# SeisSol_ParamStudies Manual

iris.christadler@lmu.de, M.Marchandon@lmu.de

https://github.com/christadler/SeisSol_ParamStudies

This is a bunch of generic scripts and templates that allows you to use SeisSol for parameter studies. It was developed for DT-Geo (DTC-E4, Block 3) to generate a catalogue for AltoTiberina

# How to Use the SeisSol_ParamStudies Workflow

1. Set-up one test case (Examples/)
   a. Use the right directory structure
   b. Adapt and test the slurm job file
2. Decide on the parameters that should change (Inputs/parameter_study_list.csv)
   a. Create a csv file for all runs
3. Create .jinja Templates (Inputs/Templates/)
   a. for all input files that should use parameters from the .csv file
   b. all input files that don't change should be copied to Inputs/Files/
4. Adapt Scripts/template_parameters.py
   a. Adapt **tp_param_list, tp_list, tp_mapping** & add **user_name** + user info below
5. Finally adapt Inputs/Templates/SeisSol_slurm_JobFarm.slurm.jinja (with help)

… now run Scripts/main.py to generate one folder for each run in Outputs/,
copy the Outputs/ directory to SuperMucNG and submit the slurm_xxx.scripts

# 1. Set-up Test Case (and copy it to the Examples/ dir)

On SuperMucNG set up your file structure:
(In your *workdir*, Output should go to *scratch*)

- inputs/
  - <xyz>.yaml
  - parameters.par
- mesh/
- seissol_bin/
  - e.g. SeisSol_Release_dskx_4_elastic
- JobSubmission.slurm

inputs/parameters.par exemple:

```
MeshFile =
'../../mesh/mesh_topo1km_7_faults_smoothed_v3_small
z'

OutputFile =
'/hppfs/scratch/0A/di35ban/AltoTiberina/outputs/000
1/0001'
```

Slurm script example:

```
cd <path in work dir>

mkdir -p <path in scratch for output>

srun -n $SLURM_NTASKS
../seissol_bin/<binary file> parameters.par

# These files are used for post-processing

myfile=<path in scratch for output>/<name>-surface
e.xdmf

echo "Myfile: $myfile"

cd <path in scratch for output>

<post processing>
```

# 2. Build Inputs/parameter_study_list.csv

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | id | nucleation | d_c | y | mu_s | R | xha_x | xha_y | xha_z |
| 2 | 0 | 1 | 0.4 | 0.6 | 0.35 | 0.65 | 295214.0 | 4817590.0 | -8238.0 |
| 3 | 1 | 1 | 0.4 | 0.6 | 0.35 | 0.7 | 295214.0 | 4817590.0 | -8238.0 |
| 4 | 2 | 1 | 0.4 | 0.6 | 0.35 | 0.75 | 295214.0 | 4817590.0 | -8238.0 |
| 5 | 3 | 1 | 0.4 | 0.6 | 0.35 | 0.8 | 295214.0 | 4817590.0 | -8238.0 |
| 6 | 4 | 1 | 0.4 | 0.6 | 0.35 | 0.85 | 295214.0 | 4817590.0 | -8238.0 |
| 7 | 5 | 1 | 0.4 | 0.6 | 0.4 | 0.7 | 295214.0 | 4817590.0 | -8238.0 |
| 8 | 6 | 1 | 0.4 | 0.6 | 0.4 | 0.75 | 295214.0 | 4817590.0 | -8238.0 |
| 9 | 7 | 1 | 0.4 | 0.6 | 0.4 | 0.8 | 295214.0 | 4817590.0 | -8238.0 |
| 10 | 8 | 1 | 0.4 | 0.6 | 0.4 | 0.85 | 295214.0 | 4817590.0 | -8238.0 |
| 11 | 9 | 1 | 0.4 | 0.6 | 0.45 | 0.75 | 295214.0 | 4817590.0 | -8238.0 |
| 12 | 10 | 1 | 0.4 | 0.6 | 0.45 | 0.8 | 295214.0 | 4817590.0 | -8238.0 |
| 13 | 11 | 1 | 0.4 | 0.6 | 0.45 | 0.85 | 295214.0 | 4817590.0 | -8238.0 |
| 14 | 12 | 1 | 0.4 | 0.6 | 0.45 | 0.9 | 295214.0 | 4817590.0 | -8238.0 |
| 15 | 13 | 1 | 0.4 | 0.6 | 0.5 | 0.7 | 295214.0 | 4817590.0 | -8238.0 |
| 16 | 14 | 1 | 0.4 | 0.6 | 0.5 | 0.8 | 295214.0 | 4817590.0 | -8238.0 |
| 17 | 15 | 1 | 0.4 | 0.6 | 0.5 | 0.85 | 295214.0 | 4817590.0 | -8238.0 |
| 18 | 16 | 1 | 0.4 | 0.6 | 0.5 | 0.9 | 295214.0 | 4817590.0 | -8238.0 |
| 19 | 17 | 1 | 0.4 | 0.6 | 0.55 | 0.7 | 295214.0 | 4817590.0 | -8238.0 |
| 20 | 18 | 1 | 0.4 | 0.6 | 0.55 | 0.8 | 295214.0 | 4817590.0 | -8238.0 |
| 21 | 19 | 1 | 0.4 | 0.6 | 0.55 | 0.85 | 295214.0 | 4817590.0 | -8238.0 |

**Example from AltoTiberina:**

Each line represents one run.

Column A holds the “id”, starting with zero. It will be used as identifier/”prefix”.

The following columns are named according to the parameters of the templates.

# 3a. Create .jinja Templates in Inputs/Templates
# 3b. Copy all other Input Files to Inputs/Files/

To create the .jinja Template, just replace the variable with {{<var_name>}}

Examples:

```
components: !FunctionMap
  map:
    sig_zz: |
     return 2670.0*(1.0-{{y}})*9.8*min(-200.0,z);
```

sig_zz.yaml.jinja

```
!LuaMap
 returns: [forced_rupture_time]
 function: |
    function f (x)
      xha = { {{xha_x}},{{xha_y}},{{xha_z}} }
```

forced_rupture_time.yaml.jinja

```
    function f (x)
     R = {{R}}
     return {
       S = (1.0/R-1.0)
     }
     end
- !GroupFilter
  # This is for the Gubbio fault
  groups: [66,67,68,69,70,71]
  components: !LuaMap
    returns: [S]
    function: |
     function f (x)
      R = {{R}}
      return {
        S = (1.0/R-1.0)
      }
```

initial_stress.yaml.jinja

# 4. Adapt Scripts/template_parameters.py

```
tp_param_list = ["nucleation", "d_c", "y", "mu_s", "R"]

# list of all jinja template files (except slurm tmpl for stage 2)
# to be filled by user
# stored in dir "../Input/Templates/*SeisSol_slurm_JobFarm.slurm.jinja"
# "AltoTiberina_forced_rupture_time.yaml" (xha) is currently not used
tp_list = ["AltoTiberina_fault.yaml",
           "AltoTiberina_initial_stress.yaml",
           "AltoTiberina_sig_zz.yaml",
           "AltoTiberina_forced_rupture_time.yaml"]

# mapping of parameters to jinja templates
# to be filled by user
# "AltoTiberina_forced_rupture_time.yaml" : ["xha"],
tp_mapping = { "AltoTiberina_fault.yaml" : ["mu_s", "d_c"],
               "AltoTiberina_initial_stress.yaml": ["R"],
               "AltoTiberina_sig_zz.yaml": ["y"],
               "AltoTiberina_forced_rupture_time.yaml": ["xha_x", "xha_y", "xha_z"]
               }
```

template_parameters.py

**tp_param_list**
all parameters/columns
from the .csv-file

**tp_list**
all jinja templates
that should be used

**tp_mapping**
mapping of parameters to
jinja templates

**user_name**
(and choose TEST or PRODUCTION)
**tp_slurm_account**
**tp_slurm_email**
**tp_output_file_dir**

# 5. Finally adapt Inputs/Templates/SeisSol_slurm_JobFarm.slurm.jinja

should not be necessary for testing the AltoTiberina catalogue

# … now run Scripts/main.py to generate one folder for each run in Outputs/

copy the Outputs/ directory to SuperMucNG, add the mesh and .asagi-files (as described in slide “Set-Up Test Case” and submit the slurm_xxx.scripts