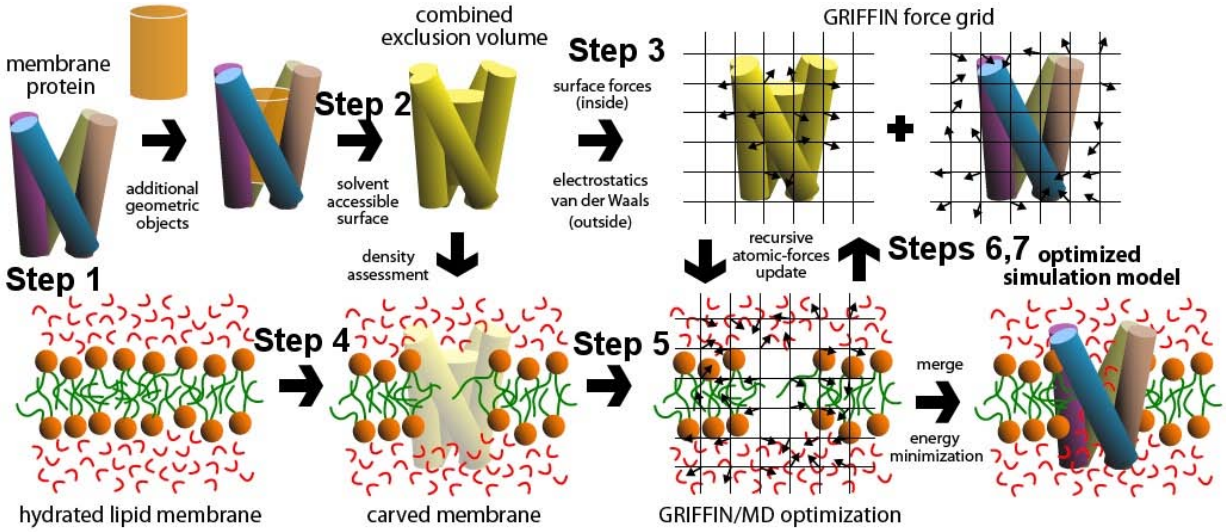
Protocol for Griffin Practical

Goal: to build a physically realistic atomistic starting model for membrane-protein simulations by means of GRIFFIN (Grid-based Force Field INput) program suite.

Structures of membrane proteins often show complex transmembrane topologies and highly irregular interfaces with the surrounding lipid bilayer. This poses a challenge for the construction of reliable starting configurations of the lipid bilayer to be used as starting models for molecular dynamics simulations.

GRIFFIN is a suite of tools, which are designed for adapting an existing membrane model to the structure of the protein. A first tool carves a cavity into the lipid bilayer, suitable for accommodating the protein. However the obtained protein-lipid interface is not optimized; therefore, the protein surface is employed to define a grid-based force-field designed to expel the remaining lipid and/or water atoms from the cavity. During a subsequent molecular dynamics simulation of the pre-carved membrane system, the implicit protein force-field is added to the standard interaction forces, thus optimizing the protein-lipid and protein-water interfaces. Geometric objects may be used as additional, lipid and/or water-specific exclusion volumes, during the carving and/or the simulation.



GRIFFIN is a standalone tool designed to work alongside any existing molecular dynamics package.

In addition to GRIFFIN, you will use the following main pieces of software:

- **NAMD**, a program for running molecular dynamics simulations (http://www.ks.uiuc.edu/Research/namd/).
- **CHARMM,** another program for setting molecular dynamics simulations (http://www.charmm.org/).
- **VMD**, a tool for visualization and analysis of proteins, nucleic acids, lipids, etc (http://www.ks.uiuc.edu/Research/vmd/).
- **xmgrace,** a 2-D plotting tool (http://plasma-gate.weizmann.ac.il/Grace/)**.**

After each step of the practical, you will be reminded to check the output files using a text editor and/or VMD. Make this a habit: ensuring that output files contain what you expect saves

a lot of time when there are errors or bugs (and forces you to think about what you should expect).

1) Obtain the starting structures and topologies for the protein and the hydrated lipid bilayer

Before using GRIFFIN, some starting files are necessary. In general, setting up the system from scratch needs time and involves complex procedures that are beyond the aim of this practical. Therefore the following files will be provided to you:

- A file containing the coordinates of the protein (pdb format). In this practical, you will use the structure of the lactose permease (lacY) of *E. coli*. Wikipedia has some interesting background on the role of the lac operon and lacY for the bacterium metabolism (http://en.wikipedia.org/wiki/LacY)
- A topology file for lacY (xpsf (CHARMM,NAMD,XPLOR psf) format). The file contains the information about the protein atom connectivity and defines all the interactions within the protein (e.g. bonds, angles, dihedrals, partial charges, etc.).
- A file containing the coordinates of a hydrated membrane (pdb format), such as a POPE lipid bilayer.
- A topology file for the POPE bilayer.

Copy the starting files from `GRIFFIN/examples/lacy/starting_files` to your working directory.

Inspect the structures of the protein and the POPE bilayer with VMD. To do this, type the following command into your Unix prompt:

```
&> vmd -m lacy.setup.pdb pope340w.equil.1.pdb
```

The VMD display and main windows will now open. You can find more information and help on using VMD at the program website.

In the VMD main window you should see two different entries. You can display/hide each molecule by double-clicking the "D" on the left of the molecule name. Open the graphical representation window (Graphics→Representations) and change the protein drawing method so that all of the structure is shown using cartoons. Create a new representation showing lacY transmembrane region by typing into the "Selected Atom" field:

```
resid 11 to 34 45 to 67 75 to 96 105 to 129 143 to 164 167 to 186 220 to 247
257 to 279 288 to 308 312 to 333 349 to 370 381 to 399
```

and change the coloring method (e.g. to ColorID→yellow). Now check that the POPE bilayer and lacY transmembrane domain are aligned. In addition the bilayer+water box should be large enough to contain the whole protein plus 8-10 Å between the protein and the box walls (it is convenient to change the display view to "Orthographic" in the Display menu). If you just want to look at the POPE bilayer, without the waters (and hydrogens), change the view by typing "`not water and noh`" in the graphical representation window.

2) Setting up the GRIFFIN starting files

First you need to merge the topologies and coordinate files into a file that GRIFFIN can read. This can be done by using the program `griffin_tools`.

By typing

```
&> griffin_tools.exe
```

you should obtain a short report about the program usage. In your case, you should use the first option `-namd2griffin`. Then you should provide the pdb file name and the xpsf topology file name of your system. The `CHARMM_PARAMETER_FILE` contains the information about the CHARMM force field. Finally the name of the output file:

```
&> griffin_tools.exe -namd2griffin lacy.setup.pdb lacy.setup.xpsf  \
      par_all27_prot_lipid.prm  lacy.gfn >& prot_setup.log
```

check the log file for errors and then repeat the process for the POPE bilayer

```
&> griffin_tools.exe -namd2griffin pope340w.equil.1.pdb pope340w.xpsf \
      par_all27_prot_lipid.prm pope340w.gfn >& pope_setup.log
```

As a result you should obtain the protein and membrane gfn files.

3) Carving Out Lipids and Water from the Bilayer

Again you should obtain some help about all the possible options by typing:

```
&> griffin_carver.exe
```

The required flags:

`-membrane FILE`	(e.g. `pope340w.gfn`)
`-solute FILE`	(e.g. `lacy.gfn`)
`-lipid_names NAME1 NAME2 …`	(e.g. `POPE`)
`-solvent_names NAME1 NAME2 …`	(e.g. `TIP3`)
`-write_carved FILE`	(the carved membrane, e.g. `pope_carved.gfn`)

The optional flags include:

`-dimensions XMIN XMAX YMIN YMAX ZMIN ZMAX Z_BILAYER CENTER Z_BILAYER_THICKNESS`	(define the box dimensions. Values should be known from the equilibration MD run of the bilayer; in this case the box dimensions were 103.5 x 103.5 x 84.1 Å. The box is centred at (0, 0, 0) and so is the bilayer. 35 Å is a good estimate for the bilayer thickness.
`-write_carved_pdb FILE`	(the same but in the more common and readable pdb format, e.g. `pope_carved.pdb`)
`-write_erased_pdb FILE`	(for your information, the file contains the lipid and water molecules that have been erased (e.g. `pope_erased.pdb`)

`-write_surf_as_pdb FILE`	(for your information, the file contains the lacY surface calculated by GRIFFIN (e.g. `lacy_surface.pdb`)
`-surf_voxel_size VALUE`	(resolution of surface grid, e.g. 0.5 Å)

When running the command, remember to save to output to a log file with the '`>&`' redirection sign and eventually check it for errors. In particular:

- check that all the input parameters have been read correctly;
- the calculated limits of the bilayer box should be similar to the ones in the input (probably slightly larger).

The program divides the box into 4 layers: the first (layer 0) contains the water molecules below (lower z coordinates) the POPE bilayer. The second and the third (layers 1 and 2) contain the lower and upper POPE leaflets, respectively. Then the fourth (layer 3) contains the waters above (higher z coordinates) the POPE bilayer.
You should observe that, while the program is processing layer 0 and 3, only water molecules are erased, whereas when processing layers 2 and 3, a list of deleted lipid molecules is given. They are sorted according the number of atoms that were buried inside the lacY surface. You may notice that some lipids remain, even though they have some buried atoms. In fact, the aim at this step is to keep the final density of lipid constant (~63 lipid/Å^2 in the case of POPE). Therefore the number of deleted lipids is determined by the volume occupied by lacY transmembrane region (that is also why the numbers of the deleted lipids in layers 2 and 3 probably differ).
Check all the output pdb files with VMD. Again it could be useful to superimpose the lacY structure as a cartoon.
At this point you should notice that some lipid molecules are present in the central crevice of the protein. You could exclude such volume to lipid molecules by adding an object (e.g. a cylinder), with the option:

```
-add_surf_objects  object.dat
```

the file `object.dat` should contain the information about the position, orientation and dimension of the object (in this case a cylinder). It should look as:

`1`	(object 1)
`POPE`	(exclude the volume only for POPE)
`Cylinder`	(object shape)
`0 0 -9`	(object origin)
`0 0 1`	(vector defining the axis direction, e.g. vertical)
`15 20`	(radius and height of the cylinder)

Create a new working directory and launch again the `griffin_carver.exe`. Now you can notice the absence of the lipids in the central crevice. The surface of the cylinder is also visible in the surface (pdb) file. Note also that the water molecules were not affected by the presence of the object.

4) *Calculation of the implicit protein force-field*

A third GRIFFIN tool enables the calculation of the implicit force-field due to the protein. This will subsequently be used during the molecular-dynamics (MD) optimization stage. This force-

field is stored on a three-dimensional grid of user-defined grid-point spacing (with a default value of 0.5 Å). Forces acting on lipid atoms due to Coulomb and van der Waals interactions with the protein volume, together with the repulsive surface-guided forces will be extrapolated from the grid at every step of the MD; this results in chemical specificity at the protein-lipid interface, in addition to shape complementarity.

This step is computationally demanding, therefore it is convenient to split the job into a certain number of subgrid calculations, and then eventually to combine the results.

It is convenient to work in a new directory (e.g. `grid`) as this procedure will create a large number of files.

```
&> mkdir grid
&> cd grid
```

As a starting point, we need a template file for launching the subgrid job. It will look like:

```
/griffin_force_grid.exe \
      -parallel               4 NNN             \
      -solute                 lacy.gfn          \
      -write_force_grid       lacy_subgrid.txt  \
      -add_surf_objects       object.dat        \
                  >& build_subgrid_NNN.log
```

The template file calls the GRIFFIN tool: the option '`-parallel`' says to the program that only a subgrid has to be calculated. But how does it determine which one? The number 4 in the first option means that 4x4x4=64 subgrids will be calculated in total. The `NNN` should be a number within the range from 0 to 63. You will use a script to iteratively create different files by substituting `NNN` with a convenient number.
With a text editor, save this file as `template_build_subgrid.sh` and then proceed with the next step.

You need a script for launching all the subgrid jobs. Before running the jobs, however, remember to copy `lacy.gfn` and `object.dat` into the working directory.

In order to fully understand this section you should be aware that your job will run over a computer cluster, namely a group of linked computers working together. Each physical unit forming the cluster is called a node and each node possesses a certain number of processors (cpus). Our cluster, `pacific`, consists of 143 nodes, each containing 8 cpus. To run a job, you should reserve a convenient number of cpus for some time by invoking the queue scheduling system (PBS (Portable Batch System)), which is the program that allocates the computing tasks among the available computational resources.
The first part of the script is needed for running the job on the cluster by reserving 64 cpus for 6 h (but jobs will end before!).

```
#!/bin/csh

#PBS -m n
#PBS -l nodes=8:ppn=8
#PBS -l walltime=6:00:00
#PBS -N YOUR_FAVORITE_NAME
#PBS -j oe
#PBS -q YOUR_RESERVED_QUEUE
```

Due to an interaction of the PBS scheduler and MPI (a language-independent communications protocol used to program parallel computers), which might be related to our specific installation, the submission scripts need to have this line:

```
unsetenv OMP_NUM_THREADS
```

The script should then change to the working directory:

```
cd $PBS_O_WORKDIR
```

The jobs can be controlled under PBS using the nodefile that contains all allocated resources. The nodefile is accessible by the environment variable `$PBS_NODEFILE`. Filtering the nodefile enables the exact job-distribution to be specified.

```
cat $PBS_NODEFILE > nodefile

set j = 1

while ($j <= 64)

  set NODE = `sed -n {$j}p nodefile`

  set k = `expr $j  - 1`
```

(Note the ticks leaning toward the left). The next command will create a new script by replacing `NNN` with the correct number

```
  cat template_build_subgrid.sh | sed "s/NNN/$k/" > build_subgrid_$k.sh
  chmod +x build_subgrid_$k.sh
```

The next command will launch the griffin program on the remote node by using `ssh`.

```
  ssh $NODE "cd $PBS_O_WORKDIR; ./build_subgrid_$k.sh" &


  @ j++

end
wait


exit
```

Now, launch the script with the qsub command:

```
&> qsub run_infiniband.qss
```

At this point, by typing at the Unix prompt:

```
&> qstat –u `whoami`
```

you should hopefully see the job running. You can see how the jobs proceed by checking one of the logs. The last line reports the number of layers that remain to be calculated. Subgrid calculations are generally slow; but in this case, they need approximately 40 min to complete.

When all the subgrids have been completed correctly, you can combine them. You can use the same `griffin_force_grid.exe` tool. At the Unix prompt type:

```
&> griffin_force_grid.exe                \
     -collect_parallel lacy_subgrid 4    \   (the subgrid files common name. Again 4
                                              means that the initial grid was separated
                                              into 4x4x4=64 different subgrids)
     -write_force_grid lacy_grid.txt     \   (The final grid file)
     >& combine_subgrids.log
```

If everything is correct you should have the large grid file `lacy_grid.txt`. Compress the subgrids and the respective log files to save space:

```
&> gzip -9 *subgrid* &
```

5) Setting up a GRIFFIN & NAMD simulation with MPI parallelization

In this step, you run a molecular dynamics simulation of the carved POPE bilayer by including the protein as an implicit representation (i.e. the grid). In this case, GRIFFIN is executed as a "daemon", which, in Unix systems, means a program that runs in the background and remains latent until the moment to perform its task.
At every MD step, the program performing the molecular dynamics run, i.e. NAMD, writes the atomic positions of the explicit system (i.e. lipid and water molecules) into a file. the GRIFFIN daemon reads the coordinates and calculates the forces rising from the interactions with the implicit protein by interpolating the grid. Then it writes a file containing all the forces. Finally, NAMD reads the force file and sums the new forces to those coming from the explicit system.

Again it is convenient to work in a new directory (e.g. `namd`) as this procedure will create a large number of files. From your main working directory type

```
&> mkdir namd
&> cd namd
```

5a) setting the NAMD input files

In order to run, NAMD needs at least three input files:

- A file containing the coordinates of the carved hydrated POPE bilayer in pdb format, e.g. `pope_carved.pdb`. You can copy this file in the current working directory together with the corresponding file in gfn format (e.g. `pope_carved.gfn`), which will be useful for the GRIFFIN setup.
- A topology file for the carved membrane in xpsf format. The old xpsf file, which was used for creating the initial gfn files in section 2, is no longer useful because the number of lipid and water molecules is changed after carving. You will need a CHARMM script (e.g. `mk_psf.inp`) in order to create a new topology file.
- A file containing the running parameters of the molecular dynamics (e.g. `equil.inp`).

In addition, you will use an additional tcl script for setting up external accessory forces (`tcl.inp`).
You can copy the examples for the `mk_psf.inp`, `tcl.inp` and `equil.inp` files from `GRIFFIN/examples/lacy/namd`

Save the files in your working directory and then start editing the `mk_psf.inp` script with a text editor. The script should be executed with the CHARMM program. Without entering into the details of every single statement, the script will generate a block of POPE lipid molecules, followed by a block of water (TIP3) molecules. Then it will write the topology files as outputs. You can find more information and help on using CHARMM at the program website (http://www.charmm.org/html/documentation/c34b1/index.html). For this practical, you just need to edit the file by introducing the correct numbers of lipids and water molecules. You can easily find them for the lipid and waters, respectively, by typing at your Unix prompt:

```
&> grep C22 pope_carved.pdb | wc -l
&> grep OH2 pope_carved.pdb | wc -l
```

Then execute the script by typing:

```
&> charmm < mk_psf.inp | \
      tee mk_psf.log
```

(the tee command will allow you to see the output on your screen and, at the same time, save it into the log file). If the xpsf file is correct you should be able to visualize both with VMD as a unique molecule entry:

```
&> vmd -psf pope_carved.xpsf pope_carved.pdb
```

Now you must edit your `equil.inp` file. Again, explaining all the possible options goes beyond the aims of this practical. Just check and edit a few things; nevertheless, you can find more information and help on using NAMD at the program website:

- the script should run 50 ps of dynamics with a time step ('`timestep`') of 1 fs. That corresponds to 50000 steps ('`numsteps`').
- Check if the file names for '`structure`' and '`coordinates`' are correct. Also '`parameters`' should point to the correct file.
- Temperature should be set to 310 K for POPE simulations: check the '`temperature`', '`langevinTemp`' and '`LangevinPistonTemp`' entries.
- The option '`useConstantArea`' will guarantee that the surface per lipid will remain constant.
- Set correctly the box dimensions and the origin (see above) with the options '`cellOrigin`' and '`cellBasisVector1`', '`cellBasisVector2`', '`cellBasisVector3`'.
- Adapt '`PMEGridSizeX`', '`PMEGridSizeY`' and '`PMEGridSizeZ`' so that they are integers larger than the corresponding box dimensions. For technical reasons, such integers should be multiples of small numbers, such as 2, 3 and 5.
- Add a section for saving coordinates at every step (e.g. into the file `namdx.tmp`), executing the external GRIFFIN program and reading the force (e.g. from the file `namdf.tmp`):

```
extForces           on
extForceCommand     PATH/griffin_messenger.exe -forces
extCoordFilename    namdx.tmp
extForceFilename    namdf.tmp
```

- Setting '`restartfreq`', '`DCDfreq`', '`outputEnergies`' and '`outputTiming`' to 100 will guarantee that the output files will be written every 100 steps.

- Switch '`tclForces`' on and check the name of the tcl script '`tclForceScript`' (e.g. `tcl.inp`)

Now edit the `tcl.inp` file. This file is needed for vertically constraining the center of mass of the lipid bilayer at the center of the box. Again, you can find more information at the NAMD website. For this practical, you just have to set the right number of POPE molecules at the beginning of the script.

5b) setting the general launching script

Also your MD job will run over the computer cluster. A common scenario will be that you want to use e.g. 2 nodes with 8 cpus in total for the simulation. While there is no problem calling 8 parallel NAMD jobs per node, this may not be possible for GRIFFIN for memory reasons. The best would be to request 8 cpus per node from your batch system, running GRIFFIN on, e.g., 2 cpus per node and NAMD on the remaining 6.
Copy the script from `GRIFFIN/examples/lacy/namd` and save it in your working directory (e.g. `run_namd.qss`). With a text editor you can edit it. The first lines of the script are necessary for reserving the 2 nodes with 8 cpus each, from the cluster scheduling system, and for requesting that the job be 2 days long.

```
#!/bin/bash

#PBS -l nodes=2:ppn=8
#PBS -l walltime=48:00:00
#PBS -N YOUR_FAVORITE_NAME
#PBS -q YOUR_RESERVED_QUEUE
#PBS -m n
```

Due to an interaction of the PBS scheduler and MPI (a language-independent communications protocol used to program parallel computers), which might be related to our specific installation, the submission scripts need to have this line:

```
unset OMP_NUM_THREADS
```

The script should then change to the working directory:

```
cd $PBS_O_WORKDIR
```

Then you have to set the variable pointing to the necessary executables:

```
griffin_msg=PATH/griffin_messenger.exe
griffin=PATH/griffin_daemon_mpi.exe
namd=PATH/namd2
mpi=PATH/mpiexec
```

The jobs can be controlled under PBS using the nodefile that contains all allocated resources. The nodefile is accessible by the environment variable `$PBS_NODEFILE`. Filtering the nodefile and redirecting the environmental variable enables the exact job-distribution to be specified.

```
cat $PBS_NODEFILE > original_nodefile.txt

cat original_nodefile.txt |                                                    \
awk '{if ($0 != prev) {for (i=0; i<2; i++) {print $0;} prev=$0;}}' >    \
```

```
gfn_nodefile.txt
```

This will create a new nodefile in which just **2** cpus per node are included. Before launching the GRIFFIN daemon, the gfn_nodefile should be read and the number of cpus evaluated from the number of lines:

```
n=$(cat gfn_nodefile.txt | wc -l)
PBS_NODEFILE=gfn_nodefile.txt
```

The next command will launch the griffin daemon on n cpus by using `mpiexec`. Of course you have to specify the proper options:

```
$mpi -np $n $griffin          \
      -force_grid             FILE    (the file with the grid; e.g.
                                        ../grid/lacy_grid.txt)
      -coordinate_format      NAME    (in this case namd)
      -lipid_names            NAME    (in this case POPE)
      -neighbor_list_update   VALUE   (forces for non buried atoms that have force=0
                                       are recalculated every VALUE steps; e.g. 10
                                       steps)
      -sforce_scale           VALUE   magnitude of the forces pushing molecules out of
                                       the implicit volume; e.g. 1.0
      -sforce_reset_min angle VALUE   useful for faster convergence. The default value is
                                       110
      -sforce_exclude_hydrogens       the program does not apply surface forces to
                                       hydrogens
      -logfile                FILE    (standard output file; e.g. griffin_daemon_3D_)
      >& mpi.log
```

After GRIFFIN is started it has to wait until NAMD writes the first coordinate file:

```
$griffin_msg -wait >& wait.log
```

Then you have to tell GRIFFIN what files will be read and written during the simulation. For example, coordinates will be written in the `namdx.tmp` file, whereas GRIFFIN will write the `namdf.tmp` file with the forces due to the implicit protein.

```
$griffin_msg -setup pope_carved.gfn namdx.tmp namdf.tmp  >& setup.log
```

Finally, NAMD can be launched. Before this step, however, we need a new nodefile for distributing NAMD over the remaining **6** cpus per node:

```
cat original_nodefile.txt |                                                    \
awk '{if ($0 != prev) {for (i=0; i<6; i++) {print $0;} prev=$0;}}' >    \
md_nodefile.txt

n=$(cat md_nodefile.txt | wc -l)
PBS_NODEFILE=md_nodefile.txt

$mpi -np $n $namd equil.inp >& equil.log
```

Finally you need the GRIFFIN termination command:

```
$griffin_msg –terminate >& terminate.log
```

6) Launching the GRIFFIN & NAMD simulation in parallel

Now, launch the script with the qsub command:

```
&> qsub run_namd.qss
```

If everything is correct, by typing

```
&> qstat –u `whoami`
```

you should see your job running on 2 nodes (NDS) and 16 cpus (TSK). Check the `gfn_nodefile.txt` which should contain 2 entries per node.
If the program runs correctly, GRIFFIN first read the grid file (check the `mpi.log`, for example by typing '`tail –f mpi.log`' at your Unix prompt (`CTRL+C` for exiting)). Only after reading the grid, can the NAMD job start. Check the `md_nodefile.txt` (which should contain 6 entries per node).
NAMD will generate the `equil.log` file; check if everything is correct. At the same time, the GRIFFIN daemon will create a log for every task running in parallel (e.g. 4). Check the log from the master node (the one ending with a '0'). Inside it is reported the number of buried atoms and the maximum depth inside the protein surface at every step.
Graph these values by typing at your Unix prompt:

```
grep "####" griffin_daemon_3D_0.log | awk '{print $3}' | xmgrace - &
```

the xmgrace window will open. you should observe that the number of buried atoms decreases during the simulation. The same should be true for the atom penetration maximum depth:

```
grep "####" griffin_daemon_3D_0.log | awk '{print $5}' | xmgrace - &
```

Finally check the trajectory (dcd format) with VMD:

```
&> vmd –psf pope_carved.xpsf pope_carved.equil.1.dcd
```

You may want to superimpose your initial lacY structure to check if GRIFFIN is working correctly.

How much time is needed for the job to complete? This could be checked from the NAMD log file. Just type at your Unix prompt:

```
&> grep TIMING equil.log
```

and you will have an idea of the time left, presumably several hours. Consequently the second part of the practical will take place on day 4 and the analysis of the results will take place on day 5.

7) Continuing the GRIFFIN & NAMD simulation (Part 2)

Application of the surface forces calculated by GRIFFIN results in a rapid decline in the number of buried atoms, until a plateau is reached, reflecting a balance between the expelling forces and the external pressure from lipid and water. However, the rate of decrease and the plateau level are dependent on the strength of the applied surface forces.
Low GRIFFIN surface forces (e.g. 1 kcal/mol/Å^2, equivalent to '`-sforce_scale 1.0`') guarantee that lipids are gently pushed apart from the protein volume. Nevertheless, the plateau is reached when many atoms still lay too deeply inside the protein surface. Therefore, such a lipid/water configuration is not suitable as a starting point for stable standard molecular dynamics simulations. Subsequent stages in which the magnitude of the surface forces is increased (up to 9 kcal/mol/Å^2) progressively empty the protein volume further. At the end of the last stage several atoms remain inside the implicit volume; however, these are expected to be within ~1.5 Å of the protein surface, so that any clashes can be resolved through standard energy minimization procedures.

In this second part of the practical, you will start a new GRIFFIN & NAMD simulation with higher surface forces.

First, repeat the analysis of the previous job. Has it finished correctly? Do the lipid conformations look reasonable to you?

In your main working directory create a new directory for running the new job (e.g. `namd2`):

```
&> mkdir namd2
&> cd namd2
```

then copy from the old `namd` directory all the input files you needed for running the old job. Edit the `run_namd.qss` script by changing the option '`-sforce_scale 1.0`' to 3.0. In order to start the new NAMD job from the final configuration of the old job, copy all the restart files (`*.rst.*`) into the current working directory. Then edit the `equil.inp` file. Changing the value of the variable `i` to 2 at the beginning of the script should guarantee that all the restart files will be read correctly (anyway, when running the simulation, check the NAMD log file if it is really so). Launch the new dynamics and repeat the analysis as in the previous section.

Summary

This practical has hopefully given an example of the steps required for building a reliable model of a protein in a hydrated lipid bilayer. In real case situations, an additional step would be necessary for creating the initial files with the structure and topology of your protein of interest and a suitable lipid bilayer. This step can be rather complex. In the case of lipid membranes, equilibrated structures are often available in the literature for a large number of lipids and mixtures, but often some kind of processing is necessary, for example, for adapting the dimensions of the simulation box to the protein. Known structure of proteins can be downloaded, for example, from the PDB (Protein Data Base).

FAQ: Can I run this for my protein/is the software free and available?

Most of the programs mentioned here are available for download – search for them with Google. Most of them are freely available for academic users and often work on more than one operating system (i.e. VMD), even if the installation could create some problems. CHARMM is available for academic users after paying 600 $ licensing fee. GRIFFIN is in an advanced revision phase and will be released very soon. In some cases, other programs can be substituted for them, e.g. you can use any structure visualization program, such as Rasmol or PyMol, it does not have to be VMD.