Phase Picking Workflow

Lachlan Adams

February 4, 2022

Contents

Introduction

This document describes the parallel arrival picking workflow, the interaction of the various subsystems in HiPerSeis and how an ensemble set of P and S arrivals are aggregated for generating body-wave tomography products.

1 Retrieve Data

The catalogues used come from a variety of sources. These are:

- Gary Gibson hypocentre catalogue - this is a catalogue in comma separated values (.csv) format containing hypocentres of seismic events in Australia from 1840 until the early 2000s.
- Bob Engdahl catalogue - these are extensible markup language (.xml) files in the SC3ML format containing information about events in the Australian region from the 1990s to late 2000s.
- ISC catalogue (International Seismological Centre, 2022) - these are .xml files in the QuakeML format which are downloaded from the International Seismological Centre web services. We only use the reviewed ISC catalogue.
- GA catalogue - these are .xml files in the QuakeML format which contain events which have been reviewed by analysts from Geoscience Australia.
- USGS catalogue - these are .xml files in the QuakeML format downloaded from a ftp server provided by the United States Geological Survey. This server contains both reviewed and un-reviewed events.

These catalogues are merged and filtered using the following criteria:

- Events in the GA database have their hypocentre replaced with those in the Gary Gibson files, if a match is found.
- Pairs of events where the hypocentre differs by less than 50km and the origin times are within 15 seconds of one another are considered to be duplicates. In this case, preferred hypocentres are Bob Engdahl $>$ ISC $>$ GA $>$ USGS.
- ISC and USGS events in the Australian bounding box ($\phi \in [105, 160]$ and $\theta \in [-40, -13]$ with $\phi, \theta =$ longitude and latitude) are filtered so that only events with an azimuthal gap of less than 180 degrees are retained.
- ISC and USGS events in the Indian Ocean ($\phi \in [15, 90]$ and $\theta \in [-60, 30]$, or $\phi \in [90, 180]$ and $\theta \in [-60, -45]$) are retained only if their magnitude is larger than 5.0.
- ISC and USGS events outside these regions are kept if their magnitude is above 5.5.
- GA events are filtered to have an azimuthal gap of less than 180°.

The events from the Geoscience Australia catalogue may be downloaded on the NCI using the following console command.

```
python hiperseis/seismic/catalogue/download_catalogues/download_GA/
↪  downloadquakeml.py --agency <agency name> --start_time
↪  <YYYY-MM-DDThh:mm:ss> --end_time <YYYY-MM-DDThh:mm:ss>
↪  --min_magnitude <min_mag> --max_magnitude <max_mag> --bounding_box
↪  <[min_lat, min_lon, max_lat, max_lon]> --output <.../output_path/>
```

The events from the ISC web services may be downloaded onto the NCI using the following console command. The 'split_by_day' command is a flag telling the algorithm to save one .xml file for each day instead of each month (default). Use this flag if there are expected to be a large amount of events over the time period specified.

```
python hiperseis/seismic/catalogue/download_catalogues/
↪  download_ISC/download_isc.py --output_path <.../output_path>
↪  --start_date <YYYY-MM-DD> --start_time <hh:mm:ss> --end_date
↪  <YYYY-MM-DD> --end_time <hh:mm:ss> --bbox <min_lon min_lat max_lon
↪  max_lat> --mag_range <min_mag max_mag> --split_by_day <True>
```

Events included in the USGS database may be downloaded using the following console command. These must be preprocessed in the next step separately to those in the other catalogues.

```
python hiperseis/seismic/catalogue/download_catalogues/
↪  download_USGS/download_usgs_catalogue.py --output_path
↪  <.../output_path> --start_date <YYYY-MM-DD> --end_date <YYYY-MM-DD>
```

2 Merge Catalogues

Merging and filtering of the catalogues can be performed on a single processor, however it is recommended that some parts of the process are performed on multiple processors. If there is a large amount of event .xml files, then the time taken to merge the catalogues may be reduced by first producing binary files which contain pickled versions of the catalogues using the pbs script in section 2.1.

2.1 Preprocessing Input Data

If desired, preprocessing of input .xml files should be performed with the following pbs script. Note that hiperseis is contained in /g/data/ha3/la8536/hiperseis/, and /g/data/ha3/la8536/VENV/para_h5py/lib/python3.7/site-packages/ contains some required python libraries. The .xml files are in /g/data/ha3/la8536/Catalogue/, and the flag –split_smaller is telling the algorithm to produce one binary file per input .xml file (otherwise one is produced per processor).

If preprocessing the USGS catgalogue, as no filters can be applied in the download step (apart from date filters), Read_USGS_XML_Multiprocessing.py can be used to apply the filters described in section 1. Arguments are the same arguments as the script included below.

```
#!/bin/bash

# PBS directives
#PBS -P vy72
#PBS -N make_event_binaries
#PBS -q normal
#PBS -l walltime=24:00:00
#PBS -l mem=100GB
#PBS -l ncpus=240
#PBS -l jobfs=100GB
```

```
#PBS -l storage=scratch/la8536+gdata/ha3
#PBS -l wd
#PBS -j oe
#PBS -M lachlan.adams@ga.gov.au
#PBS -m bae


export PATH=/g/data/ha3/la8536/.local/bin:$PATH
export PYTHONPATH=/g/data/ha3/la8536/hiperseis/:
↪ /g/data/ha3/la8536/VENV/para_h5py/lib/python3.7/site-packages/:
↪ $PYTHONPATH
export LC_ALL=en_AU.UTF-8
export LANG=en_AU.UTF-8

module purge

module load pbs
module load python3/3.7.4
module load hdf5/1.10.5p
module load openmpi/4.0.1

module list

mpirun -np $PBS_NCPUS python3
↪ /g/data/ha3/la8536/hiperseis/seismic/catalogue/merge_catalogues/
↪ Read_XML_Multiprocessing.py --paths /g/data/ha3/la8536/Catalogue/
↪ --output_path /g/data/ha3/la8536/Catalogue/event_binary_files/
↪ --split_smaller True
```

2.2 Filtering Catalogue

On a single processor, the script Filter_Catalogues.py can be run in the command line to filter all input catalogues. The algorithm can either read the output of Read_XML_Multiprocessing.py to save time, or can read the event .xml files directly (if the dataset is small).

Criteria currently used for filtering are described in section 1. If using the preprocessed pickled catalogues (.pkl files), set the flag –read_from_preprocessed to True and leave –GA_path (GA catalogue), –ISC_path (ISC catalogue), –USGS_path (USGS catalogue)), and –EHB_path (Engdahl catalogue) blank. Otherwise leave –preprocessed_path blank. –GG_path is the directory containing the Gary Gibson catalogue.

```
python3 hiperseis/seismic/catalogue/merge_catalogues/
↪ Filter_Catalogues.py --preprocessed_path <.../path/> --GG_path
↪ <.../path/> --output_path <.../path/> --read_from_preprocessed True
```

Or

```
python3 hiperseis/seismic/catalogue/merge_catalogues/
↪ Filter_Catalogues.py --GA_path <.../path/> --ISC-path <.../path/>
↪ --EHB_path <.../path/> --GG_path <.../path/> --USGS_path <.../path/>
↪ --output_path <.../path/>
```

2.3 Converting Catalogue to CSV

The conversion from the Obspy earthquake catalogue format to a format usable by the software for later parts of this workflow is a computationally expensive operation. For

this reason it should be run on multiple processors.

The Convert_Catalogues_To_CSV.py script reads the output from section 2.2 on multiple processors, converts each event into a list format containing all required information, and saves each event as a separate .csv file. This is because the time taken to collate the output of each processor into a single object within the multiprocessing environment is larger than the time taken to save the output to disk and read it again using the script in section 2.4. In this step, station coordinates are matched with each pick, if an inventory of stations is available. In this case, –station_path points to the directory in which the FDSN station .xml inventory and ISC International Registry (IR) .kml inventory are stored. To perform this step of the workflow, run the following PBS script.

```
#!/bin/bash

# PBS directives
#PBS -P vy72
#PBS -N make_csv
#PBS -q hugemem
#PBS -l walltime=10:00:00
#PBS -l mem=500GB
#PBS -l ncpus=48
#PBS -l jobfs=200GB
#PBS -l storage=scratch/la8536+gdata/ha3
#PBS -l wd
#PBS -j oe
#PBS -M lachlan.adams@ga.gov.au
#PBS -m bae


# this is tested working in NCI gadi system.
export PATH=/g/data/ha3/la8536/.local/bin:$PATH
export PYTHONPATH=/g/data/ha3/la8536/hiperseis/:
↪ /g/data/ha3/la8536/VENV/para_h5py/lib/python3.7/site-packages/:
↪ $PYTHONPATH
export LC_ALL=en_AU.UTF-8
export LANG=en_AU.UTF-8

module purge

module load pbs
module load python3/3.7.4
module load hdf5/1.10.5p
module load openmpi/4.0.1

module list

mpirun -np $PBS_NCPUS python3 /g/data/ha3/la8536/Catalogue/
↪ merge_catalogues/Convert_Catalogues_To_CSV.py --preprocessed_path
↪ <.../path/> --station_path <.../path/> --output_path <.../path/>
```

2.4 Merging Catalogues

On a single processor, the following script reads in all output files from section 2.3, and merges them into a single file after sorting the events by origin time. Perform this step by running the following console command.

```
python3 /g/data/ha3/la8536/Catalogue/merge_catalogues/
Merge_Catalogues.py --input_path <.../path> --output_file
<.../path/merge_catalogues_output.csv>
```

The output will be a .csv file with the following structure. Every pick row between event rows is considered as belonging to the event above it.

```
event_rows = [#source, YYYY, MM, DD, hh, mm, ss, lon, lat, depth,
↪  n_phase, mb, ms, ml, mw, event_id, azim_gap, index]
pick_rows = [sta, cha, loc, net, lon, lat, elev, phase, YYYY, MM, DD, hh,
↪  mm, ss, ang_dist]
```

3 Parallel Picker for Automatic Arrival Detection

To increase the amount of data available, arrivals from Adaptable Seismic Data Format (ASDF) files are scoured to attempt to match these arrivals with events in the merged catalogue. This can increase the ray path coverage of the region we wish to analyse. A list of available ASDF files is contained on the NCI in /g/data/ha3/Passive/SHARED_DATA/Index/asdf_files.txt.

The output of the algorithms in section 2.4 or 2.3 is read by the script /hiperseis/seismic/pick_harvester/pick.py and the picks in the files listed within asdf_files.txt are paired with these events, if a match is found. The script outputs two files, p_combined.txt and s_combined.txt, which contain P and S arrivals detected on all stations, for all events in the CSV events catalog.

To perform this step of the workflow, run the following pbs script. Arguments are (1) file containing asdf file names, (2) output path from section 2, (3) output path.

```
#!/bin/bash

# PBS directives
#PBS -P vy72
#PBS -N pick_pbs_job
#PBS -q hugemem
#PBS -l walltime=24:00:00,mem=1470GB,ncpus=96,jobfs=1000GB
#PBS -l storage=scratch/la8536+gdata/ha3
#PBS -l wd
#PBS -j oe
#PBS -M lachlan.adams@ga.gov.au
#PBS -m bae


export PATH=/g/data/ha3/la8536/.local/bin:$PATH
export PYTHONPATH=/g/data/ha3/la8536/hiperseis/:
↪  /g/data/ha3/la8536/VENV/para_h5py/lib/python3.7/site-packages/:
↪  $PYTHONPATH
export LC_ALL=en_AU.UTF-8
export LANG=en_AU.UTF-8

module purge

module load pbs
module load python3/3.7.4
module load hdf5/1.10.5p
module load openmpi/4.0.1

```

```
27 module list
28
29 mpirun -np $PBS_NCPUS python3
   ↪ /g/data/ha3/la8536/hiperseis/seismic/pick_harvester/pick.py
   ↪ /g/data/ha3/Passive/SHARED_DATA/Index/asdf_files.txt
   ↪ /g/data/ha3/la8536/Catalogue/Outputs/
   ↪ /g/data/ha3/la8536/Picking/Step1 >
   ↪ /g/data/ha3/la8536/Picking/Step1/run_output.txt
```

The output will be two .txt files with the following structure. One file contains 'P' picks, while the other contains 'S' picks.

```
pick_rows = [event_id, origin_time, mag, elon, elat, edepth (km), net,
↪ sta, cha, arrival_time, slon, slat, az, baz, dist, residual, snr,
↪ qualityMeasureCWT, domFreq, qualityMeasureSlope, bandIndex, nsigma]
```

4 Creating Inputs for SSST Relocation

4.1 Seismic Data

Before performing relocation of events using the additional picks and Source Specific Station Terms (SSSTs), the picks found in section 3 must be merged with the event output file from section 2. To be used by the relocation algorithm, the files are converted into a numpy structured array with the following data type.

```
dtype = [('event_id', 'U20'), ('pick_id', 'U30'), ('stat', 'U10'),
↪ ('net', 'U5'), ('cha', 'U10'), ('elon', 'single'), ('ecolat',
↪ 'single'), ('edepth', 'single'), ('origin_time', 'double'), ('mag',
↪ 'half'), ('slon', 'single'), ('scolat', 'single'), ('selev', 'half'),
↪ ('phase', 'U8'), ('arrival_time', 'double'), ('ptt', 'single'),
↪ ('tcor', 'half'), ('residual', 'half'), ('snr', 'half'),
↪ ('qualityMeasureCWT', 'half'), ('domFreq', 'half'),
↪ ('qualityMeasureSlope', 'half'), ('bandIndex', 'uint8'), ('nSigma',
↪ 'uint8')]
```

Running the following python script will produce a file in the correct format for input in the next section.

```
python3 convert_for_relocation.py --event_file <Output of merged
↪ catalogue> --p_combined <p_combined.txt> --s_combined
↪ <s_combined.txt> --output_path <output_path>
```

Additionally, if the iLoc relocation algorithm is to be used (see section 5.3), .xml files in the 'SC3ML' format must be created to be read into a SeisComp3 mysql database (Helmholtz-Centre Potsdam - GFZ German Research Centre for Geosciences and gempa GmbH, 2008). This is done using the following script, on the Amazon Web Service instance described in section 5.3.

```
mpirun -np <number of processors> python convert_array_to_catalogue.py
↪ --event_file <Output of convert_for_relocation> --config_file
↪ <configuration file> --output_path <output_path>
```

The configuration file in this instance contains a list of network IDs for temporary deployments. We may wish to exclude picks on these networks from the relocation procedure if they are detected using the automatic picker. If they are excluded, they still have their travel time residuals calculated and phases redefined, but are not fed into iLoc. The configuration file should look similar to the format below.

```
[default]
temp_networks = 7B, 7D, 7E, 7F, 7G, 7J, 7Q, 7T, 7U, 7W, 7X, OA, W1, AQ
ignore_temp_networks = True
```

4.2 Pre-computed Predicted Travel Times

The phase redefinition and earthquake relocation algorithms use a set of pre-computed travel time tables to perform travel time predictions. These may be created using the TauP toolkit (Crotwell et al., 1999). Alternatively, if the iLoc relocation algorithm is to be used (see section 5.3), the iLoc travel time tables can be used.

If the iLoc travel time tables are to be used, they must be converted into a simpler format to be read by the algorithm in section 5.2 or 5.3. This can be done using the following script.

```
python3 tt_table_calculator.py --iloc_path <path_to_iloc_tables>
↪  --output_path <output_path> --from_iloc_tables True
```

If the travel time tables are to be computed using the TauP toolkit, the same script may be used, however a configuration file is needed to tell the algorithm the phases, epicentral distance coordinates, and depth coordinates to use for the tables. The script may be run using the following command.

```
python3 tt_table_calculator.py --model <model_name> --config_file
↪  <config_file_name> --output_path <output_path>
```

The configuration file should be a .txt file in the format below. Rows are (1) phase, (2) epicentral distance in degrees, (3) depth in kilometres.

```
-----------------------------------------------------
[P]
dists = 0.000 1.000 2.000 3.000 4.000 5.000
depths = 0.000 1.000 2.500 5.000 7.500 10.000 15.000

[Pg p]
dists = 0.000 0.025 0.050 0.075 1.000
depths = 0.000 1.000 2.500 5.000 7.500 10.000

[S]
dists = 0.000 1.000 2.000 3.000 4.000 5.000
depths = 0.000 1.000 2.500 5.000 7.500 10.000 15.000
-----------------------------------------------------
```

This config.txt file tells the algorithm to produce 3 tables. The first table will be a 6×7 grid of travel times for 'P' waves. The second will be a 5×6 grid of travel times for the first arriving 'Pg' or 'p' wave. The third will be a 6×7 grid of travel times for 'S' waves.

5 Redefining Phases and Relocating Events

There are two methods that may be used for phase redefinition and event relocation.

The first of these methods is named iLoc. iLoc (Bondar and Storchak, 2011) is an earthquake phase identification/redefinition code that updates each event origin location by iteratively optimizing some misfit function that factors in a number of variables including travel-time residuals, etc. iLoc works off of parametric data from a SeisComp3 server and while iLoc also runs off of text input, the extreme slowness of execution using this method renders it almost unusable.

The second method is a heavily modified version of the Singleloc earthquake location algorithm included in the XCorLoc package (Lin, 2018). This algorithm also uses a nonlinear minimisation technique to find a best fit hypocentre for each event, however doesn't have phase redefinition capabilities.

Using either of these methods, the phase redefinition and earthquake relocation algorithm performs the following procedure iteratively.

1. Read pick information from input file created in section 4.1.
2. For each pick, calculate predicted travel times for all available phases.
3. Redefine phase for each pick as that corresponding to the smallest travel time residual.
4. Compute time corrections for each station. These may either be static station terms or source specific station terms. Apply the corrections to the picks.
5. Compute new best fitting hypocentres using minimisation algorithm.
6. Reject new hypocentres if they differ by more than a certain threshold distance from original hypocentre. Otherwise update hypocentres for events.

The iLoc relocation algorithm makes use of Brian Kennett's ellipticity correction algorithm to account for variation in travel times due to Earth's ellipticity (Kennett and Gudmundsson, 1996). The default algorithm uses the same coefficient table with a different interpolation method to perform ellipticity corrections.

The use of static station terms (SSTs) when performing earthquake relocation can account for differences in seismic travel times caused by heterogeneity in the region about a receiver. This method applies the median travel time residual of all arrivals at a station as a time correction.

Using source specific station terms (SSSTs) can improve the results further by accounting for heterogeneity along the path travelled between a source and a receiver. In this method a spatially varying time correction is applied to the picks. For the method employed in this workflow, time corrections are calculated in the following way.

1. Find all arrivals at a station of a particular phase.
2. For each event corresponding to these arrivals, find all other events within a threshold distance.
3. Compute the median travel time residual for the event and its nearest neighbours (within threshold distance).
4. Apply this median residual as a correction to the arrival time corresponding to this event.

If there is an insufficient number of nearest neighbours then a SSST may not be meaningful, as we may have situations occurring where a pick simply has its travel time residual subtracted from its arrival time. In this case the relocation algorithm will compute a misfit of 0. If such a situation arises then the algorithm will not compute a time correction.

This section details the workflow for the use of either the iLoc relocation algorithm, or the modified Singleloc relocation algorithm. In each of the following subsections, the configuration file referred to is contained in appendix A.

5.1 Phase redefinition only

If a user only wishes to calculate travel time residuals, the phase redefinition and relocation algorithm may be run on the NCI with option "–no_relocation" set to True in the configuration file. This may be done using the following command.

```
python3 hiperseis/seismic/ssst_relocaiton/relocation/main.py
↪ --config_file <configuration file name> --tt_table_path <travel time
↪  table directory> -- input_path <path containing picks.npy> --elcordir
↪ <path containing elcordir.tbl> --iteration 0 --output_path <path
↪  containing picks.npy>
```

Alternatively, iLoc has the capability to redefine phases without relocation. If this is desirable instead, follow the procedure in appendix D instead of those contained in this section and section 6.

5.2 Relocation using modified Singleloc algorithm

This relocation and phase redefinition method may be run on the NCI and uses the .npy binary file produced in section 4.1 as input. The algorithm should be run for multiple iterations to obtain stable best-fitting hypocentres for all events available.

The relocation algorithm has been written in Fortran to improve computation time. The Fortran subroutines used are contained in "relocation_heloer.f" and are all written so that they may be converted into Python functions using the "numpy.f2py" compiler. A library compiled using numpy.f2py may not be compatible with updated (or downgraded) version of Python or of Numpy. Because of this, it is a good idea to reproduce the library before running the relocation algorithm. To do this, run the following command.

```
python -m numpy.f2py -c -m relocation_helper relocation_helper.f
```

This command will take the file "relocation_helper.f", find all the subroutines within it that may be converted into Python functions, and will produce a library "relocation_helper.so" (or "relocation_helper.pyc" on Windows) which may be imported by the relocation algorithm.

To perform relocation of the events and redefinition of phases, run the following pbs script. In this case the number of iterations is set to 5 (in line 33). The first run produces an output file containing updated hypocentres if no corrections are used, and a text file containing names of events which have an unstable hypocentre. These events are not to be used for computing time corrections. The subsequent runs use the original input pick file, plus this unstable event text document, to perform 5 iterations of the relocation algorithm using SSST or SST corrections. If not using SSST or SST corrections, comment out the for loop.

```
1  #!/bin/bash
2
3  # PBS directives
4  #PBS -P vy72
5  #PBS -N relocate_events
6  #PBS -q normal
7  #PBS -l walltime=00:30:00
8  #PBS -l mem=950GB
9  #PBS -l ncpus=240
10 #PBS -l jobfs=200GB
11 #PBS -l storage=scratch/la8536+gdata/ha3
12 #PBS -l wd
13 #PBS -j oe
14 #PBS -M lachlan.adams@ga.gov.au
15 #PBS -m bae
16
17 export PATH=/g/data/ha3/la8536/.local/bin:$PATH
18 export PYTHONPATH=/g/data/ha3/la8536/hiperseis/:
   ↪ /g/data/ha3/la8536/VENV/para_h5py/lib/python3.7/site-packages/:
   ↪ $PYTHONPATH
19 export LC_ALL=en_AU.UTF-8
20 export LANG=en_AU.UTF-8
21
22 module purge
23
24 module load pbs
25 module load python3/3.7.4
26 module load hdf5/1.10.5p
```

```
module load openmpi/4.0.1

module list

mpirun -np $PBS_NCPUS python3
↪  hiperseis/seismic/ssst_relocation/relocation/main.py --config_file
↪  <configuration file name> --tt_table_path <travel time table
↪  directory> -- input_path <path containing picks.npy> --elcordir <path
↪  containing elcordir.tbl> --iteration 0 --output_path <path containing
↪  picks.npy>

for i in `seq 1 5`;
do mpirun -np $PBS_NCPUS python3
↪  hiperseis/seismic/ssst_relocation/relocation/main.py --config_file
↪  <configuration file name> --tt_table_path <travel time table
↪  directory> -- input_path <path containing picks.npy> --elcordir <path
↪  containing elcordir.tbl> --iteration $i --output_path <path
↪  containing picks.npy>;
done
```

The progress of execution of this script may be tracked by looking at 'out.txt' in the output directory. Upon completion this directory will contain a series of .npy binary files containing the output pick list after each iteration (picks0.npy, picks1.npy, etc) and a file 'unstable_events.txt' containing the names of events for which the hypocentre changed by more than 'hypo_thr_dist_deg' degrees (specified in configuration file) in a single iteration. Note that at each iteration the algorithm searches for "unstable_events.txt" in the same directory where it searches for the "picks$i.npy" file. At the completion of this step, go to step 6.

5.3 Relocation using iLoc algorithm

Running the relocation algorithm using iLoc requires a few extra steps, mainly for setting up the SeisComp3 database on the Amazon Web Service (AWS) instance. The procedure uses the .npy binary file as well as the .xml files produced in section 4.1 as input. The algorithm should be run for multiple iterations to obtain stable best-fitting hypocentres for all events available.

In order to use iLoc, SeisComp3 is used for data management. For this reason, an AWS instance is used for this step, as SeisComp3 can not be used on the NCI. X-forwarding should be used to connect to the AWS instance, currently accessible at centos@54.153.234.37 using a parametric key.

If starting from scratch, or if the database needs to be reset from scratch, follow the steps in appendix B. To prepare the SeisComp3 database for iLoc processing, follow the procedure in appendix C. The steps for preparing a SeisComp3 instance for ingesting XML events are:

1. Delete all existing inventories.
2. Adapt and ingest all available inventories.
3. Create bindings for inventories.
4. Delete all preexisting events and arrivals.
5. Ingest .xml event files into SeisComp3

To perform relocation and phase redefinition using iLoc, run the following command in the command line. Note that in this instance, 5 iterations of the algorithm are performed. If not using SSST or SST corrections, set the number of iterations to 1.

```
mpirun -np 30 python3
↪  hiperseis/seismic/ssst_relocation/relocation/main.py --config_file
↪  <configuration file name> --tt_table_path <travel time table
↪  directory> -- input_path <path containing picks.npy> --elcordir <path
↪  containing elcordir.tbl> --iteration 0 --output_path <path containing
↪  picks.npy> --relocation_method iloc

for i in `seq 1 5`;
do mpirun -np 30 python3
↪  hiperseis/seismic/ssst_relocation/relocation/main.py --config_file
↪  <configuration file name> --tt_table_path <travel time table
↪  directory> -- input_path <path containing picks.npy> --elcordir <path
↪  containing elcordir.tbl> --iteration $i --output_path <path
↪  containing picks.npy> --relocation_method iloc;
done
```

The progress of execution of this script may be tracked by looking at 'out.txt' in the output directory. Upon completion this directory will contain a series of .npy binary files containing the output pick list after each iteration (picks0.npy, picks1.npy, etc) and a file 'unstable_events.txt' containing the names of events for which the hypocentre changed by more than 'hypo_thr_dist_deg' (set in the configuration file) degrees in a single iteration. Note that at each iteration the algorithm searches for "unstable_events.txt" in the same directory where it searches for the "picks$i.npy" file. At the completion of this step, go to step 6.

6 Extracting Picks from Relocated Events

The format of the file produced by the relocation and phase redefinition algorithm is not able to be used for tomography. The file can be converted in the correct format by running the following script. Effectively this is the inverse function for that in section 4.

```
python3 convert_after_relocation.py --event_file <Output of SSST routine>
↪  --output_path <Output path>
```

The output will be three separate .txt files ('P' picks, 'S' picks, and all combined) in the following format.

```
pick_rows = [event_id, origin_time, mag, elon, elat, edepth (km), net,
↪  sta, cha, arrival_time, slon, slat, az, baz, dist, residual, snr,
↪  qualityMeasureCWT, domFreq, qualityMeasureSlope, bandIndex, nsigma]
```

If a user wishes to produce diagnostic plots, with residuals calculated taking into account the time corrections applied, set the argument '–include_tcor' to True. This adds an extra column to the output files. However, these files can not be read into the tomographic inversion software. To generate the plots, run the following command.

```
python3 Plots.py --files <list of file names> --output_path <output path>
↪  --lonmin <value> --lonmax <value> --latmin <value> --latmax <value>
↪  --plot_residuals True --plot_epicentres True --plot_raypaths True
↪  --plot_depth_cross_section True --depth_cross_sect_params <lon1 lat1
↪  lon2 lat2 dist> --tcor_available True --show True
```

If a user chooses to create plots of depth cross sections for fault lines, the events will be projected along a line from $(lon1, lat1)$ to $(lon2, lat2)$ is they are less than $dist$ degrees from the line. The projection is done in Cartesian coordinates so this function should be revised if the distance scale is large enough that a Cartesian approximation of earth would not be sufficient. Some example output plots are shown in appendix E.

References

I. Bondar and D. Storchak. Improved location procedures at the International Seismological Centre. *Geophysical Journal International*, 2011. doi: https://doi.org/10.1111/j.1365-246X.2011.05107.x.

H. P. Crotwell, T. J. Owens, and J. Ritsema. The TauP Toolkit: Flexible seismic travel-time and ray-path utilities. *Seismological Research Letters*, 1999. doi: https://doi.org/10.1785/gssrl.70.2.154.

Helmholtz-Centre Potsdam - GFZ German Research Centre for Geosciences and gempa GmbH. The SeisComP seismological software package. *GFZ Data Services*, 2008. doi: http://dx.doi.org/10.5880/GFZ.2.4.2020.003.

International Seismological Centre. On-line Bulletin. 2022. doi: https://doi.org/10.31905/D808B830.

B Kennett and O Gudmundsson. Ellipticity corrections for seismic phases. *Geophysical Journal International*, 1996. doi: 10.1111/j.1365-246X.1996.tb01533.x.

G. Lin. The Source-Specific Station Term and Waveform Cross-Correlation Earthquake Location Package and Its Applications to California and New Zealand. *Seismological Research Letters*, 2018. doi: https://doi.org/10.1785/0220180108.

A Relocation configuration file

An example configuration file is shown below.

```
[iLoc]
hypo_thr_dist_deg = 0.25
hypo_thr_time_sec = 10.0
corr_thr_dist_deg = 0.25
thr_p = 5.0
thr_s = 10.0
no_relocation = False
correction_method = None
iloc_redefine_phases = False
iloc_use_rstt = False
phases = P, Pg, Pn, Pb, S, Sg, Sn, Sb

[default]
hypo_thr_dist_deg = 0.25
hypo_thr_time_sec = 10.0
corr_thr_dist_deg = 0.25
thr_p = 5.0
thr_s = 10.0
no_relocation = False
correction_method = SSST
reloc_dlat = 0.25
reloc_ddep = 10.0
reloc_nit = 5
reloc_norm = 1
reloc_sfrac = 0.5
phases = P, Pg, Pn, Pb, S, Sg, Sn, Sb
temp_networks = 7B, 7D, 7E, 7F, 7G, 7J, 7Q, 7T, 7U, 7W, 7X, OA, W1, AQ
```

Commands common for both methods are the following.

- hypo_thr_dist_deg - float, describing the maximum allowed distance (degrees) an epicentre may move during relocation before being classed as unstable.
- hypo_thr_time_sec - float, describing the maximum allowed time (seconds) an event's origin time may change during relocation before being classed as unstable.
- corr_thr_dist_deg - float, describing the radius of the sphere (degrees) to use for SSST time correction calculation.
- thr_p - float, describing the maximum size (seconds) of a P wave travel time residual allowable when redefining phases.
- thr_s - float, describing the maximum size (seconds) of a S wave travel time residual allowable when redefining phases.
- no_relocation - boolean, which if True will cause the algorithm to only perform phase redefinition without relocation.
- correction_method - string. May be 'None', 'SST', or 'SSST' describing the type of travel time correction desired.
- phases - list of string. This list of phases will be the ones used during the first pass of the phase redefinition algorithm. If a match isn't found from this list, all available phases are checked (e.g. reflected waves).

Commands specific for iLoc include the following. Both of these should be set to false, unless desired when using no travel time corrections. Allowing a travel time correction to take place and then allowing iLoc to change the phase is emaningless, as is using the RSTT models if the corrections are computed without them.

- iloc_redefine_phases - boolean, which if True will allow iLoc to redefine phases during its relocation procedure.
- iloc_use_rstt - boolean, which if True will allow iLoc to use the RSTT model for relocation.

Commands specific to the default algorithm include the following.

- reloc_dlat - float, describing the initial spacing between grid points for the grid search algorithm. This decreases by 'reloc_sfrac' amount each iteration.
- reloc_ddep - float, decribing the initial depth spacing between grid points for the grid search algorithm. This also decreases by 'reloc_sfrac' amount each iteration.
- reloc_nit - integer, number of iterations to use for default relocation algorithm.
- reloc_norm - integer. Set to 1 if L1 norm is used for minimisation routine, or 2 if L2 norm (RMS) is to be used.
- reloc_sfrac - float, describing the proportion that reloc_dlat and reloc_ddep will shrink by each iteration. E.g. if reloc_sfrac = 0.5, then on each iteration, reloc_dlat and reloc_ddep are halved.
- temp_networks - list, describing temporary seismic station deployments which are not to be used for relocation. Picks on these networks still have their phases redefined and residuals calculated.

B Resetting SeisComp3 Database

The following steps will reset the SeisComp3 database on the SeisComp3 Amazon Web Service (AWS) instance.

B.1 Drop the Database

Run mysql in the command line as user root, using password seiscomp3. Then, perform the following.

```
mysql > DROP DATABASE seiscomp3;
mysql > CREATE DATABASE seiscomp3 CHARACTER SET utf8 COLLATE
utf8_bin;
mysql > grant usage on seiscomp3.* to sysop@localhost identified
by ‘sysop’
mysql > grant all privileges on seiscomp3.* to sysop@localhost;
mysql > grant usage on seiscomp3.* to sysop@’%’ identified by
‘sysop’;
mysql > grant all privileges on seiscomp3.* to sysop@’%’;
mysql > flush privileges
```

Exit the database.

B.2 Setup New Database

Run the following script (contained within the seiscomp3 distribution) in the command line to add all tables and information to the database needed to run seiscomp3.

```
$ mysql < /opt/seiscomp3/share/db/mysql.sql
```

B.3 Setup and Configure SeisComp3

Run the following in the command line.

```
$ /opt/seiscomp3/bin/seiscomp stop
$ /opt/seiscomp3/bin/seiscomp setup
```

Enter the default values until it asks you “Create database [yes]:”. Select no and continue with the default values until the completion of setup.

In the command line, run the following.

```
$ /opt/seiscomp3/bin/seiscomp enable seedlink scautopick scautoloc
scamp scmag scevent
```

Start with an empty inventory in /opt/seiscomp3/etc/inventory/, and then create an empty file “inventory_empty.xml” in this folder. Import a dataless seed volume into this file, then update the configuration by running the following in the command line.

```
$ /opt/seiscomp3/bin/seiscomp exec import_inv dlsv
inventory_empty.xml
$ /opt/seiscomp3/bin/seiscomp update-config
```

B.4 Start SeisComp3

Finally, restart SeisComp3 by running the following in the command line.

```
$ /opt/seiscomp3/bin/seiscomp start
```

C Preparing SeisComp3 for iLoc Processing

C.1 Delete All Existing Inventories

To start off with a clean slate, log into the instance and remove all preexisting inventories from /opt/seiscomp3/etc/inventory. Note that this step is not necessary for subsequent iterations of processing, if the existing inventories have not changed or when some new inventories need to be added. When new inventories need to be added to the existing pool, simply copy them into this folder and follow the steps below.

In the command line, run the following.

```
$ scinv sync
$ scinv sync keys
$ scinv sync
$ scinv sync keys
```

The repetition of both commands is needed due to a SeisComp3 oddity. This will remove all keys from the database corresponding to stations no longer included in it. As detailed on the SeisComp3 project site https://www.seiscomp.de/seiscomp3/, with the current SeisComp3 release key files are no longer necessary which means that this step will need to be changed if the SeisComp3 version is updated.

C.2 Adapt and Ingest All Available Inventories

There are two sources of inventories:

- The folder /g/data/ha3/Passive/SHARED_DATA/Inventory/networks_sc3ml/ on the NCI contains SC3ML inventories. These files are copied to the folder /opt/seiscomp3/etc/inventory on the AWS instance.
- FDSNStationXML inventories for temporal deployments (provided by Jason) which are generally found in the folders /g/data/ha3/Passive/_ANU/ and /g/data/ha3/Passive/_AusArray/ on the NCI.

FDSNStationXML inventories in the above folders do not contain valid response objects. After attaching 'bogus' response objects to each channel we convert each inventory into SC3ML format as follows.

- Make a new folder on the AWS. Change directory into this folder and create two sub-folders named 'input' and 'output'. Copy all the FDSNStationXML inventories from the above folders on the NCI to the 'input' folder.
- From the command line, run the following.

  ```
  $ export PYTHONPATH=\$PYTHONPATH:/home/centos/hiperseis/
  ```

- Read all FDSNStationXML inventories in the input folder, insert valid response objects and produce the required SC3ML inventories in the output folder above. This is performed by running the following in the command line.

  ```
  $ python
  /home/centos/hiperseis/seismic/inventory/fdsnxml_convert.py
  input/ output/ --response-fdsnxml /home/centos/hiperseis/
  seismic/inventory/responses/LE3dLite.xml
  ```

  These files are then copied to the folder /opt/seiscomp3/etc/inventory on the AWS instance.

Run the following from the command line to synchronise the new inventory.

```
$ scinv sync
```

With the completion of this step, one is now able to directly download FDSNStationXMLs from the SeisComp3 server through a webservice query as follows.

```
$ wget http://ip_address:port/fdsnws/station/1/query -O
fdsn_inventory.xml
```

The IP address will change each time an AWS instance is restarted. Also note that SeisComp3 runs the fdsnws webservice on port 8081, configurable through /opt/seiscomp3/etc/fdsnws.cfg.

C.3 Create Bindings for Inventories

For each station ingested into the SeisComp3 inventory, a "binding" must be created. See https://www.seiscomp.de/seiscomp3/doc/seattle/2013.046/apps/global.html for more details. To do this, perform the following procedure.

1. Enter the command "scconfig" into the command line. This will open up the configuration window GUI, so X-forwarding must be enabled to perform this step.
2. On the left side of the GUI, click on bindings.
3. On the right side of the GUI, enter the drop down list for the "global" folder, and find the binding "__SHZ".
4. Drag and drop this onto the "Networks" folder on the left side of the screen. See figure 1 for a guide.
5. Exit the GUI. Click "yes" when asked if you wish to save the updated configuration.

C.4 Delete all Pre-Existing Events and Arrivals

On the AWS instance, delete all preexisting events and arrivals as follows.

```
$ scdbstrip --days 1 -d mysql://sysop:sysop@localhost/seiscomp3
```

It might take a while (hours). Running the command in a screen session will allow the user to disconnect from the AWS during execution of the command. In case the above fails (another SeisComp3 oddity), manually cleanse the database of all preexisting arrivals as follows.

```
$ mysql -e"delete Object from Object, Pick where Object._oid=
Pick._oid"
```

C.5 Ingest .xml Event Files Into SeisComp3

We now ingest the XML files (output of step 4.1) on the AWS instance as follows.

- Copy all the XML files to the AWS instance in a new folder and change directory into it. While copying files to AWS, gadi-dm.nci.org.au (instead of gadi) can support 10× network transfer rates.
- Run the following from the command line.

  ```
  for axml in `ls *.xml`; do sed -i 's%<seiscomp
  xmlns="http://geofon.gfz-potsdam.de/ns/seiscomp3-schema/0.10"
  version="0.10">%<seiscomp
  xmlns="http://geofon.gfz-potsdam.de/ns/seiscomp3-schema/0.9"
  version="0.9">%g' $axml; done
  ```

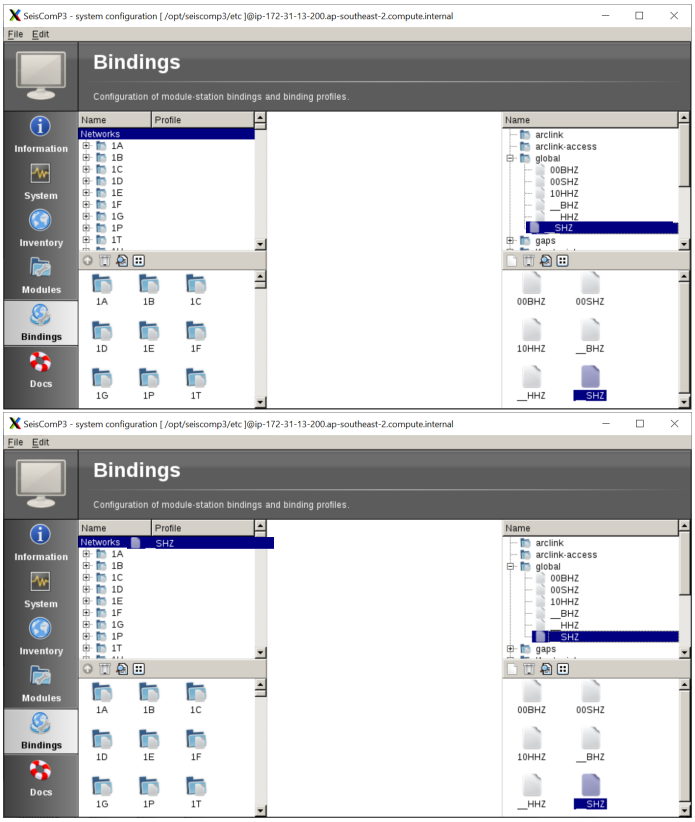

Figure 1: Drag and drop the "SHZ" binding onto the "Networks" folder.

- Run the following from the command line.

  ```
  for i in `ls *.xml`; do scdb -i $i -d
  mysql://sysop:sysop@localhost/seiscomp3 -b 1000; done
  ```

  This may take a while (several hours). Running the above in a screen session will allow a user to disconnect from the AWS during execution of the command.

D Using iLoc without SSSTs

An alternate to the SSST algorithm is to use iLoc with Regional Seismic Travel Time (RSTT) models for phase redefinition and relocation. The procedure to do so is contained below, which replaces the steps in section 5 and 6.

D.1 Run iLoc to Identify/Redefine Phases

The iLoc phase identification code is launched using an MPI-parallel script as follows.

- From a terminal on the AWS instance, run the following.

  ```
  export OPENBLAS_NUM_THREADS=1
  export PYTHONPATH=$PYTHONPATH:/home/centos/hiperseis/
  export nprocessors=(number of processors on AWS)
  ```

- Create a new folder on the AWS and change directory into it.

- Launch iloc_phase_ident.py using the following command.

```
mpirun -np $nprocessors python
↪ /hiperseis/iloc_rstt/iloc_phase_ident.py 1900-01-01T00:00:00
↪ 2022-01-01T00:00:00 ./ --update-db 1 > ./phase_ident.out
```

Upon successful completion, the output folder should contain a bunch of two-columned text files (one for each processor). The first column lists event-ids and the second column shows the iLoc convergence status for each event. Note that we expect a small percentage of events not to converge not only due to some stations metadata being missing, as discussed above, but also due to potential inconsistencies in the arrivals in the curated catalogue. The file, phase_ident.out, contains verbose output from iLoc that may be useful for debugging purposes.

D.2 Extract Updated Events

We now extract only those events for which the iLoc solutions in step D.1 converged and output them in text format. An MPI-parallel script, sc3_extract_events.py, is launched on the AWS instance as follows.

- From a terminal on the AWS instance, run the following.

```
$ export OPENBLAS_NUM_THREADS=1
$ export PYTHONPATH=$PYTHONPATH:/home/centos/hiperseis/
```

- Create a new folder on the AWS instance and change directory into it.
- Run the following in the command line. Assume that <input> is the directory from which iloc_phase_ident.py was run,

```
$ mpirun -np $nprocessors python
/home/centos/hiperseis/iloc_rstt/sc3_extract_events.py
<input> /tmp/ ensemble
```

Upon successful completion, the current folder should contain three files:

- ensemble.all.txt: all arrivals
- ensemble.p.txt: only P, Pg arrivals
- ensemble.s.txt: only S, Sg arrivals

E Plots

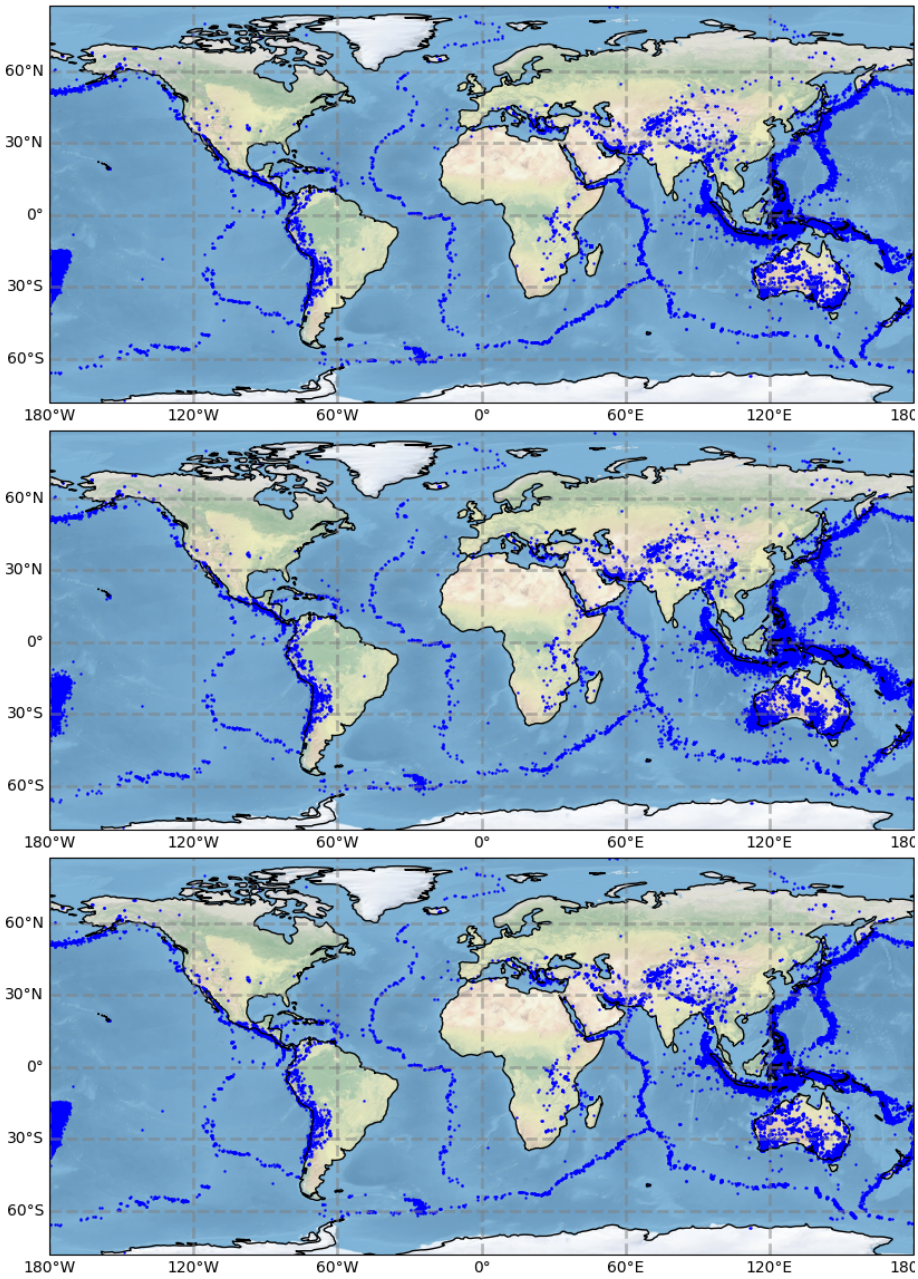

Figure 2: Reported epicentres (top), epicentres relocated using the workflow in appendix D (centre), and epicentres relocated using the SSST workflow (bottom) globally.

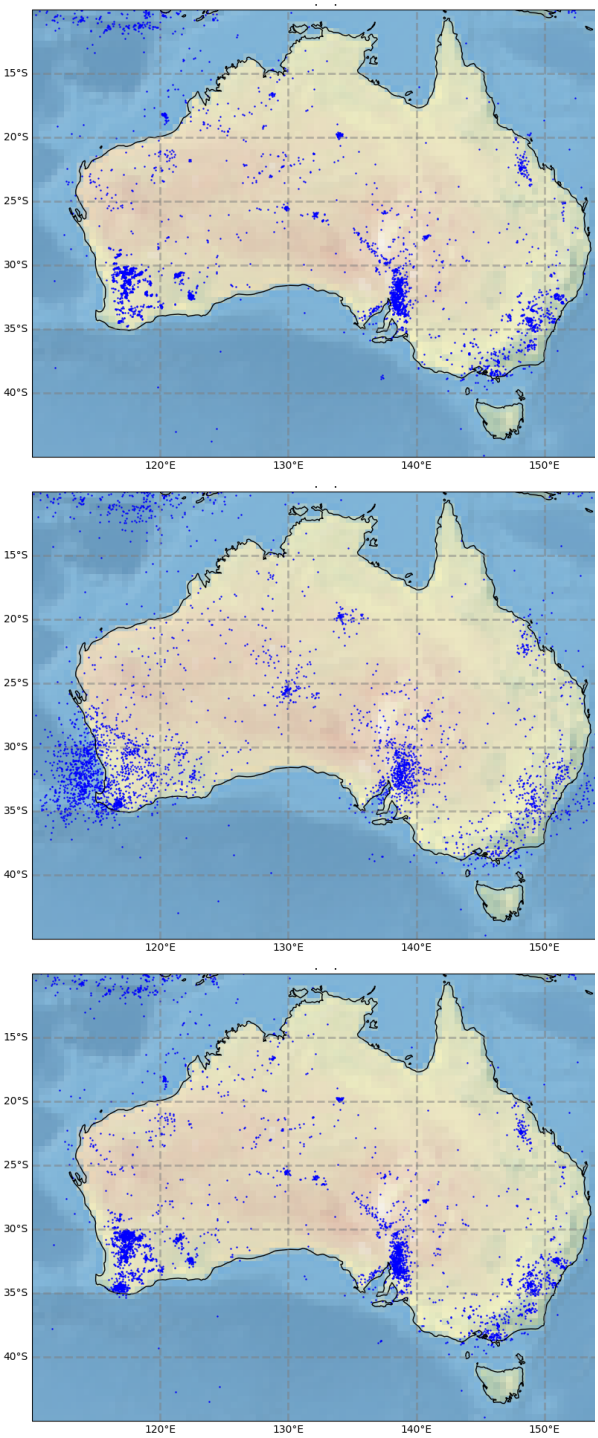

Figure 3: Reported epicentres (top), epicentres relocated using the workflow in appendix D (centre), and epicentres relocated using the SSST workflow (bottom) in Australia.

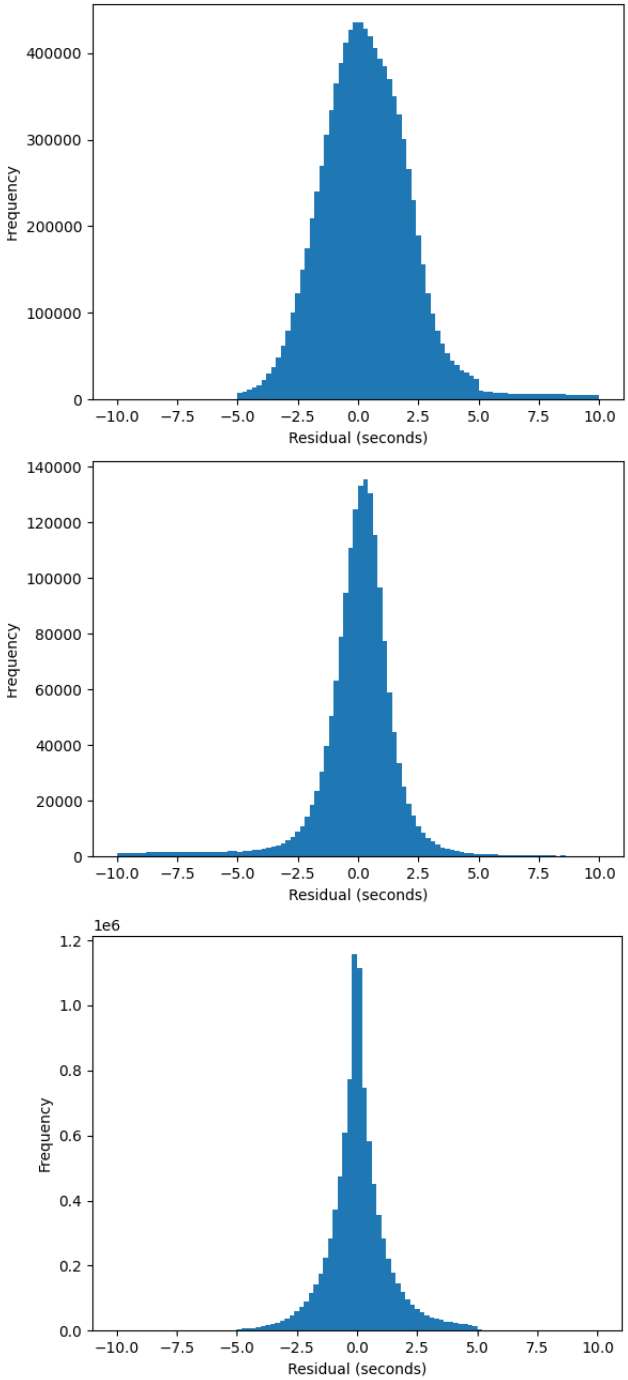

Figure 4: Reported P -wave travel time residuals (top), residuals after relocation using the workflow in appendix D (centre), and residuals after relocation using the SSST workflow (bottom) globally.

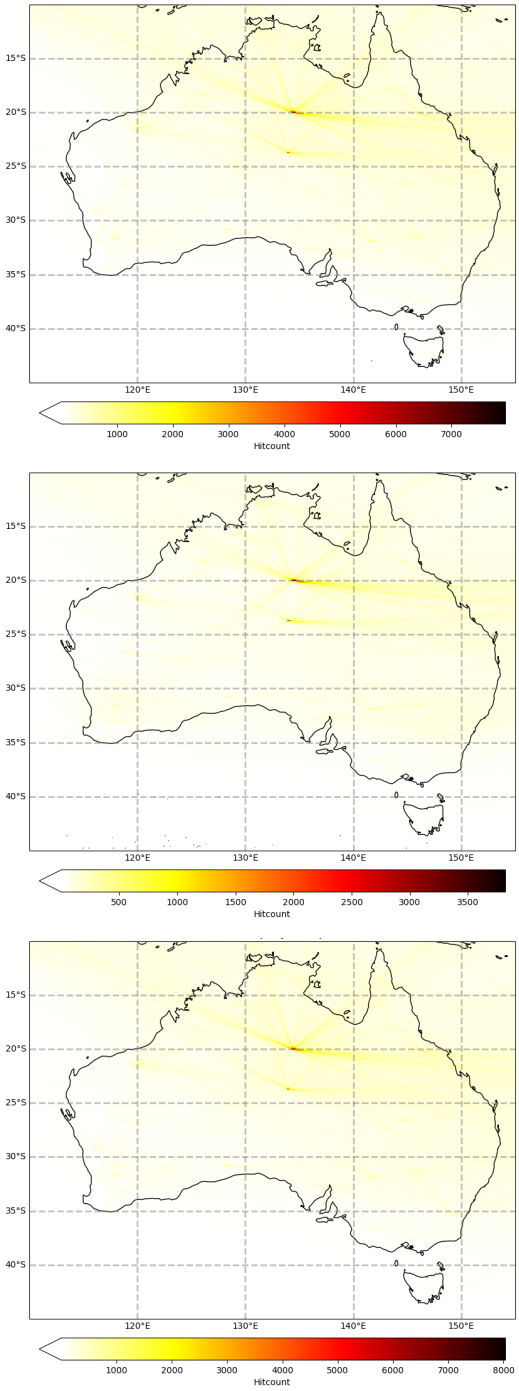

Figure 5: P-wave raypath coverage for reported picks (top), picks after applying workflow in appendix D (centre), and picks after applying SSST relocation (bottom) in the Australian region. To avoid saturation, only 1 in 10 rays are shown.