February 20th-25th, 2011

SeisSol2D Workflow

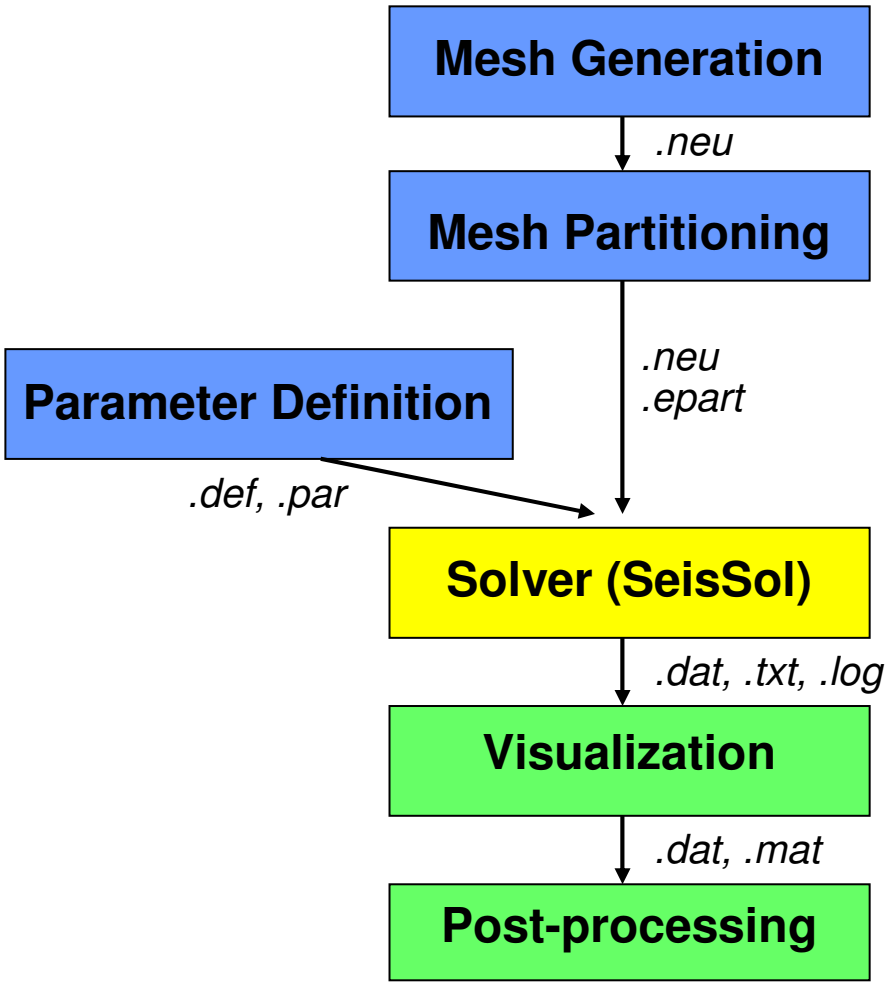
Christian Pelties & Martin Käser

Department of Earth and Environmental Sciences, LMU München, Germany

The SeisSol2D Source Code

• numerical scheme:	discontinuous Galerkin finite-elements
• programming language:	mainly FORTRAN 90
	few FORTRAN 77 subroutines
• number of subroutines:	304
• number of files:	54
• number of directories:	3
• number of lines of source code:	~ 50.000
• number of developers:	~ 6
• pre-processing:	mesh generation, mesh partitioning, parameterization, (e.g. material distribution, …)
• post-processing:	visualization of wave field (snapshots, seismograms), peak ground motion, stress distribution, seismic signal processing, …

The SeisSol Workflow



Commercial software: - Gambit (Fluent)
- ICEM CFD (Ansys)

Graph partitioner: METIS (usage via MATLAB script)

Material definition, Source/receivers, numerical parameters, boundary conditions, etc. (ASCII)

Computation (Fortran 90 using MPI)
Run-time control: StdOut, IRREGULARITIES, etc.

Seismograms, Wave field snapshots (MATLAB scripts, Tecplot)

Signal processing: filtering, etc. (MATLAB)

Discretization

What numerical methods do you know or have you heard of ?

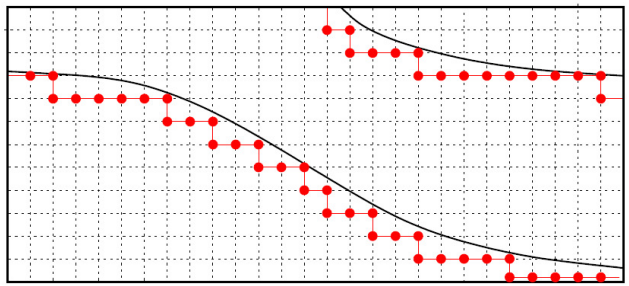
Finite Differences (FD), Finite Elements (FE), Finite Volumes (FV), etc.

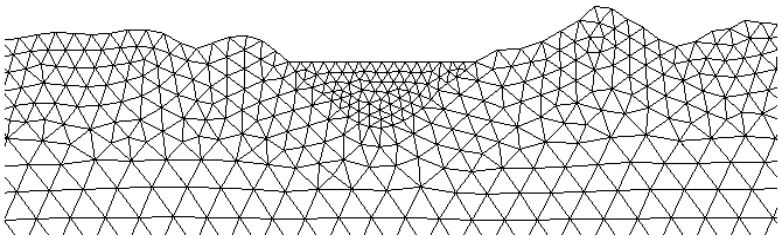
What do numerical method usually require ?

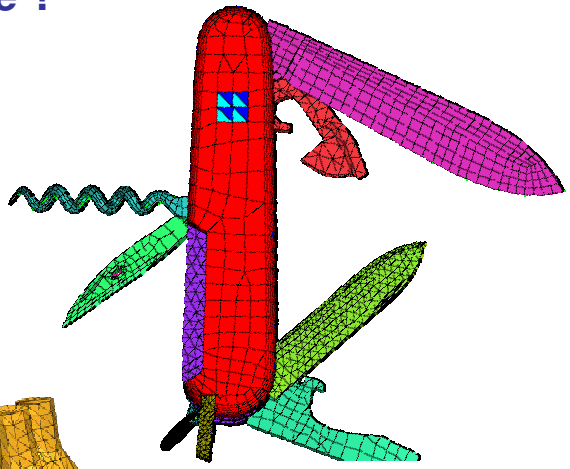
points or particles, regular grids, unstructured meshes, etc.

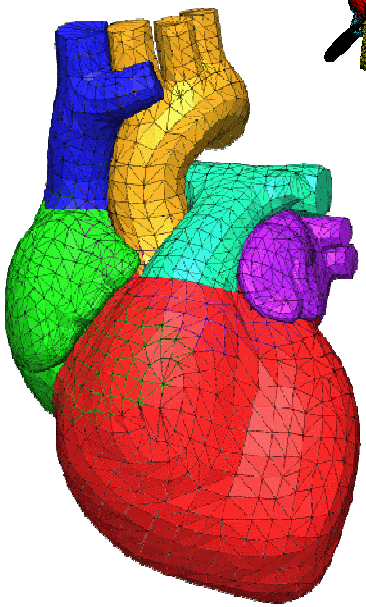
What is an *"unstructured mesh"* ?

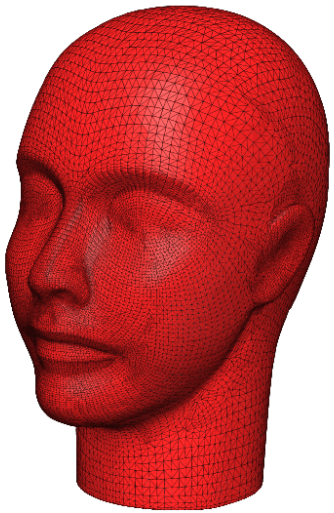
A mesh where element indices do not show a logical structure







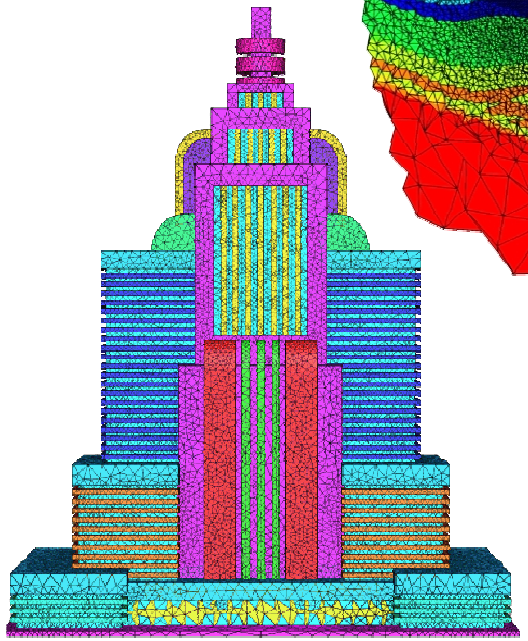


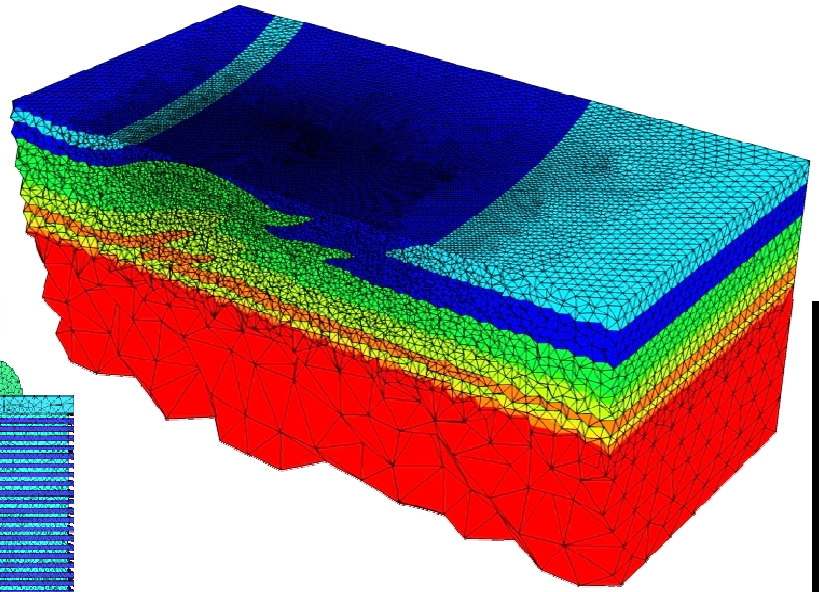


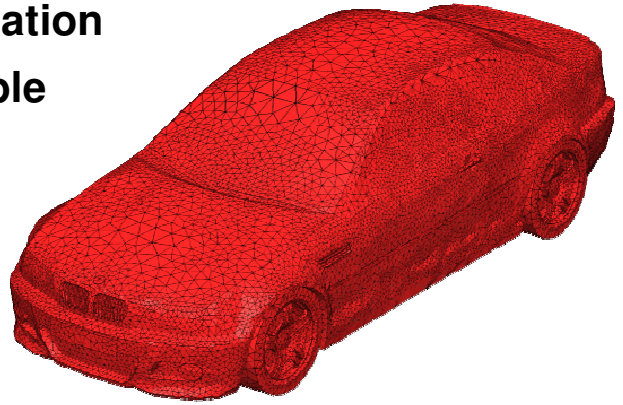
Discretization

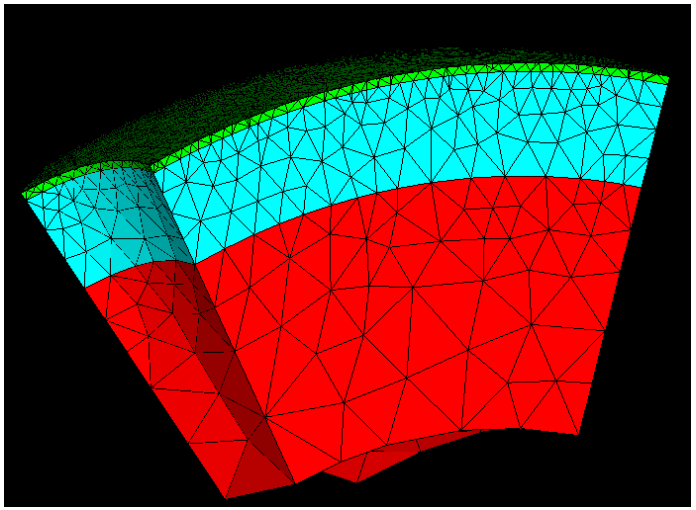
The discretization subdivides a continuous physical model into a set of points, elements, control volumes, cells, etc.

- ➔ **numerical methods approximate functions discretely on these points or elements**
- ➔ **the finer the mesh, the more accurate the approximation**
- ➔ **unstructured meshes are geometrically more flexible**





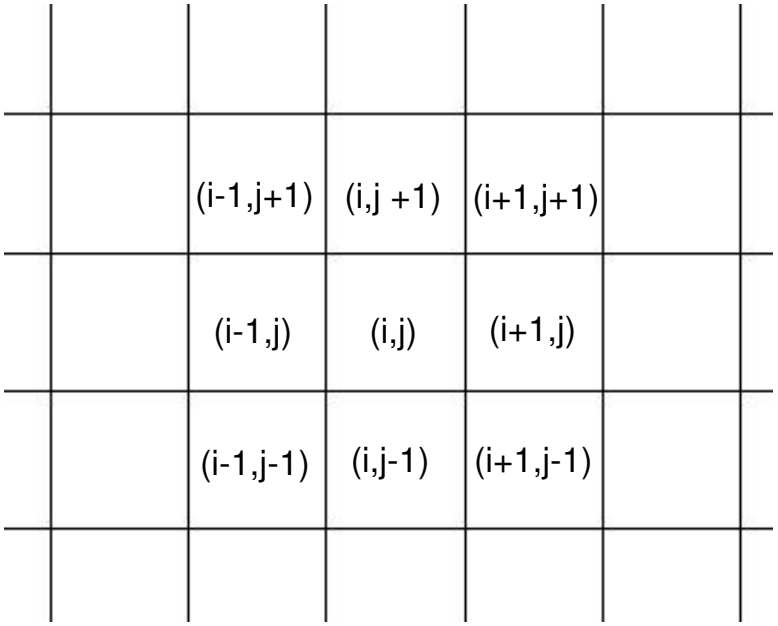




Discretization of a physical model in 2D

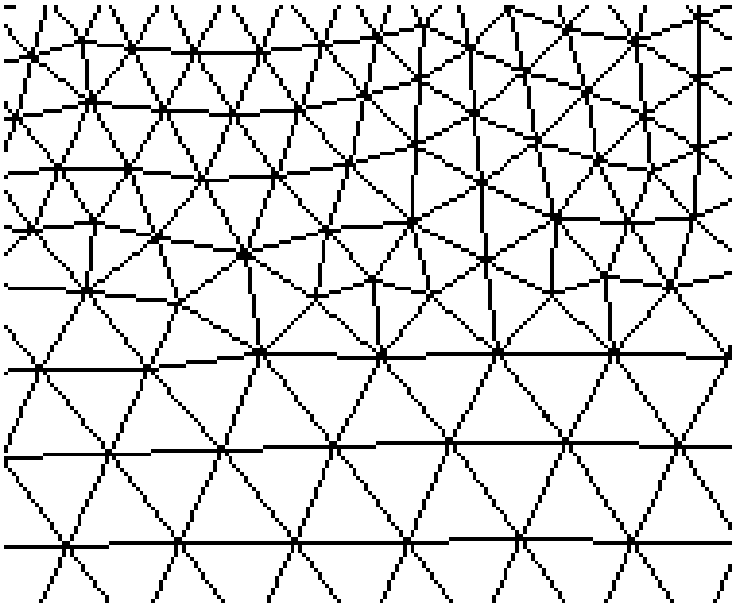
Discretization (= the process of Mesh Generation) can still be simple in 2D

➔ we might still be able to use indices in a logical structure



➔ In structured meshes it is easy to know the neighbour elements to pass information

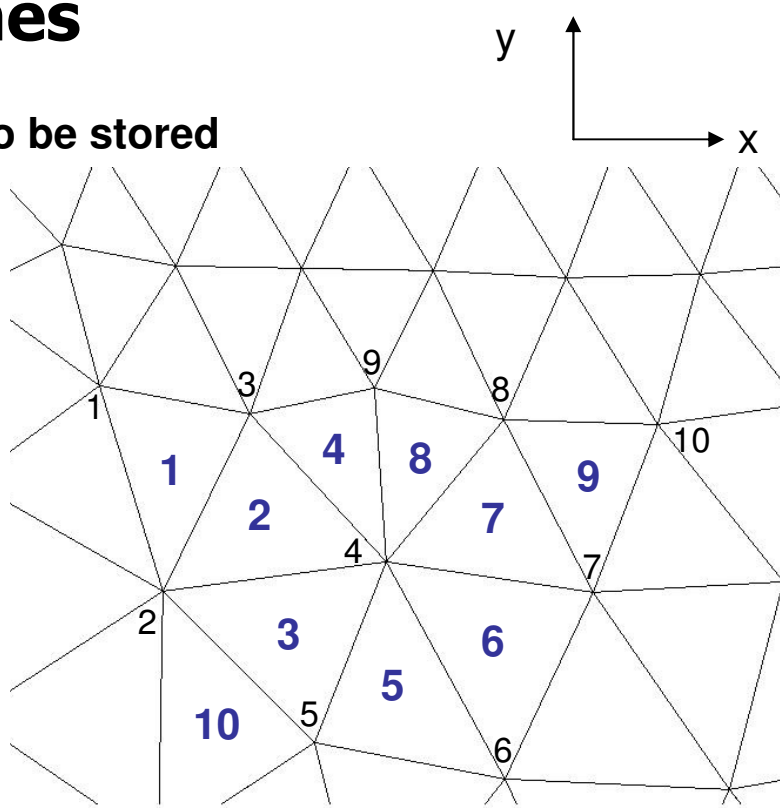
But how do you distribute indices in this mesh and identify your neighbour ???



Connectivity matrix of meshes

➔ **in unstructured meshes, all vertices have to be stored**

➔ **the connectivity matrix defines elements**



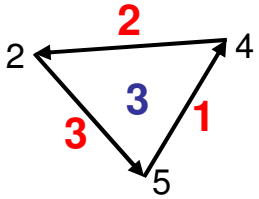
list of vertices

1	x_1	y_1
2	x_2	y_2
3	x_3	y_3
4	x_4	y_4
…	…	…

list of elements
(= connectivity matrix)

1	1	2	3
2	2	4	3
3	5	4	2
4	4	9	3
5	5	6	4
6	6	7	4
7	4	7	8
8	9	4	8
9	8	7	10
…	…	…	…

Note : the **numbering of vertices** has to be oriented, e.g. here **counter-clockwise !!!**



➔ this gives the edges that define the neighbours

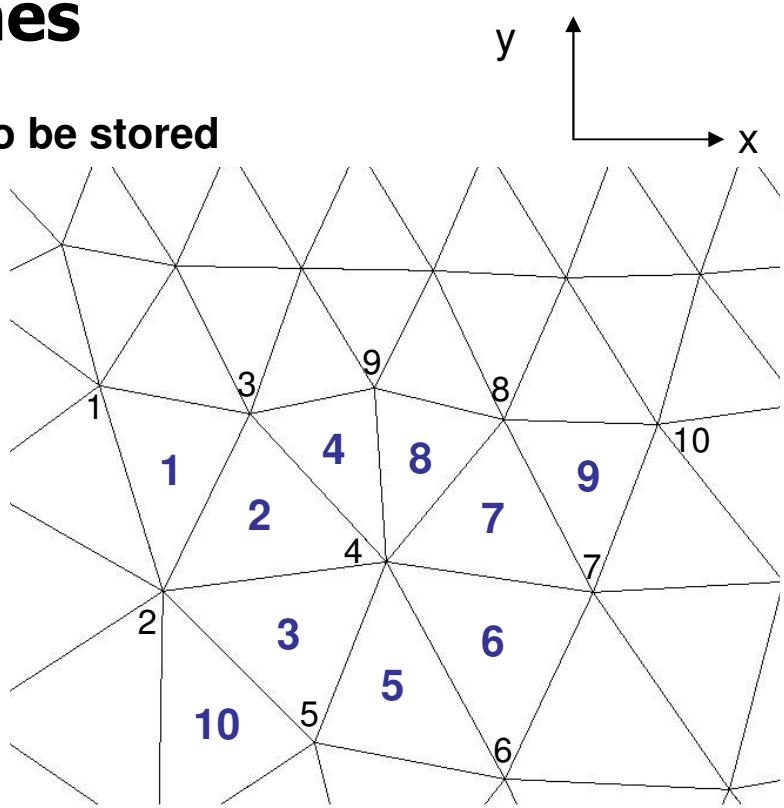
list of neighbours

1	…	2	…
2	3	4	1
3	5	2	10
…	…	…	…
7	?	?	?
…	…	…	…

Connectivity matrix of meshes

➔ **in unstructured meshes, all vertices have to be stored**

➔ **the connectivity matrix defines elements**



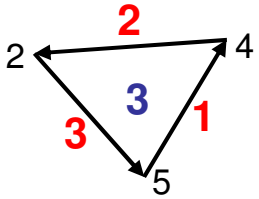
list of vertices

1	x_1	y_1
2	x_2	y_2
3	x_3	y_3
4	x_4	y_4
…	…	…

list of elements
(= connectivity matrix)

1	1	2	3
2	2	4	3
3	5	4	2
4	4	9	3
5	5	6	4
6	6	7	4
7	4	7	8
8	9	4	8
9	8	7	10
…	…	…	…

Note : the **numbering of vertices** has to be oriented, e.g. here **counter-clockwise !!!**



➔ this gives the edges that define the neighbours

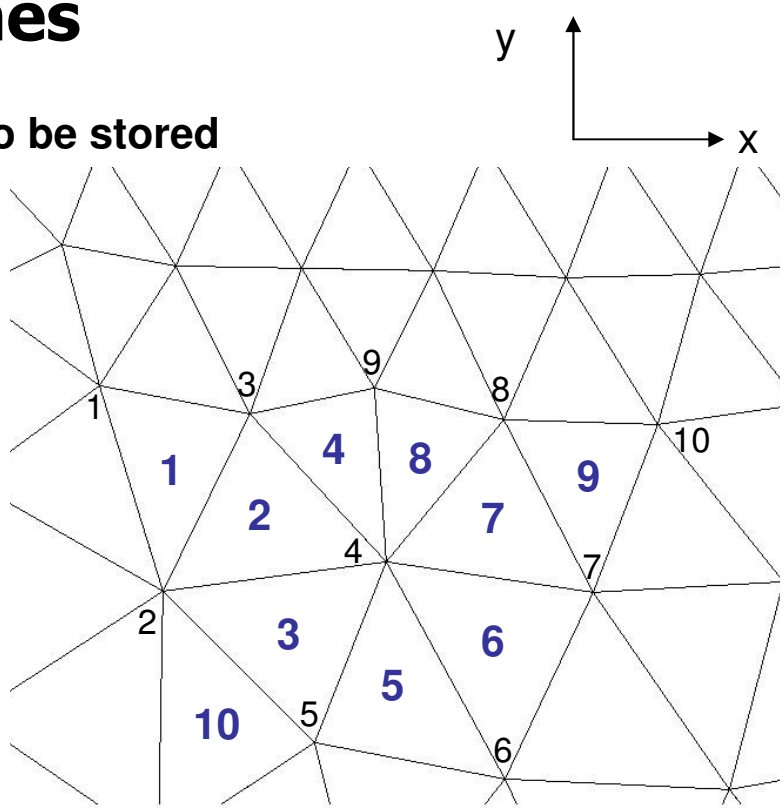
list of neighbours

1	…	2	…
2	3	4	1
3	5	2	10
…	…	…	…
7	6	9	8
…	…	…	…

Connectivity matrix of meshes

➔ **in unstructured meshes, all vertices have to be stored**

➔ **the connectivity matrix defines elements**



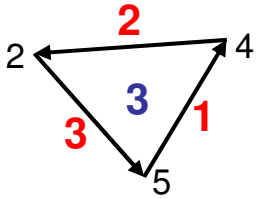
list of vertices

1	x_1	y_1
2	x_2	y_2
3	x_3	y_3
4	x_4	y_4
…	…	…

list of elements
(= connectivity matrix)

1	1	2	3
2	2	4	3
3	5	4	2
4	4	9	3
5	5	6	4
6	6	7	4
7	4	7	8
8	9	4	8
9	8	7	10
…	…	…	…

Note : the **numbering of vertices** has to be oriented, e.g. here **counter-clockwise !!!**



➔ this gives the edges that define the neighbours

list of neighbours

1	…	2	…
2	3	4	1
3	5	2	10
…	…	…	…
7	6	9	8
…	…	…	…

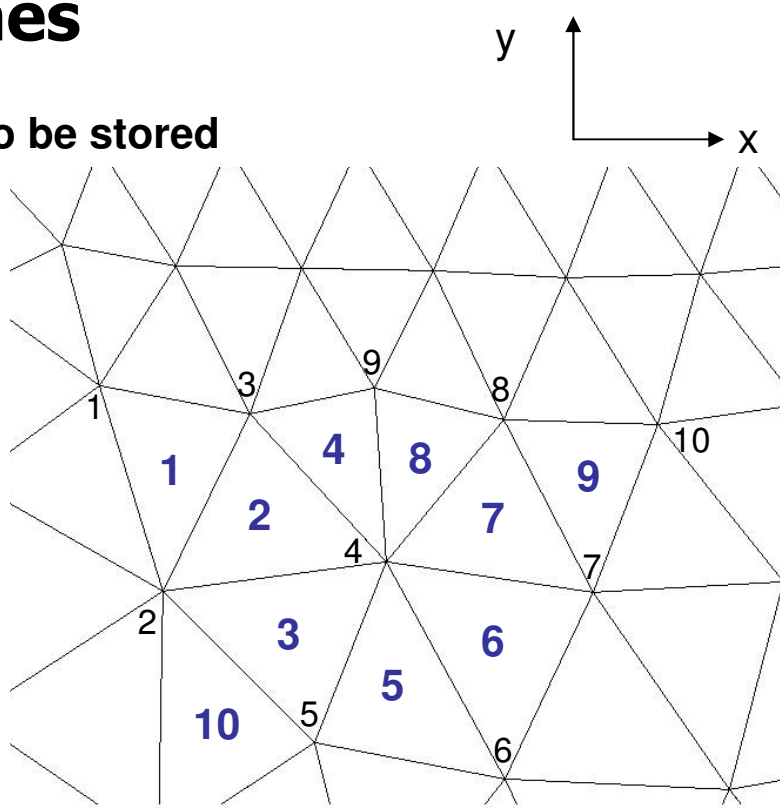
list of neighbour sides

1	…	3	…
2	2	3	2
3	3	1	…
…	…	…	…
7	?	?	?
…	…	…	…

Connectivity matrix of meshes

➔ **in unstructured meshes, all vertices have to be stored**

➔ **the connectivity matrix defines elements**



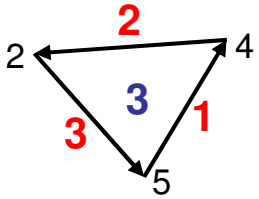
list of vertices

1	x_1	y_1
2	x_2	y_2
3	x_3	y_3
4	x_4	y_4
…	…	…

list of elements
(= connectivity matrix)

1	1	2	3
2	2	4	3
3	5	4	2
4	4	9	3
5	5	6	4
6	6	7	4
7	4	7	8
8	9	4	8
9	8	7	10
…	…	…	…

Note : the **numbering of vertices** has to be oriented, e.g. here **counter-clockwise !!!**



➔ this gives the edges that define the neighbours

list of neighbours

1	…	2	…
2	3	4	1
3	5	2	10
…	…	…	…
7	6	9	8
…	…	…	…

list of neighbour sides

1	…	3	…
2	2	3	2
3	3	1	…
…	…	…	…
7	2	1	2
…	…	…	…

The Mesh File .neu

```
CONTROL INFO 2.3.16
** GAMBIT NEUTRAL FILE
basin
PROGRAM:            Gambit    VERSION:  2.3.16
 Jan 2011
     NUMNP     NELEM     NGRPS    NBSETS     NDFCD     NDFVL
       909      1686         2         3         2         2
ENDOFSECTION
   NODAL COORDINATES 2.3.16
         1 -1.99632899778e+003 -1.93267624127e-012
         2  9.85389565112e+003  1.13686837722e-013
         3 -1.74944931759e+003  0.00000000000e+000
         4 -1.50256963740e+003  0.00000000000e+000
                 …
       907 -7.78723410147e+003 -3.07694414949e+003
       908 -8.49870831127e+003 -3.55421101275e+003
       909 -6.52218493125e+003 -3.91849972757e+003
ENDOFSECTION
      ELEMENTS/CELLS 2.3.16
        1  3  3        3       1      51
        2  3  3        2      49     100
        3  3  3        2     100      50
        4  3  3       50     100      98
                 …
  1685  3  3      903     906     900
  1686  3  3      907     903     908
ENDOFSECTION
```

Header information

The Mesh File .neu

```
CONTROL INFO 2.3.16
** GAMBIT NEUTRAL FILE
basin
PROGRAM:                Gambit     VERSION:  2.3.16
 Jan 2011
     NUMNP     NELEM     NGRPS    NBSETS     NDFCD     NDFVL
       909      1686         2         3         2         2
ENDOFSECTION
   NODAL COORDINATES 2.3.16
         1 -1.99632899778e+003 -1.93267624127e-012
         2  9.85389565112e+003  1.13686837722e-013
         3 -1.74944931759e+003  0.00000000000e+000
         4 -1.50256963740e+003  0.00000000000e+000
                 ...
       907 -7.78723410147e+003 -3.07694414949e+003
       908 -8.49870831127e+003 -3.55421101275e+003
       909 -6.52218493125e+003 -3.91849972757e+003
ENDOFSECTION
      ELEMENTS/CELLS 2.3.16
         1  3  3         3       1      51
         2  3  3         2      49     100
         3  3  3         2     100      50
         4  3  3        50     100      98
                 ...
      1685  3  3       903     906     900
      1686  3  3       907     903     908
ENDOFSECTION
```

Nodal coordinates

The Mesh File .neu

```
CONTROL INFO 2.3.16
** GAMBIT NEUTRAL FILE
basin
PROGRAM:                Gambit     VERSION:  2.3.16
 Jan 2011
     NUMNP     NELEM     NGRPS    NBSETS     NDFCD     NDFVL
       909      1686         2         3         2         2
ENDOFSECTION
   NODAL COORDINATES 2.3.16
         1 -1.99632899778e+003 -1.93267624127e-012
         2  9.85389565112e+003  1.13686837722e-013
         3 -1.74944931759e+003  0.00000000000e+000
         4 -1.50256963740e+003  0.00000000000e+000
                 …
       907 -7.78723410147e+003 -3.07694414949e+003
       908 -8.49870831127e+003 -3.55421101275e+003
       909 -6.52218493125e+003 -3.91849972757e+003
ENDOFSECTION
      ELEMENTS/CELLS 2.3.16
        1  3  3        3      1     51
        2  3  3        2     49    100
        3  3  3        2    100     50
        4  3  3       50    100     98
                  …
     1685  3  3      903    906    900
     1686  3  3      907    903    908
ENDOFSECTION
```

Elements
= connectivity

The Mesh File .neu

```
CONTROL INFO 2.3.16
** GAMBIT NEUTRAL FILE
basin
PROGRAM:              Gambit     VERSION:  2.3.16
 Jan 2011
     NUMNP     NELEM     NGRPS    NBSETS     NDFCD     NDFVL
       909      1686         2         3         2         2
ENDOFSECTION
   NODAL COORDINATES 2.3.16
         1 -1.99632899778e+003 -1.93267624127e-012
         2  9.85389565112e+003  1.13686837722e-013
         3 -1.74944931759e+003  0.00000000000e+000
         4 -1.50256963740e+003  0.00000000000e+000
                 …
     907 -7.78723410147e+003 -3.07694414949e+003
     908 -8.49870831127e+003 -3.55421101275e+003
     909 -6.52218493125e+003 -3.91849972757e+003
ENDOFSECTION
      ELEMENTS/CELLS 2.3.16
       1  3  3       3      1     51
       2  3  3       2     49    100
       3  3  3       2    100     50
       4  3  3      50    100     98
                  …
    1685  3  3     903    906    900
    1686  3  3     907    903    908
ENDOFSECTION
```

Elements
= connectivity

Element identifier
(not important)

The Mesh File .neu

```
ELEMENT GROUP 2.3.16
GROUP:        1 ELEMENTS:      300 MATERIAL:        4 NFLAGS:        1
                basin
     0
     1     2     3     4     5     6     7     8     9    10
    11    12    13    14    15    16    17    18    19    20
                …
   281   282   283   284   285   286   287   288   289   290
   291   292   293   294   295   296   297   298   299   300
ENDOFSECTION
      ELEMENT GROUP 2.3.16
GROUP:        2 ELEMENTS:     1386 MATERIAL:        4 NFLAGS:        1
                bedrock
     0
   301   302   303   304   305   306   307   308   309   310
   311   312   313   314   315   316   317   318   319   320
                …
  1671  1672  1673  1674  1675  1676  1677  1678  1679  1680
  1681  1682  1683  1684  1685  1686
ENDOFSECTION
```

Elements in basin

The Mesh File .neu

```
ELEMENT GROUP 2.3.16
GROUP:        1 ELEMENTS:      300 MATERIAL:        4 NFLAGS:        1
                basin
     0
     1     2     3     4     5     6     7     8     9    10
    11    12    13    14    15    16    17    18    19    20
               …
   281   282   283   284   285   286   287   288   289   290
   291   292   293   294   295   296   297   298   299   300
ENDOFSECTION
     ELEMENT GROUP 2.3.16
GROUP:        2 ELEMENTS:     1386 MATERIAL:        4 NFLAGS:        1
                bedrock
     0
   301   302   303   304   305   306   307   308   309   310
   311   312   313   314   315   316   317   318   319   320
               …
  1671  1672  1673  1674  1675  1676  1677  1678  1679  1680
  1681  1682  1683  1684  1685  1686
ENDOFSECTION
```

Elements in bedrock

The Mesh File .neu

```
BOUNDARY CONDITIONS 2.3.16
                     106     1    18     0     6
      928   3   2
      946   3   2
            ...
      948   3   3
      934   3   3
ENDOFSECTION
BOUNDARY CONDITIONS 2.3.16
                     105     1    40     0     6
      932   3   3
      942   3   3
            ...
      940   3   2
      930   3   2
ENDOFSECTION
BOUNDARY CONDITIONS 2.3.16
                     101     1    72     0     6
      927   3   1
      944   3   3
            ...
      938   3   2
      933   3   1
ENDOFSECTION
```

Boundary conditions

Identifier of boundary element type:

101 = free surface	boundary
102 = non-conforming	boundary
103 = dynamic rupture	boundary
104 = inflow	boundary
105 = absorbing	boundary
106 = periodic	boundary

Index of boundary element

Index of local side of the boundary element that carries the boundary condition

Mesh Partitioning with METIS:

You can get the mesh partitioner METIS as free-ware from:
http://glaros.dtc.umn.edu/gkhome/views/metis

After installation you have to extract the connectivity matrix from the mesh file (e.g. **basin.neu**) and put a simple header to identify the number of element and the element type:

1 = triangle
4 = quadrilateral

and save it as a METIS file, e.g. **basin.met**

basin.met

```
1686   1
   3      1     51
   2     49    100
   2    100     50
  50    100     98
   ...
 903    906    900
 907    903    908
```

First transform the mesh to a graph via the command

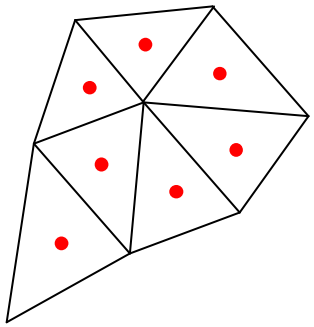
mesh2dual basin.met

Then partition the graph into the desired number **np** of processors (= subdomains) via the command

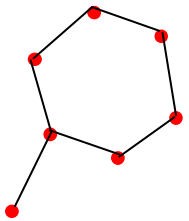
pmetis basin.met.dgraph np

and get the resulting file

basin.met.epart.np



mesh



graph

The Parameter File .par

The parameter file (typically with suffix **.par**) contains 12 blocks:

- SeisSol Version
- Equations
- Initial Condition
- Boundaries
- Source Terms
- Sponge Layer
- Mesh
- Discretization
- Output
- Abort Criteria
- Analysis of Data
- Debugging Modus

All simulation parameters are defined in the **.par** file.

The **.par** file will be discussed in detail during the excercises!

How to get and setup the seissol2d code:

Execute the following command (for SeisSol group members):

svn checkout https://user@svn.geophysik.uni-muenchen.de/svn/seissol2d/trunk

This will create a folder **trunk** in the directory, where you executed the command.
Within the folder **trunk** you will find the following subdirectories:

./common
./main
./Maple
./Matlab
./src

Create a new directory **./examples** for your own applications by

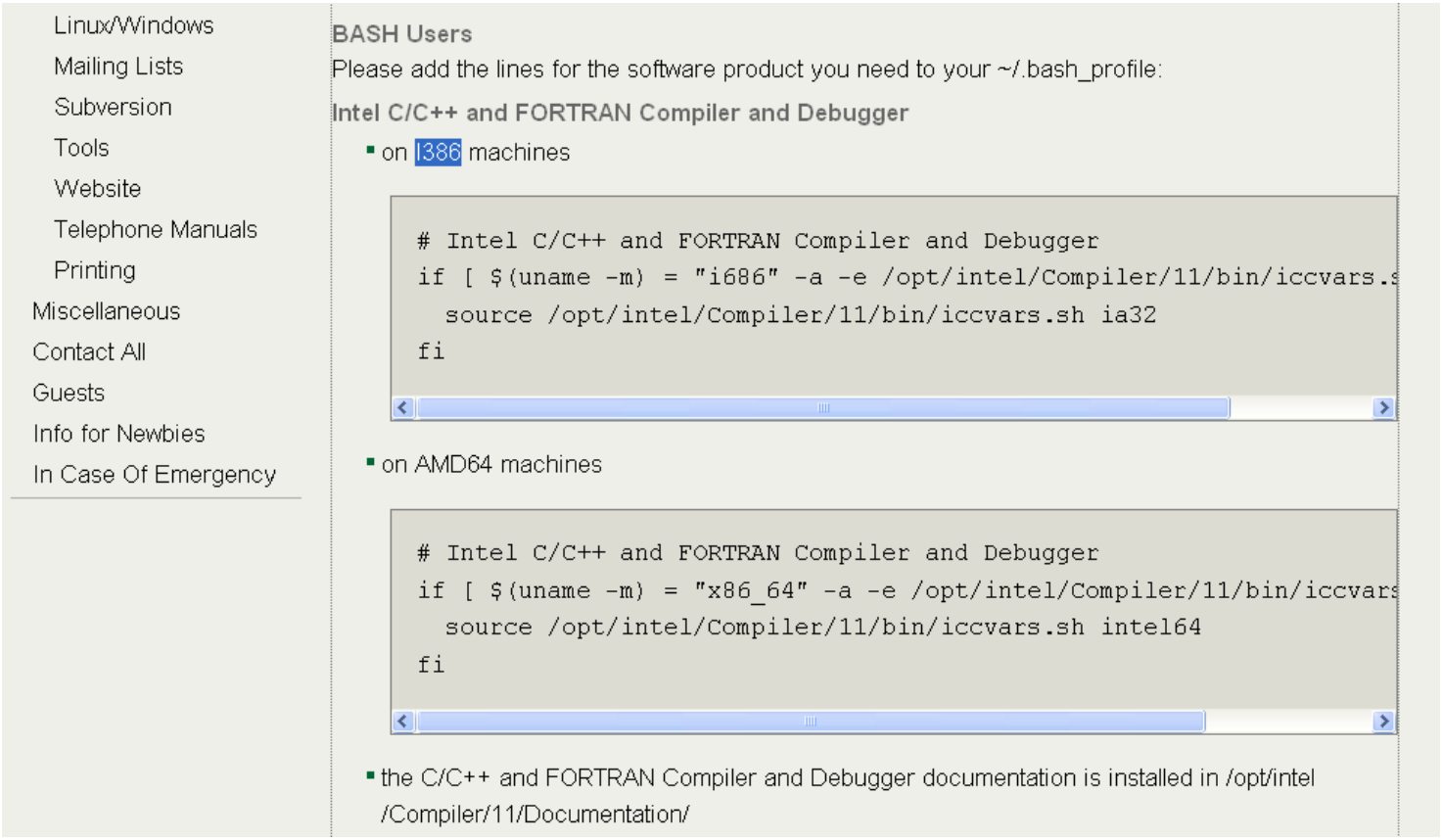
mkdir examples

How to compile the seissol2d code (1):

Make sure you have the **Intel compiler** available:
Follow the description on our **intranet** web-pages

intranet → IT Service/HowTo's → Applications → Intel Software

for I386 and AMD 64 machines.

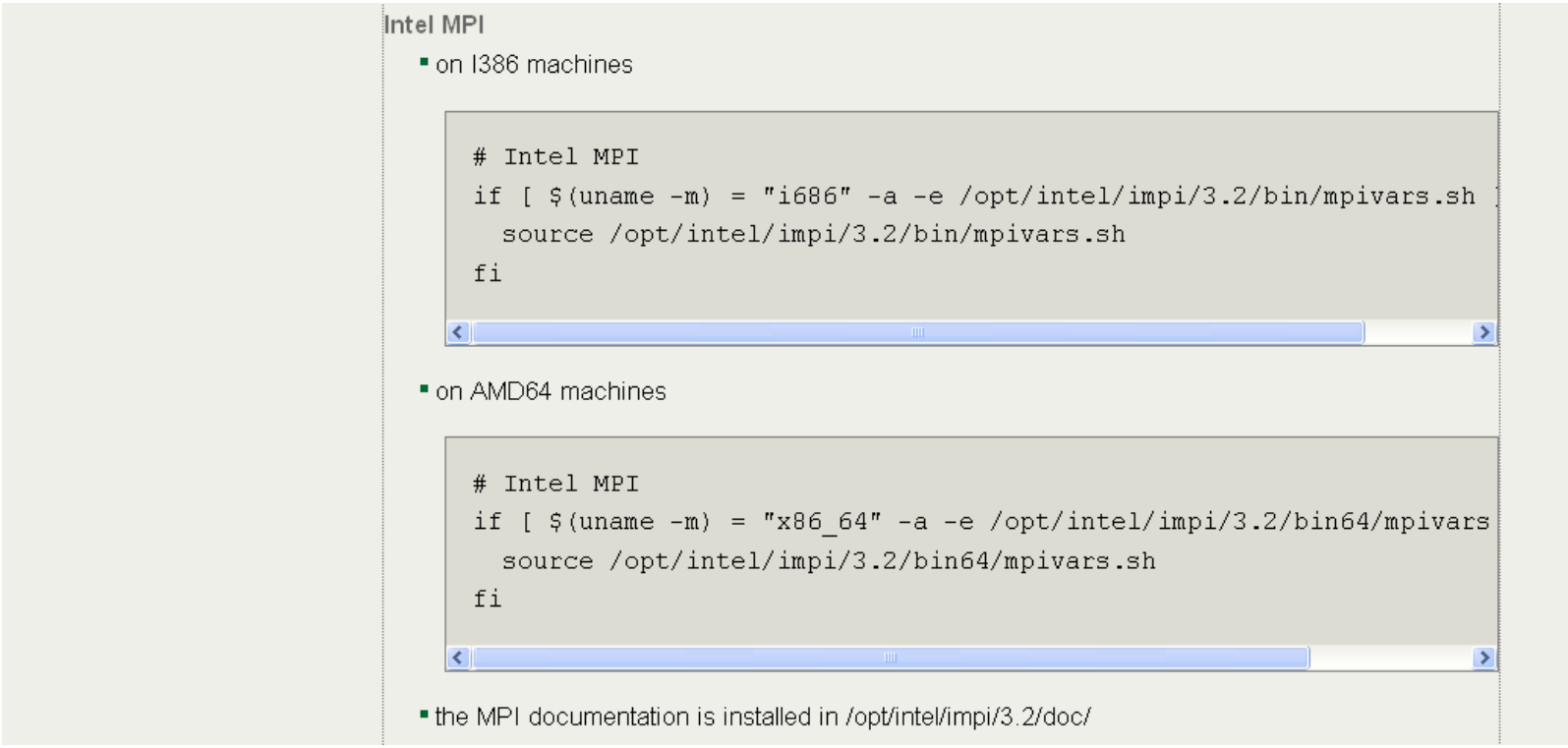


How to compile the seissol2d code (2):

Make sure you have the **Intel MPI** available:
Follow the description on our **intranet** web-pages

intranet → IT Service/HowTo's → Applications → Intel Software

for I386 and AMD 64 machines.



Intel MPI

- on I386 machines

```
# Intel MPI
if [ $(uname -m) = "i686" -a -e /opt/intel/impi/3.2/bin/mpivars.sh ]
  source /opt/intel/impi/3.2/bin/mpivars.sh
fi
```

- on AMD64 machines

```
# Intel MPI
if [ $(uname -m) = "x86_64" -a -e /opt/intel/impi/3.2/bin64/mpivars
  source /opt/intel/impi/3.2/bin64/mpivars.sh
fi
```

- the MPI documentation is installed in /opt/intel/impi/3.2/doc/

How to compile the seissol2d code (3):

If your setup is correct, go to the **./main** directory and type:

make clean

make

After a successful compilation ending with

…

```
==========================================
==========================================
=  installed ./../bin/seissol2dxx
==========================================
==========================================
```

additional directories should have been created:

./lib
./bin

where **./bin** contains the executable

seissol2dxx

How to run a SeisSol simulation:

To run a simulation go to your working directory, e.g.

./examples/basin

and start the simulation with e.g.

./seissol2dxx basin.par

or on a cluster system (i.e. TETHYS)

mpirun.openmpi -np 32 -nolocal -machinefile TETHYS.machines.32.G1 seissol2dxx basin.par

that uses the executable **seissol2dxx**

on 32 cores defined in **TETHYS.machines.32.G1**

and the simulation parameters defined in **basin.par**

Output files generated by a SeisSol simulation:

Each core writes its

number of core

log-file: **IRREGULARITIES.0000.log**

progress-file: **StdOut0000.txt**

number of receiver

seismograms: **output-pickpoint-00001-0000.dat**
(= time series of seismic ground motion at one postion in space)

number of time step

snapshots: **output-000000000.0000.tri.dat**
(= spatial slice of seismic ground motion on **mesh vertices** at one position in time)

snapshots fine-output: **output.GF.000000000.0000.dat**
(= spatial slice of seismic ground motion as **polynomial coefficients in each element** at one position in time)

➔ the snapshot fine-output has to be **post-processed** for visualization on an **additional, regular, fine visualization grid**

➔ the post-processing is done with **dgvisuxx**

How to visualize SeisSol simulation results:

The main results from a SeisSol simulation are:

- seismograms (= time series of seismic ground motion at one postion in space)
- snapshots (= spatial slice of seismic ground motion at one position in time)
- fine snapshots (= spatial slice of seismic ground motion at one position in time on a fine visualization mesh)

All output is generated in **tecplot**-format, remotely available through the LRZ.

Alternatively, **Gnuplot**, **Python** or **matlab**-scripts can be used for visualization.

Provided matlab-scripts in the repository:

for seismograms:	Reformat_seissol_seismograms.m Plot_seissol_seismograms.m	**(➔ data reduction!)**
for snapshots:	Plot_seissol_snapshot.m	
for fine snapshots:	Plot_seissol_snapshot_fine.m	**(➔ after post-processing with dgvisuxx)**

Post-Processing of Galerkin Fine (GF) output:

The Galerkin Fine (GF) output contains the **polynomial coefficients** of the approximation for every element.
Therefore, the **resolution can be much higher** through the element-internal structure of the wave field than for the normal snapshot output.

dgvisuxx is a tool to evaluate the polynomial approximation on a user-defined regular visualization grid

The required input file (e.g. **visu_basin.in**) can have the following form:

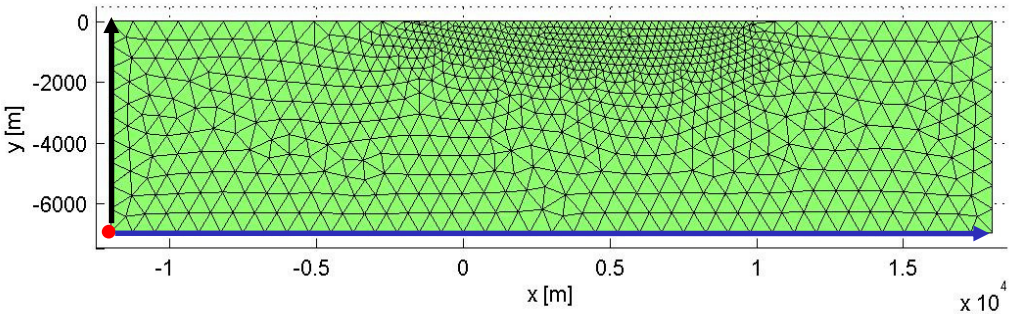
```
 1                          ! Cartesian mode (yes=1)
-12000.    -7000.    0.     ! Coordinates of origin
 30000.       0.     0.     ! Vector of 1st  Cartesian axis
     0.    7000.     0.     ! Vector of 2nd Cartesian axis
400                         ! Number of samples on 1st  axis
100                         ! Number of samples on 2nd axis
1                           ! MPI input data (no=0, yes=1)
32                          ! Number of CPUs
output.GF.000000000                ! MPI root filename
GF_basin.dat                       ! Output file name
1                           ! Output format (Tecplot=1)
sigma_xx                           ! Variable name 1
sigma_yy                           ! Variable name 2
sigma_xy                           ! Variable name 3
u                           ! Variable name 4
v                           ! Variable name 5
0                           ! End of file indicator
```

Create a file called DGPATH in your working directory, containing the absolute path to your Maple directory, e.g.

/home/messuser/seissol2d/Maple/

Then execute the post-processing via:

./dgvisuxx < visu_basin.in



Visualize data:

To visualize the seismograms (***-pickpoint-00001.dat**) you can either use Python or simply gnuplot.

Start gnuplot with

gnuplot

The command to plot the second column against the time is

plot '****-pickpoint-00001.dat' u 1:2 w l**

Visualize data:

To visualize the snapshots which include mesh information use:

```
./visz_snap.py
```

as follows:

```
./visz_snap.py output-000000000.tri.dat
```

You produce a readable fine-output that respects the high-order polynomials with:

```
./dgvisuxx < visu.in
```

The result

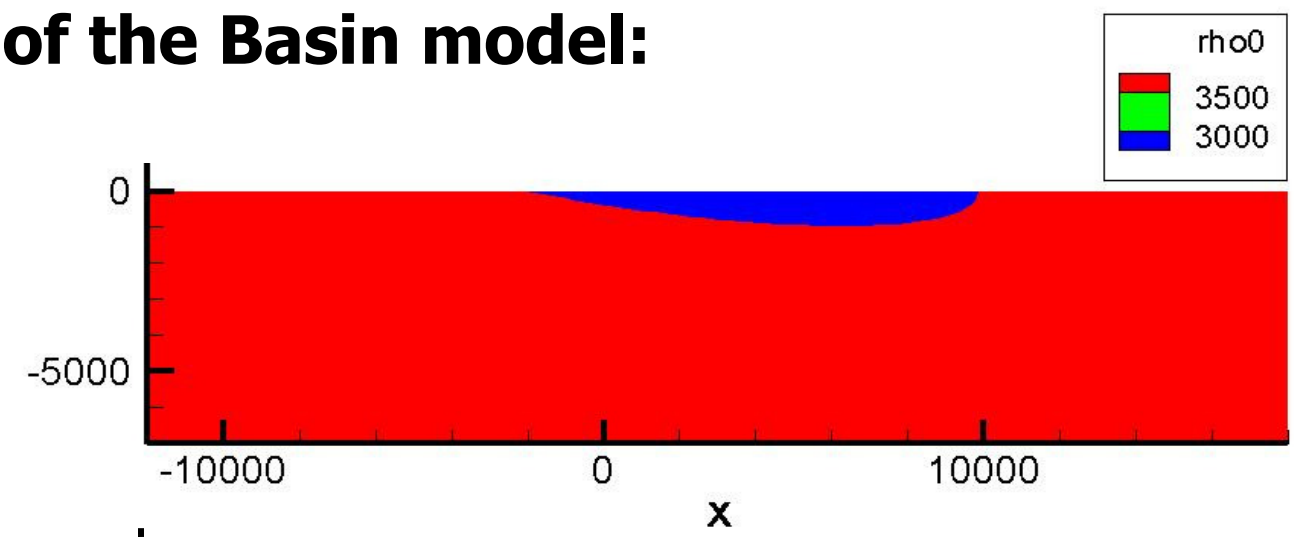
```
GF_output_t0.dat
```

can be visualized with

```
./visz_fine.py GF_output_t0.dat
```

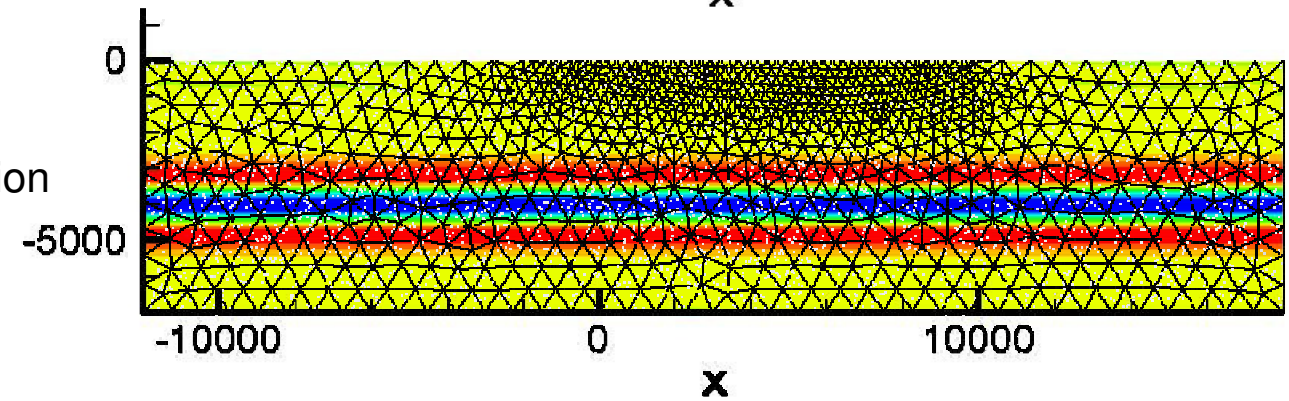
Sketch of the Basin model:

Density ρ

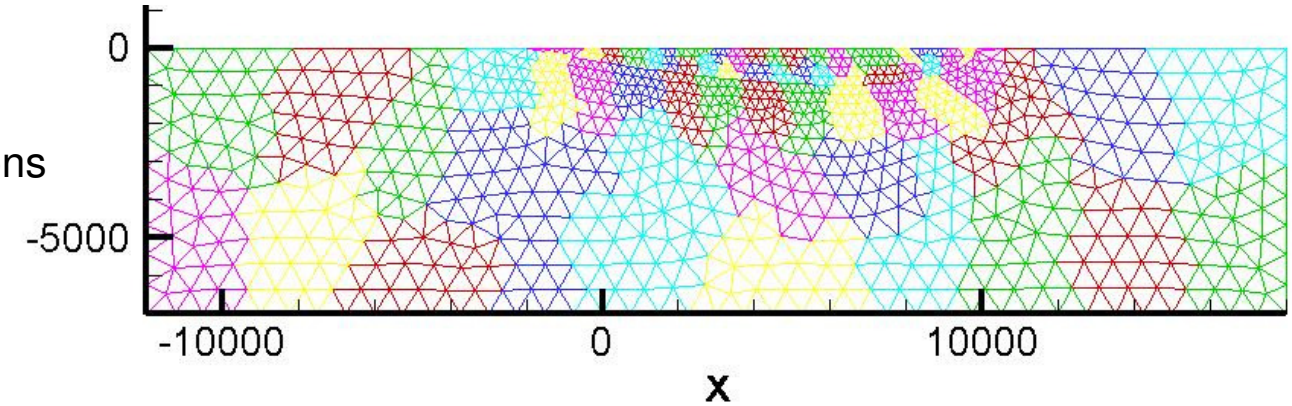


Mesh

initial condition



MPI partitions



What will I find in my seissol2d folder?

/bin

- Contains the executables

/examples

-Contains the example models and setups

- your working directory

/Maple

- Contains the basis functions

/seissol2d_demo

- Source code

/seissol_papers

- Fundamental publications

/visualization

- Contains the visualization tools

seissol_course.pdf

seissol2d_workflow.pdf

seissol2d_docu.pdf

Seissol_practicals.pdf