Netgen/NGSolve Tutorial Part 0 - Installing Netgen/NGSolve

Christoph Lehrenfeld

March 16, 2015

Contents

1 How to run Netgen/NGSolve?

There are several ways in order to set up a running Netgen/NGSolve. This might depend on:

- The operating system (Windows / Linux / Mac)
- Whether you need the executables only or if you need to compile the main code / addon packages

We will use the current version of the 6.1-dev branch of Netgen and NGSolve.

1.1 Debian-package

On Linux distributions as Debian or Ubuntu, you can download and install the packages from here:

- Netgen Debian package
- NGSolve Debian package

Install the packages using the following commands

```
sudo dpkg -i netgen-6.1-dev-Linux.deb ngsolve-6.1-dev-Linux.deb
sudo apt-get -f install
```

Afterwards, reboot.

1.2 Running Netgen/NGSolve remotely on "matrix"

On matrix there is pre-installed Netgen/NGSolve which you can use if you remotely connect to matrix.

1.2.1 Connecting to matrix

1. From Windows: Install some X-client, such as "MobaXterm" and add a new (ssh)connection to matrix.asc.tuwien.ac.at with the username xfem (xfem@matrix.asc.tuwien.ac.at), login and proceed as any linux user.
2. From Linux: Connect to matrix using ssh:

   ```
   ssh -Y xfem@matrix.asc.tuwien.ac.at
   ```

1.3 Self-compiled

In case you want to change stuff in the source code or want to achieve a local installation, you should proceed as follows:

1.3.1 Prerequisites

Make sure that you have the following packages installed

- You need a recent compiler, we advise gcc in version 4.7 or higher.
- We advise to have python installed, in version 3.2 or higher (you can compile netgen/ngsolve also without python support)

- If you want to use python support (you do!), make sure that you have boost-python installed in a version larger than 1.54
- You will need tcl / tk in version 8.5 and the tcl-packages "tix". Make sure to install according packages in their "dev"-version to have the suitable header files installed.
- cmake (>=2.8.9) (or autotools) for the build system
- libxmu6

1.3.2 directory structure

For ease of presentation we use a directory structure with three main directories:

- "src" for the source codes of Netgen/NGSolve and additional packages
- "build" for builds and
- "inst" for installations (results)

In the following we assume that you have chosen a base directory where three (empty) directories "src", "build" and "inst" have been created. This base directory will be denoted as ${BASEDIR} in the following.

1.3.3 Netgen

1. Getting the source Make sure you have git installed. Then,

   ```
   git clone git://git.code.sf.net/p/netgen-mesher/git src/netgen
   ```

   gives "src/netgen"

2. Building from the source

   (a) Configuring change into the "build" directory and create a new directory netgen (results in: "build/netgen")

   ```
   cd build
   mkdir netgen
   cd netgen
   ```

   Next, use cmake to configure from the sources. To this end call "cmake" and use the link to the source directory as final argument

```
cmake XYZ ../../src/netgen
```

with parameters XYZ where you can set a lot of options. You can also use ccmake or cmake-gui (if installed) to select the options there. Important options are

- The installation directory, here "-DINSTALL_DIR=${BASEDIR}/inst"
- Release type, here a reasonable choice is "-DCMAKE_BUILD_TYPE=RELEASE"

So your configuring command could look like this:

```
cmake -DINSTALL_DIR=${BASEDIR}/inst -DCMAKE_BUILD_TYPE=RELEASE ../../src/netg
```

Alternatively you can also use autotools to configure your installation, if you prefer autotools.

(b) Building Now, call

```
make
```

You may want to add "-jx" with x the number of threads you want to use for the compilation. If everything goes smooth you can install the resulting build calling

```
make install
```

(c) Finishing the installation Finally you have to set the environment variable "NETGENDIR" to the location of the executable, eg. by

```
export NETGENDIR="${BASEDIR}/inst/bin"
```

or

```
setenv NETGENDIR "${BASEDIR}/inst/bin"
```

(depends on your linux distribution). You may want to add the corresponding line to your .bashrc, s.t. it is automatically set whenever you log in into your bash-shell.

(d) Test the installation Now the installation should be finished. Test it with calling netgen

```
netgen
```

in ${BASEDIR}/inst/share/netgen you can find several geometry and mesh files which you can use to try if netgen does what it should do.

1.3.4 NGSolve

The installation of NGSolve is pretty similar.

1. Getting the source

   ```
   git clone git://git.code.sf.net/p/ngsolve/git src/ngsolve
   ```

2. Building from the source

   (a) Configuring change into the "build" directory and create a new directory ngsolve (results in: "build/ngsolve")

   ```
   cd build
   mkdir ngsolve
   cd ngsolve
   ```

   Next, use cmake to configure from the sources. To this end call "cmake" and use the link to the source directory as final argument

   ```
   cmake XYZ ../../src/ngsolve
   ```

   with parameters XYZ where you can set a lot of options. You can also use ccmake or cmake-gui (if installed) to select the options there. Important options are

   - The installation directory, here "-DINSTALL_DIR=${BASEDIR}/inst"
   - Release type, here a reasonable choice is "-DCMAKE_BUILD_TYPE=RELEASE"
   - If you have intel mkl installed you have to activate it with "-DUSE_MKL=ON -DMKL_ROOT=/opt/intel/composer_xe_2015.1.133/mkl/" where you should replace the MKL_ROOT with your correct path

   So your configuring command could look like this:

   ```
   cmake -DINSTALL_DIR=${BASEDIR}/inst -DCMAKE_BUILD_TYPE=RELEASE ../../src/ngsol
         -DUSE_MKL=ON -DMKL_ROOT=/opt/intel/composer_xe_2015.1.133/mkl/
   ```

   Alternatively you can also use autotools to configure your installation, if you prefer autotools.

   (b) Building Now, call

   ```
   make
   ```

   You may want to add "-jx" with x the number of threads you want to use for the compilation. If everything goes smooth you can install the resulting build calling

```
make install
```

(c) Test the installation Now the installation should be finished. Test it with calling netgen

```
netgen
```

and see if you get a message saying that the module NGSolve-6.1-dev has been loaded. In ${BASEDIR}/inst/share/ngsolve you can find example PDE problems which you can use to try if ngsolve does what it should do.

2 Getting started with the exercise content

2.1 On matrix

change in to the directory work ("/home/xfem/work") and create a directory with a unique name. The content of the lecture can be found on the homepage, but is also available in "/home/xfem/content". Please copy whatever you want to use to your local work directory "/home/xfem/work/myname".

```
cd ~
cd work
cd myname
cp ~/content/xfem_exercise . -r
```

3 Resulting files

- installngs.pdf
- installngs.html