Ground motion simulation run manual (23p01)

The instructions for how to run a ground motion simulation on the NeSI systems of Maui and Mahuika are given below.

The method for running on the Kisti system Nurion is the same, however no examples are provided here.

These instructions use the automated workflow described here: Automated workflow pipelines

In code blocks on this page variables are given in <> brackets, optional parameters are given in [], examples given in {}, and comments are given in (). All script arguments are placed on a new line for clarity, they should follow the script in use.

1) Generate fault selection file

The fault selection file must first be generated.

This file will be a list of all events or faults to be considered for realisation generation.

The format of this file is two columns: The name of the event/fault followed by the number of realisations for the event/fault. The number of realisations should be suffixed by the letter 'r'

A script to generate the fault selection file from a given nhm and list of faults is available here: Simulation count calculator in Python3

2) Generate source parameter files

Once the fault selection file has been generated the source files must be created.

This step can be performed with a slurm script.

This step is to generate realisation files from either NHM of GCMT sources. Further more an uncertainty version can be given to generate realisations with uncertainty characteristics.

Notably gcmt subduction interface faults must have the arguments "--common_source_parameter tect_type SUBDUCTION_INTERFACE" added to use the Skarlatoduis magnitude-area scaling relationship.

```
python $gmsim/Pre-processing/srf_generation/source_parameter_generation/generate_realisations_from_gcmt.py
        <path to fault selection file>
        <path to gcmt file>
        <type of fault to generate> (Currently 1 or 2 for gcmt sources)
        [--n_processes <number of cores to use> (Default 1, max number of events/faults)]
        [--version <Name of perturbation version to use>]
        [--vel_mod_1d <Location of the 1d velocity model to use for srf generation>]
        [--aggregate_file <Location to place the realisation aggregation file> (Will cause a crash if this file
already exists)]
        [--source_parameter <parameter name> <Location of file containing source specific parameters>
(Repeatable)]
        [--common_source_parameter <parameter name> <parameter value>]
        [--vs30_median <Location of vs30 sigma file> (File has two columns, one of station names, the other of
vs30 values)]
        [--vs30_sigma <Location of vs30 sigma file> (Same format as the median file, except with sigma instead
of median)]
        [--cybershake_root <Location of simulation root directory>]
        [--checkpointing (Prevents recreation of previously created files)]
python $gmsim/Pre-processing/srf_generation/source_parameter_generation/generate_realisations_from_nhm.py
        <path to fault selection file>
        <path to nhm file>
        [(The same optional arguments as for gcmt shown above)]
```

eg.

```
python $gmsim/Pre-processing/srf_generation/source_parameter_generation/generate_realisations_from_gcmt.py
list1.txt GeoNet_CMT_solutions_20190418_Mw_3.5_5.0_CD_3.0_20.0_NSe_3_Quality_ChowTomo.csv 1  --vel_mod_1d
/scratch/hpc91a02/gmsim_home/VelocityModel/Mod-1D/Cant1D_v3-midQ_OneRay.1d --aggregate_file rel_agg.csv --
cybershake_root .
python $gmsim/Pre-processing/srf_generation/source_parameter_generation/generate_realisations_from_nhm.py v23p8.
txt ../NZ_FLTmodel_2010_v18p6.txt
```

For more information, see Source Modelling for GMSim

Note: This needs to be run from the simulation folder.

Note: generate_realisations_from_nhm.py doesn't need a positional type argument as only type 4 is currently available. An optional argument --type is available and prevents any non type 4 jobs from running

More information on the srf generation step is available here: Validation source perturbation

3) Install simulation

The simulation must be installed to set up all the shared parameter files and output directories.

For CyberShake runs the gmsim version root_defaults.yaml should be changed to have Ds_multiplier 0.75 and PGV_Threshold 2.0

```
python $gmsim/workflow/workflow/automation/install_scripts/install_cybershake.py
        <Path to root of simulation directory>
        <Path to fault selection file>
        [<gmsim version to use> (Defaults to 16.1) (Must exist in workflow/workflow/calculation
/gmsim_templates)]
        --no_check_vm (Required for the 21p09 version of automated workflow, allows automated VM generation)
        [--seed <Seed to use for HF> (Defaults to using a different random one for each realisation)]
        [--stat_file_path <Path to the station list to use>]
        [--extended_period (Adds additional pSA periods to IM_calc, usually used for plotting the pSA)]
        [--log_file <Path to the log file to use> (Defaults to 'cybershake_log.txt')]
        [--keep_dup_station (Keep stations that would be removed for snapping to a grid point previously
snapped to by another station)]
```

e.g.

```
python $gmsim/workflow/workflow/automation/install_scripts/install_cybershake.py . list.txt 20.4.1.4 --
no_check_vm --keep_dup_station --stat_file_path /nesi/project/nesi00213/StationInfo
/non_uniform_whole_nz_with_real_stations-hh400_v20p3_land.ll
```

This runs install in your current directory for gm sim version 20.4.1.4 keeping duplicate stations for the 20p3 non_uniform station list.
Update: --no_check_vm option has been removed.

4) Run pre-processing steps

This section covers the generation of source model and velocity model files, along with installing each realisation.

For those familiar with the 20p07 run manual, Velocity models are generated in the same manner as simulations are run.

The main point to note here is that a separate task configuration file should be used in order to only run the pre-processing steps.

(Cybershake runs: Ensure you have updated the PGV threshold and DS multiplier in the root params file, check Cybershake versioning Table, usually PGV = 2 DS = 0.75)

(Note: When running pertubations you may run into issues for them taking forever to complete when having a high n_runs value > 100, to solve this reduce n_runs)

```
python $gmsim/workflow/workflow/automation/execution_scripts/run_cybershake.py
    <Path to root of simulation directory>
        [<Path to the task configuration file to be used> (Defaults to a file in the slurm_gm_workflow
repository which runs EMOD3D, HF, BB, IM_calc and clean_up on all realisations)]
        [--sleep_time <How long each sub script should sleep for before restarting> (Defaults to 5, generally
shouldn't need to be changed)]
        [--n_max_retries <How many times an individual task should be attempted before requiring user
intervention>]
        [--n_runs <The number of jobs that can run simultaneously> (Takes either 1 value, or a number of values
equal to the number of available HPCs]
        [--log_folder <Location log files should be placed>]
        [--debug (Print debug log messages to the terminal)]
```

Example:

```
python $gmsim/workflow/workflow/automation/execution_scripts/run_cybershake.py . gmsim/workflow/workflow
/examples/task_config_prepro.yaml
```

Update 06 Dec 2023 $USER argument is no longer needed

5) Estimate corehours

After velocity models have been generated core hour estimation should be run to determine the amount of core hours required to run the simulation.

```
python $gmsim/workflow/workflow/automation/estimation/estimate_cybershake.py
        <Path to the simualation root dir>
        [--fault_selection <Path to the fault selection file> (Ignored if --runs_dir is given)]
        [--output <Path to save the output> (Does not save, only displays if not given)]
        [--verbose (Show estimated core hours for each fault, not just the whole simulation)]
```

If the number of core hours required is greater than 1000, then a core hour request should be submitted via slack.

6) Run simulation steps

Finally the simulation can be run

```
python $gmsim/workflow/workflow/automation/execution_scripts/run_cybershake.py
    <Path to root of simulation directory>
        <The user name to be used with the HPC scheduler>
        [<Path to the task configuration file to be used> (Defaults to a file in the slurm_gm_workflow
repository which runs EMOD3D, HF, BB, IM_calc and clean_up on all realisations)]
        [--sleep_time <How long each sub script should sleep for before restarting> (Defaults to 5, generally
shouldn't need to be changed)]
        [--n_max_retries <How many times an individual task should be attempted before requiring user
intervention>]
        [--n_runs <The number of jobs that can run simultaneously> (Takes either 1 value, or a number of values
equal to the number of available HPCs]
        [--log_folder <Location log files should be placed>]
        [--debug (Print debug log messages to the terminal)]
```

Details of the configuration file contents are available here: Auto submit wrapper

7) Monitor simulation

The run script will provide an overview of the progress of the simulation.

To watch the database state and your jobs in the Maui and Mahuika scheduler queues the following command can be used

```
watch "python $gmsim/workflow/workflow/automation/execution_scripts/query_mgmt_db.py <path to simulation root
directory> --config <path to config file>; squeue -u $USER -M maui; squeue -u $USER -M mahuika"
```

If more than 40 tasks are set to be run, then it is advisable to use the argument "–mode count" with query_mgmt_db.py

```
python $gmsim/workflow/workflow/automation/execution_scripts/query_mgmt_db.py
    <Path to simulation root directory>
        [<Name of fault or event to inspect> (All events/faults are shown otherwise)]
        [--config <Path to task configuration file used to call run_cybershake.py>]
        [--mode {error,count,todo,retry_max,detailed_count} [{error,count,todo,retry_max,detailed_count} ...]
(Multiple values can be given, some modes work in combination with each other)]
        [--mode-help (Flag that displays information about the different modes)]
```