The gOpenMol[1] Effort

Leif Laaksonen
Center for Scientific Computing, Espoo, FINLAND
Version 3.00 (23.09.2004)

URL: http://www.csc.fi/gopenmol/

Introduction

A few words about version 3.00

This version has been prepared by the same people as version 2.3. More of the kernel code has been rewritten with some extensions in the contour facility.

A few words about version 2.3

This version has been prepared by the same people as version 2.2. There are not so many new features but some parts of the source code has been modified or rewritten for the speed and formability.

A few words about version 2.2

This version took a long time to finish and I strongly doubt it would never have been finished without the help from Scott Anderson, Kevin Boyd, Eero Häkkinen and Minna Varis. New features have been added including the possibility to extend

[1] *Maintained by Leif Laaksonen (Leif.Laaksonen@csc.fi)*

Center for Scientific Computing
P.O. Box 405
FIN-02101 Espoo FINLAND
Phone: +358 9 4572378
Fax: +358 9 4572302

gOpenMol trough plugins (dll files on Windows and sharable objects on Linux/Unix) and old routines have been improved. Minna has designed the new gOpenMol web pages and created a splendid basis for further support extensions.

A few words about version 2.1

This version of gOpenMol is the product of a fruitful collaboration between the author and Scott Anderson, Jim Kress and Arlen Viste. There are some new features like the cartoon display type of protein secondary structures and the hydrogen bond display engine. The filter programs for GaussianXX and PcGamess have been modified to allow the display of density gradient vectors. gOpenMol has now also a general display engine to display vectors. The internal usage of Tcl variables has been improved and documented.

The author is grateful for all the suggestion, commens and critics over the years. This shows that there are users who care. Please continue your active role in this process.

A few words about version 2.0

This is a release with some new features and some of the engines "under the hood" have been rewritten. Scott Anderson at the University of Utah has been kind to extend his tutorial. It can be found at the end of this document and in the help/tutorials directory. The new features include arbitrary cut and clip planes defined from three atoms or coordinates. The atom coordinate input engine has been extended and should work now for molecular systems up to 100,000 atoms. It will of course work even for bigger systems but it takes some time to assign the atom types.

A few words about version 1.4

The current version took a long time to finish. A lot longer in fact than I would have imagined. There are not so many new things you can see on the screen but a lot of the code inside gOpenMol has changed. This time I have tried to debug the code a lot more than ever before. I have received invaluable help from Scott Anderson at the University of Utah. Without the help from Scott some of the new features would not have been ready for this release.

Some user code has also been included in the current version. Sigismondo Boschi at CINECA (High Performance Systems) in Italy provided the code for hardware based stereo (stereo in a window) on SGI equipment. Robert Best contributed with some byte swap code and finally pushed me to implement an on-the-fly byte swap for the binary files gOpenMol reads. Now it is possible to read binary trajectories and plt files from other hardware platforms without the formatting/unformatting in between.

A list of some of the new features in version 2.0 can be found in appendix 5.

How it all started

The gOpenMol program has a long future behind. The history goes back to the late 80's when I become curious to know if even I was capable of programming graphics for the at the time new Silicon Graphics workstations. Learning the C-language and the usage of the Graphics Library (GL) was an exciting trip to new areas in computer technology. I can still remember the first exciting time I was able to configure the workstation to send and accept mail messages and take a telnet connection to the computers at the Center for Scientific Computing (CSC). The graphics was not always what I believed to see but with the help from some professionals at CSC I was able to converge to something that at least distantly looked like molecules. Originally I was not very pleased with the commercial software available for the analysis of molecular dynamics trajectories and I firmly was in the believe that even I could program something more helpful.

That's about the background to the SCAREROW[2] program, which was introduced during the beginning of the 90's. In fact I still believe that there are some SCARECROW users out there. My colleague Aatto Laaksonen pointed out, already at an early stage, that if I would like to get more users I should change the name of the program to something more attractive. Anyway, I think the program has got some keen users over the years. That's at least the impression I got from all the messages I've received over the years. It was all fun and I haven't regretted the time I've spent in front of the workstations over the years.

Unfortunately SGI was not fast enough to react to the change that was happening in the workstation market. SGI created the very nice OpenGL graphics engine and other players ported it to their own hardware platforms. At the same time the Intel processors become powerful enough and the Intel/PC based graphics cards become fast enough for useful rendering. For most graphics applications in the molecular modelling field the Intel or Alpha processors with an OpenGL supported graphics card become well enough. At some stage I decide to give the old SCARECROW a push and port it to the Intel/Alpha Windows/PC platform. The SCARECROW program already had a primitive scripting engine but I also wanted something more powerful.

In the middle of the 90's prof. Geerd Diercksen asked me if I was interested in making a graphical interface for his OpenMol[3] program. Combining my interest to rewrite SCARECROW with Geerd's offer to write a graphical interface for OpenMol I decided to create gOpenMol. I first tried to use the X-Windows and Motif for the

[2] The SCARECROW program for the analysis and display of molecular dynamics trajectories. http://www.csc.fi/lul/chem/scarecrow/jmg_scare.html

[3] OpenMol set of programs at http://www.mpa-garching.mpg.de/~opmolsrv/MolPhysGroup/index.html.

graphical interface and Tcl for the scripting engine but found it far too difficult. At CSC we had already been using Tcl/Tk[4] for a long time so it was a rather smooth translation when I finally did the transition from Motif to Tcl/Tk. At this stage I had also realized that the Unix programming tools we had available was poor compared to the tools available on the Windows platform. I want to thank you Geerd for all the discussions we have had and your hospitality during my numerous visits to Garching, over the years.

It took me only a couple of days to get the non-graphical part of gOpenMol, originally written for Unix/X-Windows/Motif, to run on my Windows NT Intel PC. The entire gOpenMol has been rewritten using the Microsoft Visual Development environment. The code created run almost as such also on the various Unix platforms. This can be done using the extremely powerful Tcl/Tk scripting and windows engine. I can recommend it for anyone creating software for a variety of platforms.

The current version of gOpenMol has unfortunately very few OpenMol features. In fact there is not too much more in common with the OpenMol package than the name. However, gOpenMol has a lot of features that you can find useful in your work. I have from the beginning wanted to create a program that looks similar and has the same functionality on a variety of hardware platforms and operating systems. I think gOpenMol has been quite a success at lest in that sense. gOpenMol looks almost the same now on the Windows and Unix platforms. Some features like the copying of the screen to the paste buffer or making of movies are not supported on the Unix platforms. This must be handled on the Unix boxes with their own tools.

I maintain a HTML based help system for the **gOpenMol** development effort in Finland. This help system is under constant development so please check the gOpenMol HTML based help files regularly at:

URL: http://www.csc.fi/gopenmol/

I will try to maintain a page with updated information about the program itself and the graphical user interface (GUI). You can also read about bug fixes, updates and new versions.

All programming efforts should have a motto. I’ve tried to include the feelings I’ve had creating this piece of software over the years using the words by Neil Young.

Motto:

One of these days
I'm going to sit down and write a long letter
To all the good friends
I've known
And I'm going to try

[4] Tcl/Tk by Scriptics at http://dev.scriptics.com/

```
   To thank them all for the good times together
   Though so apart
   We've grown

      Neil Young, One of these days
```

I owe a lot to very many friends and colleagues over the many development years. It would be impossible to mention you all but I still have to mention some. Thank you very much Scott Anderson, Geerd Diercksen, Henrik Konschin, Aatto Laaksonen, Jeffry D. Madura and Juha Ruokolainen. Without your support, help and encouragement there would not have been any Scarecrow or gOpenMol.

If you use gOpenMol in your publications please use the following reference:

L. Laaksonen: A graphics program for the analysis and display of molecular dynamics trajectories. *J. Mol. Graphics 10 (1992) 33;*
D.L. Bergman, L. Laaksonen and A. Laaksonen: Visualization of Solvation Structures in Liquid Mixtures*. J. Mol. Graphics & Modelling 15 (1997) 301.*

What is gOpenMol?

gOpenMol is a program or toolbox for the display of molecules and their properties and the analysis of molecular structures and dynamics trajectories. gOpenMol, as a analysis and display toolbox, can be used:

- for the static display of molecular structures and properties calculated with external programs
- for the analysis and display of molecular dynamics trajectories
- for the display of molecular orbitals, electron densities and other properties
- together with electrostatic potentials from programs like the GaussianXX, GAMESS and Jaguar
- together with electrostatic potentials from the UHBD (University of Houston Brownian Dynamics), AutoDock and the GRID programs.

gOpenMol includes also a set of programs and utility functions.

The software:

gOpenMol is entirely written in C and the graphics is using OpenGL. The windowing and the scripting are implemented using Tcl/Tk. The Tcl/Tk scripts defining the graphics layout are plain text files and can thus very easily be modified. The graphical interface can heavily be changed while using gOpenMol. In fact it is possible to write an entirely new molecular graphics package inside gOpenMol using the available Tcl/Tk tools and functions. You can import, display and analyse several different input coordinate file formats and binary trajectory file formats. The program has a graphical user interface (GUI) and an internal line command interpreter based on the Tcl/Tk.

http://www.tcl.tk/

gOpenMol can be used for a wide range of analysis and display tasks like the display of isocontour surfaces and cross sections of grid data.

gOpenMol can be understood as two applications talking to each other through the command line interpreter:

1. the kernel part containing the C-coding and the graphics (OpenGL) part
2. and the Tcl/Tk based GUI.

These two parts talk to each other through the command line interpreter located in the kernel part. When an option is selected from the GUI the equivalent line command is generated and sent to the interpreter. This is a very handy and simple procedure. However, the opposite is not that easy. The kernel can of course talk with the GUI but it is more complicated and needs a bit more coding and I haven't been able to create a neat and convenient way to do that.

To achieve a flexible way to extend gOpenMol without any recompilation of the kernel part I have implemented some of the commands using Tcl/Tk scripting. This means that the kernel C-coding for a command is in fact calling a Tcl script that does the command and then returns back and gives the control again to the kernel. Now by just modifying the Tcl code part the command can be extended. For example some of the coordinate input routines call Tcl functions that reads in the atom symbols and coordinates and then gives the control back to the kernel modules to detect the atom connectivity matrix, which is computationally very extensive.

To learn more please have a look the Tcl scripts in the data directory.

Requirements to run gOpenMol

The gOpenMol program is currently supported on the following platforms:

- Windows 95/98/NT
- Linux both Intel and Alpha using MESA
- IBM AIX using OpenGL and MESA
- Compaq Alpha OSF/1 using MESA
- SGI using OpenGL or MESA

The main platforms are Windows and Intel Linux. The versions for the other platforms will be available with a small delay.

A rough guess is that you need at least 32 MB of memory and about 50 MB of disk space to run gOpenMol. No special graphics hardware is needed for the Windows and Unix MESA versions. Today the OpenGL supported Windows display cards are very powerful and produce very satisfying graphical possibilities. MESA is using X-Windows so gOpenMol can be used also over the network. The special OpenGL supported graphics cards on the Windows platform gives a very convenient rendering speed. To be able to render acceptable graphics quality it is recommended that you run your display in either 16 bit graphics mode (64 Kcolours) or in the true colour mode (16 Mcolours).

Today most of the PC graphics cards already support OpenGL and can display by default true colour (16 Mcolours).

Supported file formats:

gOpenMol is hopefully a multipurpose analysis program. Currently the program has all of the molecular dynamics analysis features of the old SCARECROW and a lot of new features. Currently gOpenMol supports a wide range of molecular coordinate input formats:

- ADF, Amsterdam Density Functional output log files. If file contains multiple coordinates the coordinates will be merged to a multi-coordinate set XMOL file to be displayed using the trajectory analysis tools.
- CHARMm/CHARMM.
- CML file format.
- Chem3D binary file (plugin must be enabled).
- DL_Poly CONFIG coordinate file format.
- A frame from a trajectory (please look at the supported trajectory formats).
- Gaussian formatted checkpoint file.
- Gromos96 coordinate file format.
- Gromacs coordinate file format.
- HyperChem.
- Insight (car files).
- Jaguar coordinates from a Jaguar output log file.
- Mol2.
- Mumod.
- OpenMol (center binary file).
- PDB (Brookhaven Protein Data Bank format).
- Tinker coordinate file.
- Spartan binary file (plugin must be enabled).
- University of Houston Brownian Dynamics (UHBD) close to PDB format.

- Xmol single and multi xyz coordinate set file.
- XYZ general xyz coordinate file.
- YASP.

gOpenMol supports several **binary** trajectory formats. Those formats marked with a "*" should be readable by gOpenMol independent of the hardware platform on which they have been created. This means that these trajectory files can be moved from one platform to an other without the need conversion. Supported file formats include:

- Amber*. Program included for the unformatting of formatted AMBER trajectories.
- Cerius2*
- CHARMm*/CHARMM*. Program included for the formatting/unformatting of CHARMm/CHARMM trajectories.
- Discover
- DL_Poly
- GROMOS* (Supporting the old GROMOS format)
- GROMACS* (trn, trr and xtc formats)
- HyperChem (I haven't tested with the latest version)
- MUMOD *
- XPLOR*
- YASP*

and the following **ascii** trajectory formats:

- Amber formatted trajectory format.
- DL_Poly trajectory format.
- GROMOS96 trajectory format.
- TINKER multi frame coordinate trajectory format.
- XMOL xyz multi-step data sets. The included Xvibs program enables the display of vibrational modes from the GAUSSIANXX and GAMESS programs.

The surfaces are displayed through plt plot files. gOpenMol should be able to read the plot files from all hardware platforms. There is now no need to format and unformat the files when moving them from one hardware platform to an other. Extensive display isocontour surfaces and cut planes for:

- Orbitals, densities, gradients, Laplacian ... from GAUSSIANXX set of programs.
- Orbitals, densities, gradients, Laplacian ... from PcGamess .
- Orbitals and densities from Jaguar.
- Orbitals and densities from TurboMole.
- Connolly type of surfaces using the ProbeSurf program.
- Electrostatic potentials.
- Display and analysis of iso-contour surfaces from the UHBD program.

- Display and analysis if iso-contour surfaces from the GRID program.
- Display of AutoDock electrostatic grid data files.

Utility programs

All these executable programs are located in the bin/ directory and the source code is in the utility/ directory.

ambera2b

Converts a formatted ascii AMBER trajectory into a unformatted one. gOpenMol can only handle unformatted AMBER trajectories.

autodock2plt

Converts a AutoDock map file to a plt file known by gOpenMol. The program creates both formatted and unformatted plt files.

charmmtrj

This program makes the formatted/binary tranformation for a CHARMM dynamics trajectory file.

contman

Performs a variety of operations on a contour (plt) file.

convert

Converts the output search results (FDAT) from Cambridge Structural Database into separate PDB files.

gamess2plt

Converts the ‘cube’ information in the PC Gamess (v. 6) PUNCH file into a binary format known by gOpenMol.

gcube2plt and **g94cub2pl**

Converts a cube file from the GaussianXX program into a binary format known by gOpenMol. Can also save the gradient and Laplacian data.

gridasc2plt

Converts a UHBD formatted PHI grid file into the plt format used by the gOpenMol progra,

gridbin2plt

Converts a UHBD unformatted PHI grid file into the plt format used by gOpenMol.

kont2plt

Convert GRID electrostatic grid files into the plt format used by gOpenMol. The program creates both formatted and unformatted plt files. The formatted plt

pltfile

This program can be used to format a binary plot file or to make a formatted file into binary. This program is helpful if you want to move the plot file from one computer hardware platform to a other platform. This is not needed anymore. gOpenMol is able to read plt files from all hardware platforms.

sybyl2amber

Converts a formatted ascii Sybyl trajectory into an unformatted AMBER one.

tmole2plt

Convert TURBOMOLE moloch grids into a format recognized by gOpenMol.

trajmerge

Merges two CHARMM trajectory files into one.

xplor2charmm

Converts an XPLOR trajectory into a CHARMM trajectory.

gOpenMol as a helper application for your Web browser

It is possible to use gOpenMol as a helper application to your favourite Unix/Linux/Windows NT/95 Web browser. Please look into your favourite Web browser setup on how to start the **rungOpenMol.bat** file in the bin/ directory.

Using gOpenMol you can transport complete 3D molecular models including solid/transparent surfaces and CPK, licorice representations of your molecular models over the network. If you use gOpenMol as a helper application you can put up gOpenMol (.gom) files on your Web server to distribute graphics. I will make an effort to make new network features into gOpenMol in the future. This interface will be based on Tcl/Tk.

Introduction to gOpenMol

UNIX:

We assume that gOpenMol is properly installed on your system and you know how to run the rungOpenMol script that sets various aliases and environment values. In case you are using MESA and you are using a 8-bit display I recommend you to set the following environment variable:

setenv MESA_RGB_VISUAL "PseudoColor 8"

This variable tells MESA about your display type and assigns a proper visual to it. Write **gopenmol** and gOpenMol starts and if your display supports OpenGL or you are using MESA you get two new windows on your screen. The first window contains a pull down menu and some buttons to be used as the input window for gOpenMol. The other one is the graphics window where the molecular structures and other graphics objects are displayed.

Before we look at the GUI in more detail it is worth describing some of the things you don't see but yet they are still there:

- When gOpenMol starts it reads a lot of information about atoms, which will be needed for the display or other purposes. This information is taken from the gopenmol/data directory.
- The user can modify a lot of the default parameters or variables at startup through two startup files:
    1. At startup gOpenMol reads first the system startup file (**gopenmolrc.tcl**) located in the gopenmol/data directory. This file contains information related rather to all users.
    2. The second file is **.gopenmolrc.tcl** from your login directory which can contain information specific for one user.

Both startup files can contain internal gOpenMol tcl-commands.

WIN32:

To start gOpenMol on the WIN32 platform run the **rungOpenMol.bat** file in the bin/ directory. This is easily done by making a shortcut from the file to your desktop.
Before we look at the GUI in more detail it is worth describing some of the things you don't see but still are there:

- When gOpenMol starts it reads a lot of information about atoms, which will be needed for the display or other purposes. This information is taken from the gopenmol/data directory.
- The user can modify a lot of the default parameters or variables at startup through a startup file:
    - At startup gOpenMol reads first the system/user startup file (**gopenmolrc.tcl**) located in the gopenmol/data directory. This file calls further tcl files to set up many features of gOpenMol.

How to begin your gOpenMol session

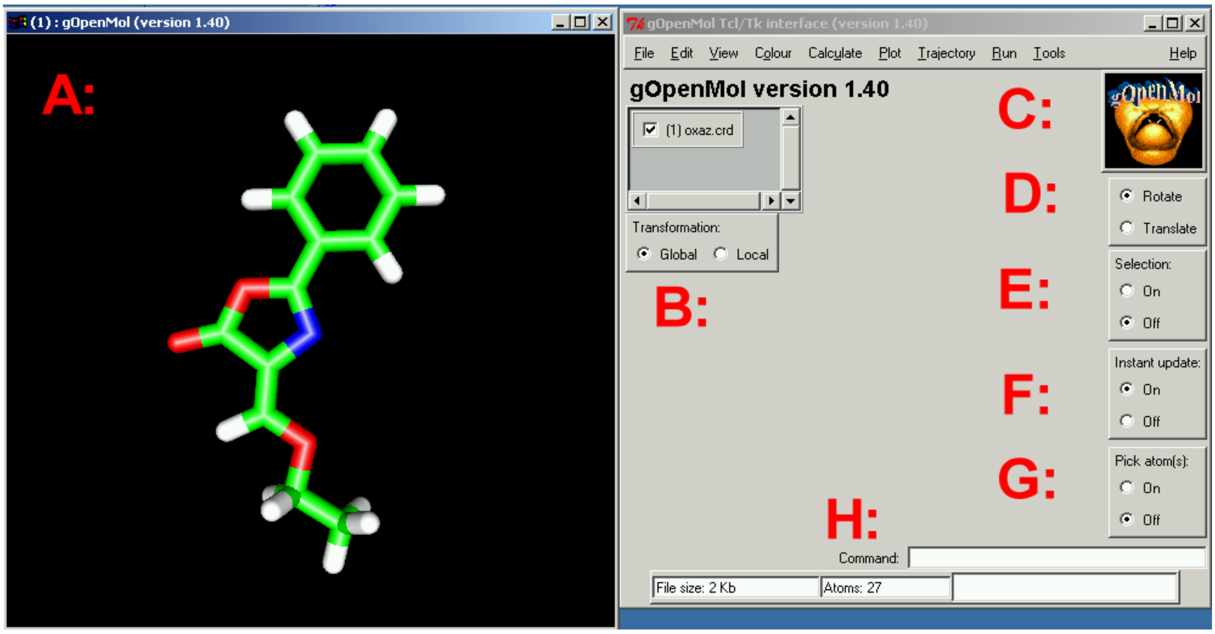
gOpenMol startup line command

You don't always need the graphics when you use gOpenMol. In fact in some cases it's an advantage not to use the graphics if you just want to use gOpenMol as a scripting engine. The command line options (you have to edit the rungOpenMol.bat or rungopenmol files in the bin/ directory) when you start gOpenMol are:

gopenmol -cInputCoord.Ext -h [-t -s] -ln -mModel.gom -pFile.Name Coord.File

-mModel.gom start GopenMol with the model file Model.gom
-cInputCoord.Ext start gOpenMol with coordinate file
-pfile.name start gOpenMol with parameter file 'File.Name'
-t start gOpenMol without any graphics
-ln do not write a log file (GOPENMOL.LOG)
-ly force gOpenMol to write the log file (GOPENMOL.LOG)
-s start gOpenMol with the hardware stereo (stereo in a window) option
-h print this help
any coordinate file with a predefined file extension (pdb, crd, …)

Start up windows:
The input window looks like (after reading in a structure):



☑ (1) oxaz.crd

Transformation: ● Global ○ Local

● Rotate ○ Translate

Selection: ○ On ● Off

Instant update: ● On ○ Off

Pick atom(s): ○ On ● Off

Command:

File size: 2 Kb

Atoms: 27

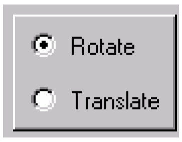
gOpenMol starts with two windows (3D graphics window and GUI window). Window A contains the 3D graphics and the other windows contain the GUI with the various buttons. The various regions are defined as:

- **A:** The graphics 3D window.
- **B:** In the GUI window there are various important regions. In the upper left corner there is a list describing the currently read input files and a checkbutton attached to the file name showing if the structure is active (checked) or inactive (not checked).
- **C:** Pressing the gOpenMol logo generates a display command for gOpenMol. Always when you want to display a scene press this button.
- **D:** Pressing the left mouse button and moving the mouse rotates the structure(s) in the 3D window if the rotation radiobutton is checked. If the translation radiobutton is checked the structure will be translated.
- **E:** It is possible to choose a full structure or fragments of a structure. If the selection radiobutton is checked the rotation or translation is applied to this selection.
- **F:** gOpenMol does not automatically redisplay a scene after a command has been applied. Checking the instant update to on will generate an automatic redisplay after all commands.
- **G:** Picking mode can be put on and off.
- **H:** The line command entry widget.

Please always remember to press the logo button to (re)display the view or click the "Instant update On"!

Rotate a structure in gOpenMol

All rotations and translations are applied according to a **fixed** coordinate system where the "imaginary" coordinate system is located so that the positive x-axis points from the left side of the screen to the right side. The positive y-axis points up from the bottom of the screen and the positive z-axis points from the screen to your nose.



Remembet to have the "Rotate" mode selected in the main widget!

The molecular object on the screen can be manipulated (rotated) in the following way:

(1) Pressing the **LEFT** mouse button and moving the mouse in the x-axis direction rotates the object around the **y-axis**.

Pressing the **LEFT** mouse button and moving the mouse in the y-axis direction rotates the object around the **x-axis**.

(2) Pressing the **RIGHT** mouse button and moving the mouse in the +y-axis direction zooms the object upwards and the -y-axis direction zooms the object downwards.

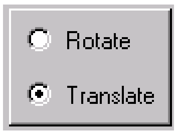
(3) Pressing the **MIDDLE** mouse button and moving the mouse in the x-axis direction rotates the object around the **z-axis**. If you only have a two button mouse you will not have this feature available.

To enable quick rotation, translation and picking it is possible to enable the translation mode by pressing the Shift key before one presses the left mouse. As long as the Shift key is and the left mouse button are pressed the structure will translate when the mouse is moved. Be careful to understand that if the Shift key will be released before the left mouse button the "Translate" state will be left on. In the other case "Translate" mode will return to "Rotate" mode.

By pressing the Ctrl key first and keeping it pressed it is possible to pick or mark atoms in the graphics screen by pressing the left mouse button. The selected atom(s) can then be dragged to an input widget. If one releases the Ctrl key before releasing the left mouse button gOpenMol will stay in the picking mode.

Pressing the Alt key and the left mouse button and moving the mouse in the x-axis direction rotates the molecule around the z-axis. This is the same as using the middle mouse on a three button mouse.

Translate a structure in gOpenMol



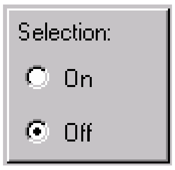
○ Rotate
● Translate

Remember to have the "Translate" mode selected in the main widget!

The molecular object on the screen can be manipulated (rotated) in the following way:

(1) Pressing the LEFT mouse button and moving the mouse in the x-axis direction or moving the mouse in the y-axis direction translates the object in that direction.

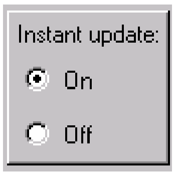
Define the set on which the action will apply



Selection:
○ On
● Off

If the selection is in the "Off" position all rotations and translations are applied on the main set (display). When the selection is in the "On" position all rotations and translations are applied on the atoms that have been selected with the "Select" facility. The select facility can be used for docking when one selects the atoms of a ligand and then translate and rotate the ligand to fit into an active site.

Control of the redisplay

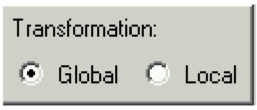

Instant update:

● On

○ Off

This controls the redisplay of the graphics. If the button is in the “On” position the display will automatically be redisplayed, when needed. If the “Off” position is selected the graphics has to be redisplayed either by pressing the gOpenMol logo or by executing the “display” line command. If one is working with a complicated picture that takes a long time to be redisplayed it is often useful not to have the redisplay on to save time.

Transformation options


Transformation:

● Global ○ Local

gOpenMol can operate on two different viewing or transformation modes:

- In the Global mode gOpenMol calculates **one** coordinate center around which the system(s) rotate. For example if you have a protein in one file and a ligand in an other file and you read both files into gOpenMol. When you read in both files into gOpenMol you would like to rotatet the whole system around a common point.
- In the local mode all the read structures rotate around their own coordinate center and translate relative to their own coordinate center. In this mode you can read in a protein and a ligan and move/rotate the ligan according to the protein structure.

By clicking on the “Global” and “Local” radio buttons you can change modes. By default changing the modes returns your structures to their starting positions. However, the user can define an option where the structures keep their current postions while changing modes. However, while this stage changes the coordinate values any surface displayed will have the structure orientation changed.

The coordinate system

gOpenMol uses a fixed global coordinate system where the centre of the coordinate system is in the middle of the graphics screen. The positive x-axis is running to the right and the negative x-axis to the left. The positive y-axis is running upwards and the negative y-axis downwards. The Positive z-axis is running out of the screen and the negative z-axis into the screen. The local coordinate system is rotating and translating with the molecule. The local coordinate system on any atom or an arbitrary position in the x, y, z space can always be displayed with the **plot axis** line command or using the “Plot ➔ Axis” widget.

The rotations and translations are always done in the global coordinate space.

The plotting of the local coordinate axis on an atom or set of atoms is helpful for example in docking.

Handling of input files

The gOpenMol program communicates with the environment in many ways. The simple way is through files. An input file can be a coordinate, trajectory, mesh data or an other file. The file can be located on the local or shared disk or it can be located on an other machines accessible through the Web or using http. When ever an input file name is give like **mycoord.pdb** gOpenMol expects the file is on a local or shared disk. If the name is given like **http://my.machine.net/mycoord.pdb** gOpenMol tries to download the file **mycoord.pdb** over the network.

Input coordinate files are handled in the following way:

1. File is read into gOpenMol
2. Atoms are assigned type and properties according to their type
3. The connectivity (bonds) are calculated
4. After the first 3 steps have been passed gOpenMol calls the tcl procedure **lulPostReadAtomCoordinates** in the **gopenmolrc.tcl** file in the **data** directory with two parameters:
    a. Type of atom coordinate set

i.	CHARMM_COORD	1
ii.	FREE_COORD	2
iii.	INSIGHT_COORD	3
iv.	AMBER_COORD	4
v.	YASP_COORD	5
vi.	MUMOD_COORD	6
vii.	GROMOS_COORD	7
viii.	MOPAC_COORD	8
ix.	PDB_COORD	9
x.	GAMESS_COORD	10
xi.	OPENMOL_COORD	11
xii.	HYPER_COORD	12
xiii.	XYZ_COORD	13
xiv.	SOCKET_COORD	14
xv.	DYNAFRAME_COORD	15
xvi.	GAUSSIAN_COORD	16
xvii.	MOL2_COORD	17
xviii.	XMOL_COORD	18
xix.	TXYZ_COORD	19
xx.	USER_COORD	20
xxi.	GROMOS96_COORD	21

xxii.	UHBD_COORD	22
xxiii.	PDBQ_COORD	23
xxiv.	ADF_COORD	24
xxv.	GROMACS_COORD	25
xxvi.	DL_POLY_COORD	26
xxvii.	JAGUAR_COORD	27

There might also be some new files supported not mentioned here.

b. Return state from the coordinate reader
   i. If the 3 steps have been successful the parameter is 0
   ii. If the 3 steps have been unsuccessful the parameter is 1

Currently the **lulPostReadAtomCoordinates** procedure looks the following:

```
#################################################################
# PROC
# execute this script always after having read in the coordinate
# file
#
# type: defines the type of the input coordinate file
#
```

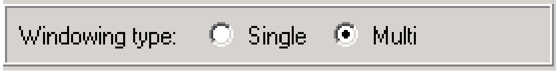
```
# there are two input states:
#
#   0: ok return from the coordinate stage
#  !0: error return from the coordinate stage
#
proc lulPostReadAtomCoordinates { Type State } {
  global gomTinkerInputConnectivity
# ERROR
  if {$State} {
# OK
  } else {
###################################################################
# this is for the TINKER coordinate files that has the connectivity
# included
#
# the array is set during the process when the coordinates are read in
# the connectivity can not be set at that stage because the connectivity
# will later be reset and recalculated
#
    if {[array exists gomTinkerInputConnectivity]} {

       catch {
        for {set i 1} {$i <= [array size gomTinkerInputConnectivity]} \
            {incr i} {

        edit bond create * * [lindex $gomTinkerInputConnectivity($i) 0] \
                         * * [lindex $gomTinkerInputConnectivity($i) 1];
            }
         }
            array unset gomTinkerInputConnectivity
     }
###################################################################
  }
}
```

This enables an automatic procedure to include commands after a file has been read into gOpenMol. Currently it supports the inclusion of the connectivity information from TINKER coordinate files.

Multiwindowing in gOpenMol

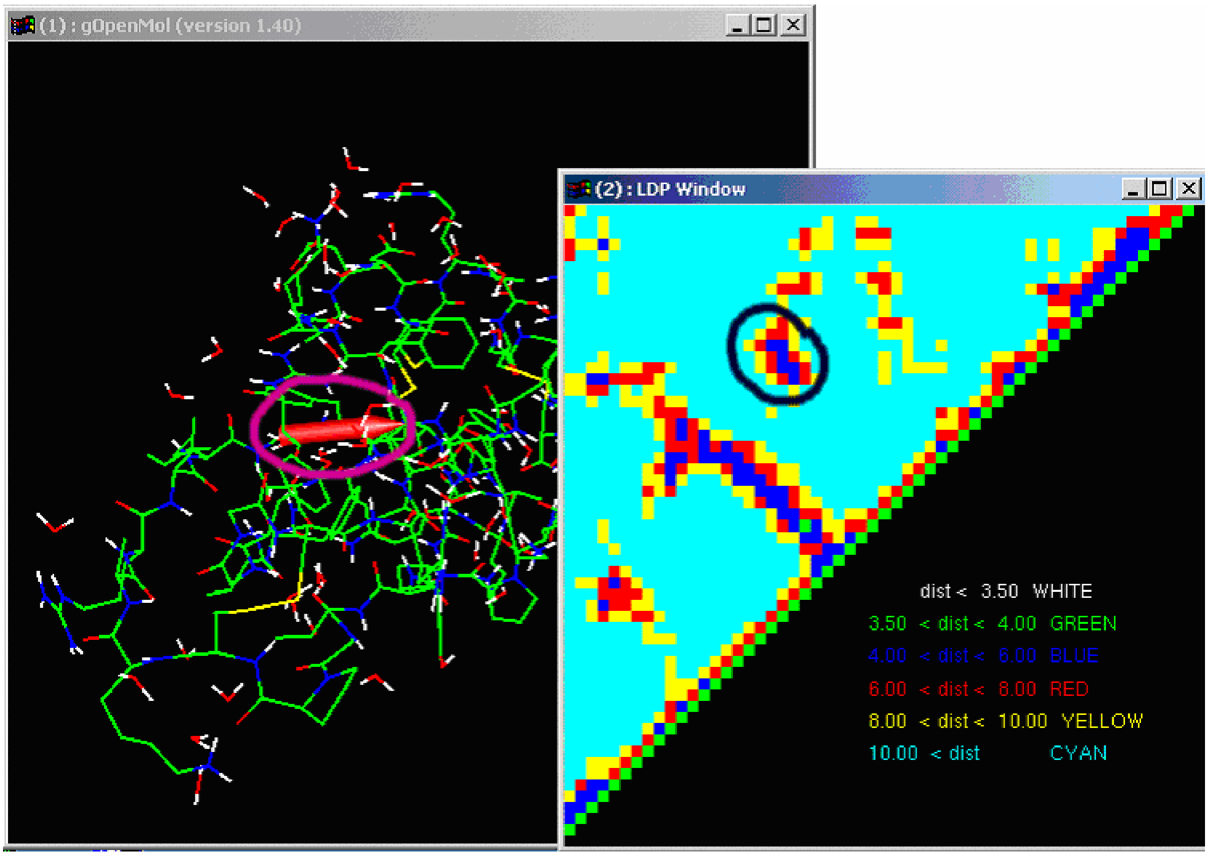
gOpenMol can in some cases use a display mode where two windows are open. The simple case is when a linear distance plot (LDP) and a structure window can be used simultaneously. A linear distance plot can be used for example to display and analyse the secondary structure of a protein or to display a distance profile for example in a docking. The multiwindow mode can be put on or of in the “Edit ➔ Display properties” widget. Please select from the “Windowing type” either “Single” or “Multi”.

- To see the effect import a protein file, for example the opt4pti.crd from the demo directory.
- Open the “Plot ➔ LDP” widget, turn the LDP plot on by clicking the “LDP plot: On”.and press the “Apply” button to select the CA atoms. Now you will see a distance plot showing the distances between the CA atoms in the structure.
- Open the “Edit ➔ Display properties” and click the Multi.



Windowing type: ○ Single ● Multi

- Now you will see two windows a window with a structure and the LDP window. If you now click with the Left mouse button in the LDP window you will see an arrow in the structure window showing the two atoms connected. The residue and atom numbers of the clicked atoms will also be shown in the startup (text) window. This helps you identify the atoms involved in the secondary structure.
- The mode can be changed again by going to the “Edit ➔ Display properties” widget and clicking “Single”.

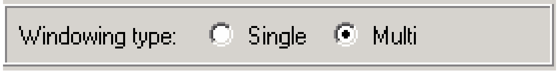


dist < 3.50 WHITE
3.50 < dist < 4.00 GREEN
4.00 < dist < 6.00 BLUE
6.00 < dist < 8.00 RED
8.00 < dist < 10.00 YELLOW
10.00 < dist CYAN

The multiwindow mode can also be used while looking at trajectories.

- Import the opt4pti.crd from the demo directory.
- Open the “Trajectory ➔ Main” widget and import the dyna4pti.dcd file.

- Open the “Plot ➔ LDP” widget, turn the LDP plot on by clicking the “LDP plot: On”.and press the “Apply” button to select the CA atoms. Now you will see a distance plot showing the distances between the CA atoms in the structure.
- Open the “Edit ➔ Display properties” and click the Multi.



Windowing type: ○ Single ● Multi

- You can now scan or loop through the structures/conformations by having one window showing the structure and the other window showing the LDP plot.

The main idea of the command interface

The menu driven interface is implemented using the line commands. The pulldown menu options generate a Tcl/Tk/gOpenMol command, which is sent to the gOpenMol parser.

The gOpenMol line command interface ***is extremely powerful and useful.***
The pulldown menus in the window (so far) are:

```
|File|Edit|View|Colour|Calculate|Plot|Trajectory|Run|Tools|Help|
```

(1) FILE menu has the options:

```
New,      Starts a new gOpenMol session and destroy all internal
          data structures and data.

Open,     Open a previously written model (structure gom file).

Close,    Close the current session and destroy all internal data.

Save,     Save the current structure and display in a model gom file.

Save as,  Save the current structure and display in a new gom file.

Import, Import various things into gOpenMol.
      •  Cluster data
      •  Coordinates in various formats
      •  Dictionary file defining the mapping between atom label to atom
         type
      •  Gaussian basis set for OpenMol
      •  Atom vector data
              ❑  Read a vector data file
                    -  CHARMM force vector file
                    -  Formatted flat file with vector data
      •  Read/edit a Tcl script and execute it
      •  Import partial atomic charges from various program output files
```

- ❑ GAUSSIAN charges from (extracted from a Gaussian output file)
    - Natural population analysis
    - Merz-Kollmann
    - Chelpg
- ❑ ICON8 atom charges (extracted from an ICON8 output file)
- ❑ MOPAC atom charges
- ❑ User defined charges

Export=, Export various things out from gOpenMol

- Cluster data
- Coordinates into some formats
- Correlation data
- Input=, Generate **raw** input to
  - ❑ GAMESS program
  - ❑ ICON8 program
  - ❑ MOPAC program
  - ❑ Probesurf program
  - ❑ User defined program
- Radial distribution function
- Root mean square deviation
- Save time series to disk

Reset=, Reset various things.

- Atom colour; Reset atom colours to default!
- Atom connectivity...; Reset/calculate atom connectivity!
- gOpenMol; Reset gOpenMol and free all reserved memory!
- View; Reset view to the startup view!

Hardcopy; Produce a hardcopy or screen dump in some bitmap formats.

EXIT; Halt program execution (Shut down program).

(2) EDIT menu has the options:

..Copy=, Copy various things to the clipboard.

- Bitmap; Copy current graphics window into clipboard
- Correlation; Copy correlation data into the clipboard
- MSD; Copy the Mean Square Displacement data into clipboard
- RDF; Copy the radial distribution function into clipboard
- Timeseries; Copy time series into clipboard.

Edit Cell, Edit the cell dimensions.

Edit Center, Define new rotation center.

Edit Connectivity, Edit (create/break) bonds.

Identify Atom, Identify an atom by picking.

Light Properties,	Edit light properties.
Material Properties,	Edit surface material properties.
Merge Structures,	Merge all the current structures into one.
Molecule,	Edit molecule (break and create bonds).
Rotate/Translate,	Control widget for rotation and translation.
Select,	Select atoms into the selction buffer.
Stere,	Stereo pair display mode.
Superimpose,	Superimpose two structures
Display attribs,	Change display attributes.
3D to 2D,	Make a 3D to 2D transformation of your system.

(3) VIEW menu has the options:

Atom colour,	Change atom colour.
Atom label(s),	Display atom label(s).
Atom tree	List and pick atoms from a tree like widget
Atom mask,	Change atom display mask.
Atom type,	Change atom type display.
Measure,	Measure molecular geometry.
Periodic table,	View and change atom parameters using the atom periodic table.
Stereo,	• Stereo pair display. • Hardware stere (SGI), stereo in a window.
Text output,	Display gOpenMol output text.

(4) COLOUR menu has the options:

Background=,	Change background colour in graphics window.
• Background,	Change background colour with a Tcl/Tk tool.
• Background,	Change background colour with a RGB tool.
• Atoms,	Change atom colour.

(5) CALCULATE menu has the options:

Average strcurure,	Calculate the average structure from a trajectory.
Cluster,	Calculate and plot cluster matrix from a

	Trajectory file.
Connectivity,	Calculate atom connectivity.
Correlation,	Calculate correlation series from time series.
Geometry,	Calculate molecular geometry.
Hydrogen bonds,	Calculate and display hydrogen bonds.
Mean sgr displ,	Calculate mean square displacement values from a trajectory file.
RDF,	Calculate a single or an average radial Distribution function from a frame or trajectory.
Superimpose,	Superimpose two structures.
Surface centroid,	Calculate the centroid for an isosurface.

(6) PLOT menu has the options:

Axis	Plot the local coordinate system.
Clip plane,	Define a clip plane for a surface.
Colour scale,	Plot a colour scale.
Contour,	Display contour surfaces.
Cutplane,	Plot a cutplane trough a contour surface.
LDP,	Plot a linear distance plot for selected atoms.
Plumber,	Plot a tube or ribbon trough selected atoms.
RMSD,	Plot root mean square deviation of atom movement.
Vector file,	Plot the set of vectors read from a file.

(7) TRAJECTORY menu has the options:

Fill=,	Fill time series vectors. * Distance array * Angle array * Torsion array
Delete=,	Delete time series vectors. * Distance array * Angle array * Torsion array
Main,	Main trajectory control widget.

Monitor=, Monitor molecular geometry.

- Distance(s)
- Angle(s)
- Torsion(s)

Time series, Manipulate time series widget.

Trace, Trace the movement of selected atoms.

Make video, Make animations.

(8) RUN menu has the options:

autodock2plt, Run the autodock2plt program to convert an AutoDock map file to the plt format (formatted and unformatted).

contman, Run the contour management program.

gamess2plt, Run the gamess2plt program to convert the The ‘cube’ information in the PUNCH file Into a format known by gOpenMol.

gCube2plt/g94cub2plt, Run the gCube2plt/g94Cub2pl to convert a GaussianXX cube file into a format known by gOpenMol.

jaguar2plt, Run the jaguar2plt program to convert Jaguar orbital data to a format known by gOpenMol.

Join Gamess IRC files Join Gamess IRC files into one.

kont2plt Run the kont2plt program to convert a Grid data file to the plt format.

pltfile, Run the Pltfile program to convert the gOpenMol plot file between binary and formatted.

probesurf, Run the Probesurf program to generate a Connolly type of surface.

tmole2plt Run the tmole2plt file to convert a TurboMole Grid file to the plt file format.

UHBD2plt Run the uhbd2plt file to convert a UHBD grid File to the plt file format.

xvibs Run the xvibs program to generate the Coordinate data to display vibrations.

(9) TOOLS menu has the options:

DLL/SO plugins, Extensions implemented as plugins
* Chem3D coordinate reader plugin
* Spartan coordinate reader plugin

```
                      * VRML export plugin
  AutoDock,           AutoDock display and analysis facility.
```

(10) HELP menu has the options:

```
  About,              Show the about text about gOpenMol.
  Demo,               Demo widget.
  Help,               Show this file.
  Peek version,       Check the gOpenMol version available.
```

(1) File menu

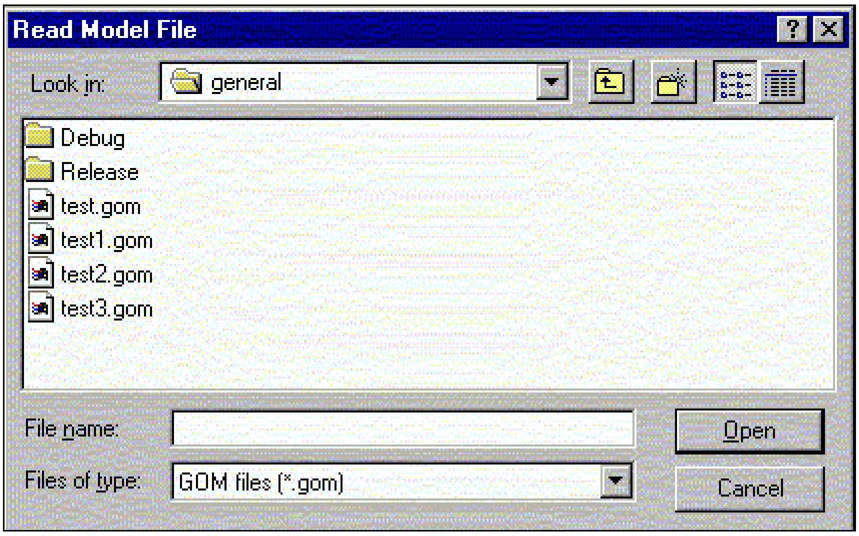
New

Start a new gOpenMol session and destroy all internal data structures and data.

Open a previously written model (structure gom file):

Read in a previously saved model (**gom**) file. The file is an ascii readable file containing information about the molecular display at the time when the file was saved. Currently it will contain information about the atoms, rotation matrix, contour data. Currently there are still things which will not be saved but eventually all the information will be saved to continue a session from the previously saved stage.

Read in the **gom** file using the following widget:

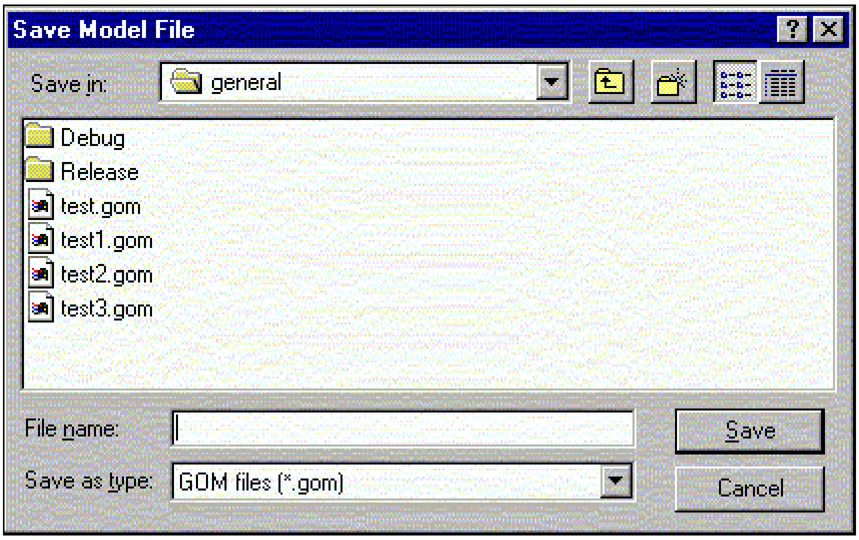


Line command option: **mopen command**

Save the current structure and display in a model gom file:

Save current display information into a **gom** file. This file can later be read back and the session can continue.

Save the session using the widget:



Save in: general

File name:

Save as type: GOM files (*.gom)

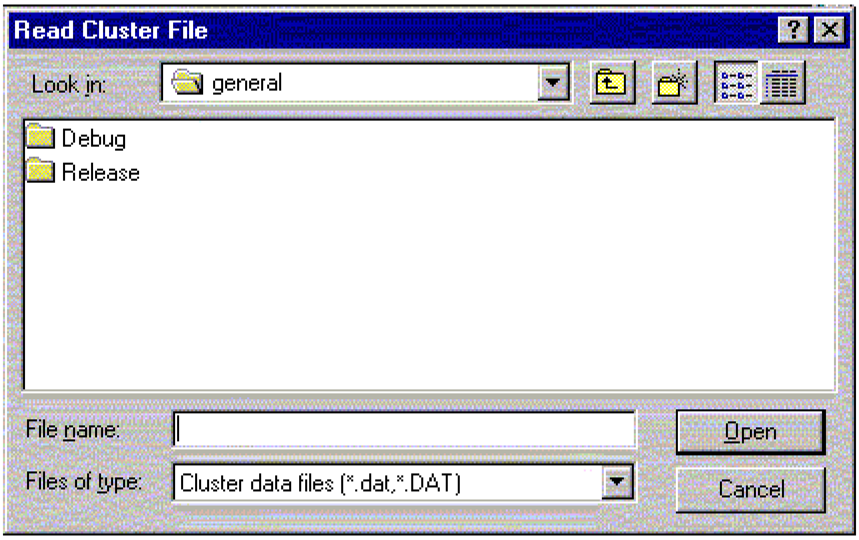
Line command option: see **msave command**

Import information into gOpenMol

Read cluster file widget

Read in precalculated cluster data for display (in future there will also be an analysis part).

The cluster data has been calculated using the **calculate cluster** menu option.



Line command option: see **import command**

Import Coordinates in various formats:

This is the main facility to import atom information (including coordinates) into gOpenMol. This is almost always the first step because gOpenMol has to know the molecular topology of the system before it can plot properties or analyse trajectory information.

The supported file types can bee seen on the right side with the radio buttons. There is also a button to indicate that the next structure is added (appended) to the current display. This new structure goes into e new data structure and it is not possible to make bonds between atoms in different data structures.

With the file browser it is possible to change the directory and filter the files with special file extensions. If the user has not done any reconfiguration the default file extensions are applied. The file extensions can be changed with the define command.

Supported file formats

gOpenMol support a variety of coordinate input files. Most are fully readable formatted files. The supported file formats are:

ADF

Amsterdam Density Functional program output log files.

AMBER

Amber coordinate files are a special dialect of PDB files. More information about the AMBER file formats can be found at http://www.amber.ucsf.edu/amber/formats.html.

CHARMm/CHARMM

CHARMM coordinate (crd) file. For the format please look into your QUANTA/CHARMM documentation or the source code.

DL_Poly

DL_Poly CONFIG coordinate file.

Frame

The coordinates are taken from a trajectory file.

GAMESS

The GAMESS PUNCH file (dat) containing the MOLPLT information produced with the MOLPLT=.TRUE. option.

Gaussian

The coordinates are imported from a GAUSSIANXX formatted checkpoint file.

GROMACS

GROMACS gro coordinate file.

GROMOS96

GROMOS96 coordinate data file.

HYPERCHEM

HyperChem input coordinate (hin) file.

INSIGHT

Insight coordinate (car) file.

Jaguar

Jaguar output log file.

Mol2

Mol2 coordinate file format.

MOPAC

Coordinates are imported from a MOPAC unformatted graphics output file.

MUMOD

Coordinate format used by the MUMOD program.

OPENMOL

Formatted coordinate input file to OpenMol.

PDB

The standard Brookhaven Protein Data Bank format.

UHBD

University of Houston Brownian Dynamics (UHBD) qcd coordinate reader.

TINKER

The Tinker coordinate input format.

USER

The user defined coordinate input reader.

XMOL

Xmol coordinate file format. The file can contain multiple entries but gOpenMol reads only the first entry.

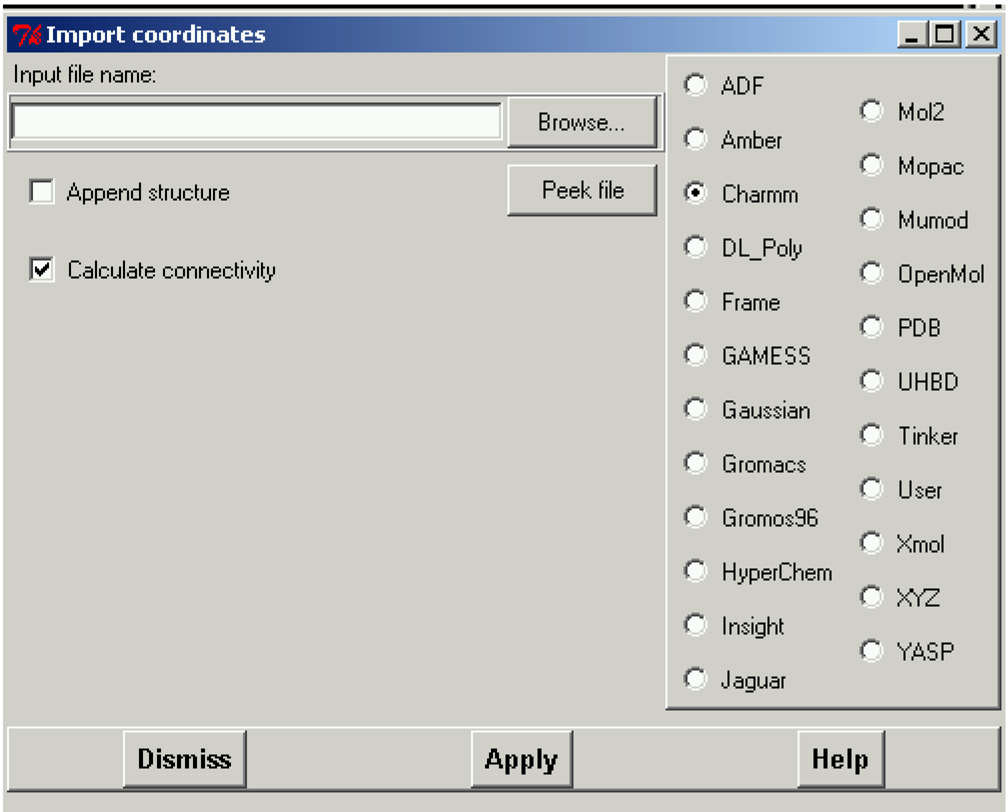
XYZ

Simple xyz coordinate input format.

YASP

Coordinate input file to the YASP program by Dr. Florian Müller-Plathe.

A file is selected to be processed by gOpenMol either by a rapid double click on the file name or a single click on the file name and a click at the **Open** button.
It is possible to "Peek" or look into a file before it is imported by pressing the **Peek** button when a file name is defined. It is also possible to choose or not to choose the calculation of the bonds for the molecular system.



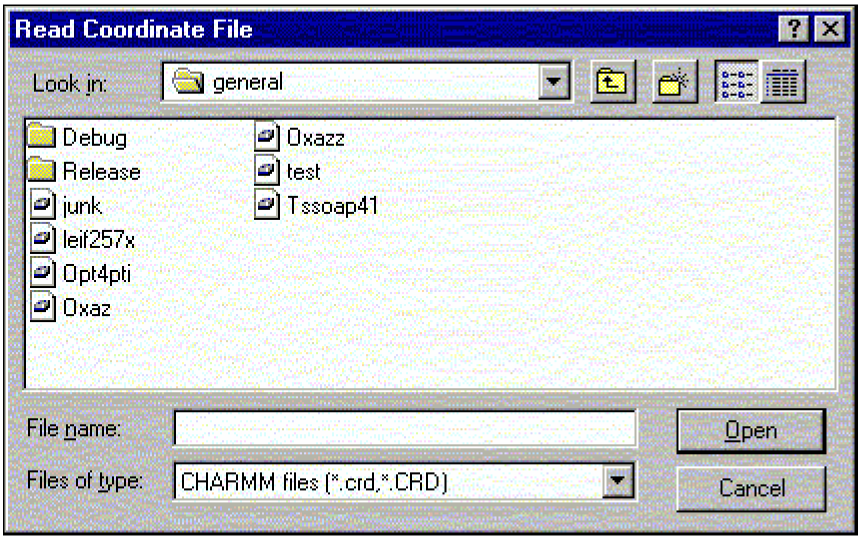
Input file name:

☐ Append structure

☑ Calculate connectivity

○ ADF
○ Amber
● Charmm
○ DL_Poly
○ Frame
○ GAMESS
○ Gaussian
○ Gromacs
○ Gromos96
○ HyperChem
○ Insight
○ Jaguar
○ Mol2
○ Mopac
○ Mumod
○ OpenMol
○ PDB
○ UHBD
○ Tinker
○ User
○ Xmol
○ XYZ
○ YASP

The file browser is started by clicking on the **Browse** button.



Look in: general

File name:

Files of type: CHARMM files (*.crd,*.CRD)

User defined input filter

The **USER** option for the input filters is a user defined Tcl script that reads a user defined input format.
The called Tcl procedure is: **lulReadUSERCoordinates *FileName Action***.

Where the *FileName* is the name of the file and the variable *Action* must be either **0** (=no append) or **1** (=append). Depending on if the structure will be appended to the structure list of if it will be only one.

```
#######################################################################
# PROC
# Input file name:                          FileName
# Read new structure or append to old list: Action (= 0 new  != 0 append)
proc lulReadUSERCoordinates {FileName Action} {

   error "User must supply the code!"
}
```

The user has to supply the code. As a suggestion the Tinker xyz input filter is implemented the same way and it looks like:

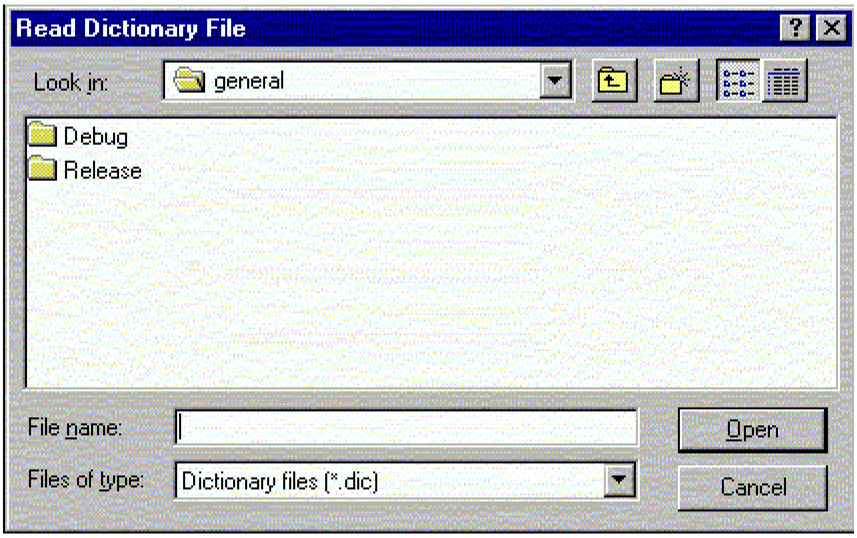
```
######################## export TINKER coordinates #####################
# PROC/Leif Laaksonen
# FileName
# Input file name:                          FileName
# Read new structure or append to old list: Action (= 0 new  != 0 append)
proc lulReadTinkerXYZCoordinates {FileName Action} {

# open file ...
   set File_p [open $FileName r]
   if {$File_p == ""} {
       catch {lulErrorDialog {ERROR: can't open TinkerXYZ file!}} errRet
       if {$errRet != ""} {
          puts "ERROR: can't open TinkerXYZ file!"
          }
       error "ERROR: can't open TinkerXYZ file!"
       return
   }

# do the job ...
   puts "Reading Tinker XYZ coordinate file '$FileName'..."

   gets $File_p InputData

   set NumAtoms [lindex $InputData 0]

   if {!$Action} {
    define structure $NumAtoms new
   } else {
    define structure $NumAtoms append
   }

   set StrucNum [show molstructures]

   for {set i 1} {$i <= $NumAtoms} {incr i} {

     gets $File_p InputData

     set AtomName [lindex $InputData 1]
     set Xc       [lindex $InputData 2]
     set Yc       [lindex $InputData 3]
     set Zc       [lindex $InputData 4]

# put the values into the structure ...
     define atom    label    $AtomName   $i $StrucNum
     define residue label    TINK        $i $StrucNum
     define segment label    TINK        $i $StrucNum
     define atom coordinates $Xc $Yc $Zc $i $StrucNum
     define atom charge      0.0         $i $StrucNum
     define atom resnumber   1           $i $StrucNum
}
# close file and return ...
     puts Done!
     close $File_p
}
```

The code is in the /data/gopenmolrc.tcl file.

Line command: **import command**

Import Dictionary file defining the mapping between atom label to atom type:

This facility imports a dictionary file. A dictionary file defines the mapping between a residue and atom name into the atom type space (and atom partial charge).



The dictionary file can look like the following:

```
*Atom dictionary file
ALA  N    32    -0.35
ALA  HN   1      0.25
ALA  CA   10     0.00
ALA  HA   3      0.10
ALA  CB   10     -0.30
```

It contains a residue name, an atom name and maps these into an atom type in the used force field (CHARMM in this case) and an atom partial charge. This is one of the possibilities to assign an atom type and an atom partial charge to the atoms.

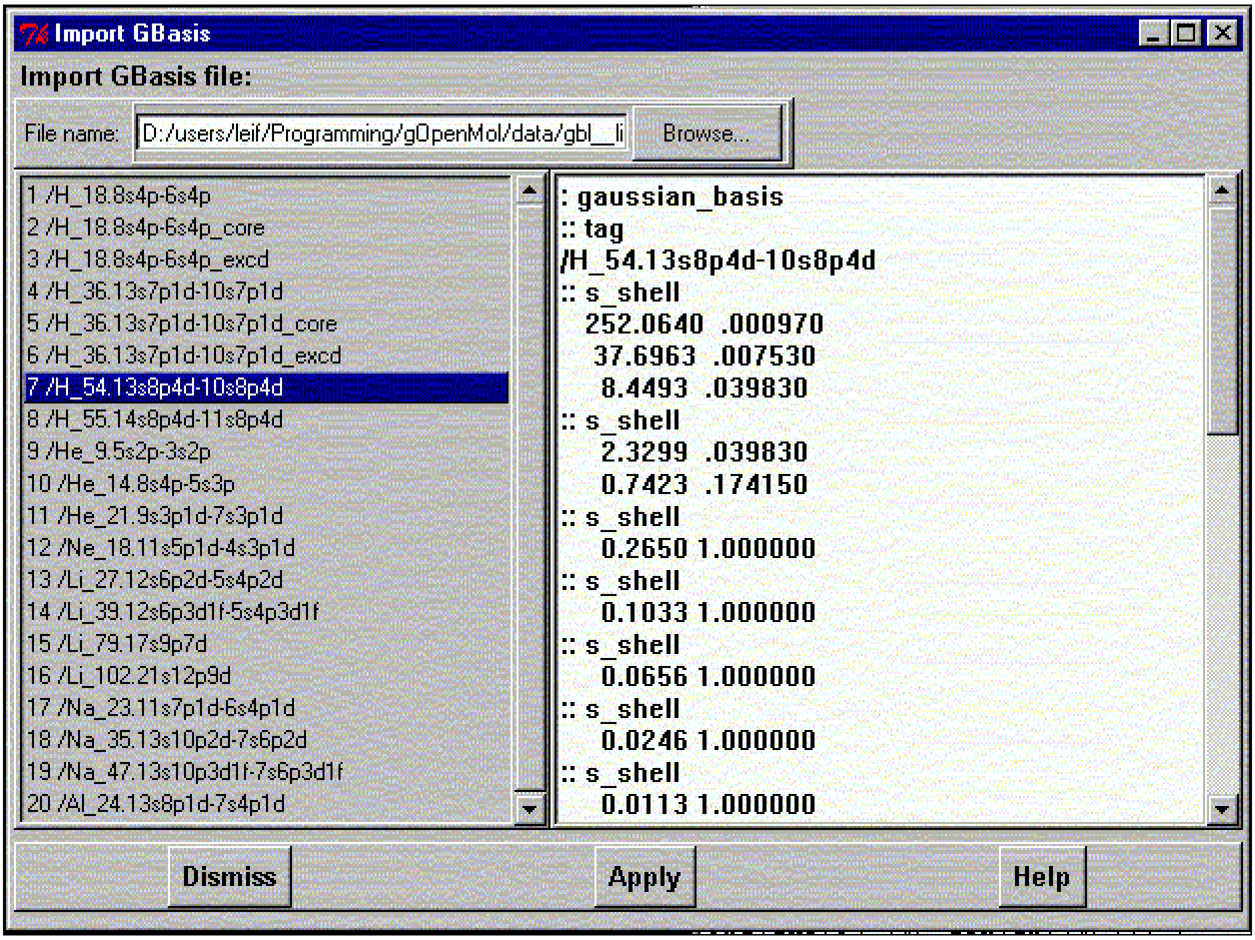
Line command: see **import command**

Import Gaussian basis set for OpenMol

This part will most likely not be helpful for anybody! Perhaps it could be a starting point for somebody who wants to write a basis set editor for a quantum chemistry package?

This is the OpenMol Gaussian Basis set widget browser/editor. As it is now implemented the file (database) can only be viewed not edited.
Use the file browser to select a data file. There is one sample file (gbl__line_input.data) in the gopenmol data directory.

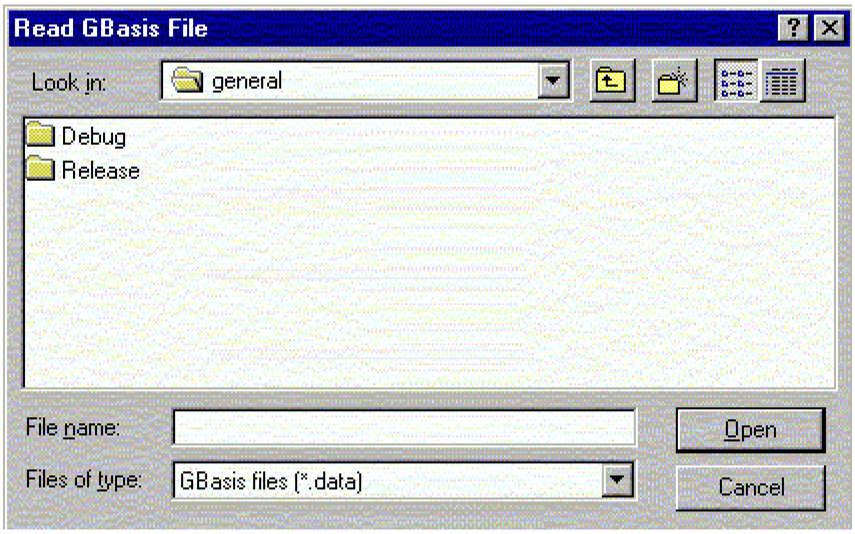
- Press the "Browse" button to select an input basis set database.
- Press the "Apply" button to inport the database.
- Press on any of the basis set tags, in the left widget, to show the basis set for that tag.



```
1 /H_18.8s4p-6s4p
2 /H_18.8s4p-6s4p_core
3 /H_18.8s4p-6s4p_excd
4 /H_36.13s7p1d-10s7p1d
5 /H_36.13s7p1d-10s7p1d_core
6 /H_36.13s7p1d-10s7p1d_excd
7 /H_54.13s8p4d-10s8p4d
8 /H_55.14s8p4d-11s8p4d
9 /He_9.5s2p-3s2p
10 /He_14.8s4p-5s3p
11 /He_21.9s3p1d-7s3p1d
12 /Ne_18.11s5p1d-4s3p1d
13 /Li_27.12s6p2d-5s4p2d
14 /Li_39.12s6p3d1f-5s4p3d1f
15 /Li_79.17s9p7d
16 /Li_102.21s12p9d
17 /Na_23.11s7p1d-6s4p1d
18 /Na_35.13s10p2d-7s6p2d
19 /Na_47.13s10p3d1f-7s6p3d1f
20 /Al_24.13s8p1d-7s4p1d
```

```
: gaussian_basis
:: tag
/H_54.13s8p4d-10s8p4d
:: s_shell
   252.0640  .000970
    37.6963  .007530
     8.4493  .039830
:: s_shell
     2.3299  .039830
     0.7423  .174150
:: s_shell
     0.2650 1.000000
:: s_shell
     0.1033 1.000000
:: s_shell
     0.0656 1.000000
:: s_shell
     0.0246 1.000000
:: s_shell
     0.0113 1.000000
```

File browser to select the basis set database.



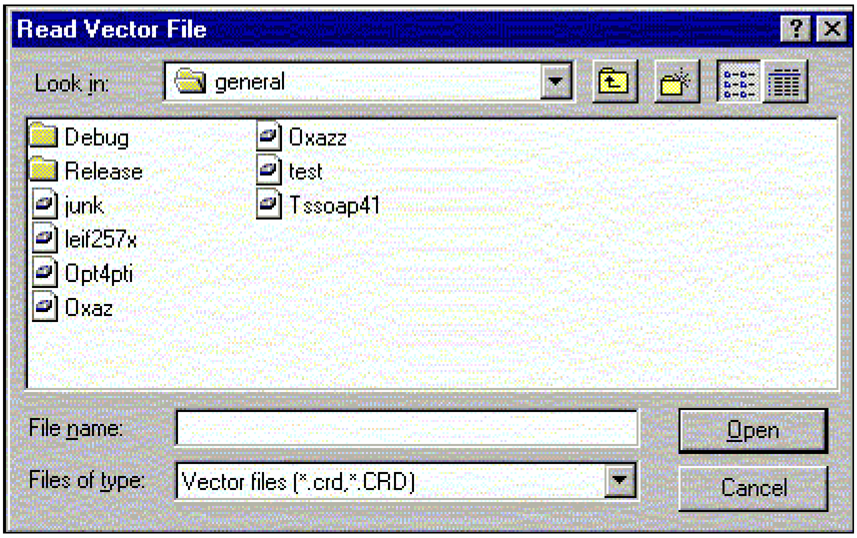
Most likely this will only be useful for a OpenMol user.

Line command: **import command**

Import Atom vector data

gOpenMol can read in two types of vector data:

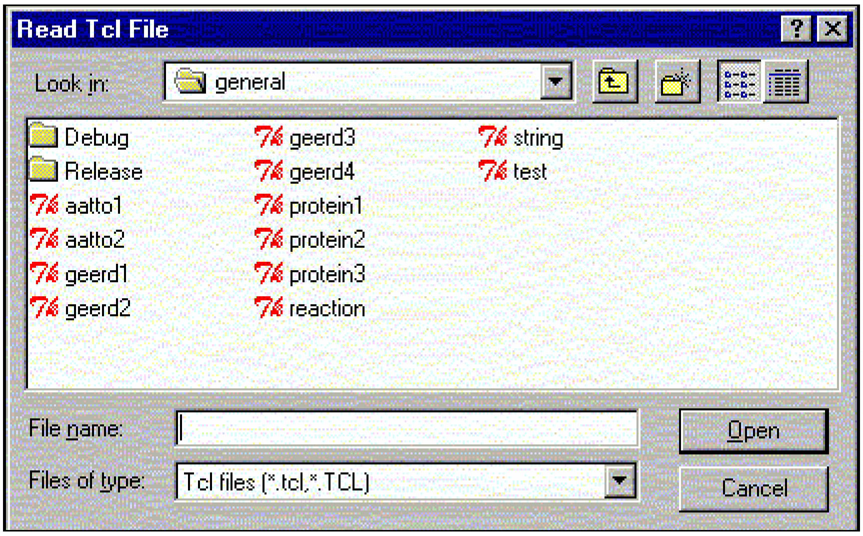
- Import information defining a vector property for the atoms in a molecule. The format is the same as in the **CRD** files apart from that the coordinate information defines now the vector direction in the x, y and z directions.
- Flat formatted text file containing the vector information. Look at the Appendix at the end of this document for the file format.



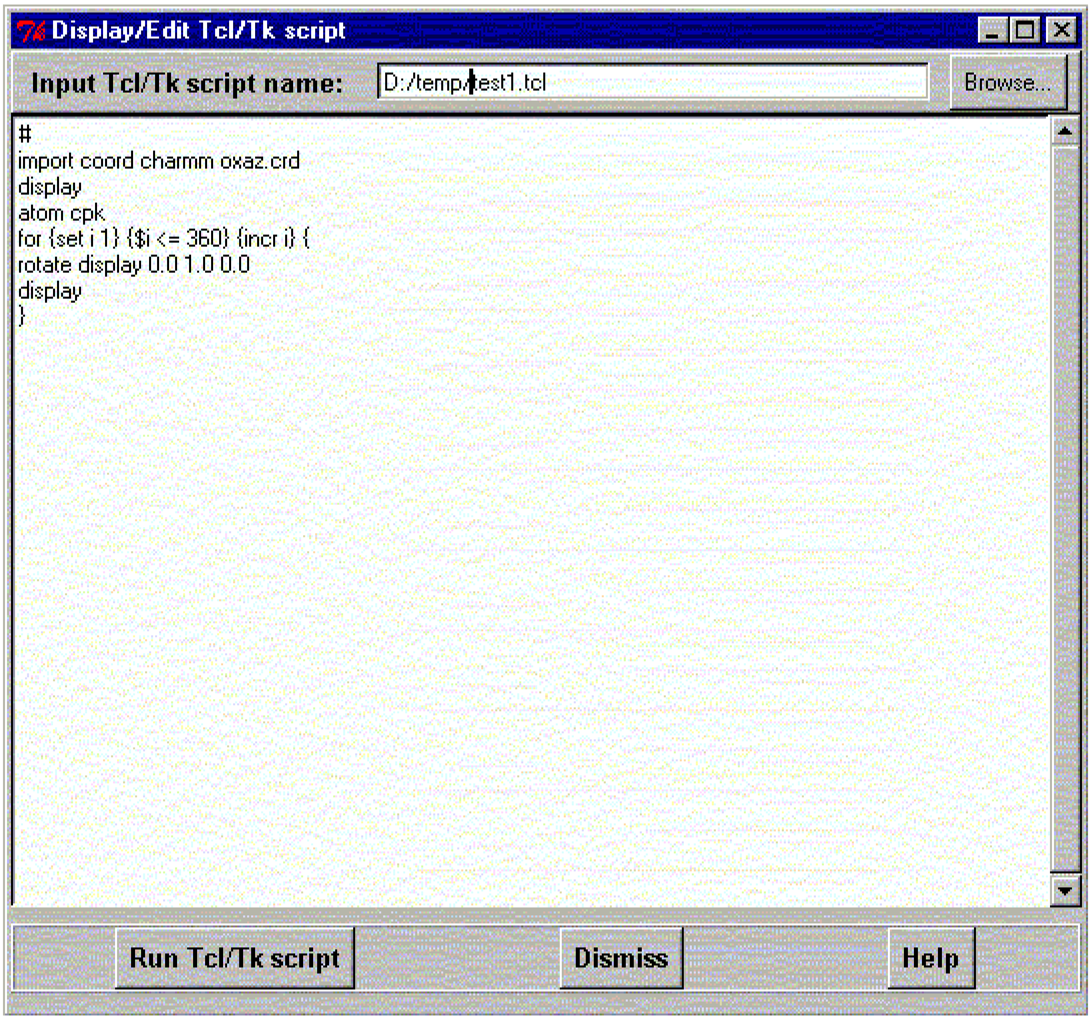
Line command: **import command**

Import, edit and execute a Tcl/Tk script

Import, edit a tcl script and execute it. It is the same as writing **source file_name.tcl** on the line command.



The Tcl code in the window can be edited.



Input Tcl/Tk script name: D:/temp/test1.tcl

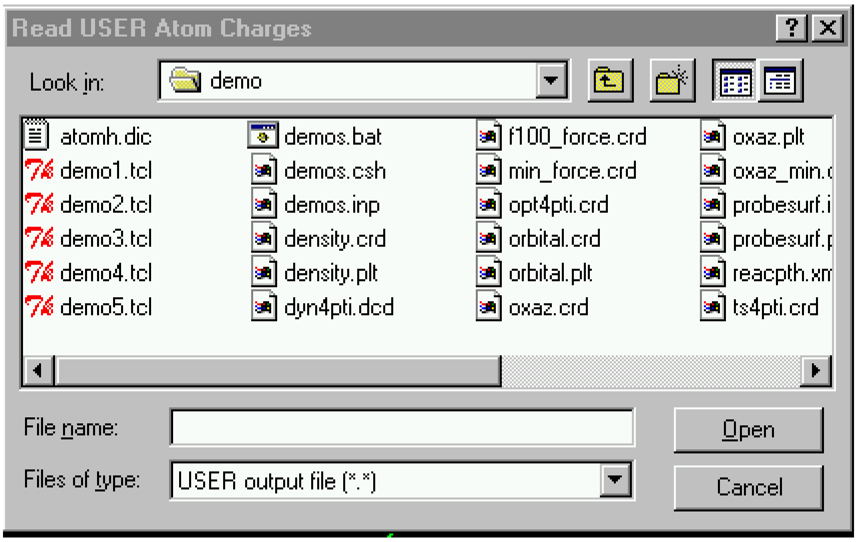
```
#
import coord charmm oxaz.crd
display
atom cpk
for {set i 1} {$i <= 360} {incr i} {
rotate display 0.0 1.0 0.0
display
}
```

Line command: see import command

Import partial atomic charges from various program output files

- GAUSSIAN charges from (extracted from a Gaussian output file)
  - ❑ Natural population analysis
  - ❑ Merz-Kollmann
  - ❑ Chelpg
- ICON8 atom charges (extracted from an ICON8 output file)
- MOPAC

gOpenMol will prompt for a file containing the atom charges.



Read USER Atom Charges

Look in: demo

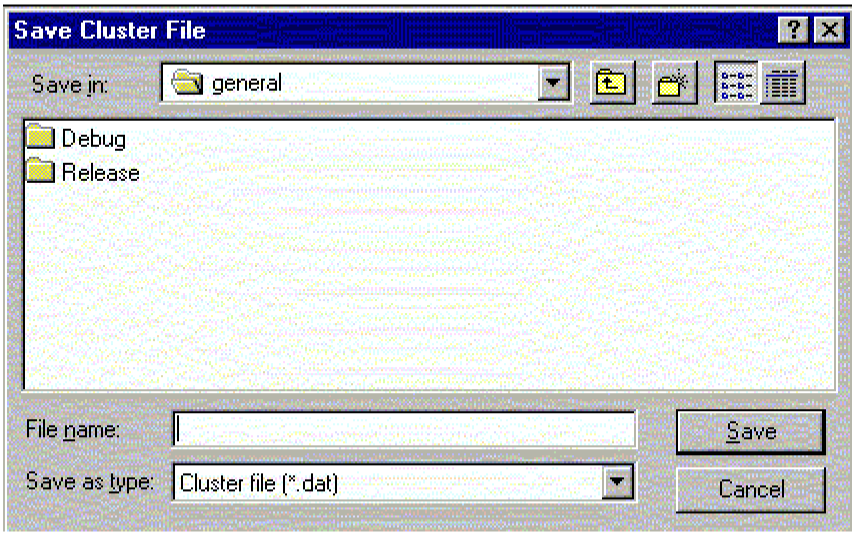
Files of type: USER output file (*.*)

The “File of type:” field will be different depending on from where the charge information is coming.

Export information out from gOpenMol

Write cluster widget

A previously calculated cluster matrix can be exported from gOpenMol to be used by other programs or to be read back to gOpenMo at a later time.



Save in: general

File name:

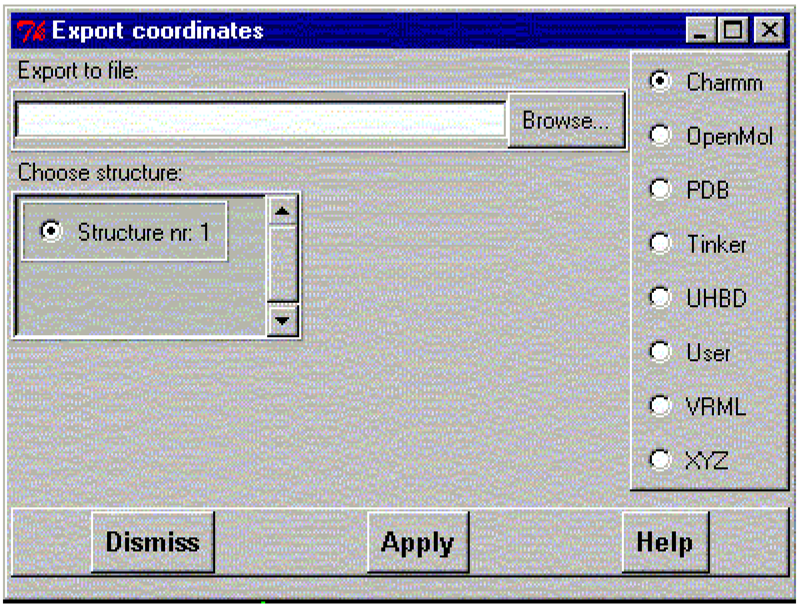
Save as type: Cluster file (*.dat)

Export coordinates widget

Export atom coordinate information in some formats. Choose first the file format and select then a file name or click at the "Browse" button to get a file selector.

If several molecular systems are read into gOpenMol select which structure will be exported.

The VRML file export facility is very crude. Currently it only supports line drawing, CPK and the licorice display styles.



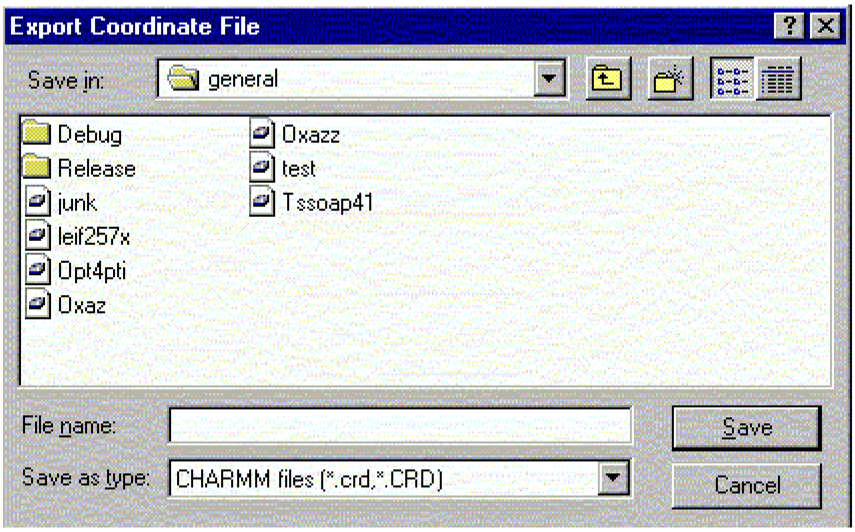
Export to file:

Choose structure:

● Structure nr: 1

● Charmm
○ OpenMol
○ PDB
○ Tinker
○ UHBD
○ User
○ VRML
○ XYZ

Write a file name or click at one of the files in the window.


Export Coordinate File

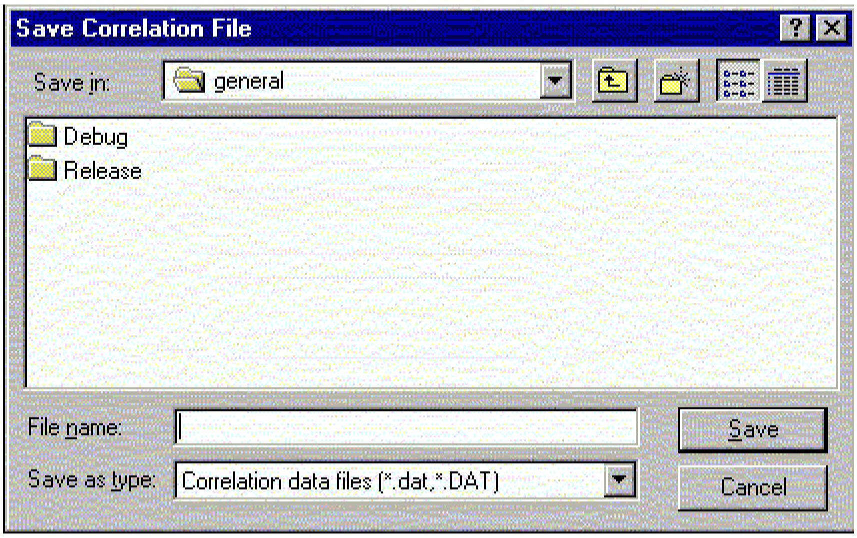
Save in: general

Save as type: CHARMM files (*.crd,*.CRD)

Line command: see **export command**

Write correlation widget

Write to disk correlation data calculated during the time analysis in trajectory analysis for molecular dynamics trajectory files.


Save Correlation File

Save in: general

Save as type: Correlation data files (*.dat,*.DAT)

Line command: see **export command**

Generate raw input to

- GAMESS program
- ICON8 program
- MOPAC program
- Probesurf program

- Own USER program

Export input for external porgrams

Using this widget it is possible to export (mostly very raw) input files for the following programs:

- **GAMESS** (raw input)
- **ICON8** Extended Huckel program (quite complete)
- **MOPAC** (raw input)
- **Probesurf**, a Connolly type of program (complete)
- **User defined**. To use this part one needs to program the output rules (Tcl/Tk coding) into the **lulMakeUSERinput** procedure in the gopenmolrc.tcl script in the data directory

ProbeSurf program input

ProbeSurf program is the program to generate Connolly type of surfaces.

The probe surface program PROBESURF generates a mesh with grid values from 0 to 100. The value 100 is at the vdW value and value 0 is at the vdW + max. probe diameter value.

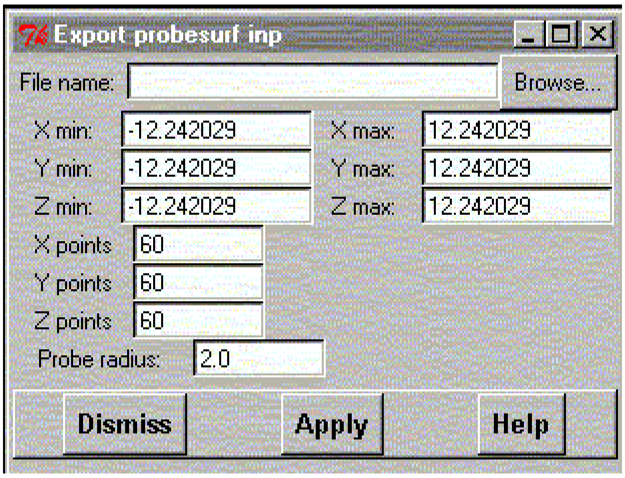
There are two methods available to calculate the values between 0.0 and 100.0.

- Direct distance (r) where the value is direct proportional to distance r and
- Squared distance (r**2) where the value is proportional to the square r**2 of the distance.

It is possible to generate the surface for different probe values. The method is based on the article by R. Voorintholt et al. (Voorintholt R., Koster M.T. , Vegter G. , Vriend G. and Hol W.G.J., "A very fast program for visualizing protein surfaces, channels and cavities", J. Mol. Graphics 7 (1989) 243-245)

Currently only the **Probesurf** program has some extra options available. It is possible to define the max and min x-, y- and z- values to define the place of the box for which the grid values are calculated. It is also possible to define the number of grid points in the x-, y- and z-directions. The max probe radius defines the largest probe radius that can be displayed using the current grid values.

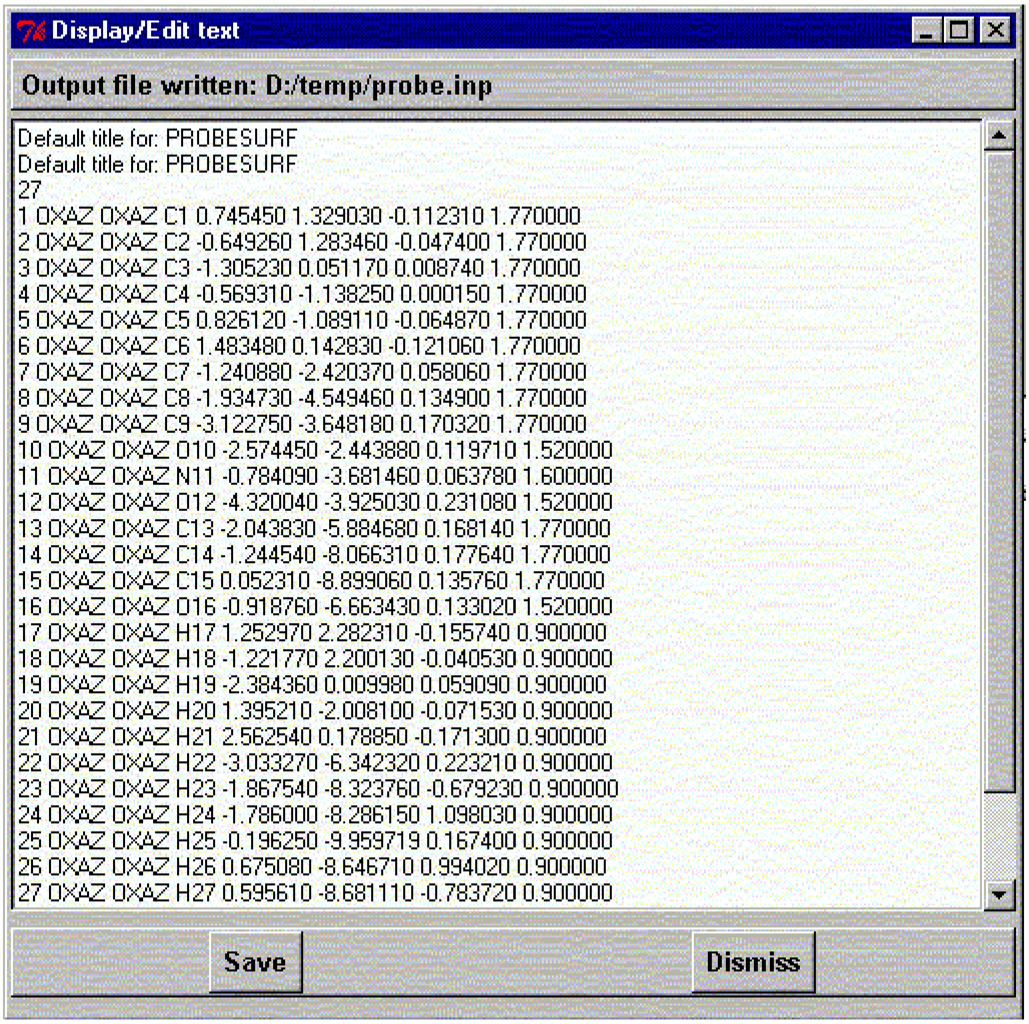
Choose the output option and file name (or click the **Browse** button to browse).



File name:

X min: -12.242029 X max: 12.242029
Y min: -12.242029 Y max: 12.242029
Z min: -12.242029 Z max: 12.242029
X points 60
Y points 60
Z points 60
Probe radius: 2.0

After pressing the **Apply** button the file is written to disk and a widget containg the file input is shown. It is possible to edit the data in the widget and save the new content to disk by clicking the **Save** button. The file will always have the same name as the original file had.



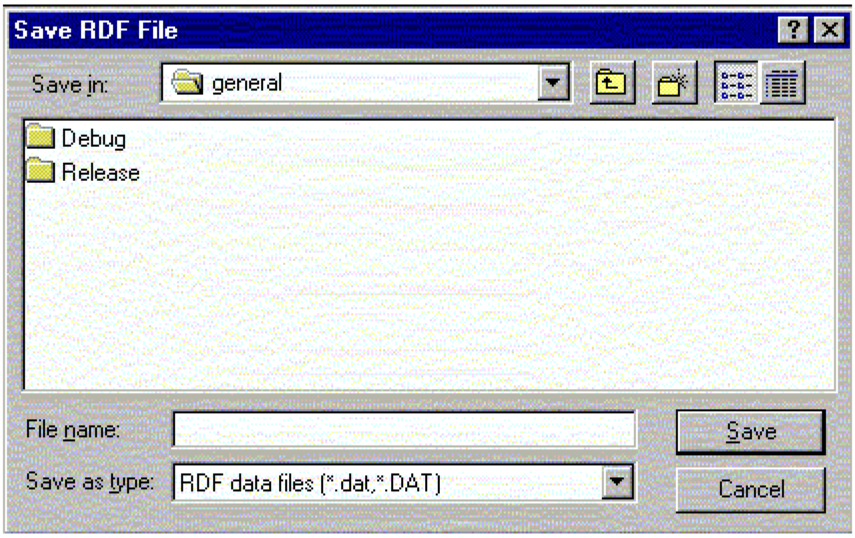
Output file written: D:/temp/probe.inp

```
Default title for: PROBESURF
Default title for: PROBESURF
27
1 OXAZ OXAZ C1 0.745450 1.329030 -0.112310 1.770000
2 OXAZ OXAZ C2 -0.649260 1.283460 -0.047400 1.770000
3 OXAZ OXAZ C3 -1.305230 0.051170 0.008740 1.770000
4 OXAZ OXAZ C4 -0.569310 -1.138250 0.000150 1.770000
5 OXAZ OXAZ C5 0.826120 -1.089110 -0.064870 1.770000
6 OXAZ OXAZ C6 1.483480 0.142830 -0.121060 1.770000
7 OXAZ OXAZ C7 -1.240880 -2.420370 0.058060 1.770000
8 OXAZ OXAZ C8 -1.934730 -4.549460 0.134900 1.770000
9 OXAZ OXAZ C9 -3.122750 -3.648180 0.170320 1.770000
10 OXAZ OXAZ O10 -2.574450 -2.443880 0.119710 1.520000
11 OXAZ OXAZ N11 -0.784090 -3.681460 0.063780 1.600000
12 OXAZ OXAZ O12 -4.320040 -3.925030 0.231080 1.520000
13 OXAZ OXAZ C13 -2.043830 -5.884680 0.168140 1.770000
14 OXAZ OXAZ C14 -1.244540 -8.066310 0.177640 1.770000
15 OXAZ OXAZ C15 0.052310 -8.899060 0.135760 1.770000
16 OXAZ OXAZ O16 -0.918760 -6.663430 0.133020 1.520000
17 OXAZ OXAZ H17 1.252970 2.282310 -0.155740 0.900000
18 OXAZ OXAZ H18 -1.221770 2.200130 -0.040530 0.900000
19 OXAZ OXAZ H19 -2.384360 0.009980 0.059090 0.900000
20 OXAZ OXAZ H20 1.395210 -2.008100 -0.071530 0.900000
21 OXAZ OXAZ H21 2.562540 0.178850 -0.171300 0.900000
22 OXAZ OXAZ H22 -3.033270 -6.342320 0.223210 0.900000
23 OXAZ OXAZ H23 -1.867540 -8.323760 -0.679230 0.900000
24 OXAZ OXAZ H24 -1.786000 -8.286150 1.098030 0.900000
25 OXAZ OXAZ H25 -0.196250 -9.959719 0.167400 0.900000
26 OXAZ OXAZ H26 0.675080 -8.646710 0.994020 0.900000
27 OXAZ OXAZ H27 0.595610 -8.681110 -0.783720 0.900000
```

Line command : see **export command**

Write rdf widget

Write a pre-calculated radial distribution function (RDF) to disk. The file can then be read into your favorite plot program to be displayed graphically.



File name:

Save as type: RDF data files (*.dat,*.DAT)

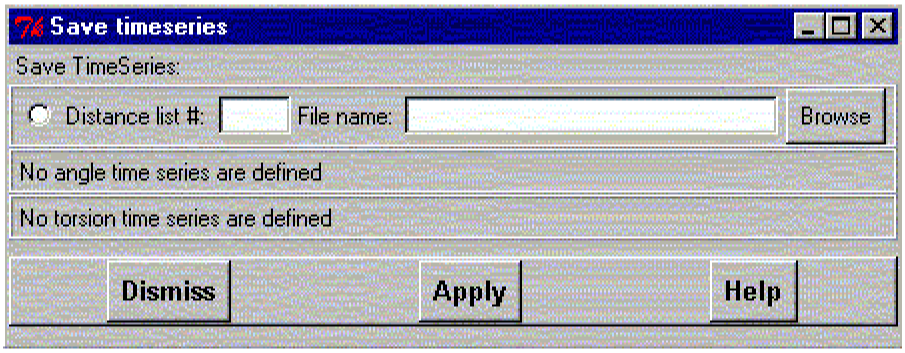
Line command: see **export command**

Copy time series to disk widget

Copy time series data to a file on the disk.

Select the type of time series array and the number of the distance, angle or torsion array to be copied to the disk. Give the file a name direct or choose a name using the file browser by clicking the **Browse** button.

To write the information on disk press the the **Apply** button.



Save TimeSeries:

○ Distance list #: File name:

No angle time series are defined

No torsion time series are defined

Line command: see **copy command**

Reset various things.

```
•  Atom colour;          Reset atom colours to default!
•  Atom connectivity...; Reset/calculate atom connectivity!
•  gOpenMol;             Reset gOpenMol and free all reserved memory!
•  View;                 Reset view to the startup view!
```

```
**************************************************************************
                    reset command
                 Leif Laaksonen CSC 1996
**************************************************************************

reset

#
# reset atom colours
#

        atomc$olours

#
# Reset gOpenMol
#

        gope$nmol

#
# Reset the view to the original
#

        rese$t  view

#
# Reset atom connectivity
#

        conn$ectivity   Segment Residue Atom
```

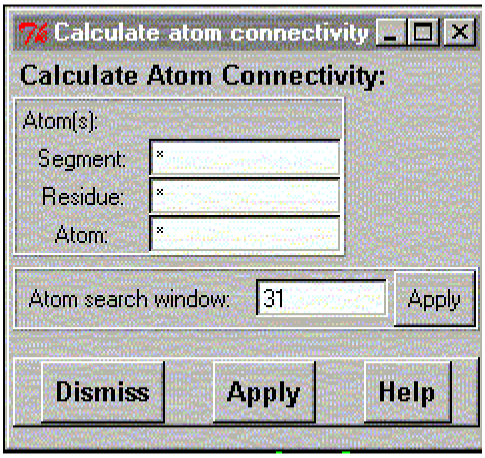
Calculate atom connection widget

This widget is helpful when one wants to recalculate atom connections (bonds) in case you you have deleted some or if you don't get all hydrogen atoms bonded in your structure.

gOpenMol uses a very simple algorithm to calculate possible bonds. It runs over all the atoms in the structure and uses a window (range) of atoms inside which it thinks the bonded atoms are located. By default there is a range defined (use **show atom window** command to display the current value) so when gOpenMol looks at atom N it checks range atoms backwards and range atoms forward from atom N to find the bonded neighbours. Some programs place added hydrogens at the end of the file, which might prevent gOpenMol from finding the right bonds.

To solve this problem the range can be redefined. Please supply a new value, inside which you think all bonded atoms will be.

It is also possible to recalculate the connectivity only for a set of atoms.



Calculate Atom Connectivity:

Atom(s):

Segment: *

Residue: *

Atom: *

Atom search window: 31

Make a hardcopy widget

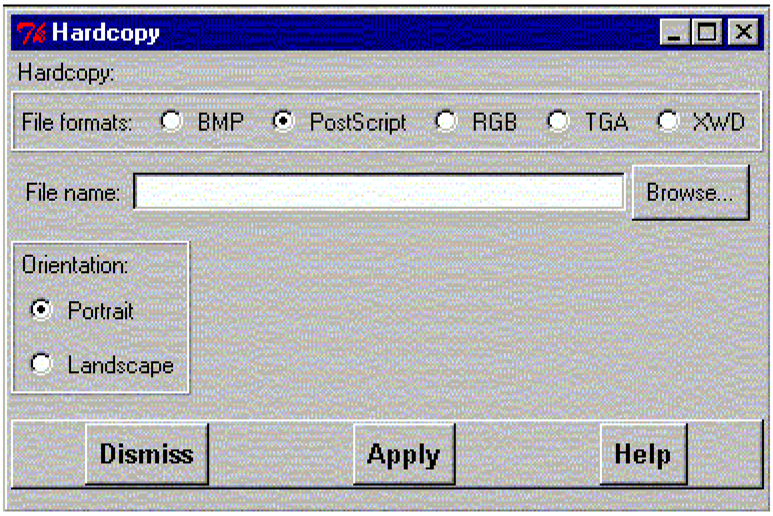
This is the main facility for producing a hardcopy of the picture in the graphics window. Currently gOpenMol supports the following output formats:

- Microsoft Windows BMP format
- PostScript format
- Silicon Graphics RGB format

- Targa format (this format is used for the mpeg animation part)
- X-Windows XWD format

These formats should be enough to export the pictures to almost any text processing or image editing program for further processing.

On top of the page are the radio buttons for the supported formats. Use the file selection browser to select a file name. For the PostScript case it is possible to define the picture orientation.



Hardcopy

Hardcopy:

File formats: ○ BMP ● PostScript ○ RGB ○ TGA ○ XWD

File name:

Orientation:

● Portrait

○ Landscape

Line command: see **hardcopy command**

Exit Halt program execution

(2) EDIT menu has the options:

Copy various things to the clipboard paste buffer (works only on Windows).

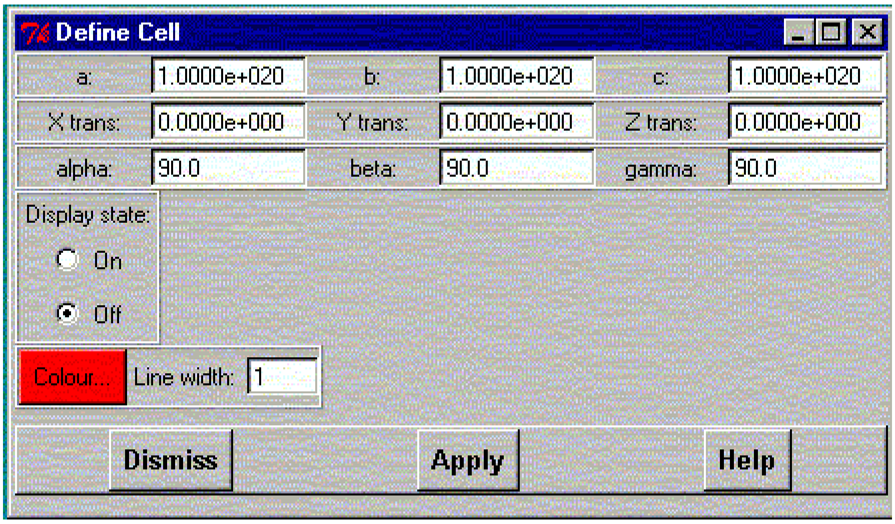
Bitmap;	Copy current graphics window into clipboard
Correlation;	Copy correlation data into the clipboard
MSD;	Copy the Mean Square Displacement data into clipboard
RDF;	Copy the radial distribution function into clipboard
Timeseries;	Copy time series into clipboard.

Define cell widget

The purpose of this widget is to define the properties of a cell (box) and control the display of the cell (box). The cell (box) can be usefull when one is displaying a molecular dynamics simulation (trajectory), that is using periodic boundary conditions.

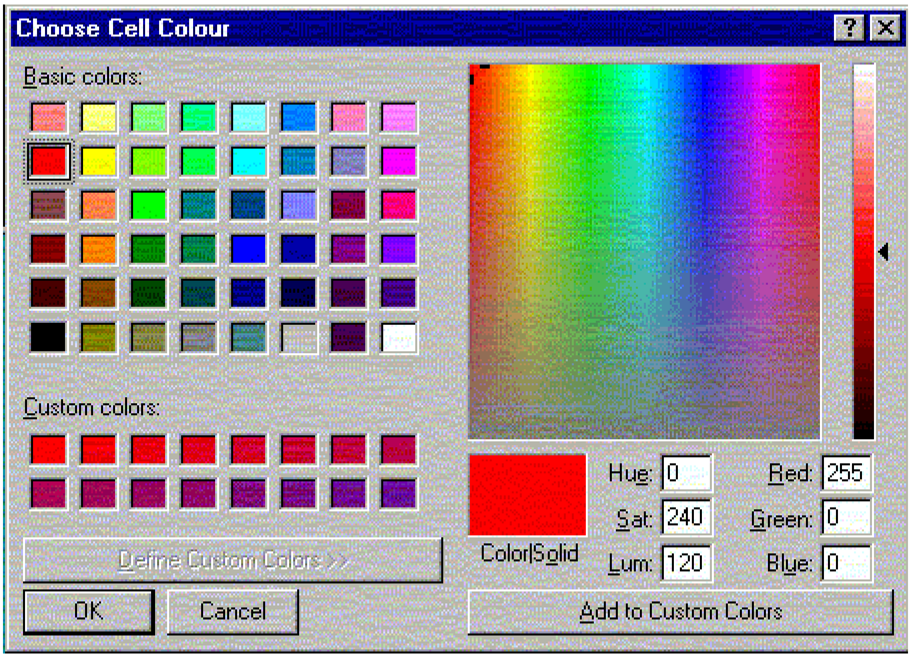
Define the cell properties, a, b and c distances (in Angstrom). The cell is by default placed in the center of the screen (or around the 0.0, 0.0, 0.0). However, the cell can be translated by defining the translation distances. The alpha, beta and gamma angles can (hopefully) some time be defined. **Currently the angles can only be 90 degrees.**

The cell can be displayed by changing the "Display state" to **ON**. The colour of the cell and the width of the cell (in pixels) can also be changed. Click on the "Colour" button to change the colour.



a: 1.0000e+020 b: 1.0000e+020 c: 1.0000e+020
X trans: 0.0000e+000 Y trans: 0.0000e+000 Z trans: 0.0000e+000
alpha: 90.0 beta: 90.0 gamma: 90.0
Display state: ○ On ● Off
Line width: 1

Pick the colour from the colour palette.



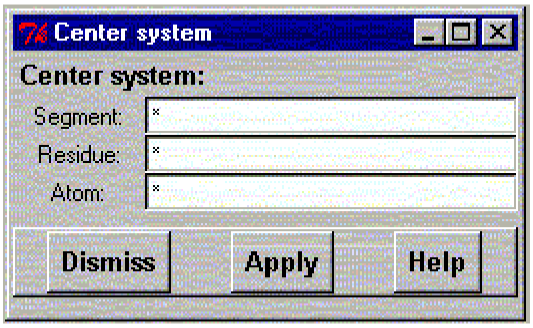
Hue: 0 Red: 255
Sat: 240 Green: 0
Lum: 120 Blue: 0

Line command: see **define command**

Center system widget

Center the system according to the given atom(s). It is also possible to give an absolute x,y z coordinate. The x coordinate value is placed in the segment field, y coordinate value in the residue field and z coordinate value is placed in the atom field.

Press then the "Apply" button.



Line command: see **center command**

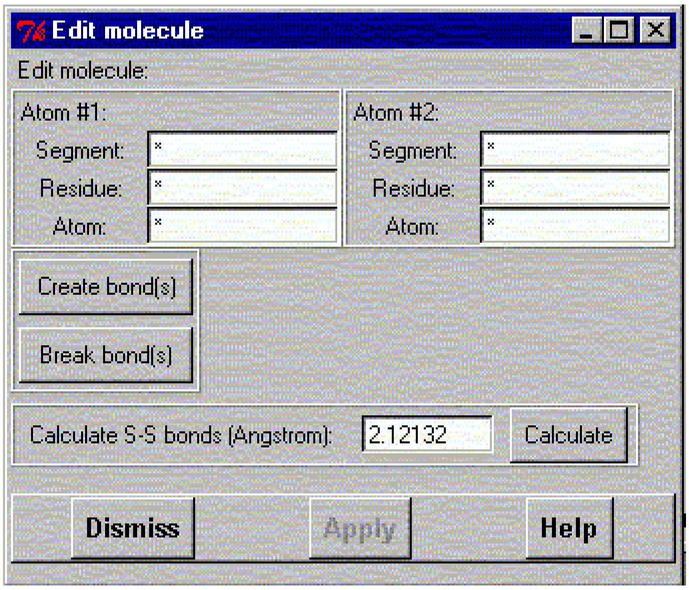
Edit molecule widget

Edit molecule geometry. Currently the only supported actions are:

- Create bond(s)
- Break bond(s)
- Find S-S bonds in proteins

Select the atom(s) and click on either the "Create bond" or "Break Bond" buttons. The S-S bond algorithm looks for S atoms inside a search radius, which can be changed by editing the input widget.

It is also possible to delete or create all bonds from a certain atom or atoms by deleting the atom selection input from the atom selection #2 list widgets.



Edit molecule:

Atom #1:
Segment: *
Residue: *
Atom: *

Atom #2:
Segment: *
Residue: *
Atom: *

Calculate S-S bonds (Angstrom): 2.12132

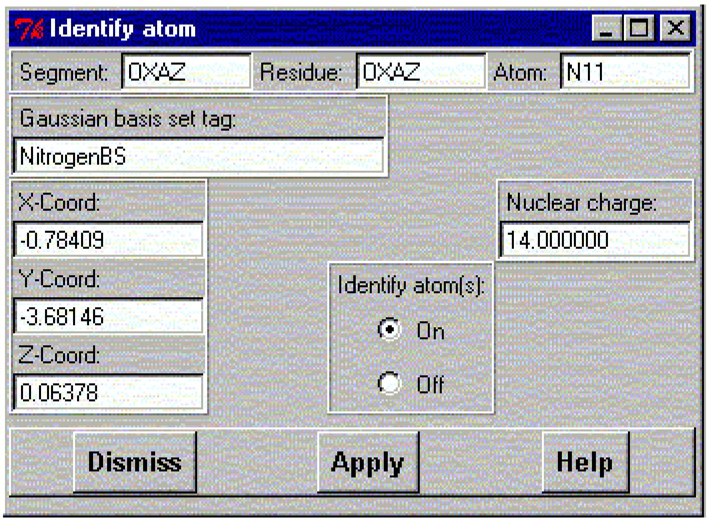
Line command: see **edit command**

Identify atom widget

This widget enables the user to click on an atom and identify the full properties for that atom.

Normally pressing the **left** mouse button and moving the mouse rotates the molecular system and pressing the **right** mouse button and moving the mouse zooms the molecular system.

Clicking the "Identify atom(s)" state to **on** turns the **left** mouse button to a pointer into the molecular system. Move the arrow over an atom and press the **left** mouse button and the widget will be filled with atom information about the that particular atom.



Segment: 0XAZ Residue: 0XAZ Atom: N11

Gaussian basis set tag: NitrogenBS

X-Coord: -0.78409

Y-Coord: -3.68146

Z-Coord: 0.06378

Nuclear charge: 14.000000

Identify atom(s): ● On ○ Off

Line command: see **No line command available**

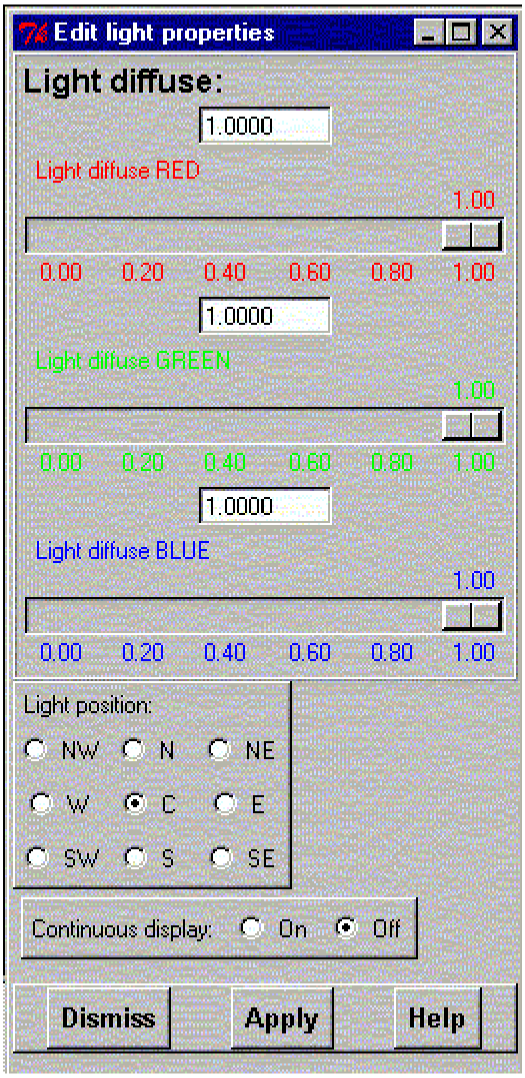
Edit light properties widget

Edit properties of the light (colour components) and the position of the light.

It is possible to the change the **red**, **green** and **blue** colour intensity of the light using the colour sliders.

The position of the light is by default along the z-axis but it can be changed according to the **north**, **north east**, **east**, **south east**, **south**, **south west**, **west** and **north west** scheme. Select the new light position by clicking the radio button.

To update the display while changing the value click the "Instant update" button.



Edit light properties

Light diffuse:

1.0000

Light diffuse RED 1.00

1.0000

Light diffuse GREEN 1.00

1.0000

Light diffuse BLUE 1.00

Light position:

○ NW ○ N ○ NE

○ W ● C ○ E

○ SW ○ S ○ SE

Continuous display: ○ On ● Off

Line command: see **define command**

Edit material properties widget

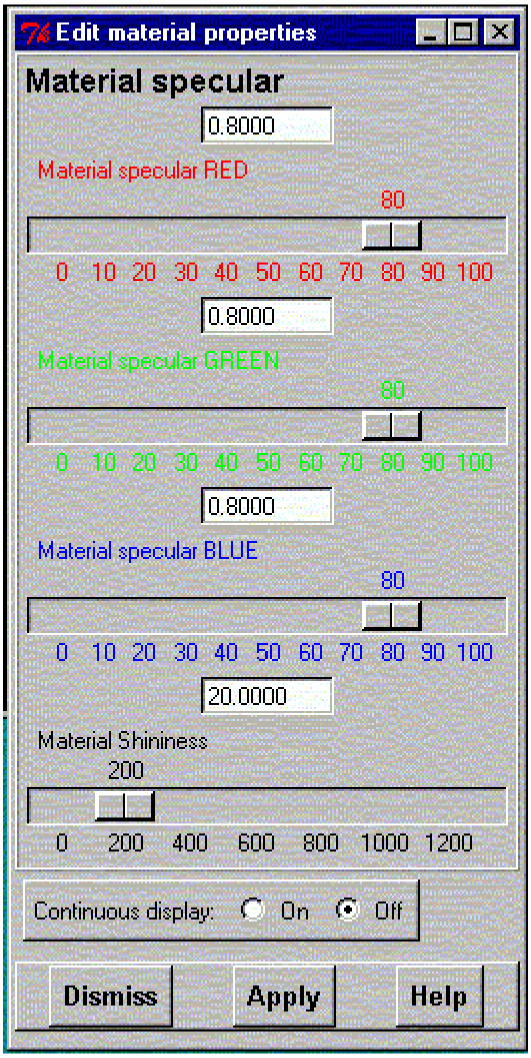
Edit surface material properties. It is possible to change the following properties:

- Material specular **red** colour.
- Material specular **green** colour.
- Material specular **blue** colour.
- Material specular **material shininess**.

The colour components define the colour of the reflecting light and the material shininess defines the "width" of the reflecting light (0 = wide and 128 = very concentrated).

Observe that the new properties are not shown until the "Display" command is executed or before the user presses the **gOpenMol logo** logi in the right upper corner.

If you want the new properties to be shown immediately while you change the slider values click the "Continuous display" **on**.



Edit material properties

Material specular

0.8000

Material specular RED

80

0.8000

Material specular GREEN

80

0.8000

Material specular BLUE

80

20.0000

Material Shininess

200

Continuous display: ○ On ● Off

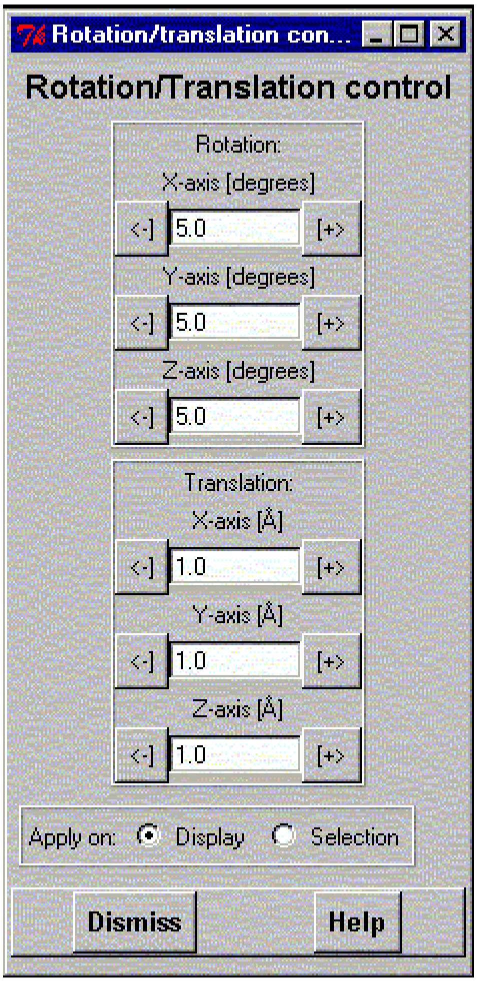
Line command: see **edit command**

Rotate/translate control widget

Rotate and translation control widget. Clicking on - or + rotates or translates the display with the value (step) defined in the entry between the signs. The step can be changed by writing a new value into the entry space.

The new display will not be showed before the "Display" command is executed or the gOpenMol logo is pressed.

The operations (rotation/translation) will be applied either on the display or the selection. According to the "Apply on" radiobutton.



Rotation:

X-axis [degrees]: <-] 5.0 [+>

Y-axis [degrees]: <-] 5.0 [+>

Z-axis [degrees]: <-] 5.0 [+>

Translation:

X-axis [Å]: <-] 1.0 [+>

Y-axis [Å]: <-] 1.0 [+>

Z-axis [Å]: <-] 1.0 [+>

Apply on: ● Display ○ Selection

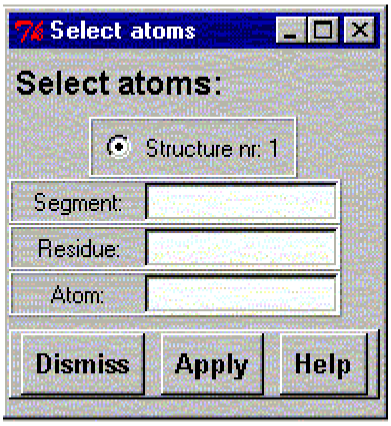
Line command: see

- **rotate command**
- **translate command**

Select atoms widget

Select atoms into a selection buffer. If there are several systems available select the one you want by clicking the structure through the radio buttons available.

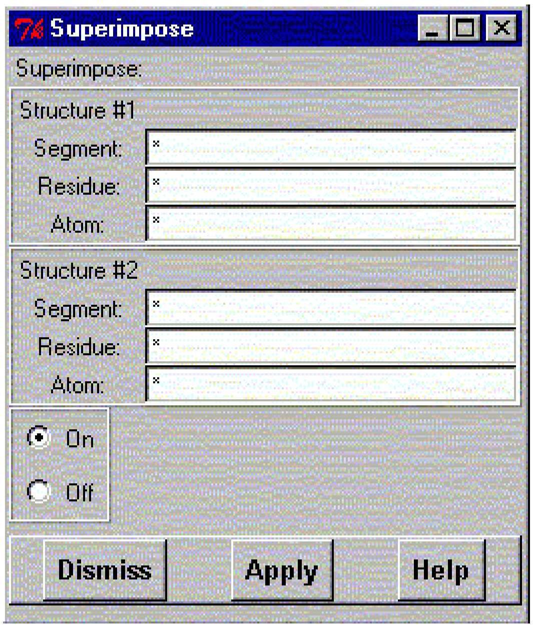
The selection buffer can be used for applying rotations or translations just to a subsection of a molecular system.



Line command: see **select command**

Superimpose atoms widget

Superimpose two structures. Select the atoms from the both sets. The number of atoms have to be the same in both sets.



Superimpose:

Structure #1
Segment: *
Residue: *
Atom: *

Structure #2
Segment: *
Residue: *
Atom: *

● On
○ Off

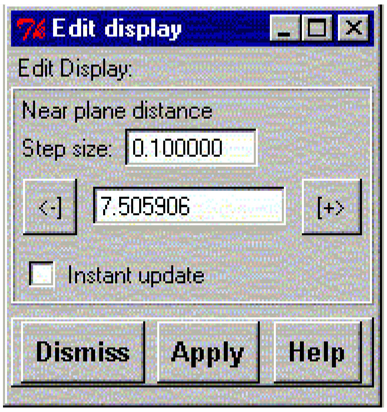
Line command: see **calculate quatfit command**

Edit display attributes widget

Define the distance to the near clipping plane. Pressing **-** and **+** signs decreases or increases the distance to the near clipping plane with the defined step length.

The new view will not be displayed before the "Display" command is applied or the **gOpenMol** logo is pressed.

To update the display after the changed value click the "Instant update" button.



Edit Display:

Near plane distance
Step size: 0.100000
7.505906
□ Instant update

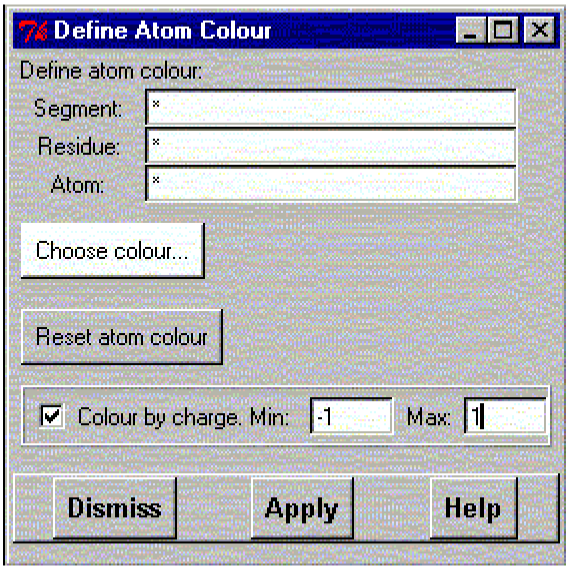
Line command: see **define command**

(3) VIEW menu has the options:

Select atom colour widget

Select atom colour by defining the atoms and click the "Choose colour" button to show the colour editor.

If the atom partial charges have been defined it's also possible to colour the atoms according to the partial charges. Define the partial atom charge range and press the "Apply" button. The colour range is **blue** for the lower limit and **red** for the upper and the colour goes through green.



Define atom colour:

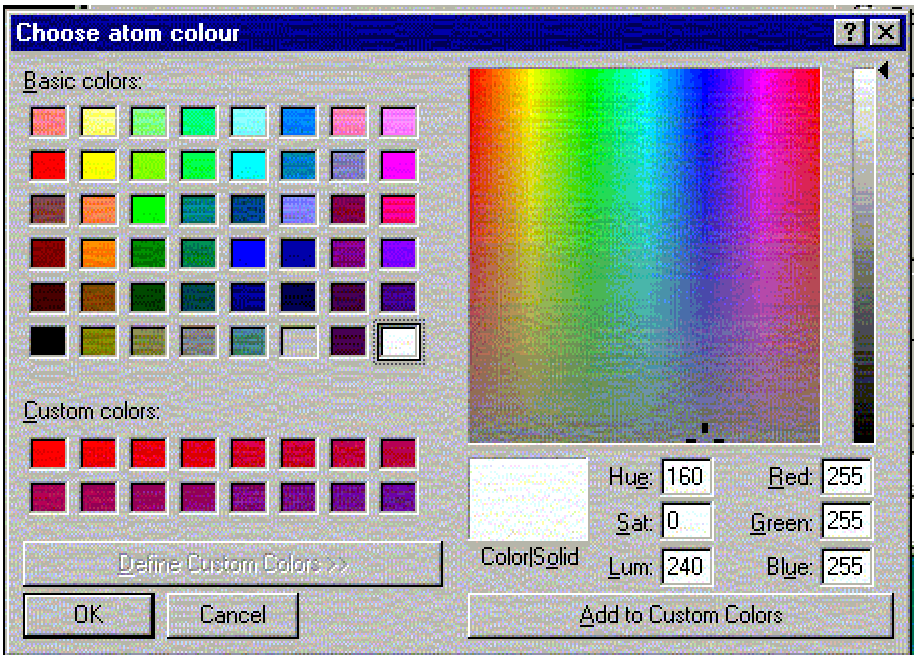
Segment: *

Residue: *

Atom: *

☑ Colour by charge. Min: -1 Max: 1

Pick or choose the colour using the colour editor and press the "Ok" button. Then press the "Apply" button to colour the selected atoms with the picked colour. The new colour will not be seen before the scene is redrawn.



Hue: 160
Sat: 0
Lum: 240
Red: 255
Green: 255
Blue: 255

Line command: see **atom command**

Atom display mask widget

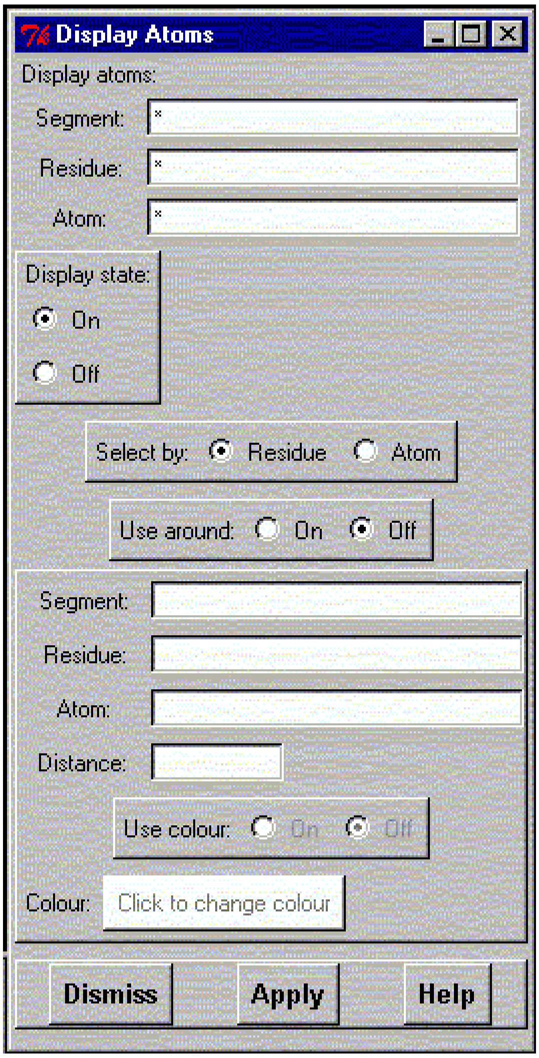
Select atom mask to be used for the display ot atoms. Select first the atoms to be displayed or not displayed by giving the atom mask end then defining the action ("On" or "Off") and press the "Apply" button.

It is also possible to use more complicated schemes:

- Display all atoms of a particular type around a predefined set of atoms within a particular radius.
- Display all atoms of a particular type around a predefined set of atoms within a particular radius in a selected colour.

The atom selection can be based on either:

- All atoms inside whole residues.
- Just the atoms that satisfy the selected radius.


Display Atoms

Display atoms:

Segment: *

Residue: *

Atom: *

Display state: ● On ○ Off

Select by: ● Residue ○ Atom

Use around: ○ On ● Off

Segment:

Residue:

Atom:

Distance:

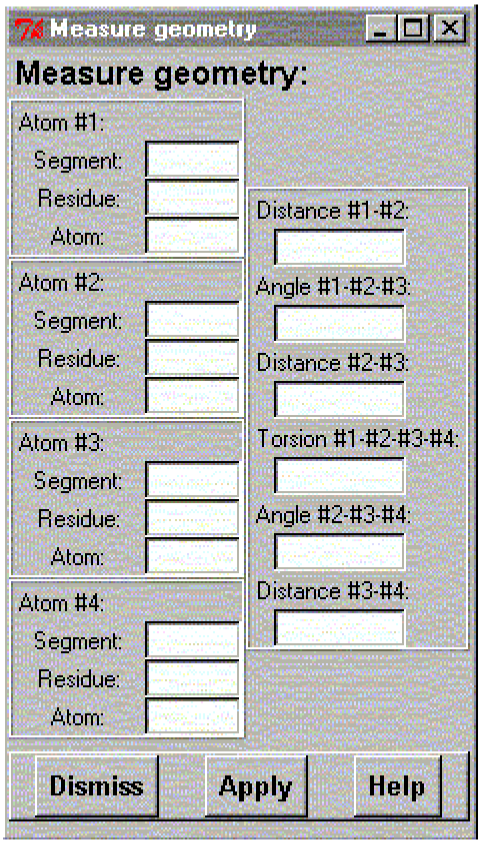
Use colour: ○ On ● Off

Colour: Click to change colour

Line command: see **atom command**

Measure Geometry Widget

Measure distance(s), angle(s) and/or torsion angle(s) between selected atoms. To define a distance you have to define two atoms, for an angle three atoms and for a torsion angle four atoms. Two atoms can only define a distance, three atoms will also define two distances and four atoms defines three distances, two angles and one torsion angle. Click on the "Apply" button and the widget will show you the values defined by the atoms.



Line command: see **calculate command**

Plot stereo pair ("crossed eye") widget

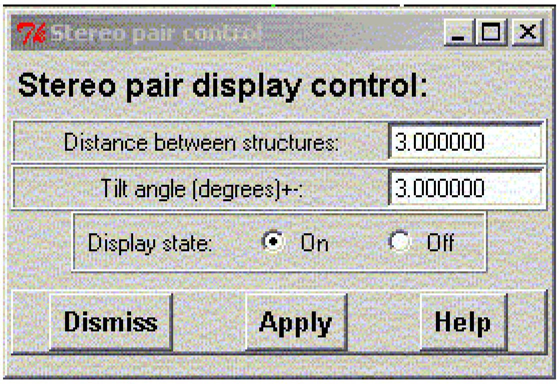
It is possible to generate a stereo pair or "crossed eye" view of a system. The view is built by duplicating the object, with a defined distance, and rotating the two pictures with a small angle.

There are two values that have to be defined:

- The distance with which the objects are separated.
- The angle with which the objects are "tilted". The left object is "tilted" -angle and the right object +angle around the y-axis.

Usually there is no need to change the "tilt" angle but the distance between the two images of the structure needs to be specified.

Click on the Display state: "On" and theon the "Apply" button to activate the display. If you need to change the distance between the two images write in a new value into the widget and press the "Apply" button.



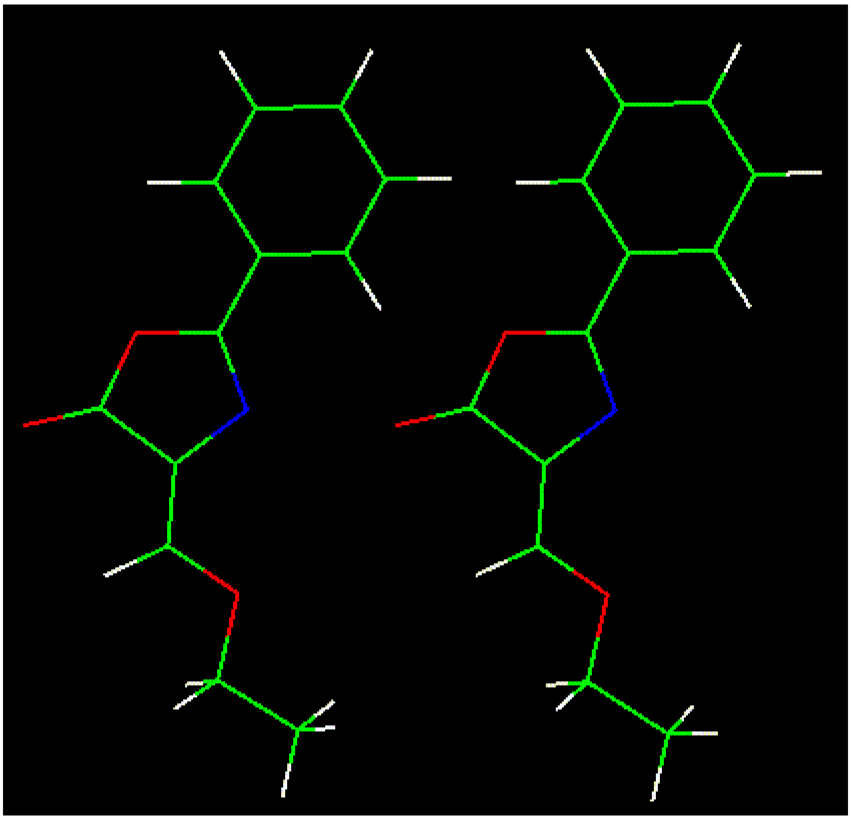
Stereo pair display control:

Distance between structures: 3.000000

Tilt angle (degrees)+-: 3.000000

Display state: ● On ○ Off

The display should look like the following when the "crossed" eye stereo is on.



Line command: **plot spair**

Plot hardware stereo widget

If your display device supports hardware based stereo (stereo in a window) it is possible to start gOpenMol in the quadstereo mode and turn the hardware stereo on.

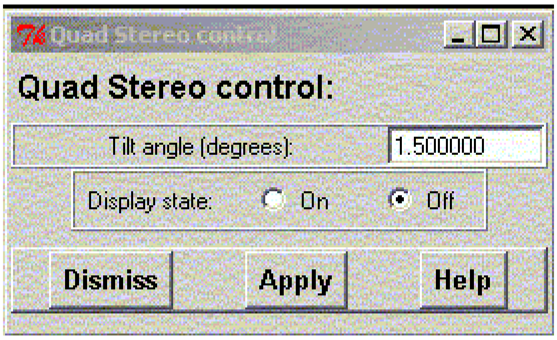
gOpenMol can be started in the quadstereo mode at startup using the "-s" startup command line option.

rungopenmol -s

There is only one value that have to be defined:

- The angle with which the objects are "tilted". The left object is "tilted" -angle and the right object +angle around the y-axis.

The default angle should work for most cases. To enable the display click the Display state: to "On" and then the "Apply" button.



Quad Stereo control:

Tilt angle (degrees): 1.500000

Display state: ○ On ● Off

Line command: **no line command available**

```
/************************************************************************/
/* Added by Sigismondo Boschi - CINECA - Italy:
   Stereo for High-end graphic machines, e.g. SGI Onyx2

   The high-end Stereo graphics is so called "quad buffer" or
   "stereo-in-a-window". It allows to have stereo and double-buffering
   without the need of splitting vertically the viewport -
   and consequenty you do not need to have full-screen resolution.

   The basic steps are:

   1) to have a stereo-capable visual (GLUT_STEREO)
   2) draw the left eye in the BACK_LEFT drawing buffer:
      glDrawBuffer(GL_BACK_LEFT);
   3) draw the right eye in the BACK_RIGHT drawing buffer:
      glDrawBuffer(GL_BACK_RIGHT);
   3) put them in the FRONT_LEFT & FRONT_RIGHT by swapping once the
      buffers:
      glutSwapBuffers();
   If you need to plot "non-stereo" staff just plot it to the BACK buffer:
   glDrawBuffer(GL_BACK);
   It will be authomatically plotted in both the back buffers. In this way
   you can have non-stereo images in a stereo-window.

 In order to setup the stereo you need to give a setmon, but the argument
 Depends on the hardware architecture. In our machine the framebuffer
 setups are in /usr/gfx/ucode/vfc/vfs/. The ones with an "s" are the stereo
 ones, e.g.:
 1280x1024_96s.vfo
 On our Onyx we use ircombine, since the machine has many graphic
 pipelines, and many channels (displays). Ircombine manage all of them.

 About left and right buffers: beware they exists only if you have a
 "quad buffer" framebuffer - also called "stereo-in-a-window" capable.
 If stereo is not active only the left buffer is projected, and so it is
 normal you see only the left eye  output. It is also possible you can
 draw the right buffer, but if stereo is not activated, you have no way
 to see it. Anyway, if it runs, it should means the display is quad
 buffer capable, but not correctly set-up. That is what comes from
 my small experience anyway.
 When you invoke "swapbuffers" both buffers are swapped in the same moment:
 back_left goes front_left and back_right goes front_right and viceversa.

 On other low-end machines, like SGI O2, usually you have not
 "stereo-in-a-window", but just "full-screen-stereo", that is, you open
 a full screen window, and the upper part of the screen is projected to
 the left eye, the lower to the right one. I can give you some pointers,
 where I have collected the information.

 - "man stereo" on an SGI
 - http://techpubs.sgi.com/ - and look here for "stereo".

 */

/**********************************************************************/
```

Contour Manager Widget

This is the main contour display widget. Through this widget you can control the contour files, isosurface values, colours, surface smoothing and many more things. The data is defined as scalar values in a grid box. Before you can display any isosurface contour you have to read in the molecular system for which the grid data has been generated (this is usually a coordinate data file).

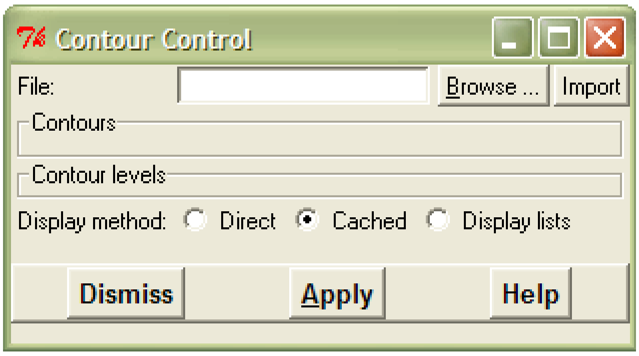
The steps to do:

1. Read the atom coordinate information (through import coordinates) for your molecular system.

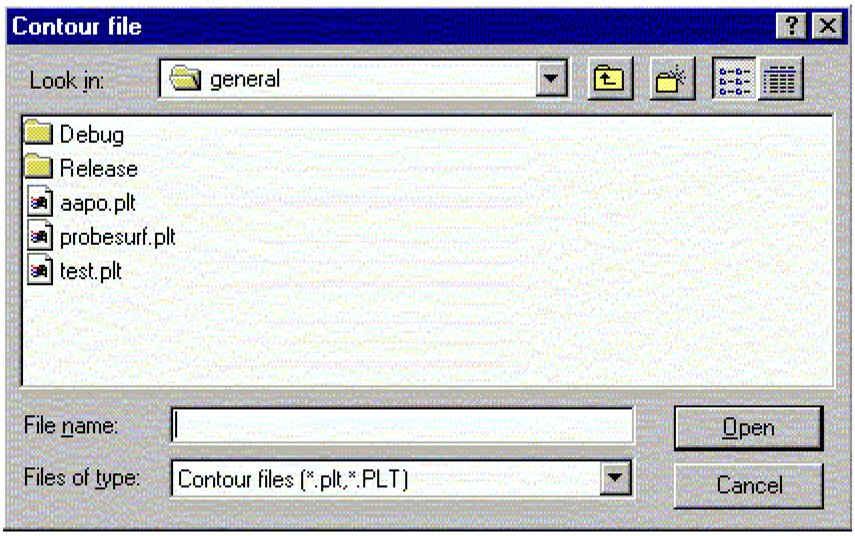
2. Open the contour manager widget (if it is not open alredy).

3. The contour data files are usually *.plt files (you can of course use your own file name extension). Click on the "Browse" button to open the file browser to see the *.plt files. Double click on the file (or a single click and then press the "Open" button) to select the file.

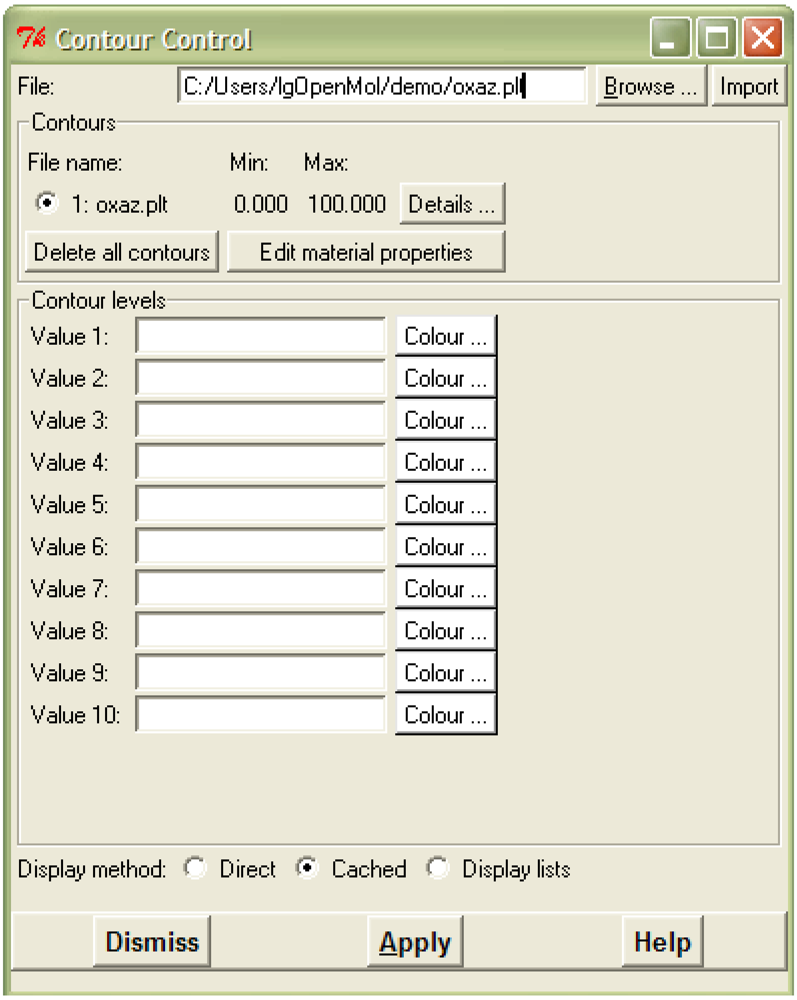
4. Press the "Import file" button to read in the grid data to gOpenMol.



File browser widget used to select a plot (*.plt) file:



After importing the plot file:



File: C:/Users/lgOpenMol/demo/oxaz.plt

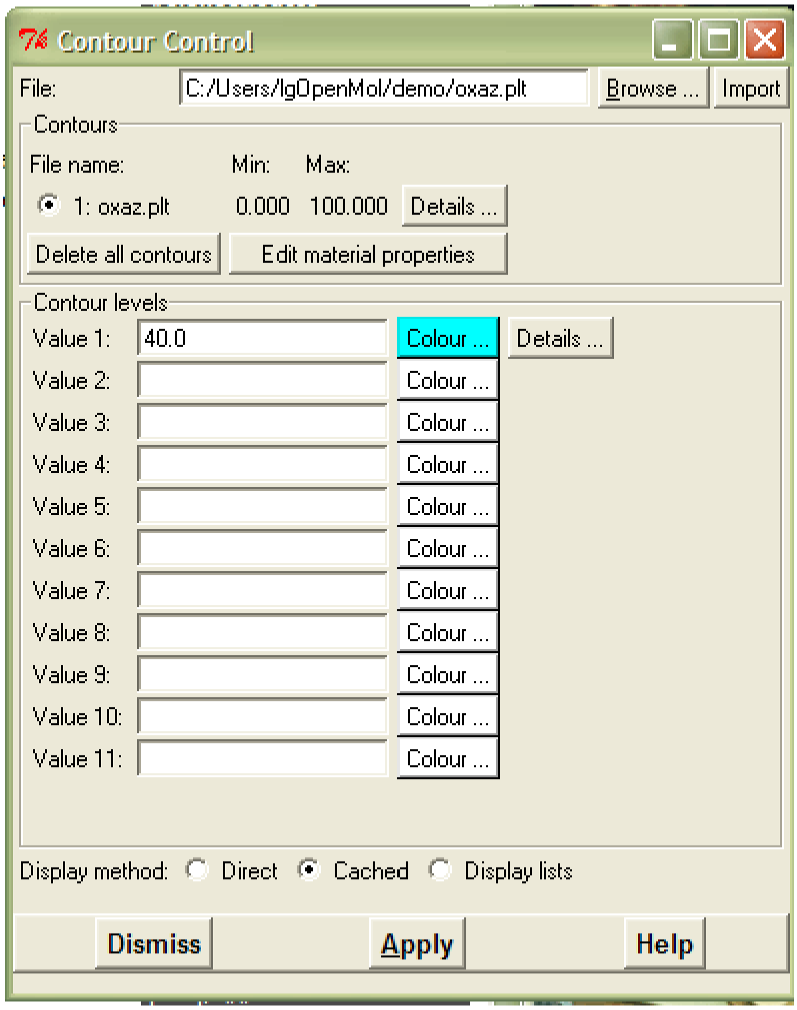
File name:	Min:	Max:
● 1: oxaz.plt	0.000	100.000

Display method: ○ Direct ● Cached ○ Display lists

1. Write the isocontour value into the input field (starting upwards).
2. Click on the "Colour(number)" button to choose a colour for the surface
3. If you want to map (colour) the surface according to the grid values of an other grid (*.plt) file press the "Mapping" button.
4. The contour surface renderer can be used in three different modes:
   - **direct** mode where the polygons are drawn directly on the screen,
   - **cached** mode where the polygons are saved into memory drawn from memory,
   - **display** lists are used for the display of the polygons.
   - the last displayed contour polygon data information is saved in an array and can be retrieved with the show command in **direct** and **cached** modes.

5. Press the "Apply" button"!

After defining the individual contour calues it is possible to change the surface details:



File: C:/Users/lgOpenMol/demo/oxaz.plt

Contours

File name:	Min:	Max:
● 1: oxaz.plt	0.000	100.000

Contour levels

Value 1: 40.0

Display method: ○ Direct ● Cached ○ Display lists

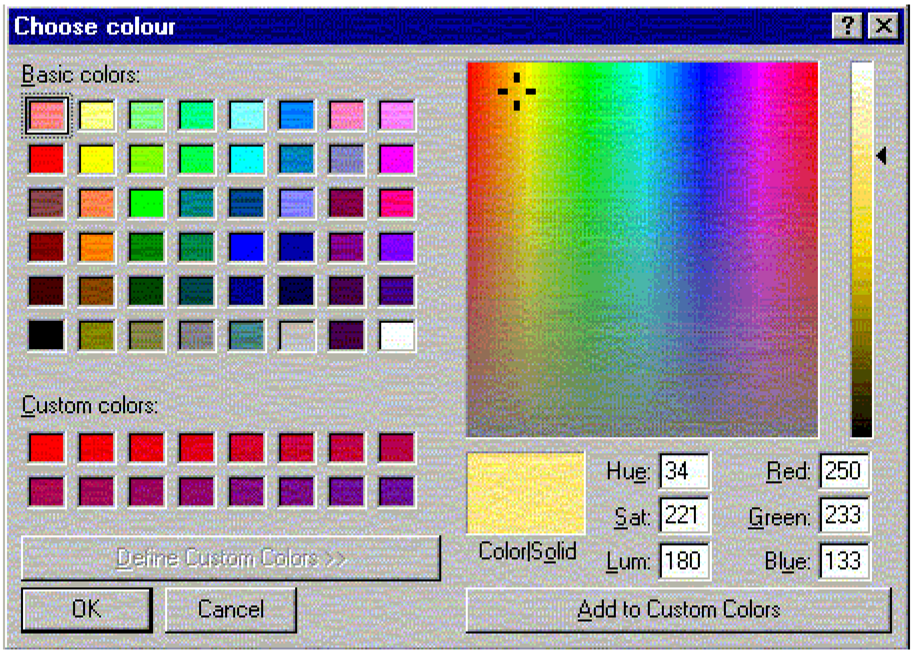
- If you want the surface to be smoothed press the "Details" button.
- If you want to make the surface transparent press the "Details" button.
- If you want to switch between mesh and solid type of surfaces press the "Details" button.
- If you want to turn a particular surface off press the "Details" button

You can change either the isocontour value, colour, smoothing by pressing the "Details" button for the chosen isocontour value.

You can also manipulate your mesh data files with the "contman" program. With the "contman" program you can add or sustract two contour files (A = B + C) or you can make a smooth transition from one contour file to an other using A = w * B + (1 - w)

C, where w is a mixing constant (0.0 ... 1.0). To have a look at the input to the "contman" program write "contman -h".

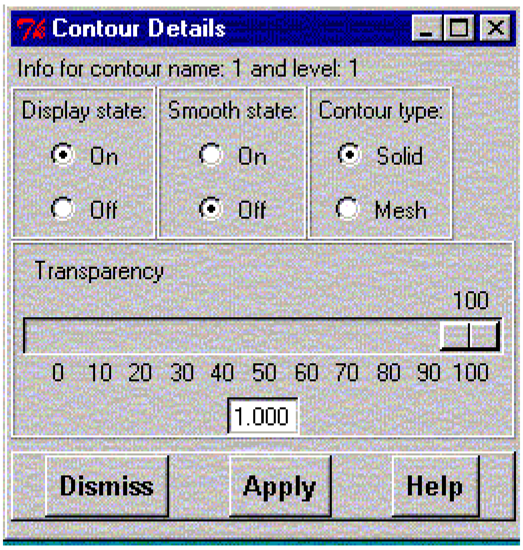
Widget to select the colour for a surface:



Hue: 34
Sat: 221
Lum: 180
Red: 250
Green: 233
Blue: 133

Widget to change detailed contour parameters:

- Click to set the display state, smoothing state, contour type (mesh/solid).
- Change the transparency of the surface.
- **Remember to press the "Apply" button to apply the selection.**



Info for contour name: 1 and level: 1

Display state: ● On ○ Off

Smooth state: ○ On ● Off

Contour type: ● Solid ○ Mesh

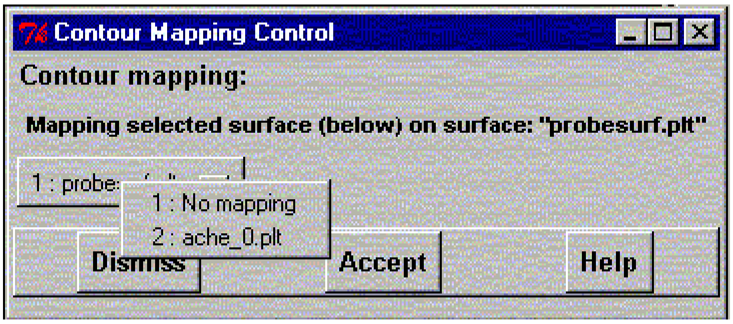
Transparency: 100

1.000

Line command: see **contour command**

Contour Mapping Widget

This is the contour mapping widget. Through this widget you can control the mapping (colouring) of an isocontour surface according to the grid values of an other grid (*.plt) file. The colouring can be scaled according to any grid value range.



Contour mapping:

Mapping selected surface (below) on surface: "probesurf.plt"

The steps to do:

1. Read the atom coordinate information (through import coordinates) for your molecular system.
2. Open the contour manager widget (if it is not open alredy).
3. The contour data files are usually *.plt files (you can of course use your own file name extension). Click on the "Browse" button to open the file browser to see the *.plt files. Double click on the file (or a single click and then press the "Open" button) to select the file.
4. Press the "Import file" button to read in the grid data to gOpenMol.
5. Repeat the previous steps to read at least one more grid (*.plt) files.
6. Click on the radio button for the file for which you want to generate the isocontour surface.
7. Click on the "Mapping" button to open the "Mapping control widget". This you already did because you got this help text.
8. Click on the option menu button to select the file from which the colouring (mapping) will be taken. Finish by pressing the "Accept" button.
9. Write the isocontour value into the input field (starting upwards). If you want to apply an other colour range than the default fill in on the same line the **min.** and **max.** values for the range. The colouring will be from min (blue) to max

(red). If the values are filled from **max.** to **min.** the colouring will be the opposite.

10. Press the "Apply" button"!

Surface details:

- If you want the surface to be smoothed press the "Details" button.
- If you want to make the surface transparent press the "Details" button.
- If you want to switch between mesh and solid type of surfaces press the "Details" button.
- If you want to turn a particular surface off press the "Details" button

Please observe that the two grid files have to have the same number of grid points in the x, y and z directions and the grid data has to be defined in the same x, y and z space for the both grid files.

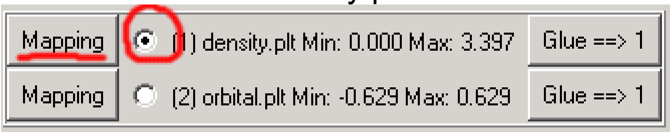
If you don't know the grid space you can run the format to unformat *.plt file converter (pltfile) in the bin/ directory.

You can change either the isocontour value, colour, smoothing by pressing the "Details" button for the chosen isocontour value.

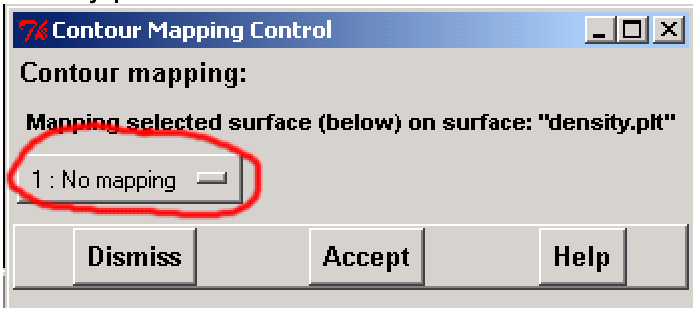
Surface mapping example

The idea of the surface mapping can be demonstrated using the density.crd coordinate file, the density.plt and orbital.plt files in the demo/ directory. The idea of the example is to plot the electron density at an isocontour value (0.1) and map the orbital grid values upon this surface. Do the following:

1. Import the density.crd file into gopenmol.
2. Import the density.plt file through the “View ➔ Contour” widget.
3. Import the orbital.plt file through the “View ➔ Contour” widget.
4. Click to select the density.plt file


Mapping
● (1) density.plt Min: 0.000 Max: 3.397
Glue ==> 1
Mapping
○ (2) orbital.plt Min: -0.629 Max: 0.629
Glue ==> 1

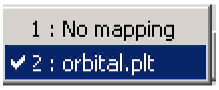
5. Press the “Mapping” button to define which grid data is to be mapped on the density.plt isocontour surface.

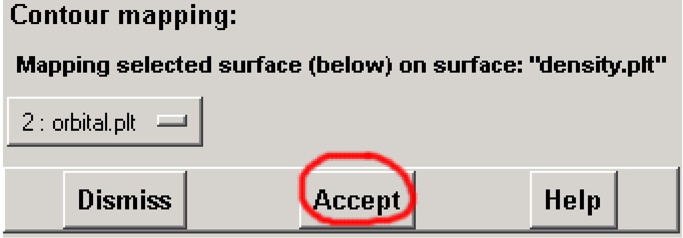


Contour Mapping Control

Contour mapping:

Mapping selected surface (below) on surface: "density.plt"

6. Select  from the list and press the “Accept” button. In some cases the “Contour Mapping Control” widget may disappear behind the “Contour Control” widget. This will not work until you have pressed the “Accept button”. Press then the “Dismiss” button to close the widget.

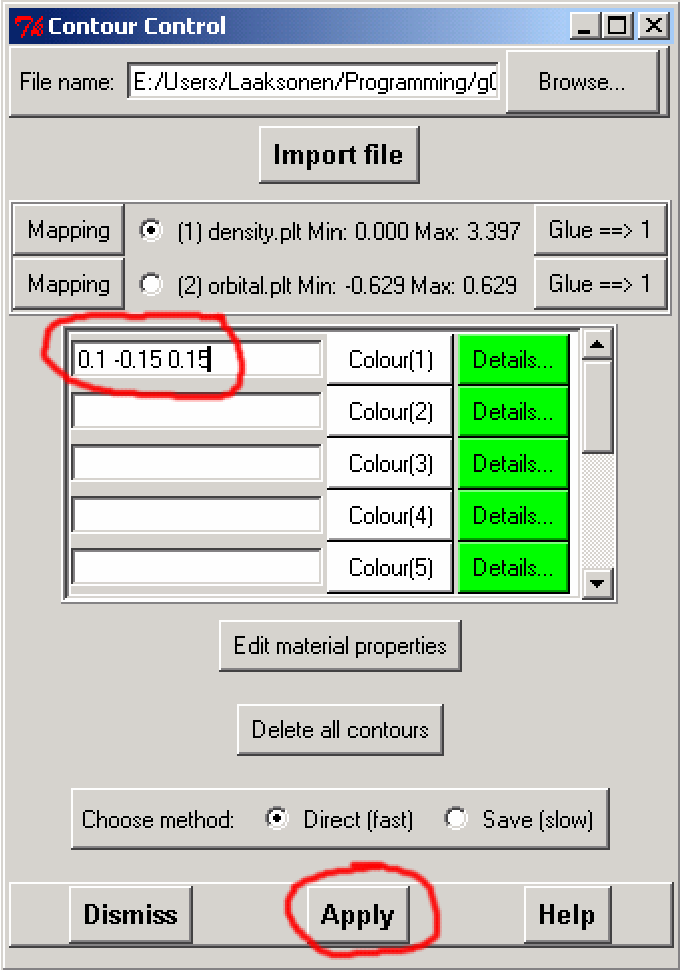


Contour mapping:

Mapping selected surface (below) on surface: "density.plt"

7. Write then “0.1 –0.15 0.15” into the first line. This means that an isocontour surface with the density value 0.1 is generated. Upon this surface the orbital values are mapped so that the values are scaled between –0.15 and +0.15. The lower value will be blue and the high value is red while the value in the

middle is green. Press then the “Apply” button.



File name: E:/Users/Laaksonen/Programming/g(

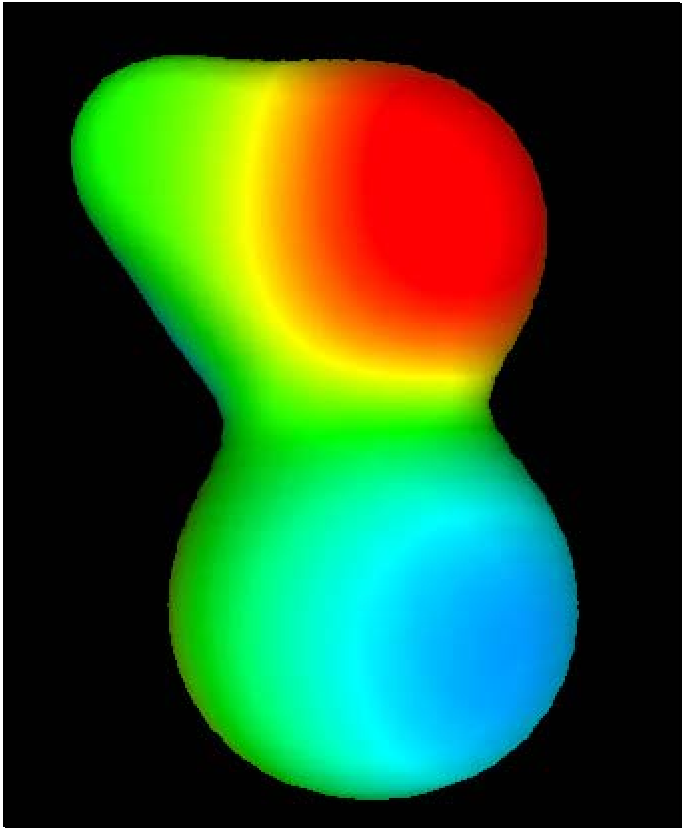
(1) density.plt Min: 0.000 Max: 3.397

(2) orbital.plt Min: -0.629 Max: 0.629

0.1 -0.15 0.15

Choose method: ● Direct (fast) ○ Save (slow)

8. You will the see this picture in the graphics screen.



9. The same can be achieved after reading the two plot files with the following line command: **contour plot {1 2} 0.1 -0.15 0.15**. Please observe that the plot files have to be read in the order density.plt and then orbital.plt. If you read the files in the opposite order the line command is: **contour plot {2 1} 0.1 -0.15 0.15**.

Surface “glueing” example

The main contour widget has a button with the text “Glue ==> 1” included. This defines the structure number to which the contour file should be glued (attached). By default any contour file is always glued (attached) to structure number 1 and the button has no meaning if there is only one structure.

The following example demonstrates how two structures are read into gOpenMol, how they are translated independently and two contour files are glued (attached) to the structures.

1. Import the coordinate file oxaz.crd in the demo/ directory.
2. Import the coordinate file orbital.crd in the demo/ directory with the append option clicked.

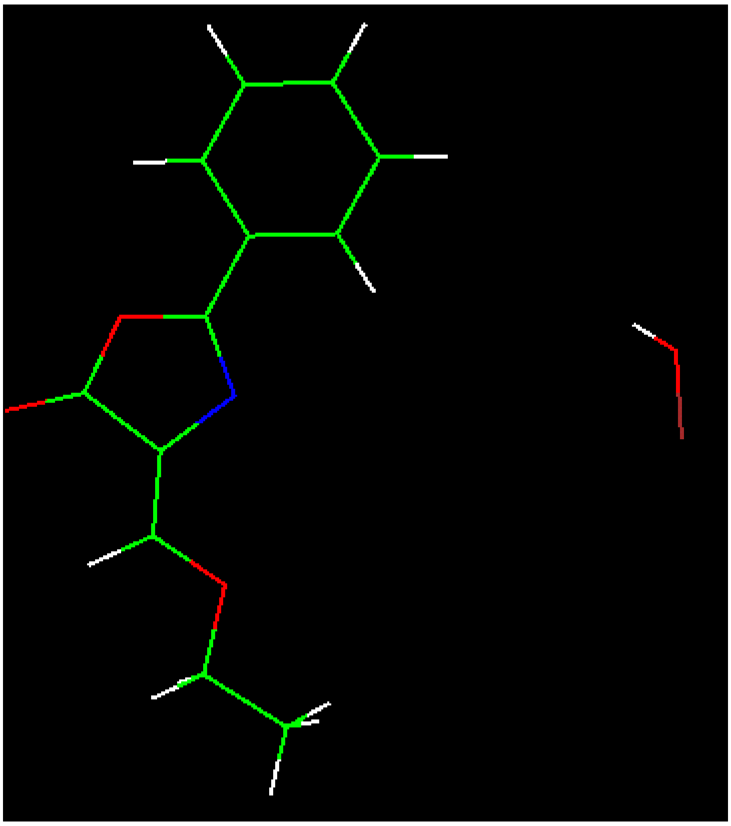
3. Click the Transformation: Local option on the main window. The two structures have now a local center around which they rotate.

4. Unclick first the orbital.crd in the main window. All rotations and translations will now only apply on the oxaz.crd structure.

5. Press the left Shift key and then the left mouse button and move the mouse. You can now move the oxaz.crd structure to a new position on the screen. Remember to release first the mouse button!

6. Click the orbital.crd and unclick the oxaz.crd.

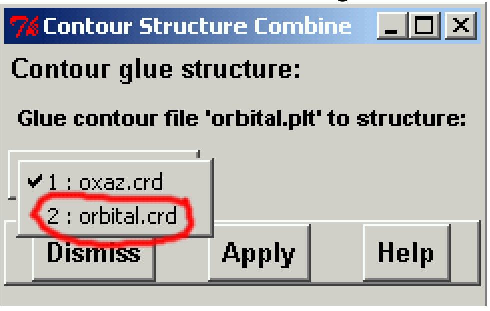
7. Press the left Shift key and then the left mouse button and move the mouse. You can now move the orbital.crd structure to a new position on the screen. Remember to release first the mouse button!



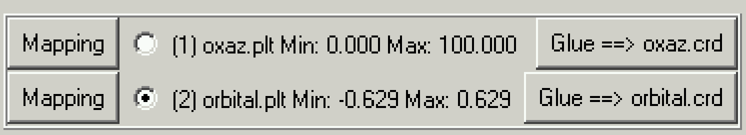
8. Open the "View ➔ Contour" widget and import the oxaz.plt (Connolly type of surface) and orbital.plt (molecular orbital) files from the demo/ directory.

9. Click on the button "Glue ==> oxaz.crd" orbital.plt file line to get the structure selection widget and select the orbital.crd. Press then the "Apply" button. Be

careful because the widget can disappear behind the contour widget.


Contour Structure Combine
Contour glue structure:
Glue contour file 'orbital.plt' to structure:
✓1 : oxaz.crd
2 : orbital.crd
Dismiss
Apply
Help

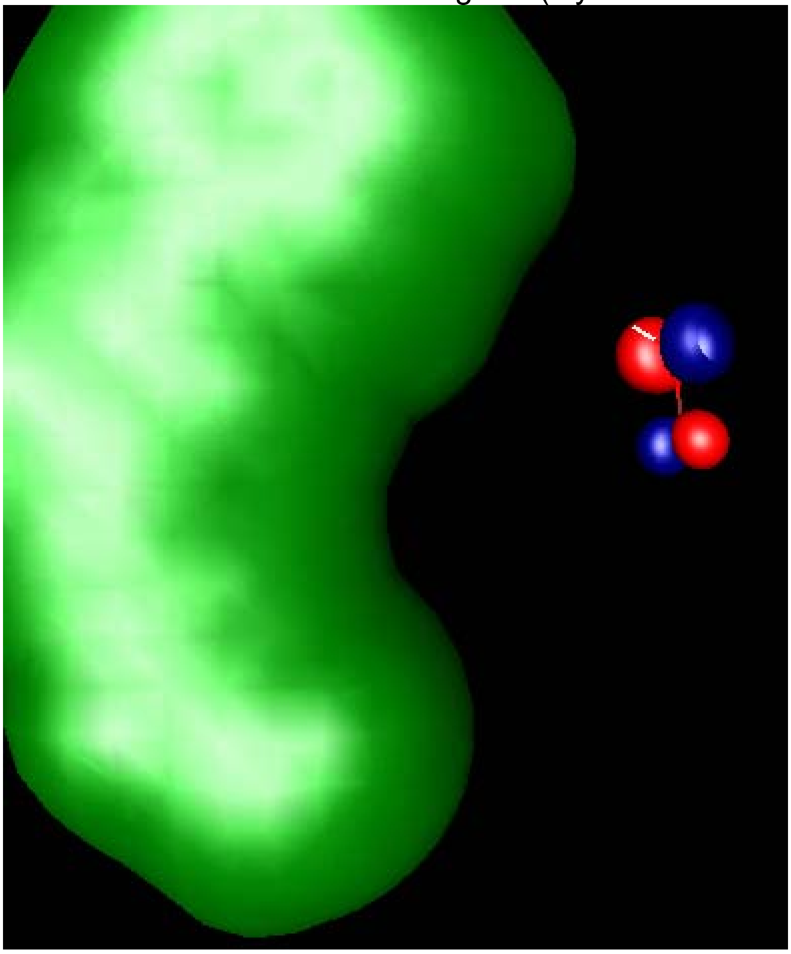
10. Your Contour widget should now look like:


Mapping
○ (1) oxaz.plt Min: 0.000 Max: 100.000
Glue ==> oxaz.crd
Mapping
● (2) orbital.plt Min: -0.629 Max: 0.629
Glue ==> orbital.crd

11. Click now the radio button for oxaz.plt and fill in the value 40. on the first input line. Select a colour for that value. Press the “Apply” button.

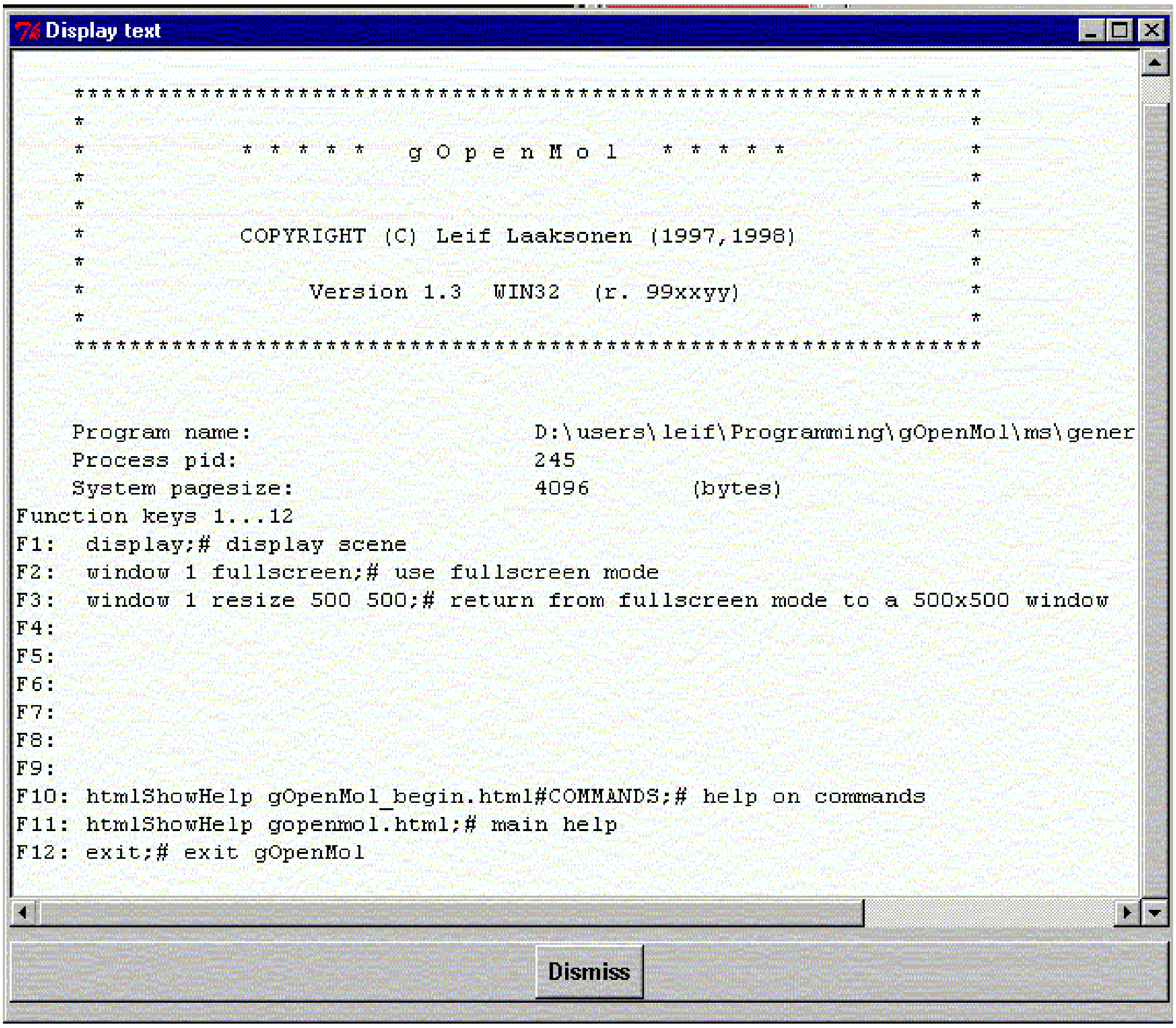
12. Click now the radio button for orbital.plt and fill in the values 0.1 on the first line and –0.1. on the second input line. Select a colour for the values. Press the “Apply” button.

13. The result should look something like (if you have used the same colours):



Text output widget

While running gOpenMol the program writes a lot of messages or results from various calculation routines to the output. The text can be seen in the window that started the process. However, if you are running Windows you will not be able to scroll the window. Some times some commands generates also a lot of output and one wants to cut and past the result to other applications. By opening the text output widget one gets a better control over the text output.



```
    *****************************************************************
    *                                                               *
    *           * * * * *   g O p e n M o l   * * * * *             *
    *                                                               *
    *                                                               *
    *           COPYRIGHT (C) Leif Laaksonen (1997,1998)            *
    *                                                               *
    *                Version 1.3  WIN32  (r. 99xxyy)                *
    *                                                               *
    *****************************************************************


    Program name:                    D:\users\leif\Programming\gOpenMol\ms\gener
    Process pid:                     245
    System pagesize:                 4096        (bytes)
Function keys 1...12
F1:  display;# display scene
F2:  window 1 fullscreen;# use fullscreen mode
F3:  window 1 resize 500 500;# return from fullscreen mode to a 500x500 window
F4:
F5:
F6:
F7:
F8:
F9:
F10: htmlShowHelp gOpenMol_begin.html#COMMANDS;# help on commands
F11: htmlShowHelp gopenmol.html;# main help
F12: exit;# exit gOpenMol
```

(4) COLOUR menu has the options:

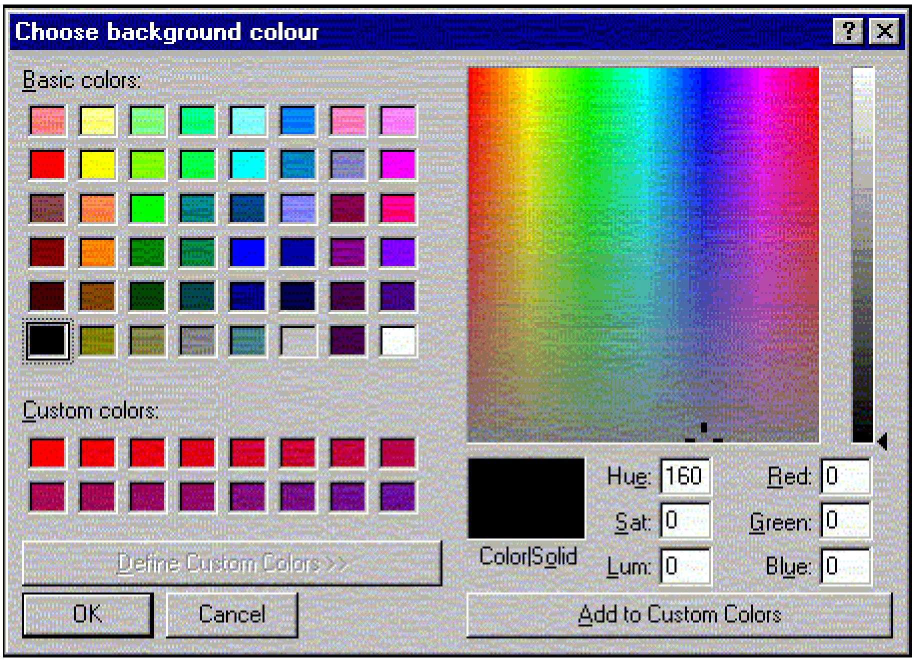
Change background colour in graphics window.

- Background, Change background colour with a Tcl/Tk tool.
- Background, Change background colour with a RGB tool.

Atoms, Change atom colour.

Select background colour widget

Select the back ground colour from the widget.

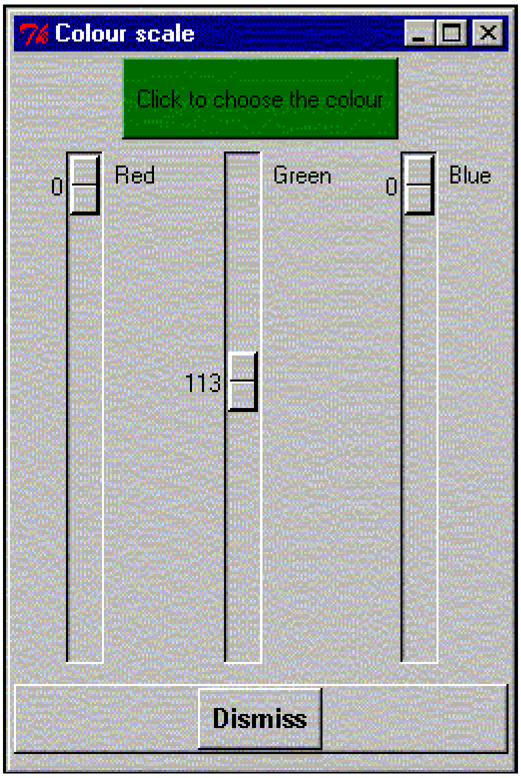


Hue: 160
Sat: 0
Lum: 0
Red: 0
Green: 0
Blue: 0

Line command: **define command**

Select background colour widget (RBG)

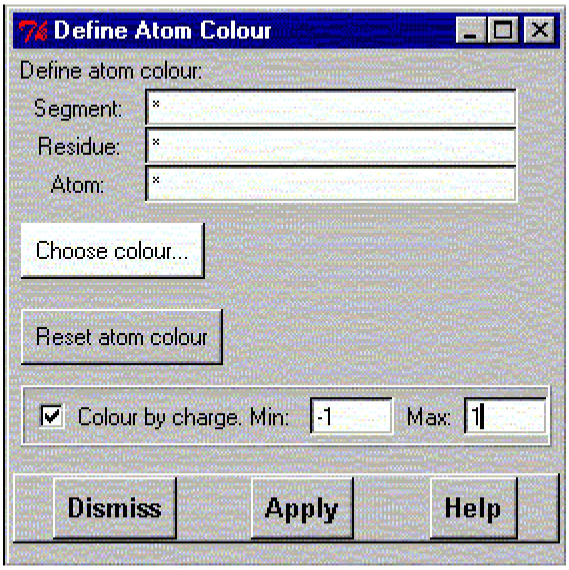
Use the red, green and blue sliders to define the background colour.



Select atom colour widget

Select atom colour by defining the atoms and click the "Choose colour" button to show the colour editor.

If the atom partial charges have been defined it's also possible to colour the atoms according to the partial charges. Define the partial atom charge range and press the "Apply" button. The colour range is **blue** for the lower limit and **red** for the upper and the colour goes through green.



Define atom colour:

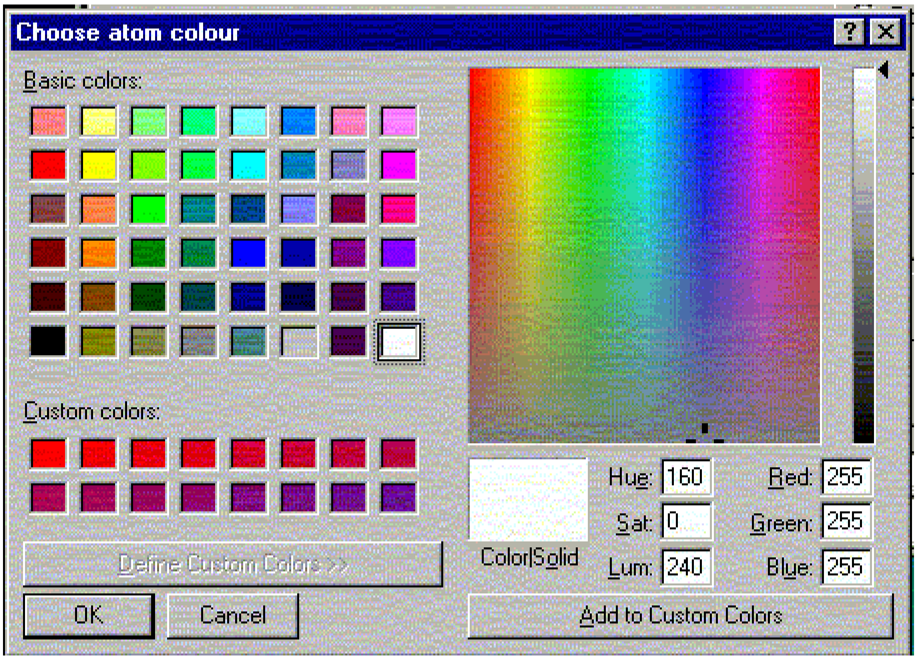
Segment: *

Residue: *

Atom: *

☑ Colour by charge. Min: -1 Max: 1

Pick or choose the colour using the colour editor and press the "Ok" button. Then press the "Apply" button to colour the selected atoms with the picked colour. The new colour will not be sen before the scene is redrawn.



Hue: 160 Red: 255

Sat: 0 Green: 255

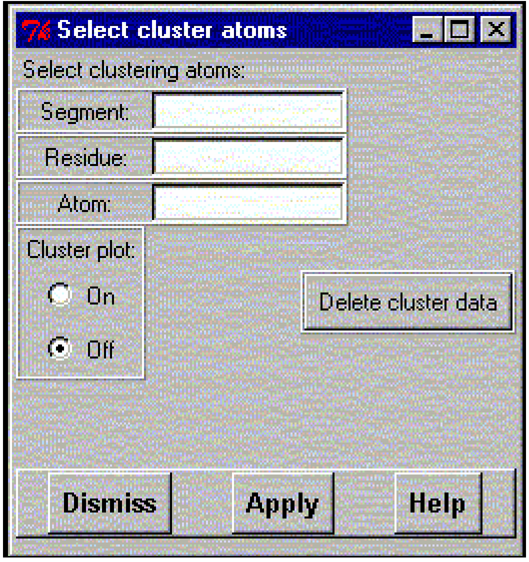
Lum: 240 Blue: 255

Line command: see **atom command**

(5) CALCULATE menu has the options:

Calculate cluster widget command

Calculate for display a cluster matrix displaying the root mean square (RMS) value, after a superimpose of the selected atoms over the frames in a trajectory.
Select the segment, residue and atoms defining the structure to be superimposed with the equivalent part in the other frames. After selecting the atoms press the **Apply** button to perform the calculation over the frames.
After the calculation the matrix can be displayed by selecting the "Cluster plot on" option.
To free the reserved matrix space press the "Delete cluster data" button.

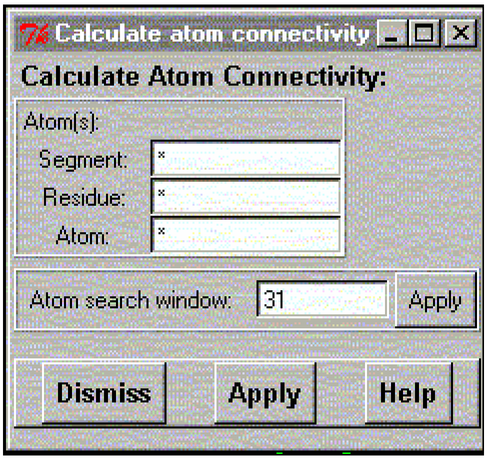


Line command option: **calculate command**

Calculate atom connection widget

This widget is helpful when one wants to recalculate atom connections (bonds) in case you have deleted some or if you don't get all hydrogen atoms bonded in your structure.
gOpenMol uses a very simple algorithm to calculate possible bonds. It runs over all the atoms in the structure and uses a window (range) of atoms inside which it thinks the bonded atoms are located. By default there is a range defined (use **show atom window** command to display the current value) so when gOpenMol looks at atom N it checks range atoms backwards and range atoms forward from atom N to find the bonded neighbours. Some programs place added hydrogens at the end of the file, which might prevent gOpenMol from finding the right bonds.
To solve this problem the range can be redefined. Please supply a new value, inside which you think all bonded atoms will be.

It is also possible to recalculate the connectivity only for a set of atoms.



Atom(s):
Segment: *
Residue: *
Atom: *

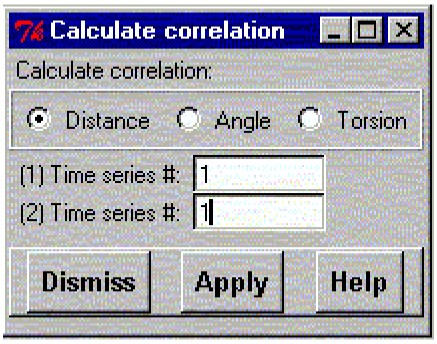
Atom search window: 31

Calculate correlation widget command

Calculate correlation and autocorrelation functions from two time series. There are three types of time series:

- Distance
- Angle
- Torsion angle

The time series fro these three possibilities are stored with a running number. Choose first type of time series and then the running number of the time series inside this group. If you choose the same time series for #1 and #2 you calculate the autocorrelation function.

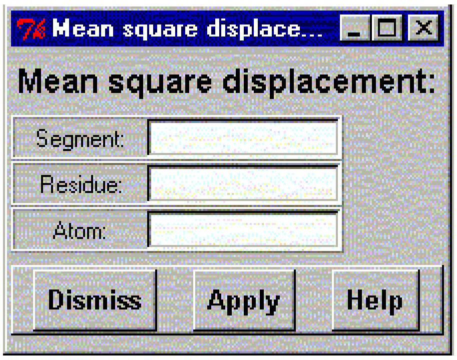


● Distance ○ Angle ○ Torsion

(1) Time series #: 1
(2) Time series #: 1

Line command: see **calculate command**

Calculate mean square displacement widget command

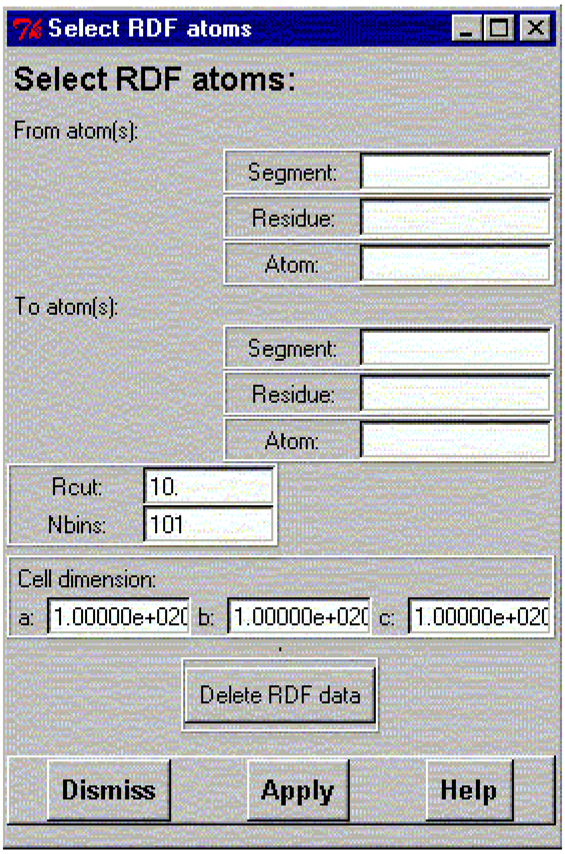


Line command: see **calculate command**

Calculate radial distribution function widget

It is possible to calculate a distribution function (g(r)) for an atom using gOpenMol from one frame or as an average from a sequence of frames like a molecular dynamics simulation. It is possible to arbitrarily choose the center atom and the atom around that.

If you have only one frame available (you have not a trajectory file defined) and you select the "Calculate RDF" from the "Calculate" pulldown menu you will get the following widget:



Rcut: 10.

Nbins: 101

Cell dimension:

a: 1.00000e+02(b: 1.00000e+02(c: 1.00000e+02(

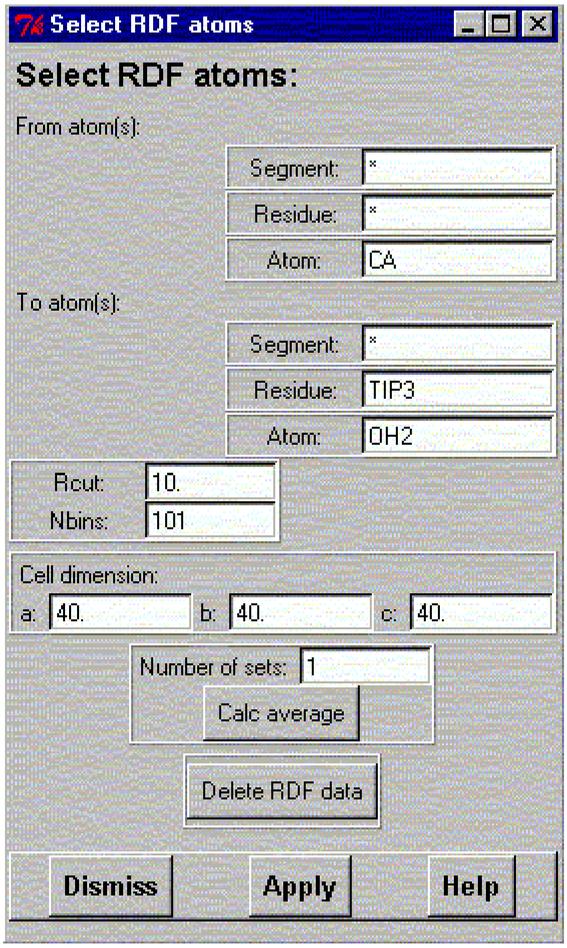
The "From atom(s):" define the center atom(s) around which the distribution will be calculated. The "To atom(s):" define the atom(s) which distribution will be calculated.

The "Rcut:" value defines the distance to which the distribution will be calculated and "Nbins:" define the number of intermediate values (bins) for which the distribution will be calulated.

If the periodic cell (most likely you have been using period boundaries), inside which your atoms and molecules are located, has not been defined before it has to be defined here.

Pressing now the "Apply" button calculates the radial distribution function of the "To atom(s)" around the "From atom(s)" for the structure displayed in the graphics window.

After pressing the "Apply" button" one gets the following widget:



Select RDF atoms

Select RDF atoms:

From atom(s):

Segment: *

Residue: *

Atom: CA

To atom(s):

Segment: *

Residue: TIP3

Atom: OH2

Rcut: 10.

Nbins: 101

Cell dimension:

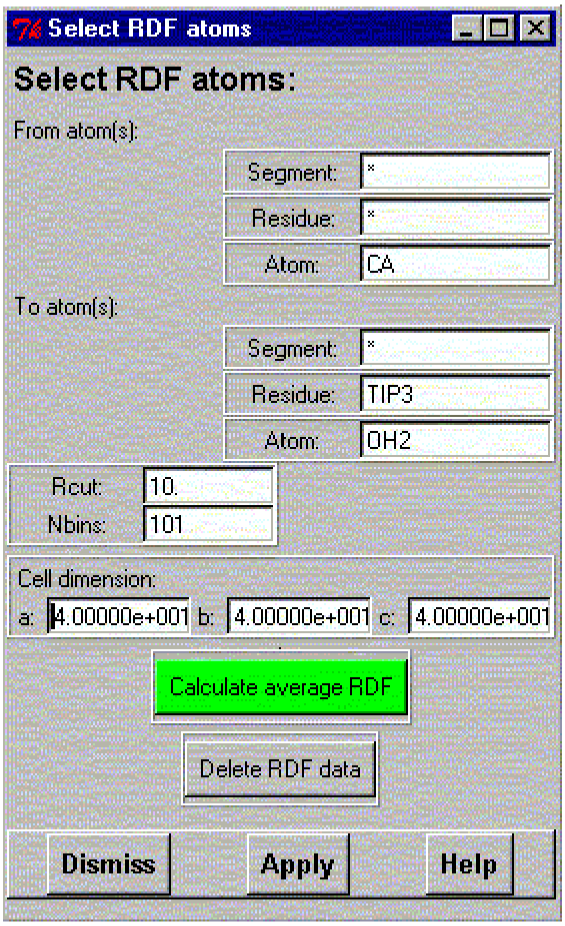
a: 40. b: 40. c: 40.

Number of sets: 1

There is now one furter entry and button displayed. An entry titled "Number of sets" defining for how many structures the RDF has been calculated and accumulated. It is thus possible to read by hand more frames into gOpenMol and successively calculate the RDF.

After the number of RDFs has been calculated (accumulated) it is possible to calculate the average for these RDFs by pressing the "calc average" button.

However, if a trajectory file has been defined **before** the "Calculate RDF" is selected from the "Calculate" pulldown menu one gets the following widget:



Select RDF atoms:

From atom(s):

Segment: *

Residue: *

Atom: CA

To atom(s):

Segment: *

Residue: TIP3

Atom: OH2

Rcut: 10.

Nbins: 101

Cell dimension:

a: 4.00000e+001 b: 4.00000e+001 c: 4.00000e+001

It is now possible to automaticly get the average calculated for the entire frame by pressing the "Calculate average RDF" button. gOpenMol goes now through your trajectory file frame by frame and calculates and accumulates the RDF. After the last frame an average RDF is calculated.

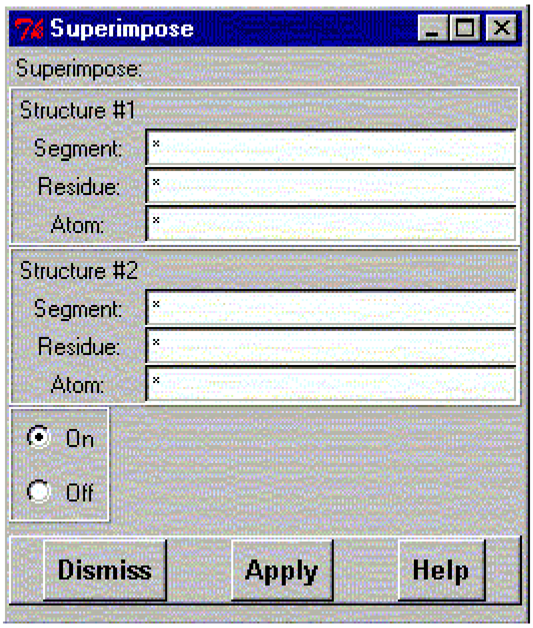
It is always possible to export the calculated (accumulated) RDF through the "export rdf" command or the "Export" pulldown menu. On the Windows platform it is also possible to copy the RDF to the Paste buffer through the "Edit" - "Copy" - "RDF" pulldown menu.

Line command: see **calculate command**

Superimpose structures widget command

Superimpose two structures by giving the atoms that should match. The number of atoms in the two sets must match but they do not necessarily have to be the same atoms.

Give the atoms for the two structures and press the **Apply** button. It is possible to display a short or a long display list of the matching structures. Select the one you want.



Superimpose:

Structure #1

Segment: *

Residue: *

Atom: *

Structure #2

Segment: *

Residue: *

Atom: *

● On

○ Off

Line command: see **calculate command**

(6) PLOT menu has the options:

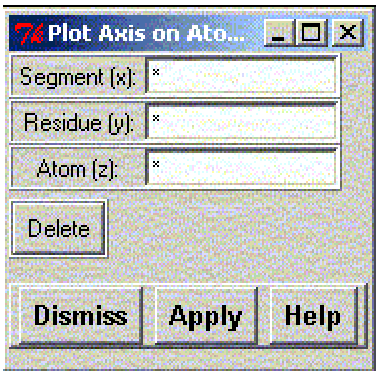
Plot axis widget

Using this widget it is possible to select atoms and plot the local coordinate system on the atoms. It is also possible to give the x, y and z coordinate where the local coordinate system is placed.

Give the atom(s) and press the "Apply" button.

The selection is additive which means that the new set is added to the old set.

The whole selection can be deleted by pressing the "Delete" button.



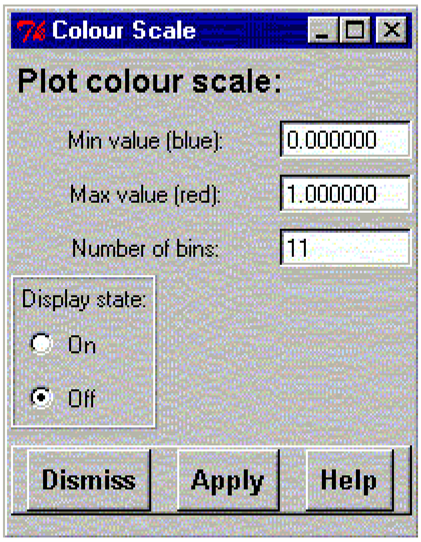
Line command: see **plot command**

Plot colour scale widget

It is possible to include into pictures, using colours to to display a range of values, a colour scale.

If a molecular dynamics trajectory is defined it is possible to plot the Root Mean Square Deviation (RMSD) of a selected set of atoms as ellipsoids. The plot is always in the global x, y, z coordinate system.

Please supply the min and max values and the number of values (bins) for the scale. The scale can be toggled between the on (display) and off (do not display) states.


Plot colour scale:

Min value (blue): 0.000000

Max value (red): 1.000000

Number of bins: 11

Display state:
○ On
● Off

Line command: see **plot command**

Contour Manager Widget

This is the main contour display widget. Through this widget you can control the contour files, isosurface values, colours, surface smoothing and many more things. The data is defined as scalar values in a grid box. Before you can display any isosurface contour you have to read in the molecular system for which the grid data has been generated (this is usually a coordinate data file).

The steps to do:

1. Read the atom coordinate information (through import coordinates) for your molecular system.
2. Open the contour manager widget (if it is not open alredy).
3. The contour data files are usually *.plt files (you can of course use your own file name extension). Click on the "Browse" button to open the file browser to see the *.plt files. Double click on the file (or a single click and then press the "Open" button) to select the file.
4. Press the "Import file" button to read in the grid data to gOpenMol.
5. Write the isocontour value into the input field (starting upwards).
6. Click on the "Colour(number)" button to choose a colour for the surface
7. If you want to map (colour) the surface according to the grid values of an other grid (*.plt) file press the "Mapping" button.
8. The contour surface renderer can be used in two different modes:
    - **direct** (fast) mode where no contour polygon data information is saved or
    - **save** (slow) mode where the last displayed contour polygon data information is save in an array and can be retrieved with the **show command**.
9. Press the "Apply" button"!

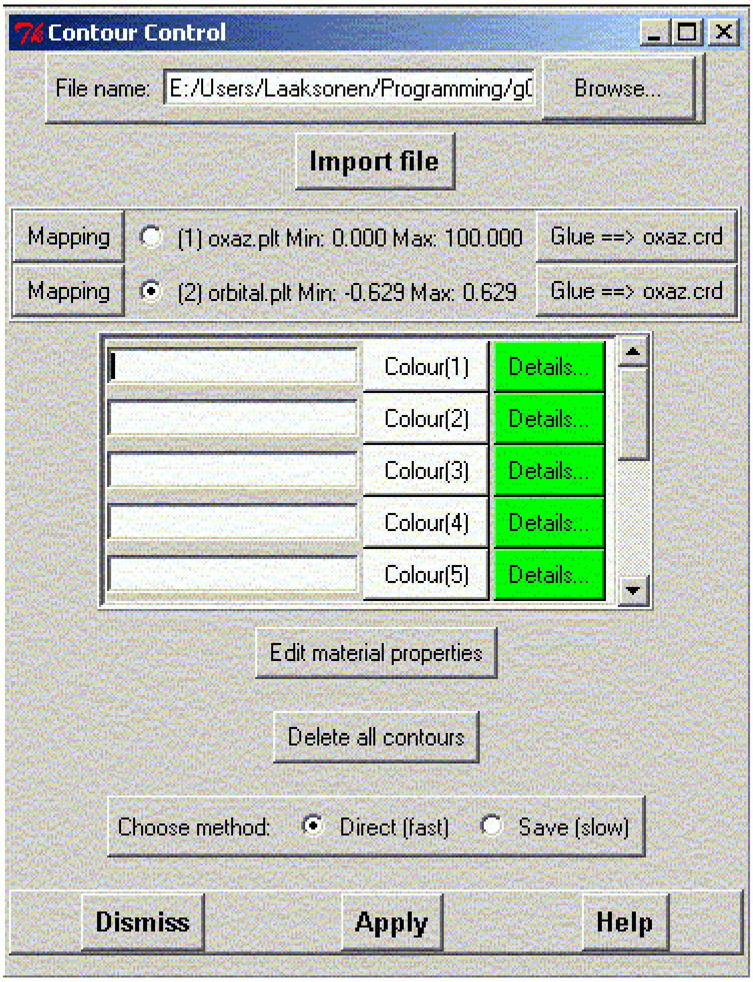
Surface details:

- If you want the surface to be smoothed press the "Details" button.
- If you want to make the surface transparent press the "Details" button.
- If you want to switch between mesh and solid type of surfaces press the "Details" button.

- If you want to turn a particular surface off press the "Details" button

You can change either the isocontour value, colour, smoothing by pressing the "Details" button for the chosen isocontour value.

You can also manipulate your mesh data files with the "contman" program. With the "contman" program you can add or sustract two contour files (A = B + C) or you can make a smooth transition from one contour file to an other using A = w * B + (1 - w) C, where w is a mixing constant (0.0 ... 1.0). To have a look at the input to the "contman" program write "contman -h".



Contour Control

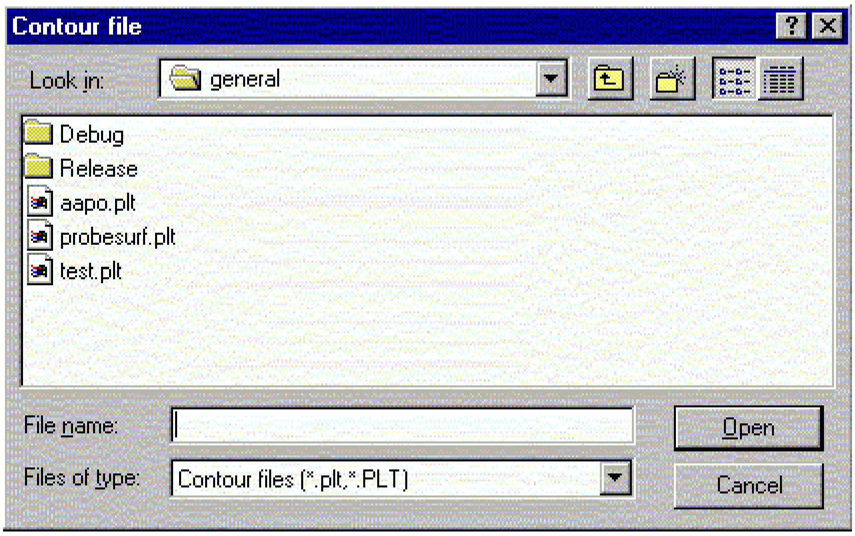
File name: E:/Users/Laaksonen/Programming/g(

○ [1] oxaz.plt Min: 0.000 Max: 100.000 — Glue ==> oxaz.crd

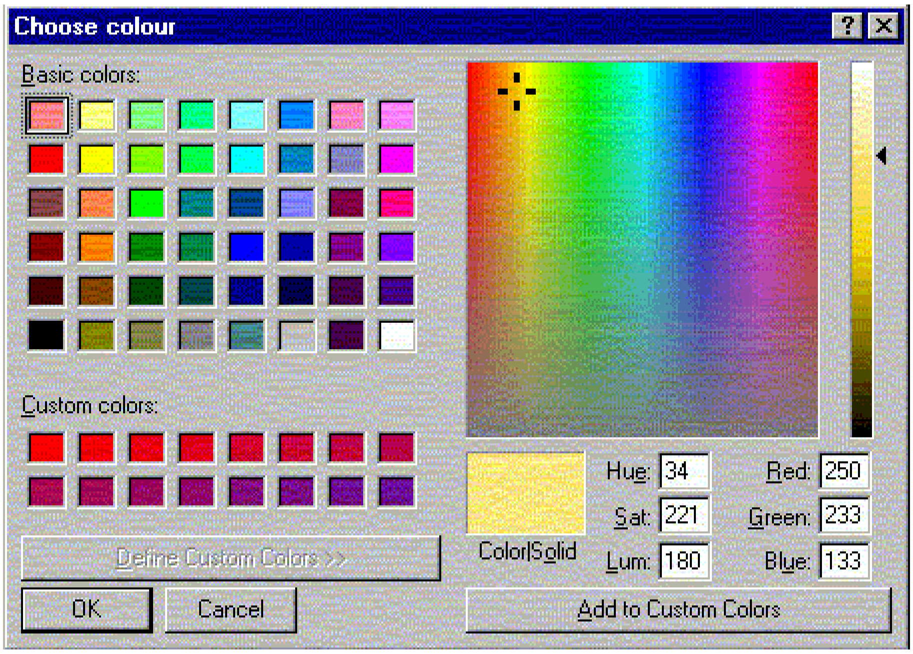
● [2] orbital.plt Min: -0.629 Max: 0.629 — Glue ==> oxaz.crd

Choose method: ● Direct (fast) ○ Save (slow)

File browser widget used to select a plot (*.plt) file:

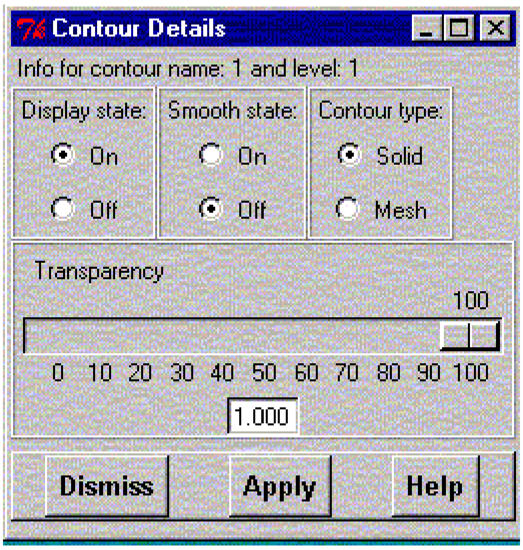


Widget to select the colour for a surface:



Widget to change detailed contour parameters:

- Click to set the display state, smoothing state, contour type (mesh/solid).
- Change the transparency of the surface.
- **Remember to press the "Apply" button to apply the selection.**



Info for contour name: 1 and level: 1

Display state: ● On ○ Off

Smooth state: ○ On ● Off

Contour type: ● Solid ○ Mesh

Transparency 100

0 10 20 30 40 50 60 70 80 90 100

1.000

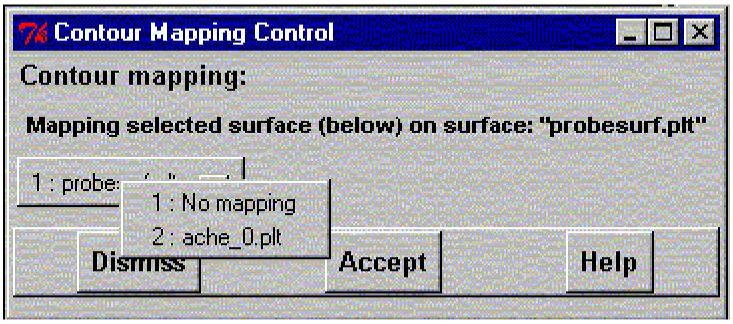
Line command: see **contour command**

****

Contour Mapping Widget
Leif Laaksonen CSC 1998

****

This is the contour mapping widget. Through this widget you can control the mapping (colouring) of an isocontour surface according to the grid values of an other grid (*.plt) file. The colouring can be scaled according to any grid value range.



Contour mapping:

Mapping selected surface (below) on surface: "probesurf.plt"

The steps to do:

1. Read the atom coordinate information (through import coordinates) for your molecular system.

2. Open the contour manager widget (if it is not open alredy).

3. The contour data files are usually *.plt files (you can of course use your own file name extension). Click on the "Browse" button to open the file browser to see the *.plt files. Double click on the file (or a single click and then press the "Open" button) to select the file.

4. Press the "Import file" button to read in the grid data to gOpenMol.
5. Repeat the previous steps to read at least one more grid (*.plt) files.
6. Click on the radio button for the file for which you want to generate the isocontour surface.
7. Click on the "Mapping" button to open the "Mapping control widget". This you already did because you got this help text.
8. Click on the option menu button to select the file from which the colouring (mapping) will be taken. Finish by pressing the "Accept" button.
9. Write the isocontour value into the input field (starting upwards). If you want to apply an other colour range than the default fill in on the same line the **min.** and **max.** values for the range. The colouring will be from min (blue) to max (red). If the values are filled from **max.** to **min.** the colouring will be the opposite.
10. Press the "Apply" button"!

Surface details:

- If you want the surface to be smoothed press the "Details" button.
- If you want to make the surface transparent press the "Details" button.
- If you want to switch between mesh and solid type of surfaces press the "Details" button.
- If you want to turn a particular surface off press the "Details" button

Please observe that the two grid files have to have the same number of grid points in the x, y and z directions and the grid data has to be defined in the same x, y and z space for the both grid files.

If you don't know the grid space you can run the format to unformat *.plt file converter (pltfile) in the bin/ directory.

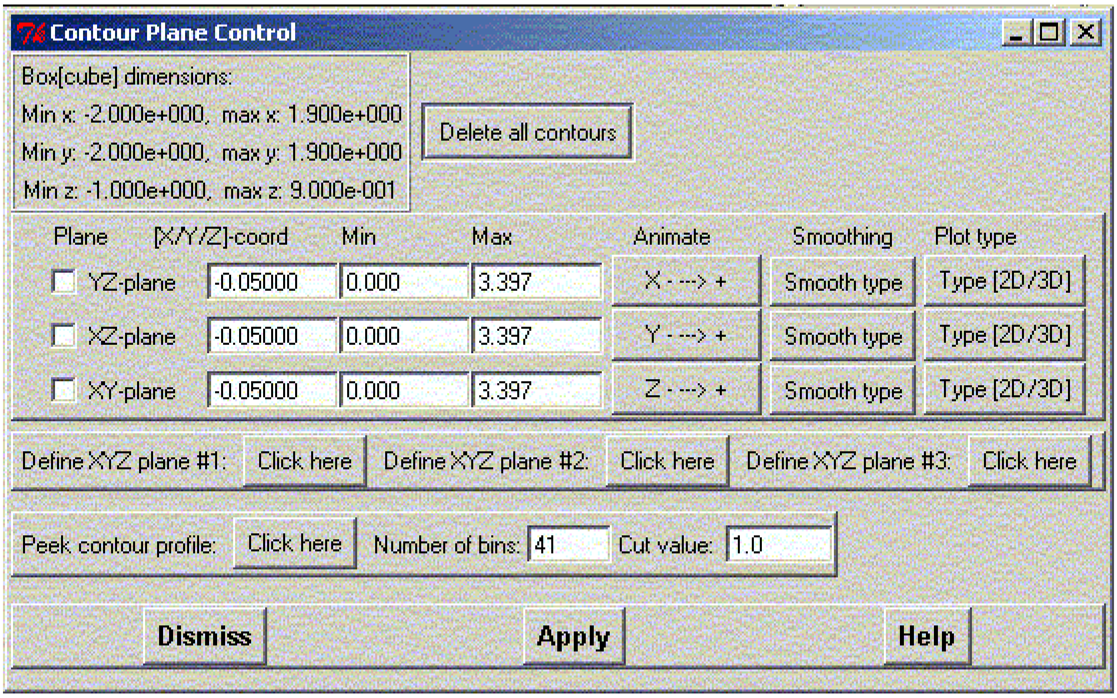
You can change either the isocontour value, colour, smoothing by pressing the "Details" button for the chosen isocontour value.

Contour Plane Manager Widget

Define a cutplane trough the contour surface grid data. The planes are defined to be perpendicular to the x, y and z axis.

It is possible to animate or move the planes "smoothly" using the animate facility behind the "X,Y and Z --->+" buttons. Press respective button to show the display along the X,Y or Z axis.

- Define first the plane (yz, xz or/and xy) by clicking the selection box.
- Then define the coordinate of the x, y or/and z axis through which the plane will go.
- Then define the minimum and maximum value of the range.
- Later there will also be a feature to smooth the range.



Contour Plane Control

Box[cube] dimensions:
Min x: -2.000e+000, max x: 1.900e+000
Min y: -2.000e+000, max y: 1.900e+000
Min z: -1.000e+000, max z: 9.000e-001

Plane	[X/Y/Z]-coord	Min	Max	Animate	Smoothing	Plot type
□ YZ-plane	-0.05000	0.000	3.397	X - ---> +	Smooth type	Type [2D/3D]
□ XZ-plane	-0.05000	0.000	3.397	Y - ---> +	Smooth type	Type [2D/3D]
□ XY-plane	-0.05000	0.000	3.397	Z - ---> +	Smooth type	Type [2D/3D]

Define XYZ plane #1: Click here Define XYZ plane #2: Click here Define XYZ plane #3: Click here

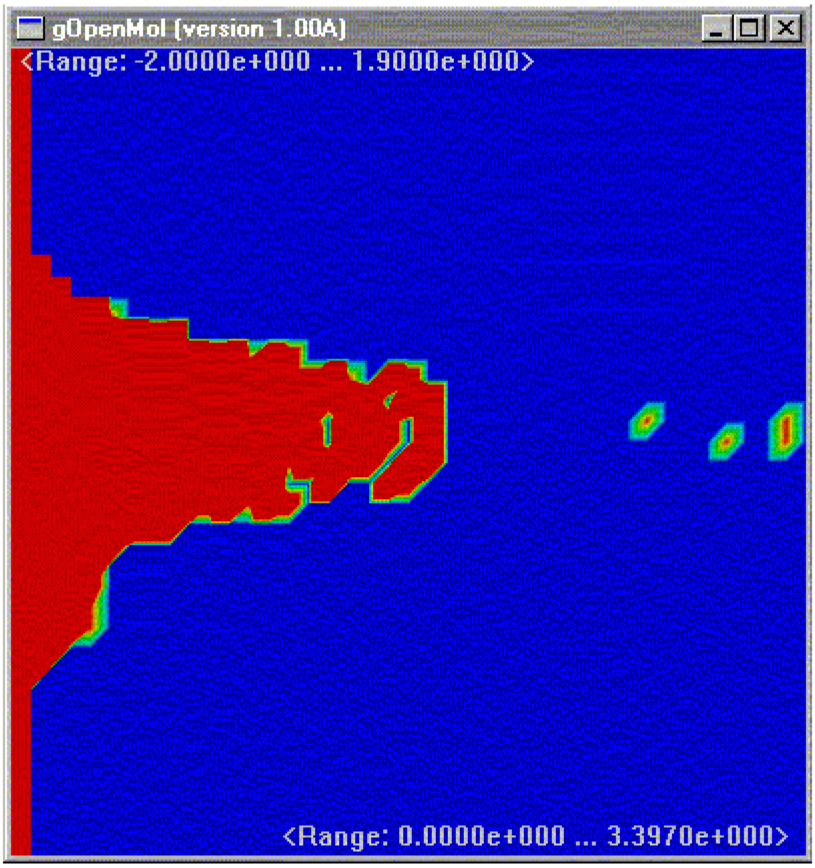
Peek contour profile: Click here Number of bins: 41 Cut value: 1.0

To delete all data click the "Delete all contours" button

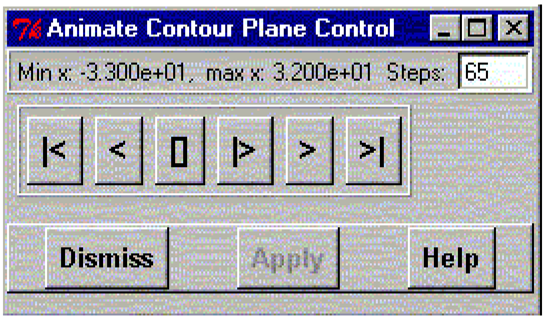
It is not always easy to find the best plane (=plain with most information). To make this procedure a bit easier a "profile graph" can be generated. The graph is generated by dividing the range of observations in bins where the grid data is then placed. A blue colour shows very few observations in that bin, while a read colour shows a high number of observations in the bin. The current picture applies a bin cut value of 1.0, which means that the region shown is the region where there is 1 or more values in the bin. This bin cut value can be changed by changing the default value 1.0 in the input widget to an other value. Using the value of 1.0 gives already a good insight into the grid space.

The horizontal-axis (in the graph) defines the min and max values of the grid data and the vertical-axis (in the graph) is the range inside which the plane can move along the chosen axis x-, y- or z-axis in the 3D space.

The following example shows that an YZ-plane can move along the x axis (in 3D) between the values -2.0 and 1.9 Angstrom, while the grid values varies between 0.0 and 3.397. From the graph one can read that most information is for a plane at x = 0.0 and a grid range of 0.0 to 1.6.


gOpenMol [version 1.00A]
<Range: -2.0000e+000 ... 1.9000e+000>
<Range: 0.0000e+000 ... 3.3970e+000>

The smooth or animate display:



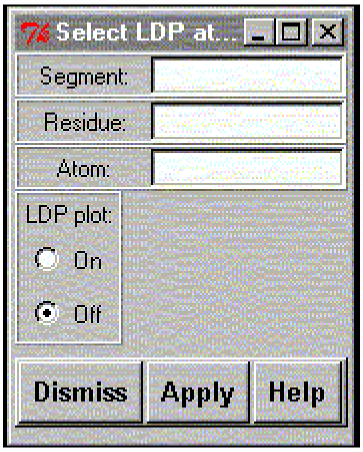
Min x: -3.300e+01 , max x: 3.200e+01 Steps: 65

Line command: see **plot command**

Linear distance plot (LDP) widget

Select the atoms to be include in the linear distance plot (LDP).

Select the atoms, set the LDP plot state to "ON" and press the apply button



Segment:

Residue:

Atom:

LDP plot: ○ On ● Off

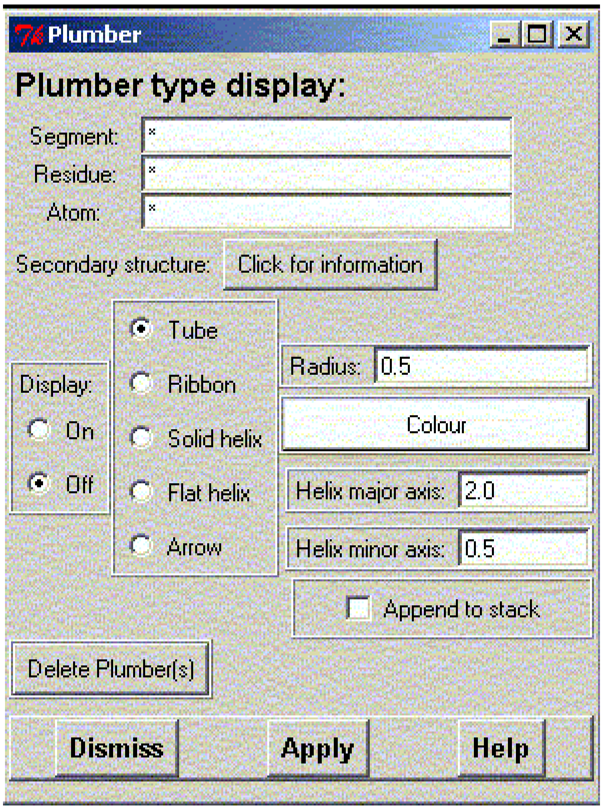
Line command: see **plot command**

Calculate and plot a tube or a ribbon through atoms widget

Select the atoms through which the ribbon/tube will go through. Then select the type Tube, Ribbon, Solid Helix, Flat Helix and Arrow, Colour and radius.

Several ribbons/tubes can be defined by selecting the "Append" mode.

The define display can be displayed by setting the Display mode "On".



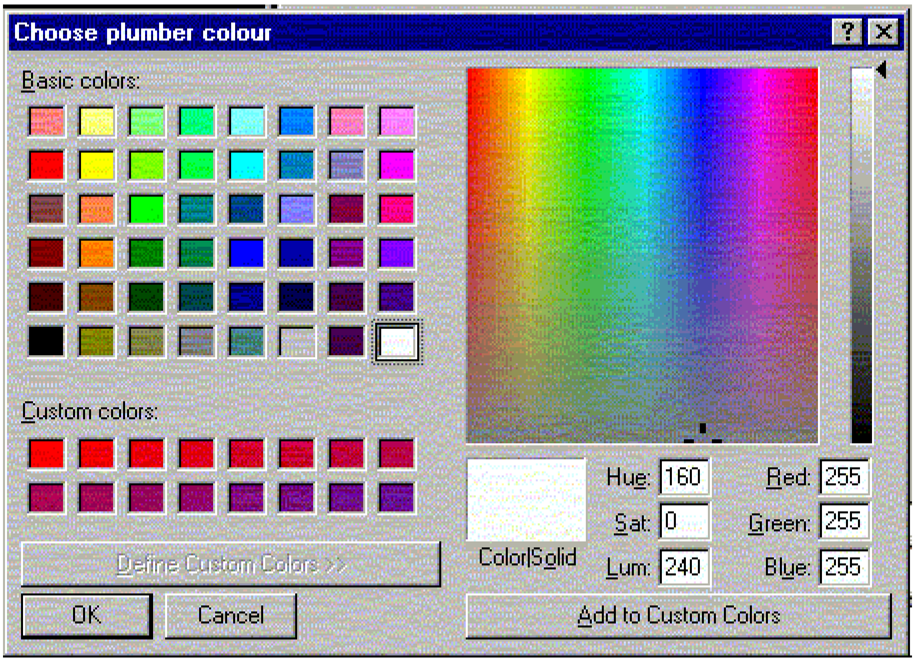
Segment: *
Residue: *
Atom: *

Display: ○ On ● Off

● Tube
○ Ribbon
○ Solid helix
○ Flat helix
○ Arrow

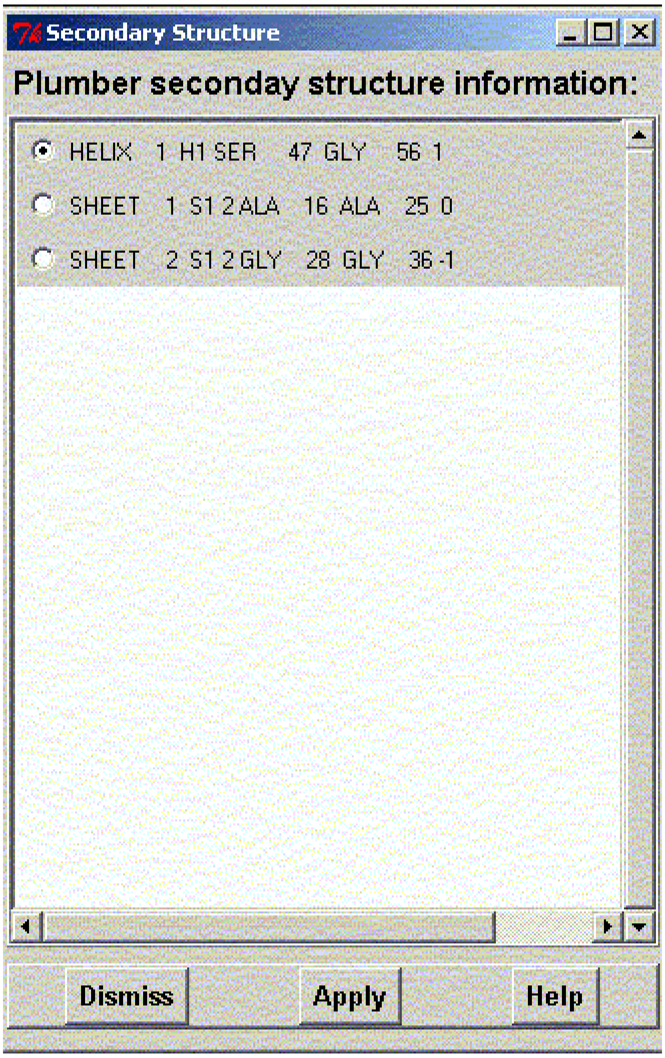
Radius: 0.5
Helix major axis: 2.0
Helix minor axis: 0.5
□ Append to stack

The tube/ribbon colour can be selected from the colour menu.



Hue: 160
Sat: 0
Lum: 240
Red: 255
Green: 255
Blue: 255

If the PDB coordinate file contain secondary structure information gOpenMol can use that information to define the objects. Select the type from the widget and click "Apply" button and the information is moved to the main Plumber widget. The only thing the user has to define is the colour.



Plumber seconday structure information:

● HELIX 1 H1 SER 47 GLY 56 1

○ SHEET 1 S1 2 ALA 16 ALA 25 0

○ SHEET 2 S1 2 GLY 28 GLY 36 -1

Line command: see **plumber command**

Calculate and plot the RMSD widget

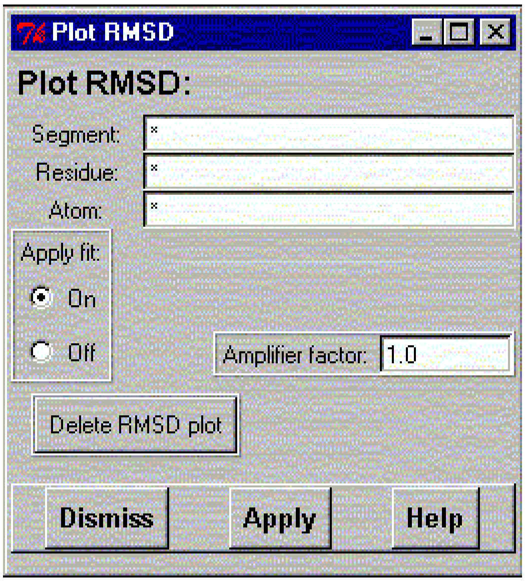
When a molecular dynamics trajector is defined it is possible to plot the Root Mean Square Deviation (RMSD) of a selected set of atoms as ellipsoids. The plot is always in the global x, y, z coordinate system.

Select the atoms and press the **Apply** button. There are two different approaches to generate the ellipsoids:

- The individual frames are **not** superimposed on the average structure (= nofit).

- The individual frames are superimposed on the average structure (= fit).

The plot can be deleted by pressing the **Delete** button. If the fluctuation has a small amplitude it is possible to magnify the amplitude by a factor.



Segment: *
Residue: *
Atom: *
● On
○ Off
Amplifier factor: 1.0

Line command: **p_rmsd Segment Residue Atom(s) Scale fit/nofit**

Plot vector widget

It is possible to plot vectors from a flat text file using this widget. Give first the name of the text file containg the vector data. It can come from a GaussianXX calculation or from an other program.

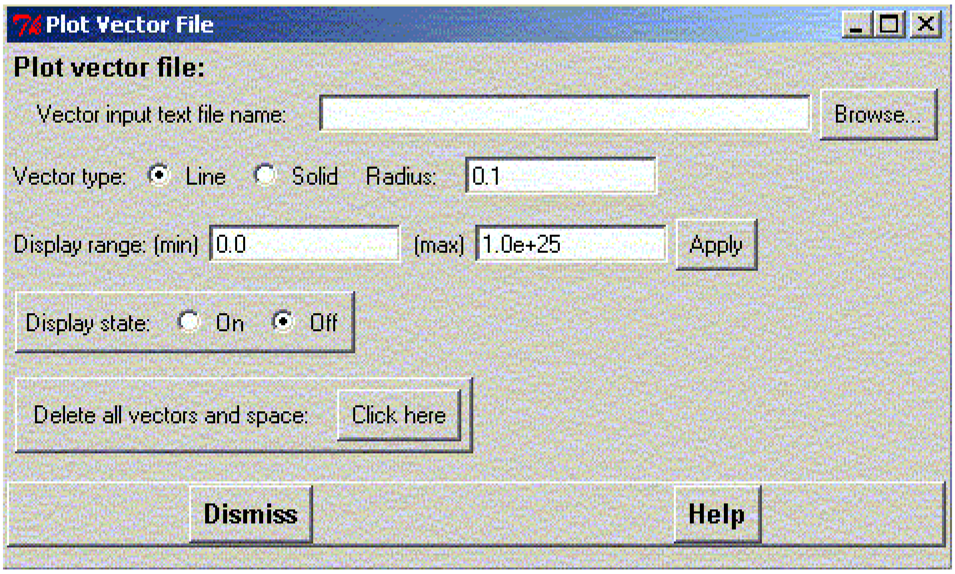
There are two different display styles available:

- Vectors are shown as a line drawing. This is the fastest display mode.
- Solid vectors with a defined radius. This is a slow display mode and should not be used for a display with many vectors.

It is also possible to give a vector length range that defines the range. Those vectors hanving a vector length inside this range will be displayed.

The vector display mode can be "On" or "Off".

If you don't need the display anymore free all the reserved vector space by clicking on the "Delete all vectord and space" button.



Plot Vector File

Plot vector file:

Vector input text file name:

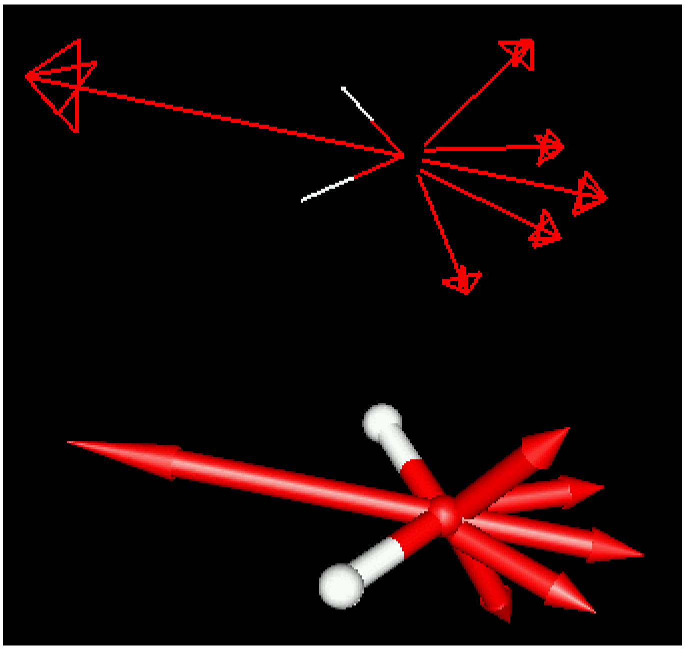
Vector type: ● Line ○ Solid Radius: 0.1

Display range: (min) 0.0 (max) 1.0e+25

Display state: ○ On ● Off

Delete all vectors and space:

Example for a water molecule with the electron gradient vectors with a length between 1.0 and 1.e^{+25} shown both as simple lines and solid vectors.



Line command: **plot command**

(7) TRAJECTORY menu has the options:

Monitor molecular geometry.

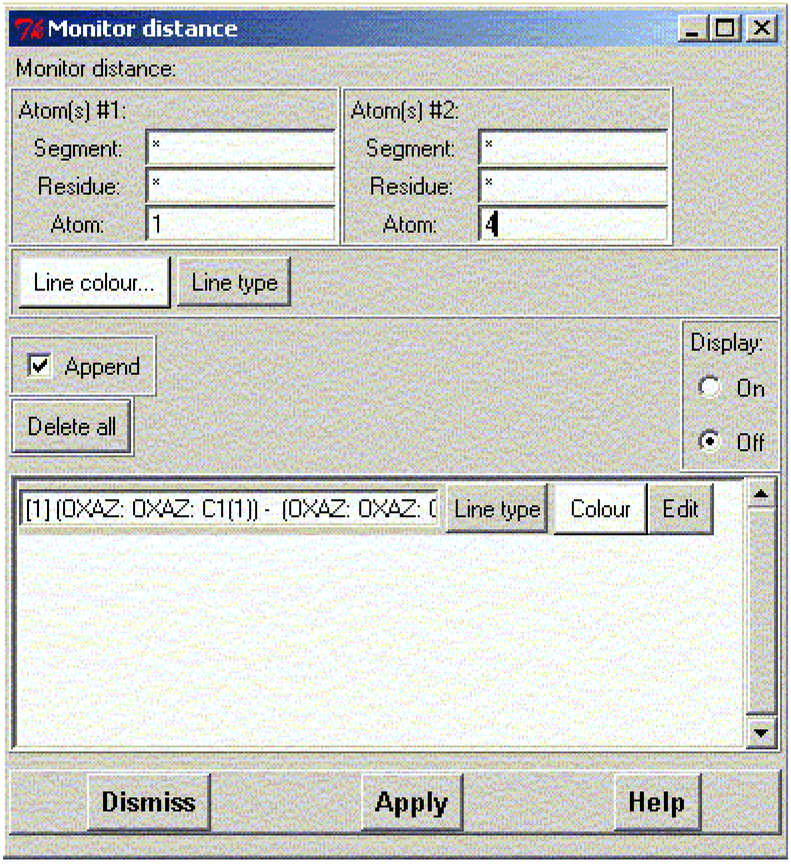
- **Distance(s)**
- **Angle(s)**
- **Torsion(s)**

Monitor Distance Widget

Select the atom in set #1 and #2 to define a distance, choose a line colour and click the apply button.

To display the selected distance put the display state to "On"

Press the "Delete all" button to delete all selected line segments



Monitor distance:

Atom(s) #1:
Segment: *
Residue: *
Atom: 1

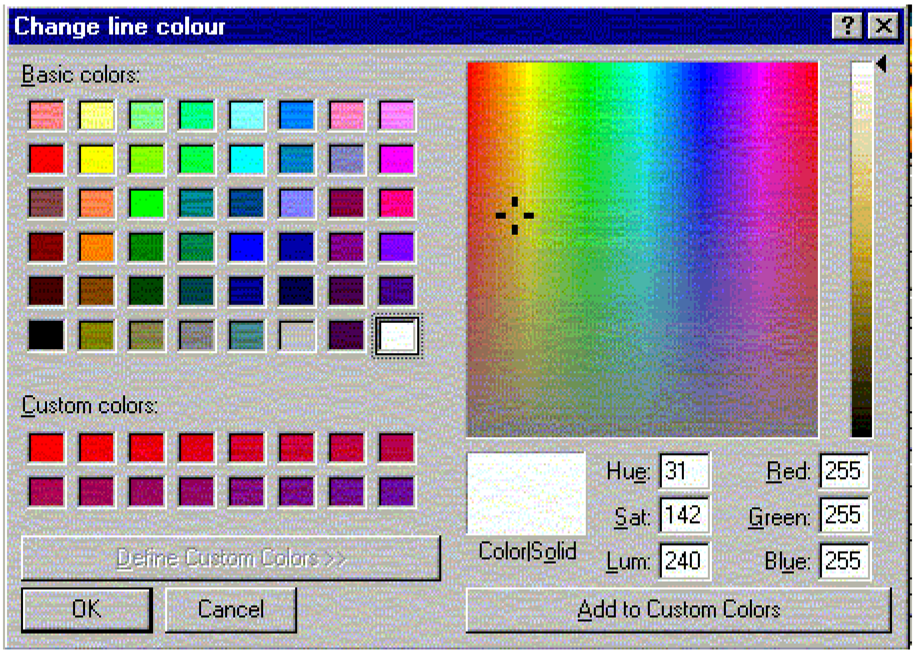
Atom(s) #2:
Segment: *
Residue: *
Atom: 4

☑ Append

Display:
○ On
● Off

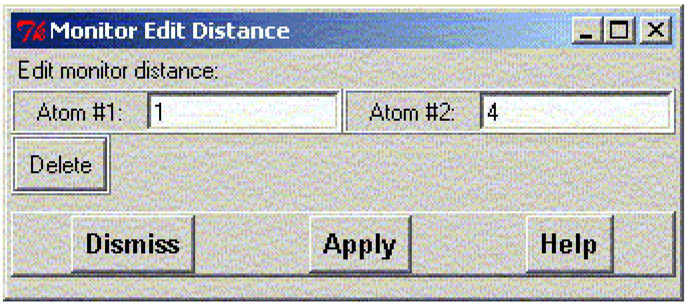
[1] (OXAZ: OXAZ: C1(1)) - (OXAZ: OXAZ: (

Choose the line colour from the colour menu.



Hue: 31
Sat: 142
Lum: 240
Red: 255
Green: 255
Blue: 255

To edit any of the defined sets press the "Edit" button. Now you can redefine the selected atoms or delete this particular defined distance.



Edit monitor distance:

Atom #1: 1
Atom #2: 4

Line command: see **monitor command**

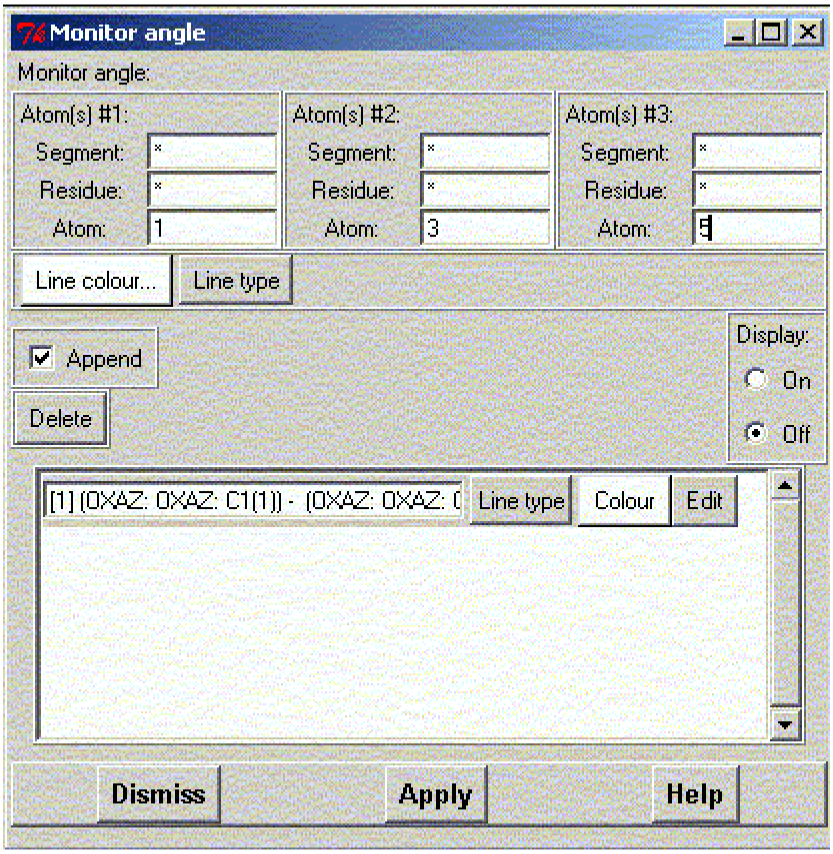
Monitor Angle Widget

Select the atom in set #1, #2 and #3 to define an angle, choose an angle colour and click the apply button.

To display the selected angle put the display state to "On"

Press the "Delete all" button to delete all selected angle segments

To edit any of the defined sets press the "Edit" button. Now you can redefine the selected atoms or delete this particular defined angle.



Monitor angle:

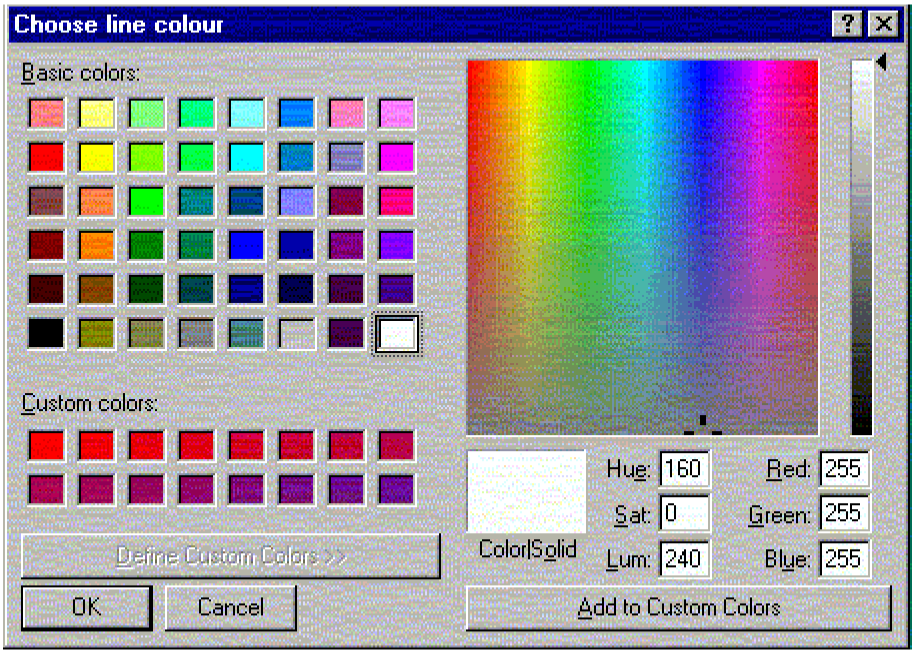
Atom(s) #1:		Atom(s) #2:		Atom(s) #3:	
Segment:	*	Segment:	*	Segment:	*
Residue:	*	Residue:	*	Residue:	*
Atom:	1	Atom:	3	Atom:	5

☑ Append

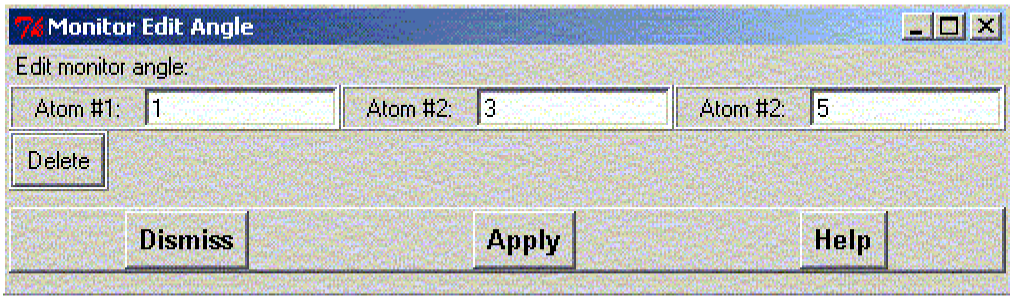
Display:
○ On
● Off

[1] (OXAZ: OXAZ: C1(1)) - (OXAZ: OXAZ: (

Choose the angle colour from the colour menu.



Hue: 160
Sat: 0
Lum: 240
Red: 255
Green: 255
Blue: 255


Monitor Edit Angle

Edit monitor angle:

Atom #1: 1 Atom #2: 3 Atom #2: 5

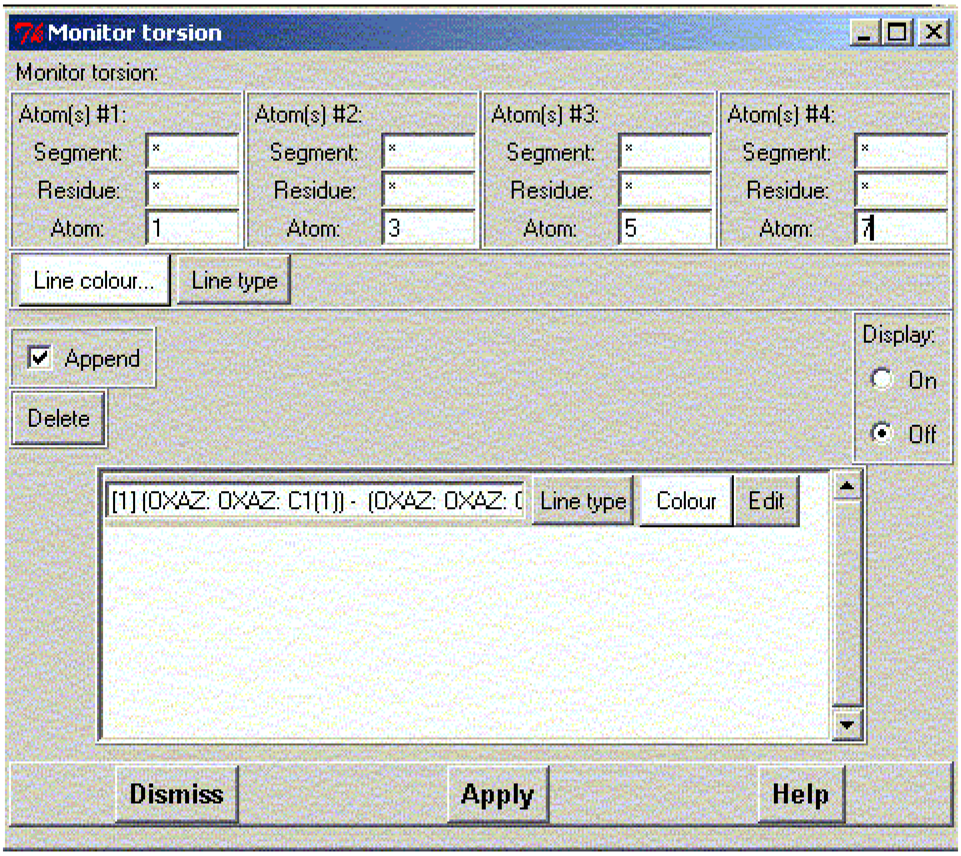
Line command: see **monitor command**

Monitor Torsion Widget

Select the atom in set #1, #2, #3 and #4 to define a torsion, choose a torsion colour and click the apply button.

To display the selected torsion put the display state to "On"

Press the "Delete all" button to delete all selected torsion segments


Monitor torsion

Monitor torsion:

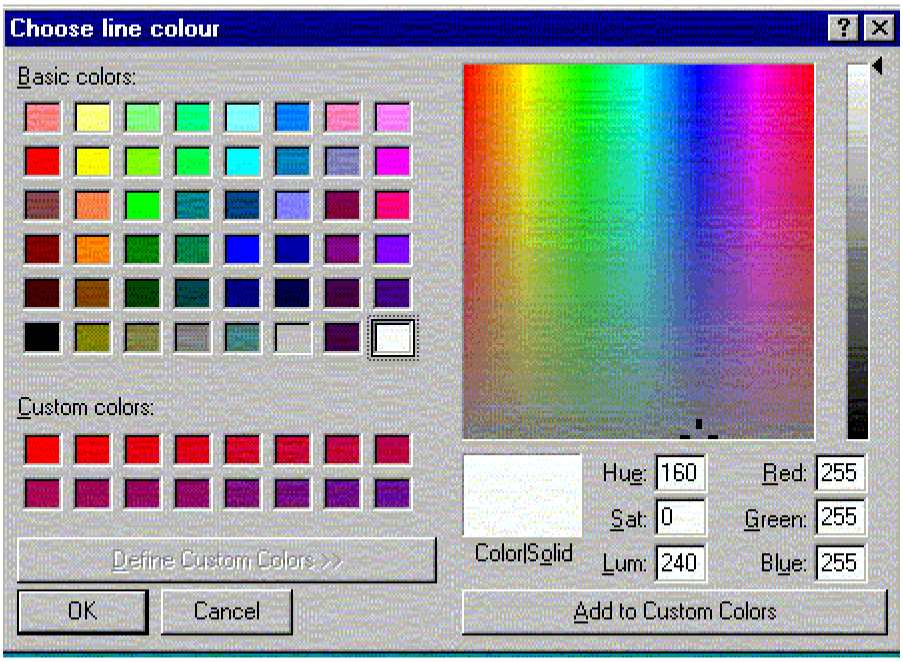
	Atom(s) #1:	Atom(s) #2:	Atom(s) #3:	Atom(s) #4:
Segment:	*	*	*	*
Residue:	*	*	*	*
Atom:	1	3	5	7

☑ Append

Display: ○ On ● Off

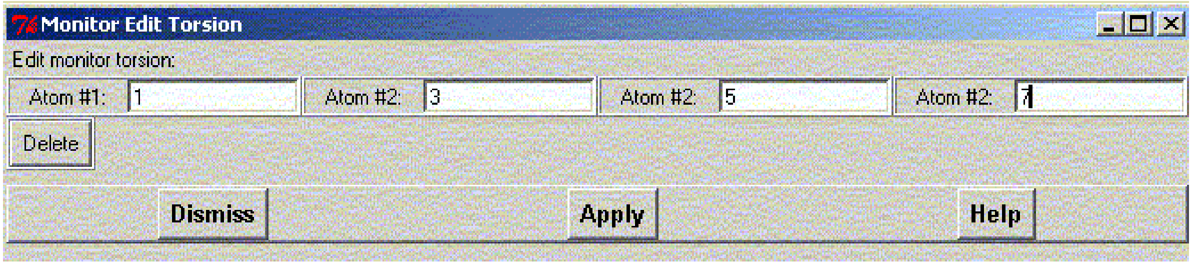
[1] (OXAZ: OXAZ: C1(1)) - (OXAZ: OXAZ: (

Choose the torsion colour from the colour menu.



Hue: 160
Sat: 0
Lum: 240
Red: 255
Green: 255
Blue: 255

To edit any of the defined sets press the "Edit" button. Now you can redefine the selected atoms or delete this particular defined torsion.



Edit monitor torsion:

Atom #1: 1
Atom #2: 3
Atom #2: 5
Atom #2: 7

Line command: see **monitor command**

Fill time series vectors.

- **Distance array**
- **Angle array**
- **Torsion array**

Fill distance array widget command

Fill the time series array according to the definitions given in the monitor distance widget.

Line command: see **fill command**

Fill angle array widget command

Fill the time series array according to the definitions given in the monitor angle widget.

Line command: see **fill command**

Fill torsion array widget command

Fill the time series array according to the definitions given in the monitor torsion widget.

Line command: see **fill command**

Delete time series vectors.

- **Distance array**
- **Angle array**
- **Torsion array**

Delete distance array widget command

Delete all stored distance array information and buffers defined in the monitor distance widget.

Delete angle array widget command

Delete all stored angle array information and buffers defined in the monitor angle widget.

Delete torsion array widget command

Delete all stored torsion array information and buffers defined in the monitor torsion widget.

Trajectory Manager Widget

This is the main trajectory control widget. Before you can process a trajectory file you have to import a coordinate file to define the molecule topology and the atom labels because some trajectories contain only atom coordinate information.

Steps to do:

1. Click of the button for the trajectory type you will analyze or look at. The buttons are on the right.
2. Select a file from using file browser (Browse button).
3. Press the "Import file" button to read the trajectory information.
4. When the file has properly been processes you can reed the number of frames in the file down on the page. There will also be display information about from which frame the display will start and at which frame it will stop and the frame step. Usually you don't have to touch this information att all.
5. To display the frames as loop starting from the first defined to the last click at the button with a triangle. To stop the display press the button with a square.

The supported trajectory formats are:

AMBER

> **Unformatted** Amber trajectory file. In the utility directory there is program for the conversion of a formatted trajector into a unformatted one. More information can be found from the AMBER Web pages http://www.amber.ucsf.edu/amber/formats.html.

AMBER(F)

> **Formatted** Amber trajectory file. In the utility directory there is program for the conversion of a formatted trajector into a unformatted one. More information can be found from the AMBER Web pages http://www.amber.ucsf.edu/amber/formats.html.

CERIUS2

> Unformatted binary CERIUS2 trajectory file.

CHARMM/CHARMm

> Unformatted binary CHARMM/CHARMm trajectory file.

DISCOVER

> Unformatted binary DISCOVER trajectory file.

GROMACS

> Unformatted GROMACS trajectory files (trn, trr and xtc).

GROMOS

> Unformatted GROMOS trajectory file.

GROMOS96

> Formatted ascii GROMOS96 trajectory file.

HYPERCHEM

Unformatted binary HyperChem trajectory file.

MUMOD

Unformatted binary MUMOD trajectory file. The MUMOD program is developed by Dr. Olle Teleman.

TINKER

Formatted TINKER trajectory file.

XMOL

Formatted ascii XMOL (multiset) trajectory file.

XPLOR

Unformatted binary XPLOR trajectory file.

YASP

Unformatted binary YASP trajectory file. The YASP program is developed by Dr. Florian Muller-Plathe (.

Trajectory display control:

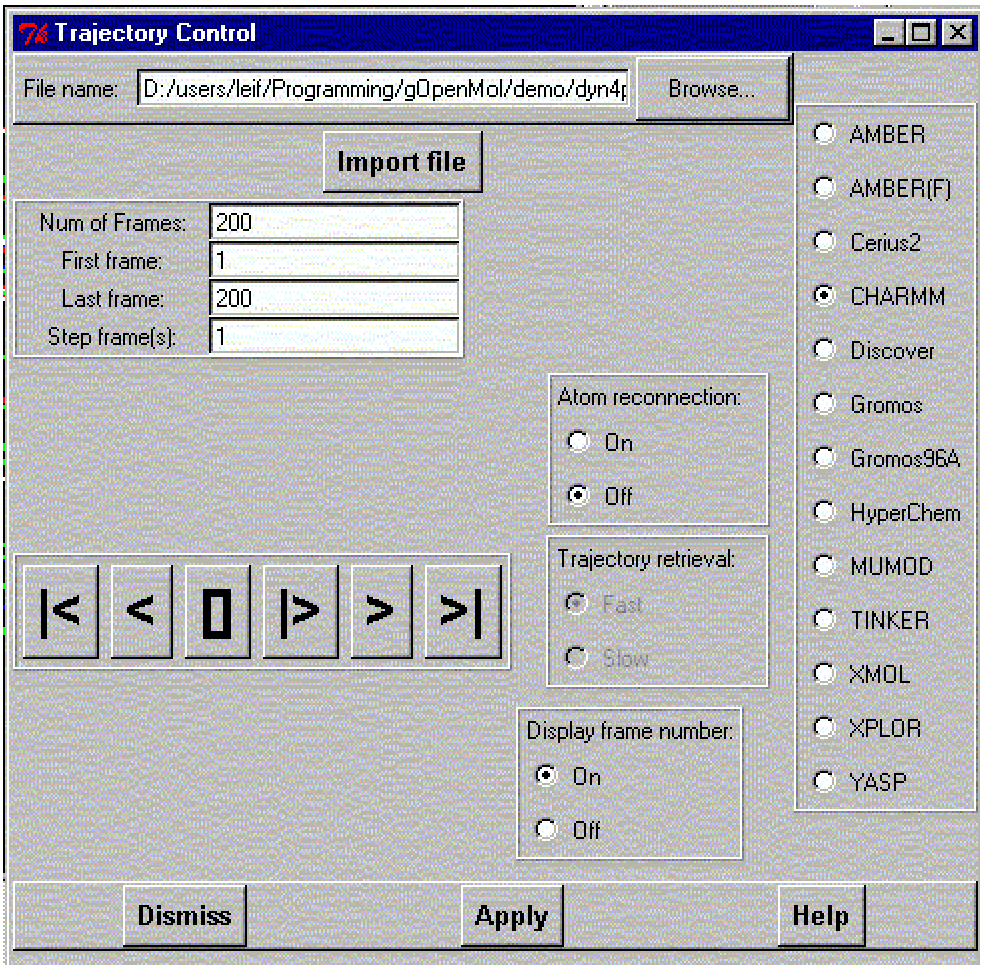
- **|<**... go to first frame
- **<**... back one frame
- **[]**... halt display loop
- **|>**... play forward one frame at the time in a loop
- **>**... forward one frame
- **>|**... go to last frame

By default the atom connection table is NOT recalculated in between the different frames. However, sometimes it's preferable to recalculate the connection table. This can be done by clicking the "Atom reconnection" to "ON".

For the formatted trajectory files there are two methods available:

- Sequential reading of frames. Very robust but quite inefficient and slow.
- Random access of the frames. Efficient but can cause problems, specially if you don't know that there is a different between Unix and Windows ascii files.

It's possible to display the frame number of the current frame. To display the frame number click "Display frame number: **on**" or **off** to turn it off.



Trajectory Control

File name: D:/users/leif/Programming/gOpenMol/demo/dyn4p

Num of Frames: 200
First frame: 1
Last frame: 200
Step frame(s): 1

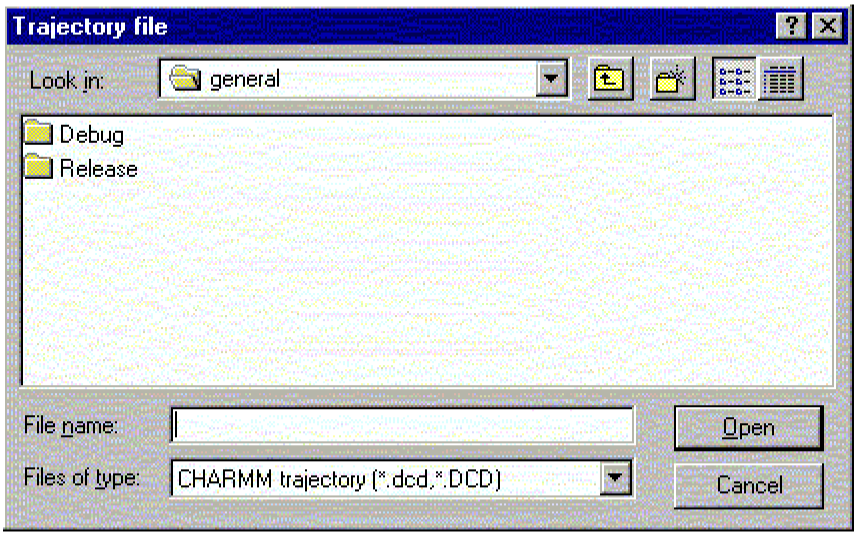
Atom reconnection:
○ On
● Off

Trajectory retrieval:
● Fast
○ Slow

Display frame number:
● On
○ Off

○ AMBER
○ AMBER(F)
○ Cerius2
● CHARMM
○ Discover
○ Gromos
○ Gromos96A
○ HyperChem
○ MUMOD
○ TINKER
○ XMOL
○ XPLOR
○ YASP

File browser to select a trajectory file.



Trajectory file

Look in: general

Debug
Release

File name:
Files of type: CHARMM trajectory (*.dcd,*.DCD)

Line command: see **trajectory command**

Manipulate Time Series Widget

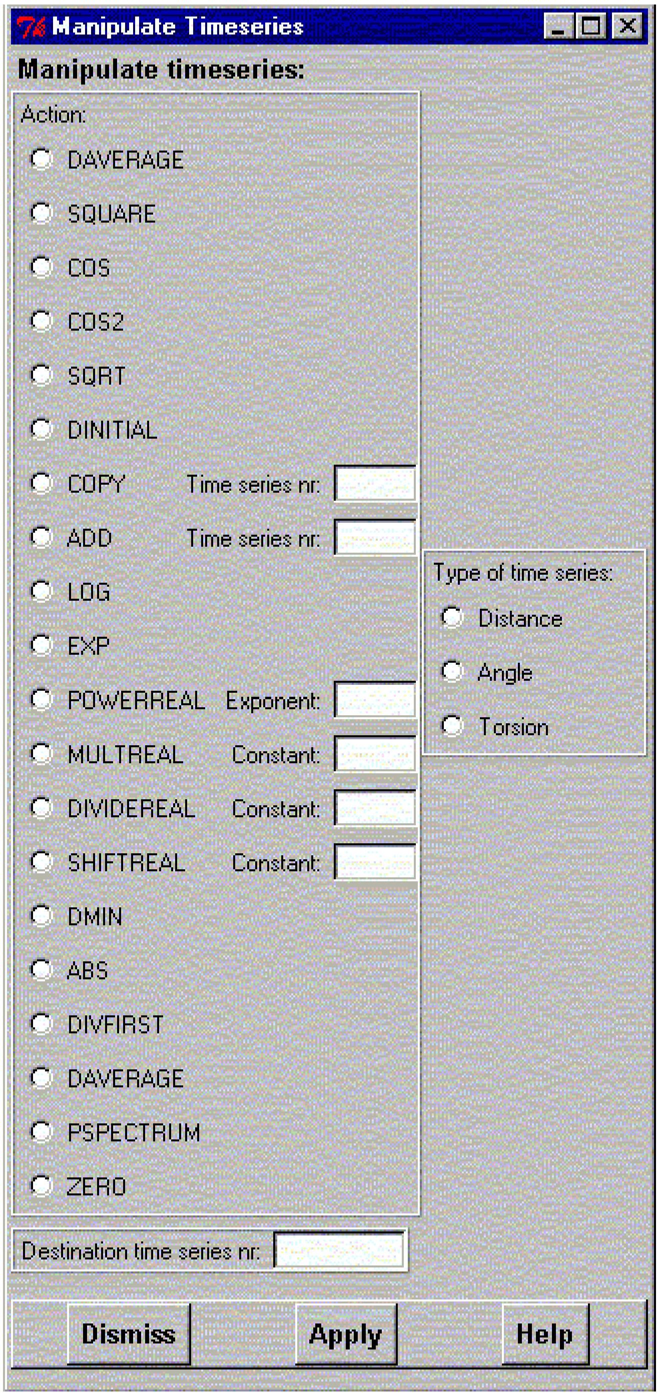
Manipulate the distance, angle and torsion time series through the following operations:

Types of operations (manipulations):

DAVErage ;	Q(t) = Q(t) -
SQUAre ;	Q(t) = Q(t) ** 2
COS ;	Q(t) = cos(Q(t))
COS2 ;	Q(t) = 3*cos(Q(t))**2 - 1
SQRT ;	Q(t) = sqrt(Q(t))
DINItial ;	Q(t) = Q(t) - Q(0)
COPY nr ;	Q(t) = Q2(t)
ADD nr ;	Q(t) = Q(t) + Q2(t)
LOG ;	Q(t) = log(Q(t))
EXP ;	Q(t) = exp(Q(t))
POWer real ;	Q(t) = Q(t) ** real
MULT real ;	Q(t) = real * Q(t)
DIVIde real ;	Q(t) = Q(t) / real
SHIFt real ;	Q(t) = Q(t) + real
DMIN ;	Q(t) = Q(t) - Q(min)
ABS ;	Q(t) = abs(Q(t))
DIVFirst ;	Q(t) = Q(t) / Q(0)
DIVMaximum ;	Q(t) = Q(t) /max(Q(t))
PSPEctrum nr ;	Power spectrum
ZERO ;	Q(t) = 0.0

Click on the radio button for the desired operation. If the operation requires a further value or index, please fill it in the input widget. Choose the distance, angle or torsion option.

Fill in the number index for the desired time series.



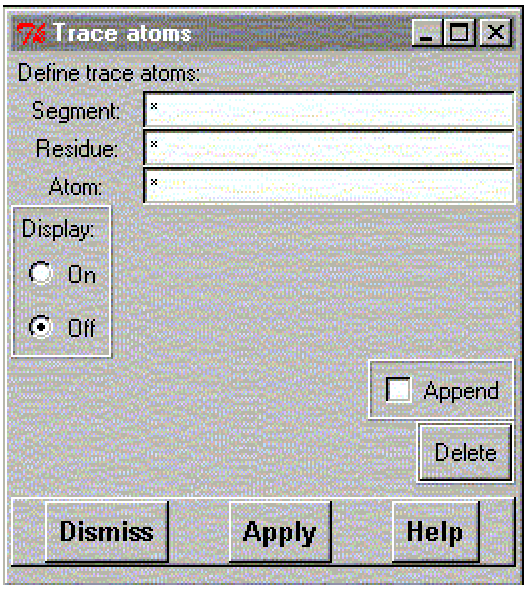
Line command: see **manipulate command**

Trace Atoms widget

Trace atom(s) or follow up the atom movement through your trajectory file.

Define first the atom(s) to trace and pres the "Apply" button. You can put the display state to "On" by clicking the "On" radio button.

The whole temporary space can be freed by pressing the "Delete" button.



Segment: *

Residue: *

Atom: *

Display: ○ On ● Off

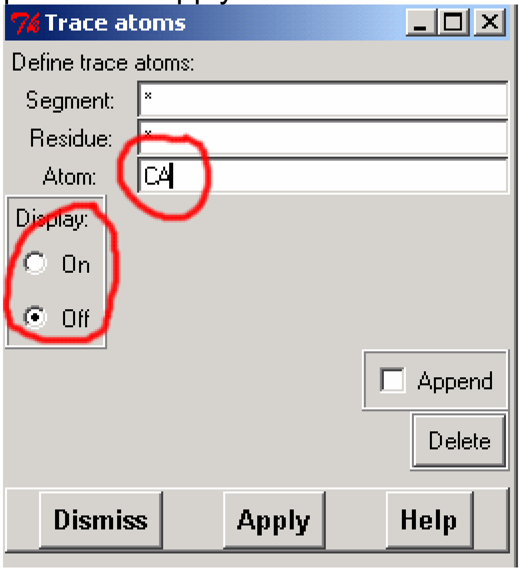
□ Append

Line command: see **trace command**

Trace example:

1. Import the opt4pti.crd file form the demo/ directory.
2. Import the dyn4pti.dcd trajectory file using the “Trajectory ➔ Main” widget.
3. Display only the back bone atoms with the command:
    a. **atom –display** ;# turn first all atoms off
    b. **atom display * * CA,N,C** ;# display the back bone

4. Open the “Trajectory ➔ Trace” widget and write into the “Atom:” field CA and press the “Apply” button.



Define trace atoms:

Segment: *

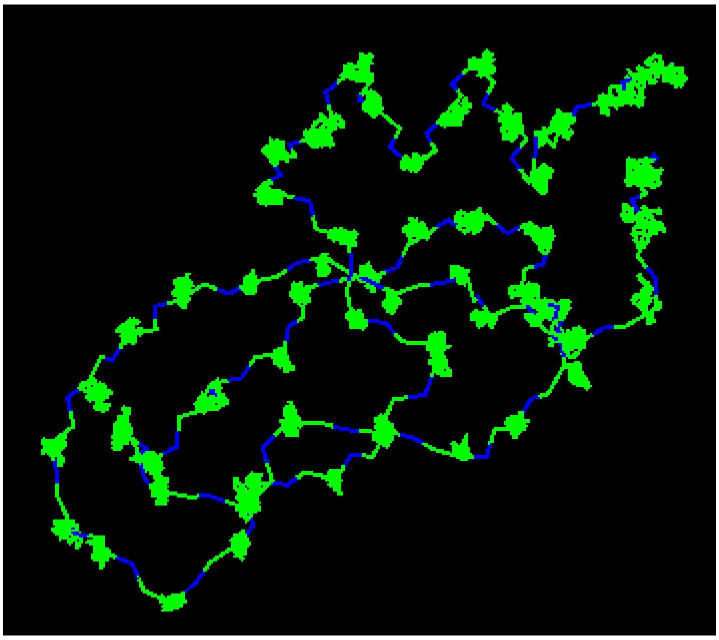
Residue: *

Atom: CA

Display:
○ On
● Off

□ Append

5. To display the trace for all CA atoms click the “Display:” to “On” and you should be able to see.

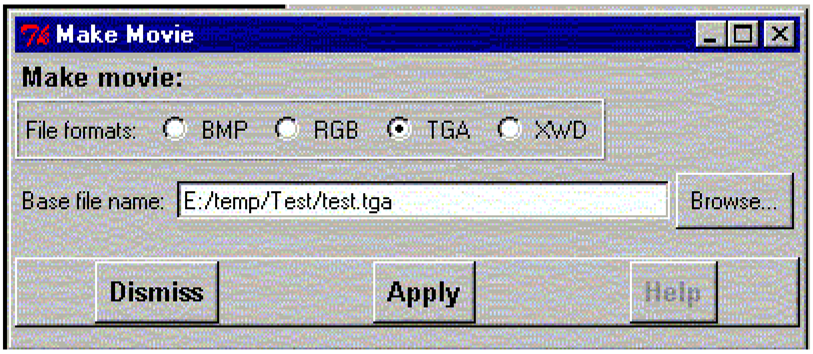


Make animation widget

The automatic creation of mpeg files works only on the Windows platform. On Unix you have to find your own program to create MPEG files from your separate frames.

Prepare animations by first producing discrete files and then make a mpg file from the files. Currently gOpenMol only supports Targa (tga) files but it is still possible to make the discrete screen dumps in other formats. Only small picture sizes can be used up to about 400 * 400 pixels with the program supplied on Windows.

- First choose the discrete file format.
- Choose then the basis file name. For example if a name **test.tga** is chosen the files will be named **test0.tga, test1.tga, test2.tga** ... By default the the files will be placed in the gOpenMol temp/ directory.
- Press the apply button and gOpenMol runs through the frames in your trajectory and makes a screendump for every frame. After this **test.mpg** will be produced.


Make movie:

File formats: ○ BMP ○ RGB ● TGA ○ XWD

Base file name: E:/temp/Test/test.tga

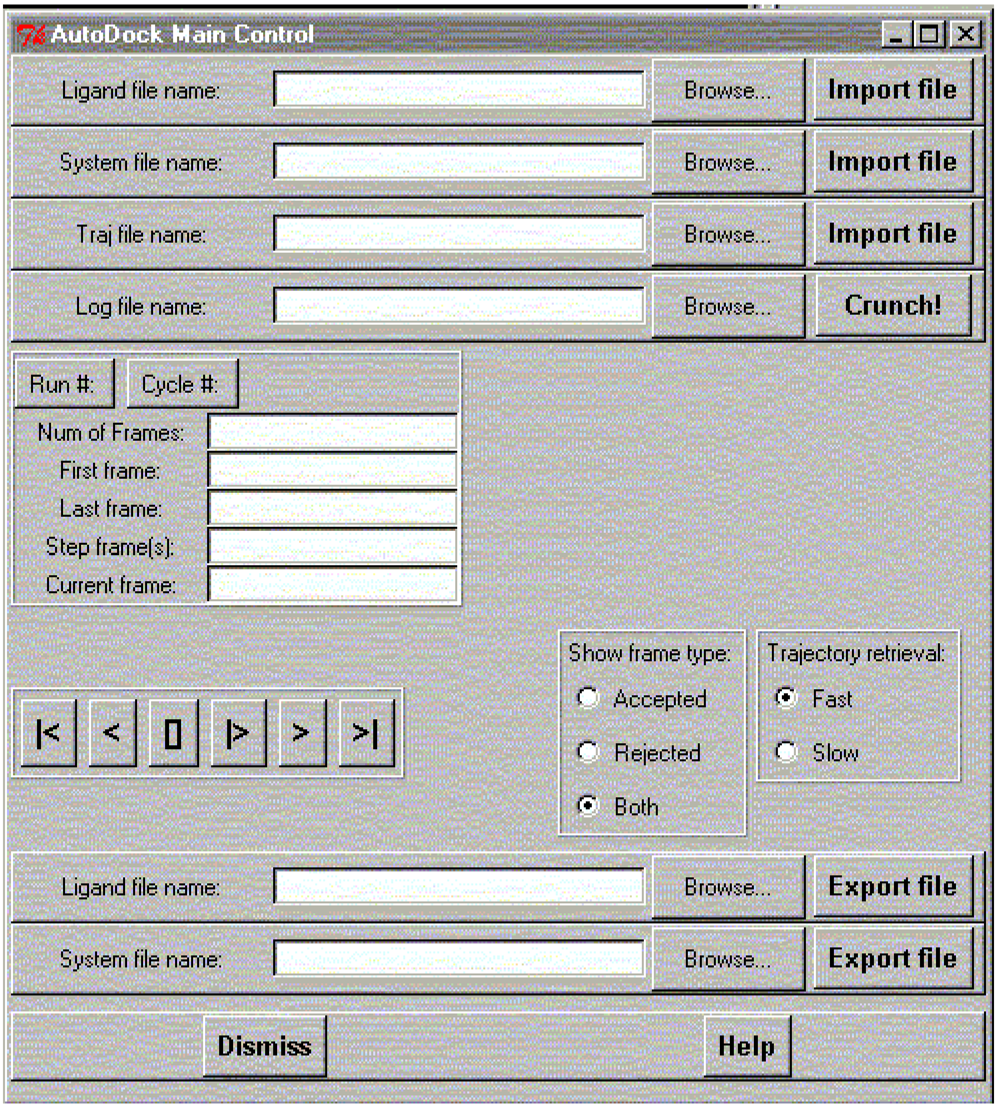
Line command: see **there is no line command for making animations**

(8) RUN menu has the options:

Run autodock2plt program widget

Run the AutoDock2plt program to convert a AutoDock map file into the plt file format used by gOpenMol. The AutoDock map file can then be displayed using the gOpenMol contour utilities

Write the input and output file names or click the **Browse** buttons to define the file names. The press the **Apply** button to do the conversion.



Show frame type:
○ Accepted
○ Rejected
● Both

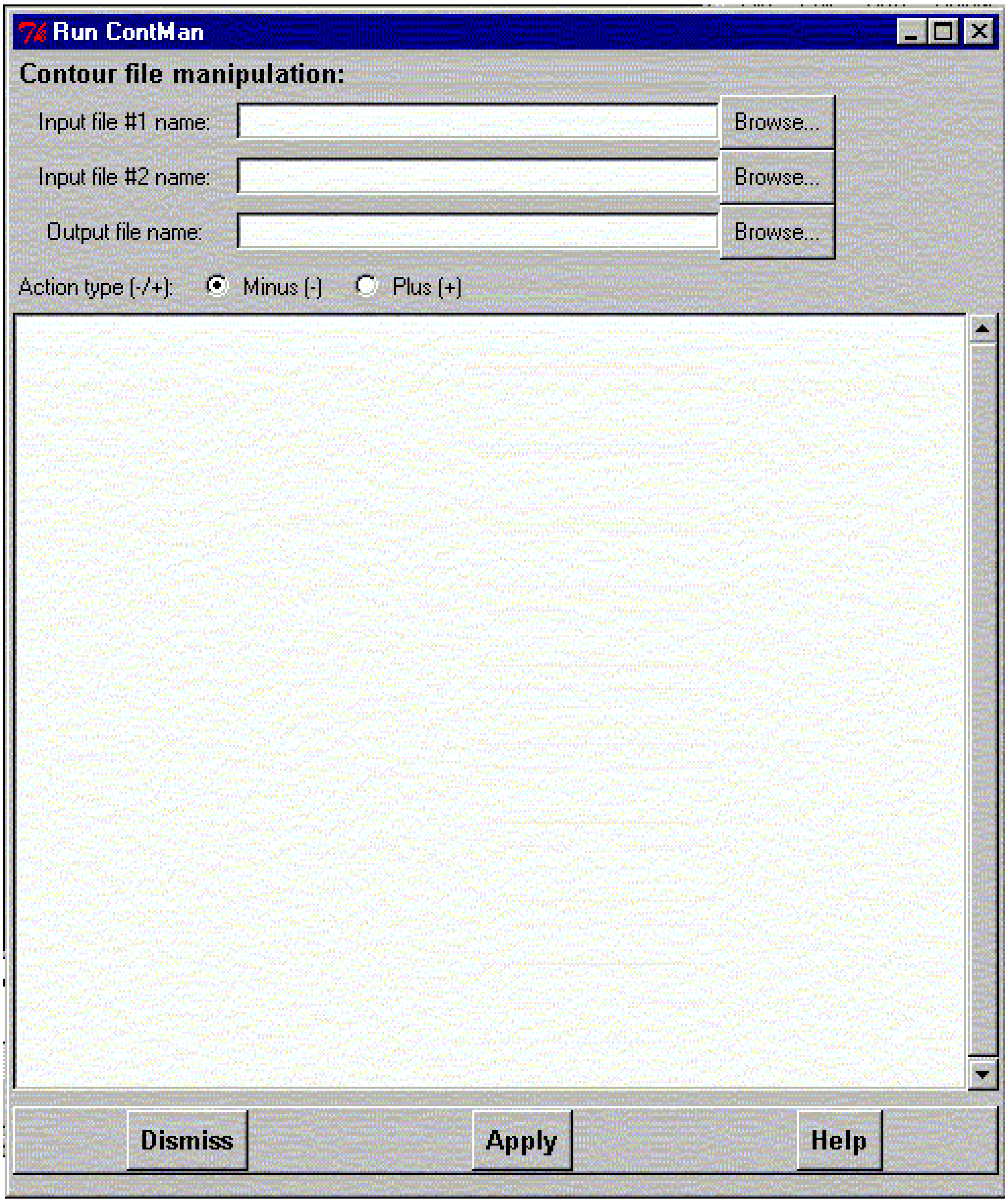
Trajectory retrieval:
● Fast
○ Slow

Line command: **can't be run from gOpenMol line mode. Use the Unix/Dos command line mode**

Run ContMan program widget

Run the ContMan to subract or add two gOpenMol plt grid mesh files and write the result out to a plt file format used by gOpenMol.

Give input1, input2 and output file names. Choose either **Minus (-)** or **Plus (+)** action. Click the **Apply** button to do the operation.



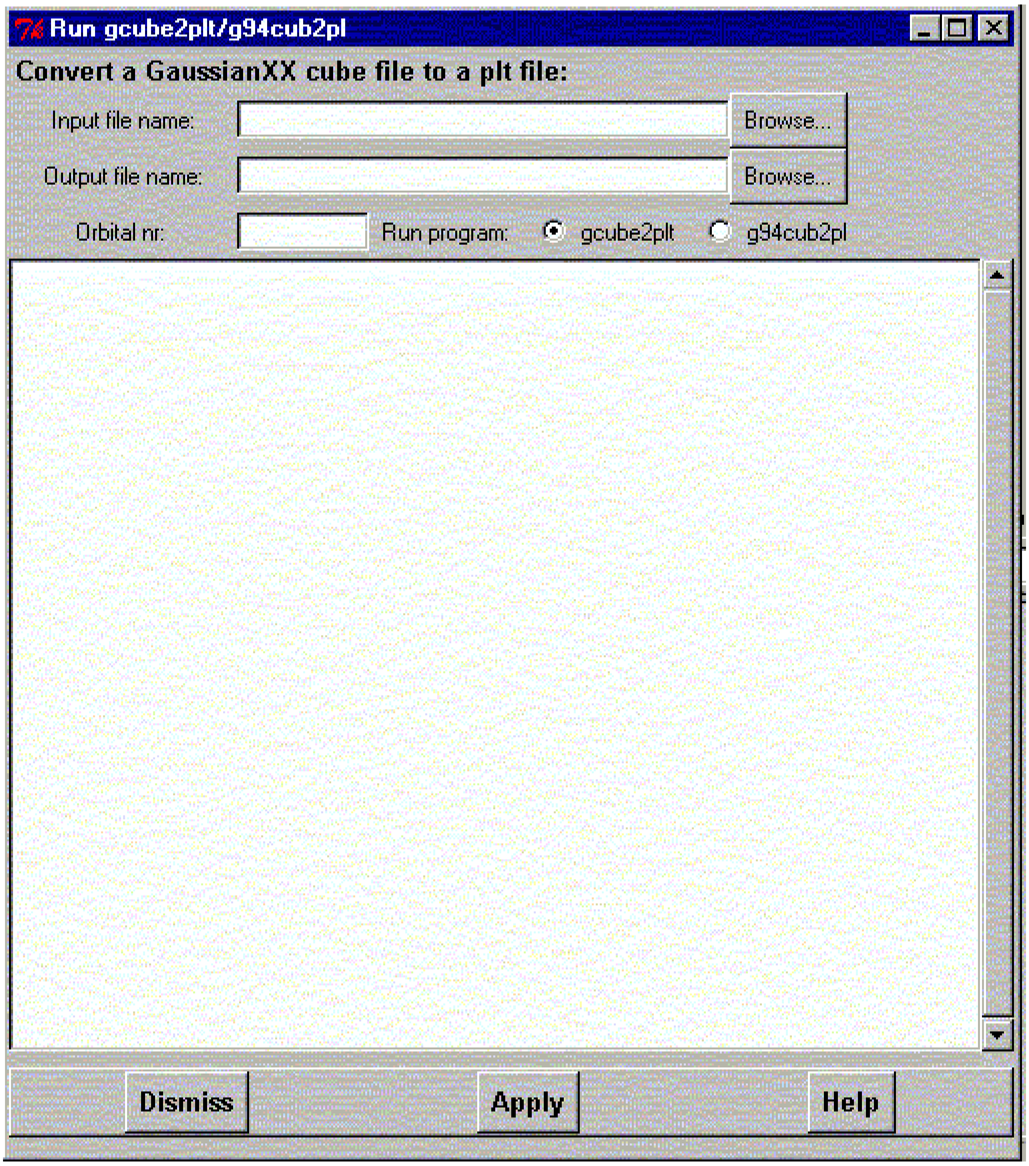
Line command: **no line command available inside gOpenMol. Use Unix/Dos shell line command.**

Run gCube2plt program widget

Run one of the two available programs to convert the Cube output from the GaussianXX program into the plt file format used by gOpenMol.

The programs available are:

- gcube2plt: the old version that converts only one orbital/density at a time. One needs to give the orbital number in the input field.
- g94cub2pl: a modified version made by Doug Moffatt that converts all the information in the file into separate files in one batch.

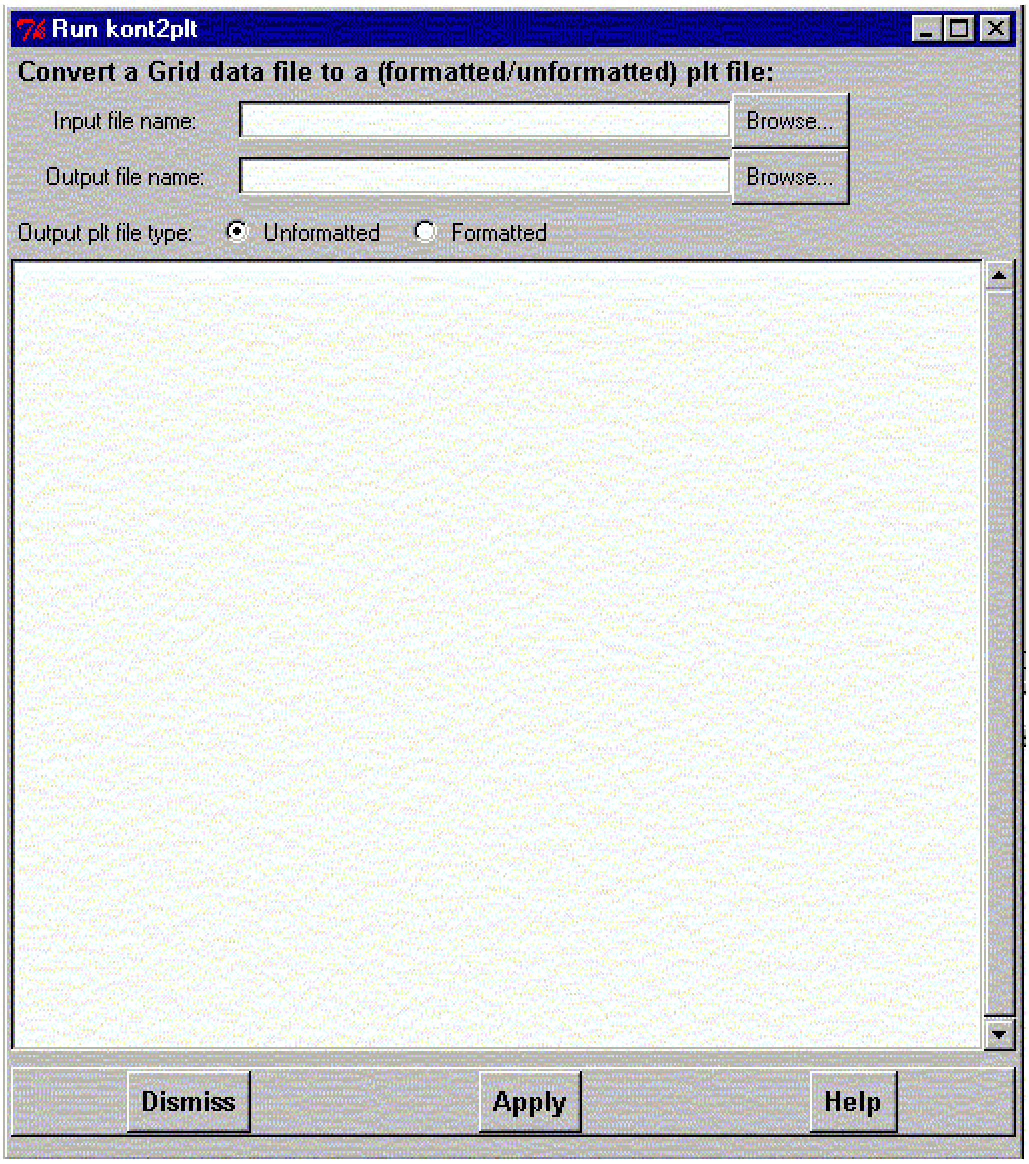


Line command: **there is no line command available inside gOpenMol. Use Unix/Dos shell line commands.**

Run Kont2plt program widget

Run the Kont2Plt program to convert a grid mesh file produced by the Grid program into a formatted or unformatted plt file format used by gOpenMol.

Give input and output file names. Choose either **unformatted** or **formatted** output plt file. Click the **Apply** button to do the conversion.



Convert a Grid data file to a (formatted/unformatted) plt file:

Input file name:

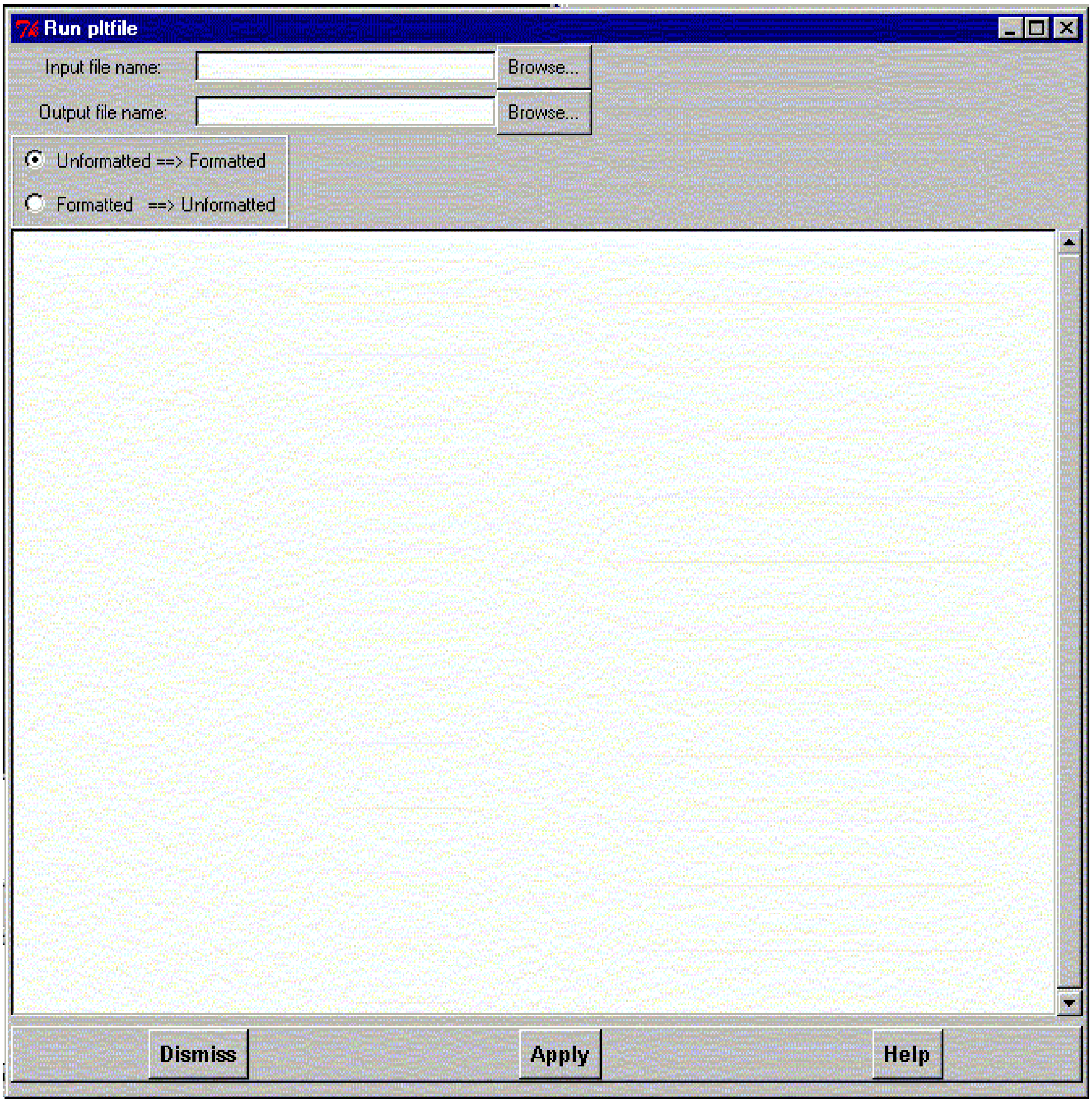
Output file name:

Output plt file type: ● Unformatted ○ Formatted

Line command: **no line command available inside gOpenMol. Use Unix/Dos shell line command.**

Run pltfile program widget

Run the plotfile to convert a plt file from binary to ASCII or ASCII to binary. The operation is mostly used to move a plt file between different hardware platforms.



Line command: see **run command**

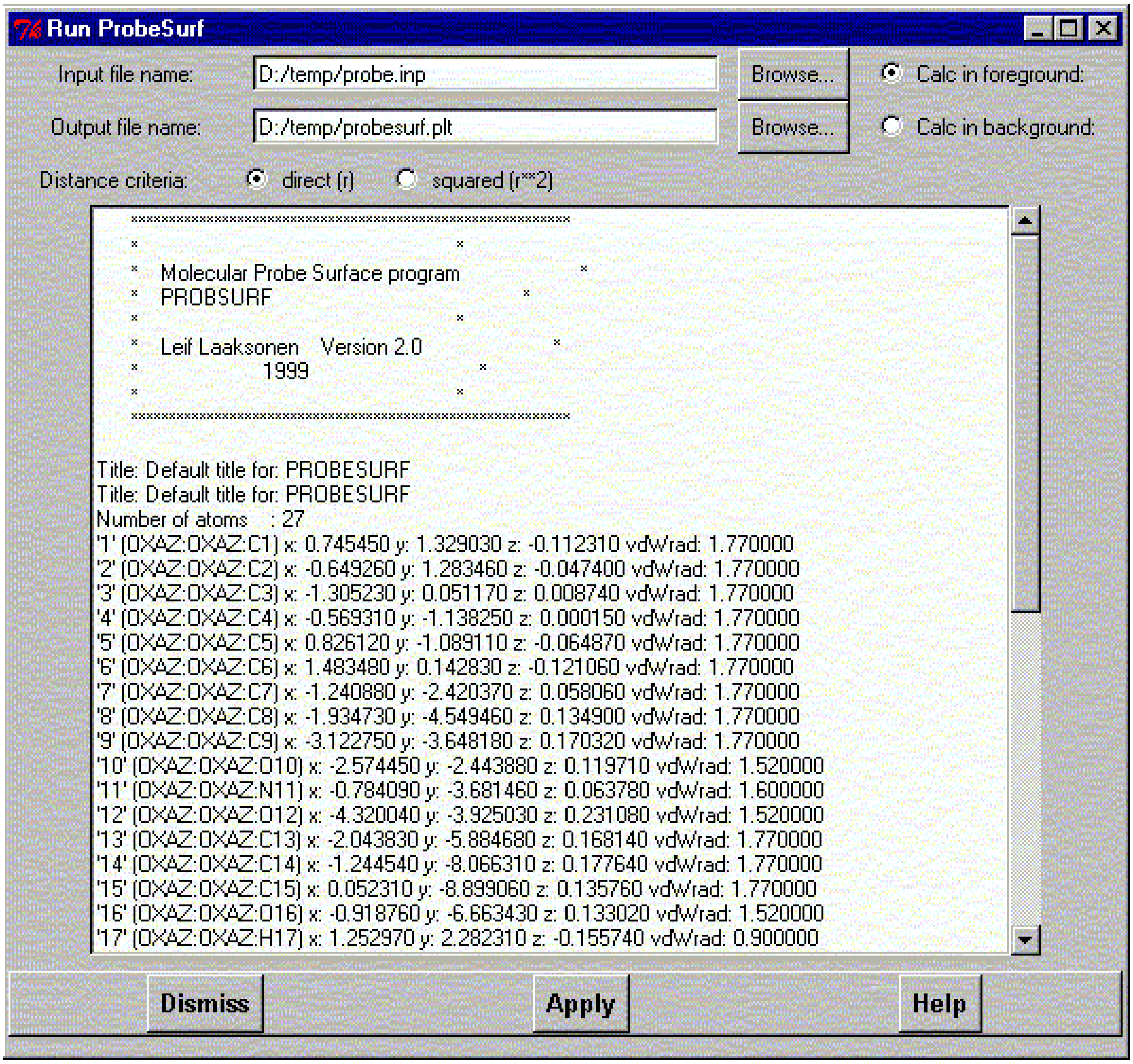
Run probesurf program widget

Run the ProbSurface program to produce a Connolly type of surface around a molecule system. You have to have an input file ready for the program which you define in the "Input file name" entry. If you don't remember the name and location of the input file you can browse by pressing the **Browse** button. The output file is the name of the contour grid file. If no name is given the program uses the default name, which will be placed in the "Output file name" entry.

The ProbSurf calculation can be done either in **foreground** or **background**. The foreground calculation locks gOpenMol until the calulation has finished. The background mode enables the further usage of gOpenMol while the calculation continues.

The ProbSurf program uses a direct **(r)** and squared **(r**2)** distance method for the calculation of the surfaces. Choose the method to be used.

After the calculation has finished it is possible to read in the contour file and display the surface. Answer **yes** or **no** to the question.



Input file name: D:/temp/probe.inp

Output file name: D:/temp/probesurf.plt

● Calc in foreground:

○ Calc in background:

Distance criteria: ● direct (r) ○ squared (r**2)

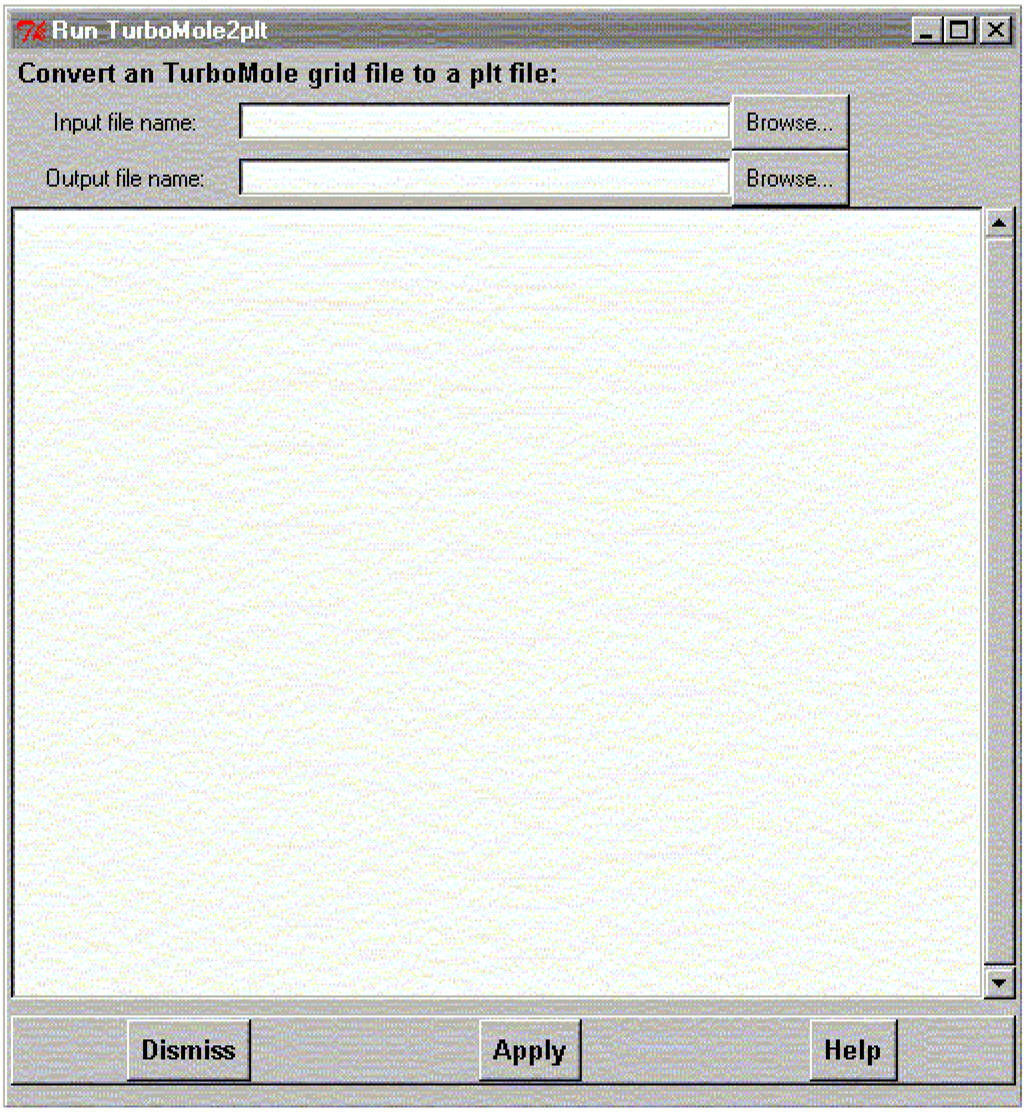
```
    *********************************************************
    *                                          *
    *   Molecular Probe Surface program                  *
    *   PROBSURF                              *
    *                                          *
    *   Leif Laaksonen    Version 2.0              *
    *                 1999                   *
    *                                          *
    *********************************************************

Title: Default title for: PROBESURF
Title: Default title for: PROBESURF
Number of atoms    : 27
'1' (OXAZ:OXAZ:C1) x: 0.745450 y: 1.329030 z: -0.112310 vdWrad: 1.770000
'2' (OXAZ:OXAZ:C2) x: -0.649260 y: 1.283460 z: -0.047400 vdWrad: 1.770000
'3' (OXAZ:OXAZ:C3) x: -1.305230 y: 0.051170 z: 0.008740 vdWrad: 1.770000
'4' (OXAZ:OXAZ:C4) x: -0.569310 y: -1.138250 z: 0.000150 vdWrad: 1.770000
'5' (OXAZ:OXAZ:C5) x: 0.826120 y: -1.089110 z: -0.064870 vdWrad: 1.770000
'6' (OXAZ:OXAZ:C6) x: 1.483480 y: 0.142830 z: -0.121060 vdWrad: 1.770000
'7' (OXAZ:OXAZ:C7) x: -1.240880 y: -2.420370 z: 0.058060 vdWrad: 1.770000
'8' (OXAZ:OXAZ:C8) x: -1.934730 y: -4.549460 z: 0.134900 vdWrad: 1.770000
'9' (OXAZ:OXAZ:C9) x: -3.122750 y: -3.648180 z: 0.170320 vdWrad: 1.770000
'10' (OXAZ:OXAZ:O10) x: -2.574450 y: -2.443880 z: 0.119710 vdWrad: 1.520000
'11' (OXAZ:OXAZ:N11) x: -0.784090 y: -3.681460 z: 0.063780 vdWrad: 1.600000
'12' (OXAZ:OXAZ:O12) x: -4.320040 y: -3.925030 z: 0.231080 vdWrad: 1.520000
'13' (OXAZ:OXAZ:C13) x: -2.043830 y: -5.884680 z: 0.168140 vdWrad: 1.770000
'14' (OXAZ:OXAZ:C14) x: -1.244540 y: -8.066310 z: 0.177640 vdWrad: 1.770000
'15' (OXAZ:OXAZ:C15) x: 0.052310 y: -8.899060 z: 0.135760 vdWrad: 1.770000
'16' (OXAZ:OXAZ:O16) x: -0.918760 y: -6.663430 z: 0.133020 vdWrad: 1.520000
'17' (OXAZ:OXAZ:H17) x: 1.252970 y: 2.282310 z: -0.155740 vdWrad: 0.900000
```

Line command: see **run command**

Run turbomole2plt program widget

Run the TurboMole2plt program to convert a TurboMole grid file into the plt file format used by gOpenMol. The TurboMole grid file can then be displayed using the gOpenMol contour utilities

Write the input and output file names or click the **Browse** buttons to define the file names. The press the **Apply** button to do the conversion.

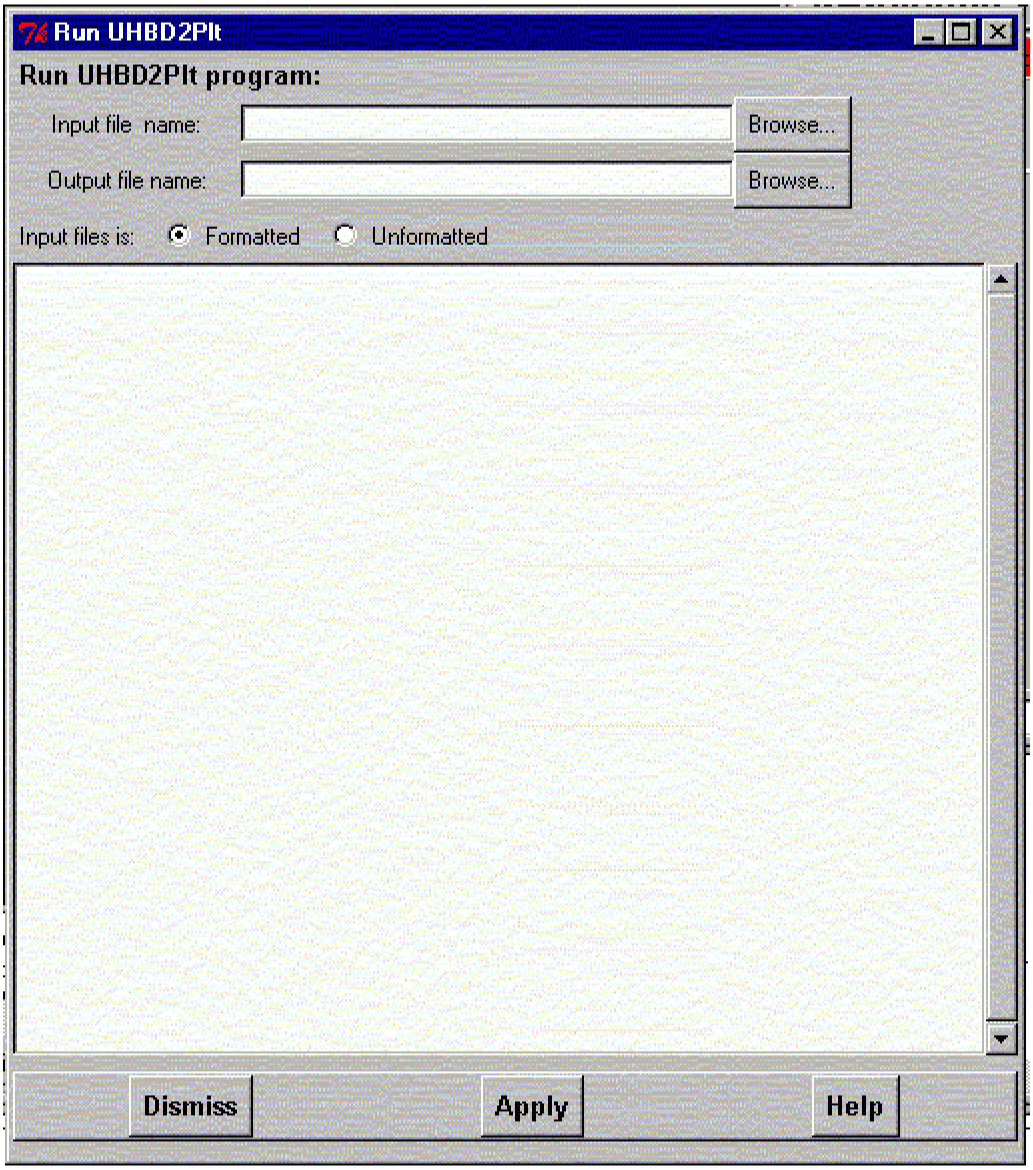


Line command: **can't be run from gOpenMol line mode. Use the Unix/Dos command line mode**

Run UHBD2plt program widget

Run the UHBD2plt program to convert a formatted or unformatted UHBD phi grid file to a plt file format used by gOpenMol.

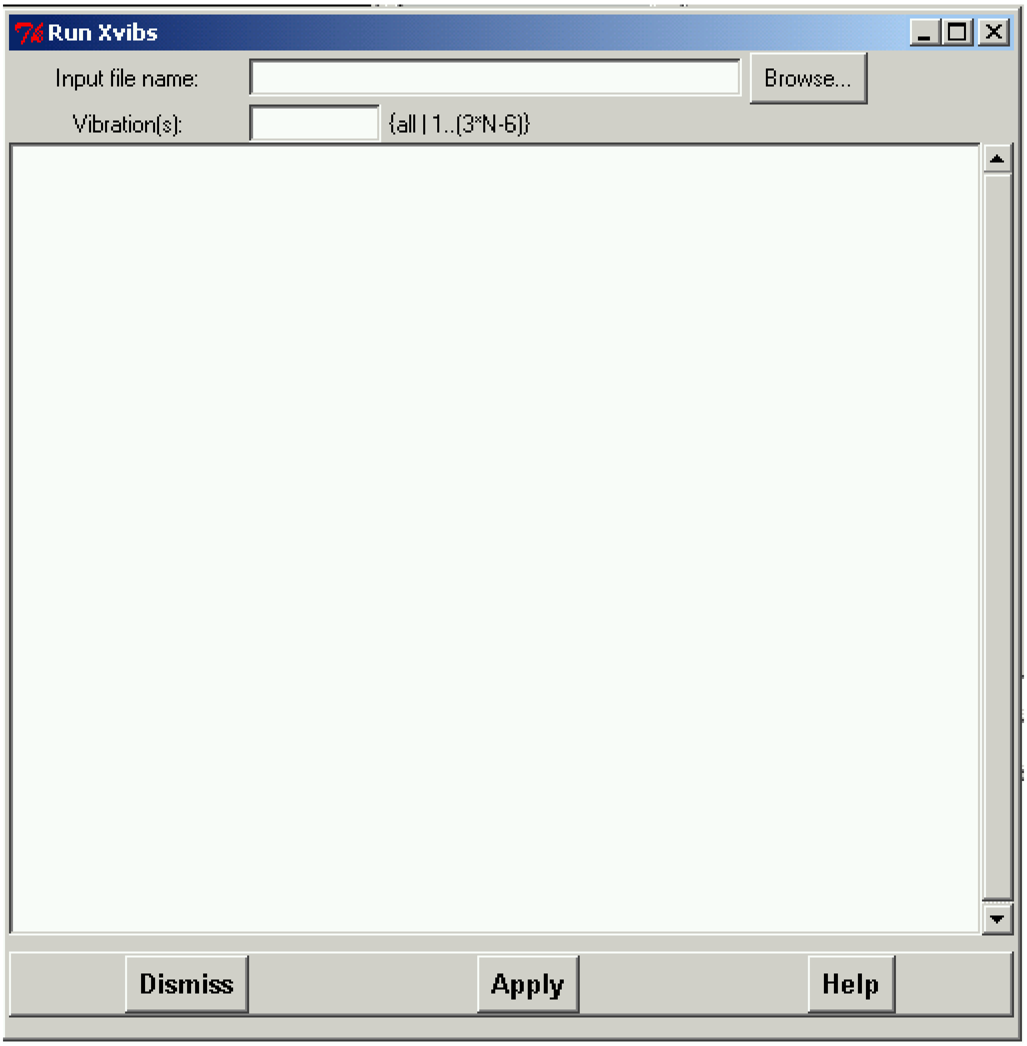
Give the input and output file names. Choose as the input file type either **formatted** or **unformatted**. Click the **Apply** button to do the operation.



Line command: **no line command available inside gOpenMol. Use Unix/Dos shell line command.**

Run Xvibs program widget

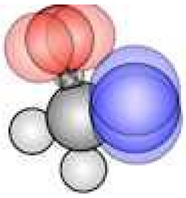
Give the input file name and specify which vibrations you want to convert into XMOL files using the xvibs program. Click on the “Apply” button to make the conversion.



Please observer the xvibs program by Bradley A. Smith (<yeldar@home.com>) included with gOpenMol is not the latest version available! If you want to run xvibs as a separate stand-alone program and use all the power of this program please download the latest version from:

http://members.home.net/yeldar/xvibs/

The xvibs program page for the xvibs version gOpenMol is using:



XVibs

A utility for animating molecular vibrations. Normal modes are read from a file automatically determined to be from Aces2, Gamess, PC Gamess, Gaussian 92/94/98, or ADF. Separate animation files are written for each vibration. The animation is a simple cosine trajectory of the normal mode vectors. The output files are in XYZ format with normal mode vectors attached to each atom.

If you would like this program extended to other input or output formats, please send your files to Bradley A. Smith and I will gladly extend it for you.

Send bug reports to Bradley A. Smith (<yeldar@home.com>).

Usage

```
xvibs [-version] [-bounce] [-frames n] file frequency_number
```

Options

`-version`	display the version of xvibs.
`-bounce`	create bouncing rather than looping animations.
`-frames n`	produce `n` frames in the animations.
`file`	the file from which to read data.
`frequency_number`	the number of the frequency to output. Specifying "`all`" will output all frequencies.

Examples

```
xvibs ch3oh_gam.out all
```

Outputs all vibrations into files named '`ch3oh_gam.`{001-018}`.xyz`'

```
xvibs ch3oh_gam.out 15
```

Outputs only vibration 15 into the file named '`ch3oh_gam.015.xyz`'

By default, XYZ files created are in loop animation. The option `-bounce` will create bouncing animations.

Download

The current release of xvibs is version 5.2, and is distributed under GNU General Public License.

Binaries are available for Java, Windows, Linux, and SGI. The source code and sample data are available in a Zip archive.

Line command: see **run command**

(9) TOOLS menu has the options:

AutoDock tools

This is the main widget for the **AutoDock** trajectory display and analysis facility.

There are first 4 file readers:

- Read a *.pdbq file (ligans)
- Read a *.pdbq file (system into which the ligand is docked)
- Read the AutoDock trajectory
- Read an AutoDock log file to extract the included initila and docked coordinate sets. The cruncher creates a set of files with the coordinates

With an AutoDock trajectory I mean the "ligand.tout" file as in the following:

The traj command which is used in the trajectory output, creates trajectories in two formats, one in PDB format which is written to the log file specified by the -l flag, and one in a raw ASCII format which is written to the standard output. For example, if the command issued was:

```
autodock -p ligand.dpf -l ligand.trj.conv.log -c < trj.conv.com > ligand.tout
```

the file ligand.trj.conv.log will contain a PDB formatted list of all the trajectory steps, while the ligand.tout file will contain raw ASCII data, with one row for each atom in the ligand, and each column containing data. The command file trj.conv.com contains only the commands:

```
traj ligand.trj
stop
```

This was taken from Web pages and manuals at:
http://www.scripps.edu/pub/olson-web/doc/autodock/

To read and display a trajectory from the AutoDock do the following:

1. First define the file name of the ligand either by directly writing the file and or by using the file browser. Then press the "Import" button to apply the file.
2. If you also want to include the system (into which the ligand is docked) do the same as in the previous section but for the system.
3. Last but not least define the trajectory file. Use the same approach as in the previous sections. Remember to press the "Import" button.

After this you should be able to see the number of frames displayed.

By default the display goes from the first frame to the last with the step 1. This can be changed by supplying new values in the respective fields.

It is possible to display the trajectory in a loop, one frame at the the backwards and forwards or go to the first and last frames.

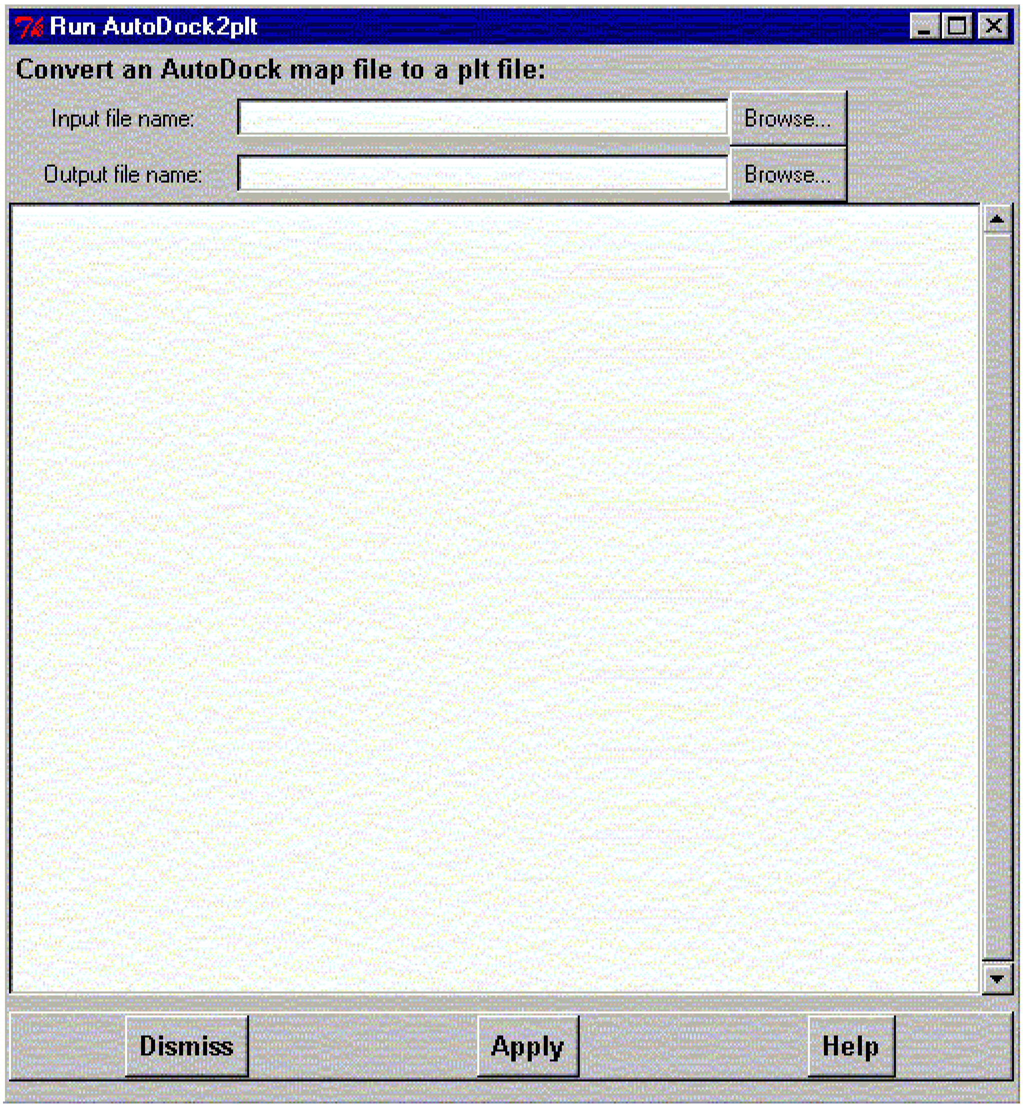
Frame control:

- **|<**... go to first frame
- **<**... back one frame
- **[]**... halt display loop
- **|>**... play forward one frame at the time in a loop
- **>**... forward one frame
- **>|**... go to last frame

There are two trajectory retrieval methods available:

> The **fast** method uses a pre-calculated index (jump) vector into the trajectory file for the various frames. This method is capable of retrieving a frame at a speed independent of the frame position in the file. However, be careful with this if you move files between Unix and Windows systems! The slow method is a "stupid" read process that always reads all the frames up to the needed frame. This process becomes very slow for big files but it is not sensitive to the characters at the end of the line.

At the bottom of the widget there is also a file writer to write out the ligand and system coordinates as *.pdbq files. This is very handy in case you import the atom partial charges and want to write a *.pdbq file with the new charges.

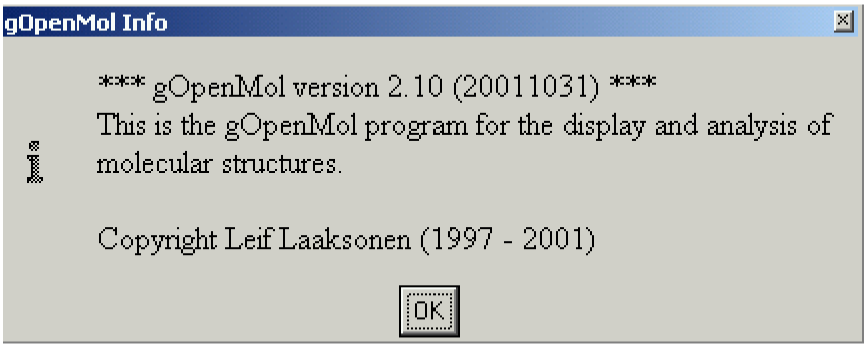


(10) HELP menu has the options:

About, Show the about text about gOpenMol.

About gOpenMol widget

Display the "About gOpenMol" information.



*** gOpenMol version 2.10 (20011031) ***
This is the gOpenMol program for the display and analysis of molecular structures.

Copyright Leif Laaksonen (1997 - 2001)

Demo, Demo widget.

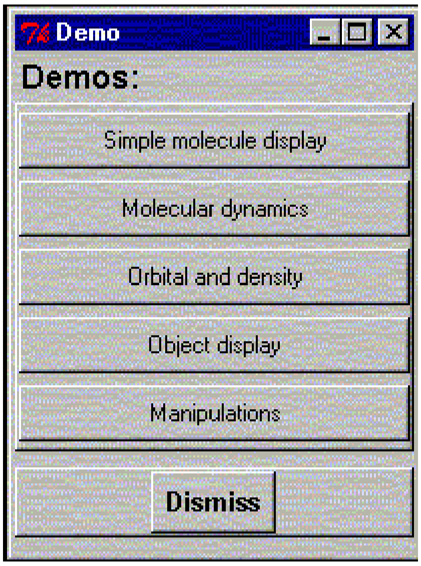
Demo widget

This is the demo widget. Click on the button to run a demo.

Examples of:

- simple molecule display
- molecular dynamics display and analysis
- orbital and electron density display and analysis
- the usage of some of the gOpenMol object display
- manipulation of structures

Click button to run your demo of choice!



Help,   Show this file.

Help widget

The help system is built upon a Tcl/Tk based Web browser that has very few features compared to the modern Netscape or Microsoft Internet Explorer Web browsers.

The Help system supports the following functions:

- **About** shows the short list of information about gOpenMol.
- **Demo** gives a list of demos available.
- **Help** shows the help pages for running gOpenMol.
- **Menu help** shows the information about the pulldown menu system.
- **Peek version** takes a http connection to the gOpenMol distribution site and checks for the latest version available. This feature will help you to upgrade and maintain gOpenMol in the future.
- **Tutorials** gives you a list of tutorials about how to run gOpenMol. If you have prepared a good tutorial please let me know and I will include it here.

Internal data structure for gOpenMol

To be able to fully use gOpenMol it's useful to understand how the atom assignment is done and how the internal storage is structured.

How are the atoms defined?

The molecule/atom information is organised as in the CHARMM/CHARMm program(s). All atoms in a molecule system can be defined/identified through a *segmen*t *name*, *residue name* and/or *residue number* and an *atom name*. This arrangement is very flexible and can be used to define quite complicated set ups and commands.

The line commands, when they want to be applied on a set of atoms, must always be given as:

Segment_name   Residue_name/number   Atom_name/number

The segment, residue and atom name can be 1-4 characters long. The names are assigned when one reads a coordinate file. For a command the data is supplied as three fields separated by ':'. Example ***set atom colour S1 * CA yellow*** sets all CA atoms in all residues in segment S1 to the colour yellow. The general identifier looks like **segment residue atom** where the segment name is always a character string. The wild character '*' and the '?' characters can be used. The '*' character matches all strings and '?' matches any character at it's position. In the residue and atom field residue numbers and atom numbers can be used or a combination of characters and integer numbers. Example: set atom colour **S1 TYR,1,10,30-45 CA,N,C** blue command sets all CA,N and C atoms in all tyrosine residues and residues 1,10 and range 30-45 in segment S1 to the colour blue.

Please note:
The interpreter for gOpenMol interprets the lower and upper case letters as different. This applies for both commands and segment/residue/atom names.

- Install gOpenMol
- Begin a gOpenMol session
- Display contour data
- Import a structure into gOpenMol
- Manipulate a structure in gOpenMol

Tcl line command interface to gOpenMol

The command line parser is implemented using the Tcl (Tool command language). The language is rich of commands and I strongly recommend you first to have a look the the Tcl commands (http://www.sunlabs.com:80/research/tcl/). For most commands it is enough just to use the four first characters to uniquely define the command. A clear exception is the 'contour' command where the first five characters are needed not to mix the command with the 'continue' statement. Through the help files I have used a dollar ($) sign to indicate there characters needed to define a command. You can also get a list of all available commands by typing the question (?) mark at the command prompt. Very many of the commands return a parameter (or several) which can be used as an input to a new command.

Please observe that when using the commands in **scripts** one needs to fill in the full command name! The dollar sign ($) in the commands indicates the length of the command or the characters needed to uniquely define the command. The dollar sign must **not** be included in the command.

Commands available:

atom	*define various properties about atoms*
calc$ulate	*calculate molecular properties*
conto$ur	*display and define isisurface plots*
defi$ne	*define general properties*
displ$ay	*display the current picture (same as clicking the logo button)*
edit	*edit molecule and molecule properties*
expo$rt	*export molecular structures or atom coordinates*
fill	*fill time series vectors*
find	*find/calculate various things*
gomError "text"	*print the "text" through the ERROR module*
gomPrint "text"	*print the "text" through the Print module*
gscal$e	*scale the current graphics display*
hard$copy	*make a hardcopy/print of the current display*
help	*help system*
impo$rt	*import molecular coordinates and other things*
mani$pulate	*manipulate time series*
moni$tor	*monitor molecular properties from a trajectory*
mope$n	*open and import an old model file*
msav$e	*save model file*
pause Fvalue	*pause for Fvalue seconds (there is also a Tcl/Tk command)*
plot	*plot various things*
plum$ber	*plot a ribbon or tube through atoms*
ptra$ce	*trace atoms through the trajectory*
quit	*stop program and exit*
rese$t	*reset the system*
rota$te	*rotate the current display*
run	*run various external programs*
sele$ct	*make a selection from the current atoms*
show	*show various things*
traj$ectory	*trajectory display and analysis control*
tran$slate	*translate the current display*
wind$ow	*window operations*

```
**************************************************************************
                             atom command
                        Leif Laaksonen CSC 1996
**************************************************************************
```

The 'atom' command controls the display and the display style of the atoms. One can choose between the atom/style **on** or **off** using the "-" sign in front of the command.

```
#
# display the atoms with name segment name Seg
#                              residue name Res
#                              atom name    Atm
#
```

```
atom  disp$lay   Seg  Res  Atm

#
# display all atoms with a radius of Float around Seg1 Res Atm1
#

      disp$lay   Seg1 Res1 Atm1 around Float

#
# display all atoms of name Seg2 Res2 Atm2 with a radius Float around
# Seg2 Res2 Atm2
#
# this function returns also the number of atoms selected
#

                                around Float Seg2 Res2 Atm2

#
# display all atoms of name Seg2 Res2 Atm2 with a radius Float around
# Seg2 Res2 Atm2 with colour Colour_name
#
# this function returns also the number of atoms selected
#

                                around Float Seg2 Res2 Atm2  Color_name

#
# display all atoms of name Seg2 Res2 Atm2 with a radius Float around
# Seg2 Res2 Atm2 with colour Int Int Int (RGB colour indexes, 0-255)
#
# this function returns also the number of atoms selected
#

                                                            "Int Int Int"

#
# display all atoms of name Seg2 Res2 Atm2 with a radius Float around
# Seg2 Res2 Atm2 with colour Float Float Float (RGB colour, 0.0 - 1.0)
#
# this function returns also the number of atoms selected
#

                                                            "Float Float
Float"

#
# turn the display off for Seg Res Atm
#

atom -dis$play   Seg  Res  Atm

#
# turn the display off for atoms within radius Float around Seg1 Res Atm1
#

     -dis$play   Seg1 Res1 Atm1 around Float
```

```
#
# turn the Seg2 Res2 Atm2 atom display off for atoms within a radius Float
# around Seg1 Seg2 Atm2
#

                                around Float Seg2 Res2 Atm2

                                around Float Seg2 Res2 Atm2  Color_name
                                                            "Int Int Int"
                                                            "Float Float
Float"

#
# change the atom colour for Seg Res Atm to Colour_name/RGB index
#

atom  color      Seg Res Atm    Color_name
                               "Int Int Int"
                               "Float Float Float"

#
# colour atoms Seg Res Atm according to the vector array value
# from Fmin to Fmax
#

                 byve$ctor      Seg Res Atm Fmin Fmax

#
# colour atoms Seg Res Atm according to their atomic partial charge value
# from Fmin to Fmax
#

                 bych$arge      Seg Res Atm Fmin Fmax

#
# display atoms Seg Res Atm as CPK spheres
#

atom  cpk        Seg Res Atm

#
# turn the CPK display off for atoms Seg Res Atm
#

atom -cpk        Seg Res Atm

#
# display atom labels (full or minumum)
# full includes segment, residue and atom names (numbers)
# by default only the atom label (name) is shown
#

atom  labe$l     Seg Res Atm
                              full
     -lab$el     Seg Res Atm
```

```
#
# display atoms Seg Res Atm in the licorice display style
#

atom  lico$rice  Seg Res Atm

#
# turn the licorice display style off for atoms Seg Res Atm
#

atom -lico$rice  Seg Res Atm

#
# scale the CPK sphere(s) for Seg Res Atm with a value of Float
#

atom  scal$e cpk   Float Seg Res Atm

#
# when executing a command to include atoms according to a radius
# the atoms included will be selected either by atom or by whole residue
# this command selects either the atom based or whole residue base
selection
# default: atom

atom  sele$ction   atom
                   resi$due



**************************************************************************
                             calculate command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# calculate average structure from a trajectory
#

calc$ulate   avst$ructure

#
# calculate backbone dihedral angles for a protein
#

             bbdi$hedrals

#
# calculate the clustering for atoms Seg Res Atm through the frames
# in a trajectory
#

             clus$ter     Seg Res Atm

#
# delete all cluster data
#
```

```
            -clu$ster

#
# (re)calculate the atom connectivity information for Seg Res Atm
#

             conn$ectivity Seg Res Atm

#
# calculate correlation array for distance/angle/torsion array(s)
# numbered Iserie1 and Iserie2
#

             corr$elation dist$ance  Iserie1 Iserie2
                          angl$e     Iserie1 Iserie2
                          tors$ion   Iserie1 Iserie2

#
# calculate geometrical center for atoms Seg Res Atm
#

             geomc$enter  Seg Res Atm

#
# calculate mass center for atoms Seg Res Atm
#

             massc$enter  Seg Res Atm

#
# calculate distance between Seg1 Res1 Atm1 Seg2 Res2 Atm2
#

             dist$ance    Seg1 Res1 Atm1 Seg2 Res2 Atm2

#
# calculate angle between Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3
#

             angl$e       Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3

#
# calculate torsion between Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3
Seg4 Res4
#

             tors$ion     Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3 Seg4
Res4 Atm4

#
# calculate mean square displacement for atoms Seg Res Atm using
# geometrical center
#

             msdi$splacement  atom         Seg Res Atm

#
```

```
# calculate mean square displacement for atoms Seg Res ATm using
# mass center
#

             msdi$splacement  mass$center  Seg Res Atm

#
# fit the atoms Seg1 Res1 Atm1 in system #1 to
#         atoms Seg2 Res2 Atm2 in system #2
# 0 = numerical display off, 1 = numerical display on
#

             quat$fit  StrucN1 Seg1 Res1 Atm1 StrucN2 Seg2 Res2 Atm2 0/1

#
# calculate root mean square fluctuation for atoms Seg Res Atm
# through the frames
#

             rmsf$luctuation  Seg Res Atm

#
# calculate radial distribution function for atoms Seg2 Res2 Atm2
# calculated from Seg1 Res Atm1. The calculation will only
# be done for the current structure (system) displayed! However
# applying the command successively will add the new calculated
# rdf to the previous one(s) calculated. This enables the calculation
# of rdf  for the full set of frames in a trajectory. To reset the
# calculation apply the "calc -rdf" command.
# To calculate the average from a set of applied "calc rdf" commands
# apply the "calc rdfmean" command.
#

             rdf          Seg1 Res1 Atm1 Seg2 Res2 Atm2 Fcut Tbin
            -rdf
             rdfm$mean



**************************************************************************
                             center command
                        Leif Laaksonen CSC 1999
**************************************************************************
#
# center the system according to the given atoms (segment, residue, atom)
# or coordinates
#
#

cent$er  syst$em  Seg Res Atm
                  Xcoord Ycoord Zcoord


**************************************************************************
                             contour command
                        Leif Laaksonen CSC 2001
**************************************************************************
```

```
#
# Command to handle the various operations for a grid data to be
# displayed as a isocontour surface
#

#
# Read a file with the name File.Name containing grid data and give
# the set the name Contour.name
#

conto$ur   file        File.Name Contour.Name

#
# Delete all contour information for all files
#

          -fil$e
           dele$te

#
# Print contour information about the files
#

           info

#
# Plot the isosurface for contour with the name Contour.Name
# using an isovalue Fvalue and colour Colour [Fvalue Colour ...]
# The colour can be given as a (1) name, (2) RGB integer value or
# (3) RGB floating values.
#

           plot        Contour.Name Fvalue  Colour Fvalue Colour  ...
           plot        Contour.Name Fvalue "Fvalue Fvalue Fvalue" ...
           plot        Contour.Name Fvalue "Ivalue Ivalue Ivalue" ...

#
# If contour mapping is used you have to give
# (1) The contour for wich one wants to define an iso value
# (2) The contour from which the property will be mapped
# (3) A min and max value inside which the colour scaling will be done
#

           plot {Contour.Name1 Contour.Name2} Fvalue  MinVal MaxVal  ...


#
# Change the transparent value for contour with the name Contour.Name
# to Fvalue. If there are several isocontour values defined apply on
# the one with index Ivalue. Fvalue is 1. for no transparency and 0.0
# for full transparency.
#

           alph$ablend Contour.Name Fvalue  Ivalue

#
# Use smoothing for the isocontour surface with the name Contour.Name
# and index Ivalue.
#

           smoo$th     Contour.Name on      Ivalue

#
# Do not use smoothing for the isocontour surface with the name
# Contour.Name and index Ivalue.
#

           smoo$th     Contour.Name off     Ivalue

#
# Display isocontour surface with name the name Contour.Name and
# index Ivalue.
#

           disp$lay    Contour.Name on      Ivalue

#
# Do not display isocontour surface with name the name Contour.Name and
# index Ivalue.
#

           disp$lay    Contour.Name off     Ivalue

#
# Change the surface display type to solid for isocontour surface with
# the name name Contour.Name and index Ivalue.
#

           type        Contour.Name soli$d  Ivalue

#
# Change the surface display type to mesh for isocontour surface with
# the name name Contour.Name and index Ivalue.
#

           type        Contour.Name mesh    Ivalue

#
# It is possible to display a contour so that the contour polygon
# values are saved in an array. The default is direct which do not save
# the information. To get the polygon data look into the show command.
# It is impossible to differ the contour levels from the data array.
# The array is just an array of triangular meshes.

           meth$od     dire$ct
                       save

#
# Command to control the colour mapping of the values from one grid file
# on the surface of an other isocontour file
# ContourName#2 will now be mappend on the surface of ContourName#1
#

           mapp$ing ContourName#1 ContourName#2
```

```
# Command to combine a contour file to a particular structure
# Glue contour with name "Contour.Name" to structure number "Structure#"
#

           comb$ine   Contour.Name   Structure#

#
# Contour clip plane commands
# Put a clip plane at Fvalue with the clipping in the +/- direction
# along the defined axis.
# Put the defined clip (x, y or z axis) plane on/off
# or to display/turn off
#
# Observe that the clip plane is always related to the local coordinate
# system!
#

           clip$plane       x+             Fvalue
           clip$plane       x-             Fvalue
           clip$plane       y+             Fvalue
           clip$plane       y-             Fvalue
           clip$plane       z+             Fvalue
           clip$plane       z-             Fvalue

           clip$plane       x              on
           clip$plane       x              off
           clip$plane       y              on
           clip$plane       y              off
           clip$plane       z              on
           clip$plane       z              off

           clip$plane       xyz1           x1 y1 z1 x2 y2 z2 x3 y3 z3
                            xyz2           x1 y1 z1 x2 y2 z2 x3 y3 z3
                            xyz3           x1 y1 z1 x2 y2 z2 x3 y3 z3

           clip$plane       xyz1           on
                                           off
                            xyz2           on
                                           off
                            xyz3           on
                                           off


**************************************************************************
            define command
            Leif Laaksonen CSC 1996
**************************************************************************

#
# define background colour by defining a colour name or
# integer or float triplet (rgb) values
#

defi$ne    bgco$lour    Colour.Name
                        "Fred Fgreen Fblue"
```

```
#
# define bond display style options:
#   smooth (atom colour changes smoothly along the bond)
#   half or default (half of the bond is in the colour of the connected
atom)
#

           bondst$yle   smoo$th
                        half
                               defa$ult

#
# define cell dimensions, line width and colour
#

            cell       dime$nsions  Fa Fb Fc
                       angl$es      Falpha Fbeta Fgamma
                       line$width   Ivalue
                       colour       ColourName
                                    "Fred Fgreen Fblue"

#
# define colour mapping for values between 0.0 ... 1.0
# the scale goes from (0.0) blue to (1.0) red through green at (0.5)
#

            colorm$apping     texture
                             rainbow

#
# define the draw buffer (default is back)
#

            drawb$uffer back
                        fron$t

#
# define the near plane distance and step values
#

            near$plane  step   Fvalue
                        valu$e Fvalue

#
# define the far plane distance and step values
#

            farp$lane   step   Fvalue
                        valu$e Fvalue

#
# define viewing angle for perspective projection
#

            viewa$ngle         Fvalue

#
```

```
# define material specular (0.0 - > 128.0)
#

            mate$rial   spec$ular Fvalue

#
# molecule line width in pixels
#

            mlin$ewidth Ivalue

#
# not in use
#

            node        Node.Name

#
# contour line width in pixels
#

            clin$ewidth Ivalue

#
# define licorice sphere and cylinder for the molecule display
# good for the ball and stick display
#

            licos$phere   Fvalue
            licoc$ylinder Fvalue

#
# define the cylinder and sphere quality
#

            cyli$nderquality Ivalue # any number > 0 (default 10)
            sphe$requality Ivalue # any number > 0 (default 10)

#
# not in use
#

            hurl           URL.Address
            webb$rowser    WebBrowser.Name

#
# define default coordinate file type and extension
#

            defa$ult       coor$dinate type ambe$r
                                            char$mm
                                            gaus$sian
                                            hype$rchem
                                            insi$ght
                                            mol2
                                            mumo$d
                                            open$mol
                                            pdb
                                            xmol
                                            xyz
                                            yasp
                            exte$nsion      ambe$r String
                                            char$mm String
                                            gaus$sian String
                                            hype$rchem String
                                            insi$ght String
                                            mol2 String
                                            mumo$d String
                                            open$mol String
                                            pdb String
                                            xmol String
                                            xyz String
                                            yasp String

#
# define default trajectory file type and extension
#

                            traj$ectory     type         ambe$r
                                                         ceri$us2
                                                         char$mm
                                                         disc$over
                                                         gromo$s
                                                         hype$rchem
                                                         mumo$d
                                                         xmol
                                                         xplor
                                                         yasp
                                            exte$nsion   ambe$r String
                                                         ceri$us2 String
                                                         char$mm String
                                                         disc$over String
                                                         grom$os String
                                                         hype$rchem String
                                                         mumo$d String
                                                         xmol String
                                                         xplo$r String
                                                         yasp String

#
# Gromos96 uses a constant with which the coordinates will be
# multiplied (default = 10.0).
#

             gromos96        coorda$mplifier Fvalue

#
# toggle rotations state between on/off
#

               rota$tion   stat$e on
                                  off

#
```

```
# toggle translation state between on/off
#

               tran$slation stat$e on
                                   off

#
# activate or deavtivate the state for picking atoms
#

               iden$ntify on
                          off

#
# define atom neighbour search window (in atoms)
#

              atom        wind$ow    Inumber

#
# defines whether the system is translated around origo or not
# this is active both at reading a new structure or when there
# already are structures read
#

              syst$em     tran$slation    on
                                          off

#
# define the projection method orthographic/perspective
#

              proj$ection       orth$ographic
                                pers$pective

#
# define window actions
# single/multi windowing
# update is automatic/manual
#

              wind$owing        sing$le
                                multi
                                upda$te      auto$matic
                                            manu$al

#
# put the trajectory frame number display on/off
#

              traj$ectory       fid           on
                                              off

#
# define the cutplane 3d magnitude. Constant with which the magnitude is
# multiplied with. The range is between 0.0 and 1.0.
#
```

```
               cutp$lane         damp$ing   Fvalue
#
# put the stereo pair display mode on/off
# define the distance between the objects in the stereo pair mode
# define the "tilt" angle for the objects (+-)
#

               spai$r            on
                                 off
                                 distance Fvalue
                                 angle    Fvalue

#
# put the harware stereo (stereo in a window mode on/off
# define the "tilt" angle for the objects (+-)
#

              quad$stereo       on
                                off
                                angl$e    Fvalue


#
# be careful with these commands ... do not use them!
#
# reconnectivity defines if the atom connection table
# is recalculated between the frames or ont.
# maxconnectivity is the max number of connections (bonds)
# an atom can have
#

               atom         coor$dinates Fx Fy Fy IAindex ISindex
                            char$ge      Fvalue   IAindex ISindex
                            labe$l       Name     IAindex ISindex
                            resn$umber   Ivalue   IAindex ISindex
                            vdw          Fvalue   IAindex ISindex
                            iden$ntify   on
                                         off
                            reco$nnectivity  on
                                             off
                            maxc$onnectivity

               resi$due     labe$l       Name  IAindex ISindex
               segm$ent     labe$l       Name  IAindex ISindex
               stru$cture   StrucName  Iatoms   new
                                                appe$nd


***************************************************************************
display command
Leif Laaksonen CSC 2001
***************************************************************************

#
# display the scene (this is usually always need)
#
```

```
# check opition (default) will look into the event queue during the display
# nocheck will not look into the event queue
# sleep Ivalue guarantees that the display takes at least Ivalue
# milliseconds
# this can be used for demos to slow down the display frequency if you have
# a very fast machine
#
disp$lay     chec$k
             noch$eck
             slee$p     Ivalue
disp$lay


***************************************************************************
                         edit command
                    Leif Laaksonen CSC 1996
***************************************************************************
#
# edit atom bonding
#

edit  bond  brea$k   Seg1 Res1 Atm1
                     Seg1 Res1 Atm1 Seg2 Res2 Atm2
            crea$te  Seg1 Res1 Atm1 Seg2 Res2 Atm2


***************************************************************************
                             export command
                        Leif Laaksonen CSC 1996
***************************************************************************

#
# export dihedral (backbone) angles for a protein
#

expo$rt  bbdi$hedrals     File.Name

#
# export coordinates in various format
# the structure number (Istruct) need also to be defined
# (1) Ball & Stick, (2) CHARMM/CHARMm, (3) CHARMM free,
# (4) Karplus = CHARMM, (5) OPENMOL, (6) PDB,
# (7) PDBQ (AutoDock), (8) TINKER xyz, (9) UHBD (qcd),
# (10) User definable, (11) Simple xyz
#

         coor$dinates  Istruct   bands$tick File.Name
                                            File.Name  disp$laymask
                                 char$mm    File.Name
                                            File.Name  disp$laymask
                                 free       File.Name
                                            File.Name  disp$laymask
                                 karp$lus   File.Name
                                            File.Name  disp$laymask
                                 open$mol   File.Name
                                            File.Name  disp$laymask
                                 pdb        File.Name
```

```
                                        File.Name  disp$laymask
                             pdbq       File.Name
                                        File.Name  disp$laymask
                             txyz       File.Name
                                        File.Name  disp$laymask
                             uhbd       File.Name
                                        File.Name  disp$laymask
                             user       File.Name
                                        File.Name  disp$laymask
                             xyz        File.Name
                                        File.Name  disp$laymask

#
# export input (raw) input to external programs
# GAMESS, extended Huckel program ICON8, MOPAC,
# OpenMol and Probesurf
#

         inpu$t    game$ss     IStrNum  File.name
                   icon$8      IStrNum  File.name
                   mopa$c      IStrNum  File.name
                   open$mol    IStrNum  File.name
                   prob$esurf  IStrNum  File.name

#
# export cluster matrix
#

         clus$ter          File.Name  "File label text"

#
# export correlation array
#

         corr$elation      File.Name

#
# export model file (default gOpenMol structure file)
#

         mode$l            File.Name

#
# export distance time series (index and file name)
#

         dist$ance         list       Ivalue   File.Name
         dist$ance         seri$es    Ivalue   File.Name

#
# export angle time series (index and file name)
#

         angl$e            list       Ivalue   File.Name
         angl$e            seri$es    Ivalue   File.Name

#
```

```
# export torsion time series (index and file name)
#

         tors$ion         list       Ivalue   File.Name
         tors$ion         seri$es    Ivalue   File.Name

#
# export radial distribution function
#

         rdf              File.name

#
# export mean square displacement data
#

         msdi$splacement  File.Name

#
# export root mean square fluctuation data
#

         rmsf$luctuation  File.name


**************************************************************************
                             fill command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# fill a pre defined distance array from a trajectory file
#

fill   dist$ance  arra$y
                  list

#
# delete distance list(s) and all free all tempoary space
#

      -dis$tance

#
# fill a pre defined angle array from a trajectory file
#

       angl$e     arra$y
                  list

#
# delete angle list(s) and all free all tempoary space
#

      -ang$le

#
# fill a pre defined torsion array from a trajectory file
```

```
#

       tors$ion   arra$y
                  list

#
# delete torsion list(s) and all free all tempoary space
#

      -tor$sion

#
# fill the atom parameter information into atom structure for
# the atom with index Inumber
#

       stru$cture  Inumber


**************************************************************************
                             find command
                        Leif Laaksonen CSC 1996
**************************************************************************


#
# Find S-S bonds in a structure
# Appply the search on all structures available or just to the structure
# with number "StructureNumber". It is also possible to give a distance
# Fbond inside which the S-S must be.
#

find   ssbo$nds   all             [Fbond]
                  StructureNumber [Fbond]


**************************************************************************
                             gscale command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# Scale the graphical display (equally in the x,y and z directions)
#

gscal$e   disp$lay   Fvalue


**************************************************************************
                             hardcopy command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# make a screen dump of the screen in a selected format
# into a file
#
# WinNr is the window number from the left upper corner or
# the running number in which the windows have been created
#
```

```
# for the PostScript option it is also possible to define the
# landscape orientation through the "-l" and the pixels/inch
# density through the "-ppixelsperinch"
#

hard$copy WinNR bmp          File.Name
                post$script  File.Name {-l | -ppixelsperinch}
                rgb          File.Name
                targa        File.Name
                xwd          File.Name


**************************************************************************
                             import command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# import coordinate (atom) information from a file
# 1: Amber          input coordinates !DO NOT USE!
# 2: Charmm *.crd   input coordinates
# 3: A frame from a trajectory file
# 4: Gaussian formatted checkpoint file
# 5: Gromacs coordinate file format
# 6: Gromos96 formatted coordinate file
# 7: HyperChem  *.hin  input coordinates
# 8: Insight *.car  input coordinates
# 9: Mol2    *.mol2 input coordinates
#10: Mopac output graphics file (NOT WORKING)
#11: OpenMol input file
#12: Brookhaven *.pdb input coordinates
#13: XMOL *.xmol      input coordinates
#14: Plain xyz *.xyz  input coordinates
#15: YASP *.yasp      input coordinates

impo$rt    coor$dinates     ambe$r      File.Name !DO NOT USE!
                            char$mm     File.Name
                            fram$e      Ivalue
                            gaus$sian   File.Name
                            groma$cs    File.Name
                            groom$s96a  File.Name
                            hype$rchem  File.Name
                            insi$ght    File.Name
                            jagu$ar     File.Name
                            mol2        File.Name
                            mopa$c      File.Name
                            mumo$d      File.Name
                            open$mol    File.Name
                            pdb         File.Name
                            xmol        File.Name
                            xyz         File.Name
                            yasp        File.Name

#
# import a pre calculated cluster matrix
#
```

```
           clus$ter         File.name

#
# import a dictionary file (containing atom information)
# into gOpenMol
#

           dict$ionary      File.Name

#
# import a vector file defining the size and direction of
# an array property attached to the atoms
#
# there are currently two options the CHARMM force file and
# a flat file containg the vector data
#

           vect$or  char$mm   File.Name
                    fatf$ile  File.Name

#
# import Gaussian basis set data base
#

           gbas$is          File.Name

#
# import a gOpenMol model file
#

           mode$l           File.Name

#
# import atom partial charges from
# 1. ICON8 output file
#

           char$ge  icon8   File.Name


***************************************************************************
                             manipulate command
                        Leif Laaksonen CSC 1996
***************************************************************************

mani$pulate   time$serie dist$ance  Action  Iserie
mani$pulate   time$serie angl$e     Action  Iserie
mani$pulate   time$serie tors$ion   Action  Iserie

The Action is one of the following:

      dave$rage
      squa$re
      cos
      cos2
      sqrt
      dini$tial
      log
      exp
      dmin
      abs
      divf$irst
      divm$aximum
      pspe$ctrum
      zero
```

```
mani$pulate   time$serie dist$ance  Action  Iserie1 Iserie2
mani$pulate   time$serie angl$e     Action  Iserie1 Iserie2
mani$pulate   time$serie tors$ion   Action  Iserie1 Iserie2
```

The Action is one of the following:

```
      copy
      add
```

```
mani$pulate   time$serie dist$ance  Action  Iserie Fvalue
mani$pulate   time$serie angl$e     Action  Iserie Fvalue
mani$pulate   time$serie tors$ion   Action  Iserie Fvalue
```

The Action is one of the following:

```
      powe$rreal
      mult$real
      divi$dereal
      shif$treal
```

```
**************************************************************************
                             monitor command
                        Leif Laaksonen CSC 1996
**************************************************************************
```

```
#
# Monitor a distance, angle or torsion
# The operation with the "-" sign deletes the whole
# preset structure
# Change the line type or the line colour of list number Ilist
#
```

```
moni$tor  dist$ance   Seg1 Res1 Atm1 Seg2 Res2 Atm2
                      colo$ur   CName Ilist
                      type      Inum  Ilist
         -dis$tance
          angl$e      Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3
                      colo$ur   CName Ilist
                      type      Inum  Ilist
         -ang$le
          tors$ion    Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3 Seg4
Res4 Atm4
                      colo$ur   CName Ilist
                      type      Inum  Ilist
          dihe$dral   Seg1 Res1 Atm1 Seg2 Res2 Atm2 Seg3 Res3 Atm3 Seg4
Res4 Atm4
                      colo$ur   CName Ilist
```

```
                    type      Inum  Ilist
         -tor$sion
         -dih$edral

#
# Put the display switch on for the defined distance, angle and
# torsion
#

          disp$lay    dist$ance  on
                                 off
                      angl$e     on
                                 off
                      tors$ion   on
                                 off
                      dihe$dral  on
                                 off


**************************************************************************
                             mopen command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# Open an gOpenMol model file
#

mope$n  File.Name


**************************************************************************
                             msave command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# Save the structure information into a gOpenMol model file
#

msav$e  File.Name


**************************************************************************
                             plot command
                        Leif Laaksonen CSC 1998
**************************************************************************

#
# Plot various things
#

#
# plot an arrow
#

plot  arro$w     Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Radius Colour appe$nd
```

```
                 Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Radius Colour
     -arro$w

#
# Plot a cut plane through the contour data
# Choose through which axis the plane goes (x,y or z)
# Give the tag name for the contour
# Give the coordinate on the axis, through which the plane goes
# Give max and min scale (blue -> red)
# Define if the cutplane will be in 2d (flat) or 3d for the respective
# plane
# Arbitrary cut planes can be defined from three coordinates.
#

      cutp$lane  x  ContourName Xcoord
                 x  ContourName Xcoord Fmin Fmax
                 x  ContourName Xcoord Fmin Fmax log10/log
                 y  ContourName Ycoord
                 y  ContourName Ycoord Fmin Fmax
                 y  ContourName Ycoord Fmin Fmax log10/log
                 z  ContourName Zcoord
                 z  ContourName Zcoord Fmin Fmax
                 z  ContourName Zcoord Fmin Fmax log10/log
                 2dx
                 3dx
                 2dy
                 3dy
                 2dz
                 3dz
                 xyz1 ContourName x1 y1 z1 x2 y2 z2 x3 y3 z3 Fmin Fmax
                 xyz2 ContourName x1 y1 z1 x2 y2 z2 x3 y3 z3 Fmin Fmax
                 xyz3 ContourName x1 y1 z1 x2 y2 z2 x3 y3 z3 Fmin Fmax
                 prof$ile ContourName x Ibins Fmin Fmax
                                      y Ibins Fmin Fmax
                                      z Ibins Fmin Fmax

     -cutp$lane  x
                 y
                 z
                 xyz1
                 xyz2
                 xyz3
#
# Show a profile of the grid data in the x-, y- or z-direction
# The data is sampled in Ibins using the Fmin Fmax scaling
#
                 prof$ile ContourName x Ibins Fmin Fmax
                                      y Ibins Fmin Fmax
                                      z Ibins Fmin Fmax

     -cutp$lane  x
                 y
                 z

#
# Plot a sphere at the defined coordinate
#
```

```
      sphe$re    Xc Yc Zc Radius Colour appe$nd
                 Xc Yc Zc Radius Colour
                 Xc Yc Zc Radius Colour Xscale Yscale Zscale
                 Xc Yc Zc Radius Colour Xscale Yscale Zscale appe$nd
     -sph$ere

#
# Plot a cylinder from (x1,y1,z1) to (x2,y2,z2) with radius Radius and
colour Colour
# It is also possible to plot a cylinder with a different radius and colour
at each ends.
#

      cyli$nder  Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Radius Colour appe$nd
                 Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Radius Colour
                 Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 {Radius1 Radius2} {Colour1
Colour2}
     -cyl$inder

#
# Plot a line from (x1,y1,z1) to (x2,y2,z2)
#

      line       Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Colour appe$nd
                 Xc1 Yc1 Zc1 Xc2 Yc2 Zc2 Colour
     -lin$e

#
# Plot a linear distance plot for the selected atoms
# Put the display state on/off
#

      ldp        atom$s  Seg  Res  Atm
                 on
                 off

#
# Plot a plane through the x,y or z axis at the defined coordinate
# Extend the plane D? in the negative and positive directions
#

      plan$e     x Xcoord Dy Dz Colour
                 y Ycoord Dx Dz Colour
                 z Zcoord Dx Dy Colour
                 x Xcoord Dy Dz Colour append
                 y Ycoord Dx Dz Colour append
                 z Zcoord Dx Dy Colour append

#
# Put cluster display toggle on/off
#

      clus$ter   on
                 off
     -clu$ster
```

```
#
# Plot a colour scale with Ilevels from (min) Fmin to (max) Fmax
# Put the display off
#

      csca$le    Ilevels Fmin Fmax
     -csc$ale

#
# Toggle a vector display on/off
# The data has been read by the "import vector" command
#

      vect$or    on
                 off
                 atom$s  Seg  Res  Atm  Radius  Scale

#
# Toggle the cell display between on/off
#

      cell       on
                 off

#
# A very simple way to plot thext
#
# Plot text (static) in Colour at x,y (0.0 - 1.0) with string
# TextString
#
# Plot text (fixed in the x,y,z space) in Colour at x,y,z with
# TextString
#

      text       Colour Xcoord Ycoord TextString
      text3      Colour Xcoord Ycoord Zcoord TextString
     -tex$t

#
# Plot the local coordinate system at selected atoms (Segment Residue Atom)
#
# Turn the plot off

      axis  Segment Residue Atom
      axis  Segment Residue Atom XaxisLength YaxisLength ZaxisLength

     -axi$s


**************************************************************************
                             plumber command
                        Leif Laaksonen CSC 1996
**************************************************************************


#
# Draw a tube or a ribbon through the selected atoms
# in ColourName and Frad wide
```

```
#

plum$ber  atom$s  Seg Res Atm Frad ColourName tube
                                              ribb$on
         -ato$ms

#
# Toggle the display state between on/off
#

          disp$lay  on
                    off


**************************************************************************
                             ptrace command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# Command to trace atoms (particle trace)
#
# Select the atoms to include in the trace set
# toggle between the display on/off states
#

ptra$ce  atom$s     Segment Residue Atom [appe$nd]
        -ato$ms
         disp$lay   on
                    off

#
# Write a probesurface input file using the traced
# atoms. This can be used to see the volume movable
# atoms cover during a simulation.
#

         write      prob$esurf  File.Name


**************************************************************************
                             reset command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# reset atom colours
#

        atomc$olours

#
# Reset gOpenMol
#

        gope$nmol

#
```

```
# Reset the view to the original
#

rese$t  view

#
# Reset atom connectivity
#

        conn$ectivity   Segment Residue Atom



************************************************************************
                             rotate command
                        Leif Laaksonen CSC 1996
************************************************************************

#
# Rotate the atom display around the x,y and z axis
#

rota$te   disp$lay    Fxrot Fyrot Fzrot

#
# Rotate only the selected atoms
#

          sele$ction  Fxrot Fyrot Fzrot


************************************************************************
                             run command
                        Leif Laaksonen CSC 1997
************************************************************************

#
# Run various programs
#
# The OutGridFile name is optional if one wants to use
# an other name from the default name (probesurf.plt)
#

run prob$esurf  InputFile
                InputFile  OutGridFileName
    xvibs       InputFile  all
                           1..(3*N-6)
                                       palindrome



************************************************************************
                             select command
                        Leif Laaksonen CSC 1996
************************************************************************

#
```

```
# Select atoms from a structure
#

sele$ct  atom$s      Seg Res Atm
        -ato$ms

#
# Select all atoms in all structures or the defined ones
#

         stru$cture  all
                     Ivalue1 Ivalue2 Ivalue3  ....
        -str$ucture  all
                     Ivalue1 Ivalue2 Ivalue3  ....

**************************************************************************
                             show command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# Show used cpu time (not on Windows 95)
#

show   cput$ime

#
# Show process information (not on Windows)
#

       proc$ess

#
# Show current time
#

       time

#
# Show save status (if program needs saving of structure
# before exit)
#

show   saves$tatus


# These commands return values which can be used as input to new commands.

#
# Return a value showing the user if the graphics is available
# Return value == 0 means no graphics, != 0 means X-Windows is on
#

       grap$hics

#
```

```
# Return molecule stick display line width
#

       mlin$ewidth

#
# Return directory where the binary files are
#

       bind$irectory

#
# Return directory where the data files are
#

       datad$irectory

#
# Return login directory
#

       homed$irectory

#
# Return background colour as a rgb triplet float float float
#

       bgco$lour

#
# Return color mapping type. The two types are: "texture" (1D texture) and
# "rainbow".
#

       colorm$apping

#
# Return distance to near clipping plane
#

       near$plane   step
                    valu$e

#
# Return distance to far clipping plane
#

       farp$lane    step
                    valu$e

#
# return perspective projection viewing angle
#

       viewa$ngle

#
```

```
# Return drawing buffer (back/front)
#

       drawb$uffer

#
# Return cylinder quality factor
#

       cyli$nderquality

#
# Return sphere quality factor
#

       sphe$requality

#
# Return a "rainbow" colour from a scale factor (blue ... green ... red)
#

       rain$bow Fvalue

#
# Return number of molecular structures defined
#

       mols$tructures

#
# Return number of atoms in atom structure Istructure
#

       numa$toms         Istructureindex

#
# Return atom number for Seg Res Atm
#

       atomn$umber       Seg Res Atm

#
# Return licorice sphere radius
#

       licos$phere

#
# Return licorice cylinder radius
#

       licoc$ylinder

#
# Return number of frames in the trajectory file
#
```

```
       numf$rames

#
# Return number of root mean square fluctuation
# atoms in the list
#

       rmsf$luctuation   leng$th

#
# Return root mean square fluctuation (total,x,y,z) for
# index Irmsfluctuation
#

                         Irmsfluctuation index

#
# Return real atom index from rmsf list for Irmsfatomindex
#

                         atomi$ndex  Irmsfatomindex

#
# Return the system translation state (translation on or off)
# If on the system will be translated around origo
#

       syst$em           tran$slation

#
# Return structure selection status (= 0 , not selected , = 1 , selected)
#

       stat$us IStructureNumber

#
# Show if the selected coordinates are ative or not (returns on of off)
#

       sele$ction

#
# Return segment, residue and atom names
#        gbasis set
#        residue number
#        cpk scale
#        coordinates x y z
#        bvalue
#        wdv value
#        colour  r g b
#        nuclear charge
#        connectivity for atom Iatomindex?
#        atom display  state (0 = atom will not be shown, = 1 will be
shown)
#        atom cpk      state (0 = atom cpk not be shown,  = 1 will be
shown)
```

```
#        atom licorice state (0 = atom licorice not be shown, = 1 will be
shown)
#        atom neighbour search window in atoms
#
# for Iatomindex in
# Istructureindex
#
#        atom selection mode (residue/atom)
#        atom connectivity is recalculated between the frames (1)
#                             not recalculated                (0)
#
#        maxconnectivity is the max number of connections (bonds)
#        an atom can have
#

       atom              segm$entname   Iatomindex Istructureindex
                         resi$duename   Iatomindex Istructureindex
                         atom$name      Iatomindex Istructureindex
                         gbas$is        Iatomindex Istructureindex
                         resn$umber     Iatomindex Istructureindex
                         cpks$cale      Iatomindex Istructureindex
                         coor$dinates   Iatomindex Istructureindex
                         bval$ue        Iatomindex Istructureindex
                         vdw            Iatomindex Istructureindex
                         colo$ur        Iatomindex Istructureindex
                         nucl$earcharge Iatomindex Istructureindex
                         conn$ectivity  Iatomindex1 Iatomindex2 ...
                         disp$laystate  Iatomindex Istructureindex
                         cpkst$ate      Iatomindex Istructureindex
                         lico$ricestate Iatomindex Istructureindex
                         wind$ow
                         sele$ction
                         reco$nnectivity
                                              maxc$onnectivity

#
# show bond style (smooth/half)
#

                         bondst$yle

#
# show atom identify stage on/off (1/0)
#

                         iden$tify

#
# Show various contour related information
#
#  Contour file name for Icontourindex
#  show number of contours defined
#  show number of levels defined for Ivalueindex
#  show contour type for Ivalueindex
#  minimum and maximum values for the contour data in Icontourindex
#  show display state for Icontourindex
#  show tag name for Icontourindex
#  show smoothing state (1 = on , 0 = off)
#  show contour cell dimensions (xmin xmax ymin ymax zmin zmax xdim ydim
zdim)
#  where the xdim, ydim and zdim are the number of points in the x,y z
#  directions.
#

       contou$r          file$name     Icontourindex
                         defi$ned
                         leve$ls       Icontourindex
                         type          Icontourindex
                         tran$sparency Icontourindex
                         alph$ablend   Icontourindex
                         mini$mum      Icontourindex
                         maxi$mum      Icontourindex
                         disp$lay      Icontourindex
                         name          Icontourindex
                         smoo$th       Icontourindex
                         cube          Icontourindex
                         cell          Icontourindex

#
# Show cutplane properties
#
# shows cutplane smoothing for the yz, xz and xy planes
#                damping or the constant with which the value between 0.0
and 1.0
#                is multiplied
#                type returns the current type for the yz, xz and xy
planes. The
#                return value can be 2d or 3d.
#

       cutp$lane         smoot$h   yz
                                   xz
                                   xy
                         damp$ing
                         type

#
# It is possible to extract the polygon data from the last
# displayed surface(s) using the polygon command.
# The first command gives the number of polygons (triangles)
# and the second gives:
# x1,y1,z1,nx1,ny1,nz1,x2,y2,z2,nx2,ny2,nz2,x3,y3,z3,nx3,ny3,nz3,
# c1,c2,c3
# where xi,yi,zi are the corned points of the triangle,
# nxi,nyi,nzi are the surface normals at the corners and
# c1, c2 and c3 are a colour value between 0.0 and 1.0,
# which goes from blue to red through green. The entry number
# EntryIndex is the index to the data structure.
# The method option gives either direct or save
# where direct means that the polygons are are not saved
# and save means that the polygons are saved to be retried
# using the data option
#
```

```
                         poly$gon   entr$ies
                                    data      EntryIndex
                                    meth$od

#
# Return cell colour
#             dimensions (a b c alpha beta gamma)
#             line width in pixels
#

       cell              colo$ur
                         dime$nsions
                         line$width

#
# Return Gaussian basis sets
#                 tag name for Ivalue index
#                 entry    for Ivalue index
#

       gbas$is           sets
                         tag        Ivalue
                         entr$y     Ivalue

#
# Return material specular
#

       mate$rial         spec$ular

#
# Return monitor distance state
#                         colour for Index
#        list of atoms defining the list
#        number of distance time series
#

       moni$tor          dist$ance  stat$e
                                    colo$r  Index
                                    type    Index
                                    list
                                    times$eries

#
# Return monitor angle    state
#                         colour for Index
#        list of atoms defining the list
#        number of distance time series
#

       moni$tor          angl$e     stat$e
                                    colo$r  Index
                                    type    Index
                                    list
                                    times$eries

#
```

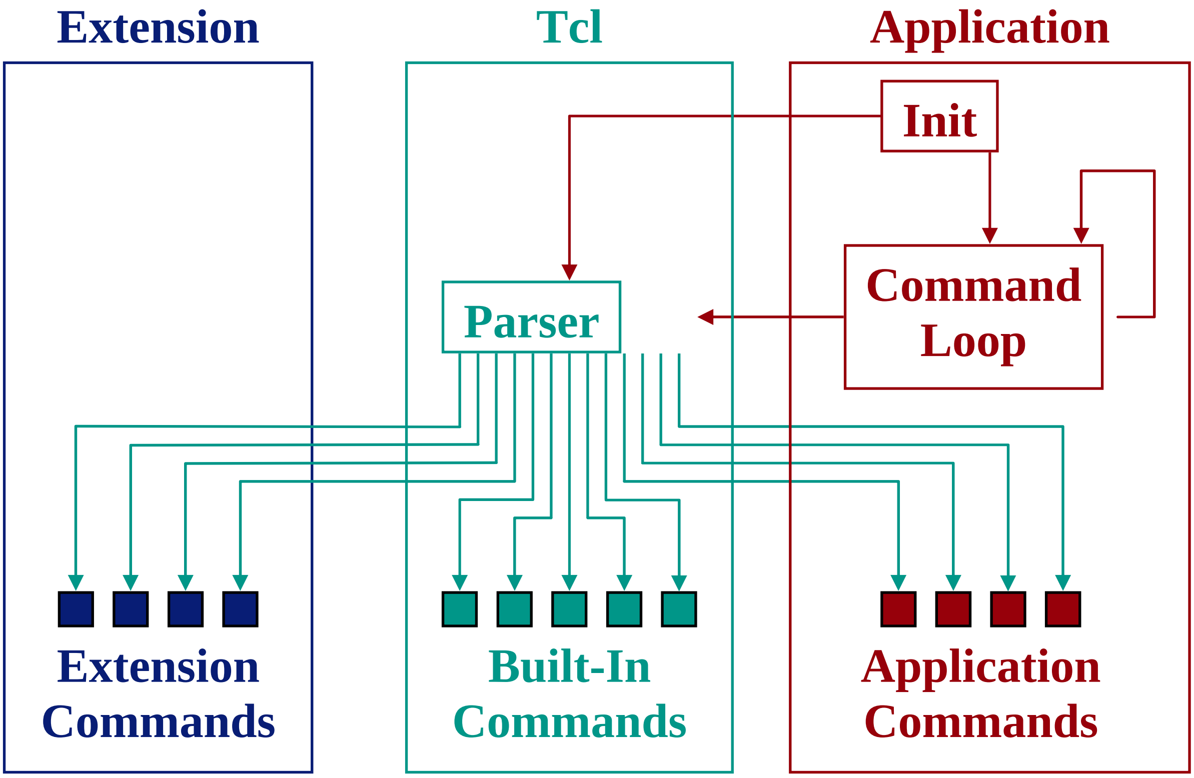
```
# Return monitor torsion  state
#                         colour for Index
#        list of atoms defining the list
#        number of distance time series
#

       moni$tor          tors$ion   stat$e
                                    colo$r  Index
                                    type    Index
                                    list
                                    times$eries

#
# Return radial distribution function
# calculate and return average from collected entries
# return number of discrete observations (bins)
# return status

       rdf               aver$age
                         obse$rvations
                         poin$ts
                         stat$us

#
# Return number of trajectory frames
# return status value to see if there is a trajectory defined
# return trajectory file name
# return trajectory display parameters (first frame, last frame, step
frame)
# return number of current displayed frame
# display frame number (on) or turn frame number display off
#

       traj$ectory       frames
                         defi$ned
                         file$name
                         disp$lay
                         curr$ent
                         fid         on
                                                       off

#
# Return the Gromos96 coordinate amplifier with which the coordinate
# values will be multiplied
#

       gromos96          coorda$mplifier

#
# Return number of trace sets defined
#

       trac$es

#
# Return rotations state
#
```

```
       rota$tion         stat$e

#
# Return translation array (x, y , z) and state  (1 == on , 0 == off)
#

       tran$slation      arra$y
                             stat$e

#
# Show the used projection method
# 0 = perspective
# 1 = orthographic
#

        proj$ection

#
# Show window state
# returns either single or multi
# Show window defined
# returns the currently active/defined number of windows
# Show window parameters for window WinNR
# returns the window location (x, y) and window width and height
#

        wind$ow   stat$e
                  defi$ned
                                 para$meters   WinNr

#
# Show pairwise (crossedeye stereo) parameters
# status returns either "on" or "off" depending on if the display is on or
off
# angle returnes the angle in which the pair is "tilted"
# distance gives the distance with which the pair is translated to each
other
#

       spai$r    stat$e
                 angl$e
                 dist$ance


**************************************************************************
                          trajectory command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# Define trajectory information
#
# Define file type and name
# Alloved types are: amber, charmm, discover, famber, gtomacs,
#                    gromos, gromos96a , hyperchem, mumod,
#                    xmol, xplor
```

```
#

traj$ectory   file     Itype   File.Name

#
# Define trajectory display limits (first , last , step)
#

              limi$ts  Ifirst  Ilast  Istep


**************************************************************************
                             translate command
                        Leif Laaksonen CSC 1996
**************************************************************************
#
# Translate the system
#

tran$slate   disp$lay    Xtrans Ytrans Ztrans
             sele$ction  Xtrans Ytrans Ztrans


**************************************************************************
                             window command
                        Leif Laaksonen CSC 1996
**************************************************************************

#
# Window id is:
# 0: Tcl/Tk control window
# 1: OpenGL molecule window
# 2: LDP window
#
# Resize window to Iwidth and Iheight
# Move window to coordinates Ix Iy
# Iconify/deiconify window
# Make the graphics window(s) full screen
#

window  WinID  resi$ze  Iwidth Iheight
               move     Ix  Iy
        WinID  icon$ify
        WinID  deic$onify
        WinID  full$screen
```

Tcl/Tk in gOpenMol

gOpenMol has the Tcl/Tk command parser included that enables the usage of the Tcl/Tk command sets and a variety of new commands defined for gOpenMol. This



feature enables the extension of the application either through C-coding or Tcl/Tk scripts.

- Application generates scripts.
- Tcl parses scripts, passes words to command procedures.
- Application extends built-in command set:
  - Define new object types in C.
  - Implement primitive operations as new Tcl commands.
  - Build complex features with Tcl scripts.

- Extensions can be developed independently:
  - Network communication, database access, security, …
  - Applications can include combinations of extensions.

When you learn more you can even send commands to gOpenMol from the tclsh program!

When gOpenMol starts it reads first in the **gopenmolrc.tcl** script that defines some essential macros and definitions to be able to run gOpenMol. At the end of this macro it tests to see if a graphics device is attached. If there is a graphics device attached it also reads in the **gopenmol_guirc.tcl** script that defines the entire GUI for gOpenMol. The gopenmolrc.tcl and gopenmol_guirc.tcl files are the two essential files defining the entire gOpenMol environment and GUI.

There is an interaction between these files and the gOpenMol engine so be warned if you edit these files. gOpenMol will also at start up and at different stages make some working variables into the Tcl/Tk space that are used again by the parser.

gOpenMol Utility Tools

There are a variety of tools available for gOpenMol. The tools mentioned here are either FORTRAN or C programs. Please look into the utility/ directory

- Program to convert a cube output from the GaussianXX program into a plot format recognized by gOpenMol.
- Program to convert TURBOMOLE moloch grids into a format recongized by gOpenMol.
- Program to convert "cube" data in PC GAMESS PUNCH files into a format recongized by gOpenMol.
- Program to manipulate a contour file in the plt-format.
- Convert a SYBYL ascii trajectory into an AMBER binary one.
- Merge two CHARMm binary trajectories into one binary.
- Convert a CHARMm trajectory between **formatted <==> unformatted**.
- Convert a multistep XMOL coordinate file into AMBER binary trajectory format.
- Extended Huckel program (ICON8) in the ICON8 directory
- Probesurf program to generate probe surfaces (Appendix 4)
- VSS program calculates the electrostatic potential from the ICON8 output (Appendix 4)
- Convert GRID and AutoDock potential energy grid files into a format recognized by gOpenMol.

External libraries used by gOpenMol

1. OpenGL library. For more information about the library please look at the web site http://www.opengl.org/. For those machines that do not have the library available the MESA3D library (http://www.mesa3d.org/) is used.
2. Tcl/Tk scripting libraries. For more information about the libraries please look at the web site http://dev.scriptics.com/.
3. The OpenGL Utility Toolkit (GLUT), originally written by Mark Kilgard and ported to Win32 (Windows) by Nate Robins.
   Unix: http://reality.sgi.com/mjk_asd/glut3/glut3.html
   Windows: http://www.xmission.com/~nate/glut.html

4. JPEG library from the Independent JPEG Group (http://www.ijg.org/):
    IJG is an informal group that writes and distributes a widely used free library for JPEG image compression. You can find our code and some supporting documentation at ftp://ftp.uu.net/graphics/jpeg/. If you can't cope with tar.gz-format archives, you may prefer the zip-format release at ftp://ftp.simtel.net/pub/simtelnet/msdos/graphics/jpegsr6b.zip.
5. Python scripting engine can be used inside gOpenMol. For more information about the libraries please look at the web site http://www.python.org.

Extending gOpenMol through own code

It is possible to extend gOpenMol using plugins that are implemented through a dll file (Windows) or sharable object (Unix/Linux). Following is a list of plugins available.

- **Module:** VRML, outputs ball-and-stick, CPK, and licorice views. Simple stick can be selected, which is a variation of licorice and trajectories with start/stop animation.
  Author: Kevin Boyd, University of New Orleans
  Link: http://137.30.117.235/chem/vrml/vrml.html
  Status: Included with the program.

- **Module:** Filters, input coordinate filters for binary Chem3D and Spartan data files.
  Author: Kevin Boyd, University of New Orleans
  Link: not available
  Status: Included with the program.

- **Module:** Ball, is a demo on how to implement a plugin that displays a sphere in the gOpenMol graphics window.
  Author: Eero Häkkinen, CSC
  Link: The source code is in the src/plugins/ball directory.
  Status: Included with the program.

There is also a tutorial for making plugins "Extending gOpenMol in the Windows Environment" by Kevin Boyd at the University of New Orleans. The link to this resource is http://137.30.117.235/chem/gopenmol.htm.

Appendix:

Appendix 1

Coordinate file formats

gOpenMol can handle a variety of coordinate and trajectory file formats. Most of the file formats are documented elsewhere.

- CHARMM crd coordinate file format
- PDB coordinate file format
- XMOL coordinate file format
- XYZ coordinate file format

CHARMM crd coordinate file format

This is modified from the CHARMM documentation:

The CARD file format is the standard means in CHARMM for providing a human readable and write able coordinate file. The format is as follows:

```
TITLE (a line starting with "*")
NATOM (I5)
ATOMNO RESNO   RES  TYPE  X     Y     Z   SEGID RESID Weighting
  I5    I5  1X A4 1X A4 F10.5 F10.5 F10.5 1X A4 1X A4 F10.5
```

The TITLE is a title for the coordinates,
Next comes the number of coordinates. If this number is zero or too large, the entire file will be read.
Finally, there is one line for each coordinate. ATOMNO gives the number of the atom in the file. It is ignored on reading. RESNO gives the residue number of the atom. It must be specified relative to the first residue in the PSF. The OFFSet option should be specified if one wishes to read coordinates into other positions.

It should also be remembered that for card images, residues are identified by RESIDUE NUMBER. This number can be modified by using the OFFSet feature, which allows coordinates to be read from a different PSF. Both positive and negative values are allowed. The RESId option will cause the residue number field to be ignored and map atoms from SEGID and RESID labels instead.
RES gives the residue type of the atom. RES is checked against the residue type in the PSF for consistency. TYPE gives the IUPAC name of the atom. The coordinates of an atom within a residue need not be specified in any particular order. A search is made within each residue in the PSF for an atom whose IUPAC name is given in the coordinate file.
The RESId option overrides the residue number and fills coordinates based on the SEGID and RESID identifiers in the coordinate file.

Example:

```
* MINIMIZED COORDINATES FOR NEW HAPT (+HYDRO) WITH CHANGED
NBONDS PARAM
*  DATE:    5/ 2/89     9:36: 3     CREATED BY USER: leif
*
   27
    1    1 OXAZ C1     0.74545   1.32903  -0.11231 OXAZ 1      0.00000
    2    1 OXAZ C2    -0.64926   1.28346  -0.04740 OXAZ 1      0.00000
    3    1 OXAZ C3    -1.30523   0.05117   0.00874 OXAZ 1      0.00000
    4    1 OXAZ C4    -0.56931  -1.13825   0.00015 OXAZ 1      0.00000
    5    1 OXAZ C5     0.82612  -1.08911  -0.06487 OXAZ 1      0.00000
    6    1 OXAZ C6     1.48348   0.14283  -0.12106 OXAZ 1      0.00000
    7    1 OXAZ C7    -1.24088  -2.42037   0.05806 OXAZ 1      0.00000
    8    1 OXAZ C8    -1.93473  -4.54946   0.13490 OXAZ 1      0.00000
    9    1 OXAZ C9    -3.12275  -3.64818   0.17032 OXAZ 1      0.00000
   10    1 OXAZ O10   -2.57445  -2.44388   0.11971 OXAZ 1      0.00000
   11    1 OXAZ N11   -0.78409  -3.68146   0.06378 OXAZ 1      0.00000
   12    1 OXAZ O12   -4.32004  -3.92503   0.23108 OXAZ 1      0.00000
   13    1 OXAZ C13   -2.04383  -5.88468   0.16814 OXAZ 1      0.00000
   14    1 OXAZ C14   -1.24454  -8.06631   0.17764 OXAZ 1      0.00000
   15    1 OXAZ C15    0.05231  -8.89906   0.13576 OXAZ 1      0.00000
   16    1 OXAZ O16   -0.91876  -6.66343   0.13302 OXAZ 1      0.00000
   17    1 OXAZ H17    1.25297   2.28231  -0.15574 OXAZ 1      0.00000
   18    1 OXAZ H18   -1.22177   2.20013  -0.04053 OXAZ 1      0.00000
   19    1 OXAZ H19   -2.38436   0.00998   0.05909 OXAZ 1      0.00000
   20    1 OXAZ H20    1.39521  -2.00810  -0.07153 OXAZ 1      0.00000
   21    1 OXAZ H21    2.56254   0.17885  -0.17130 OXAZ 1      0.00000
   22    1 OXAZ H22   -3.03327  -6.34232   0.22321 OXAZ 1      0.00000
   23    1 OXAZ H23   -1.86754  -8.32376  -0.67923 OXAZ 1      0.00000
   24    1 OXAZ H24   -1.78600  -8.28615   1.09803 OXAZ 1      0.00000
   25    1 OXAZ H25   -0.19625  -9.95972   0.16740 OXAZ 1      0.00000
   26    1 OXAZ H26    0.67508  -8.64671   0.99402 OXAZ 1      0.00000
```

```
  27    1 OXAZ H27    0.59561  -8.68111  -0.78372 OXAZ 1      0.00000
```

XMOL coordinate file format

Modified from the XMOL help files

XMOL(XYZ) datafiles specify molecular geometries using a Cartesian coordinate system. The XMOL(XYZ) format supports multi-step datasets. Each step is represented by a two-line "header," followed by one line for each atom.
The first line of a step's header is the number of atoms in that step. This integer may be preceded by whitespace; any- thing on the line after the integer is ignored. The second line of the header leaves room for a descriptive string. This line may be blank, or it may contain some information pertinent to that particular step, but it must exist, and it must be just one line long.
Each line of text describing a single atom must contain at least four fields of information, separated by whitespace: the atom's type (a short string of alphanumeric characters), and its x-, y-, and z-positions. Optionally, extra fields may be used to specify a charge for the atom, and/or a vector associated with the atom. If an input line contains five or eight fields, the fifth field is interpreted as the atom's charge; otherwise, a charge of zero is assumed. If an input line contains seven or eight fields, the last three fields are interpreted as the components of a vector. These components should be specified in angstroms.

The XMOL(XYZ) format doesn't contain connectivity information!

Example:

```
10
Input Geometry
          C     0.000000      0.000000      0.000000      0.000000
          H     1.070860      0.000000      0.000000      0.000000
          H     -0.357652     1.007862      0.000000      0.000000
          H     -0.357386     -0.503905     -0.874441     0.000000
          O     -0.476922     -0.672592     1.167050      0.000000
          C     -1.050823     -2.776752     3.226941      0.000000
          H     0.019804      -2.793463     3.212141      0.000000
          I     -1.458293     -4.270705     4.689464      0.000000
          H     -1.392643     -1.763433     3.233782      0.000000
          H     -1.428068     -3.273182     2.356576      0.000000
```

XYZ coordinate file format

This is a very simple coordinate format containing:

- The first line contains the number of atom records (lines)
- From the second record forward comes a record with:
    - Atom symbol
    - x coordinate
    - y coordinate
    - z coordinate

Example:

```
27
C1    0.74545   1.32903  -0.11231
C2   -0.64926   1.28346  -0.04740
C3   -1.30523   0.05117   0.00874
C4   -0.56931  -1.13825   0.00015
C5    0.82612  -1.08911  -0.06487
C6    1.48348   0.14283  -0.12106
C7   -1.24088  -2.42037   0.05806
C8   -1.93473  -4.54946   0.13490
C9   -3.12275  -3.64818   0.17032
O10   -2.57445  -2.44388   0.11971
N11   -0.78409  -3.68146   0.06378
O12   -4.32004  -3.92503   0.23108
C13   -2.04383  -5.88468   0.16814
C14   -1.24454  -8.06631   0.17764
C15    0.05231  -8.89906   0.13576
O16   -0.91876  -6.66343   0.13302
H17    1.25297   2.28231  -0.15574
H18   -1.22177   2.20013  -0.04053
H19   -2.38436   0.00998   0.05909
H20    1.39521  -2.00810  -0.07153
H21    2.56254   0.17885  -0.17130
H22   -3.03327  -6.34232   0.22321
H23   -1.86754  -8.32376  -0.67923
H24   -1.78600  -8.28615   1.09803
H25   -0.19625  -9.95972   0.16740
H26    0.67508  -8.64671   0.99402
H27    0.59561  -8.68111  -0.78372
```

Appendix 2

Plot file (plt) format

The plot files are regular 3D grid files for plotting of molecular orbitals, electron densities or other molecular properties. The plot files are produced by several programs. It is also possible to format/unformat plot files using the **pltfile** program in the utility directory.

It is also possible to produce plot files with external (own) programs. Produce first a formatted text file and use then the **pltfile** program to unformat the file for gOpenMol. The format for the plot files are very simple and a description of the format can be found elsewhere in this manual. gOpenMol can read binary plot files from different hardware platforms independent of the system type (little or big endian machines).

Format of the binary *.plt file

The *.plt grid data files are the main enginees for ploting isocontour surfaces. The *.plt files can either be generated directly by various programs inside the gOpenMol package or be imported from other programs.

Programs inside gOpenMol generating *.plt grid data files:

- The **probsurf** program generates the grid data for plotting Connolly type of surfaces.
- The **vss** program generates the grid data for plotting electrostatic potentials.

To assist in moving the binary *.plt files between different hardware platforms the **pltfile** program is included. Using the program it is possible to format a binary grid file and move it to an other platform and unformat the file again. The program can of course also be used interface the grid data from other programs with gOpenMol.

The *.plt file binary and formatted file formats are very simple but please observe that unformatted files written with a FORTRAN program are not pure binary files because there are file records between the values while pure binary files do not have any records between the values. gOpenMol should be able to figure out if the file is pure binary or FORTRAN unformatted but it is not very well tested.

Binary *.plt (grid) file format

Record number and meaning:
#1: Integer, rank value must always be = 3
#2: Integer, possible values are 1 ... 50. This value is not used but it can be used to define the type of surface!

Values used (you can use your own value between 1... 50):

1. : VSS surface
2. : Orbital/density surface
3. : Probe surface

200: Gaussian 94/98

201: Jaguar

202: Gamess

203: AutoDock

204: Delphi/Insight

205: Grid

Value 100 is reserved for grid data coming from OpenMol!

#3: Integer, number of points in z direction
#4: Integer, number of points in y direction
#5: Integer, number of points in x direction
#6: Float, zmin value
#7: Float, zmax value
#8: Float, ymin value
#9: Float, ymax value
#10: Float, xmin value
#11: Float, xmax value
#12 ... Float, grid data values running (x is inner loop, then y and last z):

1. Loop in the z direction
2. Loop in the y direction
3. Loop in the x direction

The formatted (the first few lines) file can look like:

```
3 2
65 65 65
-3.300000e+001 3.200000e+001 -3.300000e+001 3.200000e+001 -3.300000e+001
3.200000e+001
-1.625609e+001 -1.644741e+001 -1.663923e+001 -1.683115e+001 -
1.702274e+001 -1.721340e+001
-1.740280e+001 -1.759018e+001 -1.777478e+001 -1.795639e+001 -
1.813387e+001 -1.830635e+001
...
```

Formatted *.plt (grid) file format

Line numbers and variables on the line:
#1: Integer, Integer. Rank and type of surface (rank is always = 3)
#2: Integer, Integer, Integer. Zdim, Ydim, Xdim (number of points in the z,y,x directions)
#3: Float, Float, Float, Float, Float, Float. Zmin, Zmax, Ymin, Ymax, Xmin,Xmax (min and max values)
#4: ... Float. Grid data values running (x is inner loop, then y and last z) with one or several values per line:

1. Loop in the z direction
2. Loop in the y direction
3. Loop in the x direction

A file in this format can then be converted into binary with the pltfile program.

For an example of what a formatted *.plt file can look like please look above.
If you are a programmer please look at the included utility programs for the code for doing the reading and writing of *.plt files

Appendix 3

The format of the colour file

The file is located at: data/colour_table.data
The first line in the colour file is a comment. The following colour table (a small part of it in fact) contains the rgb code and the name defined for that rgb code. You can add your own colour to the table by editing the file. The colour table file is also used by the colour browsers. As it is now implemented only the first 200 colours are included in the browser.
The rgb codes and their names:

```
* Colour table to be used by gOpenMol (1997-11-21)
255 255 255     white
  0   0   0     black
255   0   0     red
  0 255   0     green
  0   0 255     blue
255   0 255     magenta
255 255   0     yellow
192 192 192     gray
238 130 238     violet
165  42  42     brown
  0 255 255     cyan
```

255 192 48	bright mustard
134 26 26	dark firebrick
255 128 144	miami pink
80 255 255	miami turquoise
80 40 30	dark brown
128 0 0	dark red
255 250 250	snow
248 248 255	ghost white
245 245 245	white smoke
255 239 213	papaya whip
255 218 185	peach puff
255 228 181	moccasin
255 248 220	cornsilk
255 255 240	ivory
255 250 205	lemon chiffon
245 255 250	mint cream
240 255 255	azure
240 248 255	alice blue
230 230 250	lavender
255 240 245	lavender blush
105 105 105	dim gray
112 128 144	slate gray
211 211 211	light grey
25 25 112	midnight blue
0 0 128	navy blue
175 238 238	pale turquoise
0 206 209	dark turquoise
0 100 0	dark green
220 220 220	gainsboro
253 245 230	old lace
250 240 230	linen
255 235 205	blanched almond
255 228 196	bisque
255 245 238	seashell
240 255 240	honeydew
105 105 105	dim grey

. .
.

The gOpenMol GUI uses sliders for dynamic colour selection but colour names can be used in the line command mode.

The format of the data/atom_param.data file

This file contains data for the display of atoms and some other useful parameters as the Lennard-Jones parameters.
The following data is taken from APPENDIX B of the QUANTA Reference Manual 1990.
The titles of the columns in the atom_param.data file are:
(1): Atom type number (type).

(2): Bonding radius (bndrad).

(3): van der Waals spheres radius (vdwrad).

(4): Sphere radius (plurad) in plots. (Not used in gOpenMol).

(5): Value (global) is used in global search for bonds.

(6,7): Value (emin and rmin) used in the calculation of van der
Waals and electrostatic energy.

(8): Atom polarizabilities (patom).

(9): The atom is either a hydrogen bond acceptor (A), donor
(D or E), or not hydrogen bonded (N)

(10): The CHARMm atom type (atype).

(11): Atom mass (mass).
Example:
(1) (2) (3) (4) (5) (6) (7) (8) (9) (10) (11)

Type bndrad vdwrad plurad global emin rmin patom hbond atype mass

1 0.40 0.95 0.100 F -0.0498 0.8 0.044 D H 1.008
The parameters in QUANTA 3.0 are similar to those used in Version 21 of CHARMm. The emin rmin values differ for HA and all metals; bndrad differs for hydrogens. Atoms with Mxx are metals; atoms with Xxx are halogens.
If you wish to extend the parameter file to include your own atom types, it is recommended to use atom numbers greater than 300.

The format of the atomXXX.dic files

The format of the atom.dic file is as following. The first position (1) is the residue name, the next position (2) is the atom name, third position (3) is the atom type according to the CHARMm type and as the last position (4) comes the atom charge.

The residue and atom names are restricted to 4 characters, atomic type is an integer and the charge is a floating point number. The user can also use the symbols '*' and '?' for or in the residue and atom names where '*' matches any characters to the end and '?' matches any character at that particular point. Your own molecules can be added to the end of the file before the default atom types.

Example of an atom dictionary file:

```
*Amino acid dictionary, polar hydrogens only, CHARMm20.3 parameters (From QUANTA)
GLY CA 12 0.100
PRO N 33 -0.200
* N 32 -0.350
* NT 36 -0.100
* H 1 0.250
* HT?? 2 0.350
* CA 11 0.100
* C 14 0.550
* O 40 -0.550
* OT 44 -0.550
* CT1 10
* CT2 10
ALA CB 13 0.000
ARG CB 12 0.000
ARG CG 12 0.000
ARG CD 12 0.100
ARG NE 32 -0.400
ARG HE 1 0.300
ARG CZ 14 0.500
ARG NH? 37 -0.450
ARG HH?? 2 0.350
ASN CB 12 0.000
ASN CG 14 0.550
ASN OD? 40 -0.550
ASN ND? 32 -0.600
ASN HD?? 1 0.300
ASP CB 12 -0.160
ASP CG 14 0.360
ASP OD? 43 -0.600
ASP HD? 2 1.000
CYS CB 12 0.019
CYS SG 70 -0.019
CYS HG 1 0.000
GLN CB 12 0.000
GLN CG 12 0.000
GLN CD 14 0.550
GLN OE? 40 -0.550
GLN NE? 32 -0.600
GLN HE?? 1 0.300
GLU CB 12 0.000
GLU CG 12 -0.160
GLU CD 14 0.360
GLU OE? 43 -0.600
GLU HE? 2 1.000
HIS CB 12 0.000
HIS CG 21 0.100
HIS ND? 34 -0.400
HIS HD? 1 0.300
HIS CD? 23 0.100
HIS CE? 23 0.300
HIS NE? 34 -0.400
ILE CB 11 0.000
ILE CG1 12 0.000
ILE CG2 13 0.000
ILE CD? 13 0.000
LEU CB 12 0.000
LEU CG 11 0.000
LEU CD? 13 0.000
LYS CB 12 0.000
LYS CG 12 0.000
LYS CD 12 0.000
LYS CE 12 0.250
LYS NZ 36 -0.300
LYS HZ? 2 0.350
MET CB 12 0.000
MET CG 12 0.060
MET SD 70 -0.120
MET CE 13 0.060
PHE CB 12 0.000
PHE CG 22 0.000
PHE CD? 24 0.000
PHE CE? 24 0.000
PHE CZ 24 0.000
PRO CB 12 0.000
PRO CG 12 0.000
PRO CD 12 0.100
SER CB 12 0.250
SER OG 45 -0.650
SER HG 1 0.400
THR CB 11 0.250
THR OG? 45 -0.650
THR HG? 1 0.400
THR CG? 13 0.000
TRP CB 12 0.000
TRP CG 21 -0.030
TRP CD1 23 0.060
TRP CD2 26 0.100
TRP NE1 34 -0.360
TRP HE1 1 0.300
TRP CE2 26 -0.040
TRP CE3 24 -0.030
TRP CZ? 24 0.000
TRP CH? 24 0.000
TYR CB 12 0.000
TYR CG 22 0.000
TYR CD? 24 0.000
TYR CE? 24 0.000
TYR CZ 22 0.250
TYR OH 45 -0.650
TYR HH 1 0.400
VAL CB 11 0.000
VAL CG? 13 0.000
WAT O??? 47 -0.834
WAT H??? 4 0.417
H2O O??? 47 -0.834
H2O H??? 4 0.417
SOL O??? 47 -0.834
SOL H??? 4 0.417
HOH O??? 47 -0.834
HOH H??? 4 0.417
DOD O??? 47 -0.834
DOD D??? 4 0.417
TIP O??? 47 -0.834
TIP H??? 4 0.417
TIP3 OH2 47 -0.834
TIP3 H1 4 0.417
TIP3 H2 4 0.417
NAP NA1 81 1.000
#
# New molecule definitions come here
#
** N??? 32
** C??? 10
** S??? 70
** HT?? 1
** H??? 3
** O??? 45
** P??? 60
** MFE? 86
** ZN?? 87
** MCA? 84
** ???? 499
```

The user can easily add his/her favourite molecules to the end of this file before the default atom types. There are two files supplied atom.dic atom types for amino acids with polar hydrogens and atomh.dic for amino acids with all hydrogen atoms. The files can be found in directory .../data. These files are made from files of the QUANTA package by MSI.

The format of the atom_conversion.data file

The purpose if this file is to control the mapping between an atom name in a coordinate file with the real atom type and symbol. For example coordinate files can contain the atom symbol CA that can be mapped to an alpha carbon (C) or calcium (Ca). The mapping works now only against the CHARMM force field. Most likely it can also be used for other force fields but then another atom_param.data file has to be used.

Field:

1 Real atom symbol or atom name as used in the periodic table of elements.
2 Name used in the coordinate file.
3 Index to the atom type in a force field (CHARMM)
4 Colour to be used for this atom.
# 5 default basis set for this atom.
.
Example (atom_conversion.data):

```
#
# Leif Laaksonen (CSC) 1994
#
# This file contains information about atom conversion
#
# (1) Real symbol of an atom and the second
# (2) Name in which the atom can be found in a data file.
# (3) Index to the atom type in a force field (CHARMM)
# (4) Colour to be used for this atom
# (5) default basis set for this atom
#
# Example:
# O   OH   47 red    OxygenBS     # Oxygen in OH
# Means that an atom label OH is equivalent to an oxygen (O) atom in OH
Ar   AR  170 cyan   ArgonBS
B    B     7 cyan   BoronBS
C    C    14 green  CarbonBS       # default carbon atom
H    H     1 white  HydrogenBS     # default hydrogen atom
H   1HA    1 white  HydrogenBS     # default hydrogen atom
```

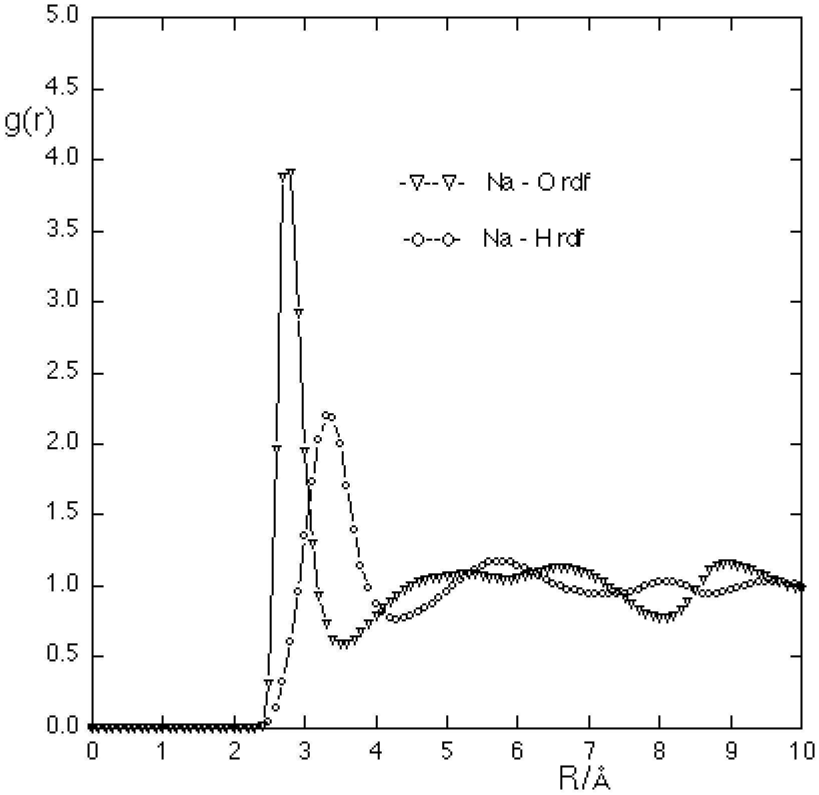
```
H   1HB    1 white  HydrogenBS     # default hydrogen atom
H   1HG    1 white  HydrogenBS     # default hydrogen atom
H   1HD    1 white  HydrogenBS     # default hydrogen atom
H   1HE    1 white  HydrogenBS     # default hydrogen atom
H   1HZ    1 white  HydrogenBS     # default hydrogen atom
H   1HH    1 white  HydrogenBS     # default hydrogen atom
H   2HA    1 white  HydrogenBS     # default hydrogen atom
H   2HB    1 white  HydrogenBS     # default hydrogen atom
H   2HG    1 white  HydrogenBS     # default hydrogen atom
H   2HD    1 white  HydrogenBS     # default hydrogen atom
H   2HE    1 white  HydrogenBS     # default hydrogen atom
H   2HZ    1 white  HydrogenBS     # default hydrogen atom
H   2HH    1 white  HydrogenBS     # default hydrogen atom
He   He  168 cyan   HeliumBS
LP   LP1   5 violet **NO BS**      # Handle LP1, LP2, KP3 and LP4
LP   LP2   5 violet **NO BS**      # Handle LP1, LP2, KP3 and LP4
LP   LP3   5 violet **NO BS**      # Handle LP1, LP2, KP3 and LP4
LP   LP4   5 violet **NO BS**      # Handle LP1, LP2, KP3 and LP4
N    N    31 blue   NitrogenBS     # default nitrogen atom
O    O    40 red    OxygenBS       # default oxygen atom
O    OH   47 red    OxygenBS
P    P    60 yellow PhosphorousBS
S    S    70 yellow SulphurBS
At   XAT 131 brown  AstatineBS
Br   XBR  94 brown  BromineBS
Cl   XCL  93 brown  ChlorineBS
Xe   XE  172 violet XenonBS
F    XF   92 brown  FluorineBS
I   XI   95 brown  IodineBS
Ac   MAC 126 magenta ActiniumBS
Ag   MAG 111 magenta SilverBS
Al   MAL  91 magenta AluminumBS
Am   MAM 159 magenta AmericiumBS
As   MAS 101 magenta ArsenicBS
Au   MAU 119 magenta GoldBS
Ba   MBA 115 magenta BariumBS
Be   MBE   6 magenta BerylliumBS
Bi   MBI 122 magenta BismuthBS
Bk   MBK 161 magenta BerkeliumBS
Ca   MCA  84 magenta Calcium BS
Cd   MCD 112 magenta CadmiumBS
Ce   MCE 124 magenta CeriumBS
Cf   MCF 162 magenta CaliforniumBS
Cm   MCM 160 magenta CuriumBS
Co   MCO  99 magenta CobaltBS
Cr   MCR  98 magenta ChromiumBS
```

Cs   MCS 89 magenta CesiumBS
Cu   MCU 96 magenta CopperBS

The radial distribution function (RDF) format

This is a simple example of what a radial distribution function (RDF) could look like.

Figure. The radial distribution function for the Na^+ atom and the oxygen and hydrogen atoms of water.



The format of the RDF function is:

#1: The radial value from the selected atom.
#2: The RDF value.

Example:

```
0.000000  0.000000
0.100000  0.000000
0.200000  0.000000
0.300000  0.000000
0.400000  0.000000
0.500000  0.000000
```

```
0.600000  0.000000
0.700000  0.000000
0.800000  0.000000
0.900000  0.000000
1.000000  0.000000
1.100000  0.000000
1.200000  0.000000
1.300000  0.000000
1.400000  0.000000
1.500000  0.000000
1.600000  0.000000
1.700000  0.000000
1.800000  0.000000
1.900000  0.000000
2.000000  0.000000
2.100000  0.000000
2.200000  0.000000
2.300000  0.000000
2.400000  0.352277
2.500000  1.300772
2.600000  3.312277
2.700000  5.033111
2.800000  4.165461
2.900000  2.429918
3.000000  2.500526
3.100000  1.918060
3.200000  0.600617
3.300000  1.319026
3.400000  1.421342
3.500000  1.006800
3.600000  0.317464
3.700000  0.751898
3.800000  0.713348
3.900000  0.813226
4.000000  0.902492
…
```

Appendix 4

The input format to the orbital and electron density program DENSITY.

The program accepts now only wave functions from the ICON8 program. The default name for the wave function file is fort.7.
Input lines:
1) Two Title lines.

2) Name of the input file to the ICON8 program.
   This file contains the atom coordinates. The coordinates
   and the atom names can also be directly included here.

3) This line defines the size of the box by defining
   the x-min, x-max, y-min, y-max, z-min and z-max.

4) This line defines the number of points in the x, y and z directions.

5) The first number defines t he orbital number to be calculated
   and the second number gives the type: 0 = orbital plot, 1 = orbital
   density and 2 = total elctron density. If the total electron
   density is defined the first number (the orbital number) has no meaning.
Example:

```
****** test input *******
****** test input *******
@icon8.inp
fort.7
-8.3200 6.5625 -13.9597 6.2823 -2.7837 2.0980
50 50 50
41 0
```

If the input is made by gOpenMol the file looks like the following:

```
Default title for: DENSITY
Default title for: DENSITY
6
0.000000 0.670000 0.000000 C
0.000000 -0.670000 0.000000 C
0.926650 1.205000 0.000000 H
-0.926650 1.205000 0.000000 H
0.926650 -1.205000 0.000000 H
-0.926650 -1.205000 0.000000 H
fort.7
-3.0 3.0 -3.0 3.0 -3.0 3.0
30 30 30
```

6 0
The main difference is now that the atom names and coordinates are included in the input file.
The program can be run with the command
density < density.inp > density.out.
The program has the following options:
-ofile.name
Write the orbital or density mesh into file file.name instead of the default file orbital.plt.

Probe input format to the surface program ProbeSurf

The probe surface program PROBESURF generates a mesh with grid values from 0 to 100. The value 100 is at the vdW value and value 0 is at the vdW + max. probe diameter value.
It is possible to generate the surface for different probe values. The method is based on the article by R. Voorintholt et al. (Voorintholt R., Koster M.T. , Vegter G. , Vriend G. and Hol W.G.J., "A very fast program for visualizing protein surfaces, channels and cavities", J. Mol. Graphics 7 (1989) 243-245).

The original article describes a method based on distance squared (r^2) but the program can now also calculate a value based on the distance.

Program line command options:

```
probsurf -iinput.file -ofile.name -llimits -m1|2 -pfile.name < input.file > output.file

        -h this text
        -i defines an input file, name can also be direct using '<'
        -o defines the name of the contour file (Default 'probesurf.plt')
        -l defines the limits as:
                numx,numy,numz,xmin,xmax,ymin,ymax,zmin,zmax
                where numx,numy,numz are the number of points in
                the x,y and z directions and the rest are the coordinate
                limits of the surface
        -m defines either squared (2) or direct (1), given as –m2 or –m1
        -p defines a log file name
```

The input format to the probe surface program PROBESURF.

Input lines:
1) Two Title lines.

2) Number of atoms.

3) All this on one line:

1: Running atom number.
2: Segment name.
3: Residue name.
4: Atom name.
5: X coordinate.
6: Y coordinate.
7: Z coordinate.
8: van Der Waals radius.

4) This line defines the size of the box by defining the x-min, x-max, y-min, y-max, z-min and z-max.

5) This line defines the number of points in the x, y and z directions.

6) The maximum probe radius.

Example:

```
Default title for: PROBESURF
Default title for: PROBESURF
27
1 OXAZ OXAZ C1 0.745450 1.329030 -0.112310 1.770000
2 OXAZ OXAZ C2 -0.649260 1.283460 -0.047400 1.770000
3 OXAZ OXAZ C3 -1.305230 0.051170 0.008740 1.770000
4 OXAZ OXAZ C4 -0.569310 -1.138250 0.000150 1.770000
5 OXAZ OXAZ C5 0.826120 -1.089110 -0.064870 1.770000
6 OXAZ OXAZ C6 1.483480 0.142830 -0.121060 1.770000
7 OXAZ OXAZ C7 -1.240880 -2.420370 0.058060 1.770000
8 OXAZ OXAZ C8 -1.934730 -4.549460 0.134900 1.770000
9 OXAZ OXAZ C9 -3.122750 -3.648180 0.170320 1.770000
10 OXAZ OXAZ O10 -2.574450 -2.443880 0.119710 1.520000
11 OXAZ OXAZ N11 -0.784090 -3.681460 0.063780 1.600000
12 OXAZ OXAZ O12 -4.320040 -3.925030 0.231080 1.520000
13 OXAZ OXAZ C13 -2.043830 -5.884680 0.168140 1.770000
14 OXAZ OXAZ C14 -1.244540 -8.066310 0.177640 1.770000
15 OXAZ OXAZ C15 0.052310 -8.899060 0.135760 1.770000
16 OXAZ OXAZ O16 -0.918760 -6.663430 0.133020 1.520000
17 OXAZ OXAZ H17 1.252970 2.282310 -0.155740 0.900000
18 OXAZ OXAZ H18 -1.221770 2.200130 -0.040530 0.900000
19 OXAZ OXAZ H19 -2.384360 0.009980 0.059090 0.900000
20 OXAZ OXAZ H20 1.395210 -2.008100 -0.071530 0.900000
21 OXAZ OXAZ H21 2.562540 0.178850 -0.171300 0.900000
22 OXAZ OXAZ H22 -3.033270 -6.342320 0.223210 0.900000
23 OXAZ OXAZ H23 -1.867540 -8.323760 -0.679230 0.900000
24 OXAZ OXAZ H24 -1.786000 -8.286150 1.098030 0.900000
25 OXAZ OXAZ H25 -0.196250 -9.959719 0.167400 0.900000
26 OXAZ OXAZ H26 0.675080 -8.646710 0.994020 0.900000
27 OXAZ OXAZ H27 0.595610 -8.681110 -0.783720 0.900000
```

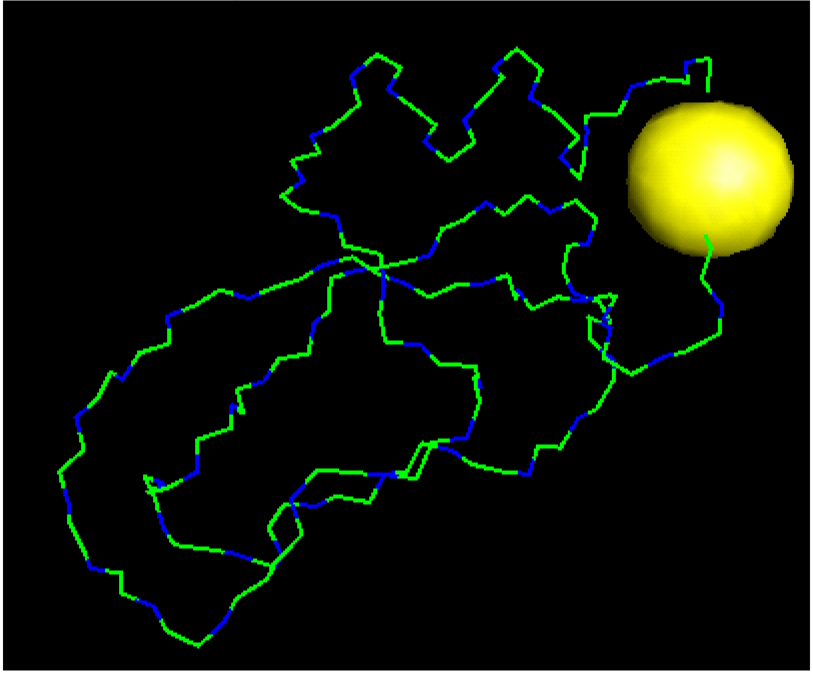
```
-15.242029 15.242029 -15.242029 15.242029 -15.242029 15.242029
60 60 60
1.900000
```

The program has the following options:
-ofile.name

Write the probe surface mesh into file file.name instead of the default file probesurf.plt.
The program probsurf is located in the bin/ directory.
When you display a probsurf generated *.plt grid file using the isocontour display engine remember that the display value range is between 100. and 0.0

The atom trace facility in the trajectory analysis has a tool to export the atom trace for an atom or atoms into the probsurf program to generate the volume inside which an atom is moving.

1. Import the opt4pti.crd file from the demo/ directory.
2. Import the dyn4pti.dcd trajectory file.
3. Open the "Trajectory ➔ Trace" widget. Write into the "Residue:" field 1 and into the "Atom:" field CA and press the "Apply" button.
4. Click the "Display:" to "On".
5. Write into the line command: **ptrace write probsurf c:/temp/test.inp**
6. Run the probsurf program with the c:/temp/test.inp to produce a plt file.
7. Import this plt file using the contour facility and display this Connolly type of surface to display the volume inside which the CA has moved during the simulation. Using a probe of about 1 Å shows the following Connolly surface:



The electrostatic potential program VSS

The VSS program calculates the electrostatic potential created by the electronic distribution and the nuclei of a molecule in the surrounding space. The electronic distribution is supposed to be obtained from the ICON8 program. It correaponds to approximation II of "Molecular electrostatic potentials: comparison of ab initio and CNDO results" by C. Giessner-Prettre and A. Pullman, Theoret. Chim. Acta (Berl.) (1972) 25, 83-88. This program is a direct translation of the fortran version into c. The input is as follows :

(1) Title.

(2) Title.

(3) Title.

(4) NAT = number of atoms

NH = number of hydrogens

IU = 0 or 1.

IU = 0 the coordinates are given in atomic units

IU = 1 the coordinates are given in Angstrom units

(5) the following NAT cards

X, Y and Z coordinates for atom i

IZ = number of valence electrons

(The hydrogen atoms must be entered last).

(6) the following NAT cards

gross atomic population and screening constant for atom i

(7) Integer always 2 (not used)

(8) Xmin, Xmax, Ymin, Ymax, Zmin and Zmax values to define the box.

(9) Number of points in the X-, Y- and Z-directions.

Example:

```
*******NO_INFO*******
*******NO_INFO*******
ETHYLENE
6 4 1
0.0000 0.6700 0.0000 4
```

```
0.0000 -0.6700 0.0000 4
0.9267 1.2050 0.0000 1
-0.9267 1.2050 0.0000 1
0.9267 -1.2050 0.0000 1
-0.9267 -1.2050 0.0000 1
4.1027 1.6250
4.1027 1.6250
0.9486 1.3000
0.9486 1.3000
0.9486 1.3000
0.9486 1.3000
2
-0.9267 0.9267 -1.2050 1.2050 0.0000 0.0000
40 40 40
```

The program has the following options:
-ofile.name
Write the isoenergy mesh into file file.name instead of the default file vsscont.plt.

Currently all programs except the ICON8 program are written in c. Dynamic memory allocation has been used in all the c programs. ICON8 currently accepts systems up to 1000 atoms, but can easily be modified to accept more atoms. The problem with the ICON8 code is the memory. To calculate system up to 1000 atoms needs some 150 Mbytes of memory. The other programs used dynamic memory allocation and use very little memory.

The internal format of a vector data file

The file is a simple formatted flat text file with the following structure.

Line #1:
- 1: the number 3 showing it is a 3 dimensional plot.
- 2: the next number shows the data type

Line #2:
- 1: number of points in the x-direction
- 2: number of points in the y-direction
- 3: number of points in the z-direction

Line #3:
- 1: min and max x coordinate
- 2: min and max y coordinate
- 3: min and max z coordinate

Line #4 – n:
- 1: x coordinate
- 2: y coordinate
- 3: z coordinate
- 4: length of vector in the x direction
- 5: length of vector in the y direction
- 6: length of vector in yje z direction

If the file contains just random vectors the first five lines have no meaning. They are not processed by gOpenMol in the current imnplementation.

Example of a flat vector file:

```
3 200
41 41 41
-2.645886 2.645886
-2.645886 2.645886
-2.645886 2.645886
-2.645886 -2.645886 -2.645886 0.000000 0.000000 0.000000
-2.513591 -2.645886 -2.645886 0.000000 0.000000 0.000000
-2.381297 -2.645886 -2.645886 0.000000 0.000000 0.000000
-2.249003 -2.645886 -2.645886 0.000000 0.000000 0.000000
-2.116709 -2.645886 -2.645886 0.000000 0.000000 0.000000
.
.
.
```

Appendix 5

The tutorial material is available in the help/tutorials/scottanderson1 directory.

The material is best viewed through the web.

Tutorials

- Scott Anderson (anderson@chemistry.chem.utah.edu):
  1. Visualizing Molecules with gOpenMol (version 2.2) from US.

     http://www.chem.utah.edu/chemistry/faculty/anderson/gopen.html
  2. Visualizing Molecules with gOpenMol (version 2.2) from Finland.

     http://www.csc.fi/gopenmol/tutorials/scott/

     - The tutorial as a zip file with the included binary files for Windows 95/98/Me/NT/2000.

       http://www.csc.fi/gopenmol/tutorials/scott/tutorial.zip
- Kevin Boyd at the University of New Orleans
  1. Extending gOpenMol in the Windows Environment using plugins.

     http://137.30.117.235/chem/gopenmol.htm

Appendix 6

The PDF file reader saves now the secondary structure information in a PDF file into tcl variables. This information is available for post processing through the following variables:

1. The array variable **gomSecondaryStructureRecords** contains the number of records of information available for the available structures.
   Example: gomSecondaryStructureRecords(1) contains the number of secondary structures available for structure number 1.
2. The array variable **gomSecondaryStructureRecord** contains the actual records.
   Example: gomSecondaryStructureRecord(1,3) contains the third secondary structure record for structure number 1.

Example: The 4pti.pdb (PROTEINASE INHIBITOR (TRYPSIN) 27-SEP-82   4PTI) file contains the following secondary structure records.

```
HELIX    1  H1 SER     47  GLY     56  1                                   4PTI  76
SHEET    1  S1 2 ALA    16  ALA    25  0                                   4PTI  77
SHEET    2  S1 2 GLY    28  GLY    36 -1                                   4PTI  78
```

After reading in the 4pti.pdb file gOpenMol will have the following information saved.

gomSecondaryStructureRecords(1):   3

gomSecondaryStructureRecord(1,1):

```
”HELIX    1  H1 SER     47  GLY     56  1                                   4PTI  76″
```

gomSecondaryStructureRecord(1,2):

```
”SHEET    1  S1 2 ALA    16  ALA    25  0                                   4PTI  77″
```

gomSecondaryStructureRecord(1,3):

```
”SHEET    2  S1 2 GLY    28  GLY    36 -1                                   4PTI  78″
```

This feature is used in the protein secondary structure cartoon display facility.

Appendix 7

List of Tcl variables defined/used by gOpenMol

Variable	Usage
gomAtomHitList	After commands involving choosing/selecting atoms the atoms selected into the operation are saved in the variable gomAtomHitList. From this variable it is possible to post-process the command by seeing which atoms have been included in the command.
gomAtomHitListActive	This Tcl variable shows if there are values available in the gomAtomHitList.
gomAtomLabelFont	Controls the default text font used when writing atom label (info) into the graphics window. The default valus is “BITMAP_HELVETICA_12”. The other options are BITMAP_8_BY_13, BITMAP_9_BY_15, BITMAP_TIMES_ROMAN_10, BITMAP_TIMES_ROMAN_24, BITMAP_HELVETICA_10, BITMAP_HELVETICA_12 and BITMAP_HELVETICA_18.
gomAtomLabelNumberLimit	By default gOpenMol has two different display types of atom labels. The simple is just the atom label from the coordinate file. The extended mode shows also the segment name, residue number and running atom index number. To display this information gOpenMol either shows the values through a C-kernel routine or through a link to Tcl. The Tcl mode is quite slow so it is not usable for a big number of atom labels. By default gOpenMol shows the label information through the Tcl interface if the numbers of labels to be shown are less than “gomAtomLabelNumberLimit”. By changing this value you can manipulate which routine gOpenMol is using. Default value is 200.
gomAtomLabelString	In the simplest and default display type (just plain atom label) the information is generated through the operation “[show atom atomname $j $i]”. By changing this further information can be included on the display. The default value is “{[show atom atomname $j $i]}”. Observe the curly brackets!

gomAtomLabelString	In the extensive display type. The information is generated through the operation "[show atom segment $j $i]:[show atom resnumber $j $i]:[show atom atomname $j $i]($j)". By changing this further information can be included on the display. The default value is "{[show atom segment $j $i]:[show atom resnumber $j $i]:[show atom atomname $j $i]($j)}". Observe the curly brackets!
gomCylinderArrowControl	Controls the way the cylinder arrow is displayed. The first value gives the hat/cylinder radius ratio and the second defines the arrow length/hat length ratio. Example: set gomCylinderArrowControl "1.3 0.7".
gomCovalentDelta	Controls the Δ value when calculating atom connections (bonds). Value is float and default value 0.4 Angstrom.
gomCurrentFrameProperty	This variable defines the coordinates, colour and font of frame number showed on the screen while looping through the frames. Default value is "{0.75 0.9 red BITMAP_HELVETICA_12}".
gomDefaultContourOpaque	It is possible to define a default opaque value for solid surfaces. The default value is "1.0".
gomDefaultContourSmooth	This variable controls if the best isocontour smoothing is used by default or not. Default value is "0" which means that the best smoothed surface will not be shown by default.
gomDefaultContourType	This variable controls the display type of isocontour surfaces. The display type can be either "mesh" or "solid". The default value is "solid".
gomDefaultGradienColor	Can be given as a colour name like "blue" or as three floating pointvalues ({0.0 0.0 1.0}). Default is red.
gomDisplayLoopSleep	Showing trajectories with molecules in the stick display mode results in a display where the frames are displayed too fast. To slow down the display frequency it is possible to define a "sleep time" in milliseconds before the next frame is displayed. This means that the frame will be displayed at least "gomDisplayLoopSleep" milliseconds. The default value is "0".

gomHydrogenBondingParams	This is still under development!
gomJPEGquality	Controls the JPEG saving quality. The value is integer and default value is 75.
gomLicoCylinder	Controls the licorice cylinder default value. Value is float and default value is 0.3 Angstrom.
gomLicoSphere	Controls the licrice sphere default value. Value is float and default value is 0.3 Angstrom.
gomMonitorDistancePrecision	Controls the distance output format (precision). Bu default the value is "%.2f".
gomMonitorAnglePrecision	Controls the angle output format (precision). Bu default the value is "%.2f".
gomMonitorTorsionPrecision	Controls the torsion output format (precision). Bu default the value is "%.2f".
gomTextFont	Controls the default text font used when writing text into the graphics window. The default valus is “BITMAP_HELVETICA_12”. The other options are BITMAP_8_BY_13, BITMAP_9_BY_15, BITMAP_TIMES_ROMAN_10, BITMAP_TIMES_ROMAN_24, BITMAP_HELVETICA_10, BITMAP_HELVETICA_12 and BITMAP_HELVETICA_18.
gomVectorDefaultType	Controls the array type when displaying a vector property (density gradient vector). The default is an array of type line when the value is 0. If one wants to show a solid array there one integer and one float values give. One for the array type (1) and a second defing the solid array cylinder radius (float).
lulCalculateAtomConnectivity	Controls the calculation of the connection table. Value can be “yes” or “no”. Default value is “yes”.
lulDoViewTransGlobalLocal	This variable controls if the viewing transformation is done when switching between Global/Local transformations. Default value is “0” which means no Viewing transformation is done.
lulInputView	This variable controls the feature to apply the viewing transformation to the atom coordinates when exporting the coordinates. The default value is “0” which means that the original coordinates are written out.
lulTraceTransformationValue	This variable controls the switching between a global (0) and a locan (1) trabsformation mode. Default value is “0” and meaning a global transformation mode.

All these variables can be changed in the gopenmol_guirc.tcl file in the data directory.

Examples of using the variables.

Example #1:

The font used to controled the atom label font is controlled by the tcl variable “gomAtomLabelFont”. The default font definition is:

```
set gomAtomLabelFont BITMAP_HELVETICA_12
```

The other possible values are:

BITMAP_8_BY_13, BITMAP_9_BY_15, BITMAP_TIMES_ROMAN_10, BITMAP_TIMES_ROMAN_24, BITMAP_HELVETICA_10, BITMAP_HELVETICA_12 and BITMAP_HELVETICA_18.

Example #2:

The precision of the monitor values shown in the graphics window are controlled by the tcl variables:

```
set gomMonitorDistancePrecision      "%.2f"
set gomMonitorAnglePrecision         "%.2f"
set gomMonitorTorsionPrecision       "%.2f"
```

It's easy to change the precision by changing the formatting value.

```
set gomMonitorDistancePrecision      "%.4f"
set gomMonitorAnglePrecision         "%.4f"
set gomMonitorTorsionPrecision       "%.4f"
```

This would change the shown precision to 4 significant digits.

Appendix 8

New features in version 2.2

This version has gone through a quite comprehensive updating and therefore this list of new features is rather long. Hopefully you will find time to read it through.

Atom connectivity

- Citeria to determine atom connectivity can be customised. A bond between two atoms is created if the distance between the atoms is equal to sum of covar radius of the atoms (a small delta is accepted). The covar radius can be changed temporarily from *View* - *Periodic table* and permanently by editing the file data/element.data.

Atom labels

- New atom label type has been added. In addition to the previeously available labels atom name and full name it is now possible to display residue name and number. This feature is specially for protein structure analysing.

  Go to *View* - *Atom label(s)...* to test this.

Atom picking

- Atom picking has got quite a few new features. The atom picking mode can be turned on and off by marking the respective radio button. When the mode is on, just click on an atom in order to pick it. Atom picking can also be used by pressing down Ctrl and clicking with left mouse button (LMB) on top of the atom one wants to select. This feature can be utilised in the following way when you want to modify an atom or an group of atoms.

  - A single atom can be picked and it can be dragged and dropped to a text widget by picking the atom pressing down LMB and dragging the cursor to the text widget.
  - If several atoms are picked, they can all be dragged and dropped to a text widget by pressing down LMB on an empty space in the 3D window and dragging the cursor to the text widget.

- The atoms can be unpicked by holding down Ctrl-button and clicking at the same time with LMB over the picked atom, if the "Pick Atoms" radio button is turned off. If "Pick atoms" is turned on, just click with LMB over the picked atom to unpick it. You can also unpick all atoms by clicking the **Unpick all**-button.

Atom selection

- Selected atoms are marked using cubes to separate them from picked atoms, which are marked with spheres. Atoms can be unselected by clicking **Unselect all** -button in "Select atoms"-window.

  Go to *Edit* - *Select...* to test this

Atom tree

- A new level has added to the Atom tree. This applies only to protein structures, where now the residues will be listed first and atoms in the residues are listed under every residue.

- Structure and segment names, residue names and numbers and atom names can be modified by clicking once on the item and then clicking it again.

  NB: *This does not mean double-clicking! There has to be a short delay between the two clickings.*

  A box will appear around the selected item and a cursor will start to blink. After the editing has been done, press enter and the item is now modified. If the modification had an effect on atom tree hierarchy, the appropriate tree items will not be created automatically. Just press the **Update**-button to update the "Atom tree".

- The element type of the atom, atom coordinates, colour, van der Waals radius and charge can be modified in the same way as the atom names. Colour can be changed either by giving a RGB-value for the wanted colour or by typing the name of the colour instead of the numerical values. If you have changed any of the parameters in the "Content"-part of the "Atom Tree"-window, the updating is done automatically and you *do not* have to click the Update-button.

  Go to *View* - *Atom tree...* to test this.

Display lists

- The program supports the use of the OpenGL display lists for all complex objects. The use of display lists reduces the need to repeatedly call the same low level graphics calls and can therefore make graphics display updating faster. This is especially true when running the program over a slow X11 connection and there is an OpenGL extension in X server.

Go to *Edit* - *Display list properties...* to turn them on.

Extension support

- Support for extensions (Dynamic Linked Libraries (DLL) in Windows and Shareable Object (SO) in UNIX) in improved.

Hydrogen bonds

- Hydrogen bonds can be recalculated for trajectory frames.
  Go to *Trajectory* - *Main...* In the "Trajectory Control"-window there are "on" and "off" radio buttons for "Hbond recalculation" where this function can be turned on and off.

- Hydrogen bonds are searched using a search subset. So if one wants to show only some of hydrogen bonds it is possible to define small search subsets around those atoms. In some circumtances this can significantly reduce the search complexity.

  Go to *Calculate* - *Hydrogen bonds...*

- The criteria how to calculate hydrogen bonds can be changed.

  Go to *Calculate* - *Hydrogen bonds* and click **Change criteria**

Import and export of the coordinate files

- gOpenMol can now identify most of the coordinate file types supported by the program based on the file extensions. In the "Import coordinates"-window this autodetection property is on by default. File format can still be selected manually but it should not be necessary unless the file formats, which do not have standard file extension, are being used. In the "Export coordinates"-window the autodetection property is off by default but can be turned on if needed.

- Ability to import from and export to atom coordinates of Chemical Markup Language (CML) file is added.

- The start-up routine uses the same autodetection property when it searches input filter for files given in command line. The coordinate file filter search routine is reimplemented so that support for a new file format can be added without recompailing the program source code.

Plumbers

- In former versions only helices and sheets were listed in plumber secondary structure window. Now the program also finds loop regions. In addition the whole protein chain is listed as trace.

  Go to *Plot* - *Plumber...* and click the **Click for information** -button next to the text "Secondary structure:" to view this.

- Two new plumber types are added: strand and trace. Trace will create a line and strand a ribbon through all selected atoms.

- By using fast draw option in *Edit* - *Display properties* the plumbers are drawn as lines during rotation so they do not disappear completely.

  Go to *Edit* - *Display properties...* and mark the Redispaly type "Fast" radio button.

- Plumbers can be glued to atoms. The points of glued plumbers are relative atom positions and will therefore move along atoms. This might be useful feature then using trajectories. The more important advantage, though, is the extra residue information in plumber list window.

  Go to *Plot* - *Plumber...* to test this feature. In the "Plumber"- window there is a box next to text "Glue to atoms". If this is marked, the gluing is on and if not, the gluing is off.

- Plumbers can now be listed, modified and deleted in the "Plumber list"- window. Listing shows the plumber type, the name of the structure containing the plumber, the segment and the residue names and the number of the first and the last residue of the plumber. Segment and residue information is available only if the plumber is glued to atoms. The colour of the plumber can be changed and the plumber can be unglued from atoms.

  Go to *Plot* - *Plumber...* and click the **List plumber(s)**- button to view and edit the Plumber list.

New command lines

- Four new command lines have been added. To view their properties, type

  ```
  help edit hbsubset,
  help find,
  help show plumber or
  help plumber
  ```

  in the command line interpreter.

New features in version 2.1

1. Tools to display a protein secondary structure using the so called cartoon display type.
2. Tools to calculate and display hydrogen bonds.
3. Extended filter program to display electron density gradients from GaussianXX and PcGamess.
4. General display engine for the display of vector data.
5. Improved usage of Tcl variables for the modification of the gOpenMol program and the post-processing of gOpenMol commands.

New features in version 2.0

1. All the discovered bugs in version 1.4 and 1.4p have been corrected.
2. Cut planes through the grid data can be defined from 3 coordinates (atoms) or planes perpendicular to the x, y and z axis.
3. Clip planes through the grid data can be defined from 3 coordinates (atoms) or planes perpendicular to the x, y and z axis.
4. Coordinate reader has been improved.
5. Coordinate reader for Jaguar output files.
6. Filter program to convert Jaguar grid data files into gOpenMol plt files.
7. Filter program to convert "cube" data in PC Gamess PUNCH files into gOpenMol plt files.

New features in version 1.4

1. All the discovered bugs in version 1.3 have been corrected.
2. New input coordinate formats have been added:
    1. Coordinate input can be read from Amsterdam Density Functional (ADF) output log files.
    2. Coordinates from a GROMACS gro coordinate file.
3. The trajectory readers have been changed to enable the reading of binary trajectories indipendent of hardware. In principle it should now be able to read

trajectories without the need to format/unformat them when moving from one hardware platform to an other.

4. New trajectory file format added:
    1. The GROMACS trn/trr/xtc trajectory file types.
5. Binary plt plot files can be moved between hardware platforms without the need to reformat them. The gOpenMol plt file reader can now do the byte swap on the fly.
6. It is now possible to select atoms by clicking on them and dragging them into an input widget. This makes it easier to pick atoms into input widgets.
7. All molecular systems read into gOpenMol has its own viewing matrix enabling the rotation/translation of individual molecules. The two modes are defined as **global** and **local** transformations. In the **global** mode all the molecular systems rotate/translate to a common center while in the **local** mode the translation/rotation will be according to an individual coordinate center.
8. To enable quick rotation, translation and picking it is possible to enable the translation mode by pressing the **Shift** key before one presses the left mouse. As long as the **Shift** key is and the left mouse button are pressed the structure will translate when the mouse is moved. Be careful to understand that if the **Shift** key will be released before the left mouse button the Translate state will be left on. In the other case Translate mode will return to Rotate mode.
9. By pressing the **Ctrl** key first and keeping it pressed it is possible to pick or mark atoms in the graphics screen by pressing the left mouse button. The selected atom(s) can then be dragged to an input widget. If one releases the **Ctrl** key before releasing the left mouse button gOpenMol will stay in the picking mode.
10. Pressing the **Alt** key and the left mouse button and moving the mouse in the x-axis direction rotates the molecule around the z-axis. This is the same as using the middle mouse on a three button mouse.
11. In the **Edit** menu there is now an **Atom Browser** that opens as a tree. The atoms in the tree can be clicked. The clicked atom will show up in the graphics screen as being selected. It also works the opposite way. Clicking an atom in the graphics window will highlight the clicked atom in the atom tree. The atom tree also shows a lot of atom information. In the future there will be a feature to edit the parameters.
12. All input file names can be given as a http link and thus be read from a Web server. If you know that there is a file named **myfile.crd** on a server named

my.server.org the input file name can be given as **http://my.server.org/myfile.crd**.

13. The currently on the screen seen coordinate orientation can be written out as a coordinate file or input file to an external program.

Index