

## Strecken eines Dekalanine (Ala)<sub>10</sub>

In diesem Computer Versuch lernen Sie, wie man mit Kraftfeldern Molekulardynamik Simulationen durchfuehrt. Die vorliegende Problemstellung ist von praktischer Relevanz bei der Bestimmung von Stabilitaeten und Faltungsprozessen von Proteinen. Genauer: der Versuch simuliert das Verhalten eines Polypeptides in Form einer alpha-Helix beim Strecken mittels eines Kraftmikroskopes. Dabei ist ein Ende der Helix an eine Oberflaeche kovalent gebunden („tethered“) und das andere Ende mit der Spitze (Cantilever) eines Kraftmikroskopes verknuepft. Nun wird die Spitze langsam hochgezogen und die auf den Cantilever wirkende Kraft wird gemessen. Dies ergibt sogenannte „Force extension curves“, welche dann in eine Arbeit (Kraft x Weg) umgerechnet werden koennen. Dabei wird die aufgewendete Kraft immer reduziert, wenn ein bestimmtes Element innerhalb der Helix bricht (z. B. eine H-Bruecke).

Im Folgenden sind die Schritte aufgefuehrt, die Sie durchlaufen sollen, um solche Force extension curves zu berechnen. Die Anleitung ist auf Englisch geschrieben, und die Befehle benoetigen die Unix shell auf den studix Rechnern.

## 4 SMD simulation

Let's run an SMD simulation of deca-alanine. It is basically a more systematic way of doing the IMD simulation we just finished. IMD simulations are done to explore the system; SMD simulations are done to analyze the system systematically.

Files needed:

- `da.psf` – protein structure
- `smd.namd` – NAMD configuration
- `smd.tcl` – Tcl script
- `par_all127_prot_lipid.prm` – CHARMM parameters
- `smd_ini.pdb` – initial coordinates

The simulation is done at a constant temperature of 300 K. The Langevin dynamics scheme is used for the temperature control. The Tcl script `smd.tcl` is used to apply external forces. Basically the script does the following. One end of the molecule (the N atom of the first residue) is constrained to the origin. The other end (the capping N atom at the C-terminus) is constrained to a point that moves along the z-axis from 13 Å to 33Å with a constant speed of 1 Å/ps. So it takes 20 ps for the full extension. For the constraints, a harmonic potential with a force constant of 7.2 kcal/mol/Å<sup>2</sup> is used.

Now let's run the simulation:

```
namd2 smd.namd > da_smd.log
```

The simulation will run on your local machine. The output will be written to the following files:

```
da_smd.log – standard output
da_smd.dcd – trajectory
da_smd.tcl.out – Tcl script output
```

## 5 Viewing SMD trajectories within VMD

Start VMD loading the trajectory just generated:

```
vmd da.psf da_smd.dcd
```

Position the molecule (rotate, translate, scale) so that you have the view of the entire length of the molecule.

Change the representation: Select the menu item Graphics→ Representations. In the Graphical Representations window, set Drawing Method to Ribbons. The Ribbon representation is good for viewing the overall structure, but you may explore any other representations or even multiple views.

Use the scroll bar at the bottom of the VMD Main window to browse through the trajectory.

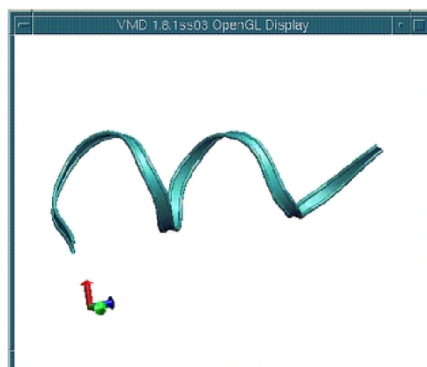


Figure 4: Deca-Alanine in ribbon representation, loaded into VMD.

An important structural change during the helix-coil transition is the breaking of hydrogen bonds. You can monitor hydrogen bonds using VMD.

Choose CPK from Drawing Methods. This will change the representation of the molecule from Ribbon to CPK.

Now let's show hydrogen bonds:

1. In the Graphics Representations window, click **Create Rep.**
2. In the **Selected Atoms** text entry, delete the word **all**, type **name N O**, and hit the Enter key. (Hydrogen bonds are formed between N and O atoms.)

3. Select HBonds from Drawing Method.

4. Change the parameters to the following: Distance Cutoff: 4.0, Angle Cutoff: 40, Line Thickness: 10.

You should see several hydrogen bonds.

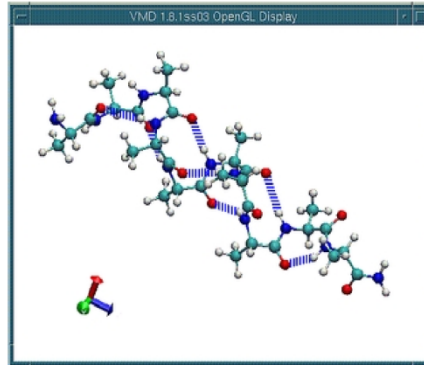


Figure 5: Deca-alanine in CPK representation. Hydrogen-Bonds are highlighted.

Again using the scroll bar at the bottom of the VMD Main window, browse through the trajectory. Observe the hydrogen bond breaking as the molecule is stretched.

Once you are done, quit VMD (the menu item File → Quit).

At this stage we have carried out a Molecular Dynamics simulation. The real challenge is now to analyze it in such a way that we gain information from it. This is done in the next step.

**In the following, green arrows indicate graphics you have to produce and store; they will go into your report. Recommended format is "png". Red arrows are questions you should answer. There is some literature and explanation that goes with the practicum and should help you to find answers to the questions.**

## 6 Analysis of the SMD trajectory

Launch VMD by typing `vmd` in a terminal window.

Within VMD select the menu item `Extension` → `Tk Console`. A console window should appear with a prompt. The following Tcl/Tk commands are executed from within the TkCon window. We will use the tag `>>` to indicate Tcl commands.

Make sure you are in the correct directory by typing:

```
>> cd ~/Workshop/10Ala-tutorial/files/
```

Open the Tcl Output file generated by your SMD simulation, and store its content to a variable `mytraj`:

```
>> set infile [open da_smd_tcl.out r]
>> set mytraj [read $infile]
>> close $infile
```

The content of the file `da_smd_tcl.out` is printed on the screen. You should see three columns:

First column: time (ps)

Second column: extension, or the end to end distance (Å)

Third column: applied force (kcal/mol/Å)

We use the following units: ps for time, Å for distance, kcal/mol for energy.

Assign columns of the `mytraj` to three separate arrays (lists), by running the following Tcl script:

```
>> source read.tcl
```

The script defines three variables, `t`, `z`, `f`, corresponding to each column of `mytraj` respectively.

Define parameters, `v` (pulling velocity) and `dt` (time step of the data):

```
>> set v 1
>> set dt 0.1
```

Recall that one end of the molecule was constrained to the origin and the other end was constrained to a moving point. The distance between the two constraints was changed from 13 Å to 33 Å with the constant velocity  $v$ . Define an array (list) `c` representing the distance between the two constraint points at each time step:

```
>> set c {}
>> set i 13
>> while {$i <= 33} { lappend c $i; set i [expr $i + $v * $dt] }
```

Plot  $z - t$  and  $c - t$  curves using the `multiplot` plugin within VMD. In the TkCon window type:

```
>> package require multiplot
>> set plot1 [multiplot -x $t -y $z -plot]
>> $plot1 add $t $c -plot
```

Q: The extension  $z$  generally lags behind  $c$ , the distance between the two constraints. How big is the lag in this case? What would you do if you want to reduce the lag further?



Now plot the force-extension curve:

```
>> set plot2 [multiplot -x $z -y $f -plot]
```



Q: Is the force generally positive or negative? Why?

The work done on the system during the pulling simulation is:

$$W(t) = v \int_0^{t'} dt' f(t') \quad (1)$$

where  $v$  is the pulling velocity.

To calculate the work  $w$  by numerical integration, run the following Tcl script:

```
>> source calcwork1.tcl
```

The work is stored in a variable (list) called  $w$ .

If a pulling simulation is performed very slowly, then the process is reversible. The work done during such a reversible pulling is equal to the free energy difference of the system between initial and final states. Thus, a reversible work curve can be considered as an exact PMF. For our system, the reversible pulling speed turns out to be about 0.0001 Å/ps (10000 times slower than the one you used).

The exact PMF obtained from a reversible pulling is stored in the file **Fexact.dat**. Read the content of the file, and store it in a variable called **Fexact**, by running the following Tcl script:

```
>> source load-Fexact.tcl
```

The array **Fexact** contains two columns, representing the potential of mean force (second column) at each extension (first column).



Plot  $W - c$  in blue together with  $F_{exact}$  in red:

```
>> set plot3 [multiplot -x $c -y $w -linecolor blue -plot]
>> $plot3 add $Fexact(1) $Fexact(2) -linecolor red -plot
```



Q: How do they compare? What is the origin of the difference?

Quit VMD:

```
>> quit
```

## 6.1 PMF calculation

The trajectory you have generated in this session is actually not practical for the PMF calculation because the pulling speed was too high. Besides, you need multiple trajectories to use Jarzynski's equality. Therefore, you will use pre-generated trajectories.

Launch VMD by typing `vmd` in a terminal window.

Within VMD select the menu item `Extensions` → `Tk Console`. The following commands are all executed from within the `TkCon` window.

Make sure you are in the correct directory by typing:

```
>> cd ~/Workshop/10Ala-tutorial/files/
```

Ten trajectories obtained from SMD simulations with a pulling speed of  $0.01 \text{ \AA/ps}$  are stored in `da.dat`. This pulling speed is 100 times slower than the one you used, but still 100 times faster than the reversible speed. Load the trajectories as well as the exact PMF:

```
>> source load-Fexact.tcl
>> source load-traj.tcl
```

The script `load-traj.tcl` defines three variable. The array `t` contains time steps of the data. `z` and `f` are matrices of 10 columns, with each column representing the extension and the applied force for each trajectory, respectively.



Plot the extension-time curve, for each trajectory. Use a different color for each curve:

```
>> set lcolor {green orange black pink brown purple yellow cyan blue red}
>> set plot1 [multiplot -plot]
>> foreach l [array names z] c1 $lcolor { $plot1 add $t $z($l) -linecolor $c1 }
>> $plot1 replot
```



And plot the force-extension curve:

```
>> set plot2 [multiplot -plot]
>> foreach l [array names f] c1 $lcolor { $plot2 add $z($l) $f($l) -linecolor $c1 }
>> $plot2 replot
```

In each figure window, you should see ten overlapping curves, with each curve corresponding to a trajectory.

Define parameters: `dt`, time step; `v`, pulling speed; `T`, temperature (including Boltzmann's constant).

```
>> set dt 0.1
>> set v 0.01
>> set T 0.6
```

Define an array `c` representing the distance between the two constraint points at each time step:

```
>> set c {}
>> set i 13
>> while {$i <= 33} {lappend c $i; set i [expr $i + $v * $dt] }
```

Calculate work for each trajectory by executing the following command:

```
>> source calcwork2.tcl
```



Plot  $W$  vs.  $c$  together with the exact PMF:

```
>> set plot3 [multiplot -plot]
>> foreach l [array names w] cl $lcolor { $plot3 add $c $w($l) -linecolor $cl -plot}
>> $plot3 add $Fexact(1) $Fexact(2) -linewidth 3 -plot
```

You should see ten work curves and one curve for the exact PMF. The thick black line is the exact PMF.



Q: Work done to the system should be on average higher than the PMF. Are all the work curves higher than the PMF, or do you also see some work curves lower than the PMF?

Let's now employ Jarzynski's equality:

$$\exp\left\{\frac{-[F(c(t)) - F(c(0))]}{T}\right\} = \langle \exp[-W(t)/T] \rangle \quad (2)$$

or

$$F(c(t)) - F(c(0)) = -T \log \langle \exp[-W(t)/T] \rangle \quad (3)$$

The right-hand side can be expanded in terms of so-called cumulants:

$$F(c(t)) - F(c(0)) = \langle W(t) \rangle - \frac{1}{2T} [\langle W(t)^2 \rangle - \langle W(t) \rangle^2] + \dots \quad (4)$$

where the cumulants up to the second order are shown.

Estimate the PMF according to three different formulas: the original formula (exponential average), the first order cumulant expansion formula (average work), and the second order cumulant expansion formula. The following Tcl script will calculate and store the three different estimates in **Fexp**, **F1**, **F2** variables.

```
>> source cumulants.tcl
```



Plot the three estimates for the PMF together with the exact PMF. Plot the exact PMF, **Fexact**, in red, the average work (or the first order cumulant expansion), **F1**, in green, the second order cumulant expansion, **F2**, in blue, and the exponential average, **Fexp**, in black:

```
>> set plot4 [multiplot -plot]
>> $plot4 add $Fexact(1) $Fexact(2) -linecolor red -plot
>> $plot4 add $c $F1 -linecolor green -plot
>> $plot4 add $c $F2 -linecolor blue -plot
>> $plot4 add $c $Fexp -linecolor black -plot
```



Q: Which estimate is the closest to the exact PMF? Does it make sense to you in view of what you learned in the lecture?

Quit VMD:

```
>> quit
```

For more information on the PMF calculation from SMD simulations, see the paper included ([paper.pdf](#)).

Final task: take the input file `smd.namd` and use a speed of 0.1 A/ps – this will produce a 200 ps trajectory. Analyze it in the same way as the other curves above and compare. What are your observations? For successful completion of this step you will need to make the following changes:

- modify `smd.namd`: from "run 10000" to "run 100000"
- modify `smd.tcl`: from "set v 0.002" to "set v 0.0002" and from "set TclFreq 50" to "set TclFreq 500"

### ***Anleitung zum Verfassen des Protokolls***

Im Protokoll muessen die Konzepte „Freie Energie“, „Entropie“, „Kraftfeld“, und der Unterschied zwischen „Kraft“ und „Arbeit“ verstaendlich diskutiert sein.

Das Protokoll besteht aus

1. Einleitung
2. Methoden und Vorgehensweise; hier sollen die obigen Begriffe diskutiert werden.
3. Antworten zu den Fragen
  - a. Fuegen sie ebenfalls Graphiken an, welche  $F(z)$  und  $F(c)$  zeigen
4. Interpretation der Resultate im Hinblick auf Experimente
5. Graphische Darstellungen
6. Referenzen

Zu 4. sind auch graphische Darstellungen des Simulationssystems anzufuegen. Beachten Sie, dass Sie die Computer-Prozeduren nicht zu detailliert beschreiben (Oeffnen von Files, Spechern der Grafik, etc. ist nicht noetig). Wo immer Sie Formeln erwaehnen, stellen Sie sicher, dass diese korrekt sind. Referenzen sind zu nummerieren. Figuren haben eine Legende, welche die Figure verstaendlich macht und Bezug darauf nimmt.

