CalculiX USER'S MANUAL
- CalculiX GraphiX, Version 1.7 -

Klaus Wittig

August 9, 2007

Contents

1 Introduction

This is a description of CalculiX GraphiX (cgx). This program is designed to generate and display finite elements (FEM) and results comming from CalculiX CrunchiX (ccx). If you have any problems using the program, this document should solve them. If not, you might send a mail to the author [3]. The next section gives some background on the functionality and mesher capabilities. The following section describes how to run the verification examples you should have obtained along with the code of the program. You might use this section to check whether you installed CalculiX correctly. Then, a detailed overview is given of the menu and all the available keywords in alphabetical order. Finally, the User's Manual ends with the appendix and some references used while writing the code.

2 Concept

This program uses the openGL library for visualization and the glut library [2] for window management and event handling. This results in very high speed if a hardware-accellerated openGL-library is available and still high speed for software-rendering (MesaGL,[1]).

The cgx has pre- and post-processor capabilities. It is able to generate and display beam, shell and brick elements in its linear and quadratic form. In addition it can display but not create penta- and tetraeder-elements.

But it is possible to generate a surface-mesh of triangles and export it in stl-format. This format can be read by an external mesher as for example

NETGEN [4]. This mesher fills the volume with tetraeder elements and is able to export abaqus format. This can be read by cgx and ccx.

The inbuild mesher creates a structured mesh based on a description of the geometry. For example it uses lines for beam elements, surfaces for shell elements and volumes (bodies) for brick elements. The program distinguishes between the mesh and the underlying geometry. Elements are made from faces and faces are made from nodes. If you move a node the faces and elements will follow. The geometry behaves accordingly to the mesh: Lines are made from points, surfaces are made from lines and bodies are made of surfaces. Surfaces might have 3 to 5 edges and bodies might have 5 to 7 surfaces. As a result of the concept, if you modify the position of a point, all related geomety will follow. In other words, if the location of geometric entities should be changed, it is necessary to move the points on which the entities rely.

It should be noted that faces exist only on free surfaces of the model.

In addition entities called "set" are available to group together whatever entities. Sets are useful to handle parts of the model. For example to manipulate or display this parts. After the mesh is created it must be written to a file to be available for the solver. Also several boundary conditions and loads can be written to files. This files have to be added into the ccx control file. Additional commands as the type of analysis, material description and so on must be added with the help of an editor. After the analysis is completed the results can be visualized by calling the cgx program again in an independent session. The post-processor functionality is mainly controlled by a pop-up menu were the pre-processor functionality is controlled by the keyboard. Shaded animations of static and dynamic results, the common color plots and time history plots can be created. Also a userfunction can be included. For example if someone needs his own functions to manipulate the result-data of a calculation or if he needs an interface to read or write his own format of result-data.

3 File Formats

The following file-formats to store or read geometric entities are available:

- fbd-format, this format consists of a collection of commands explained in the section "Commands" and it is mainly used to store geometrical information like points, lines, surfaces and bodies. But it can also be used to define a batch job.
- stl-format, this format describes a shape using only triangles (see the read command to handle edges generated by NETGEN).

Common CAD formats will be supported by stand-alone interfaces which translate into fbd-commands.

The following file-formats can be written to define the mesh and certain boundary-conditions:

- Abaqus, which is also used by the CalculiX solver ccx,
- Ansys,
- Code Aster,
- Samcef,
- duns, a free cfd-code [5],
- isaac, a free cfd-code [6],
- OpenFOAM, a free cfd-code [7] (So far, only 8-noded brick-elements are supported),
- Nastran,
- tochnog, a free fem-code [8] (So far, only 8-noded brick-elements are supported).

The following solver-input-file-format can be read to check the mesh and certain boundary-conditions:

- Abaqus, this is also used by the CalculiX solver ccx.

The following file-formats to read solver results are available:

- frd-format, files of that format are used to read results of previous calculations like displacements and stresses. This format is described in section "Result Format" and is also used by ccx.
- duns, a free cfd-code [5],
- isaac, a free cfd-code [6],
- OpenFOAM, a free cfd-code [7].

For a closer description how to read this formats see "Program Parameters" and how to write them see the "send" command.

4 Getting Started

For installation look in the INSTALL file. After the program is installed on your machine you should check the functionality. For this purpose some examples are included in the distribution. Before we start you should read the section "Input Devices". Then we begin with a result file called result.frd. Just type

"cgx result.frd"

and some information is echoed in the xterm and a new window called main window appears on the screen, see figure 1. Now you should continue with the subsection “Dataset” and “Entity” which explains how to display results. After seeing the values you might play around a bit with the “Menu”. See also the commands “steps”, “max”, “min”, “scal” which might be used to modify the representation. But watch out if you type a command, the cgx window MUST stay active and not the xterm from which the program was started. So better stay with the mouse pointer in the cgx window. At next “Quit” the program and type

“cgx -b geometry.fbd”

in the xterm. The program starts again but now you see only a wire-frame of the geometry. Move the mouse-pointer into the new window and type “mesh all”. The mouse-pointer MUST stay in this window during typing and NOT in the xterm from which the program was started. After you see “ready” in the parent xterm the mesh is created. To actually see it type “plus ea all”. Now you see the mesh in green collor. To see the mesh as a wire-frame chose in the main menu“Viewing” and continue to the entry “Toggle Element Edges” and then again in “Viewing” choose “Points”. To see the mesh illuminated chose in the main menu “Viewing” and continue to the entry “Show Elements With Light”. To see it filled chose in the main menu “Viewing” and continue to the entity “Fill”. Mostly it is sufficient to see only the surface elements, for this purpose chose in the main menu “Viewing” and continue to the entry “Toggle Surfaces/Volumes”. If you start the cgx in the post processor mode as you did in the first example (cgx result.frd) the surface mode is automatically set. To see the interior of the structure chose in the main menu “Viewing” and continue to the entity “Toggle Culling Back/Front”. To save the mesh in the format used by the solver type “send all abq”. To store the mesh in the result format type “send all frd”.

To create a new model start the cgx by typing

“cgx -b file”

where “file” will be the name of the new model if you later exit the program with the command “exit”. The way to create a model from scratch is roughly as follows, create

- points with “qpnt“ or “pnt“,
- lines with “qlin”,
- surfaces with“qsur”,
- Bodies with “qbod”.

If possible you should create higher geometry by sweeping or copying geometry with “swep” or “copy”. It should be mentioned that you might also write a file with basic commands like “pnt” to create the basis for your construction and read it with the "read" command. Most commands can be used in batch mode. Therefore the user might write a command file for repeated actions.

After the geometry is created the divisions of the lines can be changed to control the density of the elements using

- "qdiv".

But before display the lines and there divisions with the command

- "plot ld all".

The default division is "4". With a division of"4" a line will have 6 nodes and will therefore be the edge of two element of the quadratic type. Now the type of the elements must be defined. This can be done individual for the entites of different sets. A new assignment will replace a previous one. Delete all previous assignments with

- "elty all"

and assign new types with

- "elty all he20".

If a mesh is already defined type

- "del mesh"

and mesh again with

- "mesh all".

Then choose the menu entity "Viewing - Show Elements With Light" to see the mesh lighted. At last export the mesh in the calculix solver format with

- "send all abq".

With this command (send) it is also possible to write boundary conditions, loads and equations to files. The equations are useful to "glue" parts together.

It is advisable to save the work from time to time without exiting the program. This is done with the command

- "save".

You leave the program either with

- "exit"

or with

- "quit".

Exit will write all geometry to an fbd-file and if a file of this name exists already then the extension of this file will be renamed from fbd to fbb. Quit just closes the program without a save.

A solver input file can be written with the help of an editor (emacs, nedit etc.). If you write a ccx command file then include the mesh, the boundary conditions etc. with the ccx command "*INCLUDE". After you finished your input-file for the solver (ccx) you might read it by calling the program again with

"cgx -c solverfile.inp"

for a final check. All predefined sets are available together with additional automatically generated sets which store boundaries, equations and more. This sets start with the "+"-sign. For example the set +bou stores all constrained nodes where the set +bou1, +bou2, +bou3 store the constrains for the individual directions. Further the set +dep and +ind store the dependend and independend nodes involved in equations etc. See which sets are defined with the command

- "prnt se".

Each line starts with the set-index, then the set-name followed by the number of all referenced entities. The sets can be specified by index or name. For example if the index of set "blade" is "5" the following commands are equivalent:

- "plot p 5"
- "plot p blade"

Predefined loads are stored as "Datasets" to be visualized. Sets with the name of the load-type (CLOAD, DLOAD) store the related nodes, faces or elements. Use the command

- "plot"

or

- "plus"

to visualize entities of sets.

Then run the input file with ccx. The result file (.frd) can be visualized with

"cgx filename.frd filename.inp"

were the file "filename.inp" is optional. With this file the sets, boundary conditions and loads used in the input-file are available together with the results.

If you have problems doing the above or if you want to learn more and in more detail about the cgx continue with the tutorial [9] and look in the appendix, section Tips and Hints and Known Problems.

5 Program Parameters

```
usage: cgx [-a|-b|-c|-duns2d|-duns3d|-isaac2d|-isaac3d|-foam|-stl|-g] ->
       filename [ccxfile]
 [-v](default) read a result file in frd-format and optional a solver
     input file (ccx) in addition which provides the sets and loads
     used in the calculation.

 -a  automatic-build-mode, geometry file derived from a cad file is
     expected
 -b  build-mode, geometry file in fbd-format is expected
 -c  read an solver input file (ccx, Abaqus)
 -duns2d  read duns result files (2D)
 -duns3d  read duns result files (3D)
 -isaac2d  read isaac result files (2D)
 -isaac3d  read isaac result files (3D)
 -foam    read the OpenFOAM result directory structure
 -stl  read an stl file (triangles)

special purpose options:
 -oldbias      convert pre-1.5 bias
 -mksets       make node-sets from *DLOAD-values (setname: "_<value>" )
```

If no option is provided then a result-file (frd) is assumed, see “Result Format”.

A file containing commands or geometric informations is assumed if the option -b is specified. Such a file will be created if you use the vda2fbd or step2fbd interface or “exit” or “save” after you have created geometry. Option -a is the same as option -b but merging, defining of line-divisions and the calculation of the interiour of the surfaces is done automatically and the illuminated structure is presented after startup. With option -a and -b the program will start also if no file is specified.

An input deck for the solver can be read with option -c, see “send”. Certain key-words are known and the affected nodes or elements are stored in sets. Default sets are +bou(dof) to store nodes which are restricted in the corresponding degree of freedom and the sets +dep(dof) and +ind(dof) to store dependent and independent nodes used in equations.

A special case are results of OpenFOAM. The results are organized in a directory structure consisting of a case containing time-directories in which the result-files are stored. The user must call cgx using the case-directory (cgx -foam case). The program will then search the time-directoies. The time directories must contain a time-file to be recognized. Or in other words each directory in this level containing a time-file is regarded as a result directory.

To convert from pre 1.5 versions start the program with the -oldbias option. This will modify the bias parameters of lines.

6 Input Devices

6.1 Mouse

The mouse is used to manipulate the view-point and scaling of the object inside the drawing area (Figure 1). Rotation of the object is controlled by the left mouse button, zoom in and out by the middle mouse button and translation of the object is controlled by the right mouse button. Inside the menu area the mouse triggers the main menu with the left button.

In addition the mouse controls the animation of nodal values. The animation will stop if the mouse pointer is not in the drawing area but will start again if the pointer enters the drawing area again. This can be prevented by pressing the middle mouse button while the mouse pointer is in the menu area. The animation can be released by pressing the middle button a second time. With the right button each frame can be released separate. This leads as well to a frozen animation which could be released by pressing the middle mouse button while inside the menu area.

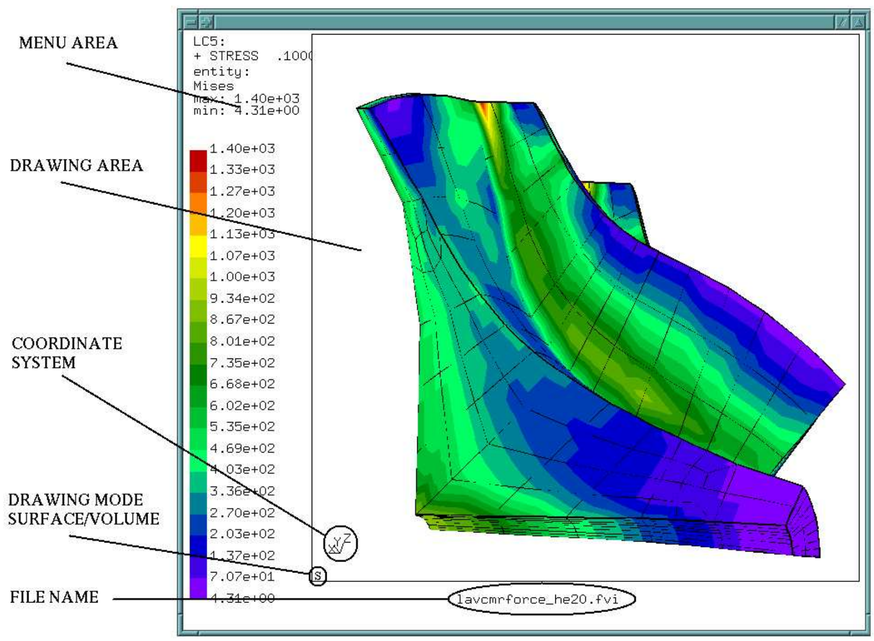


Figure 1: structure of the main-window

6.2 Keyboard

The Keyboard is used for the command line input and moreover for specifying the type of entities while selecting them with the mouse pointer. The command line is preferable in situations where pure mouse operation is not convenient (i.e. to define a certain value). Therefore most preprocessor commands are only available over the command line. The stream coming from the keyboard is echoed in the parent-xterm but during typing the mouse pointer must stay inside the main window. Otherwise the commands will not be recognized by the program.

7 Menu

The main menu pops up when pressing the left mouse-button when the mouse pointer is inside the menu-area of the main window (Figure 2). At next the entities inside the main menu will be explained:

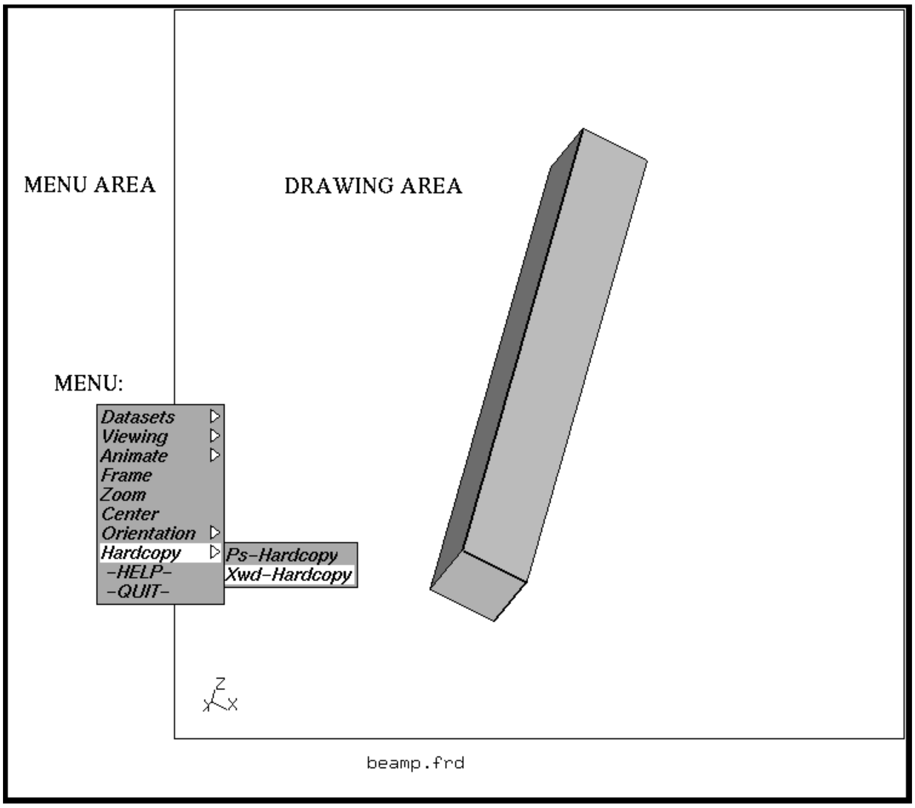


Figure 2: structure of the main-menu

7.1 Dataset

Datasets are selected with the menu-item "Dataset". A dataset is a block of nodal values. These could be displacements due to a linear analysis or for a specific time-step during a nonlinear analysis. It could also contain other values like stresses, strains, temperatures or something else. To select a dataset be sure that the mouse-pointer is inside the menu area. Then press the left mouse button and move the mouse-pointer over the menu entry "Dataset", then continue to the right. A sub-menu pops up showing all available datasets with a leading number and might be followed by a dataset-value (usually time or frequency) and a dataset-description. Move the mouse-pointer over a dataset you are interested in and release the left mouse button. The dataset is now selected. Just an "Entity" must be chosen to see the values. This Dataset might also contain automatically calculated values like the v. Mises stress and the maximum principal stress (see Pre-defined Calculations and Result Format).

7.1.1 Entity

To select data from the Dataset its also necessary to specify the entity (i.e. dx for a displacement Dataset). It works in the same way as for selecting the dataset but instead releasing the left mouse button over a Dataset continue to the bottom of the sub-menu. There a menu item "Entity" is located. Continue from that item to the left and release the mouse button when the pointer is over the entity you want to see. Now the data are displayed in the drawing-area.

7.2 Viewing

How to change properties and styles of the displayed structure will be explained at next:

7.2.1 Show Elements With Light

This is the default viewing of the mesh if the program was started in the viewing mode. If used later on any ongoing animation will be interrupted and no values are displayed. If the command plot or plus was used to display elements, only these elements are rendered.

7.2.2 Show Bad Elements

This option presents the elements which have a negative Jacobian value at least at one integration point. The solver ccx can not deal with those elements. So far only TET and HEX elements are checked. These elements are stored in the set called _NJBY.

7.2.3 Fill

This is the default mode and forces the element faces to be rendered.

7.2.4 Lines

The edges of the element faces will be displayed. This is especially useful if the elements without lighting are displayed. This happens when the "plot" command was used to display the elements with there default color.

7.2.5 Dots

The corners of the element faces will be displayed. This is especially useful if values inside the structure should be checked. For example to identify values inside the structure (see Toggle Surfaces/Volumes).

7.2.6 Toggle Culling Back/Front

This removes the element faces of the volume elements pointing to the user, either of all elements or of the surface of the structure, depending on the state of "Toggle Surfaces/Volumes". With that option the user can visualize internal structures like cracks or a core of a hollow structure.
For shell elements the behavior is different. Initially only the front faces are lighted and the back faces are dark. This is helpful to determine the orientation of the elements. If you want to change the orientation use the command "qflp". If you just want to see all elements lighted then use this option.

7.2.7 Toggle Model Edges

Per default all free element edges are shown, the user might remove them with that option.

7.2.8 Toggle Element Edges

Per default just the free element edges are shown, the user might add all edges to the structure with that option.

7.2.9 Toggle Surfaces/Volumes

This switches the way how much of each element is displayed. Either all faces of the elements or just the element faces on the surface of the structure. Depending on the state of "Toggle Culling Back/Front" either the faces pointing to the user or the faces pointing away from him are displayed. The default is just to show the surface pointing to the user. In the lower left corner of the drawing area of the main window (see Figure 1) is a character printed, indicating if the program is in the surface mode "s" or in the volume mode "v".

7.2.10 Toggle Move-Z/Zoom

Instead of zooming the view of the structure with the help of the middle mouse button, it is also possible to move a clipping plane through the structure to get a view of the inside. The clipping plane is parallel to the screen and will be moved in direction to- and from the user by pressing the middle mouse button and moving the pointer up and down while inside the drawing area. Usually it needs some mouse movements until the clipping plane has reached the structure. Depending on the hardware this functionality could be slow.

7.2.11 Toggle Background Color

With that option it is possible to switch between a black and a white background.

7.2.12 Toggle Vector-Plot

It is possible to add small "needles" to the plot which point with there head in the direction of the vector. Only entities which are marked in the database as vectors will be affected. See "Nodal Results Block" how entities are marked as vectors. Internally calculated vector-results like the worst principal stress are marked automatically.

- The command takes effect after the next entity selection.
- If one component or the value of a vector is selected then the option takes effect.

This option can be used in combination with "Animate Toggle Dataset Sequence".

See also the keyboard command "lc" how to select datasets and entities without the menu. In this case also entities which are NOT marked in the dataset as vectors can be displayed with vector-needles. This command line approach is the only way to display duns-cfd-results with vector-needles. See also the command "scal" how to manipulate the length of the vectors.

7.2.13 Toggle Add-Displacement

It is possible to display results on the deformed structure. For example you can display a stress field on the deformed structure. Some actions in the correct order have to be taken before this feature should be chosen.

- If you know a suitable amplification factor for your displacements then use the "scal" command to issue this value but this can also be done later.
- Then choose the Dataset and the Entity you want to display
- and then the Dataset of the Displacements to be used (but no entity).
- At last use this toggle and the effect takes place.

7.3 Animate

This option allows the animation of nodal displacements. It is possible to create this sequence from just one Dataset, see "Start". This is useful for displaying mode-shapes. See also "Toggle Dataset Sequence" to create a sequence from multiple Datasets to visualize dynamic responses.

7.3.1 Start

Creates a sequence of display-lists to visualize displacements (for example mode-shapes). The program recognizes displacements just by the name of the dataset. This name must start with the letters "DISP", otherwise the animation will not start (see "Nodal Results Block").

7.3.2 Tune-Value

Controls the amplitude of the animation. If "Toggle Real Displacements" was chosen before, the tune-value is equivalent to the amplification of the animation.

7.3.3 Steps per Period

To determine how much display lists for one period of animation will be used. If "Toggle Dataset Sequence" was chosen, then these number of display lists will be interpreted as one period (see Time per Period).

7.3.4 Time per Period

To determine how much seconds one period should need.

7.3.5 Toggle Real Displacements

To see the correct displacement of each node. The animation can be controlled with the help of the mouse.

7.3.6 Toggle Dataset Sequence

Creates a sequence of display-lists to visualize values of a sequence of Datasets. The Datasets must be of the same type, for example only displacements or only stresses. This is very important because otherwhise the program might crash! To activate the animation choose the first Dataset to be displayed, then the second and then the last one. Finally choose the entity. The first two datasets define the spacing between the requested datasets and the third-one defines the last dataset to be displayed. This function will only work if the requested datasets are stored in a unique sequence. Or in other words, if you want to see all displacement-datasets in a sequence, then the number of datasets between all displacement-datasets must be the same.

7.4 Frame

To see the complete structure.

7.5 Zoom

To zoom into a section of the window. After this option was chosen, an imaginary rectangle determining the new window area has to be created by choosing two opposite corners of that imaginary rectangle by pressing the left mouse button on each of these positions.

7.6 Center

To choose a new center point for the structure. After this option was chosen, pick either a node, a point or the corner of an entity. To easily find the element corners choose "Toggle Element Edges" before.

7.7 Orientation

7.7.1 +x View

To look along the x-axis.

7.7.2 -x View

To look against the x-axis.

7.7.3 +y View

To look along the y-axis.

7.7.4 -y View

To look against the y-axis.

7.7.5 +z View

To look along the z-axis.

7.7.6 -z View

To look against the z-axis.

7.8 Hardcopy

To create a hard-copy during animation it is useful to stop the animation first with the middle mouse button while inside the menu area of the main window and then release one picture after the other with the right button until the desired amplitude or step is reached.

7.8.1 Tga-Hardcopy

To create a window dump in tga format. You might use the program "convert" [10] to convert this format to others.

7.8.2 Ps-Hardcopy

To create a window dump in postscript format. The program convert must be installed.

7.8.3 Gif-Hardcopy

To create a window dump in gif format. The program convert must be installed.

7.8.4 Png-Hardcopy

To create a window dump in png format. The program convert must be installed.

7.8.5 Start Recording Gif-Movie

All frames during an animation are stored. The recording ends after the right mouse button was pressed while in the menu area. Finally all frames are combined in the file "movie.gif" which can be displayed with various tools (Netscape [11] or realplay). If the animation is stopped with the middle mouse button while in the menu area then also the movie stops until it is released by pressing the middle mouse button again. See also "movi" for the keyboard options.

7.9 Help

Starts the online help and displays this document. Works only if the specified html-viewer is available. Default is Netscape [11] but this can be changed in the "cgx.h" file. The search-path for the documentation is also defined in the "cgx.h" file. Please make sure that the documentation is in the specified location or change the path in the "cgx.h" file and recompile the program after the object-files were deleted.

7.10 Quit

This terminates the program without a save.

8 Commands

This section is a reference to all commands and there parameters in alphabetic order. If a command is typed the mouse-pointer must be located in the main window (Figure 1). Only the echo of the input stream is visible in the parent-xterm. The keywords are not case sensitive but the names of entities are case sensitive. Each reference starts with a short description of the command. The following syntax is used for these descriptions:

```
'..':   Keyword (either uppercase or lowercase)
<..>:   Parameter (case-sensitive)
[..]:   logical block of parameters or optional parameter
(..):   Remark
|   :   OR
&   :   AND
-   :   from-to
->  :   command continues in the next line
RETURN  press the RETURN key
```

Entities with the exception of nodes and elements are referenced by names which can be build from letters and numbers. Usually one to four characters should be used but no more than eight are permitted. If a new entity uses an existing name the old one will be lost. To overcome the problem "alias" names can be used. An alias name is defined with the ! sign in front. For example:

LINE !L1 %P1 %P2 %SET

will create a line with the alias name L1 and will use the alias names P1 and P2 to define the end-points and uses the set SET to define the point sequence between the end-points.

8.1 area

```
'area' <set>
```

This keyword is used to calculate the area and the center of gravity of a set of shell-elements.

8.2 asgn

```
'asgn' 'n'|'e'|'p'|'l'|'c'|'s'|'b'|'S'|'L'|'se'|->
       'alpha'|'beta'|'nadapt' <value>
```

This keyword is used to define the first node- or element number which will be used for the next mesh generation. And it is used to redefine the leading character of new entities. The default is P for points, L for lines, C for combined lines (lcmb), A for surfaces, B for Bodies, Q for nurb lines (nurl), N for nurb surfaces (nurs) and S for sets. For example

asgn p U

will assign the character U as the leading character to all newly created names of points. The automatically created names of geometric entities use 4 characters.

If all possible names with the chosen leading letter are in use then the alphabetically next letter is chosen as a leading letter, so after PZZZ follows Q000. If no more letter follows then the amount of letters per name is increased. The maximum number is 8. Each entity has his own name-space. So different entities might use the same name. Remark: Currently nurbs-lines are reduced to splines sharing the same name. The original nurbs-lines are not saved in a file after leaving the program. The unstructured mesher [13] uses the tree parameters alpha, beta, nadapt for mesh-control. Current default is 0.4 for alpha and beta and 4 for nadapt.

8.3 bia

```
'bia'  <line> [<bias>]|
'bia'  <set>  [<bias>]|
              ['mult'|'div' <factor>]]
```

This keyword is used to define the bias of a single line or of a set of lines (see qadd). The bias defines the ratio of the length of the first element to the length of the last element. For example,

bia all 4.5

will force a ratio in which the last element is 4.5 times bigger than the first one. Real numbers are permitted since version 1.5 (see alse qbia). To convert from pre 1.5 versions start the program with the -oldbias option.

8.4 body

```
'body' <name(char<9)>|'!' [<surf1> <surf2>]|
                          [<surf1> <surf2> <surf3> <surf4> ->
 <surf5> <surf6>]
```

This keyword is used to define or redefine a volume (body). Each body must have five, six or seven surfaces to be mesh-able but it is sufficient to specify just the "top" and the "bottom" surfaces. The "top" and the "bottom" surfaces are those which are three- or five-sided or have differend line-divisions at opposide sides. Missing surfaces between the "top" and "bottom" surfaces will be created automatically if not already existent but all needed lines must exist. To be more precise only single lines or existing combined lines (lcmb) will be detected. The user must define the missed surface himself if just a chain of lines is defined between two corner points of the "top" and "bottom" surfaces before he can successfully use the body command. It is a more convenient way to define a body than the command gbod but exactly six or two surfaces must be specified otherwise the body will not be created. The most convenient way to define bodies is to use the command qbod. For example, body b1 s1 s2

will look for the missed surfaces and if necessary create them if all lines between the corner points of s1 and s2 are defined and will then create the body b1. Or for example,

body ! s1 s2 s3 s4 s5 s6

will create at first an unused name for the body. This is triggered by the sign `!`. Then it creates the body based on the six surfaces. The body will not be mesh-able if the surfaces are not connected.

8.5 call

```
'call' <parameters>
```

This keyword is used to allow the user to create his own functionallity in the file "userFunction.c". The data-structures for the mesh and datasets are available. The default function calculates the hydrodynamic stresses with the command:

call hydro

See "User-Functions" for the details.

8.6 comp

```
'comp' <set> 'u'|'d'|'e'
```

This keyword is used to add all entities to the specified set (see seta) which depend on the already included entities (u, up), or to include all entities necessary to describe the already included entities (d, down).
For example the set "lines" is made from lines and should also include all dependent points:

comp lines do

Or the set "lines" should also include all surfs and bodies which depend on the lines:

comp lines up

In some cases you will need only the end-points of lines. With the option (e, edges) only end-points are included in the set.

8.7 copy

```
'copy' <set> <new_set> ['scal' <fx> <fy> <fz> <pnt>]|
```

```
['tra' <dx> <dy> <dz>]|
['rot' <p1> <p2> <alfa>]|
['rad' 'x'|'y'|'z'|'p'<PNT> <dr>]|
['mir' <P1> <P2>]
```

This keyword is used to create a copy of a set (see seta about sets). Only geometry but no nodes or elements can be copied. The copy is included in the new set. Several transformations are available. For example scal creates a scaled copy, the scaling factors fx, fy, fz could be chosen independently,

copy part1 part2 scal 2 P0
copy part1 part2 scal 1 1 2 P0

tra will create a copy and will move it away by the vector dx, dy, dz relative to point P0,

copy set1 set2 tra 10 20 30

rot will create a copy and will move it around the axis defined by the points p1 and p2 by alfa degrees,

copy set1 set2 rot p0 px 20.

rad will create a copy and will move it radially to the x-, y- or z-axis or will create a spherical section if just a single point is defined,

copy cylinder1 cylinder2 rad x 20.
copy sphere1 sphere2 rad pP0 10.

mir will create a mirrored copy. The mirror-plane is placed normal to the direction running from P1 to P2 and placed at P2,

copy section1 section2 mir P1 P2.

8.8 corrad

```
'corrad' <set>
```

This keyword is used to modify arc-lines. The center points of arc-lines included in the set are moved in a way that each arc-line will run tangentially into a connected straight line. But because the end-points of the arc-lines are not moved only one side of each arc-line will run into a connected line. Therefore for each arc-line only one connected straight line should be included into the set.

8.9 del

```
'del'  ['p'|'l'|'c'|'s'|'b'|'S'|'L'|'se' <entity>]|
       ['mesh']|
       ['pic']
```

This keyword is used to delete an entity, the whole mesh (see also qdel) or an background-picture. For example,

del se part

will delete the set part but all included entities are still there. There is only no set part any more. The following entities are known:

Points p, Lines l, Combined lines c, Surfaces s, Bodies b, Nurb Surfaces S, Nurb Lines L and Sets se.

When an entity is deleted, all dependent higher entities are deleted as well. If a background-picture was loaded with the "read" command it can be deleted with:

del pic

8.10 div

```
'div'  <line> [<division>]|
'div'  <set>  [<division>]|
              ['mult'|'div' <factor-div> <factor-bias>]|
              ['auto' <node-dist> <angle>]
```

This keyword is used to define the division of a single line or of lines included in a set (see qadd). The division controls the number of nodes that will be created if the geometry is meshed (see elty and mesh). For example,

div all 4

attaches the division of 4 to all lines. With the keyword mult or div in combination with a value it is possible to multiply or divide already asigned divisions:

div all mult 2.

Or in case you need a starting-point for the individual divisions you can use the option auto with the optional parameters node-dist and angle. Where node-dist is the maximum allowed distance between nodes and angle is the maximum allowed angle defined by three sequential nodes. If one parameter is not fullfilled then the division is halfed. Default values are defined in the file cgx.h div all auto

Some rules must be fulfilled before a geometry is meshable. For linear elements (ie. qu4 or he8) the sum of all divisions of each surface must be even. In case of quadratic elements (ie. qu8 or he20) this sum must be divisible by 4 without residue. Opposite edges of a given surface might have different divisions. For example on the left side of a given surface the division is 8 and on the right side it is only 4. But only two opposite surfaces of a body can use this feature. This surfaces are called top and bottom surfaces. All other surfaces of this body must have unique divisions on opposite edges.

8.11 elem

```
'elem' <nr> <firstNode> .. <lastNode> 'be2'|'be3'|'tr3'|'tr6'|->
'qu4'|'qu8'|'he8'|'he20'
```

This keyword is used to define elements based on nodes and the type (see section Element Types in the appendix). For example,

elem 1 1 2 3 4 qu4

creates a shell element with four nodes.

8.12 elty

```
'elty' [<set>] ['be2'|'be3'|'tr3'|'tr3u'|'tr6'|'qu4'|->
             'qu8'|'he8'|'he8r'|'he8i'|'he20'|'he20r']
```

This keyword is used to assign a specific element type to a set of entities (see section Element types in the appendix). The element name is composed of the following parts: The leading two letters define the shape (be: beam, tr: triangle, qu: quadrangle, he: hexaeder), then the number of nodes follow and at last an attribute describing the mathematical formulation or other features (u: unstructured mesh, r: reduced integration, i: incompatible modes). If the element type is omitted the assignment is deleted. If all parameters are omitted the actual asignments are posted:

elty

will print only the sets with assigned elements. Multiple definitions are possible. For example,

elty all

deletes all element definitions. If the geometry was already meshed, the mesh will NOT be deleted and if the mesh command is executed again after new assignments had taken place, additional elements will be created.

elty all he20

assigns 20 node brick-elements to all bodies in the set all.

elty part1 he8

redefines that definition for all bodies in the set part1.

elty part2 qu8

assigns 8 node shell elements to all surfaces in set part2.

8.13 exit

```
'exit'
```

This command terminates the program and saves the geometry (if any) to a file named as the input file but with the extension .fbd. If a file with that name exists already, then this file will be saved with the new file extension .fbb as a backup (see also save).

8.14 frame

```
'frame'
```

This command fits the model into the drawing area.

8.15 gonly

```
'gonly' 'on'|'off'
```

This keyword is used to erase the contents of the menu area. Sometimes useful for generating pictures.

8.16 gbod

```
'gbod' <name(char<9)>|'!' 'NORM' '+|-' <surf> '+|-' <surf> ->
.. ( 5-7 surfaces )
```

This keyword is used to define or redefine a body in the most basic way (see also qbod). Each body must have fife to seven surfaces to be mesh-able. But six surfaces are recommended. The first two surfaces should be the “top” and the “bottom” surfaces. For example,

gbod B001 NORM - S001 + S002 - S005 - S004 - S003 - S006 will create a body B001. The keyword "NORM" is necessary but has no meaning up to now. Then follow the surfaces with a sign "+" or "-" in front that indicates the orientation of each surface. These signs will be corrected automatically if necessary. If the name of the body should be automatically generated, then just type "`!`" instead of a name.

8.17 graph

```
'graph'  [<amplitude> 'amp']|[<material> 'mat']|[<set> 'nr'|'time'|'descr']
```

This keyword is mainly used to create time history plots of nodal values. For this a sequence of datasets (see Toggle Dataset Sequence) is scanned. The values of the pre-selected nodes stored in the referenced set will be written to a file called "graph.out". A gnuplot command file will be written called "graph.gnu" and executed. The resulting file "graph.ps" will be automatically displayed with the postscript viewer. Default is "ghostview" but this can be changed in the "cgx.h" file.

Before such a graph can be drawn, a set with the nodes of interest must be created (see qadd) and a sequence of datasets must be selected (see Toggle Dataset Sequence). Either before or after the selection of the datasets an Entity must be specified. Then the command can be typed and executed. If the correct entity was chosen before the sequence was defined, then the generation of display lists will be suppressed. The user might then use the graph command to generate the history-plot. The command

graph set nr

will display the values on the node-positions vs. the dataset-number. The command

graph set time

will display the nodal-values vs. the dataset-values (usually time) if available and command

graph set descr

will display the nodal-values vs. the dataset-description (makes only sense if the description is a numerical value).

In addition a second data-file is written "graph.out2" which stores the node-number and the node-coordinates and all values at this node-position for all datasets. This file can be used to display values over node-positions etc. But this is not automaticaly prepared. It is up to the user to generate a suitable plot out of this data.

The “graph” files can be edited and combined, for example to plot one type of value vs. one other type of value.

If an ccx- or abaqus-input-file was read then it is also possible to create time history plots of the amplitudes (*AMPLITUDE in ccx) or the material-propperties can be displayed (*MATERIAL in ccx):

graph amplitude-name amp

graph material-name mat

8.18 gsur

```
'gsur' <name(char<9)>|'!' '+|-' 'BLEND|<nurbs>' '+|-' <line|lcmb> '+|-' ->
<line|lcmb> .. (3-5 times)
```

This keyword is used to define or redefine a surface in the most basic way (see also qsur). Each surface must have three to five lines or combined lines (see lcmb) to be mesh-able. But four edges are recommended. For example,

gsur S004 + BLEND - L002 + L00E + L006 - L00C

will create the surface S004 with a mathematical positive orientation indicated by the “+” sign after the surface name. The keyword”BLEND” indicates that the interior of the surface will be defined according to Coons [12] or a NURBS surface is referenced. Then follow the lines or lcmbs with a sign “+” or “-” in front that indicates the orientation. These signs will be corrected automatically if necessary. If the name of the surface should be automatically generated, then use “!” instead of a name.

8.19 gtol

```
'gtol' [<geometric-tol>]
```

This keyword is used to enquire the default geometric tolerance:

gtol

or to determine the default geometric tolerance.

gtol value

Up to this distance the merg command will recognize points or nodes as equally located.

8.20 hcpy

```
'hcpy' ['ps'|'xwd'|'gif'|'png']|[make [ls]]|[clean] (def:xwd)
```

This keyword creates a hardcopy in one of the above formats. Default is xwd. Use the program "convert" to convert to other formats if needed.

hcpy ps

will create a ps file.

hcpy make ls

will bundle all ps files created so far in one file using the landscape (ls) format. The ls parameter might be ommited. A maximum of six pictures are placed on each page.

hcpy clean

will remove all "hcpy" files. See also the commands "text", "rot", "lc", "max", "min" which might be used in hcpy batch files. These commands should be used in the shown order before using the "hcpy" command.

8.21 help

```
'help'
```

This keyword prints a short overview of all commands.

8.22 lc

```
'lc' <1.Dataset-Nr> <2.Dataset-Nr> <n.Dataset-Nr> 'a'|->
['e' <entity-nr> (up to 4 times)]
```

This keyword is used to define a Dataset and an Entity, for example

lc 1 e 1

will display the first entity of the first Dataset. Or to start the animation of displacements right away:

lc 1 a

Also sequences can be defined:

lc 2 4 10 e 1

Here the 1st entity of each second Dataset is selected. The selection starts at the second- and ends at the 10th dataset. If more than one entity is defined then a vector-plot will be displayed. If also a 4th entity is defined then this entity will be used for the basic color-plot:

lc 2 4 10 e 12 13 14 15

8.23 lcmb

```
'lcmb' <name(char<9)>|'!' ['+|-' <line> '+|-' <line> '+|-' ->
<line>..(up to 14 lines)]|
                        ['ADD' '+|-' <line> '+|-' <line> ->
'+|-' <line>..(up to 14 lines)]
```

This keyword is used to define, extend or redefine a combined line (lcmb). Combined lines are necessary if the edge of a surface should be made of more than one line. Usually the user does not create lcmbs directly. It will be created automatically during the process of defining a surface with the command qsur. There is no limitation to the number of lines in general but with one command not more than 14 lines can be specified. To specify more than that or to extend an existing lcmb a modified command has to follow. For example,

lcmb U260 + U249 - U248 - U247 - U243 - U237 - U236 - U231 - U219

defines the lcmb U260 with 8 lines and their orientation in the lcmb. The following command

lcmb U260 ADD - U218 - U217

extends the lcmb U260 by two additional lines.

8.24 length

```
'length' <set>
```

This keyword is used to calculate the length of all lines stored in a set.

8.25 line

```
'line' <name(char<9)>|'!' <p1> <p2> <cp|seq> <div>
```

This keyword is used to define or redefine a line. A line depends on points. A line can only be defined if the necessary points are already defined. There are three different types of lines available. The straight line

line l1 p1 p2 4

is defined by its name l1, the name could have up to 8 characters, by the points p1 and p2 and optionally by the division. The arc

line ! p1 p2 cp 4

needs in addition a center point cp. The radius changes linear from p1 to p2 if the center-point cp is excentric. The name is chosen automatically (triggered by the character !). The spline

line l1 p1 p2 seq 4

needs in addition a so called sequential-set seq (use the command seqa to define such a set). This set seq stores the spline points between the end-points in the right order. The spline function is described in [14]. Usually a line would be defined interactively. This would be done with the command qlin.

8.26 mata

```
'mata' <Material-Nr> <set>
```

This keyword is used to assign a material-number to certain elements. So far this feature is usefull only if you use nastran as a solver. The material-number is an numeric attribute to each element of the mesh and will be stored with the mesh if the frd-format or nastran-format is used (see send). For example

mata 7 part

assigns the material-number 7 to all elements in the set part. Elements can have just one material-number. The default number is 1. This numbers will be saved with the mesh if the database is written to the file-system with the command:

send all frd or send all nas

If the frd-file is used later the material-numbers are available from the beginning.

8.27 mats

```
'mats'
```

This keyword is used to get an overview of all material numbers defined.

8.28 max

```
'max' <value>
```

This keyword is used to define the upper value in the scale in the menu area of the window (see Figure 1).

8.29 merg

```
'merg' 'n'|'p'|'l'|'s' <set> <gtol> 'nolock'
```

This keyword is used to merge close points and nodes or equally defined entities. The following entities are known: Nodes n, Points p, Lines l, Surfaces s. For example to merge points included in the set point-set type

merg p point-set

Only entities included in the set are considered. The value gtol determines the distance up to which nodes and points are considered as equal. The parameter nolock will force the merging even if the dependent entities like lines will be degenerated afterwards, for example a line will have two equal points.

8.30 mesh

```
'mesh' <set> ['fast' |& 'block' |& 'nolength' |& 'noangle']
```

This keyword is used to start the meshing of the model. The element types must be defined before with the command elty. Already existing elements will not automatically be deleted. Therefore its possible to start cgx in the viewing mode (-v) with a mesh alone, then to create bodies and fill them with additional elements. To delete a mesh use the command del mesh. See also the command "send" to describe areas for boundary-conditions.

In case a blocked grid for cfd-calculations should be generated use the additional parameter "block":

mesh setname block

see also the section "Remarks Concerning Duns and Isaac".

The mesh optimizer is controlled with the additional parameters "nolength" and "noangle". This parameters switch off length and angle optimizations of elements. The parameter "fast" suppresses the Nurbs-based meshing and uses the fast coons-algorithm for surface meshing.

Usually all elements are generated in a structured way. That means that no holes in a surface or volume are permitted. The only exception is the element type "tr3u". This type is generated in an unstructured way using the mesher from [13]. In this case holes are permitted and a surface comming from a cad system should be meshable. But unfortunately it does not work for all possible combinations of divisions or all possible shapes of surfaces so far. Also the surface MUST reference a given NURBS-surface to be meshable. This is usually the case if the data are derived from a cad-system with the interface-programs (vda2fbd).

8.31 mids

```
'mids' <set> ['lin']
```

This keyword is used correct the midside nodes of higher order elements stored in a set. It is performed atomatically if a new mesh is created or if nodes are projected to target surfaces. Without the parameter lin the correction will use a circular track defined by the corner-nodes and the midside node. With the parameter lin the corrected position is halfway between the corner-nodes.

8.32 min

```
'min' <value>
```

This keyword is used to define the lower value in the scale in the menu area of the window (see Figure 1).

8.33 minus

```
'minus' 'n'|'e'|'p'|'l'|'s'|'b'|'S'|'L' <set>
```

This keyword is used to remove entities of a set from the screen (see also plus). The following entities are known:

Nodes n, Elements e, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and Nurb Lines L

Only the set which was used to display the entities can be used to remove them.

8.34 move

```
'move' <set> ['scal' <fx> <fy> <fz> <pnt>]|
             ['tra' <dx> <dy> <dz>]|
             ['rot' <p1> <p2> <alfa>]|
             ['rad' 'x'|'y'|'z'|'p'<pnt> <dr>]|
             ['mir' <P1> <P2>]
```

This keyword is used to move nodes or points in a set. Related entities will be moved as well. For example to move a line it is necessary to include their points in the set (see comp). Several transformations are available:

For example scal will scale the entities of the set, the scaling factors fx, fy, fz can be chosen independently and a reference point can be used,

move part scal 2
move part scal 1 1 2

move part scal 2 P0
move part scal 1 1 2 P0

tra will move it away by the vector dx, dy, dz,

move all tra 10 20 30

rot will move it around the axis defined by the points p1 and p2 by alfa degrees,

move all rot p0 px 20.

rad will move it radially to the x-, y- or z-axis or to a single point,

move cylinder rad x 20.
move sphere rad pP0 10.

mir will mirror the set. The mirror-plane is placed normal to the direction running from P1 to P2 and placed at P2,

move part mir P1 P2.

8.35 movi

```
'movi' [delay <sec>]|[start]|[stop]|
       [make <nr> <nr> <prolog.gif>]|[clean]
```

This keyword is used to start or stop the recording of a movie. After "start" all frames will be stored in single gif files until the "stop" command is issued. See also the menu options "Start Recording Gif-Movie". Use the option "make" to assemble the movie from the single files. An already existing movie can be copied in front. With the option "delay" a time-delay in seconds between the single frames can be specified and with the option "clean" all single gif-files will be erased.

Example for a command sequence:

```
movi delay 0.01
movi start
movi stop
movi make 0 500
movi clean
```

8.36 msg

```
'msg' 'on|off'
```

This keyword is used to enable or disable full printout during runtime. Useful for debugging purposes. The default is “off”.

8.37 node

```
'node' <nr> <x> <y> <z>
```

This keyword is used to define or redefine a node. For example

node 23580 10. 0. 1.

defines a node with the number 23580 at the position x=10, y=0, z=1.

8.38 nurl

```
'nurl' <name(char9)>|'!' ['DEFINE' ['COMPACT'] <pstart> <pend>  ->
                        <deg> <npnt> <nknt> <div>]|
                        ['CONTROL' <index> [<pnt>|<x y z>] <weight>]|
                        ['KNOT' <index> <value>]|
                        ['END']
```

This keyword is used to define a nurbs line. So far this command is only used to read a nurbs-line definition. This line will be converted automatically into a spline. Nurbs-lines can be displayed but will not be saved. There are two possible ways of definition. Either by using predefined point-names or by specifying the coordinates explicitly. When the coordinates are defined then the parameter “COMPACT” must be used as shown above but when the point names are used then this parameter must be omitted. CAD-interfaces might use this functionality.

8.39 nurs

```
'nurs' [<name(char9)>|'!' ['DEFINE' ['COMPACT'] <u.deg> <v.deg> ->
                         <u.npnt> <v.npnt> <u.nknt> <v.nknt>]|
                        ['CONTROL' <u.index> <v.index> [<pnt>|<x y z>] ->
                        <weight>]|
                        ['KNOT' <U>|<V> <index> <value>]|
                        ['END']] |
        [ <!> <setname(containing surfaces)>]
```

This keyword is used to define a nurbs shape. Surfaces might use nurbs to define the interior geometry. There are two possible ways of definition. Either by using predefined point-names or by specifying the coordinates explicitly. When the coordinates are defined then the parameter “COMPACT” must be used as shown above but when the point names are used then this parameter must be omitted. CAD-interfaces might use this functionality.

There is also a small nurbs-building capability in cgx. It is possible to use surfaces which do not reference to a given nurbs to build one. This nurbs will follow the Coons-algorithm but can be modified by moving the interiour points. The surfaces must be stored in a set. For example

nurs ! surfaceSet

will define nurbs for all surfaces stored in the set surfaceSet.

8.40 ori

```
'ori' <set>
```

This keyword is used to trigger the orientation of the entities. This is done automatically and it should never be necessary to use it manually.

8.41 plot

```
'plot' 'n'|'e'|'f'|'p'|'l'|'s'|'b'|'S'|'L'&'a'|'d'|'v' <set> ->
'w'|'k'|'r'|'g'|'b'|'y'|'m'|'i'
```

This keyword is used to display the entities of a set. Entities already visible will be erased. The following entities are known:

Nodes n, Elements e, Faces f,Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and Nurb Lines L

The entites can be displayed in the following colors:

White w, Black k, Red r, Green g, Blue b, Yellow y, Magenta m, illuminated i

So far only elements and surfaces can be displayed with lighting effects using color i. To display the entities with attributes, use the identifier in combination with a, d or v. For example

plot la all

will display all lines with there names. The attribute d works only for lines,

plot ld all

will display all lines with there division and bias, that is the number of spaces between nodes which will be created on them if meshed and the ratio of the length of the first element to the length of the last elemen. The division uses the last two numbers and the bias the leading numbers. If there is more than one number in front of the division the shown bias is multiplied by a factor of ten (10130 means a div of 30 and a bias of 10.1). The attribute v works only for nodes,

plot nv set1

will display the nodes in set set1 with there values. This is only meaningful in post-processing mode (see Dataset how to assign values to nodes). To plot additional entities see plus.

8.42 plus

```
'plus' 'n'|'e'|'f'|'p'|'l'|'s'|'b'|'S'|'L'&'a'|'d'|'v' <set> ->
'w'|'k'|'r'|'g'|'b'|'y'|'m'|'i'
```

This keyword is used to display the entities of an additional set after a plot command was used (see also minus). The following entities are known:

Nodes n, Elements e, Faces f, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and Nurb Lines L

The entites can be displayed in the following colors:

White w, Black k, Red r, Green g, Blue b, Yellow y, Magenta m, illuminated i

So far only elements and surfaces can be displayed with lighting effects using color i. To display the entities with attributes, use the identifier in combination with a, d or v. For example

plot la all

will display all lines with there names. The attribute d works only for lines,

plot ld all

will display all lines with there divisions, that is the number of nodes that will be created on them if meshed. The attribute v works only for nodes,

plot nv set1

will display the nodes in set set1 with there values. This is only meaningful in post-processing mode (see Dataset how to assign values to nodes).

8.43 pnt

```
'pnt' <name(char<9)>|'!' [<x> <y> <z>]|
                         [<line> <ratio> <times>]|
```

```
[<P1> <P2> <ratio> <times>]|
[<setname(containing nodes)>]
```

This keyword is used to define or redefine a point. There are four possibilities to define a point: Just with coordinates, for example:

pnt p1 11 1.2 34

or,

pnt ! 11 1.2 34

where the name is chosen automatically. It is also possible to create points on a line or in the direction from P1 to P2 by defining a spacing (ratio) and times, for example

pnt ! L1 0.25 3

or

pnt ! P1 P2 0.25 3

will create 3 new points at the positions 0.25, 0.5 and 0.75 times the length of the line or the distance from P1 to P2, and it is also possible to create points on the positions of existing nodes. The command

pnt ! set

will create new points on the positions of the nodes included in the specified set. Usually points would be defined interactive. This would be done with the command qpnt.

8.44 prnt

```
'prnt' ['se'|'sq' <RETURN|set>]|
       ['n'|'e'|'f'|'p'|'l'|'s'|'b' <entity>]|
       ['amp' <RETURN|amplitude>]|
       ['mat' <RETURN|material>]
```

This keyword is used to print the contents of a set or the definition of other entities. The following entities are known:

Nodes n, Elements e, Faces f, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S, Nurb Lines L, Sets se and Sequences sq

To see all known sets, type

prnt se

or type

prnt sq

to see all known seqential sets. To see the contents of a specific set, type

prnt se setName

To print the definition of a line, type

prnt l lineName

If an ccx- or abaqus-input-file was read then it is also possible to print the amplitudes (*AMPLITUDE in ccx) or the material-propperties (*MATERIAL in ccx):

prnt amp amplitude-name

graph mat material-name

8.45 proj

```
'proj' <set> <target-set>  ['tra' <dx> <dy> <dz> <offset>]|
                           ['rot' <p1> <p2> <offset>]|
                           ['rad' 'x'|'y'|'z' <offset>]|
                           ['nor' <offset>]
```

This keyword is used to project geometry onto a triangle mesh. This mesh can be already existent or it will be automatically created based on surfaces included in the target-set. After the projection all existing elements and nodes and all element definitions will be deleted.

Several transformations are available. For example tra will move points in the direction of the vector dx, dy, dz onto elements or surfaces included in set2. Alternatively an offset could be specified as well,

proj set1 set2 tra 0. 0.5 0.7

rot will move points around the axis defined by the points p1 and p2 onto elements or surfaces included in set2,

proj set1 set2 rot p0 px

rad will move points radially relative to the x-, y- or z-axis onto elements or surfaces included in set2. Alternatively a set of lines could be used instead of surfaces as the target-set. Then the geometry will be moved onto an imaginary rotational surface defined by these lines,

proj set1 set2 rad x

nor will move points in a direction normal to the target surface onto surfaces included in set2. An offset might be specified:

proj set1 set2 nor 0.7

If a point does not hit any surface from the target-set, then it will not be moved.

8.46 qadd

```
’qadd’ <set> RETURN <’w’|’a’|’i’|’r’|’n’|’e’|’f’|’p’|’l’|’s’|’b’| ->
’S’|’L’|’q’|’u’>
```

This keyword is used to add entities to a set (see also seta and group auto). To add entities move the mouse pointer over the entity and press the following keys, for Nodes n, Elements e, Faces f, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and for Nurb Lines L. To catch more than one entity with one stroke type “a” before and create a rectangular picking area by pressing two times the “r” key. Both strokes define opposite corners of the rectangle. To catch only one entity type “i” before. Press “q” to quit the command or “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.47 qali

```
’qali’ RETURN ’w’|’p’|’n’|’q’
```

This keyword is used to align a plane defined by three points or nodes with the screen (working plane). This is useful if a point or a node should be moved manually along a defined plane (see qpnt). To define the plane move the mouse pointer over the first entity and press “n” if its a node or a “p” if its a point. Then define the next two entities in the same way. Press “q” to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.48 qbia

```
’qbia’ RETURN ’w’|’a’|’i’|’c’|’1’-’9’|’ 10’-’ 99’|’q’
```

This keyword is used to change the bias of a line. The bias defines a coarsening or refinement of the mesh along a line. The number defines the ratio of the length of the first element to the length of the last element at a given line. It works by pressing a number between 1 and 9 when the mouse pointer is at the position of a line. To define numbers between 10 and 99 press the space bar when the mouse pointer is at the position of a line and then the number. To select more than one line with one stroke, type “a” before and create a rectangular picking area by pressing two times the “r” key. Both strokes define opposite corners of the rectangle. To select only one line type “i” before. Press “c” to change the direction of the bias. Press “q” to quit the command (see alse bia).

8.49 qbod

```
'qbod' <name>(optional) RETURN 'w'|'b'|'a'|'i'|'r'|'s'|'g'|'q'|'u'
```

This keyword is used to create a body (see also gbod and body). The user might specify a name in the command-line or by picking an existing body with the key “b”. Otherwise the program chooses an unused name. It is possible to create the body out of five to seven surfaces which are needed to define a body or just of two opposite surfaces, but then these two surfaces must be connected on their corner points by lines. To be more precise only single lines or existing combined lines (lcmb) will be detected. If a combined line would be necessary but does not exist then the user should define a surface using this lines which will create the necessary combined-line. Other missed surfaces will be created automatically. To catch more than one surface with one stroke, type “a” before and create a rectangular picking area by pressing two times the “r” key. Both strokes define opposite corners of the rectangle. Type “s” to select surfaces. To catch surfaces individually type “i” before (its also the default). After selecting exactly six or two opposide surfaces press “g” to generate the body. Press “q” to quit the command or “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.50 qcnt

```
'qcnt' RETURN 'w'|'n'|'p'
```

This keyword is used to define a new center-point or -node by pressing “n” or “p” when the mouse pointer is at the position of a node or a point.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.51 qcut

```
'qcut' RETURN 'q'|'n'|'p'
```

This keyword is used to define a cutting plane trough elements to visualize internal results. The plane is defined by picking three nodes or points. The menu option “Show Elements With Light” will display the whole model again and will delete the plane.

8.52 qdel

```
'qdel' RETURN 'w'|'a'|'i'|'r'|'p'|'l'|'s'|'b'|'S'|'L'|'q'
```

This keyword is used to delete entities (see also del). Higher entities (depending ones) will be deleted to. To delete more than one entity with one stroke, type “a” before and create a rectangular picking area by pressing two times the “r” key. Both strokes define opposite corners of the rectangle. To delete only one entity type “i” before. Press “q” to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.53 qdis

```
'qdis' RETURN 'w'|'n'|'p'|'q'
```

This keyword is used to measure distances between two nodes or points. Move the mouse pointer over one entity and press the following key, for a Node n or for a Point p. Then select a second entity. Press “q” to quit the command. After you selected two entities you will get information about the distance and the angle between them relative to the origin and to the x, y, z-axis.

- dist: global distance
- dx, dy,dz: distance in the three cartesian directions
- da: global angle
- dax, day, daz: angle around x, y, z
- dr: global radius difference
- drx, dry, drz: radius difference around x, y, z

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.54 qdiv

```
'qdiv' RETURN 'w'|'a'|'i'|'1'-'9'|' 10'-' 99'|'q'
```

This keyword is used to change the division of a line by pressing a number between 1 and 9 when the mouse pointer is at the position of a line (see also div). To define numbers between 10 and 99 press the space bar when the mouse pointer is at the position of a line and then the number. To select more than one line with one stroke, type "a" before and create a rectangular picking area by pressing two times the "r" key. Both strokes define opposite corners of the rectangle. To select only one line type "i" before. Press "q" to quit the command. General rules are described in "div".

It is also possible to measure distances between two pixels on the screen. Just press the key "w" on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.55 qenq

```
'qenq' RETURN 'w'|'a'|'i'|'r'|'n'|'e'|'f'|'p'|'l'|'s'|'b'|'S'| ->
'L'|'q'
```

This keyword is used to gain information about entities. It is especially useful to get the values on particular nodes. But the position of nodes or points are also given in cartesian and cylindrical coordinates (see Figure 3, axyz are the 3 angles around x,y and z, rxyz are the 3 radii around x, y and z). In a second row the sets to which the picked entity belongs are listed. To see details of entities move the mouse pointer over the entity and press one of the following keys, for Nodes n, Elements e, Faces f, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and for Nurb Lines L. To catch more than one entity with one stroke, type "a" before and create a rectangular picking area by pressing two times the "r" key. Both strokes define opposite corners of the rectangle. To catch only one entity type "i" before. Press "q" to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key "w" on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.56 qflp

```
'qflp' RETURN 'w'|'e'|'s'|'a'|'i'|'q'
```

This keyword is used to invert the outer- and inner side (the orientation) of shell-elements and surfaces. The orientation of shell-elements or surfaces can be seen by the color of the interior. The outer face reflects light, if the face is dark-grey it is the back-side. If "Toggle Culling Back/Front" was selected before then both sides are illuminated. To flip the orientation of a surface and all related shell-elements select either a shell-element with the "e" key or if the illuminated surfaces are displayed select the surface with the "s" key. If only

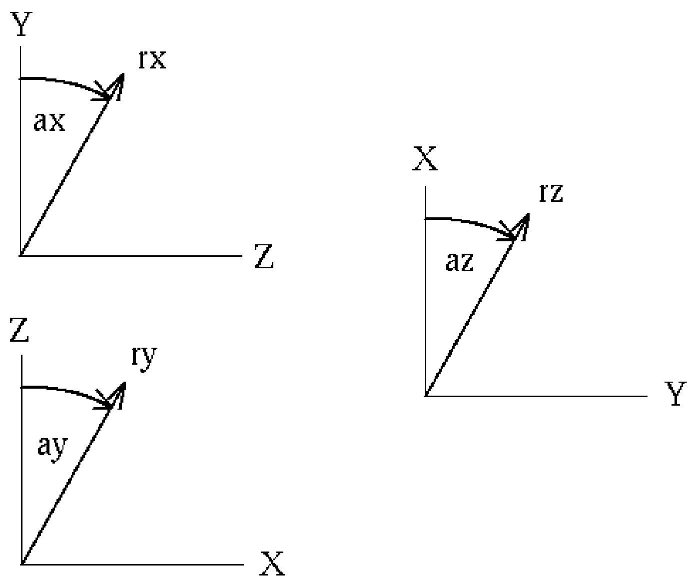


Figure 3: Definition of the cylindrical system

elements are in the database flip them with the “e” key. Choose all in range with the “a” key or individuals with the “i” key before. Press “q” to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.57 qlin

```
'qlin' <name>(optional) RETURN 'w'|'b'|'c'|'m'|'s'|'t'|'g'|'l'|'u'
```

This keyword is used to define a sequence of lines (or just one, see also line). The user might specify a name in the command-line or by picking an existing line with the key “l”. To start a sequence of lines move the mouse pointer over a point and press the key “b” (begin).

To define just a straight line go to the end point and press “g” (generate). This point is also the starting point for the next line.

If the line is a sequential line (spline) then the points of the sequence must be defined with the key “t” in the correct order, but the last point must be selected with the key “g”.

If the line is an arc then the center point must be selected with the “c” key or one point on the line between the end-points must be selected with the “m” key, but the last point must be selected with the key “g”.

In case you need to split a line during the process, you can do that by pressing the “s” key.

Press “q” to quit the command or “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.58 qpnt

```
'qpnt' <name>(optional)RETURN 'w'|'p'|'g'|'m'|'s'|'S'|'u'
```

This keyword is used to create or move points (see also pnt). The user might specify a name in the command-line or by picking an existing point with the key “p”. To create a point move the mouse pointer to the desired location and press the key “g” (generate). To move a point move the mouse pointer to the desired point and press “p” (pick). The point is now marked. If you picked the wrong point, just pick the same again and pick then the correct one. The point can now be moved in different ways: Either in the screen-plane, for this go to the desired position and press “m” (move). Or the point can be moved to the position of a second point, for this go to the second point which coordinates should be used and press “p” again. Also a normal projection to a nurbs related surface (from cad-systems) is feasible by choosing the target-surface with the “s” key or a NURBS-surface with the “S” key, either before or after the point was marked. See qali or the section Orientation how to rotate the model into a certain position. Press “q” to quit the command or “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.59 qnod

```
'qnod' RETURN 'w'|'p'|'m'|'u'
```

This keyword is used to move nodes (see also node). To move a node move the mouse pointer to the desired node and press “p” (pick) then go to the desired position and press “m” (move). See qali or the section Orientation how to rotate the model into a certain position. Press “q” to quit the command or “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.60 qrem

```
'qrem' <set> RETURN 'w'|'a'|'i'|'r'|'n'|'e'|'f'|'p'|'l'|'s'|'b'|'q'
```

This keyword is used to erase entities from a set (see also setr). To remove entities move the mouse pointer over the entity and press the following keys, for Nodes n, Elements e, Faces f, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S and for Nurb Lines L. To catch more than one entity with one stroke type "a" before and create a rectangular picking area by pressing two times the "r" key. Both strokes define opposite corners of the rectangle. To catch only one entity type "i" before. Press "q" to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key "w" on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.61 qseq

```
'qseq' RETURN 'nr'
```

This keyword is used to redefine a straight- or arc-line as a sequence-line with "nr" spline-points. This is usefull in case the line should be projected onto a set of lines, surfaces or elements (see proj).

8.62 qspl

```
'qspl' RETURN 'w'|'s'|'q'
```

This keyword is used to split one or more lines at a certain position. A point is created at the splitting position, the original line is deleted and two new lines will appear instead. All lines running through the selected location will besplitted at once and the newly created splitting points will be merged to one if they are closer to each other than defined by "gtol". To split a line move the mouse pointer over the line and press the "s" key. Press "q" to quit the command.

It is also possible to measure distances between two pixels on the screen. Just press the key "w" on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.63 qsur

```
'qsur' <name>(optional) RETURN
'w'|'a'|'b'|'i'|'r'|'1'-'9'|'g'|'q'|'u'|'s'|'S'
```

This keyword is used to create a surface (see also gsur). The user might specify a name in the command-line or by picking an existing surface with the key "s". For the definition three to five edges of the surface must be defined. To create a surface move the mouse pointer over the first line of the first edge and select it with the “1” key. If more than one line is necessary to define the first edge select the following ones one after the other with the “1” key. Each marked line is listed in the mother xterm. If all lines of the first edge are selected select the lines of the second edge by pressing the “2” key, then the third and eventually the fourth and fifth. The last selected line must match the first. All lines defining an edge will create a combined line (see lcmb) and this combined line will show up in the definition of the surface. Then generate the surface by pressing the “g” key and you might continue with the next surface. By default the interior of the surface is defined according to Coons [12] which is a blended function. But it can also follow an embedded NURBS surface. To relate the surface to an existing NURBS surface select the NURBS surface with the “S” key (uppercase). Or assign the blended function with the “b” key. To quit the command use the “q” key or use “u” to undo the last action.

It is also possible to measure distances between two pixels on the screen. Just press the key “w” on the positions of the two pixels. The distance is calculated in the scale of the displayed geometry.

8.64 quit

```
'quit'
```

This keyword is used to terminate the program without save.

8.65 read

```
'read' [<geo-file> ['add']]|
       [<ccx-file> 'inp' ['add']]|
       [<mesh-file> <setname>]|
       [<edge-file> 'edg']|
       [<list-file> '-n'|'-e'&<column>]|
       [<picture.xwd>]
```

This keyword is used to read files or commands from a file. Most commands can be read from a file but not all of them. In general all basic commands to create geometry are understood and up to know this is the only way to read pre defined geometry or a mesh during run-time. To read commands from a file like pnt, line, node, or seta and so on, type

read filename

this will eventualy replace entities if their names were already in use. To prevent this, type

read filename add

this forces the program to change the names which are already in use to unused ones. Therefore no existing geometry will be overwritten. This command is usefull to combine independend models.

If a mesh exists, it is possible to read files written in the result format (.frd) during runtime to define sets of nodes or elements. The command

read mesh.frd part

will create a set of the name part and all nodes and elements listed in the file mesh.frd will be added to that set. But this will NOT create or modify the nodes and elements. All nodes and elements must exist before and will not be canged.

It is also possible to read an ccx-input-file with loads, boundary-conditions and sets:

read filename inp

The loads and boundary-conditions are stored in automatically created sets which start wit a leading '+'. For example the ccx command '*BOUNDARY' will create '+bou' in cgx. The ccx-set-contents are stored in cgx-sets. The load-values are stored in Datasets. If the parameter "add" is used, then existing node- or element-numbers are not overwritten and the program choses new numbers.

In case NETGEN is used for meshing then the edges which are generated by NETGEN can be included for modifications:

read filename edg

will create beam elements based on the defined edges. When the modifications are done the beam elements can be exported in the NETGEN-edge format with the "send setname stl" command. See also "How to deal with cad-geometry".

If a file with the numbers of nodes or elements are not available in the result format (.frd) then a so called list-file could be read instead. In such a file either numbers of nodes or elements can be defined. The numbers found in a specified column is interpreted as a node- or element-number. If "-n" is specified the numbers are interpreted as nodes and if "-e" is specified the numbers are interpreted as elements. For example

read list-file.txt -e3

will define a set storing names of elements from the third column of file list-file.txt. The created set-name is always the name of the file.

If a pixel-file in xwd-format of a certain geometry is available it can be included as a background-picture. The user can then create geomety based on this picture:

read pixelfile.xwd

8.66 rep

```
'rep' <set>
```

This keyword is used to refresh entities of a certain set. This is done automatically but with one exception: The rendering of the interior definition of surfaces must be triggered manually with this command. The rendered surfaces can be displayed with “plot” or “plus” using color “i”.

8.67 rnam

```
'rnam' <set> <set>
```

This keyword is used to rename a set.

8.68 rot

```
'rot'  ['u'|'d'|'r'|'l'|'c' <angle>]|
       ['x'|'-x'|'y'|'-y'|'z'|'-z']
```

This keyword is used to rotate the view-direction to the model. For example

rot u 10

will rotate the view-direction 10 degrees upwards. The meaning of the other letters is down d, right r, left l, clockwise c. The remaining letters will rotate the view into a pre-defined direction. For example the user will look in x direction after the command

rot x

and against the x direction with the command

rot -x

and so on.

8.69 save

```
'save'
```

This keyword is used to save the geometry (if any) to a file named as the input file but with the extension .fbd. If a file with that name exists already, then this file will be saved with the new file extension .fbb as a backup.

8.70 scal

```
'scal' ['s'|'v'|'d'] [<value>]
```

This keyword is used to scale several items independently. For example the command

scal s 0.5

will scale the range of values presented in the color plot by a factor of 0.5. The command

scal v 5.

will scale the length of vectors by a factor of 5. The command

scal d 2.

will scale the deformed shape by a factor of 2. Without a value it restores the default value. See also "Toggle Vector-Plot", "Toggle Add-Displacement", "Dataset".

8.71 send

```
'send' <set> 'abq'|'ans'|'ast'|'duns'|'fbd'|'frd'|'nas'| ->
             'stl'|'tcg' <RETURN>|
             ['lc <nr>']|
             ['tmf']|
             ['sta' <refspeed>]|
             ['crp' <timefact> <refspeed> <writefreq>]|
             ['comp']|
             ['mpc' <rotation> <v1> <v2> <v3>]|
             ['names']|
             ['spc' <dofs(1-6)> <value>]|
             ['slide' 's'|'rx'| ->
  'ry'|'rz'|'tx'|'ty'|'tz']|
             ['pres' [<value>]|['lc'<nr> 'e'<nr>]]|
             ['film' [[<temp>]|['lc'<nr> 'e'<nr>]] ->
  [[<coeff>]|['lc'<nr> 'e'<nr>]] ]|
             ['rad' [[<temp>]|['lc'<nr> 'e'<nr>]] ->
  [[<emis>|['lc'<nr> 'e'<nr>]] ]|
             ['dflux' [[<load>]|['lc'<nr> 'e'<nr>]] ]|
             ['cflux' <load>]|
             ['force' <f1> <f2> <f3>]|
             ['sur']
'send' <set> ['duns'|'isaac'|'foam'] [<base-type> <set>]| ->
  ['cyclic' <set> <set> 'tx'|'ty'|'tz'|'rx'|'ry'|'rz'| ->
```

```
                           'cx'|'cy'|'cz'|<vx,vy,vz>]
'send' <dep-set> <indep-set> 'nas'|'abq'|'ans' ['cycmpc' ->
  'rx'|'ry'|'rz'|'cx'|'cy'|'cz'<segments> 'c'|'u'<NR>]| ->
  ['areampc' <dofs(1-6,t)> 'c'|'u'<Nr>]| ->
  ['areampc' 'slide']| ->
  ['gap' <vx> <vy> <vz> <tol>]
```

This keyword is used to send data to the file-system. The following formats are known (but not all options for all formats are fully supported so far); The geometry-format (fbd)

send all fbd

and a surface description with triangles (stl)

send all stl

Beside the stl file a so called edge file for NETGEN will be created if be2 elements are included in the mesh. The following mesh-related formats are known: Femview and Fam (frd), Nastran (nas), Abaqus (abq), Ansys (ans), CodeAster (ast), Samcef (sam) and Tochnog (tcg) but only ccx (Abaqus) is fully supported. If no further parameter follows then just the definition of the nodes and elements will be written:

send all abq

will write the mesh in the format used by Abaqus and the CalculiX solver. If the parameter “lc” follows

send set abq lc 1

then the mesh plus temperatures stored in Dataset 1 is written. If the parameter “tmf” follows

send set abq tmf

then the mesh plus all temperatures with the necessary solver context for tmf calculations is written. If the parameter “sta” follows

send set abq sta 2900.

then the mesh plus all temperatures with the necessary solver context for a static calculations is written. Here the value for “speed” is a reference value for eventual nummerical values in the TEXT-Block of the Datasets (see “Nodal Results Block”). If the parameter “crp” follows send set abq crp 1. 2900. 1

then the mesh plus all temperatures with the necessary solver context for creep calculations is written. Here the value for "timefact" scales the time-values of the Datasets, "speed" is a reference value for eventual nummerical values in the Text-Block and scales values from a default load-file and writefreq limits the output to the result-file.

Special cases are the cfd-solvers Duns, Isaac and OpenFoam. The boundary patches are an integral part of the mesh. So it is necessary to specify all boundary patches when writing the mesh. All free surfaces of the mesh must be specified. This is an example for OpenFoam:

send all foam cyclic cyc1 cyc2 cx patch in patch out wall wall

will write the so called polyMesh description to the file-system. After the send command the set (all) with the mesh is specified, then the format (foam), then cyclic boundary conditions (cyclic cyc1 cyc2 cx) between set cyc1 and cyc2 of the axi-symmetric case around x (cx), then boundary conditions of type patch for set "in" (patch in), then boundary conditions of type patch for set "out" (patch out) and boundary conditions of type wall for set "wall" (wall wall).

The symmetric boundary-conditions (base-type: cyclic) can be axi-symmetric (c) around x,y,z or rectangular (r) in direction of x,y,z. Only for OpenFoam and in the rectangular (kartesian) case also a vector pointing in the direction of the symmetry can be specified (ie: 1.,1.,0.)

For isaac and duns the same syntax has to be used:

send all duns viscous-wall profil subsonic-inflow in subsonic-outflow out

send all isaac WALL profil FARFIELD far EXTRAPOLATE out

The command

send set frd

will write nodes and/or elements in frd format. This file could be read in again later on to define sets during post-processing (see read) or just to look at it in a new session. If the parameter "comp" follows and the set includes not all nodes used by the included elements then missed nodes will be detected and stored as well in the set before the set is written, for example:

send elem frd comp

will at first include all nodes used by the elements to the set elem and will then write the file.

If the parameter “names” follows the solver then just the element- or node-numbers are written. This is useful for defining sets for the solvers:

send steel abq names

will write the the element numbers included in “steel”. This file could be used to define a set for material assignments. If the solver-format “duns” is used then related numbers of surfaces (3D) or lines (2D) will be written. This information is necessary to apply boundary-conditions to duns. The numbers are used in the connectivity file (duns.conn) which will be created by the mesh-command. The user has to refer this numbers in the duns.script2 file when describing boundary conditions.

It is also possible to define and send some constrains to the file system. To constrain degrees of freedom (dof) of selected nodes use the parameter “spc” in combination with the numbers of the constrained dofs and optionally a forced deflection. For example:

send hinge abq spc 123

will constrain the translational degrees of freedom of the nodes in set “hinge”. Or

send hinge abq spc 12356

will leave just one rotational degree of freedom unconstrained. And

send moved-nodes abq spc 1 0.1

will move the nodes in set moved-nodes by 0.1 in 1 direction.

The parameter “slide” in combination with one additional parameter will create equations that force the selected nodes to move only along a plane. If the additional parameter “s” is specified then the specified nodes will be attached to their element faces. But this will work only if enough nodes are selected to fully describe the element faces to which the nodes belong. For example you type:

send sur abq slide s

and this should attach the node with the node-nr. 1 to a plane parallel to the surface of the corresponding element-face. This element uses the node 1, 2, 3, 4, 5, 6, 7 and 8 where 1, 2, 3 and 4 are on the surface of the mesh. The program will detect this element face and constrain the node only if the nodes 2, 3 and 4 were also included in the set “sur”. The detected elements or faces of volume-elements will be assigned to the set. So it is possible to check the detected elements. Instead of providing a set with the necessary nodes, you might instead provide a set with shell-elements or/and faces of volume-elements. The nodes will be derived from them. In most cases this is the most convenient way.

If nodes should slide radially to the y axis then use “ry”:

```
send sur abq slide ry
```

or use “ty” if the nodes should move only tangential to the y axix. In both cases the axial movement is still permitted.

The parameter “pres” is used to assign pressure values to element faces. For example

```
send surf abq pres 0.05
```

will assign the pressure value 0.05 to all element-faces in set surf. If a dataset with nodal values is available it is also possible to use this values instead of using a uniform value for all faces. For example

```
send surf abq pres lc1 e1
```

will specify the dataset-nr 1 and the entity-nr 1 to be used instead of a uniform value.

The parameter “film” is used to assign free-stream temperatures and thermal heat coefficients to element faces. It works similar to the “pres” option. For example

```
send surf abq film 1200. 0.5
```

will assign the film temperature of 1200 and the coefficient 0.5 to all element-faces in set surf. If datasets with nodal values are available it is also possible to use this values instead of using a uniform value for all faces. For example

```
send surf abq film lc1 e1 0.5
```

will specify the dataset-nr 1 and the entity-nr 1 to be used for the temperature instead of a uniform value, or

```
send surf abq rad lc1 e1 lc1 e2
```

will specify also the dataset-nr 1 with the entity-nr 2 to be used for the film-coefficients instead of a uniform value.

The parameter “rad” is used to assign a sink temperatures and the emissivity to element faces. It works similar to the “pres” option. For example

```
send surf abq rad 1200. 0.5
```

will assign the sink temperature of 1200 and the emissivity 0.5 to all element-faces in set surf. If datasets with nodal values are available it is also possible to use this values instead of using a uniform value for all faces. For example

send surf abq rad lc1 e1 1200.

will specify the dataset-nr 1 and the entity-nr 1 to be used for the temperature instead of a uniform value, or

send surf abq rad lc1 e1 lc1 e2

will specify also the dataset-nr 1 with the entity-nr 2 to be used for the emisivity instead of a uniform value.

The parameter "dflux" is used to assign an energy stream to element faces. It works similar to the "pres" option. For example

send surf abq dflux 0.5

will assign the flux of 0.5 to all element-faces in set surf. If a dataset with nodal values is available it is also possible to use this values instead of using a uniform value for all faces. For example

send surf abq dflux lc1 e1

will specify the dataset-nr 1 and the entity-nr 1 to be used instead of a uniform value.

The parameter "cflux" is used to assign an energy stream to nodes. For example

send surf abq cflux 0.5

will assign the flux of 0.5 to all nodes in set surf.

The parameter "force" is used to assign force values to nodes. For example

send nodes abq force 1. 20. 0.

will assign the specified forces to all nodes of the set nodes.

The parameter "mpc" is used to create input for the user-subroutine umpc which forces all nodes from "set" to rotate by an average value specified with "rotation" around the vector v. The value has to be in degree were 90 degree is orthogonal. For example

send nodes abq mpc 4. 1. 0. 0.

will assign the nodes of the set "nodes" to the user-subroutine umpc and will force them to rotate by 4 degree around the x-axis.

The special purpose parameter "sur" will write the surfaces of the mesh either in abaqus-format for the *SURFACE command or in frd-format as shell elements with a dataset describing the normals. For example

send top abq sur

will write the elements in set top together with the face-identifier of the free element-faces.

In addition it is possible to define so called “cyclic symmetry” equations. These equations are used when just a section of a rotation-symmetric part like a disk is modeled. These equations force the two cutting planes of such a section to move exactly equal in the cylindrical system. If the coordinate system of the displacements for the solver is rectangular (xyz) then the syntax is:

send dep indep nas cycmpc rx12 c1

Here “dep” is the set containing the nodes of the dependent side. These nodes will be replaced by the solver with the independent nodes from the set “indep”. In this case the equations will be written in the nastran format “nas” (in nastran called MPC). The parameter “rx12” defines the displacement system as rectangular “r”, the rotational axis is “x” and the “12” defines the number of segments in 360 deg, therefore the angle of the segment is 360 deg /12. If omitted then the value is calculated for each node separate. The “c” triggers the correction of the position of the dependent nodes to a position defined by the angle of the segment (highly recommended), “u” would prevent the correction. The “1” will be the identifier for the equations if the format is nas (nastran). In case the format would be ans (ansys) then the “1” would be the number of the first equation. No number is required for abq (abaqus and calculix). If the coordinate system of the displacements is cylindrical (rtz) then the example would be:

send dep indep nas cycmpc cx12 c1

Only the “r” from “rx12” is changed to “c”. A thermal connection is created with:

send dep indep nas cycmpc tx c

The thermal connection is triggered by the “t”. See comments above for the single parameters.

Another useful method is to “glue” independent meshes together. For this the dependent nodes are tied to independent elements by equations. Choose the finer mesh for the dependent side. The equations are based on the shape-functions of the element types. For example

send dep indep nas areampc 123 c1

will connect the nodes in the set dep to element-faces described by nodes included in the set indep. The set dep must contain all nodes which should be "glued" and the set indep should contain all nodes of the elements surfaces to which the dep nodes should be glued. The numbers "123" are the degrees of freedom which will be connected ("t" will create a thermal connection). The "c" triggers the correction of the position of the dependent nodes to a position on the surface of the independent elements (highly recommended), "u" would prevent the correction. The "1" will be the identifier for the equations if the format is nas (nastran). In case the format would be ans (ansys) then the "1" would be the number of the first equation. No number is required for abq (abaqus and calculix).

A special case is considered with the "slide" option in combination with the "areampc" option. For example if the mesh of a turbine-blade and a disk should be connected to each other then a sliding condition between this parts is necessary. The command:

send dep indep abq areampc slide

will connect the nodes in the set dep to element-faces described by nodes included in the set indep but only in the direction perpendicular to a plane defined by nodes of the dep-set. Therefore all dep-nodes and all indep-nodes must lie in the same plane and will slide in the same plane!

Also it is possible to define so called "gap" elements and related control-commands. These elements will connect parts if they are closer as a certain distance. For example if the distance is zero (contact). The command:

send dep indep abq gap 1. 0. 0.

will connect the nodes in the set dep and indep with gap-elements but only if they match each other in the direction 1.

8.72 seqa

```
'seqa' <set> ['pnt' <point> .. <=>]|
             ['afte'|'befo' <point> <point>.. <=>]|
             ['end' <point> .. <=>]
```

This keyword is used to create or redefine a set marked as a sequential set. This set is used to define splines (see line, with the command qlin such a set is automatically created). To begin such a set type for example:

SEQA Q003 PNT P004 P005 P006 P00M P00N

Because of the parameter PNT the program will create or overwrite the set

Q003. The command will continue in the next line if the sign “=” is found:

SEQA Q003 PNT P004 P005 P006 =
P007 P008 P009

The parameter AFTE will insert additional points after the first specified point. The parameter BEFO will insert additional points before the first specified point and the parameter END will add additional points to a sequential set.

8.73 seta

```
'seta' <set> 'n'|'e'|'p'|'l'|'c'|'s'|'b'|'S'|'L'|'se' <name ..> |
              ['n'|'e' <name> '-' <name> <steps>]
```

This keyword is used to create or redefine a set (see also qadd). All entities like points or bodies and so on must be stored at least in one set to be reachable. The set “all” is created automatically at startup and will be open (see seto) all the time unless explicitly closed (see setc). To add entities to the set “dummy” type:

seta dummy p p1 p2

This will add the points p1 and p2 to the set. The following entities are known:

Nodes n, Elements e, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S, Nurb Lines L and names of other sets se. If other sets are added then the contents are NOT automatically added. Please use also “comp do” to add the contents.

The program will automatically determine the type of the entities if not specified, but then the names must be unique. More than one name can be specified. A minus sign between two names specifies a range of names with steps of “steps”.

8.74 setc

```
'setc' [<set>]
```

This keyword is used to close an open set. Without parameter setc will close the last opened set.

8.75 seto

```
'seto' [<set>]
```

This keyword is used to enquire open sets:

seto

Or to mark a set as open:

seto set

All newly defined or redefined entities will be members of all open sets. See setc how to close an open set.

8.76 setr

```
'setr' <set> 'n'|'e'|'p'|'l'|'s'|'b'|'S'|'L'|'se' <name ..>
```

This keyword is used to remove entities from a set (see also qrem. The entity will not be deleted. It is just not longer a member of that set. To remove entities from the set dummy type:

setr dummy p p1 p2

This will remove the points p1 and p2 from the set. The following entities are known: Nodes n, Elements e, Points p, Lines l, Surfaces s, Bodies b, Nurb Surfaces S, Nurb Lines L and names of other sets se. The program will automatically determine the type of the entities if not specified, but then the names must be unique.

8.77 split

```
'setr' <set1> <set2>
```

This keyword is used to split lines and to create points on surfaces included in set1 where they cross surfaces from set2.

8.78 steps

```
'steps' <value>
```

This keyword is used to define the number of different colors in the post-processor mode. The default is 21.

8.79 surf

```
'surf' <name(char<9)>|'!' <line|lcmb> <line|lcmb> <line|lcmb> ->
<line|lcmb>
```

This keyword is used to define or redefine a surface. It is a more convenient way to define a surface than the command gsur but it requires 3, 4 or 5 edges (lines or lcmbs) otherwise the surface will not be created. The most convenient way to define surfaces is to use the command qsur. If the name of the surface should be automatically generated, then just type “!” instead of a name.

8.80 swep

```
'swep' <set> <set> ['scal' <fx> <fy> <fz> <P0>|<div>]|
                        ['tra' <dx> <dy> <dz> <div>]|
                        ['rot' <p1> <p2> <alfa> <div>]|
                        ['rad' 'x'|'y'|'z'|'p'<pnt> <dr> <div>]|
                        ['mir' <P1> <P2> <div>]
```

This keyword is used to sweep geometry (no nodes or elements) into the next higher dimension. Sweeping a point will create a line, sweeping a line will create a surface and sweeping a surface will create a body. At first a copy of the first set (see seta and copy) will be created. The copy is included in the second set. Then the connecting lines and surfaces are created and at last the bodies. The divisions of the new lines between set1 and set2 is specified with the parameter "div" or the default is used. Several transformations are available. For example scal creates a scaled copy, the scaling factors fx, fy, fz could be chosen independently,

swep part1 part2 scal 2
swep part1 part2 scal 1 1 2 P0

tra will create a copy and will move it away by the vector dx, dy, dz relative to point P0,

swep set1 set2 tra 10 20 30

rot will create a copy and will move it around the axis defined by the points p1 and p2 by alfa degrees (the connecting lines will be arcs),

swep set1 set2 rot p0 px 20.

rad will create a copy and will move it radially to the x-, y- or z-axis or will create a spherical section if just a single point is defined,

swep cylinder1 cylinder2 rad x 20.
swep sphere1 sphere2 rad pP0 10.

mir will create a mirrored copy. The mirror-plane is placed normal to the direction running from P1 to P2 and placed at P2,

swep section1 section2 mir p1 p2.

8.81 text

```
'text' <string>
```

This keyword is used to define a comment. This commend will show up in the menu area of the main window below the file name.

8.82 tra

```
'tra' 'u'|'d'|'l'|'r' <relative-distance>
```

This keyword is used to move the model in the window. For example

tra u .1

will move the model 0.1 times the window-heigh upwards. The meaning of the other letters is down d, right r, left l.

8.83 volu

```
'volu' <set>
```

This keyword is used to calculate the volume and the center of gravity of a set of volume-elements.

8.84 zap

```
'zap'  <set>
```

This keyword is used to delete all entities of a set and the set itself. All depending entities will be deleted as well.

8.85 zoom

```
'zoom' [<scale>]|[<p1x> <p1y> <p2x> <p2y>]
```

This keyword is used to scale the model in the window. For example

zoom 2

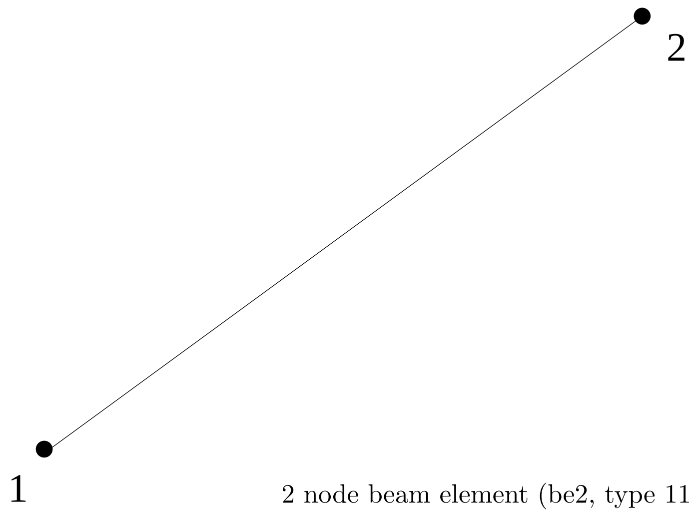
will increase the size of the representation of the model by a factor of 2. A certain region of the model can be specified with two corner points of an imaginary rectangle. The coordinates are relative to the graphic-window which has its origin at the left/lower corner and as a fraction of the edge-lengths. For example

zoom 0. 0. 0.5 0.5

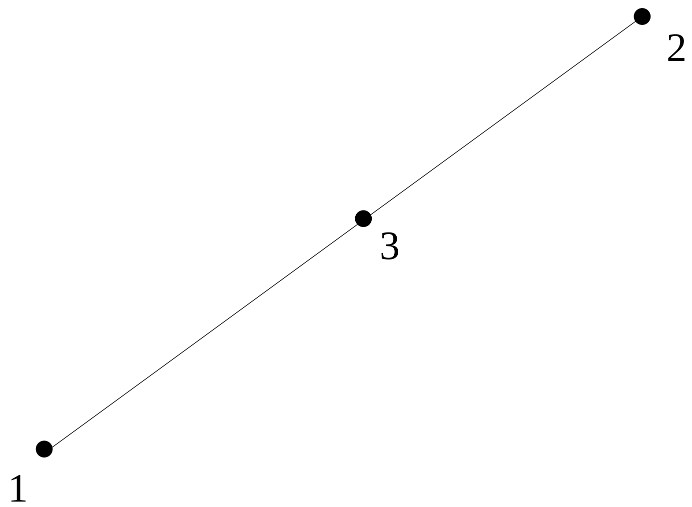
will display the third quadrant of the window scaled by a factor of 2.

9 Element Types

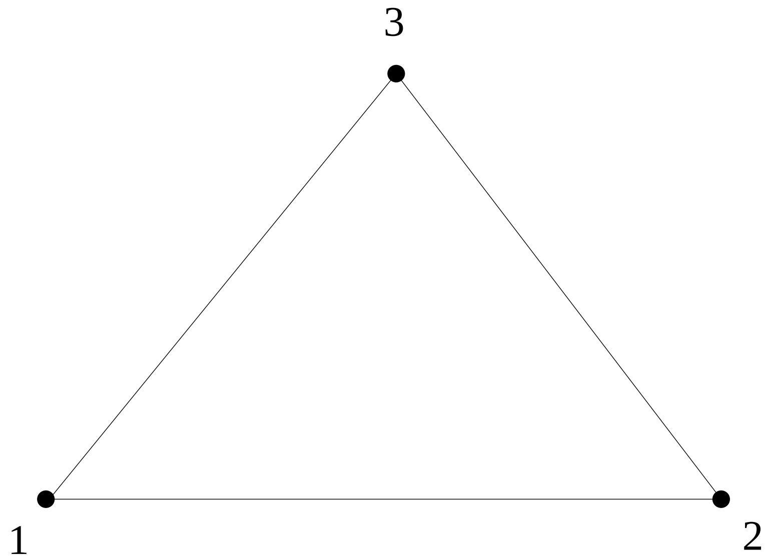
Node numbering of the elements and the type numbers used in the Result Format (frd-file). The solvers might use different node-numbering rules.



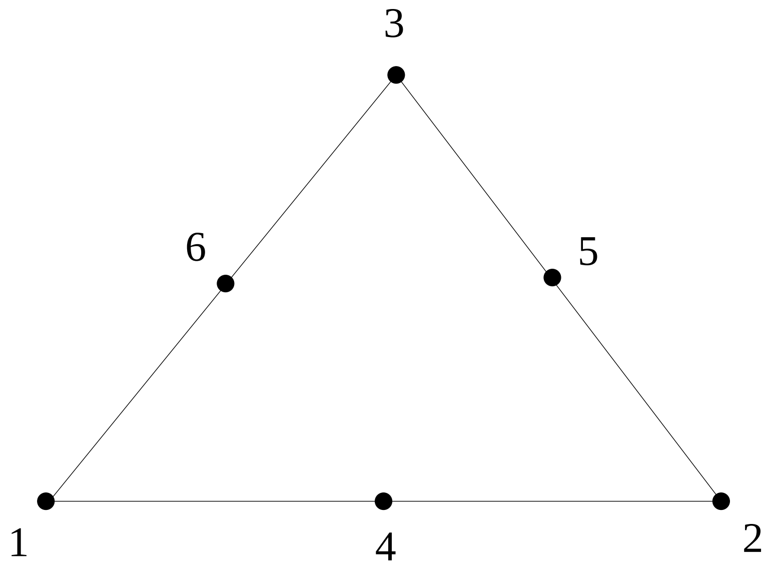
2 node beam element (be2, type 11)



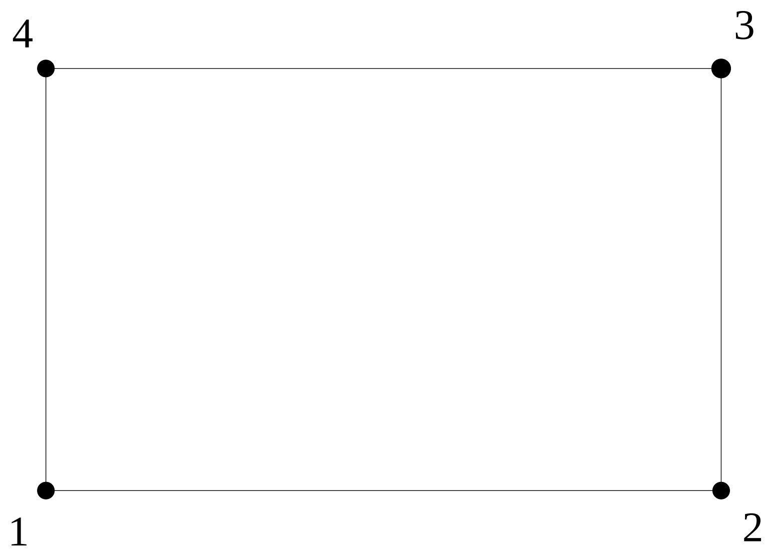
3 node beam element (be3, type 12)



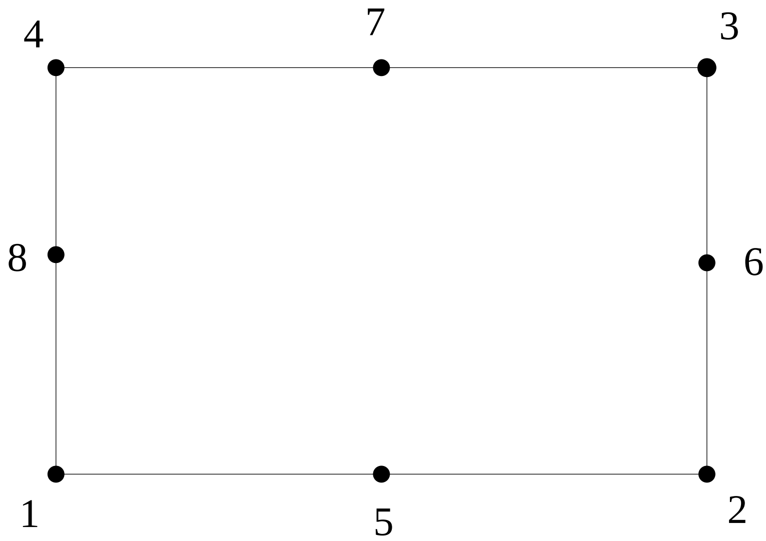
3 node shell element (tr3, tr3u, type 7)



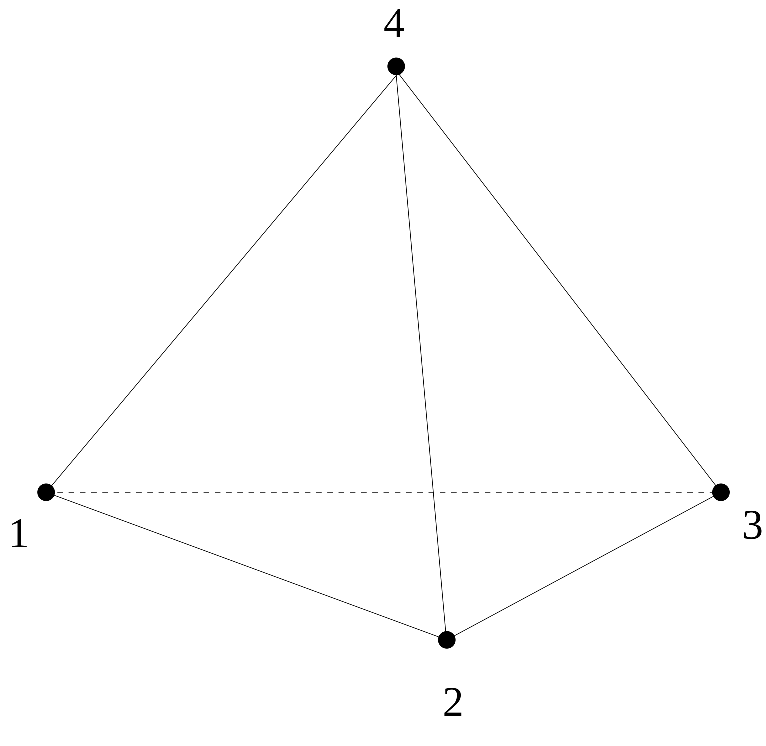
6 node shell element (tr6, type 8)



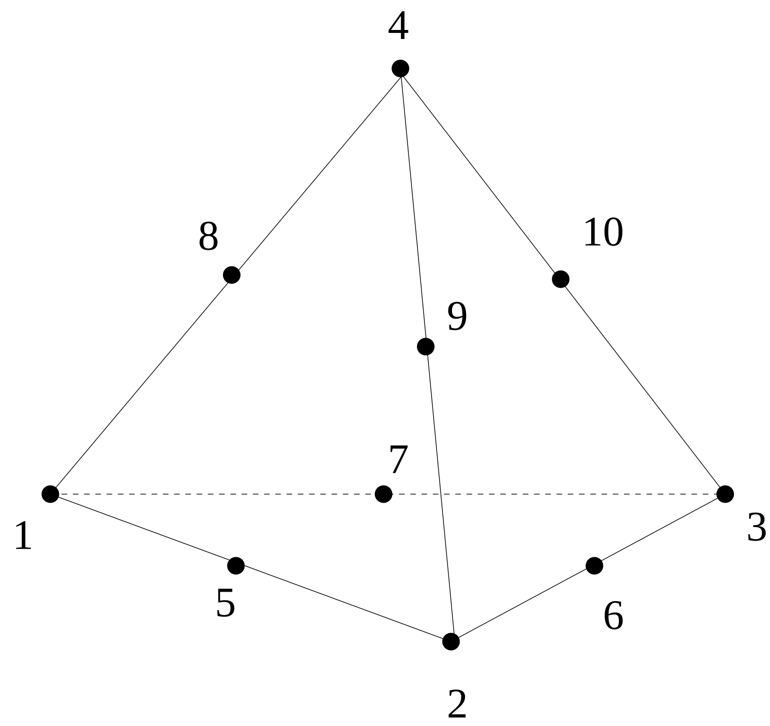
4 node shell element (qu4, type 9)



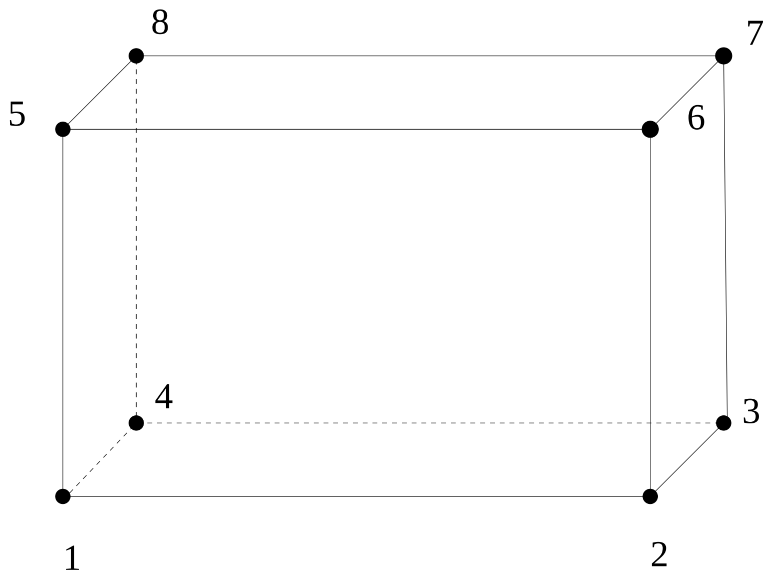
8 node shell element (qu8, type 10)



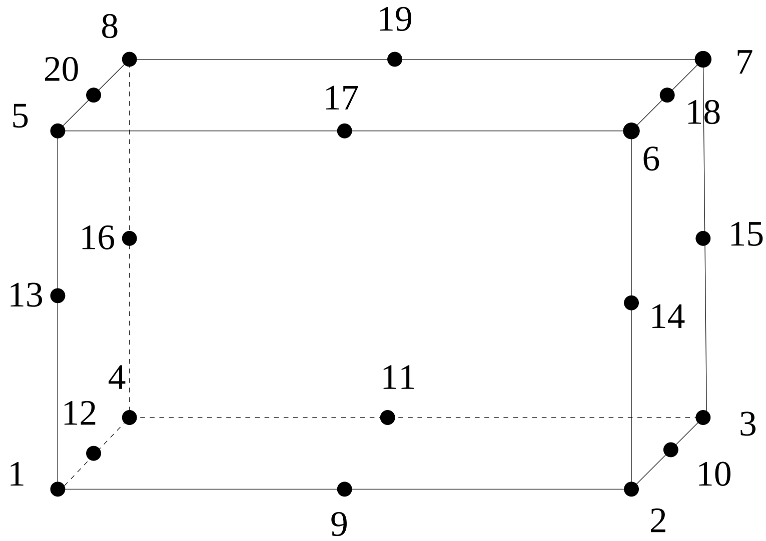
4 node tet element (type 3)



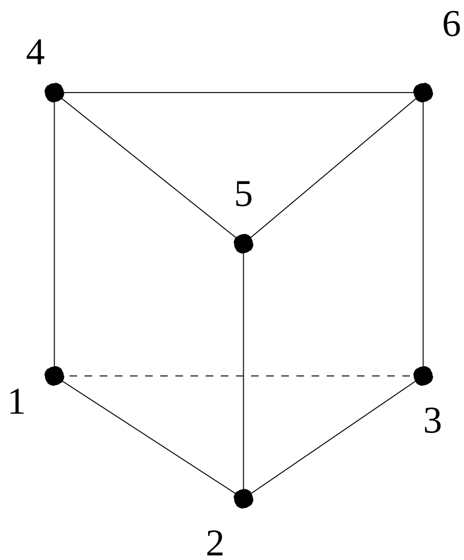
10 node tet element (type 6)



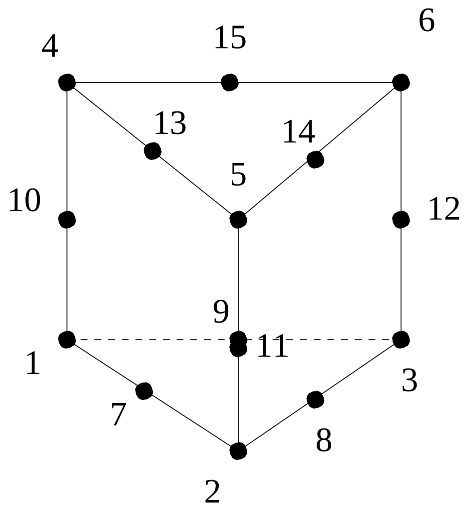
8 node brick element (he8, type 1)



20 node brick element (he20, type 4)



6 node penta element (pe6, type 2)



15 node penta element (pe15, type 5)

10 Result Format

Listing of the format of the mesh and the nodal results. The data are stored in fixed format. The format is able to store element- and node-numbers up to '99999' in the short form or up to '9999999999' in the long form. An example for the short form is shown below:

```
>    1Ctest              !''1C'' defines a new dataset of name ``test''
>    2C                  !''2C'' starts a block of node coordinates
> -1    1 0.00000E+00 0.00000E+00 0.00000E+00 ! 1. node
> -1    2 0.10000E+01 0.00000E+00 0.00000E+00 ! 2. node ....
> -3                     !end of the current block
>    3C                  !''3C'' starts a block of element definitions
> -1    1    4    0    0 !first element, type of that element is 4 (he20)
> -2    1    2    3    4   13   14   15   16    5    6    7    8    9   10   11
> -2   12   17   18   19   20  !twenty nodes defining that element
> -1 ...
> -2 ...
> -2 ...
> -3                     !end of the current block
>  100CL101              !''100C'' starts a user defined result block
> -4  DISP        3    1                !Attribute Header Record (Dataset)
> -5  D1          1    2    1    0      !Component Definition Record (Entity)
> -5  D2          1    2    2    0
> -5  D3          1    2    3    0
> -1    1 0.00000E+00 1.00000E+00 1.00000E+00  !Nodal Values
> -1    2 1.00000E+00 0.00000E+00 0.00000E+00
> -3                     !end of the current block
> 9999                   !end of data
```

10.1 Model Header Record

```
Purpose: Defines the name of the model (not used but necessary)

Form: KEY, CODE, NAME
Format:(1X,'   1','C',A6)
```

10.2 Nodal Point Coordinate Block

```
Purpose: Defines the nodal coordinates

1. Record:
Form: KEY, CODE, FORMAT
Where: KEY    = 2
       CODE   = C
       FORMAT = Format indicator
                0 = short format
                1 = long format
Format:(1X,'   2','C',67X,I1)

Following records:
Form: KEY, NODE, X,Y,Z
Short Format:(1X,'-1',I5,3E12.5)
Long Format:(1X,'-1',I10,3E12.5)
Where: KEY    = -1
       NODE   = node number
       X..    = koordinates

Last Record:
Form: KEY
Format:(1X,'-3')
```

10.3 Element Definition Block

```
Purpose: Defines the topology of the elements

1. Record:
Form: KEY, CODE, FORMAT
Where: KEY    = 3
       CODE   = C
       FORMAT = Format indicator
                0 = short format
                1 = long format
Format:(1X,'   3','C',67X,I1)
```

```
The following block of records must be repeated for each element:
  The first record initializes an element definition:
  Form: KEY, ELEMENT, TYPE, GROUP, MATERIAL
  Short Format:(1X,'-1',I5,3I5)
  Long Format:(1X,'-1',I10,3I5)
Where: KEY    = -1
       NODE   = element number
       TYPE   = element type, see section ``Element Types''
       GROUP  = element group number, see command ``groups''
       MATERIAL= element material number, see command ``mats''.

  Then the nodes in the correct order have to follow:
  Form: KEY,NODE,NODE,NODE,NODE, ...
  Short Format:(1X,'-2',15I5)
  Long Format:(1X,'-2',10I10)
Where: KEY    = -2
       NODE   = node number

Additional lines must follow if more nodes are used.

Last Record:
Form: KEY
Format:(1X,'-3')
```

10.4 Nodal Results Block

```
Purpose: Stores values on node positions

1. Record:
Format:(1X,' 100','C',6A1,E12.5,12X,20A1,I2,I5,10A1,I2)
Values: KEY,CODE,SETNAME,VALUE,TEXT,ICTYPE,NUMSTP,ANALYS,FORMAT
Where: KEY    = 100
       CODE   = C
       SETNAME= Name (not used)
       VALUE  = Could be frequency, time or any nummerical value
       TEXT   = Describing text or any ascii characters
       ICTYPE = Analysis type
                0  static
                1  time step
                2  frequency
                3  load step
                4  user named
       NUMSTP = Step number
       ANALYS = Type of analysis (description)
       FORMAT = Format indicator
```

```
                0  short format
                1  long format

2. Record:
Format:(1X,I2,2X,8A1,2I5)
Values: KEY, NAME, NCOMPS, IRTYPE
Where: KEY    = -4
       NAME   = Dataset name to be used in the menu
       NCOMPS = Number of entities
       IRTYPE = 1  Nodal data, material independant
                2  Nodal data, material dependant
                3  Element data at nodes (not used)

3. Type of Record (and more):
Format:(1X,I2,2X,8A1,5I5,8A1)
Values: KEY, NAME, MENU, ICTYPE, ICIND1, ICIND2, IEXIST, ICNAME
Where: KEY    = -5
       NAME   = Entity name to be used in the menu
       MENU   = 1
       ICTYPE = Type of entity
                1  scalar
                2  vector with 3 components
                4  matrix
               12  vector with 3 amplitudes and 3 phase-angles in degree
       ICIND1 = sub-component index or row number
       ICIND2 = column number for ICTYPE=4
       IEXIST = 0  data are provided
                1  data are to be calculated by predefined
                   functions (not used)
       ICNAME = Name of the predefined calculation (not used)
                ALL  calculate the total displacement if ICTYPE=2
This record must be repeated for each entity.

4. Type of Record (and more):  (not used)
This record will be necessary in combination with the request for
predefined calculations. So far all types of predefined calculations
are done by default.
Format:(1X,I2,2I5,20I3)
Values: KEY,IRECTY,NUMCPS,(LSTCPS(I),I=1,NUMCPS)
Where: KEY    = -6
       IRECTY = Record variant identification number
       NUMCPS = Number of components
       LSTCPS = For each variant component, the position of the
                corresponding component in the attribute definition
```

```
The following records are data-records and the format is repeated for
each node.

In case of material dependant data
REMARK: Implemented only for NMATS=1
- first line:
Short Format:(1X,I2,4I5)
Long Format:(1X,I2,I10,3I5)
Values: KEY, NODENR, NMATS
Where: KEY    = -1
       NODENR = Node number
       NMATS  = Number of different materials at this node (unused)
- second and following lines:
Short Format:(1X,I2,I5,6E12.5)
Long Format:(1X,I2,I10,6E12.5)
Values: KEY, MAT, XX, YY, ZZ, XY, YZ, ZX ..
Where: KEY    = -2
       MAT    = material-property-number if KEY=-2 (unused)
       XX..   = data


In case of material independant data
Short Format:(1X,I2,I5,6E12.5)
Long Format:(1X,I2,I10,6E12.5)
Values: KEY, NODE, XX, YY, ZZ, XY, YZ, ZX ..
Where: KEY    = -1 if its the first line of data for a given node or
                -2 if its a continuation line
       NODE   = nodenumber or blank if KEY=-2
       XX..   = data
```

11 Pre-defined Calculations

Listing of the automatically calculated additional results.

11.1 Von Mises Equivalent Stress

Entity name: Mises

$$\sigma_{vM} = \frac{1}{\sqrt{2}}\sqrt{(\sigma_x - \sigma_y)^2 + (\sigma_y - \sigma_z)^2 + (\sigma_z - \sigma_x)^2 + 6\tau_{yz}^2 + 6\tau_{zx}^2 + 6\tau_{xy}^2}$$

11.2 Special Surface Stress

Entity name: xx+yy

$$\sigma_{(xx+yy)} = \sigma_{xx} + \sigma_{yy}$$

11.3 Principal Stresses

Entity names: P1, P2, P3, worstPS, worstPSx, worstPSy, worstPSz

The principal stresses σ are named P1, P2, P3. From the three principal stresses σ the absolute maximum value will be calculated and named worstPS. For example if a node has the three values 100, 0 and -110 MPa then -110 MPa would be shown. The three principal stresses σ_1 σ_2 σ_3 are derived from the following equation:

$$\begin{bmatrix} \sigma_{xx}-\lambda & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy}-\lambda & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz}-\lambda \end{bmatrix} \begin{bmatrix} nx \\ ny \\ nz \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

They are given by the three roots of the equation (stress tensor is symmetric: $\sigma_{xy} = \sigma_{yx}$ etc.):

$$\sigma^3 - (\sigma_{xx} + \sigma_{yy} + \sigma_{zz})\sigma^2 + (\sigma_{xx}\sigma_{yy} + \sigma_{yy}\sigma_{zz} + \sigma_{zz}\sigma_{xx} - \sigma_{xy}^2 - \sigma_{yz}^2 -$$

$$\sigma_{zx}^2)\sigma - (\sigma_{xx}\sigma_{yy}\sigma_{zz} + 2\sigma_{xy}\sigma_{yz}\sigma_{zx} - \sigma_{xx}\sigma_{yz}^2 - \sigma_{yy}\sigma_{zx}^2 - \sigma_{zz}\sigma_{xy}^2) = 0$$

In addition the the three components of the vector of worstPS are stored as well. The entity-names are: worstPSx, worstPSy, worstPSz.

11.4 Tresca Stresses

Entity name: Tresca
This is the maximum shear-stress:

$$Tresca = 0.5 * (\sigma_1 - \sigma_3)$$

11.5 Cylindrical Stresses

Entity names: SXX, STT, SRR, SXT, STR, SRX

The Cylindrical Stresse are truly cartesin stresses in a cylindrical system regarding the node-psition. So the stress-tensor is rotated individually for each node in a way that the x-direction is considered as the cylindrical axis and y points in the tangential direction and z in the radial direction.

12 User-Functions

The user might define his own functions to manipulate the mesh or the results with the user function stored in file “userFunction.c”. See the command “call” how to call a user function. The user can generate new nodes, elements or datasets or extend or manipulate existing datasets or interfaces to other software. The file “userFunction.c” includes an example which calculates the hydrostatic stress and stores the result in a new dataset. There the user can see how to deal with datasets.

A Known Problems

A.1 Program is not responding

If the program seems to hang then leave the window with the mouse pointer and go in again. If that does not help then probably a command is waiting for input. Stay with the mouse pointer inside the window and press the "q" key several times. Another very popular error is to move the mouse-pointer into the xterm were the typed commands and the protocol is visible. Please, the mouse-pointer MUST stay in the main-window during typing!

A.2 During Meshing

If a body can not be meshed although no obvious error is in its definition then the reason could be that the the shape of the body is very far from being brick-like. Define more bodies to improve their shapes. For the moment there are some restrictions for the definition of five- or seven-sided bodies. For example the first two surfaces in the body-topology (see gbod) have to be defined in the same order. That means, the first line of the first surface has to be connected with the first line in the second surface by one of the remaining surfaces. This weaknesses will disappear in the next release.

A.3 Program generates a segmentation fault

Write a mail to the author and, if possible, add the input-file.[3].

B Tips and Hints

B.1 Selection of entities

Some knowledge is necessary to efficiently select entities (nodes, points ..). As a golden rule in complicated situations never try to create a set just by "adding" the entities to the set (see qadd). It is much better to catch a bigger group which includes certainly the requested ones. Then display the set with the plot command and remove all unwanted entities with the qrem command. Entities are selected if at least one pixel of it is inside the selection rectangle. You might add all lower entities (as points for example) by typing "comp set do" to fully describe the contents of the set. This is necessary for example if you had selected surfaces and you want to move them in space. Only points have locations and therefore nothing will happen unless you completed the set by the related points with "comp".

B.2 How to select only nodes on the surface

Most times you need to select nodes only on the surface of the mesh. This can easily be done if the mesh is displayed in the surface mode (see "Toggle Surfaces/Volumes") using the menu-entry "Show Elements With Light" or during post-processing of results. To find node possitions more easily add the element edges to the view (see Toggle Element Edges).

B.3 How to enquire node numbers, values at nodes and more

A very common problem is how to get the exact value on a node position during post-processing. To actually get the value add the element edges to the view (see Toggle Element Edges) and type qenq and press the RETURN key (during typing the mouse-pointer MUST stay in the main window, do NOT move the pointer into the xterm). Then move the mouse-pointer over the location of a node and press the "n" key. The node-number, the value at that node and the location will be written in the xterm from which the cgx was started.

B.4 How to generate a time-history plot

So called time history plots can be created based on a previous displayed sequence of data-sets (see "Toggle Dataset Sequence") with the command "graph". Sometimes the user wants to avoid the animation. This can be done if the required entity was selected before the sequence of datasets were selected. Do not select the entity then but use the graph command instead. It is also possible to use only the command line. For example:

- lc 1 e 3
- lc 1 2 10
- graph set t

will produce a time history plot for the nodes stored in set over all loadcases from 1 to 10 for the entity nr 3.

In case you need nodal-values along a sequence of nodes (i.e. stress vs. x-axis) you might use the "prnt" command. Then you grap the information from the xterm with the mouse-pointer and store it in a file with the help of an editor. The contents of that file could be displayed with gnuplot. Or you use the file "graph.out2" which stores the node-number and the node-coordinates and all values at this node-position for all datasets. This file is also generated by the "graph" command.

B.5 How the mesh is related to the geometry

Very often you need the embedded nodes or elements of geometric entities to apply boundary conditions. If you understand the underlying concept you will be able to get them.

Geometric entities are the mothers of nodes and elements (so to say) and will remember them. In turn if an entity is not the mother of a given node it will not remember it. Therefore a body will only know about nodes which are not placed on surfaces, lines or points. A surface will only know about nodes which are not placed on lines or points. And so on.

Therefore if you need the nodes on a surface and not only the ones just inside the surface, then create a set with this surface and do a “comp set do”. This will add the lines and points together with there nodes to the set.

On the other hand if you have only a node and you need the geometric entity in which the node is embedded you might also type “comp set do”.

B.6 How to deal with cad-geometry

A mesh based on hexader-elements has the better performance during the calculations but if the mesh should be derived from a cad-geometry it is often more convenient to create a tetraeder mesh as to modify or rebuild the geometry to make it meshable with hexader-elements. In this case the user can mesh the surfaces with unstructured triangles and export them in the stl-format (see “send”) as a basis for an auto-mesher [4]. The following steps have to be done:

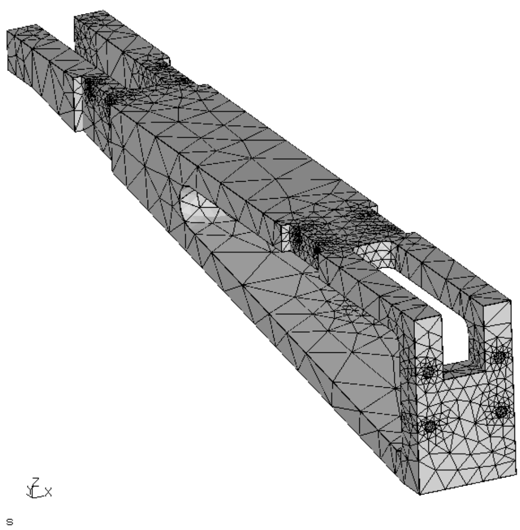


Figure 4: CAD-file meshed with tet-elements using NETGEN

- Use the interface program to convert the cad format to fbd format

  `(ie: vda2fbd cad.vda > cad.fbd).`

  Sometimes the header of the vda file causes some trouble and must be modified to make the program run.

- Start cgx with that file (cgx -a cad.fbd). Usually some warnings appear on the screen but the program will fix that automatically (triggered by -a). After all geometry has been read the program will merge points and lines to close the volume. Then all nurbs related surfaces are trimmed. Finally all surfaces are rendered and the result is displayed in the main window. The whole process might need some time. If problems occur the user might start the program with option -b instead of -a (cgx -b cad.fbd). Then no automatic post-processing is done and the user has to fix the geometry manually. See the commands merg, div, qdiv, rep. Quite often adjacent surfaces do not use common lines. Make sure the divisions match. This has to be done because later on every created triangle must have three adjacent triangles.

- Flip the surfaces in a way that its normal direction points outwarts (use "qflp" with key "a"). If a surface points inwards it is not illuminated and appears only dark grey.

- Define the element type, usually: elty all tr3u. In rectangular situations the element type tr3 gives better results. If you use NETGEN [4] later on for the generation of the tet-mesh then you should also prepare edges for the mesher. Usually it is good enough if you mesh all lines Use be2 elements for the edges.

- Mesh all (mesh all). If some surfaces can not be meshed then modify the divisions of the lines of this surfaces (see "qdiv") or chose the structured element type tr3 for this surfaces. Or change some parameters of the unstructured mesher (see "asgn"). See also the commands "qdel" and "qele" to fix elements manually.

- Visualize and check the mesh (plus ei all).

- Export the mesh (send all stl). If the model consists of several unconnected parts separate them in single sets and send one after the other in stl format (see qadd qrem comp). Beside the stl file a so called edge file for NETGEN will be created if be2 elements are included in the mesh.

- Create the tet-mesh outside of cgx. If you use NETGEN then better use the self created edges. In NETGEN open the stl-doctor and go in the edges menu. There delete all edges with "all undefined" then load the edges with "load edgedata" and activate them with "candidate to confirm". Then mesh the model. Use Abaqus format if you export the mesh out of NETGEN.

- Back in cgx delete the existing mesh (del me) and import the Abaqus-tet-mesh (read mesh.msh inp) or start a new session (cgx -c mesh.msh). If the mesh is composed of several parts then assemble all parts by reading one after the other with "read".

- Create sets of nodes, faces or elements to create boundary conditions and export them with “send”.
- Create the input deck for ccx with an editor and start the calculation.
- Look at the results with cgx (cgx results.frd).

B.7 Remarks Concerning Duns and Isaac

If you intent to create a 2D-mesh for the cfd-code duns or isaac you have to watch that all surfaces are created in the same order. That means, that all surfaces must be defined clockwise or counter-clockwise. For isaac they have to be counter-clockwise if you look in z direction. The block-structure can be detected only in this case. You can check the mesh by simply mesh it with shell-elements and display it illuminated. All elements must be either dark or lighted. See also “Program Parameters”, “mesh”, “send” with option “nam”.

B.8 Remarks Concerning OpenFOAM

The mesh can be written in OpenFOAM polyMesh format with the command “send”. If you work in the polyMesh-directory of the OpenFOAM case then all mesh-related files will be already in place. So far the physical-type is not written in the boundary file as it is not mandatory. The results of an OpenFOAM calculation can be viewed by specifying the parameter -foam and the case. See also “Program Parameters”.

Since openFoam-version 1.3 uses a new format it is required to convert the resulting files. After exporting the OpenFOAM mesh, i.e. boundary, cells, faces, points, copy these 4 files to an OpenFOAM case, say, project (that will be under project/constant/polyMeh). Then, from the parent directory of project, do ”checkMesh . project”. This will convert the mesh to OpenFOAM 1.3 format (it will create 2 new files, owner and neighbour). That is it!

B.9 Remarks Concerning Code Aster

From Paul CARRICO (2005/02/12)

Brief presentation of CODE ASTER: Code Aster is an implicit solver under GPL licence from the French company EDF (Electricit de France).

Code Aster and its documentations is downloadable at the following address:

http://www.code-aster.org

(NOTA : the documentation is in French at the moment but many users are translating it in English and in German)

The following list is not exhaustive but it briefly presents the capabilities of the solver:

- Mechanical modelisations: Code aster allows linear and non linear calculations in static, dynamic, acoustic etc. Many mechanical laws are implemented in the solver such as damage, fatigue, creep, viscosities (elastic, plastic) etc. For isotropic and non-isotropic materials (orthotropic ones for example). Because of EDF fields, the materials used in Code Aster are essentially Metallic ones and Geomaterial one, but there are probably some others.

- Thermal and thermomechanical calculations: The Thermal solver performs linear and non-linear calculations for pure thermal but also for thermomechanical simulations.

- Input interface: EFICAS is the input interface for coding the input file, but it's not a pre-processing as you can find in many commercial code.

- Tools: Many tools are provided with Code Aster: HOMARD for mesh refinement, GIBI and GMSH for post-processing.

CGX to ASTER export format (i.e HOWTO use this export format) The export format allows to export meshes and sets from CGX to Code Aster for linear and quadratic 1D, 2D and 3D eleme nts. For this, just type: 'send all aster' for exporting a complete mesh. The file will have the '.mail' extension. Type 'send set aster nam' for exporting the GROUP-NO (node group) and the GROUP-MA (mesh group) which compose the set. The file will have '.nam' extension

The later item is particularly useful to apply boundary conditions onto (DOF, pressure, force, displacement, temperature, etc.). Since of these boundary conditions are coded with EFICAS in the input file, I thought it was not necessary to develop another features than the 2 previous ones !

B.10 Remarks Concerning Samcef

From Paul CARRICO (2006/04/17)

BASIC TUTORIAL FOR HOWTO USE THE SAMCEF EXPORT FORMAT

```
1- Definition of the points
K: pnt p0 0 0 0
K: pnt p1 1 0 0
K: pnt p2 0 1 0
K: pnt p3 2 1 0
K: plot pa all

2- Definition of the lines
K: plus l all
K: qlin (link the points p0 p1 p3 p2 p2 p0)
```

3- Creation of the first surface
K: qsur

4- Creation of l0 (between p0 & p1) and l1 (between p1 & p3) sets
K : qadd lo
K : qadd l1

4- Creation of the 2 other surfaces
K : swep l0 l1 tra 0 -3 0
K : swep l1 l1b tra -3 0 0

5- Creation of the SYMETRY set
K : plot s all
K : qadd SYMETRY (use both a and rr keys to select all the surfaces)

6- Creation of the volumes
K : swep SYMETRY s1 swep tra 0 0 1 (all the volume will be automatically created)

7- Looking for common points, lines and surfaces
In the order :
K : merg p all
K : merg l all
K : merg s all

8- Creation of the LOAD set and ANCHORAG one
K : qadd LOAD (use rr keys to select the surface)
K : qadd ANCHORAG (use rr keys to select the surface)

NOTA : It's easy to verify the different sets ; for example :
K : plot b all (you can see all the volumes)
K : plus s LOAD (you can see the set LOAD)
K : plus s ANCHORAG
K : plus s SYMETRY

9 - Mesh
K : plot ld all
K : div all mult 2
K : elty all HE20 (to specify HEXAEDRA with 20 nodes)
K : elty LOAD qu8 (to mesh the set LOAD otherwise no quads will be created)
K : elty ANCHORAG qu8
K : elty SYMETRY
K : mesh all (to mesh the part with all.dat name)
K : send all sam (to export the mesh into Samcef format)
K : send LOAD sam nam (to export groups into Samcef format)
K : send ANCHORAG sam nam (see previous remark)

```
K : send SYMETRY sam nam

10- Modifications
It's possible now to make some modifications :
a- open all.dat file with your favorite text editor (Vi for me)
b- open ANCHORAG.nam & the SYMETRIC.nam files and do the same as previously
c- concatenate under linux the files using the following schema :
        cat all.dat LOAD.nam > s1.m
        cat s1.m ANCHORAG.nam > s2.m
        cat s2.m SYMETRY.nam > part.dat
(all the sx.m files will be erased afterward)
d- open PART.dat file and go to the end => then add RETURN
e- the mesh file now works with Samcef


Another interesting way : add for each .nam file an input in your bank file:
input "part.dat"
input "LOAD.nam"
input "ANCHORAG.nam"
etc. ...

11 IMPORTANT REMARK

After, it's possible to modify the mesh into BACON (extrusions, etc. ...) ;
that's why the element hypothesis is not added at the end of the file ;
=> you must define the element definition AFTER the last mesh modification
(.HYP MINDLIN)

12- Comments

if you've any remark or any comment or any suggestion to improve this export
format, please send a mail to paul.carrico_at_free.fr
```

C Simple Examples

The following listings show simple geometry input-files. The pictures show this geometry together with their labels and the generated mesh. The models were made based on three points. Two points defined one axis of rotation and one was the basis of several "swep" and "merge" operations. In case of the sphere the surfaces on the pole had to be redefined using only three lines per surface.

C.1 Disc

```
PNT py       -0.00000        1.00000        0.00000
PNT p0       -0.00000       -0.00000        0.00000
```

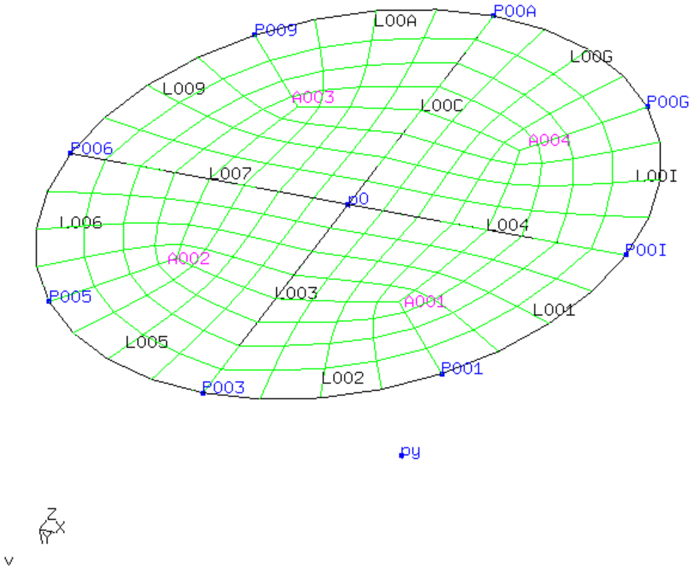


Figure 5: disc made of four 90 degree segments

```
PNT P001     0.70711      -0.00000      -0.70711
PNT P003    -0.00000      -0.00000      -1.00000
PNT P005    -0.70711      -0.00000      -0.70711
PNT P006    -1.00000      -0.00000       0.00000
PNT P009    -0.70711      -0.00000       0.70711
PNT P00A     0.00000      -0.00000       1.00000
PNT P00G     0.70711      -0.00000       0.70711
PNT P00I     1.00000      -0.00000      -0.00000
LINE L001 P00I P001 p0 4
LINE L002 P001 P003 p0 4
LINE L003 P003 p0 8
LINE L004 p0 P00I 8
LINE L005 P003 P005 p0 4
LINE L006 P005 P006 p0 4
LINE L007 P006 p0 8
LINE L009 P006 P009 p0 4
LINE L00A P009 P00A p0 4
```

```
LINE L00C P00A p0 8
LINE L00G P00A P00G p0 4
LINE L00I P00G P00I p0 4
GSUR A001 + BLEND  - L003 - L002 - L001 - L004
GSUR A002 + BLEND  - L007 - L006 - L005 + L003
GSUR A003 + BLEND  - L00C - L00A - L009 + L007
GSUR A004 + BLEND  + L004 - L00I - L00G + L00C
ELTY all QU4
```

C.2 Cylinder

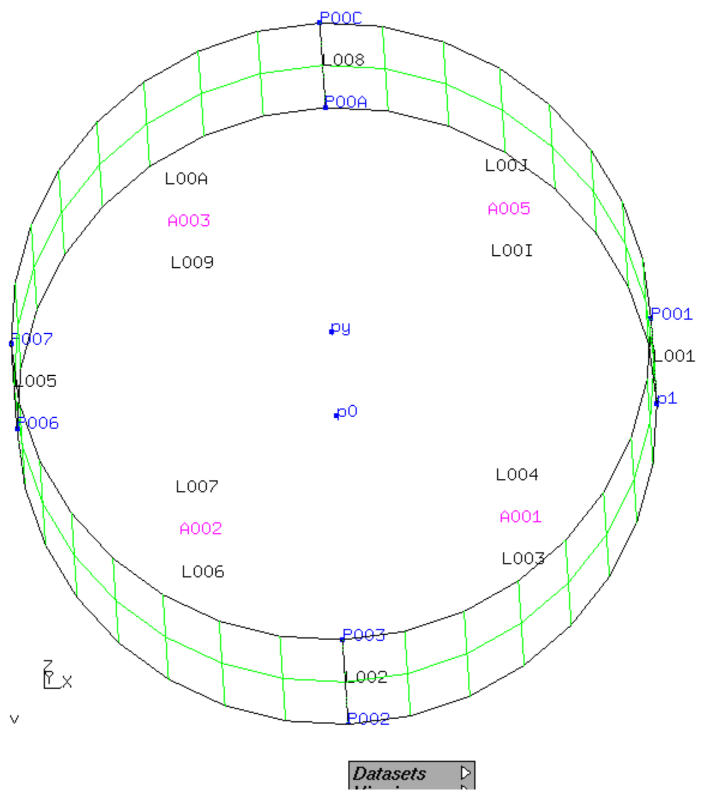


Figure 6: cylinder made of four 90 degree segments

```
PNT p0      -0.00000      -0.00000       0.00000
PNT py      -0.00000       1.00000       0.00000
PNT p1       1.00000      -0.00000       0.00000
PNT P001     1.00000       1.00000       0.00000
PNT P002    -0.00000      -0.00000      -1.00000
PNT P003    -0.00000       1.00000      -1.00000
```

```
PNT P006    -1.00000       -0.00000        0.00000
PNT P007    -1.00000        1.00000        0.00000
PNT P00A     0.00000       -0.00000        1.00000
PNT P00C     0.00000        1.00000        1.00000
LINE L001 p1 P001 2
LINE L002 P002 P003 2
LINE L003 p1 P002 p0 8
LINE L004 P001 P003 py 8
LINE L005 P006 P007 2
LINE L006 P002 P006 p0 8
LINE L007 P003 P007 py 8
LINE L008 P00A P00C 2
LINE L009 P006 P00A p0 8
LINE L00A P007 P00C py 8
LINE L00I P00A p1 p0 8
LINE L00J P00C P001 py 8
GSUR A001 + BLEND  - L001 + L003 + L002 - L004
GSUR A002 + BLEND  - L002 + L006 + L005 - L007
GSUR A003 + BLEND  - L005 + L009 + L008 - L00A
GSUR A005 + BLEND  - L008 + L00I + L001 - L00J
ELTY all QU4
```

C.3 Sphere

```
PNT py      -0.00000        1.00000       -0.00000
PNT p1       1.00000       -0.00000       -0.00000
PNT P001     0.70711       -0.00000       -0.70711
PNT P003    -0.00000       -0.00000       -1.00000
PNT P006     0.70711        0.50000       -0.50000
PNT P008    -0.00000        0.70711       -0.70711
PNT P00C     0.70711       -0.00000       -0.00000
PNT P00K     0.70711        0.70711       -0.00000
PNT P00L    -0.00000       -0.00000       -0.00000
PNT P00N    -0.00000        1.00000       -0.00000
LINE L001 p1 P001 P00L 8
LINE L002 P001 P003 P00L 8
LINE L003 p1 P006 P00L 8
LINE L004 P006 P008 P00L 8
LINE L006 P001 P006 P00C 8
LINE L008 P003 P008 P00L 8
LINE L00A p1 P00K P00L 8
LINE L00C P00K P00N P00L 8
LINE L00G P006 P00K P00C 8
LINE L00J P008 P00N P00L 8
GSUR A005 + BLEND  - L003 + L001 + L006
GSUR A002 + B\begin{appendix}
```

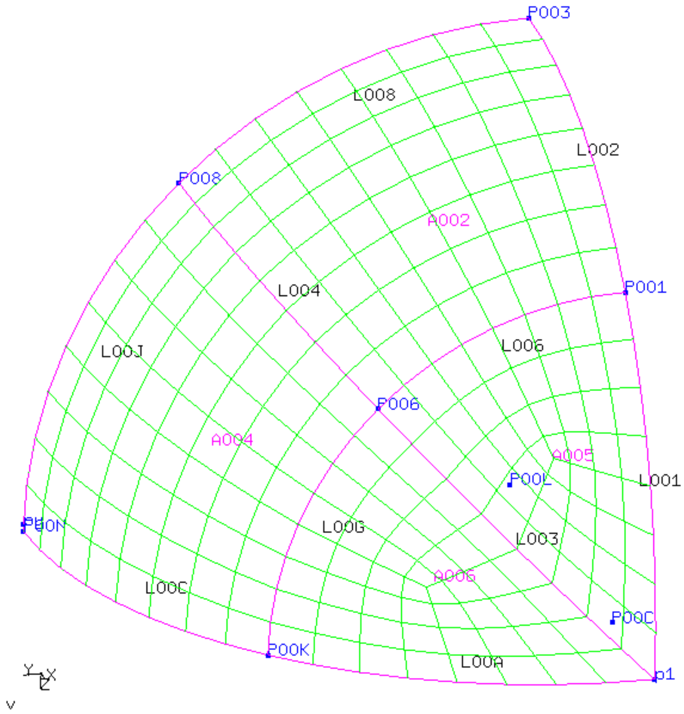


Figure 7: Segment of a Sphere

```
LEND  - L002 + L006 + L004 - L008
 GSUR A006 + BLEND  + L003 + L00G - L00A
 GSUR A004 + BLEND  - L004 + L00G + L00C - L00J
 ELTY all QU4
```

C.4 Sphere (Volume)

```
 PNT py        0.00000        1.00000        0.00000
 PNT p1        1.00000        0.00000        0.00000
 PNT P006      0.70711        0.50000       -0.50000
 PNT P008      0.00000        0.70711       -0.70711
 PNT P00C      0.70711        0.00000        0.00000
 PNT P00K      0.70711        0.70711        0.00000
 PNT P00L      0.00000        0.00000        0.00000
 PNT P00N      0.00000        1.00000        0.00000
 LINE L001 p1 P00L 8
 LINE L002 P00L P008 8
```

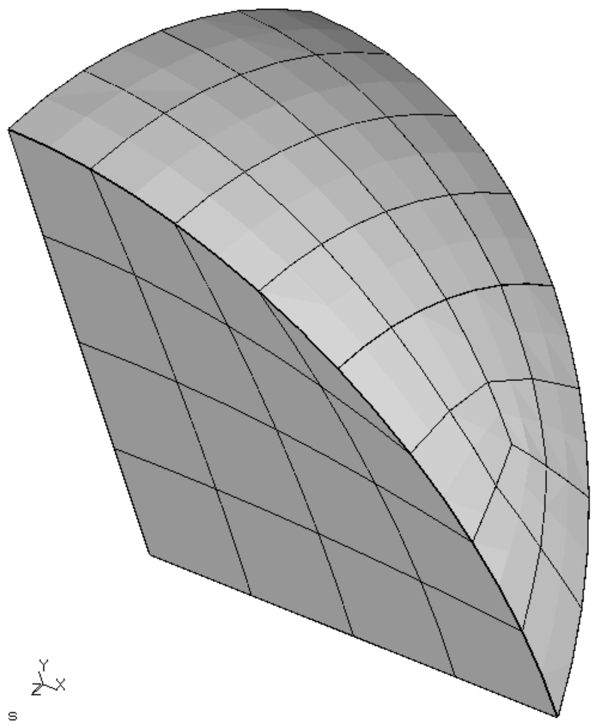


Figure 8: Segment of a Sphere (Volume)

```
LINE L003 p1 P006 P00L 8
LINE L004 P006 P008 P00L 8
LINE L005 P00L P00N 8
LINE L00A p1 P00K P00L 8
LINE L00C P00K P00N P00L 8
LINE L00G P006 P00K P00C 8
LINE L00J P008 P00N P00L 8
GSUR A001 + BLEND - L003 + L001 + L002 - L004
GSUR A002 + BLEND - L005 - L001 + L00A + L00C
GSUR A006 + BLEND + L003 + L00G - L00A
GSUR A004 + BLEND - L004 + L00G + L00C - L00J
GSUR A003 + BLEND + L002 + L00J - L005
GBOD B001 NORM + A006 - A003 - A004 + A002 + A001
ELTY all HE20
```

C.5 Airfoil for cfd codes

All surfaces must be oriented in the same way. The sets are used to define areas for the boundary conditions.

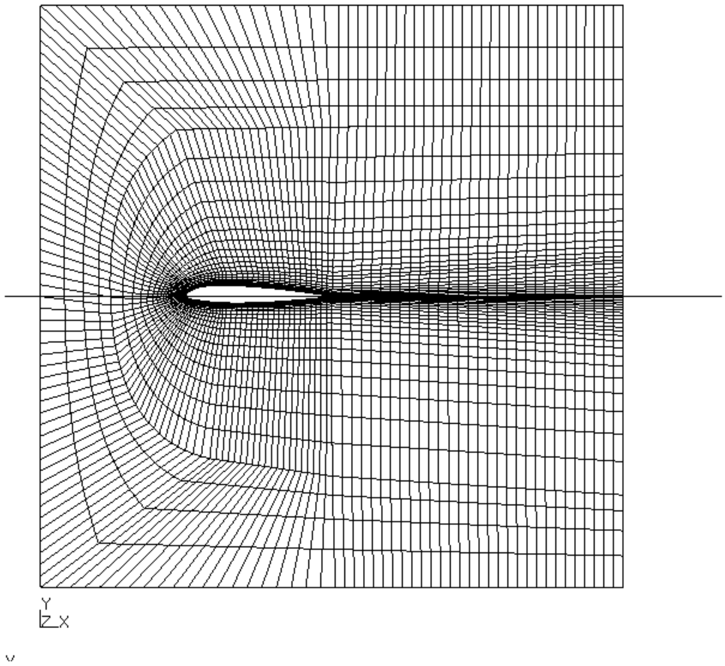


Figure 9: Airfoil for cfd codes

```
PNT P002    -0.24688        0.00667        0.00000
PNT P003    -0.24375        0.00903        0.00000
PNT P004    -0.23750        0.01228        0.00000
PNT P005    -0.23125        0.01450        0.00000
PNT P006    -0.22500        0.01608        0.00000
PNT P007    -0.21250        0.01798        0.00000
PNT P008    -0.20000        0.01875        0.00000
PNT P009    -0.18750        0.01900        0.00000
PNT P00A    -0.17500        0.01888        0.00000
PNT P00C    -0.15000        0.01785        0.00000
PNT P00E    -0.12500        0.01602        0.00000
PNT P00G    -0.10000        0.01368        0.00000
PNT P00I    -0.07500        0.01090        0.00000
```

PNT	P00J	-0.05000	0.00770	0.00000
PNT	P00K	-0.02500	0.00420	0.00000
PNT	P00L	-0.01250	0.00230	0.00000
PNT	P00O	-0.25000	0.00000	0.00000
PNT	P00P	-0.24688	-0.00308	0.00000
PNT	P00R	-0.24375	-0.00427	0.00000
PNT	P00S	-0.23750	-0.00565	0.00000
PNT	P00T	-0.23125	-0.00653	0.00000
PNT	P00V	-0.22500	-0.00730	0.00000
PNT	P00W	-0.21250	-0.00875	0.00000
PNT	P00X	-0.20000	-0.00993	0.00000
PNT	P00Z	-0.18750	-0.01070	0.00000
PNT	P010	-0.17500	-0.01115	0.00000
PNT	P011	-0.15000	-0.01120	0.00000
PNT	P012	-0.12500	-0.01043	0.00000
PNT	P013	-0.10000	-0.00917	0.00000
PNT	P014	-0.07500	-0.00750	0.00000
PNT	P015	-0.05000	-0.00540	0.00000
PNT	P016	-0.02500	-0.00308	0.00000
PNT	P017	-0.01250	-0.00175	0.00000
PNT	P019	0.00000	0.00000	0.00000
PNT	P1	-0.50000	-0.50000	0.00000
PNT	P2	0.50000	-0.50000	0.00000
PNT	p3	0.50000	0.50000	0.00000
PNT	p4	-0.50000	0.50000	0.00000
PNT	P01A	-0.18162	0.01898	0.00000
PNT	P01B	-0.18180	-0.01094	0.00000
PNT	P046	-0.27025	0.01256	0.00000
PNT	P059	-0.26599	0.28688	0.00000
PNT	P049	-0.25144	0.02439	0.00000
PNT	P05A	-0.35589	0.17566	0.00000
PNT	P04C	-0.22636	0.03241	0.00000
PNT	P04D	-0.20128	0.03643	0.00000
PNT	P05C	-0.38027	0.00958	0.00000
PNT	P04G	-0.12604	0.03833	0.00000
PNT	P04H	-0.07588	0.03616	0.00000
PNT	P04I	-0.02572	0.03231	0.00000
PNT	P05D	-0.31932	-0.21136	0.00000
PNT	P04L	-0.27652	-0.00154	0.00000
PNT	P04M	-0.27025	-0.00803	0.00000
PNT	P05F	-0.20962	-0.27840	0.00000
PNT	P04P	-0.25373	-0.01567	0.00000
PNT	P04R	-0.22676	-0.02143	0.00000
PNT	P04T	-0.20124	-0.02394	0.00000
PNT	P05G	0.01132	0.29145	0.00000
PNT	P04W	-0.12604	-0.02508	0.00000

```
PNT P04X    -0.07588       -0.02519        0.00000
PNT P04Z    -0.02572       -0.02355        0.00000
PNT P05H     0.00065       -0.30887        0.00000
PNT P052    -0.18074        0.03754        0.00000
PNT P054    -0.18133       -0.02465        0.00000
PNT P056     0.00118        0.02891        0.00000
PNT P058     0.00010       -0.02250        0.00000
PNT P05I     0.01244        0.50000        0.00000
PNT P05J     0.00610       -0.50000        0.00000
PNT P05L     0.50000       -0.34112        0.00000
PNT P05M     0.50000        0.29206        0.00000
PNT P05N     0.50000        0.05780        0.00000
PNT P05S     0.50000       -0.05314        0.00000
PNT P05V     0.50000       -0.00217        0.00000
PNT P00N    -0.23448        0.01345        0.00000
PNT P02M    -0.23471       -0.00608        0.00000
PNT P03B    -0.24164        0.02804        0.00000
PNT P03C    -0.23405       -0.02029        0.00000
PNT P03E    -0.24536        0.00794        0.00000
PNT P03P    -0.24464       -0.00400        0.00000
PNT pl1     -1.00000        0.00000        0.00000
PNT pl2      1.00000        0.00000        0.00000
SEQA S006      pnt  P01A P00A P00C P00E P00G P00I P00J P00K P00L P019
SEQA S007      pnt  P019 P017 P016 P015 P014 P013 P012 P011 P010 P01B
SEQA S00W      pnt  P03B P049 P046 P04L P04M P04P P03C
SEQA S00R      pnt  P052 P04G P04H P04I P056
SEQA S00S      pnt  P058 P04Z P04X P04W P054
SEQA S00T      pnt  P05F P05D P05C P05A P059
SEQA S001      pnt  P01A P009 P008 P007 P006 P005 P00N
SEQA S00E      pnt  P03E P002 P00O P00P P03P
SEQA S00L      pnt  P052 P04D P04C P03B
SEQA S00A      pnt  P02M P00T P00V P00W P00X P00Z P01B
SEQA S00X      pnt  P03C P04R P04T P054
SEQA S002      pnt  P00N P004 P003 P03E
SEQA S00P      pnt  P03P P00R P00S P02M
LINE L003 P01A P052 910
LINE L00C P01A P019 S006 120
LINE L00E P019 P01B S007 120
LINE L004 p4 P1 150
LINE L05F P2 P05L -204
LINE L05S P05L P05S -912
LINE L05C P1 P05J 120
LINE L006 P01B P054 910
LINE L007 P019 P056 910
LINE L008 P058 P019 -910
LINE L00G P05S P05V -210
```

```
LINE L00N P03B P03C S00W 130
LINE L03R P052 P056 S00R 120
LINE L00I P05V P05N 210
LINE L03S P058 P054 S00S 120
LINE L04V p4 P059 -204
LINE L04W P059 P052 -912
LINE L04X P054 P05F 912
LINE L04Z P05F P1 204
LINE L050 P05F P059 S00T 150
LINE L052 P059 P05G 120
LINE L054 P05F P05H 120
LINE L056 P05H P058 -912
LINE L058 P056 P05G 912
LINE L059 p3 P05I 130
LINE L05A P05I p4 120
LINE L05D P05J P2 130
LINE L05V P056 P05N 130
LINE L05I P05M p3 204
LINE L05L P05N P05M 912
LINE L05Z P058 P05S 130
LINE L06C P019 P05V 130
LINE L06F P05M P05G 130
LINE L06H P05G P05I 204
LINE L06I P05L P05H 130
LINE L06J P05H P05J 204
LINE L001 P01A P00N S001 -210
LINE L00A P03E P03P S00E 120
LINE L00K P052 P03B S00L 110
LINE L009 P02M P01B S00A 210
LINE L00O P03C P054 S00X 110
LINE L002 P00N P03E S002 -205
LINE L00L P03P P02M S00P 205
LINE cl pl1 pl2 120
LINE L005 P00O P019 120
LCMB C001  + L001 + L002 + L00A + L00L + L009
LCMB C004  + L00K + L00N + L00O
GSUR A001 + BLEND + L003 + C004 - L006 - C001
GSUR A002 + BLEND + L006 - L03S + L008 + L00E
GSUR A003 + BLEND + L00C + L007 - L03R - L003
GSUR A004 + BLEND - L008 + L05Z + L00G - L06C
GSUR A005 + BLEND + L06C + L00I - L05V - L007
GSUR A00I + BLEND - L04W - L050 - L04X - C004
GSUR A00J + BLEND - L04V + L004 - L04Z + L050
GSUR A00K + BLEND + L04Z + L05C - L06J - L054
GSUR A00L + BLEND + L06J + L05D + L05F + L06I
GSUR A00N + BLEND + L04V + L052 + L06H + L05A
```

```
GSUR A00O + BLEND - L06H - L06F + L05I + L059
GSUR A00P + BLEND + L04W + L03R + L058 - L052
GSUR A00R + BLEND + L04X + L054 + L056 + L03S
GSUR A00S + BLEND - L058 + L05V + L05L + L06F
GSUR A00T + BLEND - L056 - L06I + L05S - L05Z
SETA wall l L05C
SETA wall l L059
SETA wall l L05A
SETA wall l L05D
SETA profil l L00C
SETA profil l L00E
SETA profil l L001
SETA profil l L00A
SETA profil l L009
SETA profil l L002
SETA profil l L00L
SETA in l L004
SETA out l L05F
SETA out l L05S
SETA out l L00G
SETA out l L00I
SETA out l L05I
SETA out l L05L
```

References

[1] OpenGL-Like Rendering Toolkit, from Brian Paul, http://sourceforge.net

[2] OpenGL Utility Toolkit (GLUT), from Mark J. Kilgard

[3] CalculiX GraphiX (cgx), from Klaus H. Wittig, klaus.h.wittig@t-online.de

[4] NETGEN, unstructured mesher, http://www.hpfem.jku.at/netgen/

[5] Duns, a two- and three dimensional cfd code, http://sourceforge.net/

[6] ISAAC, a two- and three dimensional cfd code, http://sourceforge.net/

[7] OpenFOAM, a three dimensional cfd code, http://www.opencfd.co.uk

[8] Tochnog, a free fem-code, http://sourceforge.net/

[9] Tutorial for CalculiX, from Dr. Guido Dhondt, http://www.dhondt.de/tutorial.html

[10] ImageMagick 5.1.0 00/01/01 Q:8 cristy@mystic.es.dupont.com. Copyright: Copyright (C) 2000 ImageMagick Studio

[11] Netscape Communication Software, http://www.netscape.com

[12] S. A. Coons, 'Surfaces for computer-aided design of space forms'. Project MAC, MIT (1964). Revised to MAC-TR-41 (1967).

[13] mesh2d, unstructured 2D-mesher from B. Kaan Karamete, Ph.D, http://home.nycap.rr.com/kaan/

[14] Paul Dierckx, Curve and Surface Fitting with Splines, Oxford. University Press, 1993