INSTALLATION GUIDE

CalculiX Graphical Pre- and Postprocessor cgx 2.8 on Mac OS X Mavericks

[1]B. Graf

Abstract. CalculiX comprises the graphical pre- and postprocessor cgx (CalculiX GraphiX) and the solver ccx (CalculiX CrunchiX). A step by step installation guide for cgx 2.8 on Mac OS X 10.9.5 Mavericks is provided. Installation from source is considered. The present guide is based on the general installation instructions for Unix/Linux which come with the cgx package. Besides the installation of cgx the instruction also covers the installation of required Mac-specific and optional additional software, whereas the latter is required for additional functionality of cgx. Except of html-help files and the installation path the installation of cgx is independent from the installation of ccx. Among others, the installation has been tested for the functionality of the additional software.

Acknowledgement. Special thanks to K. Wittig (developer of cgx, MTU Aero Engines Munich, Germany) and M. Woopen (installation of Netgen with GUI, AICES, RWTH Aachen, Germany) for their support.

[1]Prepared by B. Graf, based on the general installation instructions for Unix/Linux by K. Wittig (developer of cgx, MTU Aeoro Engines, Munich, Germany).

Dr. B. Graf
Consulting Engineer
Zurich
graf_bernhard@bluewin.ch

RELEASE NOTES

Revision	Remarks
Draft, June 6, 2014	Installation of cgx 2.7 on Mac OS X 10.9.3 Mavericks.
February 10, 2015	Installation of cgx 2.8 on Mac OS X 10.9.5 Mavericks. Changes: - Compilation of cgx with unmodified GCC 4.9 instead of modified GCC 5.1 to avoid problem of GCC 5.1 with function overloading. The latter solved former problem (segfault) for post processing of tensorial quantities with cgx 2.7. Installation instructions of GCC 4.9 added. - Optional installation of Netgen with GUI added in Appendix 2. - Editorial changes.

CONTENTS

APPENDICES

1. INTRODUCTION

1.1 General

CalculiX comprises the graphical pre- and postprocessor cgx (CalculiX GraphiX) and the solver ccx (CalculiX CrunchiX). The present guide covers the installation of cgx on Mac OS X Mavericks (cf. [4] for the installation of ccx). Except of html-help files and the installation path the installation of cgx is independent from the installation of ccx. Accordingly, the sequence of the installation of cgx and ccx may be chosen freely.

For a brief introduction to the installation of cgx on Unix/Linux, please read the instructions by K. Wittig [1]: "Section B: Installation from Source". Based on the aforementioned reference [1], in the present document a step by step installation guide for cgx on Mac OS X is provided. Installation from source is considered.

The installation of cgx requires the pre-installation of Mac-specific software like Xcode, XQuartz, the GCC (GNU compiler collection) for OS X and MacPorts. The subsequent installation of cgx may be partitioned into the following two steps: (1) standalone installation of cgx, (2) optional installation of additional software which provides additional functionality of cgx. The sequence of the installation steps 1, 2 may be chosen freely. A standalone installation of cgx according to point 1 above (without installation of additional software) already provides a powerful pre- and post processing tool. Furthermore, all of the additional software or parts of it may be installed. If the additional software (all or partial) is not installed some of the additional functionality of cgx gets lost. The aforementioned additional functionality is further discussed below.

Section 1.2 contains an overview of the functionality of the aforementioned additional software and may help for the planning of the installation (cf. also Tab. 2 for an overview of the additional functionality). For the build of cgx all of the additional software is considered with regard to required flags, parameters, path settings etc. However, none of the additional software actually needs to be installed for the build of cgx. Accordingly, you may install the additional software (all or partial, depending on your needs) before or after the installation of cgx. The latter, therefore, requires no re-compilation, if cgx has been build before exactly as explained in the present guide.

Section 2 contains an overview of the required software and Section 3 a summary of prearrangements which are required for the installation of cgx. The installation instructions are contained in Sections 4 to 8. The installation should follow the sequence as given by Sections 4 to 8 with the following exceptions a, b for the installation of additional software according to Section 7: (a) all or part of the additional software may be installed before or after the installation of cgx according to Section 8, (b) sequence of installation of additional software may be chosen freely. First of all, in Section 4 the installation path and the file system are explained as well as the installation of the source code and html-help files of cgx. In the subsequent two Sections 5 and 6 the Mac-specific software is installed (Xcode, XQuartz, GCC, MacPorts) which is required for the installation of cgx and the aforementioned additional software. The additional software is installed in Section 7. Finally, the installation of cgx is completed in Section 8. Furthermore, Section 8 covers the

following subjects: launch of cgx and test of installation, how to get started with cgx, test examples. In Section 9 the functionality of the aforementioned additional software is tested. Section 10 contains hints and the final Section 11 a list of known Mac-specific problems. References are provided on page 50 and are followed by Appendix 1 in which the input of a test for tet meshing is provided. Appendix 2 is provided for the optional installation of Netgen (automatic 3D tetrahedral mesh generator) with GUI. The release notes are provided on page i.

1.2 Planning Your Installation and Functionality of Optional Additional Software

As already explained in Section 1.1 above, a standalone installation of cgx (without the below discussed additional software) is possible and already provides a powerful pre- and post processing tool. For a standalone installation of cgx skip all instructions for the installation of additional software. For the build of cgx in Section 8.1 it is recommended, however, to consider the path settings and parameters for the additional software. The latter allows for the subsequent installation of additional software without re-build of cgx. Conversely, none of the additional software actually needs to be installed for the build of cgx.

For additional functionality of cgx, the subsequently discussed optional additional software may be installed. The additional software as well as details of the functionality are listed in Tab. 1 and Tab. 2 below as follows (cf. also [1] for the additional software):

- *Tab*. 1. Additional software sorted by name in column 1. The functionality is explained in column 2 of Tab. 1. Cross references to the installation of the additional software given in column 2 of Tab. 1.
- *Tab*. 2. Additional software sorted by cgx menu items “MI” and cgx commands “C”. The functionality is explained in column 2 of Tab. 2. Cross references to the test of the additional software are given in column 2 of Tab. 2.

The aforementioned data from Tab. 1 and Tab. 2, therefore, may help for the planning of the installation of additional software. Accordingly, for example 3D tet meshing with target element size requires the installation of the additional software “Netgen” (cf. column 1 of Tab. 1 for details).

The functionality of the additional software is tested in Section 9 below (cf. also column 2 of Tab. 2 for cross references to Section 9).

Name and **Release** of additional software	**General, Function, Usage and Installation**	[1]**Mac App** with similar functionality
ImageMagick 6.8.8-6	• *General*. Suite for image manipulations. • *Function*. Provides "convert" command-line tool. • *Usage*. cgx uses "convert" for hardcopy of drawing area of cgx main window. • *Installation*. Section 7.2.	**Grab**
PSUtils p17	• *General*. Collection of utilities for image manipulations. • *Function*. Provides "pstops" command-line tool. • Usage: cgx uses "pstops" to bundle several ps (PostScript) files in one file. • *Installation*. Section 7.3.	-
gs 9.10 (GhostScript)	• *General*. Collection of utilities for image manipulations. • *Function*. Provide "gs" command-line tool. • *Usage*. cgx uses "gs" for file manipulations in context with usage of PSUtils. • *Installation*. Section 7.4.	-
gv 3.7.4 (GhostView)	• *General*. Post script viewer. For several reasons, the Mac Preview App has been installed instead of gv (cf. Section 7.5 for the installation of gv for details). • *Function*. Provides "preview" command-line tool (implemented with shell script according to Section 7.5). • *Usage*. Present cgx-installation uses "Preview" to display ps-plotfiles which are generated during pre- or post processing. • *Installation*. Section 7.5.	**Preview**
Firefox 35.0 (html-browser of your choice, Firefox chosen)	• *General*. html-Browser of your choice used to display cgx and ccx html-help files. • *Function*. Provides "browser_name" (Firefox for name chosen) command-line tool (implemented with shell script according to Section 7.6). • *Usage*. cgx uses "browser_name" to display cgx and ccx html-help files. • *Installation*. Section 7.6.	-
Netgen 5.1	• *General*. Automatic 3D tetrahedral mesh generator. • *Function*. Provides routine "ng_vol" for tet meshing. • *Usage*. cgx uses "ng_vol" to allow for tet meshing with a target element size. • *Installation*. Section 7.7. Installation of Netgen without GUI sufficient for functionality of cgx. Optional installation of Netgen with GUI considered in Appendix 2.	-
Gnuplot 4.6.0	• *General*. Command-driven function and data plotting program for 2D- and 3D plots. • *Function*. Provides "gnuplot" command-line tool. • *Usage*. cgx uses "gnuplot" to generate ps-plotfiles for xy-plots (path plots, history plots etc.). cgx also provides a script with gnuplot commands. • *Installation*. Section 7.8. Note. cgx provides (default) ASCII file with raw data for x-y-plots. If gnuplot is not installed, therefore, you may use the afore mentioned ASCII file for plotting with a plot program of your choice.	-

[1]Apps pre-installed on Mac. For details of usage of Apps cf. test of functionality of additional software in Section 9.

Table 1 Additional software sorted by name. Installation tested for software releases according to column 1. General, function, usage and cross reference to installation according to column 2. Mac Apps with similar functionality listed in column 3. In Tab. 2 below the additional software is sorted by cgx menu items (MI) and cgx commands (C).

[1]**MI** (cgx menu item) [1]**C** (cgx command) of cgx	**Functionality and test examples** (functionality of cgx menu items MI and cgx commands C)	**Name** of additional software required for MI, C
MI "**Hardcopy**" C "**hcpy**"	• *Functionality*. Generate hardcopy (formats, amongst others: tag, ps, gif, png). • *Test example*. Section 9.3.2.	**ImageMagick**
C: "**hcpy make ls**"	• *Functionality*. Bundles ps-plotfiles into at least one file. All ps-plotfiles of some folder are bundled. • *Test example*. Section 9.3.3. Note. PSUtils applied to bundle files, GhostScript for file manipulations and Preview for display of ps-plotfiles.	**PSUtils** **GhostScript** **Preview**
MI: "**Graph**" C: "**graph**"	• *Functionality*. For x-y-plots (path plots, history plots etc.): (1) cgx generates ASCII file with x-y-plot data and gnuplot-command file (script), (2) gnuplot applied to generate ps-plot-file, (3) Preview applied to display ps-plotfile. • *Test example*. Section 9.3.4. Note. Functionality according to points 1, 2 above provided by gnuplot.	**Gnuplot** **Preview**
MI: "**Help**"	• *Functionality*. Amongst others, displays cgx and ccx html-help files. • *Test example*. Section 9.3.5.	**html_browser**
[2]C: "**mesh all**"	• *Functionality*. Generate tet mesh (example of command sequence for tet meshing according 1st column). In particular, the installed routine "ng_vol" allows for tet meshing with a target element size. • *Test example*. Section 9.3.6. Note. Routine "ng_vol" of Netgen applied for tet meshing.	**Netgen**

[1]For details, concerning cgx menu items and cgx commands cf. cgx User's Guide [2].

[2]For details of tet meshing cf. cgx User's Guide [2] and "README" in folder "examples/cad" of cgx.

Table 2 Additional software sorted by cgx menu items MI and cgx commands C. Functionality and references to test examples according to column 2. Names of additional software, which is required for functionality of MI, C listed in column 3. In Tab. 1 above the additional software is sorted by name.

2. SOFTWARE REQUIREMENTS

2.1 Software Releases

Unless otherwise noted, the installation has been tested for the software releases as specified below (cf. titles of subsections below for the releases).

2.2 Mac OS X Specific

2.2.1 OS X Mavericks 10.9.5

The installation was tested for version 10.9.5 of OS X Mavericks. Amongst others, the Unix-like Darwin OS, OpenGL and Aqua are inherent parts of Mac OS X.

The aforementioned OpenGL is required for the installation of cgx, respectively for the GUI of cgx (no separate installation of OpenGL according to [1] required). Aqua is the default GUI on Mac. The additional software gnuplot (cf. Section 1.2 and Tab. 1 above for details) will be installed for Aqua. The command-line interface to OS X is provided by the Terminal App (cf. Section 2.2.2 and Section 3.2 for details).

2.2.2 Xcode 5.1.1 – Provides Command Line Tools (Terminal) and Associated Modified GCC 5.1 (GNU Compiler Collection)

Xcode is a broad suit for software development on Mac and is provided by Apple. Several capabilities of Xcode are required for the present installation as further outlined below. However, because Xcode is not an inherent part of OS X, it needs to be installed separately from OS X (cf. Section 5.1 for the installation).

The following capabilities of Xcode are required for the present installation: (1) command-line tools (CLTs) for OS X, respectively the Terminal App, (2) modified (by Apple) version 5.1 of the GCC (GNU compiler collection). The aforementioned CLTs are provided by the Terminal App (cf. Section 3.2 for details), also denoted as Terminal or Terminal window. As will be shown below, Unix commands for software installation will be entered via Terminal. Furthermore, cgx will be launched via Terminal. Except of the source code of cgx, the aforementioned modified GCC 5.1 will be applied for compilation. For the compilation of cgx the unmodified GCC 4.9 is required because the modified (by Apple) GCC 5.1 is not suitable for the compilation of cgx (cf. Section 2.2.3 and Section 3.1 below for details and Section 5.2 for the installation of the GCC 4.9). The aforementioned CLTs and GCC 5.1 are installed together with Xcode in Section 5.1 below.

Notes. Xcode is a broad suite for software development on OS X. For the installation of cgx, however, only the so called “Command Line Tools“ (CLTs) of Xcode are required (installation of Xcode requires 5 GB, the installation of CLT only ca. 300 MB). Among others, the CLTs include the already aforementioned version 5.1 of the modified GCC. The CLTs provide commands like “make“, whereas the latter is required to build executables from source.

Alternatively to Xcode, only the CLTs may be installed. The installation of the CLTs, however, is not subject of the present installation instructions. Link for download of the CLT:

http://osxdaily.com/2014/02/12/install-command-line-tools-mac-os-x/

2.2.3 GCC 4.9 - Unmodified GNU Compiler Collection for OS X Mavericks

The unmodified GCC 4.9 is required for the compilation of cgx and is not identical with the modified GCC 5.1 from Section 2.2.2 above, which is installed together with Xcode (cf. also explanation of Xcode in Section 2.2.2 above and Section 3.1 below for further details). The installation of GCC 4.9 is subject of Section 5.2.

2.2.4 XQuartz 2.7.5

XQuartz is the implementation of the X-Window-System X11 on Mac. XQuartz, therefore, provides the X11 graphics environment which is required to run the GUI of cgx on Mac. From XQuartz also the command-line interface: “xterm“ is provided. For the present installation, however, the Terminal window will be used instead of xterm. The installation of XQuartz is subject of Section 5.3, because XQuartz is not part of OS X.

2.3 cgx 2.8 - CalculiX GraphiX, Version 2.8

The installation instructions for cgx are contained in Section 4 and Section 8 below. First of all, the source code is installed in Section 4, followed by the build of cgx in Section 8. The source code of cgx is installed in Section 4 prior to the build of cgx in Section 8 because for the installation of the optional additional Netgen software (cf. Tab. 1) a file from the cgx-package is required.

As already explained in Section 1.1 above, optional additional functionality of cgx may be obtained by the installation of the additional software from Section 2.5. The installation of the additional software is subject of Section 7. For the build of cgx in Section 8 all of the additional software is considered with regard to path settings, flags etc. However, none of the additional software needs to be actually installed for the build of cgx. Accordingly, the additional software may be installed (all or partial) before or after the build of cgx.

2.4 MacPorts 2.3.3

MacPorts is required for the installation of some of the additional software. The installation is subject of Section 6 because MacPorts is not part of OS X.

Notes. MacPorts is a tool for compiling, installation and upgrade of open source software on OS X. Similar tools, like Homebrew or Finck may work as well but have not been tested for the present installation of additional software.

2.5 Optional Additional Software – ImageMagick 6.8.8-6, PSUtils p17, gs 9.10, gv 3.7.4, Firefox 35.0, Netgen 5.1, gnuplot 4.6.0

The (optional) additional software is summarized together with details for version and functionality in Tab. 1 and will be installed in Section 7 below (cf. also Section 2.3 above for a brief overview of the installation procedure).

3. PRE-ARRANGEMENTS FOR INSTALLATION

3.1 Default Compilers GCC 5.1 and Compiler g++ 4.9 for Compilation of cgx

The modified GCC 5.1 (GNU compiler collection) which is installed together with Xcode is considered as the default compiler on your Mac (cf. Section 5.1 for the installation of Xcode and details concerning the version and installation path of GCC 5.1). GCC 5.1 includes versions 5.1 of the individual compilers: gcc (GNU c compiler), g++ and clang. The modified GCC 5.1, however, has a problem with function overloading and, therefore, is not suitable for the [2]compilation of cgx.

Because of the aforementioned compiler problem, in addition the unmodified GCC 4.9 was installed and applied for the compilation of cgx. Amongst others, the unmodified GCC 4.9 includes versions 4.9 of the individual compilers: gcc, g++. The unmodified GCC 4.9 will be installed together with other Mac-specific software in Section 5 below and is applied in Section 8.1 for the compilation of cgx.

The present installation of Mac-specific software, therefore, ends up with the installation of two versions of the GCC. The modified GCC 5.1 is installed under the default path: /usr/bin and, therefore, is the default GCC compiler on your Mac. The unmodified GCC 4.9 is installed under the default path: /usr/local/bin and, therefore, is considered as the non-default GCC. For compilation of cgx with g++ 4.9, therefore, the full path: /usr/local/bin/g++ to the executable of g++ has to be specified (cf. the installation of cgx in Section 8.1 below). Furthermore, unless otherwise noted, the default GCC 5.1 will be considered for compilation.

The compilers GCC 5.1 and GCC 4.9 will be installed and tested in Section 5 below. The unmodified GCC 4.9, respectively g++ 4.9.2 will be applied in Section 8.1 below for the compilation of cgx. Unless otherwise noted, the default GCC 5.1 is considered for compilation.

3.2 Terminal, Terminal Commands, HOME, LOGNAME, Login and Permissions

General. The following notes are provided for Mac users who usually don't work in Unix environments. Also search the web for more instructions concerning the below discussed subjects.

Terminal. First of all, as further explained below, "Terminal", respectively the Terminal App is a pre-installed application on your Mac. The notation "Terminal" or "Terminal window" here is also used for the bash (bourne-again shell) Unix-shell which is provided by the Terminal App. Terminal provides the command-line interface to OS X, respectively

[2]In particular, the compilation of function: #define abs(x) ((x) >= 0 ? (x) : -(x)) in the cgx-routine: "extUtil.h" causes a compiler error. A bug fix was not available at the time the installation of cgx was tested.

the interface for the input of Unix-commands which are required for the installation of cgx. Furthermore, cgx may be launched from a Terminal window (cf. Section 8.2 for details).

To open the Terminal application, double click the Terminal-App in your "/Applications/Utilities" folder. The Terminal icon then will appear in your Dock and a Terminal window will pop up on your screen. Use "Help" in the menu bar of "Terminal" to gain further instructions (how to open additional Terminal-windows, how to open further tabs in some window etc.) or visit e.g. the following site for additional instructions: http://www.maclife.com/article/feature/25_terminal_tips_every_mac_user_should_know.

Terminal Commands. Commands, respectively Terminal commands, which are highlighted in the following with red letters, are entered after the $-prompt in the Terminal window (cf. examples below). For execution of commands use the return key (also denoted as enter key).

HOME, LOGNAME. After a new Terminal window was opened, the shell wakes up in your home directory, denoted as HOME (shortcut ~; use keystrokes: alt + N for tilde ~). Use command env (for environment):

$ env <return>

to display the path to your HOME (usually: HOME=~=/Users/LOGNAME). The env-command also displays your LOGNAME as well as the path: PWD to your current (private) working directory. The path to your current working directory, respectively current folder, also may be displayed with command: $ pwd <return> (pwd for "print working directory").

Login. If you are not already logged in with your LOGNAME, use command: $ login <return> in Terminal. Afterwards, you will be asked for your LOGNAME and password to login.

Permissions. For several installation steps you need expanded access rights to the file system. To gain these rights, here the sudo-command will be used; syntax (examples are given below):

$ sudo command <return>.

Afterwards, the OS will ask you first for your password before the command is executed. If required, in the present instruction the sudo-command is set for installation commands. However, if you run into any permission-problems, first try to use sudo as specified above. If sudo doesn't work, you must login as root.

3.3 Editors, .bash_profile, Editing of Hidden File .bash_profile and Usage of .bash_profile to Set Paths and Export Paths

General. The following notes are provided for Mac users who usually don't work in Unix environments. Also search the web for more instructions concerning the below discussed subjects.

Editors. The Finder App, respectively the "Finder" is a convenient tool for file handling on your Mac. However, many of the files on your Mac are hidden in [3]Finder windows, amongst others the file: ".bash_profile" which is required later for the cgx installation. In the following it will be explained how hidden files may be created, opened and edited. Furthermore, the path setting in file: ".bash_profile" is briefly explained.

File .bash-profile. The file .bash-profile is located in your HOME and amongst other settings contains path settings, respectively is used to [4]export paths. The dot in the first place of the file name marks file ".bash_profile" as a hidden file which, therefore, is not displayed in the Finder window. A simple example for a .bash_profile with path settings is given below.

Editing of hidden files, respectively of file .bash_profile. The editing will be demonstrated for setting, respectively exporting paths with .bash_profile. First, execute the following commands in Terminal (don't' type in green # comment; shortcut ~ for your HOME is used; keystrokes for typing tilde ~: ⌥ + n):

```
$ cd ~  <return>    # Change directory (cd) to your home directory ~ (Home),
                    # where your .bash_profile is located.

$ ls -l   <return>  # Long (option -l) list (command ls) displays visible files in ~.
                    # Note that file .bash_profile is not displayed. Command ls –l,
                    # therefore, displays all visible files.

$ ls -la   <return> # Long (option "l") list (ls) for all (option "a" for all visible and
                    # invisible) files in ~. Now, .bash_profile is contained in the list
                    # of files. Command ls –la, therefore, displays all visible and
                    # invisible files.
```

If .bash_profile is not listed after the execution of the last command: ls -la, then create it now with the following commands:

```
$ cd ~  <return>    # Change directory (cd) to your home directory HOME.

$ touch .bash_profile  <return>    # Create .bash_profile.

$ ls -la  <return>    # Verify, if .bash_profile has been created.
```

Next, open .bash_profile with editor TextEdit with commands:

```
$ cd ~  <return>    # Set home directory as your working directory.
```

[3]Note. Commands for displaying hidden files in Finder are available, but for several reasons will be not used here.

[4]Note. If you open a Terminal window, the OS "exports" the path settings found in .bash_profile and then uses these paths to search for executables. Furthermore, amongst others, the path: /usr/bin is a default path which the OS uses to search for executables.

$ open -e .bash_profile <return> # Opens .bash_profile in TextEdit.

Set paths in file .bash_profile and export paths. Now, we are ready for editing of the .bash_profile, respectively to enter a new path-setting in .bash_profile. Next an example is given, how paths may be set in .bash_profile. For path setting you may add the following lines in your .bash_profile:

```
#!/bin/bash

export PATH=$PATH

export PATH=$PATH:/Users/LOGNAME/NETGEN_INSTALL/bin
```

In the above lines, "export" is a Unix command, PATH an environment variable, $PATH the value of that variable, colon ":" is used as separator between paths. The first line updates the PATH variable, the second line sets the added path: /Users/LOGNAME/NETGEN_INSTALL/bin to some executable which is contained in directory "bin". Add as any many paths as you need. Finally, save file .bash_profile and quit TextEdit. The required path settings for the considered software are specified in the installation instructions below.

To activate the changes, respectively to export the new paths in .bash_profile use commands:

$ cd ~ <return> # Set home directory.

$ source .bash_profile <return>

in current Terminal window. Alternatively, you may open a new Terminal window, because as soon as a new Terminal window is opened the OS will automatically execute the commands in .bash_profile, i.e. export path settings etc.).

Control the path settings with the following command:

$ echo $PATH <return>.

Notes. The OS also automatically searches the default paths: /usr/bin as well as /usr/local/bin for executables.
Instead of TextEdit, also one of the command line editors may be used (vim-editor etc.).

4. INSTALLATION OF CALCULIX FILES

4.1 General

In the present section the cgx source code as well as the cgx- and ccx-help files will be downloaded and installed. The installation path may be chosen according to Section 4.2. Furthermore, the file system for the installation of cgx and ccx is explained in Section 4.2.

After the installation of Calculix files continue installation in the sequence of Section 5 to Section 8.

4.2 Installation Path "calculix_inst_path" and File System

Installation Path

Both cgx and cxx are installed for the same installation path. Choose option A or option B for the installation [5]path "calculix_inst_path":

- *Option A* (default installation path):

"calculix_inst_path" =/usr/local

- *Option B* (example for user defined path):

"calculix_inst_path"=~/Applications=/Users/LOGNAME/Applications

Option A is in compliance with the Unix file system hierarchy standard. For option B the path may be chosen freely. For option B an example is given above, for which cgx will be installed in directory "Applications" under your home directory "~" (cf. Section 3.2 for the home directory).

Furthermore, the path according to option A is default for the installation of Calculix and, therefore, in the source code of ccx and cgx all paths are already set in accordance with option A. If option B is used instead, some paths must be set manually for the installation of cgx (cf. Section 8.1 below for details).

Prior to the installation the directories of the path "calculix_inst_path" must exist. Create them now, if necessary. Use the Finder to create directories or the sequence of Unix commands: $ cd "path_to_Directory" <return> (change to directory: "path_to_Directory" under which a subdirectory has to be created), then type: $ mkdir "dir_name" <return> to create the subdirectory "dir_name". Furthermore, use: $ ls –l <return> to list contents of some directory, use sudo before commands (e.g. sudo mkdir) if the permission for making of directories is denied.

[5]Note. The notation "calculix_inst_path" here is used for documentation reasons only and, therefore, is not an environment variable for the installation of cgx. Otherwise stated, replace "calculix_inst_path" in commands with full path specification according to options A, B.

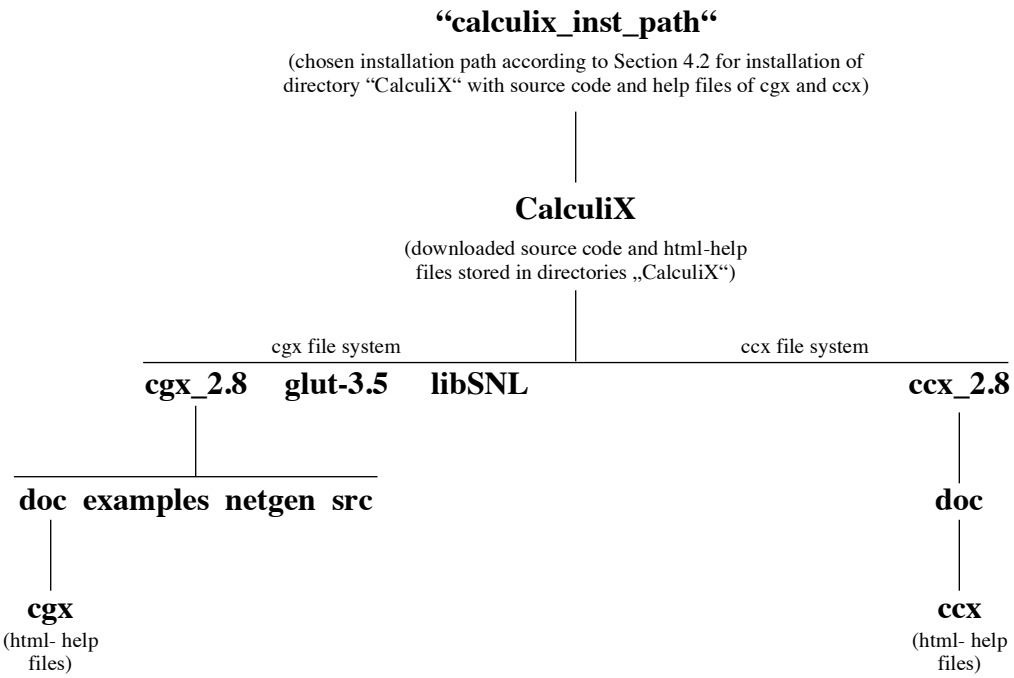

Figure 1 File system after installation of source code and html-help files of cgx as well as ccx html-help files.

File System

The file system for the installation of cgx and ccx is depicted in Fig. 1. After the installation, source code and html-help files of cgx and ccx are contained in directories “CalculiX”. Install directories “CalculiX” under your installation path as explained in the subsequent subsections. The resulting file system is depicted in Fig. 1. Accordingly, both cgx and ccx are installed under path: “calculix_inst_path”/CalculiX/. The file system will be further explained below.

4.3 Install cgx Files

Steps a, b:

(a) Download cgx_2.8 source code from Calculix site: http://www.dhondt.de/. The cgx-package: cgx_2.8.all.tar.bz2 will be downloaded to your Desktop (directory: ~/Desktop) and unpacked to folder: “CalculiX“. If not, unpack by double clicking the package icon of cgx_2.8.all.tar.bz2.

(b) Move folder “CalculiX“ from Desktop to “calculix_inst_path” using drag and drop or with Terminal command (“calculix_inst_path” according to Section 4.2 above):

$ mv -r ~/Desktop/CalculiX “calculix_inst_path” <return>

The source code of cgx now is contained in directory: “cgx_2.8” under directory “CalculiX” (cf. Fig. 1). In addition the directories for the libraries glut-3.5 and libSNL are installed under directory: “CalculiX” (cf. Fig. 1). The aforementioned directory: “cgx_2.8”

contains directory “src” with the source code of cgx as well as directory “doc” for the documentation, directory “examples” with examples and directory “netgen” with netgen-files (cf. Fig. 1).

4.4 Install cgx html-Help Files

In cgx a help menu for cgx is provided. For the aforementioned menu the html-help files of cgx are required.

Steps a, b for installation of the cgx html-help files:

(a) Download htlm-help files: “the documentation html“ of cgx from Calculix website: http://www.dhondt.de/. The help file package: cgx_2.8.htm.tar.bz2 will be downloaded to your Desktop (directory: ~/Desktop). If the ending .bz2 is missing, please rename the filename to cgx_2.8.htm.tar.bz2.

(b) The package cgx_2.8.htm.tar.bz2 next will be moved from the Desktop to the “calculix_inst_path” and then will be unpacked and installed. Type and execute the following Terminal commands (without comments, “calculix_inst_path” chosen according to Section 4.2 above):

```
$ cd “calculix_inst_path” <return>    # Change to Calculix installation directory.

# Now move (mv) cgx_2.8.htm.tar.bz2 from Desktop to “calculix_inst_path”:

$ mv ~/Desktop/cgx_2.8.htm.tar.bz2 “calculix_inst_path”  <return>

$ bunzip2 cgx_2.8.htm.tar.bz2 <return>    # Unpack package.

$ tar -xvf cgx_2.8.htm.tar  <return>    # Install package.
```

The last command installs the cgx html-help files of cgx. The html-help files are now contained in directory “cgx” under the previously installed directory: “doc” (cf. Fig. 1 for the file system):

“calculix_inst_path”/CalculiX/cgx_2.8/doc/cgx

If option B from Section 4.2 was chosen for the installation path the above specified path is required in Section 8.1 to complete the installation of cgx. The download packages are no longer required and, therefore, may be deleted.

4.5 Install ccx html-Help Files

In cgx a help menu for ccx is provided. For the aforementioned menu the html-help files of ccx are required.

Steps a, b for installation of the ccx html-help files:

(a) Download htlm-help files: “the documentation html“ of ccx from Calculix website: http://www.dhondt.de/. The help file package: ccx_2.8.htm.tar.bz2 will be downloaded to your Desktop (directory: ~/Desktop). If the ending .bz2 is missing, please rename the filename to ccx_2.8.htm.tar.bz2.

(b) The package ccx_2.8.htm.tar.bz2 next will be moved from the Desktop to the calculix_inst_path and then will be unpacked and installed. Type and execute the following Terminal commands (without comments, “calculix_inst_path” chosen according to Section 4.2 above):

```
$ cd “calculix_inst_path” <return>    # Change to Calculix installation directory.

# Now move (mv) ccx_2.8.htm.tar.bz2 from Desktop to “calculix_inst_path”:

$ mv ~/Desktop/ccx_2.8.htm.tar.bz2 “calculix_inst_path”  <return>

$ bunzip2 ccx_2.8.htm.tar.bz2 <return>    # Unpack package.

$ tar -xvf ccx_2.8.htm.tar  <return>    # Install package.
```

The last command installs the ccx html-help files which are required from cgx. The html-help files are now contained in directory “ccx” under directory: “../ccx_2.8/doc” (cf. also Fig. 1 for the file system):

“calculix_inst_path”/CalculiX/ccx_2.8/doc/cgx

If option B from Section 4.2 was chosen for the installation path the above specified path is required in Section 8.1 to complete the installation of cgx. The download packages are no longer required and, therefore, may be deleted.

5. INSTALL MAC-SPECIFIC SOFTWARE

5.1 Xcode 5.1.1 and Modified GCC 5.1

Xcode may be downloaded for free from “Categories“ in the App Store. Follow installer instructions. The Xcode App will be installed in your “Applications“ folder.

As already mentioned before in Section 2.2.2, the modified GCC 5.1 (GNU compiler collection) comes together with Xcode and is installed under the default path: /usr/bin. The modified GCC 5.1 includes the individual compilers: gcc (GNU c-compiler), g++, Clang.

The installation of GCC 5.1 now will be tested exemplarily for g++. If the installation was successful, after the execution of the Terminal command: $ which g++ <return> the path to the default g++ compiler is displayed as follows:

```
/usr/bin/g++
```

After execution of the Terminal command: $ g++ -v <return> (option -v for version) the screen output for the version of g++ should look as follows:

```
Configured with: --prefix=/Applications/Xcode.app/Contents/Developer/usr --with-gxx-include-dir=/usr
                                                                          /include/c++/4.2.1
Apple LLVM version 5.1 (clang-503.0.40) (based on LLVM 3.4svn)
Target: x86_64-apple-darwin13.2.0
Thread model: posix
```

As already explained in Section 3.1 above, in addition the unmodified GCC 4.9 is required for the compilation of cgx and, therefore, will be installed next.

5.2 Unmodified GCC 4.9

For the installation of the unmodified GCC 4.9 (GNU compiler collection) point 2 of the instructions: https://wiki.helsinki.fi/display/HUGG/Installing+the+GNU+compilers+on+Mac+OS+X for OS X Mavericks is applied.

Installation steps 1 – 4:

(1) Download pre-build GCC 4.9, respectively gunzipped file gcc-4.9-bin.tar.gz from site:
http://sourceforge.net/projects/hpc/files/hpc/gcc/gcc-4.9-bin.tar.gz/download?use_mirror=cznic&download=

(2) Open Terminal window and change to directory where gcc-4.9-bin.tar.gz was downloaded (usually directory "Downloads" or directory "Desktop"). If for example gcc was downloaded on your Desktop use terminal command:

$ cd ~/Desktop <return>.

(3) If gcc-4.9-bin.tar.gz was not automatically unzipped, then unzip file with Terminal command:

$ gunzip gcc-4.9-bin.tar.gz <return>

(4) For installation of GCC 4.9 use terminal commands:

$ sudo tar xvf gcc-4.9-bin.tar -C / <return> # Installs GCC 4.9 under default directory:
# /usr/local/bin.

After step 4, amongst others the compilers gcc and g++ are installed in /usr/local/bin.

The test of the installation will be explained exemplarily for g++. If the installation was successful the Terminal command: $ which /usr/local/bin/g++ <return> will display the path to the unmodified GNU g++ compiler as follows (screen output):

```
/usr/local/bin/g++
```

Furthermore, after execution of the Terminal command: $ /usr/local/bin g++ -v <return> (option -v for version) the screen output for the version of g++ should look as follows:

```
Using built-in specs.
COLLECT_GCC=/usr/local/bin/g++
COLLECT_LTO_WRAPPER=/usr/local/libexec/gcc/x86_64-apple-darwin14.0.0/4.9.2/lto-wrapper
Target: x86_64-apple-darwin14.0.0
Configured with: ../gcc-4.9-20141029/configure --enable-languages=c++,fortran
Thread model: posix
gcc version 4.9.2 20141029 (prerelease) (GCC)
```

As already explained in Section 3.1 above, the unmodified g++ 4.9.2 will be applied for the compilation of cgx in Section 8.1 below. The path for g++: /usr/local/bin/gcc will be set accordingly in the Makefile of cgx (cf. Section 8.1 for details).

5.3 XQuartz 2.7.5

Download XQuartz from site: http://www.macupdate.com/app/mac/26593/xquartz and then follow installer instructions. The XQuartz-App will be installed in your "/Applications/Utilities" folder.

6. INSTALL MACPORTS

Download MacPorts for OSX 10.9 Mavericks from site: http://www.macports.org/install.php. Then follow installer instructions.

During installation your .bash_profile will be modified automatically. For activation of the modifications either close current Terminal window and then start a new one or follow the instructions in Section 3.3 above.

For the test of the installation type in Terminal window:

$ sudo port self update <return>

The installation works if the first line of the screen output looks as follows:

```
--->  Updating MacPorts base sources using rsync
```

Further instructions are provided on site: https://www.macports.org/.

7. INSTALL ADDITIONAL SOFTWARE

7.1 General

As already explained in detail in Section 1.1 above, depending on your needs none, all or parts of the additional software may be installed. Furthermore, the sequence of installation may be chosen freely. The functionality of cgx which is provided by the additional software is listed in Tab. 2.

Furthermore, for the installation of cgx in Section 8.1 none of the additional software needs to be actually installed. However, for the installation of cgx in Section 8.1 all parameters (paths, flags etc.) for the additional software are set. The latter allows for the later installation of additional software without re-compilation of cgx.

7.2 ImageMagick 6.8.8-6

- From ImageMagick the command-line tool “convert“ is provided. (cf. Tab. 1 for the functionality of “convert“ in cgx).
- Download package: “ImageMagick 6.8.8-6 for Mac OS X 10.5 - 10.9 - Requires XQuartz“ from site: http://cactuslab.com/imagemagick/. Then follow installer instructions.
- During installation your .bash_profile will be modified automatically. For activation of the modifications either close current Terminal window and then start a new one or follow the instructions in Section 3.3 above.
- For the test of the installation use Terminal command:

  $ convert <return>

  The installation works if the first line of the screen output looks as follows (followed by several other lines for the usage of ImageMagick):

  ```
  Version: ImageMagick 6.8.8-6 Q16 x86_64 2014-02-17 http://www.imagemagick.org
  ```

- The functionality of cgx which is provided by the now installed “convert“ command-line tool will be tested in Section 9.3.2 below.

7.3 PSUtils p17

- From PSUtils the command-line tool “pstops“ is provided (cf. Tab. 1 for the functionality of “pstops“ in cgx). PSUtils is installed with MacPorts.
- First run self update of MacPorts with Terminal commands:

  $ sudo port selfupdate <return>

$ sudo port upgrade outdated <return>

- Next install PSUtils with command:

  $ sudo port install psutils <return>

- During installation your .bash_profile will be modified automatically. For activation of the modifications either close current Terminal window and then start a new one or follow the instructions in Section 3.3 above.

- For the test of the installation use Terminal command:

  $ pstops <return>

  The installation works if the first line of the screen output looks as follows:

  ```
  pstops release 1 patchlevel 17
  ```

- The functionality of cgx which is provided by the now installed "pstops" command-line tool will be tested in Section 9.3.3 below.

7.4 gs 9.10

- From gs 9.10 (GhostScript) the command-line tool: "gs" is provided (cf. Tab. 1 for the functionality of "gs" in cgx).

- Download package-installer for gs 9.10 from site: http://pages.uoregon.edu/koch/.

- Choose: "Open with installer (default)".

- After the download is completed the installer window opens automatically. Follow the installer instructions.

- GhostScript is installed for the default path: /usr/local. Accordingly, the executable: "gs" is located in /usr/local/bin/gs. No further path-settings required.

- For the test of the installation use Terminal command:

  $ gs –v <return>

  The installation works if the first line of the screen output looks as follows:

  ```
  GPL Ghostscript 9.10 (2013-08-30)
  ```

- The functionality of cgx which is provided by the now installed "gs" command-line tool will be tested in Section 9.3.3 below.

7.5 gv 3.7.4 – Preview Installed Instead

- The application program: "gv" is a postscript viewer. However, for the reasons explained in the note below, for the present installation the Mac-App "Preview" will be in-

stalled as postscript viewer for cgx instead of “gv“ (cf. also Tab. 1 for the functionality of “Preview“ in cgx). For the installation a shell script is provided which enables cgx to use Preview.

Note. The postscript viewer “gv“ is based on the older viewer: “Ghostview“ which is set in cgx as the default viewer. As an alternative, “gv“ may be set instead of “Ghostview“ (cf. Section 8.1 for details). More information about the usage of “gv“ and “Ghostview“ is available in Section 8.26 in [2].

In June 2014, i.e. at the time when the installation of cgx on Mac OS X has been first accomplished, neither an installation of “Ghostview“ nor of “gv“ seemed to work on OS X. The situation was as follows:

- Build of “Ghostview“ or “gv“ from source seems to be a task of the very specialist because of the many dependencies of the afore mentioned packages. Build from source, therefore, has been discarded.
- Adequate binaries for “gv” and “Ghostview“ were not available.
- For “Ghostview“no installer was available. For “gv“ a MacPort exists. However, the installation with Terminal command: $ sudo port install gv <return> failed.

To avoid the installation problems with “Ghostview“ and “gv“, therefore, the Mac Preview App has been chosen as postscript viewer.

- Next, a shell script will be provided and installed in /usr/local/bin which enables cgx to use “Preview“ as postscript viewer. The filename of the aforementioned script is also the name of the viewer. Accordingly, the name of the viewer will be set for the installation of cgx in Section 8.1 below.

  Installation steps a, b:

  (a) Create the shell script, respectively a file with filename: “preview“ (or any other filename you like) in a directory of your choice: “dir_your_choice“ (for example your HOME: /Users/LOGNAME) by using an editor of your choice. Insert the following lines in file “preview“, then safe and quit the file:

```
#!/bin/bash

# Shell script allows launch of Preview from Terminal window with

# command: $ preview “filename or path to file“ <return>

# "open" opens Preview as default viewer, filename or path to file passed via $1

open $1
```

  (b) Enter the following commands in Terminal window (without comments and full path: "dir_your_choice" as chosen above):

```
$ cd “dir_your_choice” <return>  # Change directory to “dir_your_choice“.

$ chmod 777 preview <return>  # Make file “preview“ executable.
```

```
$ cp preview /usr/local/bin  <return>   # Copy preview in: /usr/local/bin which is
                                        # one of the default search paths of the
                                        # OS for executables.
```

Note. An alternative installation of the script is explained under point (b) in Section 7.6 below (cf. the "Note" at the end of point (b) in Section 7.6 for details).

- The filename of the shell-script from point (a) above now is the name of the postscript viewer and is required in Section 8.1 below to complete the installation of cgx.
- Consider that a postscript file: "test.ps" is located on your desktop. For the test of the installation then use Terminal commands:

  $ cd ~/Desktop <return>

  $ preview test.ps <return>

  If the installation was successful, after the execution of the last command the file: "test.ps" will be displayed with Preview.
- The functionality of cgx which is provided by the now installed "preview" command-line tool will be tested in Section 9.3.3 and Section 9.3.4 below.

7.6 Firefox 35.0 (or any other html-browser)

- In cgx a html-browser is used to display the cgx- and ccx-help files (cf. also Tab. 1 for the functionality of the html-browser). For the sake of simplicity the installation will be demonstrated for Firefox. For other browsers, replace "Firefox" with the name of your browser (Safari etc.) in the instructions below.

  It is assumed, that Firefox is already installed. The Firefox-App then is located in your "/Applications" folder.
- Next, a shell script is provided which enables cgx to open Firefox. The script will be installed in /usr/local/bin (cf. the note below for an alternative installation of the script). The filename of the script is also the name of the html-browser. Accordingly, the name of the browser will be set for the installation of cgx in Section 8.1 below.

  Installation steps a, b:

  (a) Create the shell script, respectively a file with filename "firefox" (or any other filename) in a directory of choice directory: "dir_your_choice" (for example your HOME: /Users/LOGNAME) by using an editor of your choice. Insert the following lines in file: "firefox", then safe and quit the file:

  ```
  #!/bin/bash

  # Shell script enables launch of Firefox from Terminal window.

  # Command for Terminal window: $ Firefox <input>
  ```

```
# "open" opens application (-a) Firefox, input passed via $1

open -a Firefox $1
```

(b) Enter the following commands in Terminal window (without comments and full path "dir_your_choice" as chosen above):

```
$ cd dir_your_choice <return>  # Change directory to "dir_your_choice".

$ chmod 777 firefox <return>  # Make "firefox" executable.

$ cp firefox /usr/local/bin  <return>  # Copy preview in /usr/local/bin which is
                                       # one of the default search paths of the
                                       # OS for executables.
```

Note. Instead of installing the script as explained above in /usr/local/bin, you may install all your scripts in one place, for example in a folder "scripts" which for example may be created in your HOME. The installation path of your scripts then is: scripts_install_path=HOME/scripts. Installation steps, respectively commands to enter in Terminal window, followed by <return>: (b1) cd HOME, (b2) mkdir scripts, (b3) cp dir_your_choice/firefox scripts_install_dir, (b4) chmod 777 firefox, (b5) insert line:

export PATH=$PATH:scripts_install_path

in your .bash_profile (cf. Section 3.3 for further explanation of .bash_profile), (b6) use instructions from Section 3.3 above to activate the changes you made in .bash_profile. Amongst others, the OS now uses the path according to step (b5) to search for the executable: "firefox".

- The filename of the shell-script from point (a) above now is the name of the postscript viewer and is required in Section 8.1 below to complete the installation of cgx.
- Consider that a html-file: "test.html" is located on your desktop. For the test of the installation then use Terminal commands:

  ```
  $ cd ~/Desktop <return>

  $ firefox test.html <return>
  ```

  If the installation was successful, after the last command the file "test.html" will be displayed with Firefox.
- The functionality of cgx which is provided by the now installed "firefox" command-line tool will be tested in Section 9.3.5 below.

7.7 Netgen-5.1

From Netgen (automatic 3D tetrahedral mesh generator) the command-line tool: "ng_vol" is provided which is used from cgx for tet meshing with a given target element size (cf. also Tab. 1 for the functionality of "ng_vol" in cgx).

The installation of Netgen with and without GUI is considered. The installation without GUI is sufficient to provide the command-line tool: “ng_vol“ for cgx.

Installation steps 1 to 8 (step 5 is for the installation of Netgen with and without GUI):

(1) Choose installation path “netgen_install_path”. For example use:

“netgen_install_path”=/users/LOGNAME/NETGEN_INSTALL.

to install Netgen in directory: “NETGEN_INSTALL” under your home directory “~”=”Users/LOGNAME”. Create directory “NETGEN_INSTALL” with Terminal commands: $ cd ~ <return> and then: $ mkdir “NETGEN_INSTALL” <return>.

Note. “netgen_install_path“ is used as an abbreviation for the installation path. In the Terminal commands given below, therefore, replace the aforementioned abbreviation with the full path specification: “netgen_install_path“=/Users/LOGNAME/netgen_install.

(2) Download Netgen 5.1 source for Linux from site:

http://sourceforge.net/apps/mediawiki/netgen-mesher/index.php?title=Main_Page.

The package netgen-5.1.tar.gz will be downloaded on your Desktop.

(3) Unpack package by double clicking the icon of netgen-5.1.tar.gz on your Desktop. Then drag folder netgen-5.1 from Desktop to “netgen_install_path”. The folder netgen-5.1 now is installed as follows: “netgen_install_path”/netgen-5.1.

(4) Use modified file “ng_vol.cpp” from the cgx package to build Netgen. Steps:

(4.1) Open file ng_vol.cpp in cgx installation directory: “calculix_install_path”/calculix/cgx_2.7/netgen with an editor of your choice.

Next, try to find character string (without quotes): “secondorder“ with your editor. The aforementioned string may appear either only once or not at all. If “secondorder“ is found (code in ng_vol.cpp not O.K.), then replace it now with (without quotes): “second_order“, then save file and quit editor. If “secondorder“ is not found (code in ng_vol.cpp is already O.K.), then quit editor.

(4.2) Rename file: “ng_vol.cpp” in the netgen-package as “ng_vol_netgen_vers.cpp” if you want to keep it. The aforementioned file “ng_vol.cpp” is located under the path: “netgen_install_path”/netgen-5.1/nglib in the netgen-package. Use the Finder to rename the file or Terminal command:

$ mv “path”/ng_vol.cpp “path”/ng_vol_netgen_vers <return>

with “path”=”netgen_install_path”/netgen-5.1/nglib.

(4.3) Copy modified file “ng_vol.cpp” (cf. point (4.1) above) from the cgx package to the location of the netgen installation: “netgen_install_path”/netgen-5.1/nglib with the Finder or with Terminal command:

$ cp “cgx_path”/ng_vol.cpp “netgen_path” <return>,

with “cgx_path”=”calculix_install_path”/calculix/cgx_2.7/netgen/ng_vol.cpp and “netgen_path”=”netgen_install_path”/netgen-5.1/nglib.

(5) Install Netgen. For the installation of Netgen without GUI follow the instructions under point (5.1) below and for the installation with GUI point (5.2). The installation without GUI is sufficient to provide the command-line tool: “ng_vol“ for cgx.

(5.1) Install Netgen without GUI. Use Terminal commands without comments:

$ cd “netgen_install_path”/netgen-5.1 <return> # Change directory to location
# of netgen installation.

The next command creates the Makefile:

$ ./configure --prefix="${HOME}/netgen_install" --disable-gui <return>

$ sudo make <return> # Build Netgen. Displayed warnings are not essential for
# the functionality of Netgen.

$ sudo install <return> # Install Netgen. Installs binaries under path:
# “netgen_install_path”/bin.

The binaries are now located under path: “netgen_install_path”/bin.

If you want to repeat the installation for some reason, then run before in Terminal window:

$ make distclean <return>.

Continue installation with point 6 below.

(5.2) Install Netgen with GUI. Follow the instructions in Appendix 2.

Continue installation with point 6 below.

(6) Set path to binaries in “netgen_install_path”/bin. Insert the following line at the end of your .bash_profile (cf. Section 3.3 for further explanation of .bash_profile):

export PATH=$PATH:”netgen_install_path”/bin

Next, activate path (use instructions from Section 3.3).

(7) Test Installation of command-line tool: “ng_vol”. Use Terminal command without comment:

$ ng_vol <return> # Execute binary ng_vol.

The installation works, if after execution of the last command the following screen output appears:

```
Netgen tet-mesher

use: ng_vol filename
```

(8) The functionality of cgx which is provided by the now installed binary: “ng_vol“ will be tested in Section 9.3.6 below (cf. also Tab. 1 for the functionality of “ng_vol“ in cgx).

7.8 gnuplot 4.6.5

From gnuplot (command-driven, interactive function and data plotting program for 2D- and 3D plots) the command-line tool: “gnuplot“ is provided which is used from cgx to display xy-plots (cf. also Tab. 1 for the functionality of “gnuplot“ in cgx).

Instructions for the installation of gnuplot with MacPorts and from source are provided. First, try to install gnuplot with MacPorts. If the latter fails, install it from source.

Installation steps 1 to 4 (step 1 for installation with MacPorts, step 2 for installation from source if step 1 doesn’t work, steps 3, 4 for the test of the installation):

(1) Install gnuplot with MacPorts. Use Terminal commands without comments:

$ sudo port selfupdate <return>

$ sudo port upgrade outdated <return>

$ sudo port install gnuplot <return> # Install gnuplot.

If the installation with MacPorts does not work, try to install gnuplot from source according to step 2 below.

If the installation is finished without error-messages, MacPorts has installed the executable: “gnuplot“ for the default path: /opt/local/bin/gnuplot. Furthermore, MacPorts automatically adds the search path: /opt/local/bin for the executable: “gnuplot“ to the path-settings in your .bash_profile. To activate the modified path-setting follow the instructions in Section 3.3.

Continue with the test of the executable “gnuplot” according to step 3 below.

(2) Install gnuplot from source, if installation with Macports according to point 1 above has failed. Installation steps 2.1 to 2.3:

(2.1) Choose installation path “gnuplot_install_path”. For example use:

“gnuplot_install_path”=~/GNUPLOT_INSTALL.

to install gnuplot in directory: “GNUPLOT_INSTALL” under your home directory “~”. Create directory “GNUPLOT_INSTALL” with Terminal commands: $ cd ~ <return> and then: $ mkdir “GNUPLOT_INSTALL” <return>.

Note. “gnuplot_install_path“ is used as an abbreviation for the installation path. In the Terminal commands given below, therefore, replace the aforementioned abbreviation with the full path specification: “gnuplot_inst_path”=/Users/LOGNAME/gnuplot_install.

(2.2) Download and install source code. Download the package: gnuplot-4.6.5.tar.gz from website: http://sourceforge.net/projects/gnuplot/files/gnuplot/4.6.5/. The zipped tar-file will be downloaded on your Desktop. Double-click package: gnuplot-4.6.5.tar.gz on your Desktop to unzip, then drag unzipped folder: “gnuplot-4.6.5” to installation directory: „gnuplot_inst_path“. The file system now looks like this: “gnuplot_inst_path”/gnuplot-4.6.5.

(2.3) Build and install gnuplot. Use Terminal commands without comments:

$ cd “gnuplot_inst_path”/gnuplot-4.6.5 <return> # Set installation directory.

$ ./configure <return> # Configures makefile.

$ sudo make <return> # Builds binary gnuplot.

$ sudo install <return> # Installs binary: “gnuplot” under default path:
# /usr/local/bin/gnuplot.

Binary: “gnuplot”, therefore, is installed under the default path: /usr/local/bin.

(2.4) Path-settings. No additional path-setting required, because the binary: “gnuplot” according to point (2.3) above is installed in directory: /usr/local/bin and, therefore, already is installed in one of the default search paths of the OS for binaries.

(3) Test installation of binary: “gnuplot” and bug fixes. Launch gnuplot from some working directory with Terminal command:

$ gnuplot <return>

If the installation was successful the screen output now looks as follows:

```
        G N U P L O T
        Version 4.6 patchlevel 5    last modified February 2014
        Build System: Darwin x86_64

        Copyright (C) 1986-1993, 1998, 2004, 2007-2014

        Thomas Williams, Colin and many others
        gnuplot home:     http:// Kelley www.gnuplot.info
        faq, bugs, etc:   type "help FAQ"
        immediate help:   type "help"  (plot window: hit 'h')

Terminal type set to 'aqua'

gnuplot>
```

and the prompt: “gnuplot>” for the input of gnuplot commands appears in the last line of the output as shown above. Furthermore, according to the screen output above, terminal type Aqua (preferred GUI for Mac) is set. Use gnuplot command: gnuplot> exit <return> to exit gnuplot.

If, however, after the command: $ gnuplot <return> the screen message appears:

```
dyld: Library not loaded: /opt/local/lib/libgd.2.dylib
Referenced from: /usr/local/bin/gnuplot
Reason: image not found
Trace/BPT trap: 5,
```

gnuplot couldn’t find the library libgd.2.dylib. The library-problem may be fixed with the following link:

$ sudo ln -s /opt/local/lib/libgd.3.dylib /opt/local/lib/libgd.2.dylib <return>

Next, launch gnuplot again and check screen output as already explained above.

Note. It is possible that besides the above mentioned library libgd.2.dylib another library is missing. In the latter case, first try to find the missing library on your Mac (use Finder or Unix command to find) and the path to this library. Then link the library analogous as explained above for libgd.2.dylib.

(4) The functionality of cgx which is provided by the now installed binary “gnuplot“, will be tested in Section 9.3.4 below (cf. also Tab. 1 for the functionality of “gnuplot“ in cgx).

8. INSTALL AND LAUNCH CGX, TEST INSTALLATION, GETTING STARTED AND TEST EXAMPLES

8.1 Install cgx

The source files of cgx already have been installed in the previous Section 4 for the installation path:

"calculix_install_path" (cf. Section 4.2 for details).

In the present section the installation will be completed. The latter first requires some minor modifications of files and finally to build and install cgx. Furthermore, for the installation of cgx the parameters (flags, paths etc.) of all of the optional additional software from Section 2.5 will be set. However, none of the additional software actually needs to be installed for the compilation of cgx. Accordingly, depending on your needs, you may install the additional software later (none, all or partial) without re-compilation of cgx (cf. also Section 7.1 for details).

Installation steps (1) to (5):

(1) *Modifications in file: "Makefile"*. The "Makefile" of cgx is located in directory: "src":

"calculix_install_path"/calculix/cgx_2.8/src/Makefile.

Modifications:

+ Open file "Makefile" with an editor of your choice.
+ Replace "lib64" with "lib" (without quotes) in the following two existing lines:

  -L/usr/lib64 -lGL -lGLU \

  -L/usr/X11R6/lib64 -lX11 -lXi -lXmu -lXext -lXt -lSM -lICE \

  Note. Link to directory: "lib64" required for 64 bit systems running under Linux only.
+ Delete "-lrt" in the following existing line:

  -lm -lpthread –lrt

  Note. Linker flag: "-lrt" for run time library (rt) required for compilation under Linux only.
+ According to Section 2.2.3, cgx will be compiled with the unmodified compiler g++ 4.9.2. Replace, therefore, "g++" with: "/usr/local/bin/g++" (without quotes) in the existing last line of "Makefile":

  g++ $(OULIB) $(OLIB) $(OUTIL) $(CFLAGS) $(LFLAGS) -o $@

With: “/usr/local/bin/g++” the unmodified g++-compiler will be invoked (cf. also Section 5.2 for the installation of the aforementioned compiler and the installation path: “/usr/local/bin” for the compiler: g++ 4.9.2).

+ Save file “Makefile” and quit editor.

(2) *Modifications in file: “cgx.h“*. The header file: “cgx.h” of the cgx source is located in directory “src”:

“calculix_install_path”/calculix/cgx_2.8/src/cgx.h.

Modifications:

+ Open file “cgx.h“ with an editor of your choice.

+ Set name of browser according to Section 7.6. Skip this step, if the name of your chosen browser is “firefox” (default browser name). Otherwise, replace pre-set browser name: “firefox” with name of your browser “your_browser_name” (without quotes) in the following two existing lines in cgx.h:

```
#define    BROWSER       {"firefox"}
```

+ Set paths to html-help files in accordance with installation of help files in Section 7.6 (cf. also Fig. 1 for the file system). Skip this step, if the html-help files are installed under the default path according to option A from Section 4.2:

Option A: “calculix_install_path”=/usr/local.

Otherwise, i.e. for option B from Section 4.2 set the chosen: “calculix_install_path” in “cgx.h” as follows: replace the five existing entries: “/usr/local“ with the full path of the chosen “calculix_install_path”. For example, replace “/usr/local“ with “calculix_install_path”=/Users/LOGNAME/Applications according to option B from Section 4.2.

+ Set name “preview” of psviewer (postscript viewer) as previously chosen in Section 7.5. Replace “gv” with chosen name: “preview” in the following two existing lines of cgx.h:

```
#define    PSVIEWER      {"gv"}
```

+ Save file “cgx.h” and quit editor.

(3) *Modifications in file “setFunctions.c “*. The file: “setFunctions.c” of the cgx source is located in directory “src”:

“calculix_install_path”/calculix/cgx_2.8/src/setFunctions.c.

Modifications:

+ Open file “setFunctions.c“ with an editor of your choice.

+ In “setFunctions.c”, replace the three existing entries: “pswrite” with: “ps2write” (without quotes).

  Note. With “ps2write“ the output device for gs (GhostScript) is set.

+ Save file “setFunctions” and quit editor.

(4) *Build and install cgx*. Use below specified Terminal commands (without comments) and replace “calculix_install_path“ in commands with full path specification according to Section 4.2.

Installation steps:

```
$ cd “calculix_install_path”/calculix/cgx_2.7/src <return> # Change to directory with
                                                          # source code and makefile.

$ sudo make <return> # Build cgx. Some warnings will appear which are not
                     # relevant for the functionality of cgx.

$ sudo rm *.o <return> # Remove object files. Use command always before
                       # re-build.
```

Command “make“ builds the binary “cgx“ in directory “src”:

“calculix_install_path”/calculix/cgx_2.7/src/cgx.

Before re-build of cgx, remove object files with Terminal command: $ sudo rm *.o <return> as already specified above.

(5) *Install binary of cgx in: /usr/local/bin*. First, check if: /usr/local/bin already exists with Terminal command: $ cd /usr/local/bin <return>. If directory “bin” already exists, then continue with: “Further steps” below. Otherwise, i.e. if the screen message: “No such file or directory” appears, then create: /usr/local/bin now with Terminal command: $ sudo mkdir /usr/local/bin <return>.

Further steps:

```
$ cd “calculix_install_path”/calculix/cgx_2.8/src <return>  # Change to directory with
                                                            # binary “cgx”.

$ sudo cp cgx /usr/local/bin/cgx_2.8 <return> # Create copy: “cgx_2.8” of binary
                                              # “cgx” in: /usr/local/bin.
```

The copy: “cgx_2.8” of the binary “cgx“ in: /usr/local/bin/cgx_2.8 enables to launch cgx from working directories. You also might change the protection with command: $ sudo chmod ao+rx /usr/local/bin/cgx_2.8 <return>.

Alternatively, instead of copying the binary cgx in: /usr/local/bin as explained above, you may set the path: “calculix_install_path”/calculix/cgx_2.7/src to the binary of

"cgx" in your .bash_profile (cf. Section 3.3 and Section 7.7 for instructions and examples). Furthermore, use commands:
$ cd "calculix_install_path"/calculix/cgx_2.7/src <return> and then: $ sudo mv cgx cgx_2.8 <return> to rename the binary for example from "cgx" to "cgx_2.8".

8.2 Launch cgx and Test Installation

Steps:

(1) Open command window. Open either a Terminal- or xterm-window to enter commands:

+ Open a Terminal window. Follow the instructions in Section 3.2 to open a bash-shell for entering of commands.

+ Open a xterm window. Launch XQuartz by double clicking the XQuartz App in your "/Applications/Utilities" folder. A xterm window will be opened automatically.

(2) Create a working directory. Create e.g. working directory: "test" in your home-directory with command (enter command in Terminal- or xterm-window): $ mkdir ~/test <return> and then change directory to directory "test" with command: $ cd ~/test <return>.

(3) Launch cgx and test installation. Launch cgx with command:

$ cgx_2.8 -b dummy.fbd <return>

The installation of cgx basically works, if after the execution of the last command the following message appears in the command window:

```
on a Darwin machine, nodename NNs-MacBook-Pro.local, release 13.4.0, version Darwin
Kernel Version 13.4.0: Sun Aug 17 19:50:11 PDT 2014; root:xnu-
2422.115.4~1/RELEASE_X86_64, machine x86_64
parameters:3 arguments:2
GL_MAX_EVAL_ORDER:10

ERROR: The input file "dummy.fbd" could not be opened.
```

and the Calculix Graphix window [6]pops up on your screen. The above error appears, if the file "dummy.fbd" not already exists. Position cursor in Calculix Graphix window and then type command: quit <return> to terminate cgx. The echo of the command is displayed in the command window.

See introductory Sections 4 to 6 in the cgx User's Manual [2] and also the subsequent Section 8.3 below how to get started with cgx.

[6]Note. If cgx is launched from a Terminal window and XQuartz was not already open, then first XQuartz will be launched automatically. The latter may take a second or two. If XQuartz was already open, however, the Calculix Graphix window will pop up immediately on your screen.

8.3 Getting Started With cgx

Consider the html-help files for ccx (cf. Section 9.3.5 for details) and references:

- User's guide of cgx [2], Sections 1 - 7 (cf. in particular Section 4 "Getting Started").

- Tutorial [3]: "Getting Started With CalculiX".

8.4 Test Examples

The functionality of the additional software will be tested in Section 9 below. In Appendix C in [2] some simple test examples are provided. An example for tet meshing of a given CAD geometry is contained in directory: "calculix_install_path"/calculix/examples/cad (for further explanations see the README in directory "cad" and also [2], Section B.11: "How to Deal With CAD Geometry").

9. TEST FUNCTIONALITY OF ADDITIONAL SOFTWARE

9.1 General

Subject of the present Section 9 is the test of the functionality of the optional additional software from Section 2.5. The considered test example will be prepared in the next Section 9.2. The subsequent test of the functionality in Section 9.3 follows the order of the menu items and commands (provided by cgx) as given in the first column of Tab. 2. In the aforementioned Tab. 2 also the following data are provided: functionality of cgx menu items and commands in column 2, names of additional software used from cgx menu items and commands in column 3.

9.2 Prepare Test Example and Hints

Except of the test for tet meshing in Section 9.3.6 below the result file: “result.frd“ will be considered as test example. The aforementioned result file: “result.frd“ is provided under the directory “examples” of the cgx installation:

“cgx_install_path”/CalculiX/cgx_2.8/examples/result.frd.

Steps 1 to 4 and hints (5) for preparation of test example:

(1) Launch a command window. Follow the instructions in Section 8.2 above. The Terminal- or xterm-window in the following is denoted as “Terminal”. Furthermore, the $-prompt of a Terminal window is considered.

(2) Enter the following commands in Terminal without comments:

```
$ cd calculix_inst_path/calculix/cgx_2.8/examples <return> # Change to working
                                                          # directory “examples”.

$ cgx_2.8 result.frd <return> # Launch cgx 2.8 and load result file “result.frd”.
```

(3) After the execution of the last command the Calculix Graphix window of cgx will pop up. The aforementioned main window is subdivided into the framed “drawing area“ and the so called “menu area“ which is located outside of the drawing area in the left margin of the window.

(4) Next, generate a fringe plot with post data from the loaded result file: “result.frd” (cf. also Section 4 in [2] for more details). Steps: Position mouse pointer in menu area of cgx window, press left mouse button to open Menu, keep button pressed and select menu item: “Datasets“ - “Disp“, then release button. Press left button again and select menu item: “Datasets” - “Entity” - “D1“, then release button. Now a fringe plot of the displacement components: “D1” is displayed in the drawing area. To display element edges press left button again in menu area and select menu items: “Viewing“ - Toggle Element Edges“, then release button. Now the element edges are displayed.

(5) Hints:

+ For entering of cgx commands in the command window the mouse pointer must be strictly located within the cgx main window, i.e. within the graphical window of cgx (cf. also [1], [2] for details).

+ Use the cgx-help menu to get help for cgx commands, respectively the cgx menu item: “Help” to display the help menu (cf. the test of the help menu in Section 9.3.5 for details).

+ See Section 6 in [2] for the mouse handling and also Section 10.2 below if you use a mouse without middle button.

+ Consider the messages (status, errors etc.) which are displayed in the command window (Terminal).

9.3 Tests

9.3.1 General – Test of cgx Menu Items (MI) and Commands (C)

The functionality of the cgx menu items (MI) and commands (C) from the first column of Tab. 2 depend on the optional additional software and will be tested below. Menu items and commands are tested in the sequence as given in the first column of Tab. 2.

9.3.2 Test MI “Hardcopy“ and C “hcpy“ Provided by ImageMagick

ImageMagick provides the “convert“ command-line tool which is used from cgx to generate hardcopies of the drawing area with MI “Hardcopy“ or C “hcpy“ (cf. also [2], Section 7.11 for “Hardcopy“ and Section 8.30 for “hcpy“). The format of plotfiles may be selected as explained below. Subsequently, the MI “Hardcopy“ will be tested.

Prearrangements: Test example from Section 9.2 (file: “result.frd”) prepared. Your current working directory, therefore, is: “cgx_install_path”/CalculiX/cgx_2.8/examples.

For a hardcopy of the fringe plot in ps (PostScript)-format position mouse pointer in menu area, press left mouse button and keep button pressed, then select menu items: “Hardcopy“ - “PsHardcopy“, then release mouse button. The command-line tool “convert“, respectively the installed additional software ImageMagick works if the file: “hcpy_1.ps” was created in the current working directory: “examples“. Check now, if “hcpy_1.ps” was created.

Next, repeat the afore-explained steps for a hardcopy to generate a second hardcopy: “hcpy_2.ps” of the same fringe plot. Directory “examples“ then contains the two ps-files: “hcpy_1.ps” and “hcpy_2.ps” which are required subsequently for the test of the cgx command C: “hcpy make ls“.

Notes. The command-line tool “convert“ may also be used as a standalone application for image manipulations. Use command in Terminal window: $ convert -h <return> to display convert options. Alterna-

tively, instead of the previously explained MI: “Hardcopy“ also the pre-installed (on Mac) Grab App may be used. Steps: (a) Double-click Grab icon in your “/Applications/Utilities” folder, (b) choose from Grab menu: “Capture“ - “Window“, (c) follow displayed Grab instructions to select window. Grab generates a tiff-file of the image which may be converted in other formats. For example, the Preview App may be used for conversion (use Preview menu items: “Save“ or “Export“ to convert the format of a plotfile).

9.3.3 Test C “hcpy make ls“ Provided by PSUtils, gs (GhostScript) and Preview

With the cgx command C: “hcpy make ls“ a number of n plotfiles: “hcpy_i.ps“; i = 1, 2, …, n which already exist in your working directory are bundled in a new ps-file: “cgx.ps“. Depending on the number n of existing files the bundled file: “cgx.ps” may contain one or more pages (cf. command: “hcpy“ in [2], Section 8.30 for more details). Furthermore, the bundled file: “cgx.ps“ is created in your current working directory and in addition is displayed with the default ps-viewer. “PSUtils” and „gs“ provide the CLTs (command-line tools) “pstops“ and “gs which are used from the command: “hcpy make ls“ to bundle files. Furthermore, the command: “hcpy make ls“ uses the CLT: “preview“ to display the file: “cgx.ps“, whereas the CLT: “preview“ makes use of the pre-installed (on Mac) Preview App (cf. Section 7.5 above for details).

Prearrangements for the test: (1) test example from Section 9.2 (result file: “result.frd”) prepared; your current working directory, therefore, is “examples” (path given in footnote 7 below), (2) files: “hcpy_1.ps” and “hcpy_2.ps” from the previous test in Section 9.3.3 are already located in working directory “examples”, (3) position mouse pointer in main window of cgx to activate the input of cgx commands.

Next, type in cgx command (with mouse pointer positioned in graphics window of cgx; consider echo of command which is displayed in your command window):

hcpy make ls <return>.

The tested additional software works if the created file: “cgx.ps“ is located in your working directory “examples“ and in addition automatically is displayed with Preview.

9.3.4 Test MI “Graph“ and C “graph“ Provided by gnuplot and Preview

In cgx the MI: “Graph“ and C: “graph“ are provided for the generation of path plots (xy-plots, also denoted as 2D plots for a path, history etc.). The capabilities of the aforementioned MI will be demonstrated below for the example of a path plot.

The additional software “gnuplot“ and “Preview“ provide the CLTs (command-line tools) “gnuplot“ and “preview“ which are used by MI “Graph“ and C “graph“ to generate and display plotfiles (cf. also [2], Section 7.9 for MI: “Graph“ and Section 8.26 in [2] for C: “graph“). Furthermore, according to Section 8.26 in [2], MI: “Graph“ and C: “graph“ generate the following files: “graph.out“ with xy-raw data, “graph.gnu“ with gnuplot plot commands, “graph.ps“ ps-plotfile (generated with gnuplot, using file “graph.gnu“). Fi-

[7] path: “cgx_install_path”/CalculiX/cgx_2.8/examples

nally, the generated file: “graph.ps“ will be automatically displayed with the default post-script viewer (“preview“ default for present Mac installation). The functionality of cgx for the CLTs “gnuplot“ and “preview“ will be tested below for the MI: “Graph“.

Prearrangements for test: Test example from Section 9.2 (file: “result.frd”) prepared. Your current working directory, therefore, is “examples” (path given in footnote 8 below).

Steps for test: move mouse pointer in main window of cgx, then press left mouse button, keep button pressed, then select menu items: “Graph“ - “Length“ for path plot, then release mouse button. Next, for the selection of nodes (path defined by selected nodes) follow the instructions in your command window. Select nodes, therefore, with left mouse button and quit selection with right button. Next, the aforementioned files are generated in the current working directory: “examples” and a Preview window with the path plot automatically pops up. In Preview you may convert the format of the generated plotfile in various other formats (use Preview menu items “Save“ or “Export“ to convert).

With the help of the raw data of the plot which are provided in file: “graph.out“ (in current working directory) you may also tailor graphs and also may use a plot program of your choice instead of gnuplot.

9.3.5 Test MI “Help“ (Display cgx- and ccx-Help Files) Provided by html-Browser

The cgx MI: “Help“ amongst others allows to display html-help files of cgx and ccx (cf. Section 4.4 and Section 4.5 above for the installation of the cgx- and ccx-help files) with a html-browser of your choice and makes use of the command-line tool: “browser_name“. For the present installation: “browser_name”=firefox was chosen (cf. Section 7.6 above for details).

Prearrangements for the test: Test example from Section 9.2 (file: “result.frd”) prepared; your current working directory, therefore, is: “examples” (path given in footnote 8 below).

Steps for test of html-browser: move mouse pointer in main window of cgx, press left mouse button, keep button pressed, then select menu item: “Help“ - “Html Manual (cgx)“, then release button. The browser window with the help-menu of cgx now will pop up. The ccx-help file may be displayed analogous as previously explained for the cgx-help file.

9.3.6 Test C “mesh“ – Tet Meshing Provided by Netgen Binary “ng_vol”

General

Tet meshing with cgx command: “mesh“ will be tested for the example of a cube. Consider the hints in Section 9.2 above, in particular for typing in cgx commands.

In the following subsections: “Step 1” to “Step 4” below, steps 1 – 4 are considered for the test of tet meshing for a cube:

[8] path: “cgx_install_path”/CalculiX/cgx_2.8/examples

- *Step 1 - Choose Option for Generation of Geometry of Cube.*
- *Step 2 - Use Prepared Geometry of Cube.*
- *Step 3 - Generate Geometry of Cube Manually.*
- *Step 4 - Mesh Cube With Tets for Test of Tet Meshing with Netgen Binary "ng_vol".*

Steps 1 – 3, therefore, are for the preparation of the geometry of the cube. The cube will be meshed in the final Step 4.

Step 1 - Choose Option for Generation of Geometry of Cube

The following two options A, B are provided for the preparation of the required geometry of the cube:

(A) Use prepared geometry of cube according to Step 2 below, then test tet meshing of cube according to the final Step 4 below.

(B) Generate geometry of cube according to Step 3 below, then test tet meshing of cube according to the final Step 4 below.

Step 2 - Use Prepared Geometry of Cube.

The present Step 2 has to be considered for option A according to Step 1 only. For option B follow Step 3 below instead.

In cgx geometry data are stored in fbd-files: <filename>.fbd. Accordingly, the prepared geometry will be loaded with the help of a fbd-file. The required code of the fbd-file is provided in Appendix 1.

Steps: (1) create working directory for the test; e.g. directory "mesh_cube" in your home directory; create directory with Terminal command: $ mkdir ~/mesh_cube <return> or with Finder, (2) open new file with TextEdit, (3) copy code from Appendix 1 (code for .fbd-file) in open file, then use TextEdit menu items: "Format" - "Make Plane Text" for plane text, (4) save file with filename "text" in new folder: "mesh_cube", then quit TextEdit. The saved File: "text.txt" will be subsequently renamed to: "test_tets.fbd" as explained below.

Next, open a terminal window and change directory to the working directory: "mesh_cube" with Terminal command: $ cd ~/mesh_cube <return>. Then rename file "text.txt" to file: "test_tets.fbd" with Terminal command: $ mv text.txt test_tets.fbd <return>.

Next, launch cgx with Terminal command:

$ cgx_2.8 -b test_tets.fbd <return>

After the execution of the last command the geometry of the cube first will be loaded and then displayed in the drawing area as follows: lines with labels ("L…"), surfaces with la-

bels (“A…“), body with label “B001“. The geometry, therefore, amongst others contains body “B001“ for the cube.

Continue with step 4 below to mesh the cube.

Step 3 - Generate Geometry of Cube Manually

The present Step 3 has to be considered for option B according to Step 1 only.

The geometry of the cube will be generated manually analogous as explained in tutorial [3] for a beam. Furthermore, in cgx geometry data (created e.g. with cgx) are stored in fbd-files: <filename>.fbd. Chosen filename for the present example: “test_tets.fbd”.

Prearrangements for test: (a) open a Terminal window, (b) create working directory for the test; e.g. directory “mesh_cube” in your home directory; create directory with Terminal command: $ mkdir ~/mesh_cube <return> or with Finder, (c) change directory to the working directory: “mesh_cube” with Terminal command: $ cd ~/mesh_cube <return>.

Next, launch cgx with Terminal command:

$ cgx_2.8 -b test_tets.fbd <return>

After the execution of the last command the cgx main window pops up and the following message is displayed in your Terminal window: “ERROR: The input file "test_tets.fbd" could not be opened“. Ignore the error message (after cgx commands “save“ or “exit“ the generated geometry will be stored in file test_tets.fbd).

Next, for the generation of the geometry of the cube follow the step by step instructions in tutorial [3] as explained below. Consider the note on the top of page 40 for the usage of command: “qadd“ in tutorial [3]. Position the mouse pointer within the drawing area of the cgx graphics window to enter cgx commands.

Steps to generate geometry of cube (consider note on usage of “qadd” on page 40 below): first of all, consider tutorial conventions at top of page 5 in [3], next start generation of geometry with line: “M: Move mouse into newly created window“ at bottom of page 5 in [3], then follow further instructions in [3] until the command: “K: Type "swep se3 se4 tra 0 0 10", E“ is reached on top of page 11 in [3], then replace the latter command with: "swep se3 se4 tra 0 0 1“ to create the cube with edge length 1. The latter swep-command also has created the geometrical entity: “body“ of the cube (cf. step 4 below for details). The geometry now is ready. Use cgx command (without quotes): “plot ba all <return>“ to display the body of the cube.

Next, save the geometry in file test_tets.fbd with cgx command: “save <return>“ or use cgx command “exit <return>“ instead (“exit“ first saves geometry, then quits cgx) if you want to quit cgx for the moment. To load the geometry after “exit“ again use Terminal command: $ cgx_2.8 -b test_tets.fbd <return>. Note that with the cgx menu item: “Quit“ nothing will be saved.

Continue with step 4 below to mesh the cube.

Note. The command sequence:

"K: Type "qadd se1", E

K: Type "a","

given at the bottom of page 7 in [3] is misleading. The following sequence works (in general for most interactive "q…" commands):

K: Type "qadd se1", E

Next, set rectangle for selection of items according to the instructions in [3] or [2], Section 8.58. After the rectangle has been set, you may type in options for command qadd. For example, in the following command:

K: Type "a"

option "a" is set for mode "all". Accordingly, after this option has been set, "mode:a" will be displayed in your Terminal window (cf. also details for command "qadd" in [2], Section 8.58). After "mode:a" is displayed in your command window continue with the instructions in [3].

Step 4 - Mesh Cube With Tets for Test of Tet Meshing with Netgen Binary "ng_vol"

- ***General***. The present subsection: "Step 4" is organized in further subsections: Step 4.1 to Step 4. 3 as follows:
  - *Step 4.1 – Use cgx Command: mesh "set_name_b" for tet meshing*
  - *Step 4.2 – Use cgx Command: mesh "set_name_trias" size for tet meshing*
  - *Step 4.3 – Use cgx Command: qdiv for control of element size via subdivision of lines*

  If you are for the moment interested only in a quick test of the basic functionality of tet meshing with Netgen, then continue now with "Step 4.1" below.

  For the aforementioned Step 4.1 direct meshing of a volume with tets is considered, whereas for Step 4.2 first the surface of the volume is meshed with trias and subsequently the volume with tets for a given element size, based on the mesh seed which is provided by the trias. Step 4.3 is for the control of the element size via the subdivision of lines which form the element edges.

  For the present example the cube was created with cgx command: "swep". The latter command also automatically creates the geometrical entity: "body", to which the geometrical properties of the volume of the cube are assigned to (if a body is not created automatically as mentioned before, you may create it with cgx command: "body" according to [2], Section 8.4). The afore mentioned entity: "body" is contained in set "all", whereas set "all" is a default set in cgx which contains all created entities (points, lines, bodies, nodes, elements etc.) and, therefore, amongst others all geometrical building blocks of the cube. Use cgx command:

  prnt se <return>

  to display sets. Amongst others, the last command displays set "all" as follows:

  "1 all stat:o n:0 e:0 f:0 p:26 l:12 s:6 b:1 L:0 S:0 se:0 sh:6".

According to the above output, amongst others set “all“ contains the geometrical entities: l, s, b of the cube, i.e. twelve lines l:12 for the edges, six surfaces s:6 for the faces, one body b:1 for the volume. The afore mentioned entities will be used below for the meshing of the cube, respectively are addressed for meshing with the help of set: “setname“=all.

In cgx the following two “mesh“ commands a, b are provided for tet meshing:

(a) mesh “set_name_b”. For direct tet meshing of bodies (without prior meshing of surfaces of bodies), after element type (te4 or te10) has been set before. Bodies must be contained in set “set_name_b“. The aforementioned command is applied under Step 4.1 below for the test of tet meshing. The element size may be controlled via the subdivision of lines as explained below (cf. also example under Step 4.3 below).

Note. According to [2], Section 8.41 also the following meshing capability is provided by cgx, but is not tested here: “Therefore, its possible to start cgx in the viewing mode (-v) with a mesh alone, and then create bodies and fill them with additional elements.“

(b) mesh “set_name_trias” size. Imagine that the volume of a body is uniquely defined by a regular surface mesh (example given in Step 4.2 below) with trias (triangular shell elements of type tr3, tr3u or tr6u). The cgx command: “mesh “set_name_trias” size“ then may be used for subsequent volume meshing of the body with tets; “set_name_trias“: set must contain surface elements, size: optional parameter for element size of tets. Accordingly, prior to volume meshing, first an adequate surface mesh must be generated. Depending on the type of chosen trias, linear or quadratic tets te4 or te10 are automatically chosen for tet meshing. Surface and volume meshing is demonstrated under Step 4.2 below for the test of tet meshing with the aforediscussed command. The element size of the trias may be controlled via subdivision of lines (cf. example under Step 4.3 below) and the element size of tets via the already aforementioned optional parameter: size (cf. example under Step 4.2 below for the usage of parameter: size).

- ***Step 4.1 – Use cgx Command: mesh “set_name_b” for tet meshing***. Task: test tet meshing for previously build geometrical model of cube. As already previously explained, amongst others set: “all“ contains the geometrical entity “body“ of the cube which is required for the subsequent direct volume, respectively tet meshing of the cube.

  Summary of meshing procedure (sequential). Body to be meshed already contained in set: “set_name_b”= all. Set element type (te4 or te10), then use cgx command “mesh “set_name_b”“ for tet meshing.

  Control of element size. Subsequently controlled by the default number of four subdivisions of lines (cf. example under point 4.3 below, how the number of subdivisions may be changed). For the default number of four subdivisions, four linear or two quadratic elements are created along some line, respectively some edge of the cube.

  Steps for tet meshing. If necessary, first launch cgx and load test example cube with Terminal command: $ cgx_2.8 -b test_tets.fbd <return>, then type in the following cgx

commands (without comments, for typing in cgx commands position mouse pointer in graphics window of cgx):

elty all <return> # Reset setting of element types.

elty all te10 <return> # Assign tet elements te10 to body contained in set “all“.

mesh all <return> # Mesh cube, respectively body contained in set “all“ with
pre-specified tets.

The last cgx-command: “mesh all“ for tet meshing, respectively tet meshing with the Netgen binary: “ng_vol” works if the meshing is finished with the following printout in your Terminal window:

“.....
36 Tet-elems in body:B001
complete set
set complete
elements deleted
ready“

Depending on your needs, you may quit now the test of tet meshing or continue with further tests.

According to the printout above, 36 tets have been generated. The generated nodes and elements are contained in set: “all“ (type cgx command: “print se <return>“ to check the contents of set “all”).

Display elements. Use cgx command (without green comment):

plot e all <return> # Plot elements (e) contained in set “all“.

and then menu items: “Viewing“ - “Fill“, then “Viewing“ - “Toggle Element Edges“ to display element edges. Instead of the afore explained sequence of commands, starting with cgx command “plot e all <return>“, you may use the cgx menu items: “Viewing“ - “Show All Elements With Light“ to display elements and then: “Viewing“ - “Toggle Element Edges“ to display element edges.

Write mesh to file. Use cgx command (without comment):

send all abq <return> # Write (optional) file all.msh with all nodes and all elements
contained in set “all“.

to create file: “all.msh“ with nodal coordinates and connectivity of elements.

Delete mesh. Use cgx command (without comment):

del mesh <return> # Deletes all mesh data.

After the mesh has been deleted the mesh is no longer displayed and in set: “all“ all entries for nodes and elements are empty (use cgx command: “print se <return>“ to check).

If you want to re-plot the still existing geometrical entities use cgx commands: plot “entity“ all <return> to plot geometrical entities, with “entity“=l for lines, “entity“=s for surfaces, “entity“=b for body (cf. also [2] for the cgx plot-command: “plot“). After the first use of command: plot “entity“ all <return> use command: plus “entity“ all <return> to add plots of additional entities (cf. also [2] for the cgx plot-commands: “plus“).

- ***Step 4.2 – Use cgx Command: mesh “set_name_trias” size for tet meshing***. Task: test tet meshing for previously build geometrical model of cube. As previously explained, amongst others set: “set_name_trias=all already contains the geometrical entities (lines, surfaces, body) of the cube which are required for the subsequent meshing.

  Regular surface mesh. For tet meshing a regular surface mesh is required. Conditions i, ii of a regular surface mesh: (i) mesh must form closed surface, (ii) orientation of unit normal vectors of surface elements such that they point from volume outwards. The latter may be controlled with the help of the front and back side of surface elements or alternatively with the help of the front and back side of geometrical surfaces (cf. further explanation and example below).

  Condition i of a closed surface mesh in general is satisfied if a surface is meshed automatically with cgx. Each trias then is joint by three neighboring trias. If surface meshes are build manually, however, condition i should be checked.

  Further explanation of condition ii of regular orientation of geometrical surfaces and surface elements. For the sake of simplicity the further explanation is restricted on geometrical surfaces, because all of the definitions and manipulations given below for geometrical surfaces also hold for surface elements and vice versa. Accordingly, the (geometrical) free surfaces of a volume are regular in the sense of condition ii, if the unit normal vectors of the surfaces point from the volume outwards. Terms and explanations:

  + *Unit normal vector of surface*. Defined uniquely by vector product and calculated accordingly in cgx.

  + *Front and back side of surfaces*. Some surface and associated normal vector may be given. Furthermore, imagine that a light source is mounted on the tip of the vector. The front side then per definition is this side of the surface which reflects light, the other one is the back side. In cgx the reflecting front side of a geometrical surface is filled in magenta (default), the non-reflecting back side in black (default). For a surface element the front side is filled in light green (default), the back side in black (default). Instead of the unit normal vector, therefore, the differently colored front and back side may be used to check, if the orientation of some geometrical surface or surface element is regular.

  + *Control of regular orientation and flip orientation*. If for some volume the reflecting front sides of free surfaces are displayed, the orientation of these surfaces is regular (cf. example below). However, if non-reflecting back sides are displayed the orientation must be flipped to become regular (cf. example below). The orientation of surface elements may be controlled and flipped analogous as previously explained for geometrical surfaces.

Furthermore, the following holds for automatic surface meshing with cgx: the orientation of surface elements is governed by the orientation of geometrical surfaces. Consequently, the orientation of surface elements may be controlled with the orientation of geometrical surfaces. The latter will be applied and checked below for the surface meshing of the cube. If surfaces are meshed manually the regular orientation of elements always should be checked and if necessary corrected analogous as demonstrated for the geometrical surfaces of the cube below.

Summary of tet-meshing procedure (sequential). First of all, check if geometrical surfaces are regular. If not, flip non-regular surfaces prior to surface meshing. Next, set element type for surface meshing with trias (tr3, tr3u or tr6u), then generate regular surface mesh. The generated mesh entities (nodes, surface elements) are contained in set: "set_name_trias=all. Finally, use cgx command: "mesh "set_name_trias" size" for tet meshing with parameter: size as explained below for the control of the mesh size. Depending on the type of chosen trias, linear or quadratic tets te4 or te10 are automatically chosen for tet meshing. After tet meshing, nodes and elements for the generated tets are contained in set: "all". The surface mesh is deleted after tet meshing.

Control of element size. The size of trias subsequently is controlled by the default number of four subdivisions of lines which form element edges (cf. example under Step 4.3 below, how the number of subdivisions may be changed). For the default number of four subdivisions four linear or two quadratic elements are created along some line, respectively some edge of the cube. The size of tets will be controlled with the parameter: "size" which is provided by the considered tet meshing command.

Steps for tet meshing. If necessary, first launch cgx and load test example cube with Terminal command: $ cgx_2.8 -b test_tets.fbd <return>. An already existing mesh may be deleted with the cgx command: del mesh <return>.

As already mentioned before, the required regular surface mesh will be created by meshing of regular geometrical surfaces of the cube. The latter requires that first of all the free surfaces of the cube are checked for regular orientation. To check the orientation, plot the front and back sides of the surfaces with the following cgx commands (without green comments; for typing locate mouse pointer in graphics window of cgx):

```
rep all <return> # Renders surfaces for plotting of front and back sides (cf.[2],
                 # Section 8.84 for details).

plot si all <return> # Plot rendered surfaces from set "all".
```

After the execution of the last command three regular front sides are filled in magenta (reflecting) and three non-regular back sides in black (non-reflecting). The aforementioned back sides, therefore, first must be flipped to become regular as required for the subsequent surface meshing.

Several procedures are provided for the flipping of geometrical surfaces and trias with the cgx command: "qflp" (cf. [2], Section 8.69 for details and also the cgx-help menu). Because for the present example some of the surfaces are already regular, the following

cgx commands (without comments) may be applied for flipping of the non-regular surfaces (also consider instruction):

```
qflp <return> # Invoke command "qflp" to flip surfaces.

Instruction. Next, select one of the regular surfaces (plotted in magenta) by positioning
             the mouse pointer on this surface.

a # Type "a" to indicate that all surfaces should be orientated like the selected one.

s # Type "s" to orientate all surfaces like the selected one. The initially black surfaces
  # are immediately flipped and filled in magenta.

q # Type "q" to quit command qflp.
```

The geometry of the cube now is ready for surface meshing with trias. Use cgx commands (without comments):

```
elty all <return> # Reset setting of element types.

elty all tr6u <return> # Assign element type tr6u to all surfaces which are contained
                       # in

mesh all <return> # Mesh all surfaces from set "all" with trias.
```

The last command creates 36 trias (use cgx command "print se" to check set "all" for elements). Display now elements with cgx command:

```
plot e all <return> # Display elements (without edges).
```

Because the meshed geometrical surface was regular all elements are also regular and, therefore, are filled in green (green fill indicates reflecting front sides of trias). Element edges may be displayed with cgx menu items: "Viewing" - "Toggle element Edges". Note that if for another example some of the created elements are non-regular (plotted in black), you may flip these elements as previously explained for geometrical surfaces. Alternatively, you may flip non-regular geometrical surfaces before surface meshing.

The trias of the surface mesh are contained in set: "all" (check with cgx command "prnt all"). Accordingly, set: "set_name_trias"=all subsequently is used in the command for tet meshing.

The volume now may be meshed with tets. For tet meshing with (optional) pre-specified element size (e.g. size=0.1) use cgx command (without comment):

```
mesh all tet 0.1 <return> # Mesh volume defined by regular surface mesh from
                          # set „all" with tets and (optional) element size 0.1.
```

The last command creates 23 tet elements and deletes the surface elements (use cgx command: prnt se <return> to check set "all" for the created elements). To keep the surface elements, you may store them before tet meshing in a ccx solver file (cf. tutorial [3] for an example and also cgx command "send all abq" below). To use the surface mesh

for tet meshing again, read the solver file with option "-v" in the command for launching of cgx.

Plot elements with cgx menu items: "Viewing" - "Show All Elements With Light" and: "Viewing" - "Toggle Element Edges" to view element edges. Further menu items for viewing are explained in [2], Section 7.2. If you use the cgx menu item: "Toggle Move-Z/Zoom" for viewing with a cutting plane, then consider Section 10.2 below for the positioning of the cutting plane if you use a mouse without middle bottom.

Write mesh to file. Use cgx command (without comment):

```
send all abq <return>  # Write (optional) file all.msh with all nodes and all
                       # elements which are contained in set "all".
```

to write nodal coordinates and connectivity of elements in ABAQUS format on file all.msh (cf. also tutorial [3] for further use of file all.msh).

Delete mesh. Use cgx command (without green comment):

```
del mesh <return>  # Deletes all mesh data.
```

After the mesh has been deleted, in set "all" all entries for nodes and elements are empty (use cgx command: "print se" to display the contents of set "all").

- ***Step 4.3 – Use cgx Command: qdiv for control of element size via subdivision of lines***. The size of trias and tets may be controlled by the number of subdivisions of lines which form the element edges. The default number of subdivisions is four and may be changed as explained below.

  Steps for changing number of subdivisions of lines. If necessary, first launch cgx and load test example cube with Terminal command: $ cgx_2.8 -b test_tets.fbd <return>. Next, use cgx command (without comment) to display lines with current setting of subdivisions:

  ```
  plot ld all <return>  # Displays current setting of subdivisions of lines which are
                        # contained in set „all".
  ```

  If you want to change the number of subdivisions use cgx command: „qdiv" (cf. also [2], Section 8.66 for details and also tutorial [3] for examples). In the following example the number of subdivisions will be set for lines which are displayed with the previously discussed command: plot ld all <return>. Use cgx commands (without comments, consider instructions):

  ```
  qdiv <return>  # Invoke command "qdiv".

  Instruction: Move mouse pointer on one of the lines to change the number of subdivi-
               sions of this line.

  8 # Type for example number "8" for number of subdivisions. Typed in number will
    # be immediately displayed for selected line. Consider also the printout in the
    # command window.
  ```

Instruction. Input of numbers greater than nine requires leading blank; for example: „ 16“.

Instruction. Next, the number of subdivisions of all of the lines will be changed, which are selected with the help of a user defined rectangle.

a # Type “a“ to select all lines within a rectangle in mode: “mode:a“.

Instruction: Set rectangle to select all lines (move mouse pointer to upper left corner of rectangle, then typ “r“, next move mouse pointer to lower right corner, then type “r“). Next move rectangle such that all wanted lines are enclosed by rectangle.

8 # Type for example number: “8“ for eight subdivisions of all selected lines.
#Afterwards, the new number of subdivisions is displayed immediately.

q # Type “q“ to quit command qdiv.

10. HINTS

10.1 General

In the following subsections Mac-specific hints are provided (cf. also the cgx installation instructions [1] and the cgx user's guide [2] for hints and tips).

10.2 Mouse Without Middle Button and Trackpad

In the instructions [1], [2] a mouse with left, middle and right mouse button is considered. If you use, for example a wireless Apple Magic Mouse without middle button, then do the following: Use left and right clicks, if in the aforementioned instructions pushing of left or right button is required, use scroll gesture (move fingertips forth and back along length side of mouse), if pushing of the middle button is required. The aforementioned scroll gesture in particular may be used to zoom, but also for several other manipulations (i.e. for cgx menu item: "Toggle Move-Z/Zoom" according to [2], Section 7.2.10). A trackpad may be used analogous as afore-explained for a mouse without middle button.

10.3 Mouse Pointer

For entering of cgx commands, the mouse pointer always must be positioned strictly within the graphical cgx main window. The mouse pointer is also used to select entities (nodes, edges etc.), for example if the cgx command "enq" is used (cf. [2], Section 8.17 for details). Procedure for usage of cgx command "enq": position mouse pointer in main window of cgx, then type cgx command: enq <return>, then position mouse near to entity you want to enquire, then enter option for entity (line, surface etc.).

10.4 Keyboard

With the following exceptions, the keystrokes given in [2], Section 6.2 also hold for Mac keyboards; use "keystrokes":

- "fn-Up Arrow" for "page up" in your command window,

- "fn-Down Arrow" for "page down" in your command window.

11. KNOWN MAC-SPECIFIC PROBLEMS

11.1 Installation of Additional Software

Problems, fix and future actions for installation of:

- *gv (post script viewer)*. Installation failed (cf. Section 7.5 for details). Fix: Preview App used instead. No future actions for installation of gv planned.

REFERENCES

[1] INSTALL. K. Wittig, December 14, 2013. Basic installation of cgx for Unix/Linux. File: “INSTALL” downloaded with cgx-package (cf. path to file: “INSTALL” in Fig. 1: “cgx_install_path”/CalculiX/cgx_2.8/INSTALL).

[2] CalculiX USER’S MANUAL - CalculiX GraphiX, Version 2.8. K. Wittig, January 17, 2015. Link for download of PDF: http://www.dhondt.de/cgx_2.8.pdf

[3] Getting Started With CalculiX. Tutorial. J. Baylor. Link for download of PDF: http://www.bconverged.com/content/calculix/doc/GettingStarted.pdf

[4] Installation Guide – CalculiX Solver ccx 2.8 on Mac OS X Mavericks. B. Graf, February 10, 2015.

INSTALLATION GUIDE

CalculiX Graphical Pre- and Postprocessor cgx 2.8
on Mac OS X Mavericks

Appendix 1: Code for File test_tets.fbd

[9]B. Graf

[9]Prepared by:
Dr. B. Graf
Consulting Engineer
8044 Zürich
graf_bernhard@bluewin.ch

```
PNT p1       0.00000      0.00000      0.00000
PNT p2       1.00000      0.00000      0.00000
PNT D001     0.00000      1.00000      0.00000
PNT D002     1.00000      1.00000      0.00000
PNT D003     0.00000      0.00000      1.00000
PNT D004     1.00000      0.00000      1.00000
PNT D005     0.00000      1.00000      1.00000
PNT D006     1.00000      1.00000      1.00000
PNT D007     1.00000      1.00000      0.00000
PNT D008     0.00000      1.00000      0.00000
PNT D009     0.00000      0.00000      0.00000
PNT D00Q     1.00000      1.00000      1.00000
PNT D00R     0.00000      1.00000      1.00000
PNT D00S     0.00000      0.00000      1.00000
PNT D019     1.00000      0.00000      1.00000
PNT D01A     0.00000      0.00000      1.00000
PNT D01B     0.00000      0.00000      0.00000
PNT D01S     1.00000      1.00000      1.00000
PNT D01T     0.00000      1.00000      1.00000
PNT D01U     0.00000      1.00000      0.00000
PNT D02B     0.00000      1.00000      1.00000
PNT D02C     0.00000      0.00000      1.00000
PNT D02D     0.00000      0.00000      0.00000
PNT D02U     1.00000      1.00000      1.00000
PNT D02V     1.00000      0.00000      1.00000
PNT D02W     1.00000      0.00000      0.00000
LINE L001 p1 p2 104
LINE L002 D001 D002 104
LINE L003 p1 D001 104
LINE L004 p2 D002 104
LINE L005 D003 D004 104
LINE L006 D005 D006 104
LINE L007 D003 D005 104
LINE L008 D004 D006 104
LINE L009 p1 D003 104
LINE L00A p2 D004 104
LINE L00B D001 D005 104
LINE L00C D002 D006 104
SHPE H001 PLN D007 D008 D009
SHPE H002 PLN D00Q D00R D00S
SHPE H003 PLN D019 D01A D01B
SHPE H004 PLN D01S D01T D01U
SHPE H005 PLN D02B D02C D02D
SHPE H006 PLN D02U D02V D02W
GSUR A001 + H001 - L001 + L003 + L002 - L004
GSUR A002 + H002 - L005 + L007 + L006 - L008
GSUR A003 + H003 - L001 + L009 + L005 - L00A
```

```
GSUR A004 + H004 - L002 + L00B + L006 - L00C
GSUR A005 + H005 - L003 + L009 + L007 - L00B
GSUR A006 + H006 - L004 + L00A + L008 - L00C
GBOD B001 NORM + A001 - A002 + A004 - A006 - A003 + A005
SETA se1 l L001
SETA se2 p D001
SETA se2 p D002
SETA se2 l L002
SETA se3 s A001
SETA se4 p D003
SETA se4 p D004
SETA se4 p D005
SETA se4 p D006
SETA se4 l L005
SETA se4 l L006
SETA se4 l L007
SETA se4 l L008
SETA se4 s A002
plot la all k
plus sa se2 g
plus sa all m
plus ba all g
```

INSTALLATION GUIDE

CalculiX Graphical Pre- and Postprocessor cgx 2.8
on Mac OS X Mavericks

Appendix 2: Installation of Netgen With GUI

[10]B. Graf

[10]Prepared by B. Graf, based on the installation instructions from M. Woopen (AICES, RWTH Aachen, Germany) for Mac OS X Snow Leopard, 2011.

Dr. B. Graf
Consulting Engineer
8044 Zürich
graf_bernhard@bluewin.ch

GENERAL

The installation of Netgen with GUI is considered. The installation has been tested for the program versions as given below. After the installation of Netgen with GUI, continue installation of Netgen for cgx with step 6 in Section 7.7.

SOFTWARE REQUIREMENTS

- *Mac-specific*. OS X 10.9.5, Xcode 5.1.1, XQuartz 2.7.5, modified GCC 5.1.1 according to Section 2.2.
- *TCL*. Besides XQuartz, the GUI of Netgen requires the installation of the following TCL-packages:
  - tcl8.5.9
  - tk8.5.9
  - Tix8.4.3
  - Togl-1.7

  It should be noted that the GUI may not work for deviating combinations of versions of tcl, tk, Tix and Togl. In other words, there is not only a dependency of tcl on tk, Tix and Togl, but also on the versions of the aforementioned packages.
- *Netgen 5.1*.

INSTALL MAC-SPECIFIC SOFTWARE

Follow instructions in Section 5.

INSTALL TCL

- *Installation path*. For the present instructions it is assumed that TCL is installed for the user defined path:

  "tcl_install_path"=/Users/LOGNAME/TCL_INSTALL.

  Accordingly, TCL will be installed in directory "TCL_INSTALL" under your home directory: /Users/LOGNAME. Create directory "TCL_INSTALL" with the Finder or Terminal commands: $ cd /Users/LOGNAME <return>, $ mkdir TCL_INSTALL <return>. You may set the installation path depending on your needs.

  For the subsequent installation of TCL the full path to the installation directory must be specified as given above. Replace, therefore, "tcl_install_path" in the instructions with: /Users/LOGNAME/TCL_INSTALL

  The libraries which are required for the GUI of Netgen are installed for the path: "tcl_install_path"/lib.

- *Install Tcl8.5.9 from source*. Download source from site:

  http://sourceforge.jp/projects/sfnet_tcl/downloads/Tcl/8.5.9/tcl8.5.9-src.tar.gz/

  The package: "tcl8.5.9-src.tar.gz" will be downloaded on your Desktop. Next, drag package from Desktop to your installation directory: "tcl_install_path", then unzip package (double click on package). The file system now looks as follows: "tcl_install_path"/tcl8.5.9, i.e. the source code now is contained in folder: "tcl8.5.9".

  Complete installation with Terminal commands (don't type in comments):

  $ cd "tcl_install_path"/Tcl8.5.9/unix <return> # Goto directory "../tcl8.5.9/unix".

  $ ./configure --prefix="tcl_install_path" <return> # Configure, i.e. create Makefile.

  followed by command sequence: $ make <return>, $ make install <return> to build and install Tcl.

  Before re-compilation use command: $ make distclean <return>, then re-configure. For more details consider README provided for Tcl.

- *Install tk8.5.9 from source*. Download source from site:

  http://de.sourceforge.jp/projects/sfnet_tcl/downloads/Tcl/8.5.9/tk8.5.9-src.tar.gz/

  The package: "tk8.5.9-src.tar.gz" will be downloaded on your Desktop. Next, drag package from Desktop to your installation directory: "tcl_install_path", then unzip package (double click on package). The file system now looks as follows: "tcl_install_path"/tk8.5.9, i.e. the source code now is contained in folder: "tk8.5.9".

  Complete installation with Terminal commands (don't type in comments):

  $ cd "tcl_install_path"/tk8.5.9/unix <return> # Goto directory "../tk8.5.9/unix".

  # Configure, i.e. create Makefile:

  $ ./configure --prefix="tcl_install_path" --with-tcl="tcl_install_path"/lib <return>

  followed by command sequence: $ make <return>, $ make install <return> to build and install Tk.

  Before re-compilation use command: $ make distclean <return>, then re-configure. For more details consider README provided for tk.

- *Install Tix8.4.3 from source*. Download source from site:

  http://de.sourceforge.jp/projects/sfnet_tix/downloads/tix/8.4.3/Tix8.4.3-src.tar.gz/

  The package: "Tix8.4.3-src.tar.gz" will be downloaded on your Desktop. Next, drag package from Desktop to your installation directory: "tcl_install_path", then unzip pack-

age (double click on package). The file system now looks as follows:
“tcl_install_path“/Tix8.4.3, i.e. the source code now is contained in folder: “Tix8.4.3“.

Complete installation with Terminal commands (don’t type in comments):

$ cd “tcl_install_path”/Tix8.4.3 <return> # Goto directory “Tix8.4.3”.

Configure, i.e. create Makefile:

$ ./configure --prefix=”tcl_install_path” --with-tcl=”tcl_install_path”/lib
--with-tk=”tcl_install_path”/lib LDFLAGS="-L/usr/X11/lib -lX11" <return>,

followed by command sequence: $ make <return>, $ make install <return> to build and install Tix.

Before re-compilation use command: $ make distclean <return>, then re-configure. For more details consider README provided for Tix.

- *Install Togl-1.7 from source*. Download source from site:

http://sourceforge.jp/projects/sfnet_togl/downloads/Togl/1.7/Togl-1.7.tar.gz/

The package: “Togl-1.7.tar.gz“ will be downloaded on your Desktop. Next, drag package from Desktop to your installation directory: “tcl_install_path“, then unzip package (double click on package). The file system now looks as follows:
“tcl_install_path“/Togl-1.7, i.e. the source code now is contained in folder “Togl-1.7“.

Next, open file “configure“ (path: “tcl_install_path“/Togl-1.7/configure) and

replace

XLIBSW=-lX11 in line 7020 of file configure

with:

XLIBSW="-L/usr/X11/lib -lX11"

Complete installation with Terminal commands (don’t type in comments):

$ cd “tcl_install_path”/Togl-1.7 <return> # Goto directory “Togl-1.7”.

Configure, i.e. create Makefile:

$ ./configure --prefix=”tcl_install_path” --with-tcl=”tcl_install_path”/lib
--with-tk=”tcl_install_path”/lib cppflags="-I/usr/X11/include" <return>

$ make <return> # Build Togl.

$ make install <return> # Install Togl.

Next, **create soft link** in directory: „tcl_install_path“/lib/Togl1.7:

$ “tcl_install_path”/lib/Togl1.7 <return> # Goto directory Togl1.7.

$ ln -s libTogl1.7.dylib libTogl.dylib <return> # Create link

Note on link. For build of Netgen the library libTogl.dylib is required, the library libTogl1.7.dylib for execution of Netgen.

Before re-compilation use command: $ make distclean <return>, then re-configure. For more details consider README provided for Togl.

INSTALL NETGEN WITH GUI

- *Installation path and installation of Netgen files*. Follow steps 1 to 4 of the installation instructions in Section 7.7. Installation path, therefore (cf. step 1 in Section 7.7):

  “netgen_install_path”.

- *Install Netgen with GUI according to step 5.2 in Section* 7.7. Use Terminal commands (don’t type in comments):

```
$ cd “netgen_install_path”/Netgen-5.1 <return> # Goto directory Netgen-5.1.

# Configure, i.e. create Makefile:

$ ./configure --prefix=“netgen_install_path” --with-tcl=“netgen_install_path”/lib
              --with-tk=“netgen_install_path”/lib --with-togl=“netgen_install_path”/lib/Togl1.7
              CPPFLAGS="-I/usr/X11/include" LDFLAGS="-L/usr/X11/lib -lGLU" <return>

$ make <return> # Build Netgen.

$ sudo make install <return> # Install Netgen.
```

  Before re-compilation use command: $ make distclean <return>, then re-configure. For more details consider README provided for Netgen.

  The **Netgen-binaries** (executable "netgen", “ng_vol” etc.) are installed under path: “netgen_install_path”/bin.

- *Set environment variables*. Set variables in your .bash_profile as follows:

```
$ cd ~ <return>  # Goto directory of .bash_profile (located in your home directory ~).

$ open -e .bash_profile <return>  # Opens .bash_profile in TextEdit.
```

  Next, insert the following three lines starting with “export” at the end of your .bash_profile without comments:

```
export PATH=$PATH:”netgen_install_path”/bin   # Sets path to executable "netgen" of netgen,
                                              # so you may invoke netgen from your
                                              # working directories.
```

```
export NETGENDIR=:"netgen_install_path"/bin"   # Set directory for Netgen.

export DYLD_LIBRARY_PATH=:"netgen_install_path"/lib/Togl1.7:"netgen_install_path"/lib:
                         "netgen_install_path"/lib:"   # Set directory for libraries.
```

Next, save and close file .bash_profile and then use the following Terminal commands to activate the new path settings:

```
$ source .bash_profile <return>  # Activates previously set variables in your
                                 # .bash_profile.

$ echo $PATH <return>  # displays path settings. Check, if path:
                       # "netgen_install_path"/bin to executable of Netgen is set.
```

LAUNCH NETGEN WITH GUI

Use Terminal commands (don't type in commands):

```
$ cd workingdirectory <return>  # Goto your working directory (create one, if it not
                                # already exists).

$ netgen <return>   # Launch Netgen.
```

After the last command the GUI of Netgen will open and you are able now to create some mesh.

COMPLETE INSTALLATION OF NETGEN FOR CGX

Continue installation of Netgen for cgx with step 6 in Section 7.7.