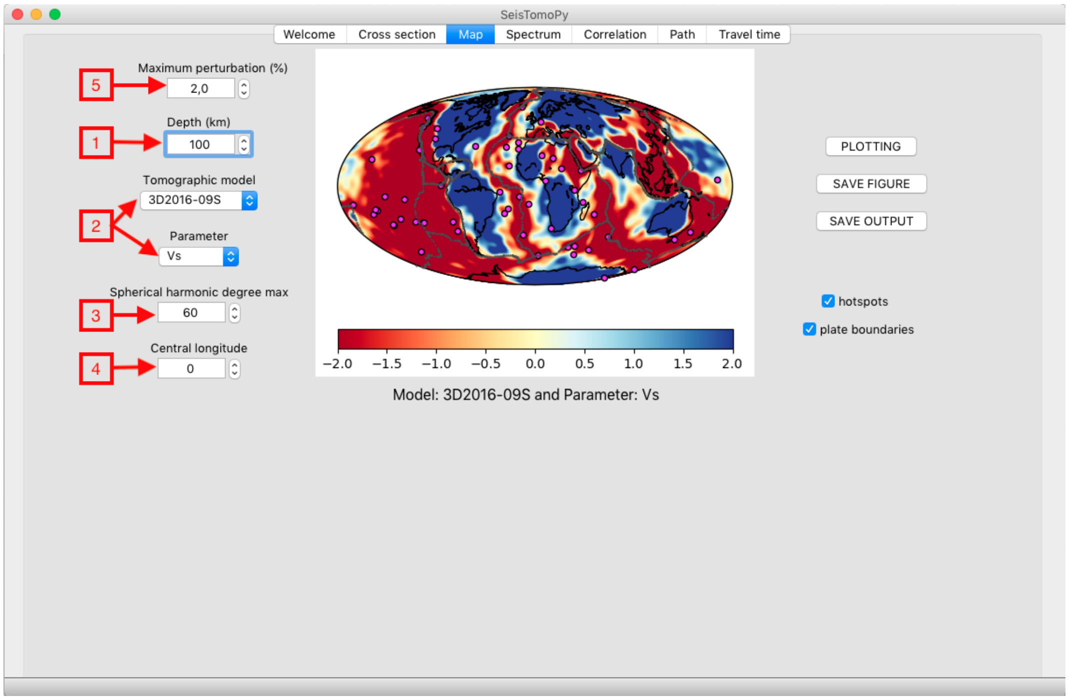



Figure 3: Summary of the different options in Map.

- `MODEL_PathPy.sph`: file that can be used in PathPy (see later).

## 6.2 Maps

On the tab Maps, one finds a screen with an empty space where the map will be plotted. The parameters are already set to default values so that one can already click on the button PLOTTING (top right button). After some seconds, the map will appear.

Parameters that can be changed include:

- 1) The depth of the map.
- 2) The tomographic model and parameter to be plotted.
- 3) The spherical harmonic degree up to which the model will be plotted. It can be at most 60.
- 4) The central longitude.
- 5) The maximum values of the velocity perturbations used for the color bar.

All these actions are summarized on Figure 3.

The plot can be saved by clicking on SAVE FIGURE (top right button). This will produce one pdf file:

- map_MODEL_PARA_DEPTH.pdf

MODEL, PARA and DEPTH, respectively, correspond to the tomographic model, parameter and depth used for generating the map.

Some useful output files can also be saved by clicking on SAVE OUTPUT. A folder entitled `output_map_DEPTH` will appear at the desired location. It contains 4 files:

- 3 files `map_NEW_MODEL_PARA.out`: files containing all the values to reproduce the map. There are 3 columns: latitude, longitude, $dln(PARA)$ (%).
- `output_map_MODEL.out`: summary file with 5 columns: latitude, longitude, $dln(V_p)$ (%), $dln(V_s)$ (%), $dln(\rho)$ (%).

## 6.3 Spectrum

On the tab Spectrum, the spectrum will be calculated. The parameters are already set to default values so that clicking on the button PLOTTING (top right button) produces a spectrum.

Parameters that can be changed include:

- 1) The tomographic models to be used.
- 2) The depth at which the spectrum is being computed.
- 3) The parameter.
- 4) The spherical harmonic degree up to which the spectrum will be computed. It can be at most 60.
- 5) It is also possible with this tool to compute the spectrum of any other model that the user wishes to use. Then the user must upload the file with the model at a given depth for a given parameter using the space entitle “Select a file” by clicking on the button “load”. The file must have a specific format. Please refer to section 11 for details about these files.

All these actions are summarized on Figure 4.

One can save the plot by clicking on SAVE FIGURE (top right button). This will produce one pdf file:

- `spectre_DEPTH.pdf`

DEPTH corresponds to the depth at which the spectrum has been computed.

Some useful output files can be saved by clicking on SAVE OUTPUT. A folder entitled `output_spectre_DEPTH` will appear at the desired location. It contains as many files as the number of tomographic models used:

- `spectre_MODEL_PARA_DEPTH.out`: file containing all the values of the spectrum. There are 2 columns: harmonic degree, spectrum.

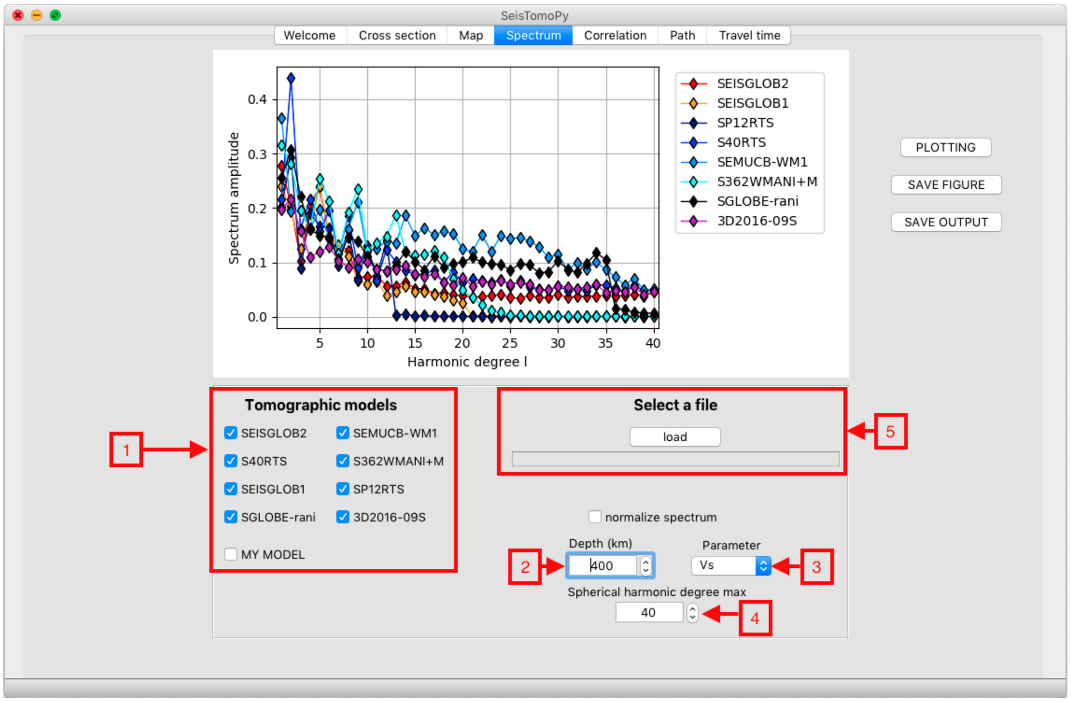



Figure 4: Summary of the different options in Spectrum.

## 6.4 Correlations

Tab Correlation calculates correlations between models. The parameters are already set to default values so that clicking on the button PLOTTING (top right button) produces a correlation.

Parameters that can be changed include:

- 1) The two tomographic models.
- 2) The two depths.
- 3) The two parameters.
- 4) The spherical harmonic degree up to which the spectrum will be computed. It can be at most 60.
- 5) It is also possible with this tool to compute the correlation with any other model that the user wishes to use. The user must upload the file with the model at a given depth for a given parameter using the space entitled "Select a file" by clicking on the button "load". The file must have a specific format. Please refer to section 11 for details about these files. If this option is to be used, then the tomographic model should be set to "None".

All these actions are summarized on Figure 5.

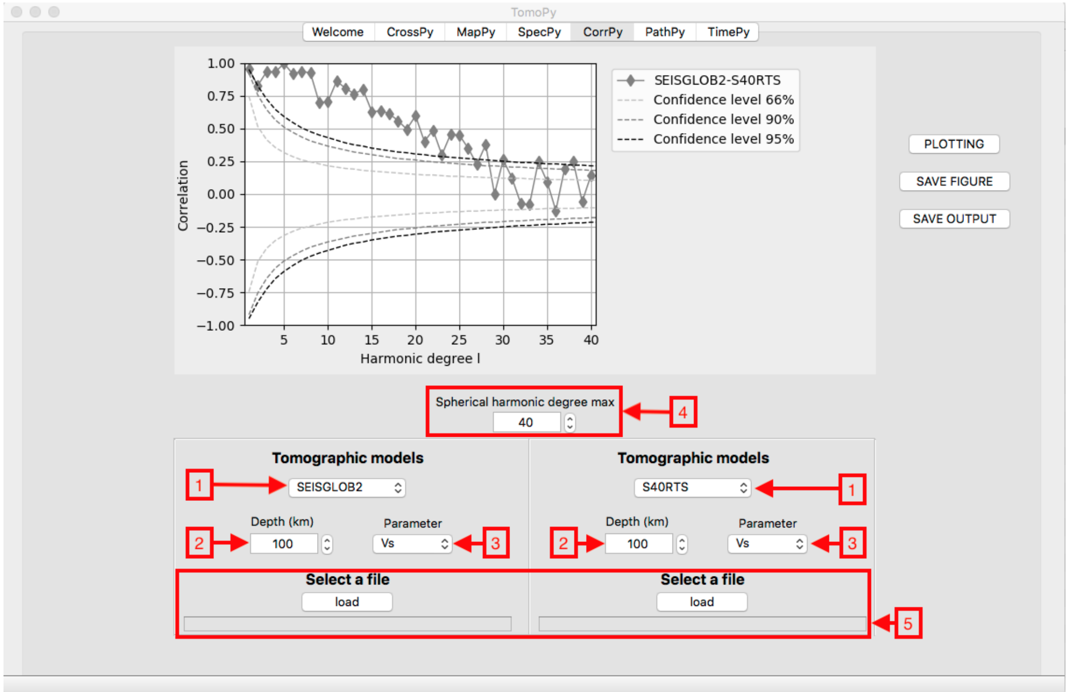



Figure 5: Summary of the different options in Correlation.

The plot can be saved by clicking on SAVE FIGURE (top right button). This will produce one pdf file:

- `correlation.pdf`

Some useful output files can also be saved by clicking on SAVE OUTPUT. A folder entitled `output\_corr` will appear at the desired location. It contains 1 file:

- `corr_MODEL1_MODEL2.out`: file containing all the values of the correlation. There are 2 columns: harmonic degree, correlation.

## 6.5 Paths

Tab Path allows to calculate paths and plot them on top of tomographic models.

The first thing to do with Path is

- 1) to upload a cross section. This can be achieved by clicking on "load cross-section" and first using a file generated by the Cross section tool (`MODEL_PathPy.sph`).
- 2) Then in order to properly read this file the user must give the starting and ending depths in the file, as well as the depth step. The same for the width of the cross section. It is then already possible to test wether the file is properly read by clicking on PLOTTING. If yes, then a cross section is plotted (see Figure 6).

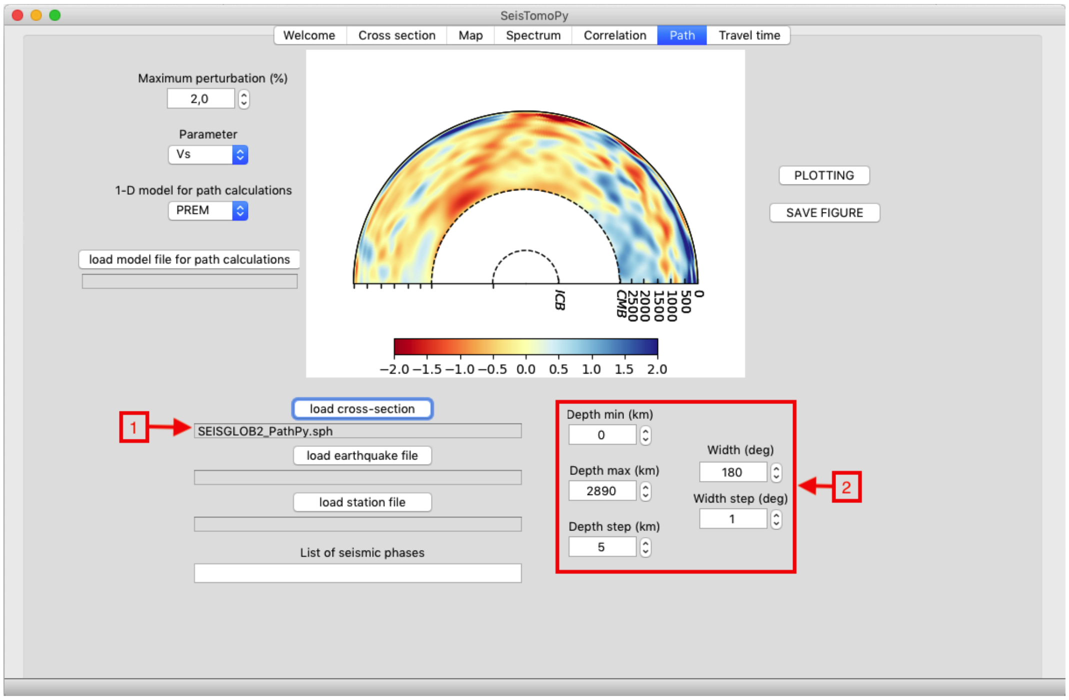



Figure 6: Summary of the different options in Path.

- 3) Then one may want to plot some seismic wave paths on top of the cross section. To do so the user must choose a 1-D model used for the paths calculations. It can be chosen within the default ones or the user can also upload a 1-D model. Please refer to section 11 for details about the model file.

- 4) The user must also upload earthquake and station files. For the earthquake and station files, they must contain two columns with the angle (in degrees) from the starting point of the profile and the depth (in km) of the earthquakes and stations. For the example Figure 7 the following files `event.xy`:

```
10 0
20 500
```

  and `station.xy`:

```
70 0
75 0
80 0
100 0
120 0
150 0
```

  have been used.

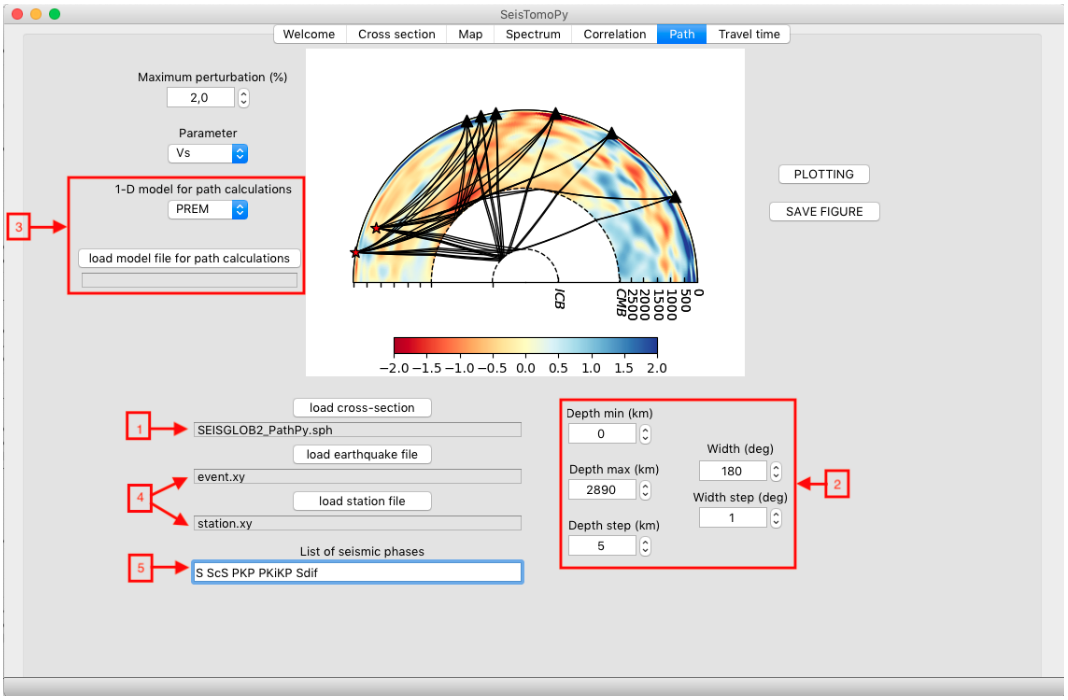



Figure 7: Summary of the different options in Path.

- 5) Finally, a list of seismic phases should be provided. The list must contain the name of the seismic phases. If there is more than one they must be separated by a blank (for instance S ScS Sdiff ...). It is possible to run the calculations by clicking on PLOTTING, and the cross section with the wave paths appears (see Figure 7).

The plot can be saved by clicking on SAVE FIGURE and a `path.pdf` file will appear at the chosen location.

## 6.6 Travel times

The last tool is Travel time. For this the user needs to

- 1) upload a file with the earthquake and station locations. This file contains 5 columns with event latitude, event longitude, event depth, station latitude and station longitude. For the example Figure 8 we used the file `test.xy`:

```
0 0 200 0 30
0 0 200 0 50
```

- 2) Then the user must specify the seismic waves to be computed. If more than one is given the different phases must be separated by blanks (for instance S ScS Sdiff).

- 3) Finally a tomographic model must be chosen. The travel time will be computed by clicking on CALCULATE. The summary of the calculations will appear in the empty space on the right (see Figure 8).

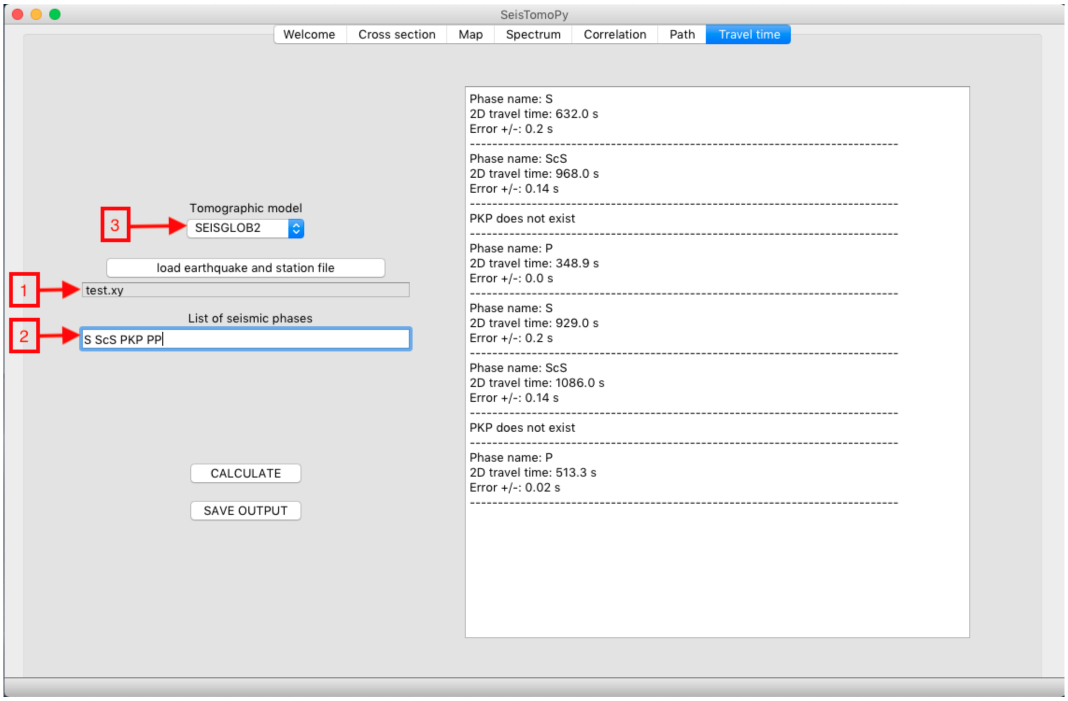



Figure 8: Summary of the different options in Travel time.

The travel times can be saved by clicking on SAVE OUTPUT which will create a folder travel_times_output.

# 7 Jupyter notebooks

Jupyter notebooks have been developped as tutorial to facilitate the use of SeisTomoPy. You can access them by going to the folder

```
$ cd SeisTomoPy_V3/Documentation/SeisTomoPy_notebook
```

Here you launch jupyter

```
$ jupyter notebook
```

A web window should open and you will see two notebooks, one for learning how to use the SeisTomoPy classes and one for learning how to use the GUI interface.

# 8 Model and parameter options

For using the tools, the tomographic models, parameters and 1-D models must be chosen. Here is the list of the acronyms to be used for the tomographic models:

```
"SEISGLOB2"
"S40RTS"
"SEMUCBWM1"
"S362WMANIM"
```

```
"SEISGLOB1"
"SP12RTS"
"SGLOBE"
"3D2016"
"MYMODEL"
```

for the parameters

```
"VS"
"VP"
"RHO"
```

and for the 1-D models

```
"prem"
"iasp91"
"ak135"
"pwdk"
```

# 9 Creating a new tomographic model file

It is possible to upload a new model in SeisTomoPy. For this the user has to generate the model file. To do so we provide a package. The user will find a directory `Create_model_file` with a README file explaining the procedure.

First the user must generate `vs.DEPTH.xyz` files at every kilometer depth and store them in the directory `Create_model_file/model` and then go to `Create_model_file/src` and run

```
$ make clean
$ make all
$ ./main
```

This will generate two model files `YOURMODEL_5km.sph` and `YOURMODEL_1km.sph` that should be copied in `SeisTomoPy/fortran_files/models`. Then one can use all the tools of SeisTomoPy referring to the new model using

```
"MYMODEL"
```

# 10 Changing plate boundary model and hot spots and earthquake catalogs

The plate boundaries of model NUVEL-1A [*DeMets et al.*, 1990], the hot spot locations [*Müller et al.*, 1993] and a catalog of earthquakes can be added. The catalog of earthquakes is not exhaustive, it has been built with a minimum of earthquakes such that plate boundaries can be seen. We used events deeper than 100 km depth with magnitudes greater than 5.9 in the time range 1976-1998 and we added all events with magnitudes greater than 5.0 in the time range 2010-2011. Magnitudes were chosen to be larger than 5.0 to keep the dataset small.

However, the user may want to use another plate boundary model, or hot spot and earthquake catalogs. To that aim it is possible to provide new files with the same format as those already provided and with the same name.

## 10.1 Plate boundary model

The file must be named `plate_boundaries.xy` and contains in the first column the longitude and in the second column the latitude of the plate boundaries. Every segment of boundary should be separated by NaN values as illustrated below.

```
...
    10.30000   -53.2000
     11.5000   -53.0000
     13.0000   -52.7000
     14.2000   -52.4000
     15.0000   -52.2000
     15.4000   -51.9000
NaN NaN
     15.3000   -51.9000
     16.1000   -51.9000
     16.4000   -52.1000
     17.3000   -52.4000
...
```

## 10.2 Hot spot catalog

The file must be named `points_chauds.xy` and contains in the first column the longitude of the hot spots and in the second column the latitude of the hot spots (see below).

```
...
42.0000 12.0000
-25.0000 -17.0000
-14.3700 -7.9500
-139.9699 -29.3722
-28.0000 38.0000
-113.0000 27.0000
164.7017 -67.3982
...
```

## 10.3 Earthquake catalog

The file must be named `catalogue.xy` and contains in the first column the longitude of the event, in the second column the latitude of the event and in the third column the depth of the event (see below).

```
...
167.81 -15.97 174
120.07 -7.37 623
166.95 -14.91 103
129.74 -6.90 186
-177.61 -19.46 453
70.64 36.33 196
-68.51 -20.54 134
...
```

# 11 Input file formats for Spectrum, Correlations and Paths

It is possible to upload model files to be used in Spectrum, Correlation and Path.

For Spectrum and Correlation the files have the same format and it is the same format as the files `map_NEW_MODEL_PARA.out` obtained with Map tool, which have 3 columns: latitude (from -89 to 89 degrees, with a step of 1 degree), longitude (from 0 to 359 degrees, with a step of 1 degree) and parameter perturbations (%). It must thus contain in total 64440 lines. Below is an extract given as example.

```
         -89            0  0.73245523974336291
         -88            0  0.72104058555001316
         -87            0  0.71041631025040164
...
          87          359  0.49352141683917167
          88          359  0.51590110372203857
          89          359  0.54502724312040074
```

For Path, the file must contain 5 columns: the radius, the distance in degrees from the starting point, $d\ln(V_p)$, $d\ln(V_s)$ and $d\ln(\rho)$ in %. It is the same format as the files `MODEL_PathPy.sph` obtained with the Cross section tool. The user sets which parameter should be plotted in the input arguments of the function `SeisTomoPy.path_plot_fromfile` or with the parameters to set on the GUI. Below is an example.

```
        3481            0    2.7162811E-002   4.9385717E-002   9.8771474E-003
        3531            0    3.3263616E-002   6.0478839E-002   1.2099678E-002
        3581            0    4.0070403E-002   7.2856784E-002   1.4571265E-002
...
        6131          180    1.2205796        2.2192353        0.4438470
        6181          180    0.4952321        0.9005191        0.1801038
        6231          180    0.7806580        1.4193792        0.2833848
        6281          180    1.5027913        2.7322743        0.5464547
```

For path the user can also upload a 1-D model. This file must have the same format as `*.nd` files used in TauP tool kit. It contains 6 columns: the depth (km), $V_p$ (km s$^{-1}$), $V_s$

(km $s^{-1}$), $\rho$ (g $cm^{-3}$), $Q_\kappa$, $Q_\mu$. For major discontinuities (mantle, outer-core and inner-core) the discontinuity must be written. For any additional discontinuity tha user would like to add it should just be added where it is desired. Below is an example.

```
   0.00      5.80000   3.20000   2.60000    1456.0     600.0
  15.00      5.80000   3.20000   2.60000    1456.0     600.0
  15.00      6.80000   3.90000   2.90000    1350.0     600.0
  24.40      6.80000   3.90000   2.90000    1350.0     600.0
mantle
  24.40      8.11061   4.49094   3.38076    1446.0     600.0
  40.00      8.10119   4.48486   3.37906    1446.0     600.0
  60.00      8.08907   4.47715   3.37688    1447.0     600.0
...
 2271.00    13.13055   7.05525   5.25729     803.0     312.0
 2371.00    13.24532   7.09974   5.30724     807.0     312.0
 2471.00    13.36074   7.14423   5.35706     811.0     312.0
 2491.00    13.38410   7.15320   5.36700     812.0     312.0
 2491.00    13.91950   7.43930   5.58170     812.0     312.0
 2571.00    14.01650   7.47650   5.62310     815.0     312.0
 2671.00    14.13980   7.52340   5.67480     819.0     312.0
 2741.00    14.22760   7.55660   5.71110     822.0     312.0
 2771.00    14.23500   7.55640   5.72670     823.0     312.0
 2871.00    14.26020   7.55550   5.77870     826.0     312.0
 2891.00    14.26530   7.55520   5.78910     826.0     312.0
outer-core
 2891.00     8.06482   0.00000   9.90349   57822.0       0.0
 2971.00     8.19939   0.00000  10.02940   57822.0       0.0
 3071.00     8.36019   0.00000  10.18134   57822.0       0.0
 3171.00     8.51298   0.00000  10.32726   57822.0       0.0
...
 4971.00    10.24959   0.00000  12.06924   57822.0       0.0
 5071.00    10.30971   0.00000  12.12500   57822.0       0.0
 5149.50    10.35568   0.00000  12.16634   57822.0       0.0
inner-core
 5149.50    11.02827   3.50432  12.76360     445.0      85.0
 5171.00    11.03643   3.51002  12.77493     445.0      85.0
 5271.00    11.07249   3.53522  12.82501     443.0      85.0
...
 6271.00    11.26064   3.66670  13.08630     431.0      85.0
 6371.00    11.26220   3.66780  13.08848     431.0      85.0
```

# 12 Future developments

Further functionalities will be added in the future:

- Displaying fully anisotropic tomographic models.
- Displaying geodynamic models obtained from modeling codes such as Stag3D [*Tackley & Shunxing*, 2002], that can then be directly compared to tomographic models.

# 13 Reference

Please if you are using SeisTomoPy please refer to

S. Durand, R. Abreu, C. Thomas, 2018,
SeisTomoPy: Fast visualization, comparison and calculations in global tomographic models,
*Seis. Res. Lett.*, **89**(2A), 658-667

# References


Bassin, C., Laske, G., Masters, G., 2000, The current limits of resolution for surface wave tomography in North America, *EOS Trans AGU*, **81**, F897

Chang, S-J, Ferreira, A.M.G., Ritsema, J., van Heijst, H. J., Woodhouse, J.H., 2015, Joint inversion for global isotropic and radially anisotropic mantle structure including crustal thickness perturbations, *J. Geophys. Res.*, **120**, 4,278-4,300.

DeMets, C., Gordon, R., Argus, D., 1990, Current plate motions, *Geophys. J Int.*, **101**, 425-478.

Durand, S., Debayle, E., Ricard, Y., Lambotte, S., 2016, Seismic evidence for a change in the large scale tomographic pattern accross the D" layer, *Geophys. Res. Lett.*, **43**(15), 7,928-7,936.

Durand, S., Debayle, E., Ricard, Y., Zaroli, C., Lambotte, S., 2017a, Confirmation of a change in the global shear velocity pattern at around 1,000 km depth, *Geophys. J Int.*, *revision*

Durand, S., Abreu, R., Thomas, C., 2018, SeisTomoPy: Fast visualization, comparison and calculations in global tomographic models, *Seis. Res. Lett.*, **89**(2A), 658-667

French, S.W., Romanowicz, B., 2014, Whole-mantle radially anisotropic shear-velocity structure from spectral-element waveform tomography, *Geophys. J Int.*, **199**, 1,303-1,327.

Koelemeijer, P. , Ritsema,J., Deuss, A., Van Heijst, H-J., 2016), SP12RTS: a degree-12 model of shear- and compressional-wave velocity for Earth's mantle, *Geophys. J Int.*, **204**(2), 1,024-1,039.

Kustowski, B., Ekstrom, G., Dziewonski, A. M., 2008, Anisotropic shear-wave velocity structure of the Earth's mantle: A global model, *J. Geophys. Res.*, **113**B06306, doi:10.1029/2007JB005169.

Moulik, P., Ekström, G., 2014, An anisotropic shear velocity model of the Earth's mantle using normal modes, body waves, surface waves and long-period waveforms, *Geophys. J Int.*, **199**3, 1,713-1,738.

Müller, R.D., Royer, J-Y., Lawver, L.A., 1993, Revised plate motions relative to the hotspots from combined Atlantic and Indian ocean hotspot tracks, *Geology*, **21**, 275-278.

Nataf, H., Ricard, Y.,1993, 3SMAC: An a priori tomographic model of the upper mantle based on geophysical modeling, *Phys. Earth Planet. Inter.*, , **95**, 101–122.

Ritsema, J., Deuss, A., van Heijst, H.J., Woodhouse, J.H., 2011, S40RTS: a degree-40 shear-velocity model for the mantle from new Rayleigh wave dispersion, teleseismic traveltime and normal-mode splitting function measurements, *Geophys. J Int.*, **184**, 1,223-1,236.

Tackley, P., Shunxing, X., 2002, The thermochemical structure and evolution of Earth's mantle: constraints and numerical models, *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **360**, 1,800 ,2,593-2,609.