A Quick Start with Netgen / NGSolve

Overview

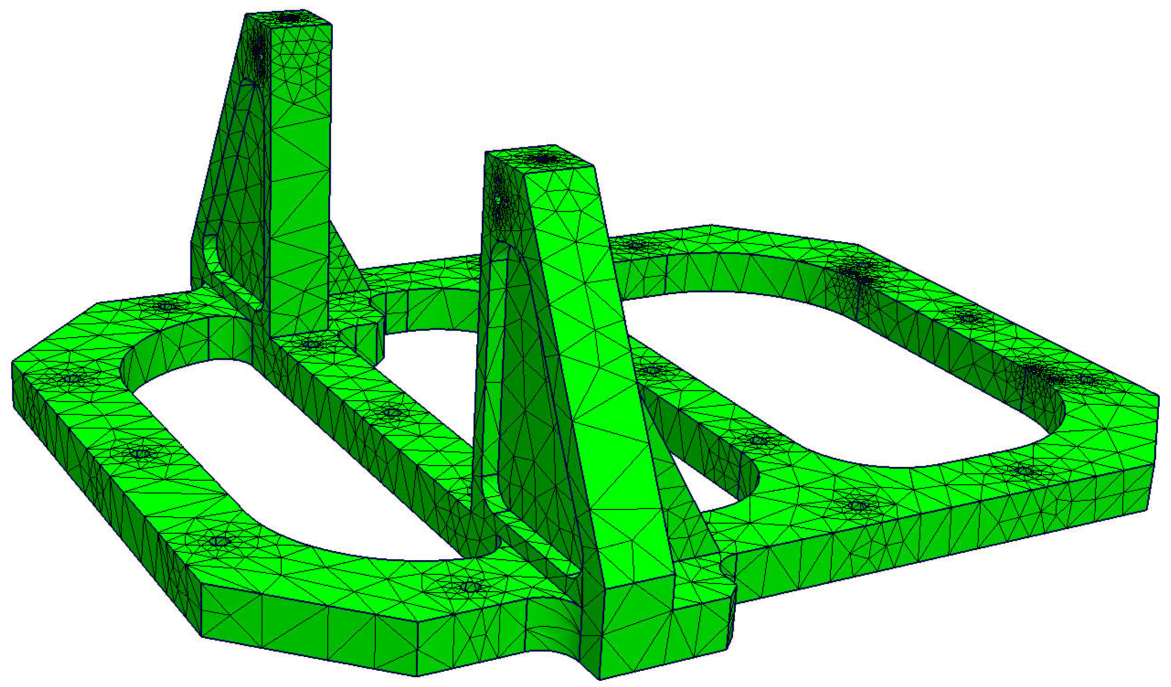
- A tour through Netgen/NGSolve
- How to install Netgen and NGSolve
- How to use the mesh generator Netgen
- How to use the Finite Element Solver NGSolve
- How to program in NGSolve

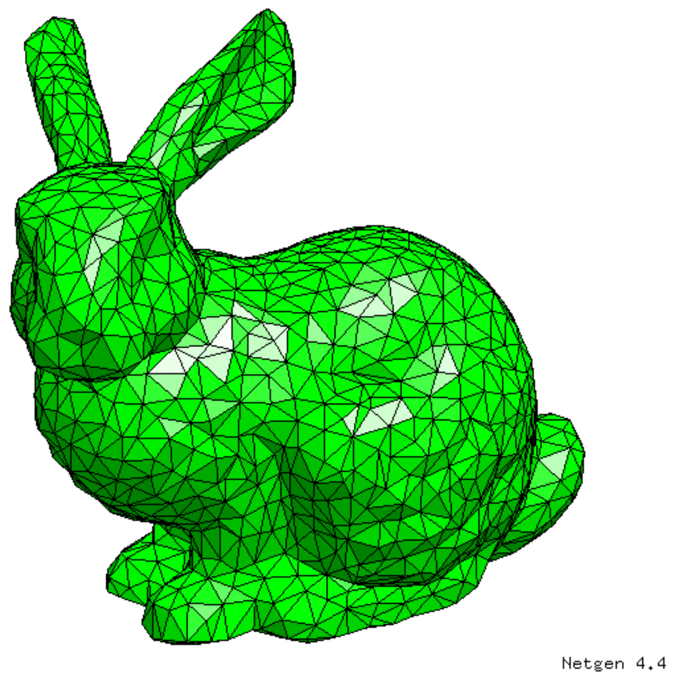
What is Netgen ?

- Netgen is an automatic 2D and 3D tetrahedral mesh generator
- Input can be provided by simple ASCII files (csg - files), or imported from CAD programs via IGES, Step, or STL files
- Netgen generates essentially unstructured triangular/tetrahedral meshes
- Netgen comes with a graphical user interface, or can be used as library
- Netgen is open source based on the LGPL license

Some Meshes generated by Netgen

A machine frame imported via the Step - format, a bunny imported from STL ...



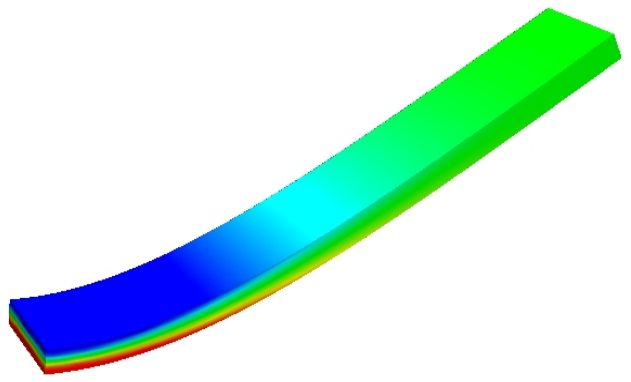


What is NGSolve ?

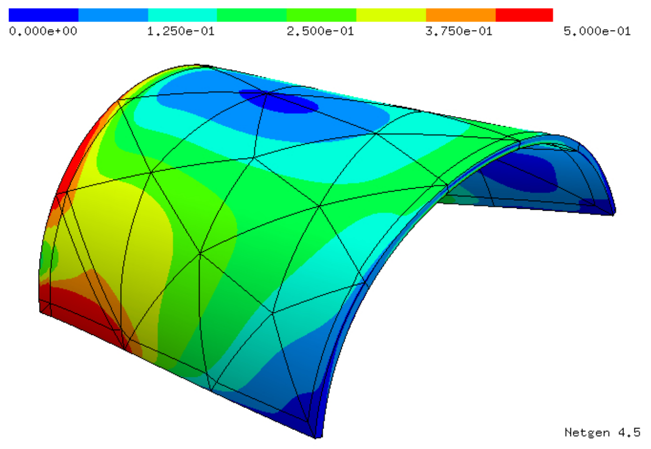
- NGSolve is a finite element library which can be linked to Netgen
- contains arbitrary order finite elements of all standard element geometries, scalar, vector-valued, hybrid DG finite element spaces
- Integrators for basic equations (heat flow, elasticity, Maxwell, ...)
- Iterative solvers, a posteriori error estimates, ...
- There are extension modules for different application areas (ngs-mech, ngs-flow, ...)
- NGSolve is open source based on the LGPL license

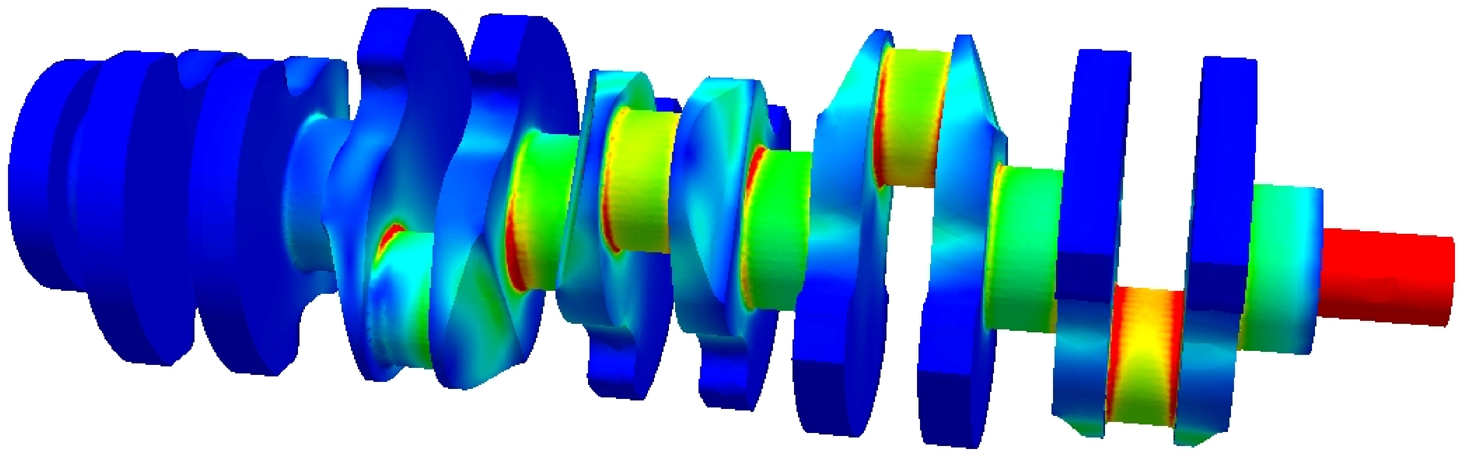
Simulation of Mechanical Deformation and Stresses

Elastic beam:



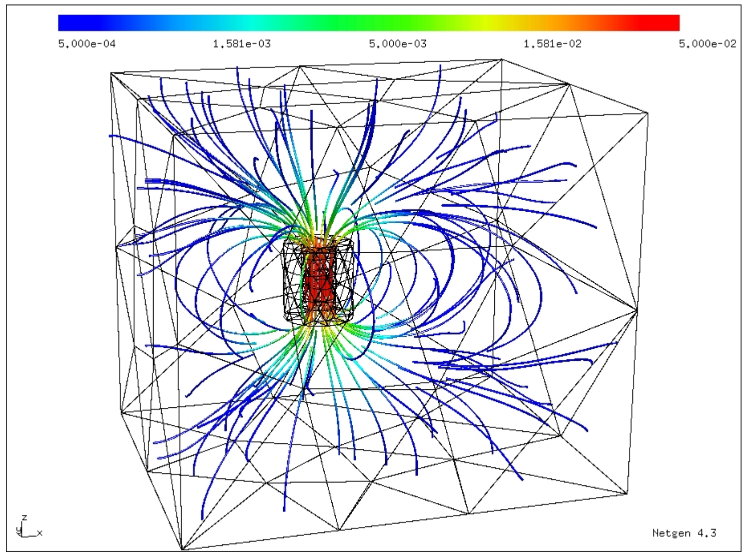
Shell structures:





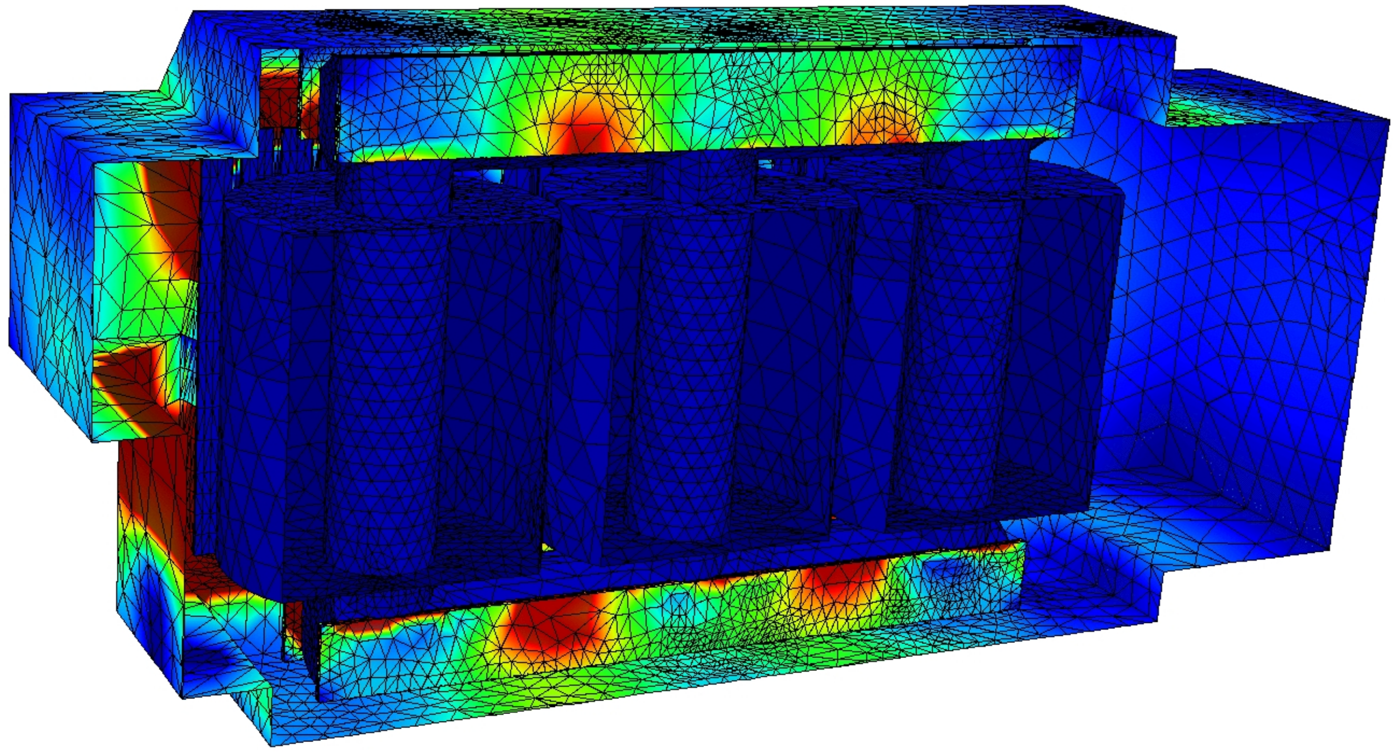
Simulation of Electromagnetic Fields

A simple coil:



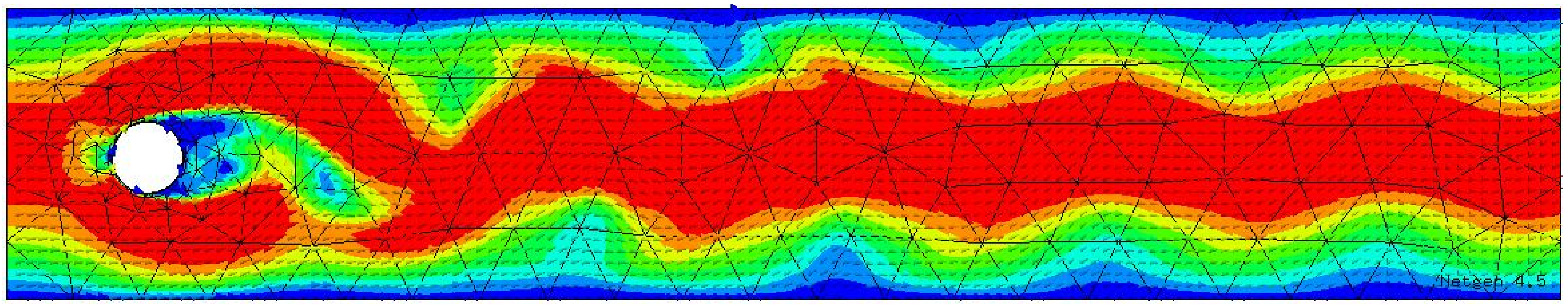
Described by
Maxwell's equations

A transformer built by Siemens - EBG, Linz:

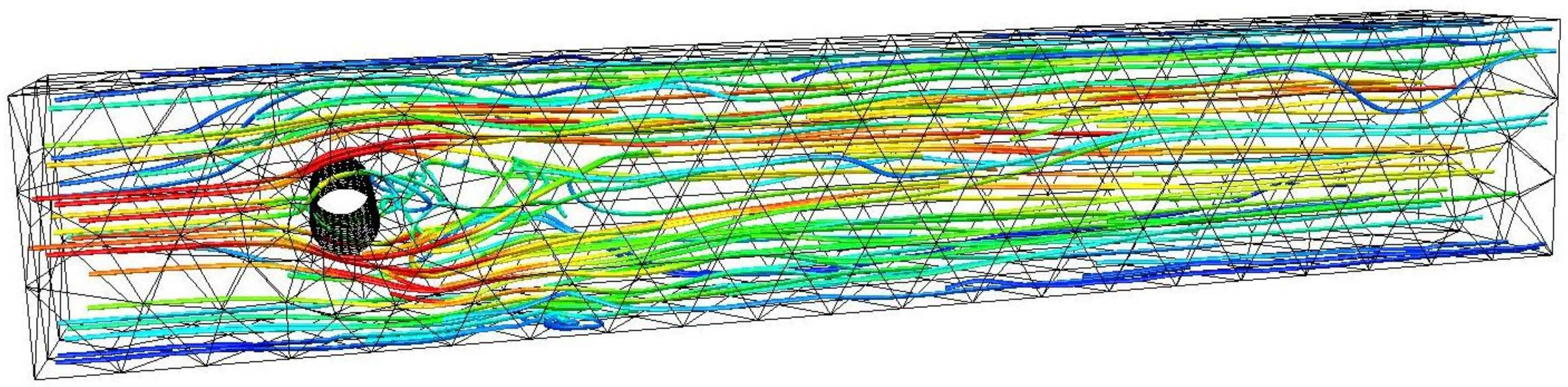


Incompressible Flows

Flow around a disk, Re $= 100$, 5^{th}-order elements:



Flow around a cylinder, Re $= 100$



Downloading and Installing Netgen/NGSolve

The codes and most detailed informations are available from sourceforge:

- http://sourceforge.net/projects/netgen-mesher/
- http://sourceforge.net/projects/ngsolve/

There are source packages, binary packages (windows), and the latest code in SVN

Latest installation information is on the Wiki - pages

- http://netgen-mesher.wiki.sourceforge.net
- http://sourceforge.net/apps/mediawiki/ngsolve

Installing Netgen/NGSolve on Windows

There are 3 different options:

- Install Netgen/NGSolve executables from the installer. This is enough to run Netgen/NGSolve.
- Compile Netgen and NGSolve yourself using Microsoft Visual C++ (free express edition is enough). This requires a couple of additional libraries such as Tcl/Tk/Tix/Togl, pthread-win32 for Netgen, and Lapack for NGSolve. You can work on the whole code.
- Install Netgen binary, and compile NGSolve yourselves. This only requires the Lapack library. This version is recommended if you want to implement some finite element stuff. It generates the dynamic link library ngsolve.dll, which must be copied into the Netgen directory.

Installing Netgen/NGSolve on Linux/Unix/Mac

1. Install 3rt party libraries as described in the installation manual on wiki (tcl-devel, tk-devel, tix, togl1.7, glut)

2. Compile and install Netgen:

   ```
   > ./configure --prefix=/opt/netgen
   > make install
   ```

   It puts the executable, tutorials, and header files into the installation directory.

3. Compile and install NGSolve. It requires some header files from Netgen

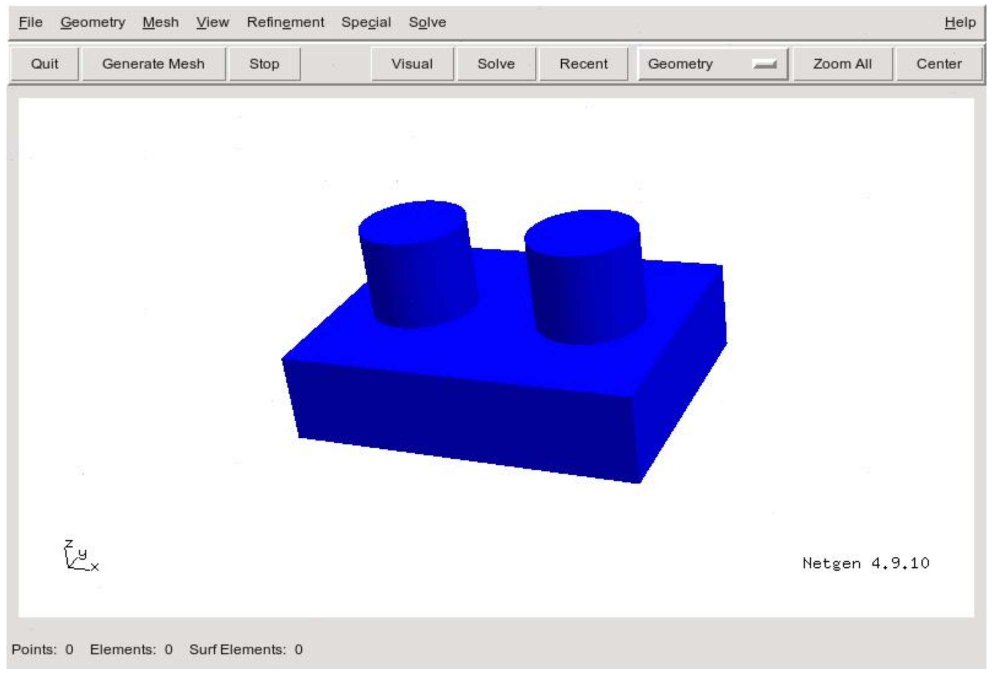
   ```
   > ./configure --prefix=/opt/netgen --with-netgen=/opt/netgen
   > make install
   ```

   It puts the shared library libngsolve.so into the installation directory.

Add the /opt/netgen/bin to your PATH, and /opt/netgen/lib to your LD_LIBRARY_PATH variable.

Running the Mesh Generator Netgen

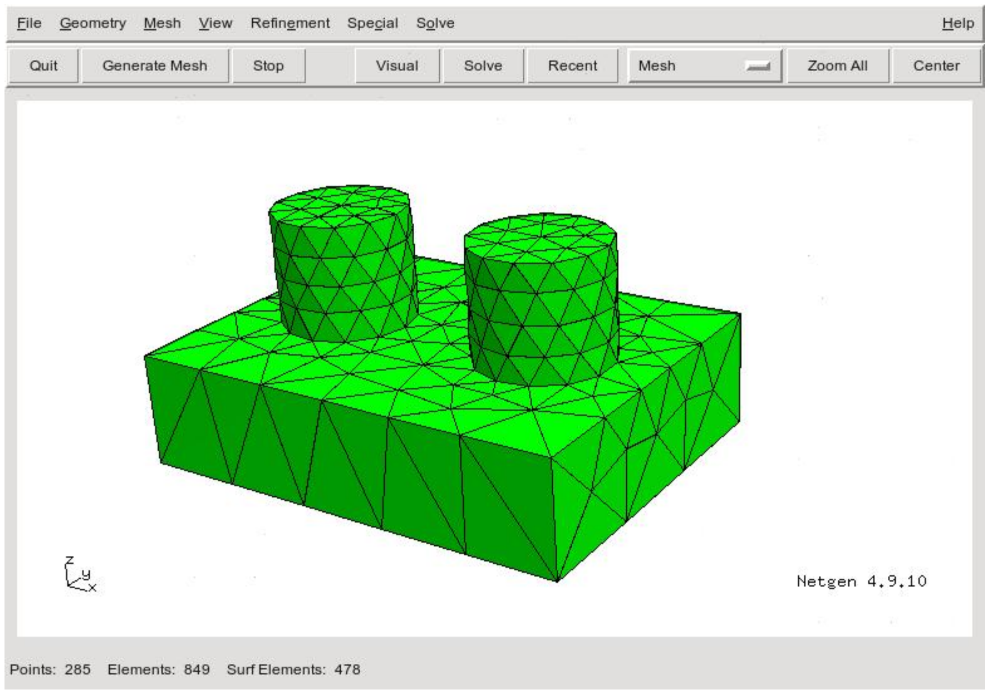
Start the program 'netgen'. Select "File → Load Geometry", and load, e.g., the file */opt/netgen/share/netgen/boxcyl.geo*. The window should look like this:



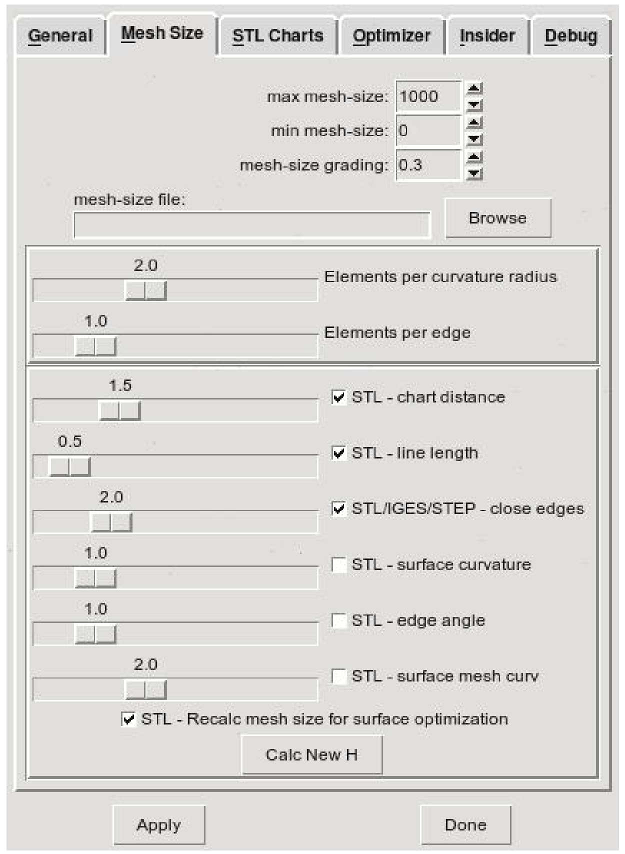
Moving the mouse keeping the left button, the middle button, or right button pressed rotates, moves, or zooms the model.

Generating the Mesh

Push the button “Generate Mesh” to generate a mesh:



Points: 285 Elements: 849 Surf Elements: 478



max mesh-size: 1000
min mesh-size: 0
mesh-size grading: 0.3
mesh-size file:

2.0 Elements per curvature radius
1.0 Elements per edge

1.5 ☑ STL - chart distance
0.5 ☑ STL - line length
2.0 ☑ STL/IGES/STEP - close edges
1.0 ☐ STL - surface curvature
1.0 ☐ STL - edge angle
2.0 ☐ STL - surface mesh curv
☑ STL - Recalc mesh size for surface optimization

With “Mesh → Meshing Options” you can set parameters. E. g., a general Mesh granularity from ’very fine’ to ’very coarse’, or a maximal mesh size.

Define the geometry

Supported Geometry formats are

- 2D geometries (the .in2d files)
- 3D constructive solid geometries (the .csg files)
- surface triangulations (.stl files)
- IGES and Step files, (requires the optional OpenCascade geometry kernel)

The 3D CSG - Files

You specify the the geometry in an ASCII file. Example:

```
algebraic3d
solid cube = orthobrick (0, 0, 0; 1, 1, 1);
solid cyl = cylinder (0.5, 0.5, 0; 0.5, 0.5, 1; 0.03);
solid main = cube and not cyl;
tlo main;
```

One defines a solid 'cube', an infinite cylinder solid 'cyl', and forms the intersection of the cube and the complement of the cylinder. The Top-Level-Object (tlo) 'main' is going to be meshed. Several tlo will lead to several regions (sub-domains).

- The syntax is described in the Netgen user manual
- Have a look into the various .geo examples

The 2D Files

You specify the geometry by a set of curves. This gives a square:

```
splinecurves2dv2
2
points
1       0       0      -maxh=0.01
2       1       0
3       1       1
4       0       1
segments
1       0       2       1       2       -bc=1  -maxh=0.1
1       0       2       2       3       -bc=1
1       0       2       3       4       -bc=1
1       0       2       4       1       -bc=2
materials
1       domain1   -maxh=0.3
```

splinecurves2dv2 is the keyword for the 2D inputs, version 2.

Next comes a global grading parameter.

The points given with index and coordinates

The curves are given by the sub-domain number on the left, sub-domain number on the right, number of points to define the curve, and the point indices. sub-domain number 0 means exteriori (which is not meshed). A straight line is defined by 2 points, a second order spline by 3.

The flag '-maxh' sets the maximal element diameter at a point, along a curve, or in a region (material).

The flag '-bc' defines the boundary condition index.

- Have a look into the various .in2d examples

Running NGSolve

When NGSolve is installed, you find a 'Solve' pull down menu item.

Select 'Solve $\rightarrow$ Load PDE', and load, e.g. the file */opt/netgen/share/ngsolve/d4_cube.pde*. Just a Poisson equation.

Push the solve button. Watch the shell window for the ongoing actions.

Push the Visual button for setting the visualization. Switch 'Scalar Function' to 'u'. Activate a clipping plane, and set 'Clipping Plane Sol.' to 'Scalar Function'.

Restart, and load example 'd5_beam.pde'. It is a mechanical deformation problem. Solve. Set 'Vector function' to 'u'. Click deformation, and set 'scale' to 100. Choose a 'Scalar function'.

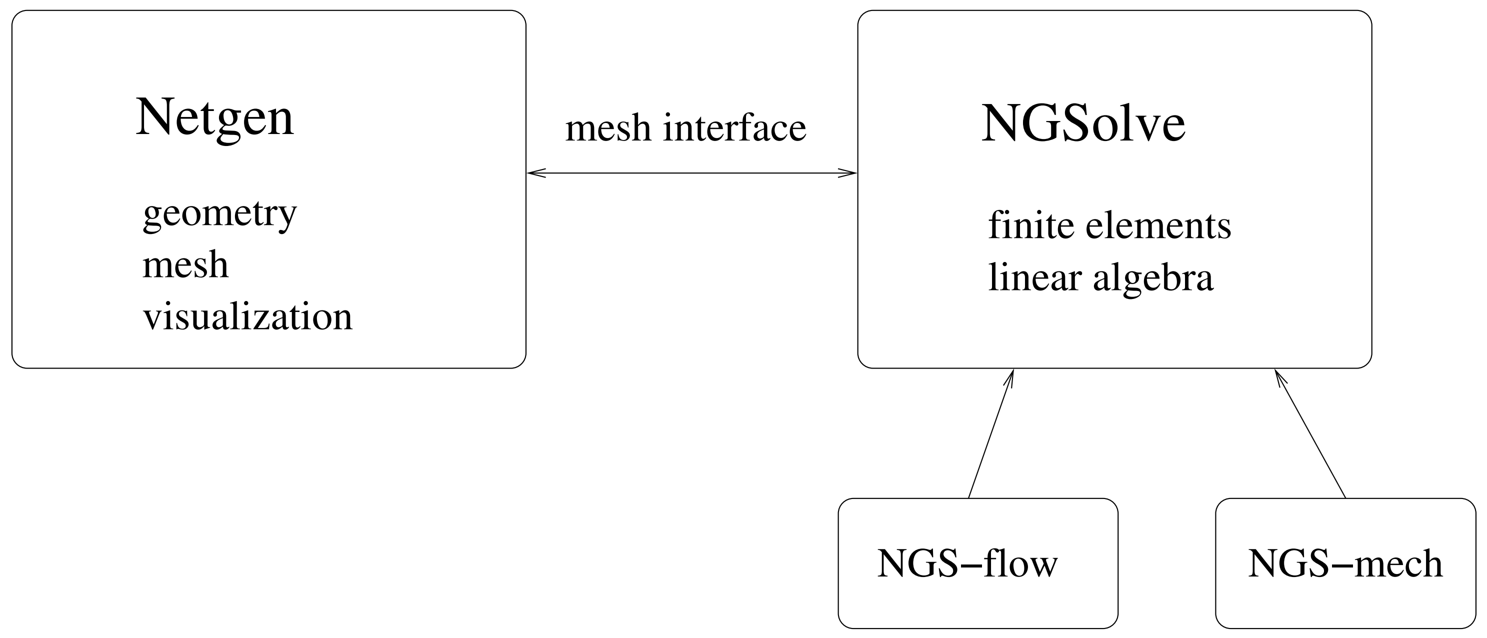
Restart, and load example 'd7_coil.pde'. It is a Maxwell problem (magnetostatics). Solve. Switch on a clipping plane. Select a Vector-Function 'flux'. Choose 'Clipping Plane Sol' to be 'Vector Function'. BUG: switch off Use Texture.

NGSolve script file for Poisson example

Find $u \in H^1$ s.t. $$\int_\Omega \lambda \nabla u \cdot \nabla v \, dx + \int_{\partial\Omega} \alpha u v \, ds = \int_\Omega f v \, dx \qquad \forall v \in H^1$$

```
define coefficient lam      1,
define coefficient alpha    1e5, 1e5, 1e5, 0,
define coefficient cf       sin(x)*y,

define fespace v -h1 -order=4
define gridfunction u -fespace=v

define bilinearform a -fespace=v -symmetric
laplace lam
robin alpha

define linearform f -fespace=v
source cf

define preconditioner c -type=multigrid -bilinearform=a -smoothingsteps=1 -smoother=block

numproc bvp np1 -bilinearform=a -linearform=f -gridfunction=u -preconditioner=c -prec=1e-8
```

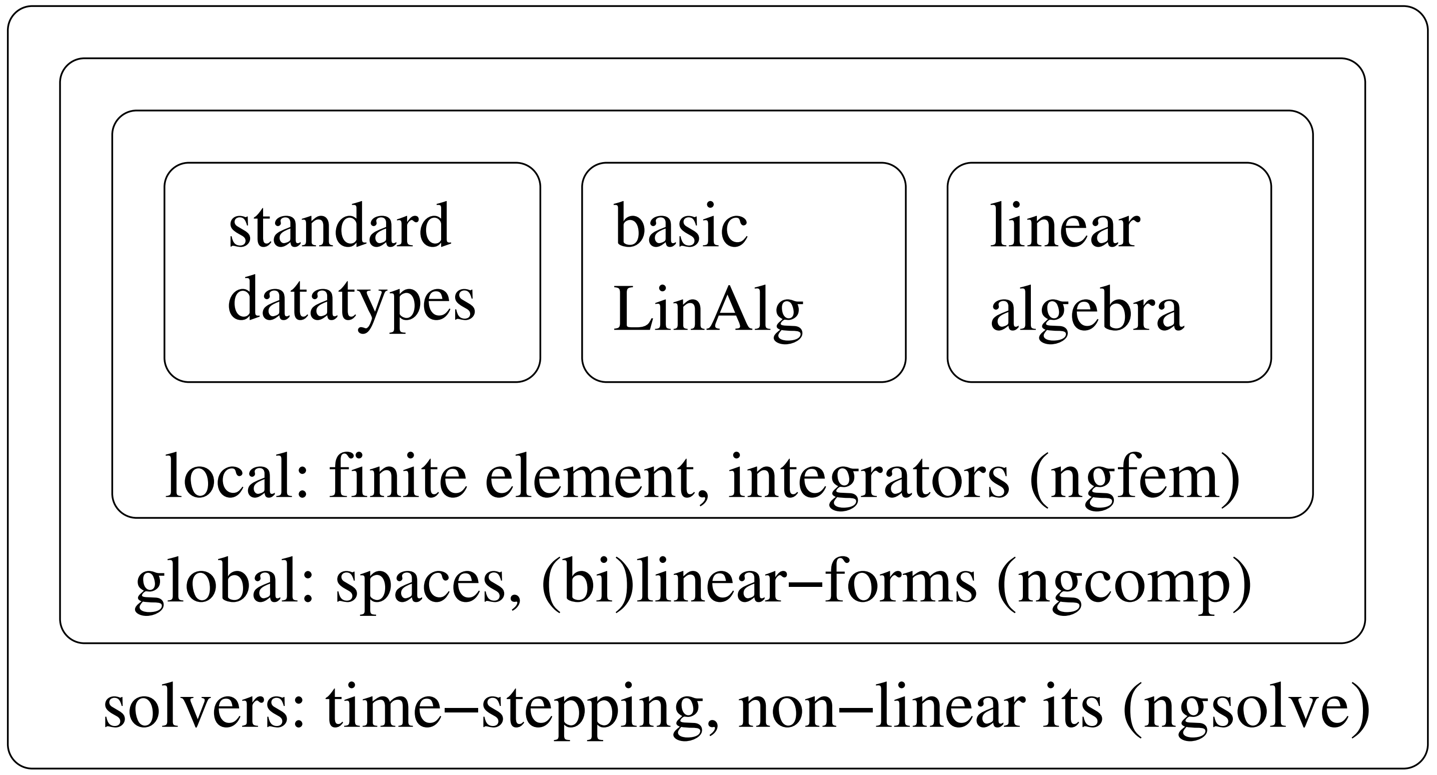
Interplay of Netgen and NGSolve



- Netgen knows the geometry, and maintains the mesh, does visualization
- NGSolve does the finite elements and linear algebra
- Connection is via the mesh interface (NGSolve: *class MeshAccess*)

Application classes can be loaded as shared libraries into NGSolve

The layers of NGSolve


standard datatypes
basic LinAlg
linear algebra
local: finite element, integrators (ngfem)
global: spaces, (bi)linear–forms (ngcomp)
solvers: time–stepping, non–linear its (ngsolve)

The layers correspond to namespaces (and sub-directories)

Central NGSolve classes

- FiniteElement (ngfem):
  Provides shape functions and derivatives on reference element
- ElementTransformation (ngfem):
  Represents mapping to physical elements, computes Jacobian
- Integrator (ngfem):
  Computes element matrices and vectors
- FESpace (ngcomp):
  Provides global dofs, multigrid-transfers and smoothing blocks
- BilinearForm/LinearForm (ngcomp):
  Maintains definition of forms, provides matrix and vectors
- PDE (ngsolve):
  Container to stores all components

Where to start programming ?

The directory 'my_little_ngsolve' provides a play ground to start NGSolve programming.

Run 'make intall' in the directory 'my_little_ngsolve'. It will generate and install a shared library with the new stuff.

1. Have a look into 'myElement.hpp' (.cpp), 'myFESpace.hpp' (.cpp), 'myIntegrator.hpp' (.cpp). Explains how to implement elements, finite element spaces, and integrators

   Exercise: Start adding elements (quadrilateral, tetrahedral element), and integrators (Neumann b.c.)

2. Have a look into 'demo_instat.cpp'. It demonstrates how to write a simpe solver for parabolic equations. Works with linear algebra objects

   Exercise: Implement the Newmark solver for second order hyperbolic equations.

3. Have a look into 'demo_coupling.cpp'. It demonstrates how to work with finite elements, how to mange dofs, and the elementtransformation.

More prgramming demos are in 'programming_demos'. These show how to write stand-alone programs using the NGSolve objects.