Full waveform inversion with ASKI using GEMINI-SPECFEM3D coupling

Wolfgang Friederich

May 3, 2024

Contents

Introduction

This document explains how to do a hybrid full waveform inversion of teleseismic body waves using a combination of the codes GEMINI, SPECFEM3D-Cartesian and ASKI. To implement the hybrid approach, we define a regional computational box within which we perform SPECFEM3D-Cartesian calculations while the incident teleseismic wavefield is computed using GEMINI that is restricted to 1D earth models. In spite of the fact that we consider waves propagating in a spherical earth, we use the Cartesian version of SPECFEM3D. This is accomplished by applying a special mapping of the SPECFEM mesh to a spherical chunk. The GEMINI wavefields are injected at the boundaries of the SPECFEM box and continued using SPECFEM3D. For inversion using ASKI, we compute sensitivity kernels obtained from the displacement and strain fields available on the SPECFEM3D computational mesh. SPECFEM3D makes available these wavefields to ASKI which performs one iteration of the inversion. The new earth model after one iteration is fed back to SPECFEM3D for performing another forward calculation for the next iteration.

In the following, we describe how GEMINI, SPECFEM3D and ASKI are set up and used to realize this workflow. Section 1 explains how the full waveform inversion is organized. In section 2, we start with the preparation of SPECFEM3D computations up to generating the parallel databases and inplementing a 1D background model. Then, in section 3, we explain how to use GEMINI to compute injection wavefields at the boundaries of the SPECFEM box. In section 4, it is described how to process the raw seismogram data to finally transform them into complex Fourier amplitudes evaluated at predefined frequencies. This section also explains how moment rate functions are estimated from the observed seimograms. Section 5 describes the setup of ASKI and how it supports running SPECFEM to get the required displacement and stress fields. Section 6 brings everything together and describes the computation of sensitivity kernels from the wavefields computed by SPECFEM, the extraction and transformation of synthetic seismograms computed by SPECFEM and finally the setup of the equation system to be solved for obtaining an update of the earth model.

1 Organizing the the full waveform inversion task

1.1 Main parameter file and inversion folder structure

All input required and output created in ASKI full waveform inversion should be organised in a dedicated folder on a large disk with fast access to the parallel computing environment. We shall refer to this folder as the "main inversion folder" in the following. All non-absolute paths mentioned below are relative to this inversion folder. Of course, this folder will have subfolders to organize the puzzling amount of data involved. SPECFEM has its own subfolder and it is very useful also to create a subfolder for GEMINI in- and output. ASKI will also create a specially designed subfolder structure. Please use the inversion folder only for one specific inversion setup. If you make deep-reaching changes, simply create a new one.

The organisational aspects of full waveform inversion are controlled by a main parameter file with the following content:

ASKI related parameters	
Variable	Recomm. Value
MAIN_PATH_INVERSION	your inversion folder
PROPERTY_SET_NAME	isoVelocitySI
PROPERTY_CORRELATION_FILE	model_property_correlation
CURRENT_ITERATION_STEP	1
ITERATION_STEP_PATH	iteration_step_
PARFILE_ITERATION_STEP	iter_parfile
REARTH	6371000.0
INVGRID_CENTER_LAT	46.20
INVGRID_CENTER_LON	10.87
PATH_GEMINI	GEMINI/
PATH_MEASURED_DATA	ASKI_data
PATH_INJECTION_SEIS	injection_seismograms/
FILE_EVENT_LIST	ASKI_events
FILE_STATION_LIST	ASKI_stations
MEASURED_DATA_FREQUENCY_STEP	0.01
MEASURED_DATA_NUMBER_OF_FREQ	9
MEASURED_DATA_INDEX_OF_FREQ	1 2 3 4 5 6 7 8 9
UNIT_FACTOR_MEASURED_DATA	1
DEFAULT_VTK_FILE_FORMAT	ASCII
Paths relative to iteration step folder	
PATH_ASKI_MAIN_FILES	aski_main_files/
PATH_KERNEL_DISPLACEMENTS	kernel_displacements/
PATH_KERNEL_GREEN_TENSORS	kernel_green_tensors/
PATH_SENSITIVITY_KERNELS	sensitivity_kernels/
PATH_SYNTHETIC_DATA	synthetic_data/
PATH_OUTPUT_FILES	output_files/
PATH_VTK_FILES	vtkfiles/
SPECFEM specific inversion parameters	
PATH_SPECFEM_INPUT	DATA/
PATH_SPECFEM_DATABASES	DATABASES_MPI/
USE_ASKI_BACKGROUND_MODEL	.true.
FILE_ASKI_BACKGROUND_MODEL	ASKI_ak135_diehl
IMPOSE_ASKI_INVERTED_MODEL	.false.
FILE_ASKI_INVERTED_MODEL	xxx.kim
COMPUTE_ASKI_OUTPUT	.true.
ASKI_HDF_OUTPUT	.true.
ASKI_USES_GPU	.true.
ASKI_type_inversion_grid	4
ASKI_wx	1800000.
ASKI_wy	1350000.
ASKI_wz	600000.
ASKI_rot_X	0.0
ASKI_rot_Y	0.0
ASKI_rot_Z	0.0

ASKI_cx	0.0
ASKI_cy	0.0
ASKI_cz	-300000.0

Most relevant for organisational tasks are the path variables starting with the main inversion path. One of its important subfolders is the iteration step path whose name is appended by the current iteration step number to the form `iteration_step_001`. `PATH_GEMINI` points to a subfolder where all GEMINI-related material is placed. The injection seismograms produced by GEMINI go to a separate folder given by `PATH_INJECTION_SEIS`. All paths in the second block of the table are relative to the iteration step folder and contain material as described by the name of the parameter. `PATH_OUTPUT_FILES` is conceived for a variety of smaller files produced by the ASKI executables which do not go into the dedicated subfolders such as waveform kernels and displacement fields. `PATH_VTK_FILES` is reserved for VTK-files used for visualization purposes. Finally, the third block contains parameters relevant when performing forward computations using the SPECFEM code. It contains the paths to SPECFEM input data and to the SPECFEM parallel databases. The script `createIterationFolders.py` creates all the required folders in the inversion directory and also copies the template parameter files to their appropriate places. It takes the following command line arguments:

- the name of the main parfile,
- the name of the template directory in the source code,
- `--citer`: the index of the current iteration.

If the iteration index is greater than 1, the script only creates a new iteration directory and its subfolders.

Action 1.1: Main inversion folder and main parameter file

Create the inversion folder on a large disk attached to the parallel environment and also the GPUs. Copy the template of the main parameter file from the source code distribution's folder `templates`, to the main inversion folder. Then, fill the main parameter file with meaningful values, especially make decisions about all the path names. Once you feel save about these names, run the script `createIterationFolders.py`.

1.2 A 1D background earth model

SPECFEM forward computations and the calculation of injection seismograms using GEMINI should be done with an identical background model to avoid jumps at the boundaries of SPECFEM's computational box. SPECFEM and ASKI offer the opportunity to overlay this model later by a 3D tomographic model either before or during inversion. An ASKI background model is a table of values specifying depth, density, P- and S-velocity, shear Q and bulk modulus Q at nodes, all values given in SI units, for example:

```
PREM as ASKI 1D background model
0.0
4
```

```
2 2 3 3
0.000        2600.000    5800.000   3200.000   600.000   57823.000
15000.000    2600.000    5800.000   3200.000   600.000   57823.000
15000.000    2900.000    6800.000   3900.000   600.000   57823.000
25000.000    2900.000    6800.000   3900.000   600.000   57823.000
25000.000    3380.683    8110.246   4490.786   600.000   57823.000
52500.000    3377.694    8093.247   4480.651   600.000   57823.000
80000.000    3374.706    8076.248   4470.515   600.000   57823.000
80000.000    3374.706    8076.248   4470.515    80.000   57823.000
115000.000   3370.902    8054.613   4457.616    80.000   57823.000
150000.000   3367.098    8032.978   4444.716    80.000   57823.000
```

The first line is a free header line, the second line specifies a variable called `zmax` which is NOT used in ASKI's `model_aski`-code if mapping to a spherical chunk is done. The third line gives the number of layers with layer boundaries marked by double nodes, and the fourth line gives the number of nodes per layer. The ASKI external model extension to SPECFEM uses a layerwise spline interpolation to evaluate model values between the nodes. Put these model files into the DATA folder. There is a python script "convertPolypremToASKIBackground.py" in the geminiUnified-package which produces such a file from the PREM polynomial model.

Once the background model is set up, it is possible to create a Gemini node earth model using the script `convertASKIBackgroundToJson.py` in the `geminiUnified/python` source folder. Check this model carefully using `plotNodeEarthmodel` because Gemini will use splines for interpolation and unwanted oscillations may arise if the values vary too irregularly with depth. Put it into the Gemini subfolder.

Action 1.2: ASKI background model

Set up an ASKI background model, convert it using the script `convertASKIBackgroundToJson.py` to a JSON-file and plot it using `plotNodeEarthmodel`. Carefully check for unwanted oscillations caused by the spline interpolation. Eventually, insert further nodes or discontinuities to remove them. Put the ASKI background model into the `DATA` folder and the converted JSON file into the GEMINI folder.

2 Preparation of SPECFEM3D computations

2.1 Using SPECFEM3D on a GPU cluster

To configure SPECFEM3D to run on a GPU cluster, the optional argument `--with-cuda` has to be passed to the `configure` script. If the user wants to select a specific version, the argument can be adapted as follows: `--with-cuda=cuda<version number>`. An example of a configuration using `gfortran`, `gcc`, `mpif90` and CUDA version 10 looks like this:

```
./configure FC=gfortran cc=gcc MPIFC=mpif90 --with-cuda=cuda10.
```

Currently, the latest supported version of CUDA is version 11. Contrary to expectation, the version that has to be specified here is not the version of the installed CUDA toolkit but is related to the architecture of the user's GPU. For example, if you have a GPU with `Turing` architecture (version 10) and the latest CUDA toolkit installed (version 11.6 as of 07.02.2022), the version that has to be put here is version 10. For additional details please refer to the SPECFEM3D manual section 2.1.

If any changes to the Makefile have been made by the user, he/she is advised to create a copy of the modified Makefile as running `configure` creates a new Makefile from Makefile.in. If Makefile.in is not changed by the user, all changes in the Makefile are overwritten. It might be necessary to add the paths for HDFLIB and HDF_INCLUDES into the Makefile.

To run a simulation on a GPU, the variable `GPU_MODE` has to be set to `.true.` in the parameter file (see next section).

2.2 The SPECFEM main parameter file

Most of what SPECFEM3D does is controlled by its main parameter file `DATA/Par_file`. It is read by all SPECFEM executables but typically only a subset of the variables defined in the parameter file are really used. In order to know which variable should be set when running a certain executable, here is a table that specifies for each variable where it is used. The SPECFEM executables are abbreviated in the table as follows: D – `xdecompose_mesh`, G – `xgenerate_databases`, M – `xmeshfem3D`, S – `xspecfem3D`. We also recommend values for each parameter which are useful when doing coupling of SPECFEM and GEMINI or when using ASKI or both. In some cases, the recommended value is "auto" signifying that this parameter is set by the script `prepare_specfem_kernel.py` to avoid inconsistencies when using ASKI .

Variable	Used in				Recommended Value
SIMULATION_TYPE		G		S	1
NOISE_TOMOGRAPHY		G		S	0
SAVE_FORWARD				S	.false.
INVERSE_FWI_FULL_PROBLEM		G		S	.false.
UTM_PROJECTION_ZONE		G	M	S	11
SUPPRESS_UTM_PROJECTION		G	M	S	.true.
MAP_TO_SPHERICAL_CHUNK		G	M	S	.true.
NPROC		G		S	choose properly
NSTEP				S	auto
DT				S	auto
LTS_MODE	D			S	false
PARTITIONING_TYPE	D				1
USE_LDDRK				S	false
NGNOD	D	G	M	S	8
MODEL		G		S	external
APPROXIMATE_OCEAN_LOAD		G		S	false
TOPOGRAPHY		G	M		false
ATTENUATION	D	G		S	false
ANISOTROPY		G		S	false
GRAVITY				S	false

PML_CONDITIONS	D	G	M	S	false
PML_INSTEAD_OF_FREE_SURFACE	D	G	M	S	false
STACEY_ABSORBING_CONDITIONS	D	G		S	true
STACEY_INSTEAD_OF_FREE_SURFACE		G		S	false
BOTTOM_FREE_SURFACE		G		S	false
UNDO_ATTENUATION_AND_OR_PML				S	false
CREATE_SHAKEMAP				S	false
MOVIE_SURFACE		G		S	false
MOVIE_TYPE		G		S	2
MOVIE_VOLUME				S	true
SAVE_DISPLACEMENT				S	true
USE_HIGHRES_FOR_MOVIES		G		S	false
NSTEP_BETWEEN_FRAMES				S	100
HDUR_MOVIE				S	0.0
SAVE_MESH_FILES		G		S	true
LOCAL_PATH	D	G	M	S	./DATABASES_MPI
NTSTEP_BETWEEN_OUTPUT_INFO				S	100
USE_SOURCES_RECEIVERS_Z				S	auto
USE_FORCE_POINT_SOURCE				S	auto
USE_RICKER_TIME_FUNCTION				S	auto
USE_EXTERNAL_SOURCE_FILE				S	false
PRINT_SOURCE_TIME_FUNCTION				S	auto
SEISMO_PATH				S	auto
NTSTEP_BETWEEN_SEISMOS				S	10000
SUPRESS_IRIS-CONVENTION				S	true
SAVE_SEISMOGRAMS_DISPLACEMENT				S	true
SAVE_SEISMOGRAMS_VELOCITY				S	true
SAVE_SEISMOGRAMS_ACCELERATION				S	false
SAVE_SEISMOGRAMS_PRESSURE				S	false
SAVE_SEISMOGRAMS_IN_ADJOINT_RUN				S	false
subsamp_seismos				S	1
USE_BINARY_FOR_SEISMOGRAMS				S	false
SU_FORMAT				S	false
ASDF_FORMAT				S	false
WRITE_SEISMOGRAMS_BY_MAIN				S	false
SAVE_ALL_SEISMOS_IN_ONE_FILE				S	false
USE_TRICK_FOR_BETTER_PRESSURE				S	false
USE_SOURCE_ENCODING				S	false
OUTPUT_ENERGY				S	false
NTSTEP_BETWEEN_OUTPUT_ENERGY				S	10
NTSTEP_BETWEEN_READ_ADJSRC				S	0
READ_ADJSRC_ASDF				S	false
ANISOROPIC_KL				S	false
SAVE_TRANSVERSE_KL				S	false
ANISOTROPIC_VELOCITY_KL				S	false
APPROXIMATE_HESS_KL				S	false

SAVE_MOHO_MESH	D	G		S	false
COUPLE_WITH_INJECTION_TECHNIQUE	D	G	M	S	auto
INJECTION_TECHNIQUE_TYPE	D	G	M	S	auto
MESH_A_CHUNK_OF_THE_EARTH	D	G	M		false
TRACTION_PATH				S	auto
FKMODEL_FILE				S	FKmodel
RECIPROCITY_AND_KH_INTEGRAL				S	false
GEMINI_SYNSEIS_FILE				S	auto
MULTIPLE_SEQUENTIAL_SOURCES				S	auto
SEQUENTIAL_SOURCES_DESCRIPTION_FILE				S	auto
NUMBER_OF_SIMULTANEOUS_RUNS				S	1
GPU_MODE				S	auto
ADIOS_ENABLED		G	M	S	false
ADIOS_FOR_DATABASES	D	G	M		false
ADIOS_FOR_MESH		G		S	false
ADIOS_FOR_FORWARD_ARRAYS				S	false
ADIOS_FOR_KERNELS				S	false

2.3 The pseudo mesh

SPECFEM3D-C uses Cartesian coordinates and Cartesian components of the displacement fields and derived tensor fields. Hence, the computational box is rectangular with prescribed lateral and vertical extents. Within this box, SPECFEM3D-C defines a hexahedral mesh by specifying the number of elements and a spacing along each spatial dimension. To obtain specific values of the Cartesian coordinates, a minimum value for each coordinate is specified in addition. We will call this mesh the "pseudo" mesh because it does not have a physical significance.

The pseudo mesh is defined in the file `DATA/meshfem3D_file/Mesh_Par_file` through the variables `LATITUDE_MIN`, `LATITUDE_MAX`, `LONGITUDE_MIN`, `LONGITUDE_MAX` and `DEPTH_BLOCK_KM`. The number of elements along the lateral dimensions are specified by the variables `NEX_XI` and `NEX_ETA` where `NEX_XI` refers to longitude and `NEX_ETA` to latitude. Note here that the variable names are rather misleading as you are supposed to provide lengths in meters for the extrema of latitude and longitude. So, in reality, these variables are simply extrema for the x-coordinate (longitude), the y-coordinate (latitude) and the depth coordinate ($-z$). x is positive east, y positive north and z positive up. Element edge lengths can be calculated by dividing the box sizes by the number of elements. The element edge length along the vertical direction is specified at the end of the file in the section "Domain regions". If we only define one region, the variable `NZ_END` gives the number of elements along depth. Values for `NEX_XI_END` and `NEX_ETA_END` have to be consistent with `NEX_XI` and `NEX_ETA`.

For reasons that become clear later, always choose symmetric bounds for x and y around 0.

There is an additional file "interfaces.dat" in which we can define the interfaces between the domain regions. For one region, we only need to specify the surface which is here conceived as being defined by a lateral grid whose points may vary with depth. Make sure that you set the variables `NXI` and `NETA` to the number of grid points along each lateral dimension which is one more than the number of elements! In addition, you need to specify the spacings here explicitly and they must be consistent with the lateral dimensions of the mesh and the number of elements.

Moreover, you specify the name of a file which contains the depths of the grid points of the surface (here "interface_01_zcoor.dat"). For a surface without topography, just set all values to zero. Finally, you must provide the number of elements in the vertical direction which should be consistent with `NZ_END`. For use with Gemini, please set `SUPPRESS_UTM_PROJECTION` to true everyhwhere.

Although we will do parallel computations later, set the value of `NPROC_XI` and `NPROC_ETA` to 1! We use a regular mesh (`USE_REGULAR_MESH` is true) and no doubling layers (`NDOUBLINGS` $= 0$). Dimensions of CPMLs are irrelevant because we will use Stacey boundary conditions later for coupling. Also domain material are irrelevant later because we will specify an external earth model. Still, choose here `NMATERIALS = 1`. For plotting the mesh choose `CREATE_VTK_FILES = .true.`.

Action 2.1: SPECFEM's mesh parameter files

Edit the file `Mesh_Par_file` in the subfolder `meshfem3D_files` of `DATA` as described above. Also edit the files `interfaces.dat` as explained above. Either use the file `interfaces_01_zcoor.dat`, if you want a flat surface or create a new one which may contain non-zero values to represent topography.

2.4 Mapping the pseudo mesh to a spherical chunk

To give the pseudo mesh a physical significance we map it to a spherical chunk. To this purpose, we choose grid points with constant value of the y-coordinate to lie on a parallel with constant latitude, and grid points with constant value of the x-coordinate to lie on a meridian with constant longitude. With symmetric bounds around 0, the surface of the chunk covers an area limited by the parallels $\pm\beta_{\max}$ and the meridians $\pm\varphi_{\max}$. Grid points with fixed x and y but differing z are a chosen to lie on a radius from earth's center. Thus, the spherical coordinates of a point with coordinates x, y, z in the pseudo mesh are given by:

$$\begin{aligned} r &= R_E + z \\ \beta &= \frac{\pi}{2} - \vartheta = \frac{y}{R_E} \\ \varphi &= \frac{x}{R_E}\,, \end{aligned} \tag{1}$$

where β is latitude, φ longitude and r radius. R_E is the radius of the earth. Since, we use SPECFEM3D-C which uses Cartesian coordinates, we need to transform the spherical coordinates to Cartesian chunk coordinates x_c, y_c, z_c relative to a pole at the equator at zero longitude with x_c pointing east, y_c pointing north and z_c pointing up with zero value at the surface. To accomplish this we first calculate standard Cartesian coordinates x_s, y_s, z_s with the z_s-axis through the north pole, the x_s-axis through the zero meridian at the equator and the y_s-axis through the equator at 90 degrees longitude with $x_s = 0$ on the equator and not at the earth's center:

$$\begin{aligned} x_s &= r\cos\beta\cos\varphi - R_E \\ y_s &= r\cos\beta\sin\varphi \\ z_s &= r\sin\beta\,. \end{aligned} \tag{2}$$

Since we want the z_c-axis through zero longitude at the equator, we first interchange z_s and x_s, i.e. $x'_s = z_s$ and $z'_s = x_s$ and $y'_s = y_s$:

$$\begin{aligned} z'_s &= r\cos\beta\cos\varphi - R_E \\ y'_s &= r\cos\beta\sin\varphi \\ x'_s &= r\sin\beta\,. \end{aligned} \tag{3}$$

Now, the z'_s-axis is at the right place but the x'_s-axis points to the north instead east. Thus, we also interchange x'_s and y'_s, i.e. $x_c = y'_s$ and $y_c = x'_s$ and $z_c = z'_s$:

$$\begin{aligned} z_c &= r\cos\beta\cos\varphi - R_E = r\cos\left(y/R_E\right)\cos(x/R_E) - R_E \\ x_c &= r\cos\beta\sin\varphi = r\cos(y/R_E)\sin(x/R_E) \\ y_c &= r\sin\beta = r\sin(y/R_E)\,. \end{aligned} \tag{4}$$

This is the mapping from the Cartesian coordinates of the grid points in the pseudo mesh to the corresponding Cartesian coordinates of the grid points in the spherical chunk.

The inverse transform is given by:

$$\begin{aligned} r &= \sqrt{x_c^2 + y_c^2 + (z_c + R_E)^2} \\ \beta &= \arcsin(y_c/r) \\ \varphi &= \arcsin(x_c/(r\cos\beta)) \\ x &= R_E\,\varphi \\ y &= R_E\,\beta \\ z &= r - R_e\,. \end{aligned} \tag{5}$$

You may find the code doing these transformations in the `SPECFEM3D/src` folder under `meshfem3D/cartesian_box_spherical_chunk_mappings.f90`.

2.5 Creating the mesh using the internal mesher

The internal mesher of SPECFEM3D-C also reads the main parameter file (`DATA/Par_file`). This large file contains many different configuration variables but only very few are used by the mesher itself. If you want to activate the mapping to the spherical chunk, you need to set the variable `MAP_TO_SPHERICAL_CHUNK` to true. Please do not use the option `MESH_A_CHUNK_OF_THE_EARTH` which was conceived for a different purpose. When using the mapping, you should set `SUPPRESS_UTM_PROJECTION` to true and `TOPOGRAPHY` to false.

Beyond that the mesher uses the following configuration variables of the main parameter file: `NGNOD` (= 8) is the number of nodes used for the shape functions. `PML_CONDITIONS` and `PML_INSTEAD_OF_FREE_SURFACE` should be both false as we will use Stacey boundary conditions later. Set `COUPLE_WITH_INJECTION_TECHNIQUE` to true and `INJECTION_TECHNIQUE_TYPE` to 4 for GEMINI and 2 for AxiSEM. `ADIOS` remains disabled.

2.6 Decomposition of the mesh

Since we will run SPECFEM3D-C on a parallel machine later, the mesh needs to be decomposed into subregions that are assigned to the individual processors. This is done by

Action 2.2: Creating the mesh with xmeshfem3D and xdecompose_mesh

Edit the variables tagged with "M" for "meshfem" and "D" for "'decompose" in SPECFEM's `Par_file` and enter values as described above. Then switch to the inversion folder and run the SPECFEM3D-C mesher `xmeshfem3D` on a single computing node. The mesher will write several files to the folder `./MESH` for later use. That this happens, please check in the file `setup/constants.h` that the variable `SAVE_MESH_AS_CUBIT` is set to true. Plot the mesh by importing the file `MESH/proc000000_mesh.vtk` into `paraview`. Logging information about the properties of the mesh is provided in `OUTPUT_FILES/output_meshfem3D.txt`. Then run `xdecompose_mesh` with the proper command line arguments also on one node.

the executable `xdecompose_mesh`. It is also run from the inversion folder on only one node. It reads the main parameter file and takes the variable `LOCAL_PATH` from there. You should keep the preset name `DATABASES_MPI` for this variable. Other important parameters for coupling with GEMINI are `STACEY_ABSORBING_CONDITIONS = .true.` and `STACEY_INSTEAD_OF_FREE_SURFACE = .false.`. The executable takes 3 arguments: the number of processes to be used later when running SPECFEM in parallel, the folder `./MESH` and the local path `DATABASES_MPI`. Output is written to the local path.

2.7 Generating the parallel databases for SPECFEM3D-C

The final step for setting up the SPECFEM computation is to create the process-specific databases by going into the inversion folder and running the executable `xgenerate_databases` on the number of nodes specified when doing the decomposition. Several parameters of the SPECFEM parameter file beyond the ones used by the mesher need to be set correctly: `SIMULATION_TYPE` is 1, `NOISE_TOMOGRAPHY` is 0 and `INVERSE_FWI_FULL_PROBLEM` is false. `NPROC` should be set to the the number of processes specified when doing the decomposition. When using ASKI, `MODEL` should be set to "aski" and you need to prepare at least an ASKI background model (see the following subsection). If you do no want to use ASKI in the first place and use an internal SPECFEM model, set `MODEL` for example to "1d_prem". Note that whenever you want to change the earth model used by SPECFEM you need to recreate the parallel databases!

Note 2.1: 3D ASKI model

Note: In order to set up a 3D ASKI model using `convertFmtomoToKim` you need the ASKI main file which is created by SPECFEM. So, in this case it is mandatory to first run SPECFEM with a different model to produce the ASKI main file and then to activate the ASKI 3D model and rerun `xgenerate_databases`.

Other parameters like `APPROXIMATE_OCEAN_LOAD`, `ATTENUATION` and `ANISOTROPY` should be set to false. Make sure that `SAVE_MESH_FILES` is set to true. The two parameters `USE_FORCE_POINT_SOURCE` and `USE_RICKER_TIME_FUNCTION` are only reproduced in the output file of `xgenerate_databases` but do not influence its behaviour. These parameters will be later set by a Python script before running `xspecfem3D`. Please set remaining parameters not mentioned here as proposed in the table.

To cooperate with ASKI, some ASKI specific extensions have been added to SPECFEM which are controlled by a separate parameter file called `Par_file_ASKI`. To use an ASKI background model here, set the following parameters in `Par_file_ASKI`:

Variable	Used in		Recomm. Value
`USE_ASKI_BACKGROUND_MODEL`	G		true
`FILE_ASKI_BACKGROUND_MODEL`	G		name of file
`IMPOSE_ASKI_INVERTED_MODEL`	G		true/false
`FILE_ASKI_INVERTED_MODEL`	G		name of file

Note 2.2: ASKI parameter file for SPECFEM

Note: In all following iterations, the `Par_file_ASKI` will be generated automatically by `prepare_specfem_kernel.py`. But, for initiating the SPECFEM parallel database, maybe even independently from ASKI, it is more convenient to set the few parameters listed above manually.

One may also specify a 3D model for computation with SPECFEM using ASKI tools. For this purpose, you may use the ASKI program "convertFmtomoToKim" which can read 3D models from FMTOMO and produce values for non-inverted properties using predefined correlations. Since using this program needs more knowledge about ASKI we postpone the description to later in the document. Whenever you want to change the earth model used by SPECFEM you need to recreate the parallel databases!

Action 2.3: Creating SPECFEM's parallel databases

Edit in SPECFEM's `Par_file` the variables tagged with "G" as described above. Then run `xgenerate_databases` with the number of processes specified during decomposition to finish preparing SPECFEM computations. When injection of GEMINI wavefields is activated, `xgenerate_databases` will produce a file `procXXXXXX_absorbing_boundaries_for_gemini.txt` which contains the coordinates of the boundary GLL points and the components of the normal vector there owned by the specific process. This file is used later when preparing the GEMINI wavefields to be injected at the boundaries. The output of `xgenerate_databases` also goes to `DATABASES_MPI`, among them VTK files for model parameters, MPI points and the free surface for each parallel process. These may be either plotted directly or be combined into one VTK file using the executable `xcombine_vol_data_vtk`. The run will also produce diverse output in the folder `OUTPUT_FILES`. One important file there is `output_generate_databases.txt` which provides a comprehensive information about the setup. It also gives an estimate of the maximum period resolved and the maximum recommended time step. There is also a log file concerning the creation of the ASKI 1D or 3D external models if activated.

Command line arguments of `xcombine_vol_data_vtk` are as follows:

- an index for the first slice (e. g. 0) and one for the last slice (e. g. 39),

- the property to be plotted (e. g. vp),
- the folder where the VTK-files of the slices are located (e. g. `DATABASES_MPI`),
- the folder to which the combined VTK file is written,
- a flag for the resolution, either low = 0 or high = 1.

3 Using GEMINI for wavefield injection

For teleseismic inversions, one may save computational costs by solving the forward problem in a hybrid way: an efficient method specialized to 1D or 2D earth models is used to produce wavefields from remote sources which are injected into the SPECFEM computational box. Using GEMINI for this purpose requires some preparatory steps which are described in the following.

3.1 Preliminary actions and considerations

3.1.1 Subfolder for Gemini

All files related to GEMNI in- and output are assembled in a subfolder of the inversion folder (called `GEMINI` or `gemini`). As for SPECFEM, the Gemini executables are run from the inversion folder and not from the Gemini subfolder! The GEMINI subfolder should have been generated already by the Python script `createIterationFolders.py`.

3.1.2 Geographical location of model box center

You need to define the geographical coordinates of the center of the model box to allow a connection of the Specfem model domain to some real-world volume. Provide latitude and longitude of the center of the box in degrees. These are contained in the main parameter file.

3.1.3 Frequency range

Think about the frequency range required for computation of synthetic seismograms. It is controlled by the `FMAX` parameter. Green FK spectra are computed up to this frequency. Since seismograms need to be low-pass filtered later, include the transition from the pass to the stop band of your filter into this frequency range.

3.1.4 Time series length

Choosing the length of the time series `TLEN` is delicate. `TLEN` controls the sampling in the frequency domain by $df = 1/\mathrm{TLEN}$. Since the seismograms are generated from a finite-banded spectrum they are basically periodic and all signals arriving later than `TLEN` are folded somewhere into the time window (though strongly diminished due to the complex-omega trick). Still, such signals may be disturbing if they are excluded from the time window.

3.1.5 Slowness limits

The wavenumber range of the Green FK spectra is controlled by the slowness limits. Since $k = \omega p$, a given slowness corresponds to a straight line in the FK domain up to which the spectrum is calculated. The inverse slowness limit indicates the smallest horizontal apparent velocity of wave signals that are still contained in the spectrum. Thus, by choosing an appropriate value, apparently slow waves like surface waves or multiple shear waves can be excluded from the spectrum. But care is required here. It should be checked for some test cases that the truncated FK spectrum yields the same waveforms for the desired phases as the complete one. The second slowness limit is used for frequencies above half of `FMAX`. Here the chosen slowness limit may be more restrictive as this part of the spectrum will be low-pass filtered later anyway.

3.1.6 Source node radii

When computing Green FK spectra, different files are created according to the radius of the source. Think about the number and location of source nodes needed to do forward computations for any earthquake given the limited accuracy of source depths in earthquake location.

3.1.7 Event list

To do test or production forward runs you need to provide a list of events as specified in section 5.2.3 for which computations are done. Start with one event or a few ones for testing purposes before trying a full catalogue.

3.2 The Gemini parameter file

As with SPECFEM, also GEMINI has a main parameter file where you can set parameters for the different executables that come with GEMINI in a single place. This avoids inconsistencies when running different executables at different times. The following GEMINI executables should be run using this parameter file:
`computeGreenFKSpectraForASKI` (GSA),
`computeGreenFKSpectraForSynthetics` (GSS),
`computeTravelTimeTable`,
`computeSyntheticSeismograms` (SS),
`computeSyntheticsForASKI` (SA),
`computeSyntheticsForSpecfemCoupling` (SC),
`computeSeismogramsFromSpectraForSpecfemCoupling` (SSC),
`prepareSpecfemCoupling`,
`computeSourceWavelet`,
`computeStationFilter` (SF).

The basic idea is to first compute frequency-wavenumber spectra for selected source depths containing all components needed for an arbitrary moment tensor or force source, and then later to compute from them synthetic seismograms for receivers at any depth, distance and azimuth for any moment tensor or force source sitting at the specified source depths. The FK spectra need to be recomputed only if a new earth model is used or if new additional source depths are desired.

The GEMINI parfile specifies the following variables:

Variable	Comment	Rec. Value
Green FK spectra and travel times		
ACCURACY		0.00001
EARTH_MODEL		name of file
DSVBASENAME		base name
ATTENUATION_MODE		choose
EXTERNAL_NODES_REARTH		6371000.0
EXTERNAL_NODES_FROM_FILE		1 / 0
EXTERNAL_NODES_HDF_FILE	if from file	file name
EXTERNAL_NODES_HDF_PATH	if from file	gllUniqueRadii
EXTERNAL_NODES_NBLOCKS	not from file	
EXTERNAL_NODES_NNOD	not from file	
EXTERNAL_NODES_DR	not from file	
EXTERNAL_NODES_SHIFT	not from file	
REC_NODE_RADIUS		
NSNOD	GSA	
SOURCE_NODE_RADII	GSA	
SOURCE_TYPE		0 / 1
TLEN		
SLOWNESS_LIMIT_1		
SLOWNESS_LIMIT_2		
GLOBAL		1 / 0
XLEN		
FMAX		
WAVENUMBER_MARGIN_FRACTION		0.2
STRESSFLAG		false
Synthetic seismograms		
FILE_EVENT_LIST		
EVENT_FILTER_TYPE		
EVENT_FILTER_SPECS		
SEISMO_TYPE	SS, SA	
AUTOMATIC_TIMESHIFT		true
TIME_SHIFT_BUFFER		20
PHASE_WINDOW_LENGTH		100
COMPUTE_SPECTRA	SC	
PATH_SYNTHETICS	SC, SSC	
FILE_SEISMOGRAMS	SSC	
FILE_SYNSEIS_FOR_ASKI	SA	
FILE_SYNSEIS	SS	
FILE_STATION_LIST	SA, SS, SF	
FILE_STATION_FILTER	SA, SS, SF	
STATION_FILTER_TYPE	SA, SS, SF	UNIT
Prepare SPECFEM coupling		
SPECFEM_DATABASES_MPI		./DATABASES_MPI
BOUNDARY_DATA		name of file
BOX_DIMENSIONS		80 60 30

BOX_CENTER		
NPROC		
RADIAL_TOLERANCE		1000
DISTANCE_TOLERANCE		0.001
Travel time table		
FILE_RAY_TABLE		
TURNING_NODES_NNOD		444
TURNING_NODES_DR		5000
RAY_TYPE		P
Moment rate function		
DATA_PATH_TO_EVENTS		/data/AlpArray_Data/dmt_database/
DATA_SUBEVENT_PATH		processed/
PATH_MEASURED_SEIS		measured_data
FILE_SYNSEIS_AT_STATIONS		
SIGSTF		0.005
MAGSTF		1.0
STF_SNR_BOUND		15
STF_RMS_BOUND		0.15
STF_MISFIT_BOUND		1.2

3.3 Preparing Gemini-to-SPECFEM coupling

Once `xgenerate_databases` has been run and coupling to Gemini was activated, the process specific files containing the location of the boundary points and the normal vectors are available. The executable `prepareSpecfemCoupling` reads these files and produces one long list by concatenating the boundary points associated with each process. Then, using the information about the location of the center of the computational box (`BOX_CENTER`), it calculates geographical spherical coordinates of the boundary points (r, θ, ϕ). Assuming that the SPECFEM pseudo-mesh is regular and that mapping to a spherical chunk was activated, the GLL-points should lie on spherical shells of constant radius. Exploiting this fact, the boundary points are sorted according to radius and their position in the original list is stored in an index array. Moreover, the unique radii of the spherical shells are extracted from the sorted list. All this information is written to a HDF file that is later used by Gemini for computing synthetic seismograms at the boundary points (`EXTERNAL_NODES_HDF_FILE`).

You need to specify the name of the HDF output file with boundary points (`BOUNDARY_DATA`) relative to the inversion folder, the SPECFEM database `SPECFEM_DATABASES_MPI`, the SPECFEM box dimensions, (`BOX_DIMENSIONS`), i. e. number of elements along each spatial direction, the geographical coordinates in degrees of the center of the SPECFEM box (`BOX_CENTER`) (these are of no importance for SPECFEM itself but are needed to connect the SPECFEM box to some real volume in the earth), the number of processes used with `xgenerate_databases` (`NPROC`) and some tolerances for radius and angular distance that define when two points are really different (`RADIAL_TOLERANCE` and `DISTANCE_TOLERANCE`).

Action 3.1: Preparation of boundary information for injection

Edit the Gemini parameter file as described above. Make sure that the box dimensions are equal to those specified in the `Mesh_Par_file` and that the coordinates of the box center are identical to those specified in the main parameter file. Then, run the executable `prepareSpecfemCoupling` with the name of the Gemini parameter file as only command line argument from the main inversion folder on one process.

3.4 Computing Green frequency wavenumber spectra

The next step is the computation of frequency-wavenumber (FK) spectra for a given earth model, predefined source and receiver radii from which synthetic seismograms may be calculated for any moment tensor or force source and for any epicentral distance and back-azimuth. This is done with the executable `computeGreenFKSpectraForASKI`. Please go through the table of parameters (GSA) required for this program. Important ones are: the earth model (`EARTH_MODEL`) which is a json-file containing radial knots and physical properties in SI-units. `DSVBASENAME` specifies the base file name of the FK spectra to be computed. The base name will be completed by the program by the source type (MOMENT or FORCE) and the index of the source node. In addition, a basename.meta file will be created. `ATTENUATION_MODE` can either be `ELASTIC`, `ATTENUATION_ONLY`, `DISPERSION_ONLY` or `ATTENUATION_AND_DISPERSION`.

The external radial nodes specify the receiver radii and are taken from the file with the boundary points written by `prepareSpecfemCoupling` (`EXTERNAL_NODES_HDF_FILE`). Therefore, you must set `EXTERNAL NODES_FROM_FILE` to 1. And you must specify the HDF path to `gllUniqueRadii` which is hardcoded into `prepareSpecfemCoupling`. The parameter `REC_NODE_RADIUS` specifies the radius where your seismometers are placed, typically at the surface at 6371000 m. The potential earthquake sources are specified by their number (`NSNOD`), their radii (`SOURCE_NODE_RADII`) and their type (`SOURCE_TYPE`) where 1 stands for moment tensor and 0 for force sources. It is best to look up the `gllUniqueRadii` in the HDF file written by `prepareSpecfemCoupling` and choose radii from there which are closest to the source radii in the event file. Choosing more nodes is not helpful as, finally, the nodes closest to the true source radii are taken only.

Important properties of the FK spectra are defined by `TLEN` giving the desired length of the seismogram and also indirectly specifying frequency sampling through $df = 1/\texttt{TLEN}$. The slowness limits define the upper wavenumber limits of the spectra. The upper limit is given by $k_{max} = \omega p_{lim} + k_{off}$ where k_{off} is a base value controlled by `WAVENUMBER_MARGIN_FRATION`. This implies that all waves with horizontal slowness higher than the slowness limit (i. e. the slow waves like surface waves and multiple S-waves) are eliminated from the FK spectrum. You may set two different slowness limits, one for the frequency range below $0.5\,\texttt{FMAX}$ and one above. For hybrid teleseismic modeling you must set `GLOBAL` to 1. `XLEN` is not used in this case. Finally, the most important parameter is `FMAX` defining the upper frequency limit up to which the spectra are calculated. When choosing this parameter you should consider that the seismograms will have to be low-pass filtered later to avoid cut-off oscillations. Thus, you should include the transition from the corner frequency to the stop band of the filter into `FMAX`.

The executable `plotGreenFKSpectra` may be used to visualize the FK spectra.

Action 3.2: Producing Green frequency wavenumber spectra

Choose values in the Gemini parameter file as explained above and run the executable `computeGreenFKSpectraForASKI` with the name of the Gemini parameter file as only mandatory command line argument. This is a parallel code which may be run on several processes. Check the results by visualizing the spectra using `plotGreenFKSpectra`.

3.5 Computing a simple travel time table

To automatically limit the seismograms to the interesting wave packets (e. g. direct P wave) you may precompute a travel time table which allows a very fast evaluation of the travel time from the source to each boundary point. This is done by the executable `computeTravelTimeTable`. Many parameters are identical to those for computing Green FK spectra (see table). In addition, you must specify a name for the output file (`FILE_RAY_TABLE`) and the `RAY_TYPE` which can be either P or S. Rays are calculated for a set of turning radii (where the ray is horizontal) which should entirely lie in the lower mantle to avoid triplications. For example choosing the distance between turning nodes (`TURNING_NODES_DR`) to 5000 m we need 444 nodes to cover the lower mantle (from the CMB at 3480 km to 5700 km).

Action 3.3: Creating a travel time table

Choose values in the Gemini parameter file as explained above and run the executable `computeTravelTimeTable` with the name of the Gemini parameter file as only mandatory command line argument. This code is run on one process. To visualize rays you may use the Python ray classes defined in `sphericaRay.py`. Application of these classes is exemplified in the Python script `testSphericalRay.py`.

3.6 Computing synthetic injection seismograms at the boundary points

Everything is ready now to compute synthetic seismograms at the boundary points. This is done by the executable `computeSyntheticsForSpecfemCoupling`. You can do this in one step (`COMPUTE_SPECTRA` is false) or you may first compute spectra and compute seismograms in a second step. There are a few parameters that have not been set before. The filtering of the seismograms derived from the FK spectra is specified by `EVENT_FILTER_TYPE` which should be set to `BUTTERWORTH_BANDPASS` (the only one that is currently implemented) and `EVENT_FILTER_SPECS` where you define order and corner frequency of high and low pass filter. It should comply with the shortest period that is stable in the SPECFEM grid. Injected seismograms should not contain shorter periods or higher frequencies. Morever, you need to specify an output folder for the seismograms (or spectra) with `PATH_SYNTHETICS` and a file giving the events to be processed (`FILE_EVENT_LIST`). The file names for the injection seismograms will be the path followed by the event id and the extension "_seis.hdf" or "_spec.hdf".

Finally, the parameter `AUTOMATIC_TIMESHIFT` controls whether the travel time table is used to automatically calculate the start time of the seismograms. For teleseismic waves, the earliest arrival will occur at the bottom face of the model box at the point with the smallest

epicentral distance from the hypocenter. This travel time minus a buffer given by the parameter TIME_SHIFT_BUFFER (denoted as reduction time t_{red}) is chosen as the starting time of all boundary point seismograms. The seismograms are shifted to the left by t_{red} by applying a phase shift to the Fourier transform (in this way phases originally arriving after the time window may be moved into the time window). The end time of the seismogram is calculated from the maximum travel time at the upper model face plus two times the length of the phase window, PHASE_WINDOW_LENGTH. Travel times at individual boundary points are used to zero values that belong to times before travel time minus buffer length. Thus, the common length of all boundary point seismograms is given by $t_{\mathrm{end}} - t_{\mathrm{red}}$. Values of window length, time shift buffer, minimum travel time, reduction time and end time are written to the seismogram HDF files. Moreover, to avoid undesired later phases being injected at the box boundary, the injection seismograms are tapered to zero at the end of the phase window. The taper length is controlled by the parameter END_TAPER_LENGTH in the Gemini paramter file.

When you choose to first compute frequency spectra, the executable computeSeismogramsFromSpectraForSpecfemCoupling does the transform into the time domain. Going this way allows to modify the filtering and window selection in a later stage without the cost of a full recomputation of the synthetics.

Both executables offer a command line option -stf to perform a convolution with an event specific moment rate function that is taken from an event subfolder of PATH_MEASURED_SEIS under the file name "stf.txt". How this moment rate function is determined from the observed data is described under section 4.1. Note that a convolution of this moment rate function with a displacement Green function yields particle velocity. Hence, the computed injection seismograms represent particle velocity and the resulting SPECFEM displacement seismograms also represent true particle velocity. When comparing measured seismograms which are particle velocity with SPECFEM seismograms one should therefore take the SPECFEM displacement seismograms with extension semd! The same applies to the Fourier transforms of measured and SPECFEM seismograms. When the -stf-option is activated, the phase window length is taken from a file phasewinlen.txt located in the event subfolder of PATH_MEASURED_SEIS. It is generated when determining the moment rate function.

Action 3.4: Injection seismograms

Make relevant edits in the Gemini parameter file as described above. Run the executable computeSyntheticsForSpecfemCoupling in parallel mode with the Gemini parameter file as mandatory argument and the flag -stf by which convolution with a moment rate function taken from PATH_MEASURED_SEIS/evid/ is activated. For testing purposes you may choose the -single option and do the computation for one event only. For production, make sure that FILE_EVENT_LIST points to the "good" events only. Parallelization of the program is achieved by sharing the boundary points among the processes.

4 Processing of observed and synthetic data

In order to solve the kernel linear system, we also need the difference between observations and predictions which form the right hand side of the equation system. For this purpose, we need to extract from the observed seismograms Fourier transformed data samples and we need to do

the same for the seismograms predicted by SPECFEM during forward computation. However, synthetic seismograms will fit only well to the observed ones if the moment or moment rate function which characterizes the temporal evolution of the earthquake process is taken into account. For this reason, the first step in preparing Fourier transformed data samples done here is to determine a moment rate function. Since the moment function itself is a monotonically increasing function of time, the moment rate function should be non-negative. We will use a non-negative least squares approach to solve this task.

4.1 Theory of moment rate function estimation

The currently implemented way of obtaining a moment rate function from observed seismograms is to calculate Gemini synthetics for the 1D reference model within a selected time window (here the direct P-wave), filter the data in the same way as the synthetics, cut the data to this time window and then construct a non-negative moment rate function per event that transforms the synthetic waveforms into the observed waveforms in a least-square sense. Since due to uncertainites in the earthquake origin time the observed P-wave train may arrive earlier than the synthetic one, we allow for acausal moment rate functions which are non-zero for negative times.

Note 4.1: Moment rate function

Note: moment rate function estimation is currently organized in the geminiUnified project because synthetic seismograms are currently computed using Gemini.

4.1.1 Non-causal convolution

We denote the moment rate function by $h(t)$ and the synthetic waveform by $s(t)$. Their discrete versions are sampled with identical interval dt and denoted by $h_i = h((i-1)dt)$ or $s_k = s((k-1)dt)$, respectively. The samples h_i are defined in the range from $-N_l < i \leq N_r$ and the synthetic samples in the range from $1 \leq k \leq K$. Thus, N_r is the number of samples of the moment rate function at positive times, and N_l gives the number of samples at negative times including sample zero. For example, the sample h_0 belongs to time $-dt$. Let us denote the output samples by f_n. We start the convolution with h_1 aligned to s_1, h_0 aligned to s_2 and so forth, as shown below,

```
5 4 3 2 1 0 -1 -2                  h(-t)
        1 2  3  4 5 6 7 8 9        s(t)
```

then shift the synthetic to the left sample by sample and end the convolution when h_1 is aligned with s_K:

```
          5 4 3 2 1 0 -1 -2        h(-t)
1 2 3 4 5 6 7 8 9                  s(t)
```

In this way, the number of output samples is identical to the number of synthetic samples. More generally, when the sample f_n is calculated, the upper limit of the sum over the moment rate

samples is at $i = \min(n, N_r)$ and the lower limit at $i = \max(-N_l + 1, n - K + 1)$. Thus, we may write the acausal convolution as (with $1 \leq n \leq K$)

$$f(n) = \sum_{i=\max(-N_l+1,n-K+1)}^{\min(n,N_r)} h(i)\, s(n-i+1)\,.$$

As an example, setting $N_l = 3$, $N_r = 5$ and $K = 9$, result in the following system of equations for f_n with $1 \leq n \leq 9$:

```
|f(1)|    | s(4)  s(3)  s(2)  s(1)  0     0     0     0     | | h-2|
|f(2)|    | s(5)  s(4)  s(3)  s(2)  s(1)  0     0     0     | | h-1|
|f(3)|    | s(6)  s(5)  s(4)  s(3)  s(2)  s(1)  0     0     | | h0 |
|f(4)|    | s(7)  s(6)  s(5)  s(4)  s(3)  s(2)  s(1)  0     | | h1 |
|f(5)| =  | s(8)  s(7)  s(6)  s(5)  s(4)  s(3)  s(2)  s(1)  | | h2 |
|f(6)|    | s(9)  s(8)  s(7)  s(6)  s(5)  s(4)  s(3)  s(2)  | | h3 |
|f(7)|    | 0     s(9)  s(8)  s(7)  s(6)  s(5)  s(4)  s(3)  | | h4 |
|f(8)|    | 0     0     s(9)  s(8)  s(7)  s(6)  s(5)  s(4)  | | h5 |
|f(9)|    | 0     0     0     s(9)  s(8)  s(7)  s(6)  s(5)  |
```

Inserting the data samples for the f_n, this system is solved in a least square sense for the h_i using a non-negative least squares algorithm.

4.2 Computation of synthetic seismograms

Computation of synthetic seismograms with GEMINI needs to be done in the following order: computation of Green FK spectra for synthetic seismograms, calculation of station filters and then computation of seismograms.

Run the parallel executable `computeGreenFKSpetraForSynthetics` on the cluster which produces Green FK spectra for potential sources at all external nodes and a receiver at the specified receiver node radius. It takes the same parameters as the executable `computeGreenFKSpectraForASKI`.

Compute station filters by running the executable `computeStationFilter` from the inversion folder. The output goes into a single HDF file for all stations. Place this file into the `PATH_MEASURED_SEIS` folder. Edit the Gemini parameter file accordingly, in particular the variables `FILE_STATION_FILTER` and `STATION_FILTER_TYPE`. The station file has the same format as ASKI's station file (see 5.2.2) and should contain all stations used in the experiment. Station and network name together with spherical coordinates and elevation in meters should be specified in the file. The first line should be filled by an "S" for spherical coordinates. Its name is specified in Gemini's parameter file by the variable `FILE_STATION_LIST`.

Now run the Gemini executable `computeSyntheticSeismograms` to calculate synthetic seismograms for events and stations specified in an event list and the station file. The event file should be copied from the inversion folder to the `PATH_MEASURED_SEIS` folder and the first line should be filled by an "S" for spherical coordinates. Parameters for the executable are set through the Gemini parameter file. In addition, it is recommended to activate the option `-split` through which the seismogram output is written into event subfolders which are created by the executable if not already present. It is also possible to activate the option `-single`

to just work on a single event. The option `--ascii` activates the generation of Ascii seismogram output in addition to HDF. Do not activate the `-stf`-option as the moment rate function is still to be determined. Place all the results into the folder `PATH_MEASURED_SEIS`.

Consider here the specification of the event filter which is applied to all seismograms set through the parameters `EVENT_FILTER_SPECS` and `EVENT_FILTER_TYPE`. It should comply with the shortest period that is stable in the SPECFEM grid. Synthetic seismograms should not contain shorter periods or higher frequencies.

Activate the `AUTOMATIC_TIMESHIFT` parameter in the Gemini parameter file. If not, you may set a reduction time in the event list. With automatic time shift activated keep the variables `PHASE_WINDOW_LENGTH` and `TIME_SHIFT_BUFFER` at the same values as chosen for the injection seismogram. Phase window length can later be shortened when determining the moment rate function.

Plot the resulting seismogram gathers using `plotHdfSeismogramGather` for HDF files and `plotAsciiSeismogramGather` for ASCII seismogram files. The graphics is interactive and allows scrolling, zooming and scaling and many other things. By specifying two seismogram gathers on the command line, one may also plot an overlay of two gathers.

Action 4.1: Gemini synthetic seismograms

Follow the steps described above to compute Gemini synthetic seismograms for moment rate function estimation.

4.3 Moment rate determination

Once the synthetic seismograms are produced, the executable `computeSourceWavelet` is used to determine the moment rate function. This executable iterates over the event list and takes the synthetic seismograms from the previously created event subfolders. It also assumes that the unfiltered (raw) but instrument-corrected observed seismograms can be found in an event specific subfolder as MiniSEED files. The path to these files is constructed in that order from the parameter `DATA_PATH_TO_EVENTS`, the event ID and the parameter `DATA_SUBEVENT_PATH`. Both paths must be specified in the Gemini parameter file. The MiniSEED files themselves follow the naming convention of "netcode.staname.location.bandcode" where the last letter of bandcode specifies the component. In the current implementation, `computeSourceWavelet` searches for the locations "empty, 00, 01 and 02" and bandocde "HHZ".

Then, `computeSourceWavelet` goes through the stations and checks if an observed waveform is available. If not, this station is skipped. If yes, the observed waveform is read from file and it is checked whether the data contains the time window of the synthetics. If not the station is skipped, too. If yes, the observed waveforms are detrended, tapered and filtered in the same way as the synthetics and their length and sampling interval are adjusted to those of the synthetics by truncation and interpolation. During this processing, the SNR of the observed waveform is estimated from the ratio of the energy of the waveform after and before the theoretical arrival time of the P-phase. If SNR is below a bound set in the parameter file, the station is eliminated. The code also checks for problems with instrument responses and automatically corrects if log-10 RMS ratios range between -12 and $+12$ in steps of 3. All other stations are eliminated from the calculation.

If SNR is acceptable a moment rate function is determined by constructing a filter that converts the synthetic into the observed waveform in a least-square sense. Further potentially bad stations are identified and eliminated based on the misfit between moment rate function convolved synthetic and observed waveforms.

After running the code, numerous output files are generated. First, the filtered and interpolated data are written to files `data_net.staname_UP` and the convolved synthetics are written to `conv_net.staname_UP` into a subfolder `per_trace`. In addition, gathers of the filtered data and convolved synthetics, respectively, are written as well allowing plotting of all traces and also overlaying data and synthetics. Moreover, the original synthetics are written as HDF and Ascii gathers for plotting. In addition, the filtered but untruncated data are written to an Ascii file for plotting. As output, the moment rate function is written to a text file `stf.txt` and information about misfits, amplitude deviations and SNR of each station into the file `trace_misifits_snr.txt`. The chosen phase window length is written to file `phasewinlen.txt`. Stations with misfit greater than the parameter `STF_MISFIT_BOUND` and amplitude deviations greater that `STF_RMS_BOUND` are successively eliminated from the calculation. A file `excluded-stations` may be created with stations to be excluded in a further run. A cleaned station file and a block for the data-model-space file is also written to file. A protocol of the run is stored in `computeSourceWavelet.log`.

Carefully check the moment rate functions, the trace misfits, the SNR, and the amplitude deviations by plotting them with the Python script `plot_trace_misfits.py`. Also check if the phase window length is appropriate or needs to be shortened. Start with the full phase window and then check if adjustment is needed. Depending on the chosen length of the phase window you may want to adjust the length of the negative and positive parts of the moment rate function as well. Also take care to obtain a smooth and not too spiky moment rate function that nicely tapers out at both ends.

The executable `computeSourceWavelet` takes the following parameters from the Gemini parfile:

- `PATH_MEASURED_SEIS` is a folder below which event subfolders are created to store synthetic and observed waveforms and the STF itself.
- `FILE_SYNSEIS_AT_STATIONS` specifies the name of the HDF file containing the synthetics for the considered event.
- `FILE_EVENT_LIST` and `EVENT_FILTER_TYPE` and `EVENT_FILTER_SPECS` are already used when computing synthetics and should stay unchanged.
- `SIGSTF` is a parameter for smoothing the STF. Small values imply strong smoothing.
- `MAGSTF` allows to magnify small signals using a waterlevel method. If no magnification is desired, set the parameter to 1.0.
- `TIME_SHIFT_BUFFER` is already used when computing synthetics and should stay unchanged.
- `STF_SNR_BOUND` defines the SNR below which stations are excluded.
- `STF_MISFIT_BOUND` defines the misfit above which stations are excluded.
- `STF_RMS_BOUND` defines the log10 amplitude deviation above and below which stations are eliminated.

There are also command line arguments:

- Mandatory arguments are the Gemini parameter file and

- the length of the phase window.
- The STF is allowed to be non-zero for negative times to allow a negative time shift of the synthetics if the P-wave arrives earlier in the observed waveforms. The option `-stfpos` specifies the length of the STF for positive times while the option `-stfneg` specifies the length of the STF for negative times.
- With the option `-single`, a single event can be chosen from the event list.

Action 4.2: Moment-rate function

Follow the instructions above and run `computeSourceWavelet`.

4.4 Preparing a station file for kernel wavefield computation

Once source time functions have been determined for the events listed in the ASKI event file, there will exist event subfolders in the `PATH_MEASURED_SEIS` directory which contain among others a file called `ASKI_clean_station_file`. This file lists all the stations that survived the quality checks done in `computeSourceWavelet`. First run the Python script `findAllStationsUsed.py` to produce a file with a unique list of all stations that passed source wavelet computation. It is these stations which serve as receivers in the SPECFEM kernel displacement and kernel Green tensor computations. But before using them, the assembled station file must be converted to SPECFEM Cartesian coordinates using the Python script `convertGeographicalStationFileToSpecfemCartesian.py`. The resulting file is placed into the main inversion folder under the name specified in `FILE_STATION_LIST`. If SPECFEM computations are done for a mesh without topography, use the option `--ignore_alt`.

4.5 Injections seismograms

If not already done, do not forget to compute the injection seismograms as described in section 3.6. Use the option `-stf` when doing so to invoke the convolution with the previously determined source time wavelets. The moment rate functions are taken from the event subfolders of `PATH_MEASURED_SEIS` with file name `stf.txt`.

Seismograms from the center line of the vertical boundaries can be created using the program `extractSynseisHdfGatherFromSpecfemCoupling`. It takes the Gemini parameter file and an event id as mandatory parameters and produces a HDF gather containing all seismograms on the W, S, E and N boundary.

5 ASKI with hybrid SPECFEM for forward computation

We are ready now to do hybrid forward calculations with SPECFEM. Since we also want to use these calculations for inversion we do the calculations in the framework of ASKI. Actually, ASKI provides the Python script `prepare_specfem_kernel.py` that helps to run SPECFEM in a consistent way. The behaviour of ASKI is controlled by a few parameter files that act at different levels. There is the main parameter file already mentioned in section 1.1

that contains specifications valid for all iterations to be done which is read by all ASKI executables. Then, there is an iteration specific parameter file called `iter_parfile` which contains parameters that may change during iteration. Finally, for use with SPECFEM, there is a `Par_file_ASKI` that is read by some ASKI-extended SPECFEM routines.

5.1 Model property definition and correlation

There are two parameters in the main parameter file related to model properties: `PROPERTY_SET_NAME` defines the set of material properties used for describing the earth model. ASKI currently handles only one property set, namely `isoVelocitySI` with parameters ρ, v_p and v_s in SI-units. `PROPERTY_CORRELATION_FILE` defines a priori correlations between properties inverted for and other properties needed for SPECFEM computations. For example, we may only invert for P-wave velocity but assume that a change in P-wave velocity may also imply a change of shear-wave velocity and density in the earth. This means, we pull along density and shear wave velocity with P-wave velocity. If we assume such a correlation, it has consequences for kernel computation as well as for the model update.

Assume that the true perturbations in the earth are given by $\Delta\rho$, Δv_p and Δv_s, while our inverted properties are $d\rho$, dv_p and dv_s. We can define a correlation between the **relative** model changes as follows:

$$\begin{aligned}
\frac{\Delta\rho}{\rho} &= b_{11}\frac{d\rho}{\rho} + b_{12}\frac{dv_p}{v_p} + b_{13}\frac{dv_s}{v_s} \\
\frac{\Delta v_p}{v_p} &= b_{21}\frac{d\rho}{\rho} + b_{22}\frac{dv_p}{v_p} + b_{23}\frac{dv_s}{v_s} \\
\frac{\Delta v_s}{v_s} &= b_{31}\frac{d\rho}{\rho} + b_{32}\frac{dv_p}{v_p} + b_{33}\frac{dv_s}{v_s}
\end{aligned} \tag{6}$$

From this, we obtain a correlation between absolute model changes as follows:

$$\begin{aligned}
\Delta\rho &= b_{11}\,d\rho + b_{12}\frac{\rho}{v_p}\,dv_p + b_{13}\frac{\rho}{v_s}\,dv_s \\
\Delta v_p &= b_{21}\frac{v_p}{\rho}\,d\rho + b_{22}\,dv_p + b_{23}\frac{v_p}{v_s}\,dv_s \\
\Delta v_s &= b_{31}\frac{v_s}{\rho}\,d\rho + b_{32}\frac{v_s}{v_p}\,dv_p + b_{33}\,dv_s\,.
\end{aligned} \tag{7}$$

Thus, if our inversion result is for example dv_p, we should define the coefficients b_{i2} to calculate true perturbations.

For kernel computation, we have the general relation between the change of some component of displacement δu_m and the model changes $\Delta\rho$, Δv_p and Δv_s:

$$\delta u_m = \int\limits_V \left(K_m^{\rho}\Delta\rho + K_m^{v_p}\Delta v_p + K_m^{v_s}\Delta v_s\right)\,dV\,. \tag{8}$$

Inserting our correlation with the properties inverted for, we get:

$$\delta u_m = \int\limits_V \left(K_m^{'\rho}\,d\rho + K_m^{'v_p}\,dv_p + K_m^{'v_s}\,dv_s\right)\,dV\,, \tag{9}$$

with

$$
\begin{aligned}
K'^{\rho}_{m} &= K^{\rho}_{m} b_{11} + K^{v_p}_{m} b_{21} \frac{v_p}{\rho} + K^{v_s}_{m} b_{31} \frac{v_s}{\rho} \\
K'^{v_p}_{m} &= K^{\rho}_{m} b_{12} \frac{\rho}{v_p} + K^{v_p}_{m} b_{22} + K^{v_s}_{m} b_{32} \frac{v_s}{v_p} \\
K'^{v_s}_{m} &= K^{\rho}_{m} b_{13} \frac{\rho}{v_s} + K^{v_p}_{m} b_{23} \frac{v_p}{v_s} + K^{v_s}_{m} b_{33} \,.
\end{aligned} \tag{10}
$$

The coefficients b_{ij} are specified in the parameter correlation file as follows:
Header Line: model parameterization
Line 2: property name followed by correlation coefficients (b(1,j),j = 1,npar)
Line 3: property name followed by correlation coefficients (b(2,j),j = 1,npar)
Line 4: property name followed by correlation coefficients (b(3,j),j = 1,npar)
The coefficients are stored in a matrix. The file should be placed in the main inversion folder to ensure consistency over several iterations.

Example 1: We invert for v_p only and want to drag ρ and v_s along:

```
isoVelocitySI
rho   0.0   0.3   0.0
vp    0.0   1.0   0.0
vs    0.0   1.8   0.0
```

Example 2: We invert for vp only and want to leave rho and vs unchanged:

```
isoVelocitySI
rho   0.0   0.0   0.0
vp    0.0   1.0   0.0
vs    0.0   0.0   0.0
```

Action 5.1: Material property correlation

Check and edit the file `model_property-correlation` in the inversion folder to values appropriate for your problem.

5.2 Further main parameters

5.2.1 More paths

`PATH_GEMINI` defines the place where all GEMINI related input and output is stored. `PATH_MEASURED_DATA` is the folder where the values of Fourier transform of the observed seismograms at the specified frequencies are put. `PATH_MEASURED_SEIS` specifies the folder for the processed observed data and also the associated GEMINI synthetics used for moment rate function estimation. `PATH_INJECTION_SEIS` is the folder for the injection seismograms computed with GEMINI and read by SPECFEM. `PATH_SPECFEM_INPUT` is SPECFEM's folder for input data. `PATH_SPECFEM_DATABASES` is the place where SPECFEM puts the parallel databases produced by `xgenerate_databases`.

5.2.2 The ASKI station file

Many tasks in ASKI are controlled by a file that specifies the stations to be worked with. The ASKI station list specifies the location of the stations in the SPECFEM cartesian coordinates. It looks as follows:

```
C
PANIX     CH    -133752.963001     71059.4324246     -1800.58929328
OGMY      FR    -385061.514947    -23382.3912922    -11690.1864372
A291A     Z3     77163.7420469     47704.0489728      -645.96684267
```

The first line specifies a cartesian coordinate system (C). The following lines give the station locations in the following order: station name, network code, x-coordinate, y-coordinate, z-coordinate. While SPECFEM3D_C needs the coordinates in its own cartesian coordinate system, station locations are commonly specified in geographical coordinates. For this reason, it is useful to first set up a station file with geographical coordinates and then use the Python script `convertGeographicalStationFileToSpecfemCartesian.py` to produce one with SPECFEM cartesian coordinates. Note that for ASKI inversion, Green tensor computations need to be done for all stations that contribute data to the inversion regardless of the fact that not all of them may be available or useful for all considered events.

Action 5.2: ASKI station file

Setup a file with stations to be used in inversion.

5.2.3 The ASKI event file

Many tasks in ASKI are controlled by a file that specifies a list of earthquake sources to be worked with. The ASKI event file looks as follows (each event one line):

```
C
191207_000000 20191207_000000_000000000    0.0000    -90.0000
70000.000 6.3  740.0  1 -0.475E+17  0.269E+17 -0.222E+17
-0.759E+17 -0.105E+17 -0.105E+17
```

The first line specifies the type of coordinate system, the 2nd line the event by event id, datetime string, latitude, longitude, depth in meters, magnitude, time shift, source type and moment tensor elements. Source type is 1 for moment tensors and 0 for forces. In case of a force source the event entry contains the three force components.

Action 5.3: ASKI event file

Setup a file with events to be used in inversion. Make sure that preceding tasks like the computation of injection seismograms worked with an event file which is identical or a superset of this file.

5.2.4 Setting frequency parameters

ASKI does the inversion in the frequency domain. Wavefields in the time domain are Fourirer transformed and the result is stored only at selected frequencies. These frequencies should be identical to those for which the measured data are available. You may set these frequencies using the parameters `MEASURED_DATA_....` to set frequency stepping `DF`, the number of frequencies `NF` and a set of indices `JF`. The frequencies are obtained as `f = JF*DF`.

5.2.5 Inversion grid center

`INVGRID_CENTER_LAT` and `INVGRID_CENTER_LON` specify the geographical coordinates of the center of the inversion grid. ASKI uses the Cartesian coordinates of the SPECFEM grid for its inversion grid. To assign densities and velocities to the inversion grid from a 3D tomographic model defined on geographical coordinates, we need to know the physical position of the grid. `REARTH` specifies the radius of the earth in meters.

5.2.6 Subfolders for iteration specific output of ASKI

The main parameter file also specifies folders whose contents is iteration-specific. Therefore, these folders are organized as subfolders to an iteration-step directory specified through the parameters `ITERATION_STEP_PATH` and `CURRENT_ITERATION_STEP` in the form `iteration_step_001`. The names of these subfolders are, however, set already in the main parameter file because it does not make sense to change their names during iterating. They serve the following purposes:

- `PATH_ASKI_MAIN_FILES`: This folder contains the ASKI main file which contains information about the inversion grid, cell neighbours and also the SPECFEM GLL-points. In addition, it holds information about the current earth model. Typically, this file is the same for all events to be treated and needs therefore to be calculated only once.
- `PATH_KERNEL_DISPLACEMENTS`: Location of displacements fields emanating from the source or injected at the boundary of the SPECFEM computational box required for computation of sensitivity kernels. These fields are the result of SPECFEM runs discussed later.
- `PATH_KERNEL_GREEN_TENSORS`: Location of displacements field backpropagated from the receiver positions assuming specfic single forces there. These fields are also results of SPECFEM computations.
- `PATH_SENSITIVITY_KERNELS`: Folder where the resulting sensitivity kernels computed from the kernel displacements and kernel Green tensors are stored.
- `PATH_SYNTHETIC_DATA`: Location of Fourier transformed synthetic seismograms.
- `PATH_OUTPUT_FILES`: In this folder, diverse output files of small size produced by the various ASKI executables are collected.
- `PATH_VTK_FILES`: Location of VTK files for visualisation

5.2.7 SPECFEM related parameters of ASKI

The third block in the main parameter file concerns parameters relevant for the ASKI extensions of SPECFEM. They serve the following purposes:

- `USE_ASKI_BACKGROUND_MODEL`: specifies if an ASKI background is used when constructing the parallel databases of SPECFEM.
- `FILE_ASKI_BACKGROUND_MODEL`: the name of the file containing the background model.
- `IMPORT_ASKI_INVERTED_MODEL`: flag specifying if a 3D Fmtomo model should be used when constructing the parallel databases of SPECFEM.
- `FILE_ASKI_INVERTED_MODEL`: the name of the file containing the Fmtomo model.
- `COMPUTE_ASKI_OUTPUT`: basic flag if SPECFEM produces output for ASKI at all.
- `ASKI_HDF_OUTPUT`: flag whether SPECFEM used HDF for ASKI output.
- `ASKI_USES_GPU`: flag if SPECFEM uses GPU also for operations related to ASKI output, e. g. Fourier transform of the displacement fields.
- `ASKI_type_inversion_grid`: type of onversion grid. Only implemented value is 4 for `specfem3d_inversion_grid`.
- `ASKI_wx`: together with `ASKI_wy` and `ASKI_wz` these variables define a subregion of the SPECFEM box in pseudo-coordinates defining the ASKI inversion domain.
- `ASKI_rot_X`: together with `ASKI_rot_Y` and `ASKI_rot_Z`, these variables define a rotation of the ASKI inversion domain relative to the SPECFEM box.
- `ASKI_cx`: together with `ASKI_cy` and `ASKI_cz`, these variables define the center of the ASKI inversion domain relative to the SPECFEM box. The ASKI extension in SPECFEM assumes that elements of the inversion grid lie in ranges $-\frac{1}{2}w_x < x - c_x < \frac{1}{2}w_x$ and so forth. With regard to the non-symmteric depth range of SPECFEM (from 0 to $-D$) and the the fact that the Z-axis points upwards, `ASKI_cz` should be negative such that $-\frac{1}{2}w_z < z - c_z < \frac{1}{2}w_z$ describes the extent of the inversion domain properly.

There are further parameters of ASKI relevant for SPECFEM which appear in the iteration parameter file. Thy are described in the next subsection.

Action 5.4: Main parameter file update

Revisit the main parameter file and set or update all values not already set right in the beginning.

5.3 The iteration parameter file

Once the ASKI main parameter file has been set up, we may create iteration step folders and subfolders with the script `createIterationFolders.py` and an initial version of the `iter_parfile` for each iteration. This file will be placed into the iteration step folders. The script gets as input the name of the main parameter file and the intended number of iterations. The following table specifies the variables set in the iteration parameter file:

ASKI iteration parameter file	
Variable	Recomm. Value
`ITERATION_STEP_NUMBER_OF_FREQ`	4
`ITERATION_STEP_INDEX_OF_FREQ`	1 2 3 4
`VTK_GEOMETRY_TYPE`	CELLS

VTK_COORDS_SCALING_FACTOR	1.0
NPROC	40
MAX_NUM_CG_ITERATIONS	10000
NITER_WINDOW_STA	20
NITER_WINDOW_LTA	200
VSCAL_SMOOTHING	1.0 1.0 1.0
VSCAL_DAMPING	1.0 1.0 1.0
DT	0.14
NSTEP	1000
GREEN_TENSOR_COMPONENT	UP
ASKI_MAIN_FILE_ONLY	.false.
ASKI_MAIN_FILE_WRITE	.true.
ASKI_DFT_method	EXPLICIT_SUMMATION
ASKI_DFT_double	.false.
ASKI_DFT_apply_taper	.true.
ASKI_DFT_taper_percentage	0.05

The parameters have the following meaning:

- ITERATION_STEP_NUMBER_OF_FREQ allows to set the number of frequencies used in the inversion to deviate from what is specified in the main parameter file. This way, the user may start the inversion with a few low frequencies and then add higher frequencies with iterations.
- ITERATION_STEP_INDEX_OF_FREQ allows to explicitly set the frequencies to be used. The chosen frequencies must however be a subset of the ones specified in the main parameter file.
- VTK_GEOMETRY_TYPE: either CELLS indicating data on inversion grids will be written on the volumetric inversion grid cells or CELL_CENTERS, indicating data on inversion grids will be written on the cell center points.
- VTK_COORDS_SCALING_FACTOR: this may be helpful if coordinate values (e.g. in m) get so large that they cause problems in paraview.
- DT: Time step in seconds for SPECFEM simulations. Look into output_generate_databases.txt for hints how to choose this value.
- NSTEP: Number of time steps for Specfem computations. Only relevant for kernel Green tensor simulations. For injection, step number is taken from sequential source description.
- GREEN_TENSOR_COMPONENT: Component of the Green tensor (= direction of single force at receiver position). Identical to data component to be used for inversion later. Give here only one; if further components are desired, do additional SPECFEM runs.
- ASKI_MAIN_FILE_ONLY: Tells if only ASKI main file is produced to check inversion grid and background model and then abort the simulation.
- ASKI_MAIN_FILE_WRITE: Tell if ASKI main file is written for first source.
- ASKI_DFT_method: Choose the method of Discrete Fourier Transform: EXPLICIT_SUMMATION means on-the-fly summation of complex values $s(t) * exp^(- i * 2\pi f t)$. the only method implemented for GPU.

- `ASKI_DFT_double`: Precision of Discrete Fourier Transform. For use with GPU, single precision is required.
- `ASKI_DFT_apply_taper` and `ASKI_DFT_taper_percentage`: Decide whether the time series should be tapered by a hanning taper (on tail) before (i.e. on-the-fly while) applying the discrete fourier transform. The value of taper percentage (between 0.0 and 1.0) defines the amount of total time for which the hanning taper will be applied at the tail of the time series.

The missing parameters `NPROC`, `MAX_NUM_CG_ITERATIONS`, `NITER_WINDOW_STA` and `NITER_WINDOW_LTA`, `VSCAL_SMOOTHING` and `VSCAL_DAMPING` are explained later in the section on solving the kernel equation system (6.9).

Action 5.5: Iteration step parameter file update

Visit the iteration parameter file of the current iteration and set values to all parameters listed above..

5.4 ASKI's parameter file for SPECFEM

The ASKI extensions of SPECFEM are managed through the parameter file `Par_file_ASKI`. Once the main and the iteration step parameter file are filled with proper values, `Par_file_ASKI` can be filled completely using the Python script `prepare_specfem_kernel.py`. Most of the parameters have been explained already and the few remaining ones will be described later in section 6.

5.5 Setting up an initial 3D ASKI inverted model based on FMTOMO

For this purpose, you may use the ASKI program `convertFmtomoToKim` which can read 3D models from FMTOMO and produce values for non-inverted properties using predefined model property correlations. The program expects the following input on the command line: the name of the ASKI main parameter file, the name of the file containing the 3D model, and a flag specifying if VTK output is produced. For running SPECFEM, add the name of the output text file to ASKI's parameter file for SPECFEM (variable `FILE_ASKI_INVERTED_MODEL`). Put the file either directly into `DATA` or establish a symbolic link.

5.6 Computing kernel displacements and kernel Green tensors

For kernel computation, forward computations from the sources to the GLL points of the SPECFEM grid and backward computations from the receiver locations to the GLL points are required. This can be done most conveniently using SPECFEM's multiple sequential sources capability by which SPECFEM can do simulations for many different sources (including injected wavefields) in one run. Most of the tasks described in the following two subsections are done by the Python script `prepare_specfem_kernel.py`.

5.6.1 SPECFEM's multiple sequential sources capability

For efficieny reasons, SPECFEM was modified to allow wavefield computations for several sources in a single run. This functionality is controlled by the `MULTIPLE_SEQUENTIAL_SOURCES` flag and the `SEQUENTIAL_SOURCES_DESCRIPTION_FILE` which are both specified in SPECFEM's parameter file. SPECFEM knows 3 modes of sequential sources: moment tensor sources (mode 1), single force sources (mode 2) and injected wavefields (mode 3). In any case, each source is described by a single line in the description file. For moment tensor sources each line contains the following entries:

- `t_shift`: must be set to zero, otherwise the SPECFEM solver does not run.
- `hdur`: half duration of the source function assumed to be of Gaussian shape for point sources. Recommended value is $5\Delta t$.
- x, y, z: SPECFEM Cartesian coordinates X, Y and Z of the hypocenter
- M_{rr}, $M_{\vartheta\vartheta}$, $M_{\phi\phi}$, $M_{r\vartheta}$, $M_{r\phi}$, $M_{\vartheta\phi}$: moment tensor components in Nm. They are transformed to box cartesian components by SPECFEM.
- `ASKI_output_ID`: the identification string of the considered earthquake
- `ASKI_outfile`: part of the path to the synthetic seismograms and also to the calculated displacement fields

For single force sources each line contains the following entries:

- `t_shift`: must be set to zero, otherwise the SPECFEM solver does not run.
- `hdur`: half duration of the source function assumed to be of Gaussian shape for point sources. Recommended value is $5\Delta t$.
- x, y, z: SPECFEM Cartesian coordinates X, Y and Z of the hypocenter.
- f_x, f_y, f_z: single force in N in SPECFEM box Cartesian components. With ASKI, the force direction is the desired receiver component of the sensitivity kernels.
- `ASKI_output_ID`: the identification string of the considered source. With ASKI, the stations locations serve as positions of the source, so the identification string contains network code and station name.
- `ASKI_outfile`: part of the path to the synthetic seismograms and also to the calculated kernel Green tensor fields.

For injection sources each line contains the following entries:

- number of time steps of the SPECFEM simulation,
- the file name of the injected wavefield on the boundary points of the SPECFEM box,
- `ASKI_output_ID`: the identification string of the considered source,
- `ASKI_outfile`: part of the path to the synthetic seismograms and also to the calculated kernel displacement fields.

5.6.2 Length of time step and simulation in SPECFEM

The length of the SPECFEM simulation is controlled by the parameters `NSTEP` and `DT`. While `DT` should be chosen according to the recommendation given in the SPECFEM output file `output_generate_databases.txt` in the folder `OUTPUT_FILES`, the number of

steps and thus the length of the simulation depends on whether kernel displacements or kernel Green tensors are computed. For kernel displacements, the simulation length should be chosen equal to the length of the injected seismograms as described in section 3.6. It is calculated by the Python script.

For kernel Green tensors (i. e. single force Green functions for sources at the station locations), the simulation length can be restricted according to the following considerations: Consider the path of a scattered wave from the box boundary to some scatterer inside the box (taking time t_d) and from there (as either P or S) to the receiver (taking time t_b which is equal to the time the back-propagated wave needs from receiver to scatterer). To get all scattered signals that arrive within the phase window of length w, the total time allowed for the scattered wave $t_b + t_d$ should be not be less than the time of the direct wave from the box boundary to the reciver (t_r) plus the phase window length, $t_b + t_d \geq t_r + w$. Thus, for the back-propagated wave, we find $t_b \geq w + t_r - t_d$. The right hand side assumes its maximum for a scatterer on the box boundary with $t_d = 0$. Thus, a lower limit for t_b is $t_r + w$. The direct travel time to the receiver can be estimated as the depth of the box divided by average P-wave velocity, H/v_p, leading to $t_b \approx H/v_p + w$.

5.6.3 The Python script

This script `prepare_specfem_kernel.py` does all the preparations required for computing kernel displacements and kernel Green tensors. It modifies the SPECFEM parameter file and creates `Par_file_ASKI` based on information from the main and iteration parameter files. It also creates an appropriate list of source descriptions which is read and worked through by SPECFEM. The script either works through the event list (for kernel displacement) or the station list (for kernel Green tensors). It checks the existence of the paths specified in the main and the iteration parameter files. For back propagation from the receiver locations, depending on the choice of Green tensor component, proper values for the force components are calculated by the script including a component transformation. The script is located in the Python folder of ASKI. As command line arguments, the script expects the following informations:

- the name of the main parameter file,
- `--disp`: flag indicating that kernel displacement fields are computed,
- `--gt`: flag indicating that kernel Green tensor fields are computed
- `--no_gpu`: flag indicating that ASKI extensions should not use the GPU.

After running the script, you may want to go through the `Par_file_ASKI` and check if all parameters were set as expected. As already mentioned, the script also modifies the SPECFEM `Par_file`. You do not need to care about the variables marked by "auto" in the table describing this file. But walk through the other variables and give them reasonable values following the recommendations in the table. Now you are ready to start the SPECFEM computations by running xspecfem3D.

5.6.4 Results produced by the SPECFEM run

After having run SPECFEM successfully, you should have received the following output in the current iteration step folder:

- the ASKI main file in folder `aski_main_files`,

Action 5.6: Kernel displacement and Green tensor wavefields

Run the script `prepare_specfem_kernel.py`. Provide the name of the parameter file and either the option `--disp` or `--gt` on the command line. Either a file `kernel_disp_specfem_injection` or `kernel_gt_specfem_forces` is created in `DATA` that provides the required source information to SPECFEM. Also go through the parameter files `Par_file_ASKI` and `Par_file` and check if the changes made by the script are correct. Then run SPECFEM most conveniently by using the script `run_executable_on_sge.py`. If any errors happen, files with names `error_messages_*.txt` may be written to `OUTPUT_FILES`. Remove these files before running SPECFEM again. You can follow the progress of the run by watching the file `output_solver.txt` in `OUTPUT_FILES`.The run `xspecfem3D` with the number of processes defined when preparing the SPECFEM databases.

- in case kernel displacements were computed: for each event, the Fourier transformed wavefields from the source (or injected) in folder `kernel_displacements` for each selected frequency plus a subfolder with SPECFEM synthetic seismograms at the stations. Specfem log files in `OUTPUT_FILES` should be copied manually to a `specfem_logs` subfolder if desired.
- in case kernel Green tensors were calculated: for each station and selected component the Fourier transformed Green tensor wavefields emanating from the receiver locations in `kernel_green_tensors` for each selected frequency plus a subfolder with synthetic seismograms . These seismograms are not needed and are reduced to one single station by the `prepare_specfem_kernel.py`-script for efficiency reasons. Specfem log files in `OUTPUT_FILES` should be copied manually to a `specfem_logs` subfolder if desired.

5.6.5 Visualization of kernel displacement and kernel Green tensor fields

The executables `kdisp2vtk` and `kgt2vtk` allow a visualization of the frequency-domain kernel displacement and kernel Green tensor wavefields, respectively, computed by SPECFEM. They are managed via the main parameter file and further command line arguments. For `kdispl2vtk` the command line arguments are:

- the name of the main parfile,
- `-evid`: the identification string of the kernel displacement object, here the event id,
- `-ifreq`: the frequency index at which the wavefield output should be extracted,
- `-ucomp`: one of the strings “ux”, “uy”, “uz”, “exx”, “eyy”, “ezz”, “eyz”, “exz”, “exy”, denoting either displacement or strain components,
- `-norm`: flag if displacement fields are normalised to 1.0.

For `kgt2vtk` the command line arguments are:

- the name of the main parfile,
- `-netsta`: the identification string of the kernel Green tensor object, here “network.station”,
- `-fcomp`: force component of the kernel Green tensor object (e. g. “UP”),
- `-ifreq`: the frequency index at which the wavefield output should be extracted,

- `-ucomp`: one of the strings “ux”, “uy”, “uz”, “exx”, “eyy”, “ezz”, “eyz”, “exz”, “exy”, denoting either displacement or strain components,
- `-norm`: flag if displacement fields are normalised to 1.0.

Both programs produce VTK files which can be rendered using `paraview`.

6 Performing sensitivity kernel based FWI with ASKI

Once the kernel displacement fields and the kernel Green tensor fields are computed using SPECFEM, ASKI programs may be used to perform a full waveform inversion step. Among these are executables for computing sensitivity kernels, for producing Fourier transformed data samples, for producing Fourier transformed data predictions, for solving the kernel linear system and finally exporting the newly found earth model.

6.1 Data and model space

A very important ingredient to ASKI full waveform inversion is the definition of data and model space to be used in the inversion step. A single datum in ASKI full waveform inversion is characterized by a **path**, defined by an event-station pair, a **weight**, its receiver **component** and the **index** defining the Fourier frequency. Moreover, each datum is a complex number consisting of **real** and **imaginary part**. For each path, the frequencies and components used in the inversion may be chosen individually. Thus, the number of data per path is two times the number of components times the number of frequencies (2*nf*ncomp). The current implementation of the data space allows to assign weights either by frequency or by path or by path and frequency. Component-wise weighting is not implemented.

A single model component is defined by its material **property** (e. g. density, P- and S-wave velocity), and its cell in the inversion grid. Thus, the number of model components is the number of properties to be inverted for times the number of inversion grid cells (nprop*ncell). Association of a certain material property to selected cells is not implemented. Each listed property is assumed to be defined on the full inversion grid.

Data and model space are defined together in a single file, the “data-model-space”-file. It is structured in two basic blocks, one for the data and one for the model space. The blocks can be preceded and followed by arbitrary comment lines. The model values block looks as follows:

- `MODEL_VALUES`, starting the model values block within which no comment lines are allowed,
- a line of the form `nprop prop_1 prop_2 ... prop_n` specifiying the property set used for all inversion grid cells,

The data samples block is a bit more elaborated:

- `DATA_SAMPLES`, starting the data samples block within which no comment lines are allowed;
- `WEIGHTING value` where `value` is either `NONE`, `BY_FREQUENCY`, `BY_PATH` or `BY_PATH_AND_FREQUENCY`; the latter two are only allowed if `PATHS SPECIFIC` is chosen in the next line;

- 3rd line: `PATHS value` where `value` is either `ALL` or `SPECIFIC`.

What follows depends on what is chosen for `PATHS`. In case of `ALL`

- a line with `nev ev_1 ev_2 ... ev_n` defining the events to be used;
- a line with `nstat stat_1 stat_2 ... stat_n` defining the stations to be used, the full set of paths is built from all combinations of events and stations;
- a line `COMPONENTS ALL` (same components for all paths);
- a line of the form `ncomp comp_1 comp_2 ... comp_n` defining the components for all paths
- a line `FREQUENCIES ALL` (same set of frequencies for all paths);
- a line of the form `nfreq ifreq_1 ifreq_2 ... ifreq_n` defining the frequency indices for all paths;
- a line of the form `nfreq w_1 w_2 ... w_n` defining `nfreq` weights for each frequency IF `WEIGHTING BY_FREQUENCY` was chosen.

If `PATHS SPECIFIC` is chosen in line 3, then much more details need to be specified. First, in the following 3 lines you need to set `COMPONENTS`, `FREQUENCIES` to either `ALL` or `SPECIFIC`. If you choose `ALL` for any of the three, you need to provide lines of the form, e.g., `nfreq freq_1 freq_2 ... freq_n` directly below. For the keyword `FREQUENCIES` you must add a further line with weights if weighting by frequency or weighting by path and frequency was chosen. If you choose `SPECIFIC` for any of the three, no further specifications directly follow that line.

After having chosen values for these three keywords and having added required lines, the next line gives the number of paths `npath` followed by `npath` blocks of lines specifying the data samples. Each block may contain:

- `event_id network_staname [w]` defining the path and optionally a weight if `WEIGHTING BY_PATH` or `WEIGHTING BY_PATH_AND_FREQUENCY` was chosen; in the latter case the weights for path and frequency are multiplied(!),
- if `COMPONENTS SPECIFIC`, add a line `ncomp comp_1 comp_2 ... comp_n` for this specific path;
- if `FREQUENCIES SPECIFIC`, add a line `nfreq freq_1 freq_2 ... freq_n` for this specific path;
- add a line with weights `nfreq w_1 w_2 ... w_n` if weighting by frequency or by path and frequency was chosen, the final weight is then the product of path and frequency weight.

The `SPECIFIC` options seem to be complicated but by using them you have full control on the data samples and weights used by the inversion. There are 3 examples of data-model-space files in ASKI's template folder.

Action 6.1: Data-model-space file

Run the Python script `createDmspaceFile.py` which helps in setting up the data-model-space file for large data sets.

6.2 Issue with frequency domain treatment of sensitivity kernels

In the time domain, the waveform sensitivity kernels with respect to density and seismic velocities are calculated as a convolution of the incoming wavefield and its derivatives and the Green tensor components and their derivatives backpropagated from the station locations. In the Fourier domain, this converts to a multiplication of the Fourier transforms of these quantities. This is however only fully true for infinitely long time series. What hapens if only time windows are considered in the inversion?

Consider a given station and an earthquake for which we have an observed and a synthetic seismogram. In the inversion, we consider only a time window starting at time S and ending at time E. In our case, synthetic and kernel seismograms start at a common time which we take as the origin of our time axis. Deviating start times of the observed seismograms are corrected for by applying a phase shift to their Fourier transforms. In the time domain, we can write down the following equation which we will want to satisfy in a least squares sense later in the inversion:

$$d(\tau) - s(\tau) = \sum_j k_j(\tau)\, m_j\,, \tag{11}$$

where $d(\tau)$ is the observed seismogram, $s(\tau)$ the synthetic one, $k_j(\tau)$ the kernel seismogram for model parameter m_j. When taking the Fourier transform, we integrate all of them from start to end time of data and synthetics, for example:

$$K_j(\omega) = \int_0^L k_j(\tau)\, e^{-i\omega\tau} d\tau\,, \tag{12}$$

where $L = E - S$ is the length of the kernel seismogram as well as that of observed and synthetic seismogram. The kernel seismogram is the convolution of variants of the incoming wavefield u_j and variants of the Green tensor field g_j at model location j:

$$k_j(\tau) = \int_0^{L_u} u_j(t) g_j(\tau - t) dt\,, \tag{13}$$

where L_u is the length of incoming wavefield (also starting at origin time S). Inserting this into the equation above, we obtain

$$K_j(\omega) = \int_0^L d\tau \int_0^{L_u} dt\, u_j(t) g_j(\tau - t) e^{-i\omega\tau}\,. \tag{14}$$

First we interchange the t and the τ integration:

$$K_j(\omega) = \int_0^{L_u} dt\, u_j(t) \int_0^L d\tau\, g_j(\tau - t) e^{-i\omega\tau}\,. \tag{15}$$

Then, we apply the substitution $s = \tau - t$. Considering t fixed within the τ-integration, its lower limit changes to $-t$ and its upper limit to $L - t$:

$$K_j(\omega) = \int_0^{L_u} dt\, u_j(t) \int_{-t}^{L-t} ds\, g_j(s) e^{-i\omega(s+t)} = \int_0^{L_u} dt\, u_j(t) e^{-i\omega t} \int_0^{L-t} ds\, g_j(s) e^{-i\omega s}\,, \tag{16}$$

where we changed the lower limit to 0 because the Green tensor seismograms vanish for negative times. The first integral is the Fourier transform of u_j but the second one may be different from the Fourier transform of g_j depending on the upper limit. The upper limit decreases from L to $L - L_u$. Hence, only if $L - L_u \geq L_g$, the length of g_j, then it is identical to the Fourier transform.

A simple solution of the problem is to extend the considered time window or to assume that zeros have been appended to it. The former approach may include new undesired signals into the time window, the latter one forces the inversion to produce a model which does not create scattered signals where the zeros were appended which is impossible. Moreover, in our approach to FWI, the choice of L_u is event-specific but the same for all stations. Secondly, the length L_g may change with the station but is the same for all events. So, we can neither adapt L_u nor L_g to a specific path.

The following considerations may help out of this dilemma. The length of the synthetic seismogram used for Fourier transform is $L = t_P + w$ where t_P is the P-wave travel time to the receiver and w the length of the phase window. Similarly, we can define the length of the incoming wavefield $L_u = \tau_P + w_s$ where τ_P is the P-wave travel time to the scatterer and w_s the length of the scattering window which must be chosen shorter than the phase window. The idea is that the incoming wavefield consists of the main P-wave train followed by smaller secondary signals with a second-order contribution to scattering. Finally, we can write the length of the Green tensor seismogram as $L_g = \theta + w_g$ where θ is either the travel time of the P or S wave and w_g is the length of the phase. With these new definitions, we can rewrite the requirement $L > L_u + L_g$ as:

$$t_P + w > \tau_P + w_s + \theta + \Omega\,. \tag{17}$$

The sum of τ_P and θ is the travel time of the scattered wave denoted by τ_{sc}. Using this, we can write

$$\tau_{sc} - t_P < w - (w_s + \Omega)\,. \tag{18}$$

For example, with $w = 120\,s$, $w_s = 60\,s$ and $\Omega = 15\,s$ we obtain 45 s of effective scattering time. Since the P-wave is faster than the S-wave, PP-scattering arrives within the phase window for a much larger region of scatterers than does PS-scattering. If we restricted the scattering region to that of PS-scattering we would miss a large part of the PP-scatterers. A way out is to compute Green tensor spectra for waveforms containing only P and additionally for waveforms containing P and S. Since the scattering region of P to S is not that big (depending on the effective scattering time) we need not store the latter on all scattering points (saving about 3/4 of disk space). Having available both P- and P-plus-S-Green tensor spectra, we can apply the latter when computing kernels if P-to-S-scattering time is less that the effective one and the former if only P-to-P scattering time is less. If both are bigger than the effective scattering time, the kernels are tapered to zero.

Since kernels for P-wave velocity only depend on the divergence of the incoming and back-propagated wavefield which vanishes for S-waves, PS-scattering need not be considered at all.

The phase and scattering windows may vary with the events but it should be possible to keep the difference constant to ensure a constant effective scattering time. According to this time, the region where P-plus-S Green tensor spectra are stored can be determined. You may use the script `plotScatteringTimeField.py` to visualize the effective scattering regions. In case the true effective scattering time of an event is less than the assumed maximum one, the scattering region can be adjusted during kernel computation.

Moreover, the number of time steps in SPECFEM should be sufficiently large to assure that the required Green tensor waveforms are available at the scattering points. These travel times can be visualized with the script `plotTravelTimeField.py`.

6.3 Timimg of data, SPECFEM synthetics, injected and back-propagated wavefield

We define here the data $d(t)$, the synthetic seismogram $s(t)$ and the injected seismogram at some point in the box, $u(t)$. All are assumed to start at source time. The back-propagated field is denoted by $g(t)$ starting at the begin of the source wavelet. Note that SPECFEM assigns a negative value to this point while SPCEFEM's zero time corresponds to the maximum of the source wavelet. Since we do not want to lose the first half of the source wavelet, we always start at the first sample and associate zero time with it.

When propagating the injected wavefield using SPECFEM, the first sample corresponds to the time when the teleseismic wave first reaches the computational box (called t_0). When computing the Fourier transform of displacement at some point inside the box we integrate up to the scattering length $\tau_P + w_s$:

$$U(\omega) = \int_{t_0}^{\tau_p+w_s} u(t)e^{-i\omega(t)}dt = \int_{0}^{\tau_P-t_0+w_s} u(t'+t_0)e^{-i\omega(t'+t_0)}dt' \,. \tag{19}$$

Here, $t' = t - t_0$ is the time in the numerical simulation and $u(t'+t_0)$ is the value of displacement at that time, say $\tilde{u}(t')$. To get closer to what is actually computed we may also write:

$$U(\omega) = e^{-i\omega t_0} \int_{0}^{\tau_P-t_0+w_s} \tilde{u}(t')e^{-i\omega t'}dt' = e^{-i\omega t_0}\,\tilde{U}(\omega)\,. \tag{20}$$

Since the SPECFEM synthetic seismograms are obtained during the kernel displacement simulations, they also start at t_{red} and we choose $t_P + w$ as upper limit. Hence, their Fourier transform is

$$S(\omega) = \int_{t_0}^{t_P+w} s(t)e^{-i\omega t}dt = \int_{0}^{t_P-t_0+w} s(t'+t_0)e^{-i\omega(t'+t_0)}dt' \,. \tag{21}$$

Here, $t' = t - t_0$ is again the time given to the sample in the numerical simulation and $s(t'+t_0)$ is the value of displacement at that time, say $\tilde{s}(t')$. To get closer to what is actually computed we may also write:

$$S(\omega) = e^{-i\omega t_0} \int_{0}^{t_P-t_0+w} \tilde{s}(t')e^{-i\omega t'}dt' = e^{-i\omega t_0}\,\tilde{S}(\omega)\,. \tag{22}$$

The Green tensor simulations effectively provide seismograms starting at time zero because ASKI does a deconvolution with the source wavelet used by SPECFEM. Let us denote these

seimograms by $g(t)$. Depending on whether only PP or in addition also PS-scattering is desired, we choose a certain value $\theta + w_g$ up to which the integration is performed:

$$G(\omega) = \int_0^{\theta+w_g} g(t)e^{-i\omega t}dt\,. \tag{23}$$

It remains to consider the measured seismogram. It starts individually at theoretical travel time minus buffer length ($t_a = t_P - b$) and ends at travel time plus phase window length ($t_P + w$)). The Fourier transform is then:

$$D(\omega) = \int_{t_P-b}^{t_P+w} d(t)e^{-i\omega t}dt = \int_0^{w+b} d(t' + t_P - b)e^{-i\omega(t'+t_P-b)}dt = \int_0^{w+b} d(t' + t_a)e^{-i\omega(t'+t_a)}dt\,. \tag{24}$$

Here, $t' = t - t_a$ is again the time given to the data sample and $d(t' + t_a)$ is its value at that time, say $\tilde{d}(t')$. To get closer to what is actually computed we may also write:

$$D(\omega) = e^{-i\omega(t_a)} \int_0^{w+b} \tilde{d}(t')e^{-i\omega t'}dt' = e^{-i\omega(t_a)}\,\tilde{D}(\omega)\,. \tag{25}$$

In the inversion, we set up equations of the form:

$$D(\omega) - S(\omega) = \sum_j U_j(\omega)G_j(\omega)m_j\,. \tag{26}$$

Expressing these in terms of the tilded quantities which are actually computed we get:

$$e^{-i\omega(t_a)}\tilde{D}(\omega) - e^{-i\omega t_0}\tilde{S}(\omega) = \sum_j e^{-i\omega t_0}\tilde{U}_j(\omega)G_j(\omega)m_j\,. \tag{27}$$

Multiplying with $\exp(+i\omega t_0)$, we obtain:

$$e^{-i\omega(t_a-t_0)}\tilde{D}(\omega) - \tilde{S}(\omega) = \sum_j \tilde{U}_j(\omega)G_j(\omega)m_j\,. \tag{28}$$

Thus, it is sufficient to compute the tilded Fourier transforms but we must multiply the tilded transform of the data with the phase factor $\exp(-i\omega(t_a - t_0))$.

6.4 Running initBasics and writeVtkFiles

The first step to using ASKI for full waveform inversion is to run the parallel ASKI executable `initBasics`. You may do this on one process or on the same number of processes used for the SPECFEM runs. `initBasics` checks the which first checks if all parameters needed are present in the parameter files and then creates all basic requirements for ASKI operations. It takes the ASKI main parameter file as mandatory argument. It reads required files like event list and station list files, the wavefield points and the kernel reference model. Furthermore, it creates the inversion grid, localizes the wavefield points inside it and computes the integration weights.

Using the executable `writeVtkFiles`, plenty of .vtk files with statistics are produced. It takes the ASKI main parameter file as mandatory argument and allows the flag `-nowp` indicating not to produce VTK files for quantities defined on the SPECFEM wavefield points. More specifically, `writeVtkFiles` produces VTK files for the inversion grid, the sum of integration weights per cell, the cell volume, density, P- and S-wave velocity on the inversion grid, the wavefield points and density, P- and S-wave velocity on the wavefield points.

6.5 Fourier transforming of observed seismograms

The Fourier transform of the observed seismograms is done by the executable program `transformMeasuredDataAsciiSeis`. The Fourier transform is applied to observed seismograms that were preprocessed during moment rate function estimation and written to an event subfolder by `computeSourceWavelet`. Processing included detrending, tapering, bandpass filtering using the same filter parameters as for the Gemini synthetics. The executable makes use of the data-model-space block written during moment rate function estimation in which all available station for the given event are listed. It reads the ASKI main parameter file from where it takes number and values of frequencies at which the Fourirer transform is calculated. It also reads values for `FILE_EVENT_LIST` and `PATH_MEASURED_SEIS` from the main parameter file. Output files are written to the folder `PATH_MEASURED_DATA` set in the ASKI main parameter file. Different start times of measured, synthetic and kernel seismograms are corrected by an appropriate phase shift of the Fourier spectral values. Please consult the section on timing below. Note that not all available seismograms are transformed but only the ones available for the events listed in the event file and the stations listed in the station file. The output is written to a single HDF file for each considered component. The HDF datasets are accessible by the path name.

For sensitivity kernel computation, this program computes and writes to HDF the travel time `tt` for each observation together with the reduction time `tred`, the phase window length `twinlen` and the end of the phase window given by `tt`twinlen-tred+. This output is necessary to calculate the residual scattering time during kernel computation.

The command-line arguments are:

- the name of the main parfile,
- `-comp`: a component name specifying the seismogram component.
- `-timing`: write timing of observed seismograms to HDF file (set this option!)

6.6 Computing sensitivity kernels

Sensitivity kernels are computed using the parallel program `computeKernelsDmspaceParallel`. The kernels are computed by multiplication of Fourier transformed Green tensor component wavefields and forward wavefields for a given event-station pair (called “path” below). By default, the kernels are integrated over the inversion grid cells using the SPECFEm integration weights, but it is also possible to calculate the kernels on the wavefield points.

The code is parallelized and shares the displacement fields on the wavefield points among the processes in the same way as SPECFEM does. For this reason, it is mandatory to use the same number of processes as used for the SPECFEM runs. This requirement is checked by the code. The code loops over the Fourier frequencies specified in the iteration parameter file and opens a HDF file to which sensitivity kernels for all paths and components are written.

Then it loops over tha paths listed in the data-model-space file and computes the kernels for all specified components and material properties. In the HDF file, the kernels are stored as 3-dimension arrays with the dimensions grid cells, properties and components. The individual HDF-datasets can be accessed by their path name. In case, kernels for a specific path, property or component are desired, a separate data-model-space file should be set up.

The command-line arguments are:

- the name of the main parfile,
- `-wp`: a flag indicating if the kernels are computed on the wavefield points only.
- `-ifreq`: a frequency index at which the kernel should be calculated. Default is all frequency indices.

For plotting a kernel may be thrown onto a 3D regular spherical grid using the program `throwKernelOnVgrid`. With the program `computeRegularsphericaGridSlices` and by setting up a text file specifying the slices, various horizontal or vertical slices can be extracted from the 3D grid and plotted using `plotSliceRegularSphericalGrid`.

VTK-files for sensitivity kernels may be produced using the executable `kernel2vtk`. It accepts the following command line arguments:

- the name of the main parfile,
- `-evid`: an event identification string specifiying the event-part of the path,
- `-staname`: a network-station name specifying the station component of the path,
- `-comp`: a vector specifying receiver components,
- `-prop`: a vector specifying property names,
- `-ifreq`: a vector specifying frequency indices, or
- `-all_ifreq`: indicating to walk through all frequencies listed in the ietration parameter file,
- `-wp`: a flag indicating that kernels on the wavefield points are plotted and expected to exist,
- `-norm`: a flag indicating if kernels are normalized to 1.

Please note, that the output VTK files (one for every frequency) might get large, dependent on the resolution of the inversion grid, since the geometry information of the inversion grid cells is contained in each VTK file. If the files become too large for you, you might consider setting `DEFAULT_VTK_FILE_FORMAT = BINARY` in your main parameter file.

Alternatively, kernels may be plotted by first throwing them onto a 3D regular spherical grid using `throwKernelOnVgrid`, then extract horizontal or vertical slices using `computeRegularSphericalGridSlices` and plot them using `plotSliceRegularSphericalGrid.py`. Slices are defined in a text file whose format is described in the `computeRegularSphericalGridSlices`. This file is read by both the Fortran and the Python code.

6.7 Fourier transforming of synthetic SPECFEM seismograms

The Fourier transform of the synthetic SPECFEM seismograms is done event-wise by the executable program `transformSpecfemSynthetics`. It transforms the synthetic seismograms written by SPECFEM after running a kernel displacement forward simulation, again

assuming that the first sample corresponds to zero time. Comparison with GEMINI synthetics has shown that the small negative time at which SPECFEM synthetics start should be ignored. The first sample of the seismograms corrsponds to source time plus reduction time which serves as origin for the time axis in the inversion. The program reads the ASKI main parameter file from which it takes the number and values of frequencies where the Fourier transform is evaluated. It also takes the geographical coordinates of the SPECFEM box center (identical to the center of the inversion grid) from there. These are needed to transform from SPECFEM cartesian to ZNE components. The location of the SPECFEM synthetics is guessed from the ASKI parameter `PATH_KERNEL_DISPLACEMENTS`. The file name is constructed from netcode, station name, band code, component and seismogram type. The latter two are command line arguments. After Fourier transform a rotation to ZNE components is done and results are written as a single HDF file to the ASKI folder `PATH_SYNTHETIC_DATA` following the file name convention `syn_comp.hdf`. Individual spectra can be accessed by their pathname.

The command-line arguments are:

- the name of the main parfile,
- `-comp`: a component name specifying the seismogram component,
- `-sem`: the SPECFEM synthetics file extension to be used (semd,semv,sema).

6.8 Decomposing the data space

To simplify the parallelization of the kernel equation system, the program `decomposeDmspace` is used to split the data space into equal path portions which are later shared among the processes. It outputs a set of data-model-space files containing these paths. The number of processes is defined by the parameter `NPROC` of the iteration parameter file.

The command-line arguments are:

- the name of the main parfile,
- the name of the data-model-space file.

The output filenames are formed from the name of the input data-model-space file extended by the rank of the process.

6.9 Solve the kernel system

This is the decisive step in which the current model is updated by solving the linear kernel equation system. We use the executable `solveCGLSKernelSystem` which uses the conjugate gradient approach to solve the equation system. This is the fastest way to obtain a solution. The conjugate gradient allgorithm is parallelized, simply sharing the rows of the kernel system matrix among the processes. The number of processes is defined by the parameter `NPROC` of the iteration parameter file.

The program first gets all the relevant paths and file names from the main and iteration parameter files. Then, it reads the set of data-model-space files which define the data and a such also the rows of the kernel system matrix associated with the different processes. Based on this information, it reads the measured data, the synthetic data and fills the kernels matrix by reading the required sensitivity kernels. Finally, it computes the difference between data and synthetics as right hand side of the kernel equation system. Based on information on the model space

and the inversion grid, it sets up the regularization equations for damping and smoothing in an efficient and also parallelized way by only dealing with the non-zero entries and sharing matrix rows among the processes. Then, the CGLS iteration is started until the max number of iteration is reached or the convergence criterion is satisfied. After that, the solution is converted to a so-called "kernel inverted model (kim)" and written as HDF to `PATH_OUTPUT_FILES`. VTK files of absolute and relative perturbations are also produced. The "kim"-file can be used with the "model-aski" feature of SPECFEM's `generate_databases` to feed back the model into SPECFEM and begin a new full waveform iteration.

The command-line arguments are:

- the name of the main parfile,
- the name of the data-model-space file,
- `-startsol`: the name of a "kim"-file containing a starting model for the inversion step,
- `-normalize`: a flag indicating a path and component-wise normalization of the data using previously determined noise levels.

6.9.1 Parameters related to the CGLS solver

`NPROC`: number of processes sharing the rows of the kernel system matrix.

`VSCAL_SMOOTHING`: specifies a global scaling factor per material property to be inverted for which is multiplied on the regularization equations for smoothing. Model component specific weighting if regularization equations is not yet implemented.

`VSCAL_DAMPING`: specifies a global scaling factor per material property to be inverted for which is multiplied on the regularization equations for damping. Model component specific weighting if regularization equations is not yet implemented

`MAX_NUM_CG_ITERATIONS`: This integer gives the upper limit of number of iterations (CG algorithm will stop then, even if convergence criterion is not yet met). This mechanism cannot be switched off! So, set to a ridiculously high value if you don't want to use it.

`NITER_WINDOW_STA, NITER_WINDOW_LTA`: These two integers define window sizes of short-term average (`sta`) and long-term average (`lta`) of the residual norm of the linear system, which are used to evaluate a termination criterion of the conjugate-gradient algorithm. The algorithm assumes a monotonically decreasing residual norm, i.e. `sta < lta`. It terminates when `sta`/`lta` equals 1 (in terms of single precision), i.e. when `sta` becomes close enough to `lta`, or when `lta` is not monotonically decreasing anymore, i.e. starts to increase. The numbers `NITER_WINDOW_STA, NITER_WINDOW_LTA` define the number of iterations over which the averrages are computed. Which values are sensible actually depends on the linear system. The termination checks based on `sta`, `lta` cannot be taken into account before iteration `NITER_WINDOW_LTA` 1+. It is required to define `NITER_WINDOW_STA < NITER_WINDOW_LTA`. A ratio of `NITER_WINDOW_LTA/NITER_WINDOW_STA = 10` was experienced to be feasible. E.g. define `NITER_WINDOW_STA = 20` and `NITER_WINDOW_LTA = 200`.

6.10 Start a new iteration

To start a new iteration, perform the following steps. Use `runAskiProgram.py` to submit most of the tasks. It will create log files and put them into the right folder. It will also copy

the text-files in Specfem's `OUTOUT_FILES` folder to the iteration specific `specfem_logs` folder.

1. change the current iteration in the main parameter file,
2. run `createIterationFolders` to create new folders and the new iteration parameter file (copied either from templates or from the previous iteration),
3. check the new iteration parameter file and edit if necessary, in particular if you want to change the frequencies,
4. delete old kernel displacements and kernel Green tensors to save disk space but keep the synthetic seismograms and the sensitivity kernels,
5. run `prepare_specfem_kernel.py --disp` to create new SPECFEM and ASKI parameter files for computation of kernel displacements; the velocity model is taken either from the previous iteration or, in case of a starting model at the beginning, from the first iteration. These models are expected in the `output_files` folder of the respective iteration.
6. run `xgenerate_databases` to install the new model in the SPECFEM databases,
7. run `xspecfem3D` to get kernel displacements and synthetic seismograms
8. run `prepare_specfem_kernel.py --gt` to create new SPECFEM and ASKI parameter files for computation of kernel Green tensors;
9. run `xspecfem3D` to get kernel Green tensors
10. While SPECFEM-GT is running,
    (a) run `transformSpecfemSynthetics` to get new synthetic spectral values for the inverted model of the previous iteration and redirect output to the iteration's `output_files` folder,
    (b) copy the (masked) dmspace-file of the previous iteration to the current one with name `ASKI_dmspace`. In case you changed the frequencies you may anyway use this copied dmspace-file to obtain the misfit for the model produced in the previous iterations.
    (c) run `computeMisfits` to obtain new misfits and also a new masked dmspace file,
    (d) rerun `computeMisfits` with the `-masked` option to obtain misifts for the cleaned data,
    (e) check misfits for individual events by running `plotMisifts.py`,
    (f) If frequencies have been changed:
        i. run `createDmspaceFile.py` to produce a new one.
        ii. run `computeMisfits` to obtain new misfits and also a new masked dmspace file,
        iii. rerun `computeMisfits` with the `-masked` option to obtain misifts for the cleaned data,
        iv. check misfits for individual events by running `plotMisifts.py`,

(g) repeat dmspace creation and misfit calculations for other frequency sets,

(h) check misfit versus norm curve using `plotMisfitVersusNorm.py`

(i) run `decomposeDmspace` to split the (masked) data model space for later use with the CGLS solver,

(j) an update of the data spectral values is not required as they are calulated once for all frequencies.

11. When SPECFEM is finished, run `computeKernelsDmspaceParallel` to calculate sensitivity kernels,

12. run `solveCglsKernelSystem` to produce a new inverted model.

13. run `updateModelPerturbations` to obtain new total model perturbations,

14. run `computeModelNorm` to get the model norm,

15. analyse misfit statistics and data fit using `plotKernelLinearSystemData.py`,

16. run `makeModelSlices.py` to visualize slices through the 3D model.

7 Source wavelet determination using ray travel time and amplitude

The ray-theoretical displacement field is given by

$$u(\mathbf{x},t) = \mathrm{Re}\left[C(\mathbf{x})F(t-T(\mathbf{x}))\right] . \tag{29}$$

Here, u stands for any displacement component, C is the complex-valued ray amplitude, T is the travel time, and $F(t)$ is the (complex-valued) analytical signal of the source wavelet. Expanding the real part, we get

$$u(\mathbf{x},t) = C^R(\mathbf{x})h(t-T(\mathbf{x})) - C^I(\mathbf{x})g(t-T(\mathbf{x}))) , \tag{30}$$

where $h(t)$ is the source wavelet and $g(t)$ its Hilbert transform. C^R and C^I denote real and imaginary part of the ray amplitude. Since, in the end we want to get Fourier coefficients of the source wavelet and since there is a simple relation between $h(t)$ and $g(t)$ in the Fourier domain, we carry out the determination of the source wavelet in the frequency domain:

$$U(\mathbf{x},\omega) = C^R(\mathbf{x})H(\omega)e^{-i\omega T(\mathbf{x})} - C^I(\mathbf{x})G(\omega)e^{-i\omega T(\mathbf{x})} , \tag{31}$$

where

$$G(\omega) = \begin{cases} -iH(\omega) & \omega > 0 \\ +iH(\omega) & \omega < 0 \end{cases} \tag{32}$$

Thus, for positive frequencies we find:

$$\begin{aligned} U(\mathbf{x},\omega) &= C^R(\mathbf{x})H(\omega)e^{-i\omega T(\mathbf{x})} - C^I(\mathbf{x})(-iH(\omega))e^{-i\omega T(\mathbf{x})} \\ &= (C^R(\mathbf{x}) + iC^I(\mathbf{x}))H(\omega)e^{-i\omega T(\mathbf{x})} \\ &= C(\mathbf{x})H(\omega)e^{-i\omega T(\mathbf{x})} . \end{aligned} \tag{33}$$

For negative frequencies, we necessarily get the conjugated-complex expression because U is the Fourier transform of a real function. Thus, it is sufficient to restrict further calculations to positive frequencies.

It is important to realize that the above expression refers to a synthetic seismogram and also source wavelet with origin of the time axis at source time. When dealing with recorded data, time is often given relative to midnight of the day the earthquake happened. Moreover, the stored recorded samples are often counted and timed with origin at the start of the record. Thus, there are three different time origins involved: source time, midnight before source time and record start time.

The idea of constructing the source wavelet is to determine values of $H(\omega)$ for selected or all FFT frequencies by matching the predicted Fourier coefficients $U(\mathbf{x}, \omega)$ to all Fourier coefficients of the recorded data available for the considered earthquake. Denoting the Fourier coefficients at station locations $\mathbf{x}_k$ relative to source time by $D_k(\omega)$ we want to minimize the following object function:

$$\chi^2 = \sum_k |D_k(\omega) - U_k(\omega)|^2 = \sum_k \left|D_k(\omega) - C_k H(\omega) e^{-i\omega T_k}\right|^2 . \tag{34}$$

We can choose a convenient time for the start of the data records from which we compute the Fourier amplitudes, and we choose the start as the source time, θ_s plus the travel time T_k minus some buffer time t_b:

$$\theta_k = \theta_s + T_k - t_b \, . \tag{35}$$

The symbol θ signifies that the time is measured relative to midnight. Moreover, the length of the data record is restricted to the phase window of the direct P-wave whose length w is individually determined from the data. The total length of the data record is then $L = w + t_b$.

Taking the data samples as is and doing a Fourier transform gives us a spectrum denoted by $\tilde{D}_k(\omega)$ which is related to the one relative to source time by

$$\tilde{D}_k(\omega) = D_k(\omega) e^{i\omega(\theta_k - \theta_s)} = D_k(\omega) e^{i\omega(T_k - t_b)} \, . \tag{36}$$

The phase factor indicates a shift of the time origin to the right (from source time to start time) being equivalent to a shift of the wavelet by the same amount to the left. Replacing $D_k(\omega)$ by $\tilde{D}_k(\omega)$ in the misfit function, we get

$$\begin{aligned}
\chi^2 &= \sum_k \left|\tilde{D}_k(\omega) e^{-i\omega(T_k - t_b)} - C_k H(\omega) e^{-i\omega T_k}\right|^2 \\
&= \sum_k \left|e^{-i\omega(T_k - t_b)} (\tilde{D}_k(\omega) - C_k H(\omega) e^{-i\omega t_b})\right|^2 \\
&= \sum_k \left|\tilde{D}_k(\omega) - C_k H(\omega) e^{-i\omega t_b}\right|^2 \\
&= \sum_k (\tilde{D}_k(\omega) - C_k H(\omega) e^{-i\omega t_b})(\tilde{D}_k(\omega)^* - C_k^* H^*(\omega) e^{i\omega t_b})
\end{aligned} \tag{37}$$

Differentiating with respect to H^* and zeroing the result, we find:

$$H(\omega) = \frac{\sum_k C_k^* \tilde{D}_k(\omega) e^{i\omega t_b}}{\sum_k |C_k|^2} \, . \tag{38}$$

Back transforming $H(\omega)$ yields the source wavelet relative to source time. Back transforming $-iH(\omega)$ yields its Hilbert transform.

8 Kernel integration of ray-theoretical wavefields

To compute sensitivities for model perturbations in an inversion cell, ASKI uses displacements fields from the earthquake source and Green displacements fields emanating from the receiver position that are both available on the wavefield points. ASKI first calculates sensitivity kernels on the wavefield points and then integrates these over the inversion cell. We denote displacements from the source by a superscript s, $\mathbf{u}^s$ for "source" and displacements from the receiver with a superscript r, $\mathbf{u}^r$. We add superscripts in the same way to expressions for strain. The kernels on the wavefield points are given by the following expressions in Cartesian components:

$$\begin{aligned} k_\lambda^{sr} &= -(e_{11}^r + e_{22}^r + e_{33}^r)(e_{11}^s + e_{22}^s + e_{33}^s) \\ k_\mu^{sr} &= -2(e_{11}^r e_{11}^s + e_{22}^r e_{22}^s + e_{33}^r e_{33}^s) - 4(e_{23}^r e_{23}^s + e_{13}^r e_{13}^s + e_{12}^r e_{12}^s) \\ k_\rho^{sr} &= \omega^2 u_j^s u_j^r \,, \end{aligned} \tag{39}$$

where the summation convention applies in the density kernel and

$$e_{ij}^{s,r} = \frac{1}{2}(\partial_j u_i^{s,r} + \partial_i u_j^{s,r}) \,. \tag{40}$$

In ray theory, displacement components are given in the form $A_j e^{iwT}$ where both the amplitude A_j and the travel time T are functions of position and available at the grid points of the propagation grid. For convenience, we define an inversion cell as follows: its center coincides with a grid point and the cell extension along each spatial direction is one grid spacing. Thus, there is only one wavefield point per inversion cell which is identical to the cell center. We assume that the amplitude is constant within the cell and that travel time varies linearly:

$$u_j^{s,r} = A_j^{s,r} e^{i\omega(T_0^{s,r} + \mathbf{p}^{s,r}\cdot(\mathbf{r}-\mathbf{r}_0))} \,. \tag{41}$$

Here, $\mathbf{r}_0$ is the position of the cell center and T_0 the travel time there. A product of the two displacements has then the form:

$$u_j^s u_j^r = A_j^s A_j^r e^{i\omega(T_0^s + T_0^r + (\mathbf{p}^s + \mathbf{p}^r)\cdot(\mathbf{r}-\mathbf{r}_0))} \,. \tag{42}$$

Strains are obtained as

$$e_{ij}^{s,r} = \frac{1}{2} i\omega (A_i^{s,r} p_j^{s,r} + A_j^{s,r} p_i^{s,r}) e^{i\omega(T_0^{s,r} + \mathbf{p}^{s,r}\cdot(\mathbf{r}-\mathbf{r}_0))} = E_{ij}^{s,r} e^{i\omega(T_0^{s,r} + \mathbf{p}^{s,r}\cdot(\mathbf{r}-\mathbf{r}_0))} \,. \tag{43}$$

Thus, all products of displacements or strains can be written as $A e^{i\omega(T_0 + \mathbf{q}\cdot(\mathbf{r}-\mathbf{r}_0))}$ with different explicit meanings of A depending on the specific term considered and $\mathbf{q} = \mathbf{p}^s + \mathbf{p}^r$. Since A is considered constant in a cell, the task is to integrate the expression $e^{i\omega(T_0 + \mathbf{q}\cdot(\mathbf{r}-\mathbf{r}_0)))}$ over an inversion cell. In this way, the sensitivity kernel becomes frequency dependent because the value of the integral depends decisively on the variation of the phase within the cell. This integration can be done analytically.

We consider the cell as a little spherical chunk of radius Δr and angular widths $\Delta\theta$ and $\Delta\phi$. In spherical coordinates, we find

$$\mathbf{r} - \mathbf{r}_0 = (r - r_0)\mathbf{e}_r + r_0(\theta - \theta_0)\mathbf{e}_\theta + r_0 \sin\theta_0 (\phi - \phi_0)\mathbf{e}_\phi \,, \tag{44}$$

where the basis vectors are also taken at the cell center. Moreover,

$$\mathbf{q}\cdot(\mathbf{r} - \mathbf{r}_0) = q_r(r - r_0) + q_\theta r_0(\theta - \theta_0) + q_\phi r_0 \sin\theta_0 (\phi - \phi_0) \,. \tag{45}$$

After performing the substitutions $\mathbf{r}' = \mathbf{r} - \mathbf{r}_0$, $r' = r - r_0$, $\theta' = \theta - \theta_0$ and $\phi' = \phi - \phi_0$, we want to calculate the integral

$$\begin{aligned} J &= A \int_{-\Delta r/2}^{\Delta r/2} \int_{-\Delta\theta/2}^{\Delta\theta/2} \int_{-\Delta\phi/2}^{\Delta\phi/2} e^{i\omega(T_0+\mathbf{q}\cdot\mathbf{r}')} \, (r'+r_0)^2 \sin(\theta'+\theta_0) \, dr' \, d\theta' \, d\phi' \\ &= A \int_{-\Delta r/2}^{\Delta r/2} \int_{-\Delta\theta/2}^{\Delta\theta/2} \int_{-\Delta\phi/2}^{\Delta\phi/2} e^{i\omega T_0} e^{i\omega q_r r'} e^{i\omega q_\theta r_0 \theta'} e^{i\omega q_\phi r_0 \sin\theta_0 \phi'} \, (r'+r_0)^2 \sin(\theta'+\theta_0) \, dr' \, d\theta' \, d\phi' \\ &= A e^{i\omega T_0} \int_{-\Delta r/2}^{\Delta r/2} e^{i\omega q_r r'} (r'+r_0)^2 \, dr' \int_{-\Delta\theta/2}^{\Delta\theta/2} e^{i\omega q_\theta r_0 \theta'} \sin(\theta'+\theta_0) \, d\theta' \int_{-\Delta\phi/2}^{\Delta\phi/2} e^{i\omega q_\phi r_0 \sin\theta_0 \phi'} \, d\phi' \, . \end{aligned} \tag{46}$$

Since r', θ' and ϕ' are small quantities varying around the cell center, we may use linear approximations as follows:

$$(r'+r_0)^2 = r_0^2 + 2r_0 r' \quad \text{and} \quad \sin(\theta'+\theta_0) = \sin\theta_0 + \theta' \cos\theta_0 \, , \tag{47}$$

and finally obtain

$$J = A e^{i\omega T_0} \int_{-\Delta r/2}^{\Delta r/2} e^{i\omega q_r r'} (r_0^2 + 2r_0 r') \, dr' \int_{-\Delta\theta/2}^{\Delta\theta/2} e^{i\omega q_\theta r_0 \theta'} (\sin\theta_0 + \theta' \cos\theta_0) \, d\theta' \int_{-\Delta\phi/2}^{\Delta\phi/2} e^{i\omega q_\phi r_0 \sin\theta_0 \phi'} \, d\phi' \, . \tag{48}$$

Thus, the integral of any product of displacements or strains can be written as the value of the product at the cell center times the product of three weights given by the integrals over radius and angles, respectively:

$$\begin{aligned} w_r &= \int_{-\Delta r/2}^{\Delta r/2} e^{i\omega q_r r'} (r_0^2 + 2r_0 r') \, dr' \\ w_\theta &= \int_{-\Delta\theta/2}^{\Delta\theta/2} e^{i\omega q_\theta r_0 \theta'} (\sin\theta_0 + \theta' \cos\theta_0) \, d\theta' \\ w_\phi &= \int_{-\Delta\phi/2}^{\Delta\phi/2} e^{i\omega q_\phi r_0 \sin\theta_0 \phi'} \, d\phi' \, . \end{aligned} \tag{49}$$

Thus, we need to solve integrals of the forms

$$J_1 = \int_{-h/2}^{h/2} e^{i\omega q s} \, ds \quad \text{and} \quad J_2 = \int_{-h/2}^{h/2} s e^{i\omega q s} \, ds \, . \tag{50}$$

Solutions are

$$J_1 = \int_{-h/2}^{h/2} e^{i\omega q s}\, ds = \left[\frac{e^{i\omega q s}}{i\omega q}\right]_{-h/2}^{h/2} = \frac{1}{i\omega q}\left(e^{i\omega q h/2} - e^{-i\omega q h/2}\right)$$
$$= \frac{2}{\omega q}\sin\frac{\omega q h}{2} = h\frac{\sin\frac{\omega q h}{2}}{\frac{\omega q h}{2}} = h\, j_0(\frac{\omega q h}{2})\,, \tag{51}$$

where $j_0(x)$ is the spherical Bessel function of zeroth order. Using partial integration,

$$J_2 = \int_{-h/2}^{h/2} s e^{i\omega q s}\, ds = \left[\frac{s e^{i\omega q s}}{i\omega q}\right]_{-h/2}^{h/2} - \frac{1}{i\omega q}\int_{-h/2}^{h/2} e^{i\omega q s}\, ds$$
$$= \frac{1}{i\omega q}\left(\frac{h}{2}e^{i\omega q h/2} + \frac{h}{2}e^{-i\omega q h/2} - \frac{2}{\omega q}\sin\frac{\omega q h}{2}\right)$$
$$= \frac{1}{i\omega q}\left(h\cos\frac{\omega q h}{2} - \frac{2}{\omega q}\sin\frac{\omega q h}{2}\right)$$
$$= \frac{h}{i\omega q}\left(\cos\frac{\omega q h}{2} - \frac{2}{\omega q h}\sin\frac{\omega q h}{2}\right)$$
$$= \frac{-ih^2/2}{\omega q h/2}\left(\cos\frac{\omega q h}{2} - \frac{2}{\omega q h}\sin\frac{\omega q h}{2}\right)$$
$$= \frac{-ih^2}{2}\left(\frac{\cos\frac{\omega q h}{2}}{\omega q h/2} - \frac{\sin\frac{\omega q h}{2}}{(\omega q h/2)^2}\right)$$
$$= \frac{ih^2}{2}\, j_1(\frac{\omega q h}{2})\,, \tag{52}$$

where $j_1(x)$ is the spherical Bessel function of first order. The first integral is purely real while the second one is purely imaginary. In the limit for $\omega \to 0$, the first integral goes to h and the second one to 0.

The weights can now be written as

$$w_r = r_0^2(J_1(q_r, \Delta r) + \frac{2}{r_0}\, J_2(q_r, \Delta r)) = r_0^2 \Delta r\left(j_0(\omega q_r \Delta_r/2) + i\frac{\Delta r}{r_0} j_1(\omega q_r \Delta_r/2)\right)$$
$$w_\theta = \sin\theta_0\, J_1(r_0 q_\theta, \Delta\theta) + \cos\theta_0\, J_2(r_0 q_\theta, \Delta\theta)$$
$$= \sin\theta_0 \Delta\theta\left(j_0(\omega r_0 q_\theta \Delta\theta/2) + i\cot\theta\frac{\Delta\theta}{2} j_1(\omega r_0 q_\theta \Delta\theta/2)\right)$$
$$w_\phi = J_1(r_0 \sin\theta_0 q_\phi, \Delta\phi) = \Delta\phi\, j_0(\omega r_0 \sin\theta_0 q_\phi \Delta\phi/2)\,. \tag{53}$$

Since $\Delta r/r_0$, $\Delta\theta$ and $\Delta\phi$ have values typically well below 10^{-2}, the imaginary parts of the weights can be safely neglected. Defining the product of the weights as

$$W(\mathbf{q}) = w_r\, w_\theta\, w_\phi = j_0(\frac{\omega q_r \Delta_r}{2})\, j_0(\frac{\omega q_\theta r_0 \Delta\theta}{2}) j_0(\frac{\omega q_\phi r_0 \sin\theta_0 \Delta\phi}{2})\, r_0^2 \sin\theta_0 \Delta r \Delta\theta \Delta\phi\,, \tag{54}$$

and indicating their dependence on the sum of the slowness vectors explcitly, then, for example, the integrated density kernel can then be expressed as follows (with summation over index j

implied):

$$K_\rho = \omega^2 \, A_j^s e^{i\omega T_0^s} A_j^r e^{i\omega T_0^r} \, W(\mathbf{p}^s + \mathbf{p}^r) = \omega^2 \, u_j^s(\mathbf{r}_0) \, u_j^r(\mathbf{r}_0) \, W(\mathbf{p}^s + \mathbf{p}^r)$$
$$= \omega^2 \, k_\rho^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^s + \mathbf{p}^r) \, . \tag{55}$$

If there arrive N_s phases from the source (e. g. direct P wave plus surface P reflection plus surface S conversion) and N_r phases from the receiver position (for example the P and the S wave), we can consider s and r as indices running over these phases and write the kernel as a double sum over all possible $N_s \times N_r$ products of these waves:

$$K_\rho = \omega^2 \sum_{s=1}^{N_s} \sum_{r=1}^{N_r} u_j^s(\mathbf{r}_0) \, u_j^r(\mathbf{r}_0) \, W(\mathbf{p}^s + \mathbf{p}^r)$$
$$= \omega^2 \sum_{s=1}^{N_s} \sum_{r=1}^{N_r} k_\rho^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^s + \mathbf{p}^r) \, . \tag{56}$$

For the λ-kernel on the wavefield points, we can write (with summation convention implied):

$$k_\lambda = -E_{kk}^r e^{i\omega T_0^r} e^{i\omega \mathbf{q}^r \cdot (\mathbf{r} - \mathbf{r}_0)} E_{mm}^s e^{i\omega T_0^s} e^{i\omega \mathbf{q}^s \cdot (\mathbf{r} - \mathbf{r}_0)}$$
$$= -E_{kk}^r e^{i\omega T_0^r} E_{mm}^s e^{i\omega T_0^s} e^{i\omega (\mathbf{p}^r + \mathbf{p}^s) \cdot (\mathbf{r} - \mathbf{r}_0)} \, . \tag{57}$$

Upon integration over the grid cell we get:

$$K_\lambda = -E_{kk}^r e^{i\omega T_0^r} E_{mm}^s e^{i\omega T_0^s} \, W(\mathbf{p}^r + \mathbf{p}^s) = -e_{kk}^r(\mathbf{r}_0) e_{mm}^s(\mathbf{r}_0) \, W(\mathbf{p}^r + \mathbf{p}^s)$$
$$= k_\lambda^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^r + \mathbf{p}^s) \, . \tag{58}$$

For multiple waves from source and receiver we can again write the double sum:

$$K_\lambda = -\sum_{s=1}^{N_s} \sum_{r=1}^{N_r} k_\lambda^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^r + \mathbf{p}^s) \, . \tag{59}$$

For the μ-kernel, we can directly write:

$$K_\mu = -2(e_{11}^r(\mathbf{r}_0) e_{11}^s(\mathbf{r}_0) + e_{11}^r(\mathbf{r}_0) e_{11}^s(\mathbf{r}_0) + e_{11}^r(\mathbf{r}_0) e_{11}^s(\mathbf{r}_0)) \, W(\mathbf{p}^r + \mathbf{p}^s)$$
$$- 4(e_{23}^r(\mathbf{r}_0) e_{23}^s(\mathbf{r}_0) + e_{13}^r(\mathbf{r}_0) e_{13}^s(\mathbf{r}_0) + e_{12}^r(\mathbf{r}_0) e_{12}^s(\mathbf{r}_0)) \, W(\mathbf{p}^r + \mathbf{p}^s))$$
$$= k_\mu^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^r + \mathbf{p}^s) \, . \tag{60}$$

For multiple waves from source and receiver we can again write the double sum:

$$K_\mu = \sum_{s=1}^{N_s} \sum_{r=1}^{N_r} k_\mu^{sr}(\mathbf{r}_0) \, W(\mathbf{p}^r + \mathbf{p}^s) \, . \tag{61}$$

To compute the integrated kernels, we can follow a very simple rule: Compute the waveform kernel at the center of the cell for all possible combinations of phases from source and receiver, multiply with the total weight calculated for the sum of the corresponding slownesses and sum over all combinations.