Einführung in Simulationen von Biomolekülen

Sabine Reißer and Tomáš Kubař

I. EINFÜHRUNG, VISUALISIERUNG UND GEOMETRIEOPTIMIERUNGEN

A. Grundlagen

Für dieses Praktikum werden die Linuxrechner im Seminarraum 110, Geb. 30.41 verwendet. Wir arbeiten unter Linux, einem freien Betriebssystem, das weniger verbreitet ist als z.B. Windows, aber für wissenschaftliche Anwendungen breite Verwendung findet. Im Unterschied zur bekannten mausgesteuerten Bedienung von Desktop-PCs ist Linux ein kommando-basiertes System, d.h. dass Nutzer den Rechner hauptsächlich über eine Textkonsole steuern. Dies ist wesentlich effizienter und schneller, bedarf aber einiger Übung. Vielfältige Informationen und Tutorien über Linux/Unix sind im Internet zugänglich, daher wird im folgenden nur ein minimaler Überblick über die üblichsten Befehle gegeben, die im Laufe des Praktikums gebraucht werden.

Zunächst fahren Sie einen der Praktikumsrechner hoch. Dann loggen Sie sich auf dem Rechner ein. Dazu verwenden Sie als Benutzernamen 'student' und als Passwort 'student'. Der Rechner startet eine graphische X-Windows Oberfläche. **Bitte beachten Sie:** Der Rechner muss nie heruntergefahren werden, und Sie müssen sich während des Praktikums nie abmelden. Um Textkommandos eingeben zu können starten Sie dann noch eine sog. Shell, ein Fenster verknüpft mit einem Kommandointerpreter. Diese finden Sie im Menu unter Favorites. In der dann startenden Shell können Sie zunächst die grundlegendsten Unixbefehle üben, z.B.:

- `whoami` – Zeigt den Benutzernamen an, unter dem Sie eingelogged sind, also im Moment *student*. Unterschiedliche Nutzer haben verschiedene Rechte und Zugriffsmöglichkeiten auf Systemressourcen, was uns hier aber nicht weiter beschäftigen soll.
- `pwd` – Zeigt das aktuelle Verzeichnis an, im Moment das home Verzeichnis, in dem Sie volle Schreib- und Leserechte haben und in dem Sie alle Dateien für das Praktikum ablegen sollten.
- `date` – gibt Uhrzeit und Datum aus
- `ls` – zeigt den Inhalt eines Verzeichnisses an. Ihr Homeverzeichnis sollte im Moment leer sein (oder Dateien früherer Nutzer enthalten). Es gibt verschiedene Modi für `ls`, die mit sog. Switches gewählt werden. Z.B. gibt `ls -l` zusätzliche Informationen über Dateien, wie Nutzerrechte aus, oder `ls -ltrh` sortiert die Dateien so, dass die zuletzt erstellte Datei am Ende der Liste steht.
- `man befehl` – Zeigt das Handbuch für einen bestimmten Befehl an. Wenn Sie wissen möchten, welche Optionen (Switches) ein bestimmter Befehl akzeptiert, können Sie dies mit `man` erfahren.
- `mkdir name` – Erstellt Verzeichnisse. Erstellen Sie in ihrem Home-Verzeichnis für die Übungen ein Subverzeichnis namens uebungen.
- `cd name` – Wechselt das Arbeitsverzeichnis. Wechseln Sie mit `cd uebungen` in das gerade erstellte Verzeichnis. Der Name des Zielverzeichnisses kann relativ zum Ausgangsverzeichnis angegeben werden, oder auch absolut, also `cd /home/student/uebungen`. Mit `cd ..` wechseln Sie in das Verzeichnis über dem aktuellen, mit `cd` ohne Name zurück in ihr Home-Verzeichnis. Ein einzelner Punkt `.` steht dabei für das aktuelle Verzeichnis.
- `cp oldname newname-or-location`, `mv oldname newname-or-location`, `rm name` – Dateioperationen um Dateien zu kopieren, verschieben (umbenennen) oder zu löschen. `cp name1 name2` erstellt eine Kopie der Datei name1 unter name2. Dabei können auch unterschiedliche Verzeichnisse angegeben werden. Mit `mv name1 name2` würde eine Datei entsprechend verschoben oder umbenannt werden; ¡name2¿ ist entweder das Zielverzeichnis oder der neue Dateiname. Dateien löschen funktioniert mit `rm name`, wobei Vorsicht geboten ist, denn löschen kennt keine Sicherheitsabfrage und kann nicht rückgängig gemacht werden. Komplette Verzeichnisse können mit `rm -r ordnername` gelöscht werden.
- `tar` – Packen, entpacken und archivieren von Dateien. Mit diesem Befehl können Dateien in Archive verpackt und komprimiert werden, z.B. zum einfacheren Versenden oder für Sicherheitskopien.

- `cat`, `more`, `less`, `head`, `tail name` – Betrachten von Textdateien. `cat` gibt die gesamte Datei auf einmal auf den Bildschirm aus, für längere Dateien bieten sich `more` oder `less` an, was seitenweises Scrollen erlaubt. `head` und `tail` geben jeweils die ersten und letzten zehn Zeilen der Datei aus.
- `wc name` – Wörter zählen. Gibt die Zeilen, Wörter und Zeichenzahl in einer Textdatei an.
- `grep` – Dateien durchsuchen. Mit `grep muster datei` wird die Datei nach Zeilen durchsucht in denen die Zeichenkombination muster vorkommt. Beispielsweise gibt

  ```
  grep HA /usr/local/run/gromacs-5.0-dftb-v6a-plumed/share/gromacs/top/*/ffbonded.itp
  ```

  alle Zeilen aller Kraftfelddateien für Bindungsparameter an, die sich auf einen bestimmten Wasserstoffatomtyp beziehen.
- `gedit`, `vim`, `nano`, `pico`, `emacs` – Texteditoren. Um den Inhalt von Textdateien zu editieren gibt es eine Vielzahl von Texteditoren, die zu groß ist um hier auch nur ansatzweise aufgeführt zu werden. Im Rahmen der Übungen genügt es, z.B. den einfachen Editor `gedit` zu verwenden. Wer weiter auf Linuxsystemen arbeiten möchten, sollte sich aber früher oder später mit einem mächtigeren Editor wie `emacs` oder `vim` anfreunden. Betrachten Sie zur Übung die PDB Datei `1HSG.pdb`.
- `alias name='befehl'` – Weist einem Befehl ein Kürzel zu. So könnten Sie z.B. mit `alias ll='ls -ltrh'` eine Kurzform für den detaillierten Dateilistenbefehl definieren.
- `history` – Befehlsgeschichte. Sie können auch mit der ↑-Taste zu früher eingegebenen Befehlen zurück.
- Mit Strg-Shift-T können Sie in neues Tab in der Shell öffnen. Das ist praktisch, wenn eine Simulation läuft und Sie an einer anderen Aufgabe weiterarbeiten möchten.
- Ebenfalls sehr praktisch ist die Möglichkeit, einen mit der linken Maustaste ausgewählten Text anschließend mit der mittleren Maustaste (oder Rad) zu kopieren. Dies geht oft schneller als die von Windows bekannte Sequenz von Strg-C Strg-V, welche allerdings in modernen Linux-Systemen auch funktioniert.
- Mit Strg-R können Sie nach einem Muster in der Befehlsgeschichte suchen.
- Mit Tab können Sie Dateinamen oder Befehle automatisch vervollständigen.
- `mc` – Midnight Commander, ein sehr praktischer Dateimanager, ähnlich zu Norton Commander oder Total Commander. `mc` enthält einen einfachen Texteditor, der mit F4 aufgerufen wird. Wir würden Ihnen gern empfehlen, `mc` so oft wie möglich einzusetzen.
- `>` – Das größer-als Zeichen leitet die Ausgabe aus einem Programm oder Befehl in eine Datei. Wenn sie den Inhalt des aktuellen Verzeichnis in einer Textdatei abspeichern möchten, tun Sie das mit `ls > was-ist-hier.txt`
- `|` – Der senkrechte Strich leitet die Ausgabe auf die Standardeingabe von einem anderen Programm oder Befehl. Diese sog. Pipe kann man auch mehrere Male auf der Kommandozeile benutzen. So kann man z.B. abzählen, wie oft man sich als Benutzer 'student' auf dem Rechner angemeldet hat: `last | grep "student" | wc -l`

ACHTUNG: Wenn Sie noch NIE unter Linux gearbeitet haben, nehmen Sie sich eine Stunde Zeit und machen Sie einen kleinen Online-Kurs, z.B. hier http://www.ernstlx.com/linux90bash1.xhtml. Das wird Ihnen die folgenden Aufgaben erheblich erleichtern.

B. Visualisierung mit VMD

Eine der wichtigsten Techniken für molekulare Simulationen ist die anschließende Visualisierung der berechneten Strukturen und Trajektorien. Wir werden dazu das Molekülbetrachtungsprogramm VMD verwenden (das unter `http://www.ks.uiuc.edu/Research/vmd` frei erhältlich ist) Bevor wir eigene Moleküle bauen, wollen wir zunächst eine fertige Röntgenkristallstruktur betrachten. Starten Sie zunächst das Programm aus der Shell heraus mit

```
vmd
```

Es öffnet sich ein Betrachtungs- und ein Kontrollfenster. Öffnen Sie den Dateibrowser unter *File – New Molecule*. Im neu geöffneten Fenster wählen Sie *Determine file type – Web PDB Download* und laden die Röntgenkristallstruktur Datei 1HEL, oder gerne eine andere, z.B. die von Ihrem Lieblingsprotein. Danach wird die Kristallstruktur im Betrachtungsfenster angezeigt. Testen Sie an diesem Beispiel die verschiedenen Funktionen von VMD:

- Drehen, Verschieben und Skalieren: Mit gedrückter linker oder rechter Maustaste lässt sich das Molekül drehen. Durch Drücken der `t`-Taste gelangen Sie in den Translationsmodus, mit `s` in den Skalierungsmodus und mit `r` zurück in den Drehmodus. Mit `=` wird die Ansicht zurückgesetzt.
- Auswahl von Teilen des Moleküls: Öffnen Sie unter *Graphics – Representations* das Darstellungsfenster. In der Zeile *Selected Atoms* probieren Sie verschiedene Selektionen aus: "water", "backbone", "resid 1 to 10", "resname CYS", etc. Selektieren Sie am Ende nur die beiden Cysteine Nummer 30 und 115.
- Namen und innere Koordinaten: Mit den Tasten `1`, `2`, `3`, `4` wählen Sie aus, Informationen über einzelne Atome, Abstände, Winkel und Dieder zu bekommen. Drücken Sie `1` und wählen Sie eines der Schwefelatome an. In der Konsole werden einige Informationen über dieses Atom angezeigt. Messen Sie weiterhin die Distanz (mit `2`) zwischen den Schwefelatomen, die beiden S–S–Cβ Winkel (Modus `3`) und den Cβ–S–S–Cβ Diederwinkel (Modus `4`). Sollte der Bildschirm mit Labels überfüllt sein, so kann man diese unter *Graphics – Labels* wieder löschen.
- Verschiedene Darstellungsmöglichkeiten: Setzen Sie die Selektierung zurück auf "all". Wählen Sie nun "Secondary Structure" und "NewCartoon" als Farb- und Zeichenstil. Wählen Sie ein paar weitere Darstellungsmöglichkeiten für das Molekül und generieren Sie eine Ansicht Ihrer Wahl, z.B. eine in der die Disulfidbrücken gut zur Geltung kommen, oder eine in der die Verteilung geladener Reste im Protein ersichtlich wird.

Für eine ausführlichere Beschreibung der (vielfältigen) Darstellungsmöglichkeiten mit VMD experimentieren Sie weiter oder nehmen Sie den User Guide zu Hilfe. VMD eignet sich gut dazu, hochwertige Strukturbilder für Publikationen über das *File – Render* Menü zu erstellen. Wir werden im Verlauf des Praktikums noch von den Möglichkeiten Gebrauch machen, Trajektorien darzustellen und Strukturen zu vergleichen.

C. Moleküle erstellen

Um im folgenden Geometrieoptimierungen oder Molekülдynamiksimulationen zu starten, benötigen Sie im wesentlichen zwei Dinge: Startgeometrien und Kraftfeldparameter. Startgeometrien stammen normalerweise aus experimentellen Daten, z.B. Röntgen- oder NMR-Strukturen, oder aus anderen Modellierungstechniken. Kraftfeldparameter sind normalerweise Bestandteil des verwendeten Modellingpaketes, können prinzipiell aber auch extra erstellt werden. Für die Vorbereitung der Eingabedateien sind im Amberpaket zwei Programme verantwortlich: Leap und Antechamber. Letzteres dient dazu, neue Moleküle zu generieren und Parameter dafür anzupassen, und wird in diesen Übungen nicht verwendet. Wir werden die graphische Version von ersterem (xLeap) verwenden, um ein Peptid zu bauen und als pdb-Datei zu speichern.

```
xleap
```

Als nächstes laden Sie eines der Amber Kraftfelder (eine verbesserte Version des amber99-Kraftfelds) und betrachten einige der vordefinierten Einheiten darin. (Sog. Einheiten sind Atome, Moleküle oder Teile von Molekülen, sowie Parameter.)

```
source leaprc.ff14SB
list
desc ALA
desc ALA.1
desc ALA.1.3
edit ALA
edit TIP3PBOX
```

Üben Sie das Betrachten, Markieren und Editieren von Molekülen und Atomen mit dem "edit" Fenster von xLeap. Das Programm stellt verschiedene Möglichkeiten zur Verfügung, solvatisierte und periodische Systeme zu erstellen, momentan beschränken wir uns allerdings auf Systeme im Vakuum.

Für die folgenden Rechnungen erstellen Sie nun das Peptid K2 als neue Einheit `peptid` mit dem Kommando

```
source leaprc.ff14SB
peptid = sequence { NLYS ILE ALA GLY LYS ILE ALA LYS ILE ALA GLY LYS ILE ALA NHE }
```

Im Prinzip hätte man das Molekül auch von Hand in eine leere Einheit zeichnen und dann alle Atomtypen und -ladungen definieren können, aber die vorgefertigten Aminosäurereste aus dem Kraftfeld erleichtern diese Arbeit natürlich enorm.

Als Startstruktur für die Simulationen möchten wir mit einer idealen α-Helix arbeiten, daher muss das Peptid noch entsprechend gefaltet werden. Die Diederwinkel φ und ψ sind entscheidend für die Sekundärstruktur von Proteinen. Im Gegensatz zur relativ starren Peptidgruppe selbst sind mehrere verschiedene Konformationen der beiden Diederwinkel denkbar. φ ist der N-terminale der beiden, der die Rotation entlang der N–Cα Bindung beschreibt (Diederwinkel C(vorherige AS)–N–Cα–C), ψ beschreibt analog die Rotation entlang Cα–C (Diederwinkel N–Cα–C–N(nächste AS)):

Wir setzen die Diederwinkel der Peptidbindung auf die Werte für eine ideale α-Helix (schauen Sie die in einem Buch oder im Internet nach):

```
impose peptid { 1 2 3 4 5 6 7 8 9 10 11 12 13 14 } {
{ "N" "CA" "C" "N" Wert-für-psi }
{ "C" "N" "CA" "C" Wert-für-phi }
}
```

Nun sollte K2 eine α-Helix sein. Speichern Sie die Struktur mit

```
savepdb peptid k2.pdb
```

Sie können auch eine Reihe von Befehlen in einer Textdatei abspeichern und diese mit

```
xleap -f Dateiname
```

sequentiell abarbeiten lassen. Ein Beispiel dazu ist die Eingabedatei `leap.in`.

Weitere Möglichkeiten von xLeap entnehmen Sie dem User Guide `Amber-Tools.pdf`.

Betrachten Sie das Peptid nun mit VMD und überprüfen Sie, ob alles so aussieht wie erwartet. Schauen Sie sich auch die Datei selbst mit einem Texteditor (z.B. Nano oder Pico) an. Versuchen Sie, sich folgende Fragen zu beantworten:

- Welche Spalte bedeutet was?
- In welcher Einheit sind die Koordinaten angegeben?
- Ist das Peptid amidiert?

D. Parametrisierung mit pdb2gmx

Die PDB Datei, die wir erstellt haben, enthält nur die Struktur des Moleküls, aber keine Atomladungen, Massen, Bindungsinformationen etc. Für die Parametrisierung wie auch für alle weiteren Schritte benutzen wir ab jetzt das MD-Simulationsprogramm Gromacs. Gromacs ist wie alle anderen bisher verwendeten Programme *open source* – das heißt, der Quellcode ist frei verfügbar und jeder kann ihn nach seiner Präferenz verändern.

Wir verwenden jetzt das Gromacs tool `pdb2gmx`, um Kraftfeldparameter zu erhalten:

```
pdb2gmx -f k2.pdb -ignh -o k2.gro
```

Wir verwenden das Amber99SB-ILDN Kraftfeld und brauchen zunächst kein Wasser. Die Outputdatei `k2.gro` ist eine Übersetzung der PDB Datei ins Gromacs-Format. Ein Blick hinein zeigt, dass die Formate sehr ähnlich sind.

Interessanter ist die Datei `topol.top`. Sie enthält alle Kraftfeldparameter und die Topologie, also die Zusammensetzung des Systems. Zeilen mit einem Semikolon `;` am Anfang sind Kommentare und sind zur Information des Nutzers gedacht.

In dieser Datei werden unter `[ moleculetype ]` die Molekültypen definiert, in unserem Fall ist das das Peptid K2. Danach werden unter `[ atoms ]` die Atome aufgelistet, in der Reihenfolge der einzelnen Seitenketten. Für jedes Atom steht nun in Spalte 2 ein Atomtyp, in Zeile 7 eine Ladung und in Zeile 8 eine Masse.

Weiter unten werden unter `[ bonds ]` die Bindungen zwischen Atomen definiert. Die ersten zwei Zahlen stehen für die Indices der Atome wie unter `[ atoms ]` definiert. Die Bindungsstärken und -abstände sind im Kraftfeld definiert, was am Anfang der Datei eingebunden wurde:

```
; Include forcefield parameters
#include "amber99sb-ildn.ff/forcefield.itp"
```

Der Pfadname ist hier ausgehend vom aktuellen Verzeichnis oder vom Verzeichnis `/usr/local/run/gromacs-5.0-dftb-v6a-plumed/share/gromacs/top`. Hier finden Sie alle Kraftfelder und die dazugehörenden Parameter. Ganz unten in der Topologiedatei (bei dem Betrachtungsprogramm Less springt man über `G` zum Dateiende) finden wir nun die Zusammensetzung des Systems mit dem Namen "Protein", was unter `[ molecules ]` nur ein Molekül mit ebenfalls dem Namen "Protein" enthält.

E. Geometrieoptimierungen

Geometrieoptimierungen suchen ausgehend von einer Startgeometrie das nächstgelegene energetische Minimum. Diese Methoden können keine Energiebarrieren überwinden und beinhalten keine Korrektur für unterschiedliche Entropien von Zuständen. Trotzdem sind sie nützliche Hilfsmittel um modellierte Strukturen zu optimieren oder für MD-Simulationen vorzubereiten.

In Gromacs werden alle Simulationen inklusive der Geometrieoptimierung mit dem Preprozessor `grompp` vorbereitet und mit dem Programm `mdrun` ausgeführt.

`grompp` benötigt verschiedene Eingabedateien, die über Switches zugeordnet werden. Die wichtigsten sind:

Switch	Dateiendung	Erklärung
`-f`	`.mdp`	Simulationsparameter
`-c`	`.gro`	Startstruktur
`-p`	`.top`	Topologie
`-n`	`.ndx`	Indexdatei
`-o`	`.tpr`	Output

In der `.mdp` Datei werden alle variablen Simulationsparameter gesetzt. Schauen Sie sich die Datei `em_steep_vacuo.mdp` an, auch hier sind Erklärungen zu finden. Die Indexdatei mit der Endung `.ndx` enthält verschiedene Gruppen innerhalb des simulierten Systems. Wir werden sie erst später brauchen, wenn wir mit Wasser und Membranen arbeiten.

Rufen Sie jetzt den Preprozessor auf:

```
grompp -f em_steep_vacuo.mdp -c k2.gro -p topol.top -o em_steep_vacuo.tpr
```

Es ist eigentlich nicht nötig, die Dateiendungen explizit anzugeben, da Gromacs diese automatisch ergänzt. Der gleiche Befehl kann also auch lauten:

```
grompp -f em_steep_vacuo -c k2 -p topol -o em_steep_vacuo
```

Im Folgenden werden wir die Dateiendungen grundsätzlich weglassen. Wenn keine Fehlermeldung kommt, starten Sie nun die Geometrieoptimierung mit

```
mdrun -v -deffnm em_steep_vacuo -nt 1
```

Der Switch `-v` steht für "verbose", also "wortreich", und gibt zusätzliche Informationen auf dem Bildschirm aus; dieser Switch ist bei allen Gromacs Programmen verfügbar. `-deffnm` bedeutet "default filename" und setzt bei allen Ausgabedateien den gleichen Basisnamen.

Hinweis: Bei allen Gromacs-Programmen kann man über den switch `-h` die Hilfe anzeigen. Hier findet sich eine Beschreibung, alle Ein- und Ausgabedateien und alle Optionen des Programms.

Plotten Sie mit `g_energy` den Verlauf der potentiellen Energie:

```
g_energy -f em_steep_vacuo -o em_steep_E_pot
```

Betrachten Sie die Datei mit Xmgrace:

```
xmgrace em_steep_E_pot.xvg
```

(dies kann direkt mit dem Switch `-w` für `g_energy` aufgefordert werden).

Vergleichen Sie nun die verschiedenen verfügbaren Minimierungsalgorithmen. Erstellen Sie zwei Kopien der Datei `em_steep_vacuo.mdp` unter den Namen `em_cg.mdp` und `em_bfgs.mdp`. Ändern Sie in den Dateien den 'Integrator' von `steep` für Steepest descent auf `cg` für Conjugate Gradient bzw. auf `l-bfgs` für den Broyden–Fletcher–Goldfarb–Shanno-Algorithmus (Vorsicht, das erste Zeichen im `l-bfgs` ist ein 'ell', nicht 'eins'). Führen Sie wie oben diese Geometrieoptimierungen durch (ändern Sie entsprechend den Ausgabenamen) und vergleichen Sie den Verlauf der potentiellen Energie, indem Sie Xmgrace alle drei `.xvg` Dateien als Argumente übergeben. Mit dem Zusatz `-legend load` zu dem `xmgrace` Befehl können Sie sich eine Legende anzeigen lassen.

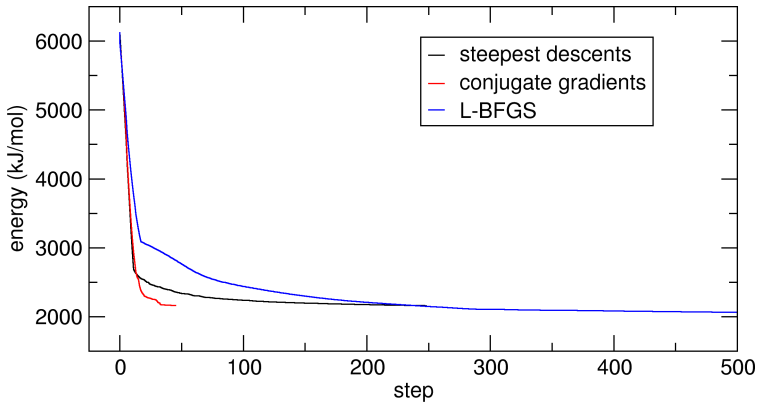

FIG. 1: Vergleich verschiedener Minimierungsalgorithmen. Die unterschiedlichen Algorithmen erreichen das nächstgelegene lokale Minimum unterschiedlich schnell. Das Minimum muss nicht dasselbe sein (z.B. L-BFGS konvergiert in ein tieferes Minimum).

II. MOLEKÜLDYNAMIKSIMULATIONEN

A. Ein Peptid im Wasser simulieren

In Moleküldynamiksimulationen wird, im Gegensatz zu Geometrieoptimierungen, die zeitliche Entwicklung eines molekularen Systems möglichst realistisch simuliert. Solche Simulationen sind in der tatsächlichen Anwendung sehr zeit- und rechenintensiv. Wir werden daher im Praktikum selbst nur kurze Zeitentwicklungen simulieren. Längere Trajektorien zur Auswertung finden Sie dann jeweils im Praktikumsverzeichnis.

Erstellen Sie nun einen Ordner `part2`. Kopieren Sie eine der geometrie-optimierten Strukturen des Peptids K2 und die Topologie-Datei aus der letzten Übung in dieses Verzeichnis.

Wir packen in dieser Übung das Peptid K2, das Sie im letzten Praktikumsteil erstellt und geometrie-optimiert haben, in eine Wasserbox, heizen diese auf, equilibrieren das System und starten eine kurze Simulation.

1. Solvatation

Zunächst packen wir das Peptid in eine kubische Box:

```
editconf -f em_steep_vacuo -o k2_newbox -c -d 1.0 -bt cubic
```

Lesen Sie unter `editconf -h` die verwendeten Optionen nach. Schauen Sie sich mit VMD die neue Struktur an. Wenn Sie `pbc box` in die VMD-Konsole eingeben, sehen Sie die Kanten der erstellten Box.

Nun füllen wir die Box mit Wasser. Die verschiedenen Wassermodelle können Sie unter `en.wikipedia.org/wiki/Water_model` nachschauen; wir benutzen das TIP3P Modell.

```
gmx solvate -cp k2_newbox -p topol -cs spc216 -o k2_sol
```

Lesen sie auch hier unter `gmx solvate -h` die Optionen nach. Die verwendete Wasserbox SPC216 enthält 216 Wassermoleküle und ist mit dem SPC Modell equilibriert (die Modelle SPC und TIP3P sind sehr ähnlich). Das Programm versucht möglichst viele dieser kleinen Boxen in der Peptidbox unterzubringen. Schauen Sie sich die solvatisierte Box wieder mit VMD an.

Wenn Sie sich nun mit `tail topol.top` das Ende der Topologie ausgeben lassen, sehen Sie unter `[ molecules ]` die Anzahl der hinzugefügten Wassermoleküle. Wir müssen jetzt noch die Molekül definition für TIP3P Wasser in die Topologie einfügen. Ergänzen Sie mit einem Texteditor vor `[ system ]` die drei Zeilen

```
;Include water topology
#include "amber99sb-ildn.ff/tip3p.itp"
#include "amber99sb-ildn.ff/ions.itp"
```

Jetzt müssen Sie das System neutralisieren, weil das Peptid eine positive Ladung trägt. (Frage: Welche Aminosäuren sind dafür zuständig?) Dazu brauchen Sie eine beliebige Inputdatei, z.B. eine leere Datei, die wir mit diesem Befehl erzeugen:

```
echo > dummy.mdp
```

Dann können Sie eine `.tpr` Datei erzeugen, und diese schließlich dem Programm geben, das die Ionen hinzufügt:

```
grompp -f dummy -o dummy -p topol -c k2_sol
genion -s dummy -o k2_ion -p topol -neutral
```

Dieses Programm ersetzt die benötigte Anzahl Wassermoleküle durch Cl^- Ionen. Geben Sie hierzu an, welche der Gruppen Wasser bezeichnet (SOL, wahrscheinlich die `13`).

2. Equilibrierung

Um das System für eine richtige Simulation vorzubereiten, müssen jetzt einige Equilibrierungsschritte durchgeführt werden. Zunächst machen wir wieder eine Energieminimierung, um eine vernünftige Startstruktur zu erhalten. Benutzen Sie dazu die Datei `em_steep.mdp`, die noch nicht eingesetzt wurde. Sie können folgende Befehle durchführen:

```
grompp -f em_steep -c k2_ion -p topol -o em
mdrun -v -deffnm em
```

Nun wird die energieminimierte Struktur bei konstantem Volumen aufgeheizt (NVT Simulation). Dazu wird das Peptid festgehalten, so dass sich nur die Wassermoleküle um es herum bewegen können. Um das Peptid festzuhalten, erstellen wir sog. Position Restraints für jedes Peptid-Atom mit `genrestr`:

```
genrestr -f em
```

Hier können Sie die schweren Atome (alle außer H) des Peptids berücksichtigen, d.h. die Gruppe "Protein-H". Wie Sie in der erstellten Datei `posre.itp` sehen können, wird jetzt die Position jedes schweren Peptid-Atoms durch ein harmonisches Potential (Feder) mit der Kraftkonstante 1000 kJ/(mol·nm^2) mit seiner Anfangsposition festgehalten. (Siehe auch Gromacs Manual, Abschnitt 4.3.1.)

Schauen Sie sich nun die Datei `nvt.mdp` an. Auch hier finden Sie Auskunft zu den einzelnen Parametern. Hier einige zusätzliche Informationen:

- `define = -DPOSRES`
  Hier wird die Variable POSRES definiert, also Position Restraints werden angewendet. Suchen Sie in der Datei `topol.top` nach "POSRES", dort wird die von Ihnen erstellte Datei `posre.itp` eingebunden. Wenn Sie später eine freie MD Simulationen muss diese Zeile in der entsprechenden `.mdp` Datei auskommentiert oder gelöscht sein.

- `gen_temp = 10.`
  Alle Atome bekommen am Anfang der Simulation durch Geschwindigkeiten stochastisch (zufällig) zugewiesen, deren Verteilung der Maxwell–Boltzmann-Verteilung für die erwünschte Temperatur enstpricht. Bei 10 K sind die Geschwindigkeiten gering und es kann keine Atombewegung in einer ungeschickten Richtung entstehen.

- `ref_t = 300. 300.`
  Im Verlauf der Simulation werden die Geschwindigkeiten über einen Thermostaten so skaliert, dass die Verteilung möglichst konstant der MB-Verteilung für die erwünschte Temperatur `ref_t` entspricht.

Führen Sie nun wieder den Preprozessor und die eigentliche Simulation aus:

```
grompp -f nvt -p topol -c em -o nvt
mdrun -v -deffnm nvt
```

Wie Sie über die Bilschirmausgabe sehen können, dauert die NVT Equilibrierung etwa eine Minute.

Schauen Sie sich nun wieder mit `g_energy` und Xmgrace den zeitlichen Verlauf der Temperatur an (analog zu Kap. I.E). Ihr Diagramm sollte in etwa wie Abb. 2 (links) aussehen. Sie sehen, dass bei 0 ps die Temperatur sehr niedrig ist. Die Ursache dafür ist die Zuteilung der Geschwindigkeiten gemäß der Boltzmann-Verteilung für 10 K. Durch den verwendeten Thermostat wird dann das System während der ersten ca. 2 ps auf die Zieltemperatur von 300 K aufgeheizt.

Erstellen Sie mit `g_energy` auch Diagramme vom Verlauf der potentiellen und kinetischen Energie, die ähnlich wie in Abb. 2 (rechts) aussehen werden. Nach einen kleinen Absenkung steigt die potentielle Energie, und illustriert so den Fakt, dass das Molekülsystem sich während der Simulation nicht mehr exakt im Energieminimum aufhält. Die kinetische Energie verläuft gleich wie die Temperatur. (Warum?)

Nun muss das System noch auf den richtigen Druck von 1 bar gebracht werden. In diesem Equilibrierungsschritt wird gleichzeitig der Druck und die Temperatur konstant gehalten, daher nennt man ihn auch NPT Equilibrierung.

Schauen Sie sich die Datei `npt.mdp` an. Das Peptid wird weiterhin mit Position Restraints festgehalten. Die Zeile

```
continuation = yes
```

zeigt an, dass die Endgeschwindigkeiten der letzten Equilibrierung jetzt als Startgeschwindigkeiten genutzt werden. Es gibt jetzt einen neuen Block zur Kontrolle des Drucks:

```
pcoupl           = Parrinello-Rahman
pcoupltype       = isotropic          ; uniform in x,y,z directions
tau_p            = 0.5                ; relaxation time in ps
ref_p            = 1.0                ; desired pressure in bar
compressibility  = 4.5e-5             ; value for water in 1/bar
refcoord_scaling = com
```

Zum Ausführen der Equilibrierung benutzen wir wieder `grompp` und `mdrun`:

```
grompp -f npt -c nvt -p topol -o npt
mdrun -v -deffnm npt
```

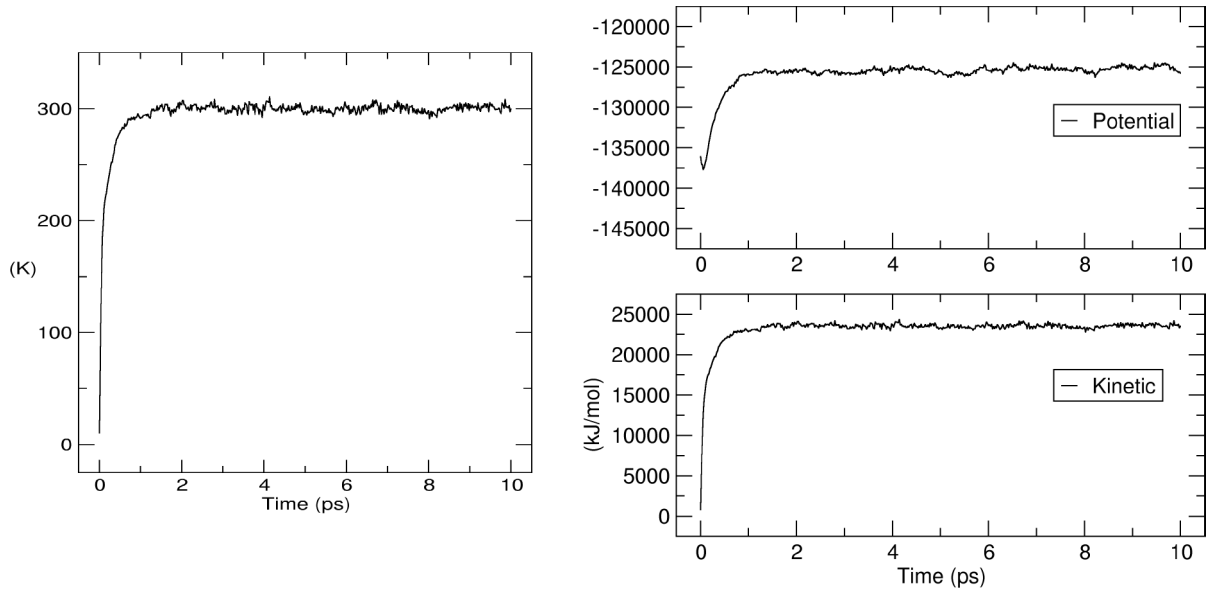

FIG. 2: NVT Equilibrierung: Verlauf der Temperatur (links), sowie der potentiellen und kinetischen Energie (rechts).

Diese 100 ps lange Equilibrierung wird etwa 5 Minuten dauern. Schauen Sie sich danach wieder mit `g_energy` und Xmgrace die Entwicklung der Temperatur und zusätzlich der Dichte an. (Oder alternativ des Volumens. Die Information ist die gleiche wie bei der Dichte, die allerdings als intensive Eigenschaft vielleicht mehr informativ ist.) Ist das System ausreichend equilibriert? Innerhalb welcher Grenzen schwanken Temperatur und Dichte?

3. MD Simulation

Nachdem wir unser System nun ausreichend equilibriert haben, starten wir eine MD Simulation, in der wir das Peptid nicht mehr festhalten. Betrachten Sie die dazugehörige Eingabedatei `md.mdp`. Was hat sich verändert? Woran sehen Sie, dass das Peptid sich nun frei bewegen kann? Welche Zeitspanne wird simuliert? Benutzen Sie nun `npt.gro` als Startstruktur und starten Sie die Simulation. Die Simulation wird etwa eine Stunde dauern.

B. Visualisierung

Verwenden Sie nun VMD um den Verlauf der Trajektorie zu visualisieren. Übergeben Sie dazu VMD als Eingabeparameter erst eine Startstruktur und dann die Trajektorie:

```
vmd npt.gro md.xtc
```

Experimentieren Sie mit dem Trajektorienplayer im Fenster “VMD Main”.

Blenden Sie zuerst die Wassermoleküle aus. Die Dynamik des Peptids ist zunächst noch schwer zu verfolgen, da die Rotation des Moleküls alle anderen Konformationsänderungen überdeckt, dies lässt sich aber mit Hilfe des *RMSD-Tools* in VMD beheben. Wählen Sie unter *Extensions – Analysis* das Fenster *RMSD Trajectory Tool*. Selektieren Sie das Peptid (mit “protein”) und wählen Sie *Align*. Damit sollte die Trajektorie translations- und rotations-bereinigt sein.

Speichern Sie den berechneten RMSD Verlauf der Simulation und plotten Sie die erhaltene Datei (Sie müssen dazu vermutlich die ersten beiden Zeilen der Datei entfernen). Verfolgen Sie als nächstes die Änderungen der beiden Peptiddiederwinkel φ un ψ in Ihrem Modell zunächst visuell. Diese beiden Diederwinkel sind entscheidend für die Sekundärstruktur von Proteinen (φ: C(–1)–N–Cα–C, ψ: N–Cα–C–N(+1)). Markieren Sie diese in VMD im Diedermodus (Taste `4`). Wählen Sie im Fenster *Main* unter *Extensions – Analysis* das Fenster *Ramachandran Plot*. Im neuen Fenster müssen Sie unter *Molecule* nochmal “npt.gro” anwählen, dann sollten Sie die Position des aktuell angezeigten Koordinatensets im Ramachandran Plot sehen. Verfolgen Sie die Bewegung des Peptids in diesem Konformationsraum während der Simulation indem Sie im Fenster *Main* die Trajektorie abspielen lassen. Die normalerweise bevorzugten

Konformationen für Proteine sind angezeigt, ein einzelnes Peptid kann aber durchaus stark davon abweichen. Ordnen Sie den farblich unterlegten Feldern Konformationen zu.

C. Ramachandran Plot und Zeitserien für Diederwinkel

Zur Analyse der Struktur und Dynamik können Sie auch bequem die entsprechenden Gromacs Programme benutzen. So berechnen Sie die RMS Abweichung von der Startstruktur

```
g_rms -s md -f md.xtc
```

(am besten zweimal "Protein-H" auswählen), und so erzeugen Sie den Ramachandran Plot

```
g_rama -s md -f md.xtc
```

Die resultierenden `.xvg` Dateien sind mit `xmgrace` zu betrachten. In dem Ramachandran Plot können Sie die Symbole ändern und verkleinern, um das Bild deutlicher zu machen.

In welchem Bereich des Ramachandran Plot finden Sie die dominante Konformation? Gibt es vielleicht auch eine andere Konformation, die zwar weniger populiert aber trotzdem noch relativ deutlich ist? Wenn ja, wie wird diese meistens bezeichnet?

Als nächstes werden die Zeitserien der beiden Peptiddiederwinkel φ und ψ aus der Simulation extrahiert. Dazu verwenden wir das Gromacs Programm `g_angle`. Wie Sie unter `g_angle -h` nachlesen können, benötigen Sie dazu eine Indexdatei, in der die zu einem Diederwinkel gehörenden Atom-Indices aufgelistet werden. In der Datei `dihedrals.ndx` wurde diese Liste schon begonenn. Vervollständigen Sie sie.

Schauen Sie sich nun die Datei `calc_dihedrals.sh` an. Hier wird `g_angle` für jeden Diederwinkel in einer Schleife aufgerufen. Führen Sie die Datei mit den erforderlichen Eingabedateien aus.

Plotten Sie die Ausgabedateien und vergleichen Sie die Ergebnisse der `g_angle` Berechnung mit den Werten von VMD. Berechnen Sie alternativ auch den Winkel über eine Peptidbindung (ω), der zwischen Cα–C–N(+1)–Cα(+1) gemessen wird.

D. Normalmodenanalyse mit MMTK

Alternativ zu MD Simulationen bieten Normalmodenanalysen eine Möglichkeit, sich einen Überblick über den Konformationsraum eines Moleküls zu verschaffen. Zudem lässt sich aus diesen, zumindest in harmonischer Näherung, die Konfigurationsentropie eines Moleküls berechnen, eine normalerweise nur schwer zugängliche Größe. Wir werden für ein kleines Testmolekül die Normalmoden zunächst mit Hilfe von MMTK berechnen und dann mit Hilfe von VMD visualisieren. MMTK ist komplett in der Programmiersprache Python geschrieben, die sich durch besonders gute Programmlesbarkeit auszeichnet. So werden z.B. Codeblöcke, die in Schleifen oder bedingt ausgeführt werden, verpflichtend mit Tabulatoren eingerückt. Um die Ergebnisse von MMTK später mit VMD zu visualisieren, setzen Sie zunächst die entsprechende Umgebungsvariable:

```
export PDBVIEWER=vmd
```

Starten Sie Ihren Code jeweils mit

```
python -i example.py
```

1. Normalmoden eines Wassermoleküls

Zunächst werden wir die Normalmoden von H_2O berechnen. Erstellen Sie dazu eine Datei `water_view.py` und fügen Sie folgende Zeilen ein:

```
from MMTK import *
molecule = Molecule('water')
molecule.view()
```

Mit der ersten Zeile binden Sie die Kernmodule von MMTK ein. In der zweiten Zeile wird ein Wassermolekül erstellt, welches durch die Anweisung in der dritten Zeile angezeigt wird. Führen Sie den Code aus.

Nun berechnen wir die Normalmoden dieses Moleküls. Benutzen Sie dazu den Code in der Datei `water_modes.py`:

```
from MMTK import *
from MMTK.ForceFields import Amber99ForceField
universe = InfiniteUniverse(Amber99ForceField())
molecule = Molecule('water')
universe.addObject(molecule)
```

Vor der Normalmodenberechnung führen wir eine Energieminimierung aus, für die wir ein Kraftfeld brauchen. Das wird in der zweiten Zeile eingebunden. In der dritten Zeile wird unser *universe* definiert. Es enthält die Randbedingungen unserer Simulation: die Umgebung (*infinite* oder *periodic*), das Kraftfeld (hier *Amber99*), sowie den Thermostaten (hier keiner angegeben). In der letzten Zeile wird schließlich das Wassermolekül dem *universe* hinzugefügt.

Die folgenden Befehle dienen der Energieminimierung:

```
from MMTK.Minimization import ConjugateGradientMinimizer
from MMTK.Trajectory import StandardLogOutput
minimizer = ConjugateGradientMinimizer(universe, actions=[StandardLogOutput(1)])
minimizer(convergence = 1.e-4, steps = 100)
```

Mit den ersten zwei Zeilen werden wieder benötigte Klassen importiert. Dann wird ein Minimizer-Objekt für das *universe* erstellt. *actions* ist optional, hier wird angegeben, dass relevante Informationen bei jedem Minimierungsschritt auf der Konsole ausgegeben werden. Die letzte Linie startet die Minimierung, welche beendet wird, wenn die mittlere Kraft auf die Atome kleiner als 10^{-4} kJ/(mol·nm) ist, oder nach 100 Minimierungsschritten. Die Geometrie des Moleküls ist bereits sehr nah am Energieminimum, daher braucht die Minimierung in unserem Fall nur wenige Schritte.

Nun beginnt die Normalmodenberechnung:

```
from MMTK.NormalModes import VibrationalModes
modes = VibrationalModes(universe)
```

Mit

```
for mode in modes:
    print mode
```

werden die Frequenzen der einzelnen Moden ausgegeben. Die Einheit ist 1 THz = 1 ps^{-1}. Rechnen Sie die Frequenzen von THz in Wellenzahlen (cm^{-1}) um.

Die ersten sechs Moden haben sehr kleine Frequenzen (die mit einem j sind imaginär). Dies sind die Freiheitsgrade des starren Moleküls: drei Translations- und drei Rotationsfreiheitsgrade. Wie verhalten sich die Zeitskalen der Vibrationen zum typischen Zeitschritt in MD-Simulationen (1–2 fs)?

Um die Moden zu betrachten schauen Sie sich die Animation an:

```
modes[6].view()
```

Wie beschreiben Sie die drei nicht-trivialen Moden?

Als Nächstes betrachten wir die Änderung der Bindungslängen und des Bindungswinkels in den drei Moden:

```
d_O_H1 = universe.distance(molecule.O, molecule.H1)
d_O_H2 = universe.distance(molecule.O, molecule.H2)
a_H1_O_H2 = universe.angle(molecule.H1, molecule.O, molecule.H2)
```

Nun erzeugen wir eine Geometrie, in der die Koordinaten aller Atome sich entprechend der ersten Normalmode verändert haben:

```
minimum_geom = copy(universe.configuration())
displaced_geom = minimum_geom + modes[6]
```

und berechnen die internen Koordinaten der veränderten Geometrie:

```
universe.setConfiguration(displaced_geom)
displaced_d_O_H1 = universe.distance(molecule.O, molecule.H1)
displaced_d_O_H2 = universe.distance(molecule.O, molecule.H2)
displaced_a_H1_O_H2 = universe.angle(molecule.H1, molecule.O, molecule.H2)
universe.setConfiguration(minimum_geom)
```

Nun berechnen wir die Änderung der Abstände bzw. des Bindungswinkels:

```
print "Change in distance O-H1:", displaced_d_O_H1 - d_O_H1
print "Change in distance O-H2:", displaced_d_O_H2 - d_O_H2
print "Change in angle H1-O-H2:", displaced_a_H1_O_H2 - a_H1_O_H2
```

Wiederholen Sie die Berechnung auch für die anderen nichttrivialen Moden und vergleichen Sie die Ergebnisse mit Ihren Beobachtungen in VMD.

2. Normalmodenanalyse eines Peptidmoleküls

Um die Normalmoden des Peptids K2 zu berechnen, nutzen Sie die Datei `protein_modes.py`. Falls Sie eine Fehlermeldung bekommen (etwas mit 'amber_atom_type'), entfernen Sie die Amidgruppe am C-Terminus des Peptids. Dazu entfernen Sie die beiden Wasserstoffatome, und ändern Sie das N-Atom in ein Sauerstoffatom (Atomname OXT, Residuenummer und -name gleich wie das vorherige O-Atom).

Darin steht im Prinzip nichts Neues. Allerdings dauern die Berechnungen wegen der Größe des Moleküls länger, weshalb die minimierte Struktur und die Vibrationsmoden zwischengespeichert werden:

```
universe.writeToFile("minimized.pdb")
[...]
save(modes, 'peptide.modes')
```

So brauchen Sie die Berechnungen bei den nächsten Schritten nicht jedesmal abzuwarten. Kopieren Sie die von Ihnen in Übung I C erstellte PDB Struktur in ihr aktuelles Arbeitverzeichnis und ändern Sie in `protein_modes.py` entsprechend den Namen. Da dieses Molekül wegen seiner Größe Hunderte von Vibrationsmoden hat (wieviele genau?), ist es unpraktisch, sich die Frequenzen in der Konsole ausgeben zu lassen. Schreiben Sie sie daher in eine Datei, indem Sie hinzufügen:

```
newfile = open("modes.list", "w")
for mode in modes:
     newfile.write("%s \n" % mode)
newfile.close()
```

Die erste Zeile öffnet eine Datei `modes.list` im Schreibmodus (`write` bzw. `w`). Dann wird für jede Mode eine Zeile in die Datei geschrieben und mit einem Zeilenumbruch (`\n`) abgeschlossen. In der letzten Zeile wird das Datei-Handle geschlossen.

Führen Sie nun `protein_modes.py` aus und werfen Sie einen Blick in `modes.list`. Mit `protein_modes_2.py` laden Sie die gerade berechneten Moden und können sie sich mit `view()` (siehe oben) in VMD anzeigen lassen.

III. FORTGESCHRITTENE SIMULATIONEN

A. Peptid in einer Membran

1. Startstruktur

Die Datei `DOPC_303K.gro` enthält eine equilibierte, solvatisierte Membran-Doppelschicht. Berechnen Sie aus den Box-Koordinaten die Area per Lipid für dieses Lipid bei 303 K.

Kopieren Sie die von Ihnen in Kapitel I C gebaute PDB Struktur von K2 in Ihr Arbeitsverzeichnis, ebenso die Topologie aus Kapitel II A 1; hier ist das Wasser zu entfernen. Öffnen Sie die Peptid-Datei mit VMD und laden Sie unter *File – New Molecule* auch die Membran. Bewegen Sie mit `8` das Peptid so, dass es etwa eine Peptidlänge von den Kopfgruppen der Lipide entfernt über der Membran liegt. Markieren Sie im *Main* Fenster das Peptid und speichern Sie die Koordinaten unter einem neuen Namen. Verfahren Sie wie in Kapitel I D, um daraus eine `.gro` Datei zu erhalten.

Kopieren Sie die orientierte Peptid-Struktur nach `k2_DOPC.gro` und fügen Sie mit

```
cat DOPC_303K.gro >> k2_DOPC.gro
```

die Membran hinzu. Öffnen Sie die Datei mit einem Texteditor und passen Sie die Anzahl der Atome in der zweiten Zeile und die Boxkoordinaten in der letzten Zeile an. Übernehmen Sie die Boxkoordinaten der Membran. Löschen Sie die drei überflüssigen Zeilen nach der Definition des Peptids. Wieviele Lipidmoleküle sind in jeder Schicht? Wieviele C-Atome hat die DOPC-Kette?

Betrachten Sie die Struktur mit VMD. Momentan liegt das Peptid nicht im Wasser. Vergrößern Sie die Box in der z-Richtung um einen Nanometer und solvatisieren sie wie in Kapitel II A 1; die resultierende Datei soll `k2_dopc_sol.gro` heißen. Betrachten Sie in VMD die neue Struktur und vergewissern Sie sich, dass rund um das Peptid Wasser ist. Wenn nicht, vergößern Sie die Box noch mehr und solvatisieren Sie erneut. Notieren Sie sich, um wieviel Sie die Box vergrößert haben.

2. Topologie

Strukturieren Sie Ihre Topologiedateien um. Kopieren Sie dazu `topol.top` einmal nach `complete.top`, einmal nach `k2.itp`. `itp` bedeutet *include topology* und ist in der Regel die Endung der Dateien, die in der `.top` Datei für das Gesamtsystem eingebunden werden. Löschen Sie aus `k2.itp` jede Zeile mit einem `include` Befehl sowie die Direktive `[ system ]`. Diese Datei soll nur noch Informationen über das Peptid enthalten.

Löschen Sie im Gegenzug aus `complete.top` alle Peptidinformationen und binden Sie `k2.itp` ein; (dies geschieht durch den Befehl `include "k2.itp"`). Binden Sie auch `DOPC.itp` ein. Binden Sie wie im Kapitel II A 1 die Definition des TIP3P Wassermodells ein. Passen Sie unter `[ molecules ]` die Anzahl der Moleküle an. Fügen Sie vor `SOL` (Solvent) noch `DOPC` ein.

3. Gegenionen und Indexdatei

Neutralisieren Sie nun das zu simulierende System mit Cl^- Ionen, wie in Sektion II A 1. Dazu kann man wie vorher eine leere `.mdp` Datei benutzen:

```
echo > dummy.mdp
grompp -f dummy -o dummy -p complete -c k2_dopc_sol
genion -s dummy -o complete -p complete -neutral
```

Dann müssen die erwünschten Gegenionen sowohl in der Koordinatendatei `complete.gro` als auch in der Topologiedatei `complete.top` auftreten.

Erstellen Sie nun eine Indexdatei mit `make_ndx -f complete`. Erstellen Sie zwei zusätzliche Indexgruppen, `Protein_DOPC` sowie `Cl_SOL`, und speichern Sie die Änderung.

4. Simulationsparameterdatei für die Equilibrierung

Erstellen Sie nun die Parameterdateien für die Equilibrierung. Kopieren Sie dazu die `.mdp` Dateien aus den Kapiteln II A 2 und II A 3 in Ihr aktuelles Arbeitsverzeichnis. Fügen Sie in `nvt.mdp` zusätzliche Kopplungsgruppen hinzu:

```
; Temperature coupling is on
tcoupl  = V-rescale
tc-grps = Protein DOPC SOL
tau_t   = 0.1 0.1 0.1
ref_t   = 480. 480. 480.
```

Fügen Sie unten zur Datei hinzu:

```
; COM motion removal
; These options remove motion of the protein/bilayer relative to the solvent
nstcomm = 1
comm-mode = Linear
comm-grps = Protein_DOPC Cl_SOL
```

Ändern Sie auch die Simulationszeit auf 100 ps. Tun Sie das Gleiche mit der Datei `npt.mdp`. Ändern Sie aber die Zeitkonstante der Temperaturkopplung auf 0.5 ps für alle Gruppen, und ändern Sie den Abschnitt zur Kopplung des Drucks:

```
pcoupl          = Parrinello-Rahman
pcoupltype      = semiisotropic
tau_p           = 0.5 0.5
ref_p           = 1.0 1.0
compressibility = 4.5e-5 4.5e-5
```

Wie Sie sehen, wird $x - y$ jetzt unabhängig von z (d.h. semiisotrop) angepasst. Ändern Sie die Simulationszeit auf 1 ns und schreiben Sie alle zwei Pikosekunden die Koordinaten aus. Modifizieren Sie entsprechend auch `md.mdp`. Setzen Sie die Simulationszeit auf 10 ns, schreiben Sie alle 10 ps die Koordinaten aus. Wir simulieren jetzt bei 480 K, da das Peptid sich so schneller der Membran annähert und sich in diese einbaut. Die physikalischen Eigenschaften des Systems während dieser Simulation entsprechen wahrscheinlich nicht mehr der experimentellen Realität. Wieviel Grad Celsius sind 480 K?

5. Kraftfeld

Bisher haben wir die Simulationen mit dem Amber99SB-ILDN-Kraftfeld ausgeführt. Für die Lipide benutzen wir jetzt die neuen Stockhom lipids, die mit Amber Kraftfeldern kompatibel sind. Dazu müssen jetzt alle Parameter an die richtige Stelle.

`forcefield.ff.zip` enthält die Kraftfeldparameter für die Stockholm lipids. Entpacken Sie den komprimierten Ordner mit `unzip`. Binden Sie nun in `complete.top` die Lipid-Kraftfeldparameter mit ein:

```
#include "forcefield.ff/ffnonbonded.itp"
#include "forcefield.ff/ffbonded.itp"
```

Führen Sie `grompp` zur Vorbereitung der Energieminimierung aus. Falls Sie Warnungen bekommen, besprechen Sie diese mit einem Betreuer und ignorieren Sie sie gegebenenfalls mit `-maxwarn`.

6. Equilibrierung

Führen Sie die Energieminimierung und dann wie in Kapitel II A 2 die Equilibrierung des Systems (mit `complete.top` statt `topol.top`) durch. Vergewissern Sie sich vorher, dass Sie in in der NVT- und der NPT-Equilibrierung Position Restraints verwenden. (NVT: 1h22, NPT: ca. 11h30, EDIT: wird wahrscheinlich doppelt so schnell gehen).

7. Distance Restraints

Sie führen gleich eine Hochtemperatursimulation bei 480 K durch. Deswegen, und weil sich das Peptid anfangs im Wasser befindet, wird es sich wahrscheinlich während der Simulation entfalten. Um das zu vermeiden, verstärken wir die Wasserstoffbrücken, durch welche die α-Helix stabilisiert wird.

Vervollständigen Sie dazu die Datei `disre.itp`. Lesen Sie im Gromacs Manual das Kapitel zu Distance Restraints nach. Messen Sie mit VMD die Abstände der Wasserstoffbrücken und setzen Sie in `disre.itp` den Maximalwert ('up1') auf den jeweils gemessenen (type ist immer 1, index ist laufende Zahl, type' ist immer 2). Achtung: in VMD werden Abstände in ångström angegeben, in Gromacs durchweg in Nanometern. Wieviele Wasserstoffbrücken zählen Sie? Was ist der maximale Abstand der Wasserstoffbrückenpartner?

Während der Equilibrierung werden alle Peptid-Atome durch die Position Restraints festgehalten. Daher brauchen wir hier keine Distance Restraints. Erst in der freien Simulation muss die Struktur stabilisiert werden. Fügen Sie daher in `md.mdp` nach dem Titel ein:

```
define   = -DDISRES
disre    = simple
disre_fc = 10000
```

Und binden Sie `disre.itp` in `complete.top` ein:

```
#ifdef DISRES
#include "disre.itp"
#endif
```

8. Eintauchsimulation bei 480 K

Führen Sie nun die MD ohne Position Restraints, dafür mit Distance Restraints bei 480 K durch. Aufgrund der hohen Temperatur bettet sich das Peptid schnell in die Membran ein.

Schauen Sie sich die Simulation an. Welcher Terminus geht zuerst in die Membran? Hat sich die Area per Lipid verändert?

9. System verkleinern

Da das Peptid sich nun in der Membran befindet, haben wir evtl. viel überflüssiges Wasser in der Box. **Bitte beachten Sie:** Dies muss nicht der Fall sein. Falls die Geometrie des Systems es nicht erlaubt, Wasser zu entfernen, machen Sie gleich weiter mit dem Abkühlen im nächsten Abschnitt. Das überflüssige Wasser kann dann nach dem Abkühlen gelöscht werden.

Kopieren Sie die Ergebnis-Struktur der MD nach `md_no_water.gro`. Löschen Sie mit einem Texteditor alles Wasser sowie Gegenionen aus der Box. Vergleichen Sie die Box-Vektoren ihrer Startstruktur (vor der Energieminimierung) mit dem von `md.gro`. Wie hat sich die Form der Box verändert?

Reduzieren Sie den Box-Vektor z von `md_no_water.gro` und solvatisieren Sie das System jetzt wieder wie oben, so dass etwa 5100 ($\pm$100) Wassermoleküle in der Box sind. Ändern Sie in `complete.top` entsprechend die Anzahl der Wassermoleküle. Fügen Sie Gegenionen wieder wie in Sektion III A 3 hinzu.

10. Abkühlen

Kühlen Sie jetzt das System ab, indem Sie in `nvt.mdp`, `npt.mdp` und `md.mdp` die Temperatur auf 30 Grad Celsius ändern. Führen Sie die Equilibrierung wie oben inklusive Energieminimierung durch.

11. MD-Simulation bei 303 K

Sie würden nun wie oben die MD-Simulation ausführen. Damit Sie damit aber nicht soviel Zeit verlieren, gehen Sie direkt zur Auswertung über. Benutzen Sie dazu die Trajektoriedatei `long-with-ions-303.xtc` und die dazugehörige `.tpr` Datei.

- Berechnen Sie mit `g_dist` den zeitlichen Verlauf des Abstands des Peptidschwerpunkts zur Membranmitte.
- Berechnen Sie mit `g_helix` den zeitlichen Verlauf der Helizität.
- Berechnen Sie mit `g_bundle` den zeitlichen Verlauf der Winkel der CA–CB Bindungen der Alanine zur Membrannormale (z-Achse).
- Berechnen Sie aus dem Mittelwert der Winkel die zu erwartenden NMR-H2-Splittings. Vergleichen Sie die Werte der Simulation mit Ihren gemessenen NMR-Werten.

IV. FREIE ENERGIE RECHNUNGEN

A. Setup

Lesen Sie zur Einführung *us.pdf*.

Bei diesem Versuch soll das Potential of Mean Force für die Rotation um den Ψ-Winkel des Dialanins mit Hilfe von Umbrella Sampling berechnet werden. Bevor die eigentliche Freie Energie Rechnung durchgeführt werden kann, muss zuerst das System aufgesetzt und equilibriert werden. Erstellen Sie mit `xleap` eine pdb-Struktur eines Dialaninpeptids, wobei der N-Terminus acetyliert (ACE) und der C-Terminus methyliert (NME) ist. Wenden Sie dieses Mal keine Faltungswinkel an. Erstellen Sie daraus wie oben mit `pdb2gmx` eine `.gro`-Datei und eine Topologie, indem Sie als Kraftfeld wieder AMBER99SB-ILDN und TIP3P als Wassermodell auswählen.

Packen Sie nun die Startstruktur in eine Box:

```
editconf -f start.gro -d 1 -princ -o boxed
```

Wenn man sich die erhaltene Struktur `boxed.gro` in VMD anschaut

```
vmd boxed.gro
```

sollte man die erzeugte Box um das Petid sehen, sobald man in die VMD-Konsole `pbc box` eingibt.

Diese muss anschließend mit Wasser aufgefüllt werden:

```
gmx solvate -cp boxed.gro -cs -p topol.top -o solved.gro
```

Um ungünstige Konformationen und Wasserpositionen, die während der MD zu Fehlern oder Problemen führen könnten, zu vermeiden, führen Sie eine kurze Energieminimierung durch. Kopieren Sie dazu `em_steep.mdp` aus Kapitel II in Ihr aktuelles Verzeichnis und führen Sie die Minimierung mit `grompp` und `mdrun` durch.

Mithilfe dieser optimierten Struktur wird zuerst eine 10 ps NVT-Equilibrierung mit Position Restraints auf den schweren Atomen des Peptids durchgeführt. Benutzen Sie dazu `nvt.mdp` aus Kapitel II.

Die ungefähre Gleichverteilung der Solventmoleküle über die Box lässt sich mit

```
g_density -f nvt.xtc -n index -s nvt -o density
```

abschätzen.

Schließen Sie eine NPT-Simulation von 500 ps an. Ändern Sie entsprechend die Simulationslänge in `npt.mdp` aus Kapitel II.

Dadurch sollte das Bulkwasser ausreichend equilibriert sein. Dies kann anhand der zeitlichen Entwicklung der Dichte überprüft werden (`g_energy` und `xmgrace`).

B. Umbrella Sampling

Beim Umbrella Sampling nutzt man harmonische Potentiale, um Bereiche der Reaktionskoordinate (bei uns der Ψ-Winkel) zu simulieren, die in einer Standard-MD nicht erreicht werden würden. In unserem Fall interessiert uns der Bereich von $-180°$ bis $+180°$. Diesen Bereich unterteilen wir in 5°-Schritte und erhalten so 73 Fenster. Innerhalb jedes Fensters hat das harmonische Potential sein Minimum in diesem Fenster. Hiermit zwingen wir das System, in diesem Bereich zu bleiben. Die gewählte Kraftkonstante muss allerdings so sein, dass die bedeckten Bereiche der benachbarten Fenster überlappen (Näheres dazu in der Auswertung).

Kopieren Sie zur Vorbereitung `topol.top` nach `dialanin.itp`. Löschen Sie alles aus dieser Datei, was nicht direkt zur Moleküldefinition von Dialanin gehört, d.h. alles vor `[ molecule_type ]` und alles nach der Definition der Diederpotentiale. Die Fenster und die zugehörigen Potentiale werden durch das Skript `create.sh` erzeugt. Werfen Sie einen Blick hinein und vollziehen Sie nach, was passiert. Ändern Sie die Anzahl der Wassermoleküle unter `SOL` in die Anzahl, die in Ihrem `topol.top` steht. Wichtig ist, dass unter `[ dihedrals ]` ein zusätzliches Diederpotential eingeführt wird, um den Ψ-Winkel in diesen Bereich zu zwingen. Die neue Topologie finden Sie nach dem Ausführen in den `GRAD_*`-Ordnern unter `us.top`.

Ein weiteres Skript (`run_opt.sh`) startet nacheinander in jedem Fenster eine Optimierung. Verstärken Sie die Optimierung, indem Sie die Abbruchkriterien verschärfen: ändern Sie in `em_steep.mdp` die maximale Kraft auf 10 kJ/mol/nm und die Anzahl der Schritte auf 750.

Schauen Sie sich die optimierten Strukturen als Film in VMD an mit `./vmd.sh`. Sie müssten eine Drehung um die CA–C Bindung beobachten können.

Sie haben nun die Startstrukturen für jedes Fenster erstellt und können mit dem eigentlichen Umbrella-Sampling beginnen. Ändern Sie dazu in `npt.mdp` den Eintrag "nsteps" so, dass Sie eine Nansosekunde simulieren. Die Koordinaten sollen alle 100 Schritte herausgeschrieben werden. Starten Sie nun `./run_npt.sh`. Bis dieses Skript durchgelaufen ist, dauert es etwa einen Tag.

C. Auswertung

Nachdem die Rechnungen für alles Fenster beendet sind, muss überprüft werden, ob der gesamte Bereich der Reaktionskoordinate abgescannt wurde. Dafür muss für jede Simulation ein Histogramm des Ψ-Winkel angefertigt werden. Sollten die Histogramme der benachbarten Fenster ausreichend überlappen, wurde der Abstand der Fenster sowie die Kraftkonstante des Extrapotentials richtig gewählt. Für die Analyse muss zuerst der Ψ-Winkel entlang der Dynamik bestimmt werden. Erstellen Sie dazu zunächst eine Indexdatei `angle.ndx` in dem Verzeichnis, in dem die `GRAD_*`-Ordner liegen. Die Datei soll eine Indexgruppe mit den Indices der 4 Atome enthalten, die den Ψ-Winkel definieren. Innerhalb der `GRAD_*`-Ordner können Sie jetzt den Verlauf des Ψ-Winkels berechnen:

```
g_angle -f npt.xtc -n ../angle -ov psi -type dihedral
```

Danach muss aus dem zeitlichen Verlauf des Winkels ein Histogramm erstellt werden

```
g_analyze -f psi -dist -bw 0.5
```

welches man mit

```
xmgrace distr.xvg
```

anschauen kann. Nachdem man dieses in allen Fenster gemacht hat, kann man den Überlapp überprüfen.

```
xmgrace GRAD_*/distr.xvg
```

Wenn man nicht die eben beschriebenen Schritte 73 mal machen will, sollte man ein kleines Skript schreiben (orientieren Sie sich an den gegebenen Skripten, um sich das Leben einfacher zu machen). Um aus den einzelnen Rechnungen eine Freie Energie Kurve zu bekommen, kann man das Programm WHAM (Weighted Histogram Analysis Method) benutzen. Dieses Program benötigt für jedes Fenster eine Datei, in der der zeitliche Verlauf des Winkels gespeichert ist. Hierfür muss von der bereits erstellten Datei `psi.xvg` der Dateikopf gelöscht werden. Er besteht aus 15 Zeilen (überprüfen Sie dies an einer `psi.xvg` Datei!), die also zu entfernen sind. Dies kann per Hand erledigt werden, oder Sie erweitern Ihr Skript mit dem folgenden Befehl:

```
tail -n+16 psi.xvg > angle.txt
```

Desweiteren brauchen Sie eine Datei, die die Pfade der eben angelegten Dateien angibt und die zugehörigen Kraftkonstante und das Minimum des Extrapotentials, die in dem jeweiligen Fenster benutzt wurden (diese Datei ist bereits gegeben: `metafile`). Die Kommandoargumente für WHAM sind:

```
wham [P/Ppi/Pval] hist_min hist_max num_bins tol temperature numpad metadatafile freefile
```

- Das erste Argument gibt die Periodizität der Reaktionskoordinate an: `P` schaltet eine Periodizität von 360° ein; `Ppi` schaltet eine Periodizität von 180° ein; `Pval` spezifiziert eine Periodizität mit einem beliebigen Wert (z.B. `P90.0`).
- `hist_min` und `hist_max` bestimmen die Grenzen des Histogramms
- `num_bins` bestimmt die Anzahl der Bins
- `tol` gibt die Konvergenztoleranz an
- `numpad` ist nur bei aperiodischen und 2D-Rechnungen wichtig; sollte nicht null sein
- `metadatafile` ist in unserem Fall `metafile`
- `freefile` ist der Name der Outputdatei

Im Standardouput stehen die Iterationsschritte und die Maximaländerungen der Freien Energie Werte. Alle 100 Schritte wird das PMF ausgegeben, was etwa so aussieht:

```
-177.500000   2.994034   0.564643
-172.500000   3.895723   0.393350
-167.500000   4.923043   0.260562
-162.500000   5.917272   0.174905
-157.500000   6.888434   0.118498
-152.500000   7.776316   0.083008
-147.500000   8.773800   0.055648
-142.500000   9.650452   0.039157
```

Die erste Spalte zeigt die Reaktionskoordinate, die zweite die enstprechende Freie Energie (in kJ/mol) und die dritte Spalte gibt die nicht normalisierte Wahrscheinlickeitsverteilung an, welche uns dieses Mal nicht interessieren soll. Das finale PMF sollte in eine andere Datei (z.B. `delta_g.txt`) kopiert werden und mit `xmgrace` angeschaut werden.

- Überprüfen sie, ob das berechnete Freie Energieminimum mit der klassischen MD übereinstimmt, indem Sie für die Standard NPT-Simulation ein Histogramm des Ψ-Winkels anfertigen. Wie verhält sich der Winkel während der Simulation? Ist er die ganze Zeit im globalen Minimum oder wird die Barriere überwunden?
- Scheint die berechnete Rotationsbarriere sinnvoll?
- Vergleichen Sie die erhaltenen Minima mit den erwarteten Winkeln für eine α-Helix und ein β-Blatt.

V. LIGAND DOCKING RECHNUNGEN MIT AUTODOCK

A. Ligand Docking

MD-Simulationen sind nicht die einzige Option um biomolekulare Wechselwirkungen zu studieren. Trotz aller Näherungen in den Kraftfeldern und stetig steigender Computerleistung ist eine sehr wichtige Frage der Biochemie bis heute nur schwer simulierbar: Das sogennante Dockingproblem, die Frage in welcher Konformation und mit welcher Bindungsstärke kleine Moleküle an makromolekulare (Protein-)Rezeptoren andocken.

Diese Frage ist zentral weil es eine der wichtigsten Funktionen von Proteinen ist, chemische Verbindungen zu erkennen, zu binden und entweder chemische Reaktionen zu katalysieren (Enzyme) oder die Bindung des Liganden in ein Signal umzuwandeln (Sensorproteine). Ligandenbindung ist nicht nur für die Grundlagenforschung zur Funktion von Zellen wichtig, sondern auch eines der Felder, in denen Computermodellierung praktische Anwendungen findet.

Sogennantes *in silico* Drug Design ist ein Teilbereich der Pharmaforschung, in dem es darum geht, die Bindungsstärke, -geometrie und Spezifizität von Testsubstanzen an Zielproteine vorherzusagen. Solche Zielproteine, die üblicherweise mit einer oder mehreren Krankheiten in Verbindung gebracht werden, zu hemmen oder zu aktivieren ist das Ziel der Wirkstoffentwicklung. Es gibt viele hundert solcher Zielproteine für die die dreidimensionale Struktur des Proteins aus z.B. Röntgenkristallstrukturanalysen bekannt ist. Solche Proteine in Massen herzustellen, aufzureinigen und die Aktivität vieler Testsubstanzen (typische industrielle Verbindungs-Datenbanken enthalten zehntausende von Stoffen) daran zu überprüfen ist extrem aufwändig und teuer, ausserdem müssen viele neue Testsubstanzen speziell synthetisiert werden.[1]

Ligand-Docking-Rechnungen arbeiten als eine Art Filter für diesen Prozess, für den man mit einfachen und möglichst schnellen Rechenmethoden versucht, eine qualitative Bindungsstärke zu berechnen, um dann die kostspieligen Experimente nur für die vielversprechendsten Kandidaten durchführen zu müssen. Selbst die effizientesten Moleküldynamik-basierten Methoden zur Berechnung Freie Energien sind hierfür bei weitem zu anspruchsvoll. Idealerweise soll für ein Rezeptor-Ligand-Paar ein Ergebnis in Minuten vorliegen.

Ligand-Docking-Programme beinhalten solche Algorithmen. Es gibt eine Vielzahl unterschiedlicher Geometriesuchfunktionen und Wege die Bindungsenergie abzuschätzen (Scoring-Funktionen), und mehrere populäre Programme sind weit verbreitet. Dazu gehören die Freien Programme Autodock oder Dock, genauso wie kommerzielle Varianten wie FlexX, Glide oder Gold. Die meisten aktuell entwickelten Programme sind von vergleichbarer Leistungsfähigkeit und typischerweise in der Lage, in günstigen Fällen bindende von nicht-bindenden Verbindungen zu unterscheiden. Allgemein gelöst ist das Dockingproblem aber heutzutage noch lange nicht.

Im folgenden soll das Medikament Indinavir mit Hilfe des Programms Autodock an sein Zielprotein, eine Protease aus dem HIV-Virus, gedockt werden. Für diesen Versuch wird angenommen, dass Grundkenntnisse in Linux und VMD vorhanden sind. Eine Installation von Autodock und der graphischen Oberfläche Autodock-Tools sollte auf den Praktikumsrechnern vorhanden sein.

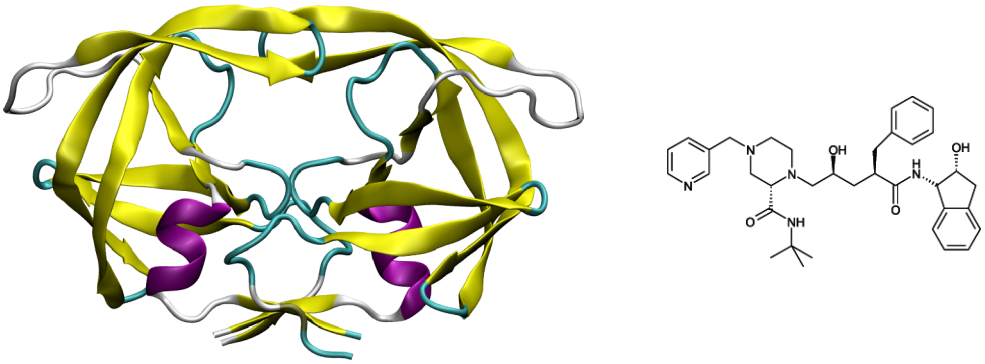

FIG. 3: Links: Die Struktur der HIV Protease I. Rechts: Indinavir, ein zugelassener Wirkstoff zur Hemmung der HIV-I Protease.

Dieser Versuch basiert inhaltlich auf dem Tutorial “Using AutoDock 4 with AutoDockTools” von Ruth Huey und Garrett M. Morris von The Scripps Research Institute, La Jolla, CA.

---

[1] Das ist einer der Gründe dafür, dass moderne Medikamentenentwicklung ein Prozess ist, der Jahre dauern und mehrere Milliarden kosten kann.

B. Visualisierung von gedockten Strukturen

Beginnen sie damit, ein Verzeichnis `docking` anzulegen und die Dateien für diesen Versuch dorthin zu speichern. Wechseln Sie in dieses Verzeichnis und starten Sie die graphische Oberfläche ADT:[2]

```
adt
```

Den Pfad müssen Sie entweder zu Ihrer Pfadvariable hinzufügen oder entsprechend eingeben.) Wählen Sie *Autodock 4.2* im kleinen Auswahlfenster. Das *ADTools-Fenster*, das sich geöffnet hat, wird verwendet, um alle Rechnungen vorzubereiten und auszuwerten.

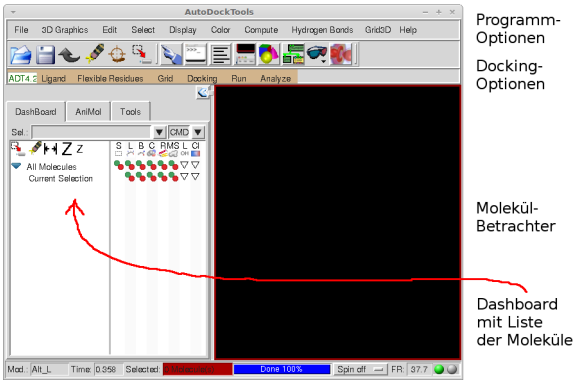

FIG. 4: Das Hauptfenster von ADT, in dem die meisten Analysen vorgenommen werden können.

Bevor wir eine neue Rechnung anfangen, sollen zunächst die Ergebnisse einer typischen Dockingrechnung visualisiert werden. Öffnen Sie dazu zunächst die Datei `ind4.dlg` in einem Texteditor. Diese enthält die gesamte Ausgabe einer Dockingrechnung. Dazu gehören Informationen über den verwendeten Algorithmus, Parametereinstellungen, Analyse des Liganden und am Ende der Datei die Strukturinformationen der vorhergesagten Positionen und Bindungsstärken der gedockten Liganden.

Übersichtlicher ist es, die Ergebnisse graphisch anzeigen zu lassen. In ADT, wählen Sie *Analyze – Dockings – Open* und wählen Sie dieselbe Datei. Eine Meldung gibt an, wieviele individuelle Ligandenplatzierung gefunden wurden. Mit *Analyze – Conformations – Play* lassen sich die einzelnen Konformationen betrachten (auch wenn dies ohne zugehörige Rezeptorstruktur noch nicht sehr hilfreich ist). Eine bessere Übersicht der berechneten Platzierungen und ihrer Energien lässt sich mit *Analyze – Conformations – Load* öffnen. Im Beispiel sind zwei Cluster von Platzierungen gefunden worden, der erste mit acht, der zweite mit zwei individuellen Bindungsgeometrien. Klicken Sie auf einzelne Lösungen, um weitere Informationen über die Resultate zu erhalten und den dargestellten Liganden in die gewählte Konformation zu drehen.

Mehr über die Verteilung auf die einzelnen Cluster lässt sich mit *Analyze – Clusterings – Show* erhalten. Betrachten wir nun die Lage der Dockings im Rezeptor. Laden Sie dazu das zugehörige Protein mit *Analyze – Macromolecule – Open* und laden sie `hsg1.pdbqt`. Überprüfen Sie, ob der Ligand wirklich in einer Bindungstasche des Rezeptors zu liegen kommt. Mit *Analyze – Conformations – Load* können Sie wieder zwischen den zehn vorhergesagten Strukturen wechseln. An diesem Punkt könnten sich nun viele verschiedene Analysen anschließen. Möglicherweise ist dieses Dockingergebnis bereits ausreichend, eventuell kann man allerdings die Rechnung mit anderen Parametern wiederholen oder die spezifischen Wechselwirkungen mit der Bindungstasche bestimmen um bessere Liganden vorherzusagen.

Alternativ könnte man die gefundenen Komplexgeometrien als Ausgangspunkt für andere Rechnungen wie MD-Simulationen verwenden. Die genauen Ziele einer solchen Analyse hängen stark vom betrachteten System ab, im folgenden wollen wir aber stattdessen betrachten, wie man eine typische Dockingrechnung durchführt, um solche Protein-Ligand-Komplexe vorherzusagen.

---

[2] Autodock lässt sich auch komplett von der Konsole aus bedienen, das graphische Interface ist in diesem Fall aber komfortabler.

C. Vorbereiten eines Rezeptors

Der erste Vorbereitungsschritt ist die Präparation der Rezeptorbindungstasche. Da üblicherweise viele Liganden an einen Rezeptor gedockt werden, kann die Vorbereitung hier etwas aufwändiger sein, aber dafür müssen diese Schritte nur einmal vor dem eigentlichen Docking durchgeführt werden. In diesem Schritt wird eine bekannte dreidimensionale Struktur eines Ziel-Proteins in die, für eine Autodock Rechnung benötigte, `.pdbqt` Datei umgewandelt.

Für dieses Beispiel wird die Rezeptorstruktur aus der Röntgenkristallstruktur der HIV-Protease I im Komplex mit Indinavir verwendet, die in der PDB Datenbank unter dem Kürzel 1HSG abgelegt ist.[3] (Sie können diese Strukturdatei auch selbst unter `www.rcsb.org` herunterladen). Öffnen Sie die Datei `1HSG.pdb` mit *File – Read Molecule*; alternativ kann die pdb-Datei auch zunächst mit VMD oder einem Editor betrachtet werden.

Die Kristallstruktur enthält noch kokristallisierte Wassermoleküle und den Liganden, die wir beide noch entfernen werden, um die Dockingrechnung mit der leeren Bindungstasche durchzuführen. Dazu wählen wir die zu entfernenden Moleküle zunächst aus: *Select – Select From String*. In dem sich öffnenden Fenster wählen sie als erstes unter *Residue* `HOH*` und unter *Atom* `*` und addieren diese, dann unter *Residue* `MK1*` und unter *Atom* `*` und addieren diese. Im Betrachtungsfenster sind der Ligand und die Wassermoleküle in gelb markiert. Was für zu entfernende Moleküle sich in der Kristallstruktur befinden und wie deren Restenamen gewählt sind, ist von Fall zu Fall unterschiedlich und lässt sich durch Betrachten der Struktur klären. Mit *Edit – Delete – Delete Atomset* lassen sich die ausgewählten Atome (unumkehrbar) löschen.

Jetzt haben wir die Proteinstruktur des Rezeptors, allerdings, wie in Röntgenkristallstrukturen üblich, noch ohne Wasserstoffatompositionen. Diese lassen sich in ADT durch *Edit – Hydrogens – Add* hinzufügen. Als letztes wird der Rezeptor in eine `.pdbqt` Datei umgewandelt und dabei auch mit Partialladungen versehen. Wählen Sie *Grid – Macromolecule – Choose* und wählen sie `1HSG` (vermutlich die einzige Wahl, wenn keine anderen Strukturdateien geöffnet sind). In dem Speicherdialog wählen Sie 1HSG.pdbqt als Dateiname. Wenn Sie diese Datei mit einem Texteditor öffnen, sehen Sie, dass das Format ähnlich einer `.pdb` Datei ist, die noch Partialladungen beinhaltet und dass ADT nur die polaren Wasserstoffatome berücksichtigt hat.

Bevor wir die Vorbereitung des Rezeptors mit der Erstellung der Potentialdateien abschließen, müssen wir zuerst noch den Liganden vorbereiten. Wählen Sie *Hide* (im Dashboard am unteren Rand des ADT Fensters) für die Darstellung von 1HSG um das Display zu leeren.

D. Vorbereiten eines Liganden

Für Liganden, ähnlich wie für Rezeptoren, müssen Topologiedateien, Atomtypen und Partialladungen festgelegt werden.[4] Darüber hinaus muss festgelegt werden, welche Diederwinkel des Liganden als flexibel betrachtet werden sollen. Im Gegensatz dazu wird die Rezeptorstruktur in dieser Übung als rigide angesehen.

Beginnen Sie damit, eine PDB Datei von Indinavir zu laden: *Ligand – Input – Open*. Dabei muss noch PDB als Dateiformat gewählt werden, bevor man die Datei `ind.pdb` öffnen kann. Im Anzeigefenster erscheint eine Nachricht, dass ADT Gasteiger-Ladungen zugewiesen hat und nicht-polare Wasserstoffe entfernt wurden.

Als nächstes werden die frei rotierbaren Bindungen ausgewählt: *Ligand – Torsion Tree – Choose Root* und *Ligand – Torsion Tree – Choose Torsions*. Die Auswahlkriterien, welche Bindungen rotierbar sind, können wie gegeben akzeptiert werden. Alternativ, testen Sie, wie sich die Bindungsauswahl ändert, wenn Rotationen um Amidbindungen erlaubt werden. Das root-Atom, das zuvor gewählt wird, hat mit der internen Aufteilung des Liganden in eine Baumstruktur zu tun, was nur die Recheneffizienz beeinträchtigt.

Um einen starreren Liganden zu simulieren, lassen sich unter *Ligand – Torsion Tree – Set Number of Torsions* noch die maximale Zahl an flexiblen Bindungen festlegen. Hier soll dieser Wert aber auf dem Maximum von 14 (oder 16 mit Amidbindungen) bleiben. Speichern Sie den Liganden mit *Ligand – Output – Save as PDBQT* unter dem Namen `ind.pdbqt`. Für eine realistischere Studie mit hunderten von Liganden lassen sich diese Schritte in einem Script automatisieren, außerdem gibt es verschiedene Möglichkeiten chemische Strukturformeln automatisch direkt in 3D-Strukturen zu überführen. Für wenige Liganden ist eine Vorbereitung per Hand aber ausreichend.

---

[3] Das ist eine etwas vereinfachte Vorgehensweise, da der Rezeptor ja schon in der optimalen Konformation vorliegt, um Indinavir zu binden, und wir aus der Struktur ableiten können, wo die Bindungstasche liegt. Wir führen also ein Re-Docking eines bekannten Liganden durch, da diese Schritte nur einmal vor den eigentlichen Dockings durchgeführt werden müssen. anstatt eines Blind-Dockings mit einem neuen Liganden. Die Vorgehensweise ist die gleiche, nur dass für tatsächliche Blind-Dockings eine gründlichere Vorbereitung und kritische Analyse notwendig sind.

[4] Also im Prinzip dieselben Schritte, die auch zur Vorbereitung einer MD-Simulation nötig wären, nur jeweils auf einem einfacheren, und schnelleren, Modellierungslevel.

E. Erstellen der Rezeptorpotentiale

Autodock verwendet die Rezeptorstruktur nicht nur direkt als atomistisches Modell, sondern berechnet daraus noch Potentialverteilungen für verschiedene Wechselwirkungstypen (wie van-der-Waals oder elektrostatische WW).

Diese Potentiale muss man zwar nur einmal pro Rezeptor berechnen, allerdings muss dazu bereits bekannt sein, welche Atomtypen im Liganden vorkommen. Am einfachsten geht dies, wenn der Ligand bereits in ADT geladen ist. Wählen Sie *Grid – Set Map Types – Choose Ligand* und wählen sie `ind` für den Indinavir Liganden. Autodock verwendet ein Grid-Modell um diese Potentiale festzulegen; sie werden für eine Reihe von Punkten berechnet die ein kubisches Gitter bilden und bei Bedarf interpoliert.

Jetzt muss festgelegt werden, welcher Teil des Rezeptors als Bindungstasche in Frage kommt. Falls man die Lage der Bindungstasche nicht kennt, kann man die Box auch so wählen, dass sie den gesamten Rezeptor einschließt, dies führt aber zu sehr langsamen Rechnungen. Für dieses Beispiel wählen wir 60, 60 und 66 als Boxdimensionen, mit einer Gitterpunktdistanz von 0.375. Als Zentrum der Box werden 16.0, 24.0 und 4.0 gewählt. Überprüfen Sie, ob die Box die leere Bindungskavität des Enzyms beinhaltet, und passen Sie die Lage oder Größe der Box entsprechend an. Schliessen Sie das Grid-Fenster mit *File – Close Saving Current*, ansonsten bleibt Ihre Auswahl nicht erhalten. Speichern Sie die Boxdefinition mit *Grid – Output – Save GPF* und speichern Sie unter `1hsg.gpf`. Mit *Grid – Edit GPF* können Sie die erstellte Datei überprüfen.

Dieses Grid Parameter File wird nun mit autogrid zur Berechnung der Parameterdateien verwendet. Wechseln Sie dazu zur Konsole in das Verzeichnis, das alle bisher erstellten Dateien enthält. Starten Sie die Potentialberechnung mit

```
autogrid4 -p GPF-Dateiname -l Output-Dateiname &
```

(passen Sie Pfade und Dateinamen entsprechend an) und warten Sie das Ende der Berechnungen ab, was einige Minuten dauern kann. Bei erfolgreichem Verlauf, lesen Sie die erstellte Logdatei und stellen Sie sicher, dass eine Reihe von `.map` Dateien erstellt wurden (eine für jeden Atomtyp des Liganden, eine elektrostatische, etc.). Damit liegt alles vor, was für eine Dockingrechnung benötigt wird.

F. Durchführen der Dockingrechnung

Jetzt werden die gerade erstellten Rezeptor- und Ligandendateien für eine Dockingrechnung ausgewählt. Selektieren Sie unter *Docking – Macromolecule – Set Rigid Filename* die von Ihnen erstellte Datei `1HSG.pdbqt` und dann unter *Docking – Ligand – Choose* den `ind` Liganden, für den nochmal eine Zusammenfassung angezeigt wird. Die Parameter der Rechnung werden gewählt unter *Docking – Search Parameters – Genetic Algorithm* um einen von Autodocks Docking-Algorithmen festzulegen. Im folgenden Fenster lassen sich Details der Rechnung festlegen Hier wählen wir unrealistisch kleine Werte um die Übung zu beschleunigen. Setzen Sie *Number of GA Runs* auf 20, *Population Size* auf 100 und die Zahl der Energieevaluationen auf *short* (250,000). Die restlichen Parameter ändern die Feineinstellungen des genetischen Algorithmus und können auf Ihren Defaultwerten belassen werden. Speichern Sie die Parameterwahl mit *Docking – Output – Lamarckian GA* z.B. unter `1hsg-ind.dpf`. Um dieses Docking Parameter File zu betrachten, wählen Sie *Docking – Edit DPF*.

Um die eigentliche Rechnung zu starten, wechseln Sie wieder zur Konsole und starten Sie das Autodock Hauptprogramm mit

```
autodock4 -p 1hsg-ind.dpf -l 1hsg-ind.dlg
```

Dies kann wiederum ca. 1 Tasse Kaffee lang dauern. Nach Abschluss der Rechnung sollte eine Docking-Logdatei vorliegen, die die berechneten Ligandenplatzerungen enthält.

G. Analyse

Öffnen Sie ihre fertige Dockingrechnung und visualisieren Sie die Ergebnisse, so wie oben mit *Analyze – Dockings – Open*. Speichern Sie den besten Dockingvorschlag als pdb-Datei.[5] Überprüfen Sie:

- Wieviele der vorgeschlagenen Ligandenpositionen liegen innerhalb der Bindungstasche? Ist dies überraschend?
- Wieviele Cluster von Platzierungen wurden gefunden? Wurden genug Dockings für ein gutes Clustering erzeugt?
- Wie unterschiedlich sind die gefundenen Platzierungen bzw. deren Energien?
- Wie groß ist die beste vorhergesagte Freie Bindungsenthalpie? Ist dies typisch für pharmazeutische Inhibitoren?
- Wie sehr unterscheidet sich Ihre vorgeschlagene Platzierung von der bekannten aus der Röntgenkristallstruktur?
- Welche wichtigen Wechselwirkungen zwischen Ligand und Bindungstasche lassen sich erkennen?

[5] Am einfachsten ist es dazu, die dazugehörigen `ATOM` Zeilen aus der `.dlg` Datei auszuschneiden. Autodocks Koordinatenexportfunktion produziert leider PDB Dateien, die mit VMD oder anderen Modelingprogrammen nicht kompatibel sind.

VI. ELEKTROSTATIK-BERECHNUNGEN

A. Elektrostatisches Potential auf der Molekülоberfläche

Im folgenden sollen implizite Solvensmodelle verwendet werden, um Freie Solvatationsenthalpien oder elektrostatische Oberflächenpotentiale zu berechnen. Das könnte zwar auch mit voll atomistischen Modellen getan werden; die Notwendigkeit, dann über alle möglichen relevanten Lösungsmittelkonfigurationen zu mitteln, bedingt aber extrem lange Simulationen.

Implizite Solvensmodelle ersetzen die Wassermoleküle in einer Simulationsbox mit aufwändigeren Potentialfunktionen, die die Umgebung eines Moleküls sich so verhalten lassen, als ob es in eine perfekt equilibrierte Wasserhülle eingebettet wäre. Für MD-Simulationen sind hier sog. Generalized-Born-Modelle beliebt, für statische Strukturen werden üblicherweise Kontinuumsdielektrikmodelle verwendet. Diese basieren auf der Poisson-Boltzmann Gleichung, einer physiko-chemischen Erweiterung der Poisson-Gleichung aus der Elektrostatik. Dabei wird für eine Molekülstruktur die Energie (und Kräfte, Potentiale, etc.) so berechnet, als ob das Molekül aus einem nichtleitenden Material mit niedriger Dielektrizitätskonstante bestünde, in das Punktladungen eingebettet sind. Dieses Molekülmodell ist dann in ein Dielektrikum hoher Dielektrizitätskonstante eingebettet. Dieses Modell ist zwar nur bedingt realistisch (Dielektrizitätskonstanten sind eigentlich makroskopische Größen ohne molekulare Entsprechungen, Partialladungen sind grundsätzlich suspekt…), aber in der Lage mit guter empirischer Parameterisierung, die elektrostatischen (insbesondere Solvatation-)Eigenschaften von Molekülen genähert zu berechnen.

Für eine Beschreibung des typischerweise verwendeten Gittermodells, lesen Sie zur Vorbereitung eventuell auch `PBS.pdf`. Wir benutzen das frei erhältliche Programm APBS (`http://www.poissonboltzmann.org/apbs`).

Für die Berechnung der Molekülоberfläche brauchen wir jetzt die Atomradien, und für die Elektrostatikberechnung die Ladungen. Sie finden sie in der Datei `AMBER.DAT` im Verzeichnis `/usr/local/run/pdb2pqr-linux-bin64-2.0.0/dat`. Das Format spezifiziert dabei:

```
<RESTNAME> <ATOMNAME> <LADUNG in e> <RADIUS in A>
```

`pdb2pqr` wandelt mithilfe dieser Datei eine `.pdb` Datei in eine `.pqr` Datei um, die zusätzlich zu den Koordinaten noch die Partialladung und den Radius eines Atoms enthält.

Führen Sie nun `pdb2pqr` aus:

```
pdb2pqr --assign-only --ff=AMBER k2.pdb k2.pqr
```

Öffnen Sie die Outputdatei. An Warnings sehen Sie, ob etwas nicht stimmt. Am Anfang des Moleküls wird die Gesamtladung angezeigt. Wenn sie +5 beträgt und keine Warnungen auftreten, wurden alle Ladungen richtig erkannt.

Lassen Sie sich nun in VMD die Ladungsverteilung anzeigen, indem Sie `k2.pqr` mit VMD öffnen. Unter *Extensions – Analysis – APBS Electrostatics* finden Sie die Schnittstelle zu APBS. Schauen Sie sich unter *edit* die Einstellungen an, wobei für diese Berechnung die Werte unverändert gelassen werden können. Klicken Sie *OK* und dann *Run APBS*. Klicken Sie beim darauffolgenden Dialog *OK* und schließen Sie das Fenster *APBS Tool*. Erstellen Sie nun unter *Graphical Representations* einen neue Repräsentation mit *Coloring = Volume* und *Drawing = Surf*. Gehen Sie auf den Trajectory-Reiter und setzen Sie die Farbskala auf −10 bis 10. Sie können die Visualisierung noch anschaulicher gestalten, indem Sie als 'Material' *Transparent* wählen und eine weitere Repräsentation in *Licorice* erstellen.

Wenn Sie sich nur diese Darstellung ansehen, was würden Sie vorhersagen, wie sich das Peptid in eine Membran einbettet? Entspricht das Ihren Erfahrungen und/oder Berechnungen?

B. Freie Solvatationsenergie

Wir nutzen APBS jetzt direkt anstatt über VMD, um die elektrostatische Komponente der freien Solvatationsenergie für K2 zu berechnen. Benutzen Sie dafür `apbs.in` und ersetzen Sie die in Großbuchstaben geschriebenen Worte durch die richtigen Werte. Speichern Sie und starten Sie die Berechnung mit

```
apbs apbs.in
```

Wie lautet das Ergebnis der freien Solvatationsenergie?

C. Vergleich mit einer Mutante

Führen Sie die letzten zwei Kapitel nochmal mit einem andered Peptid aus, bei dem im Vergleich zu K2 zwei Lysine durch Leucine getauscht wurden. Vergleichen Sie die Ergebnisse.

VII. EXTENDED SAMPLING METHODEN

In vielen Situationen ist es viel zu ineffizient, Konfigurationen des Molekülsystems mittels einer gewöhnlichen MD Simulation zu generieren. Das 'Abtasten' des Konfigurationsraumes würde einfach viel zu lange dauern – man müsste viel zu lagen simulieren, um alle relevanten Konfigurationen (z.B. Konformationen eines Peptids) mit den richtigen Wahrscheinlichkeiten in der Trajektorie zu sehen. Um diesen schwerwiegenden Nachteil umzugehen, wurden sog. Extended Sampling Methoden entwickelt.

A. Vorbereitung des Systems – ein Dipeptid

Die Vorbereitung des zu simulierenden Systems wird für beide Methoden gemeinsam sein. Wir werden ein sehr einfaches System untersuchen – ein Dipeptid in wässriger Lösung.[6] Um die Übung interessanter zu machen, möge jede/r Teilnehmer/in eine andere Aminsäure simulieren, so dass jede/r ein anderes Ergebnis bekommen wird. Am Ende werden die Ergebnisse verglichen und diskutiert.

Das Dipeptid kann im `xleap` gebaut werden, mit folgenden Kommandos, wobei das Alanin (`ALA`) mit einer anderen Aminosäure ersetzt werden kann:

```
source leaprc.ff14SB
pep = sequence { ACE ALA NME }
savepdb pep pep.pdb
quit
```

Die resultierende PDB Datei kann anschließend mit Programmen aus dem Gromacs Paket bearbeitet werden, um die Topologie zu bekommen und das System solvatisieren.

Die Vorbereitung des Systems folgt der Übung I D und II, und wird hier nur kurz zusammengefasst:

1. Topologie erstellen.

   ```
   pdb2gmx -f pep.pdb -ignh -o pep.gro
   ```

2. Kubische Box erstellen und solvatisieren.

   ```
   editconf -f pep.gro -o pep.edit -c -d 1 -bt cubic
   gmx solvate -cp pep.edit -o pep.box -cs spc216 -p topol
   ```

3. Falls eine geladene Aminosäure gewählt wurde – System neutralisieren.

   ```
   echo > dummy.mdp
   grompp -f dummy -p topol -c pep.box
   genion -s dummy -o pep.ion -p topol -neutral
   ```

   Bei `genion` muss die Wasser enthältende Gruppe explizit gewählt werden.

Damit ist das zu simulierende System gebaut und die Topologiedatei inklusive Wasser und ggf. Gegenion erstellt.

Weiter muss das System equilibriert werden. Das Verfahren folgt wieder Übung II und besteht aus folgenden Schritten:

---

[6] Mit 'Dipeptid' verstehen wir einen Aminosäurenrest, der mit den Schutzgruppen CH_3CO– (Acetyl) und –$NHCH_3$ (Methylamino) ein vollständiges Molekül darstellt. So ein Dipeptid weist ein komplettes Paar von Winkeln $\varphi - \psi$ auf, und es ist daher sinnvoll sich mit einem Ramachandrandiagram zu befassen.

1. Kurz energieminimieren. Falls ohne Gegenion simuliert wird (ungeladene Aminosäure), soll `pep.box` anstatt `pep.ion` genommen werden.

   ```
   grompp -f steep -c pep.ion -p topol -o steep
   mdrun -deffnm steep
   ```

2. Auf 300 K aufheizen.

   ```
   grompp -f heat -c steep -p topol -o heat
   mdrun -deffnm heat
   ```

3. Barostat einschalten und Dichte equilibrieren.

   ```
   grompp -f equil -c heat -p topol -o equil
   mdrun -deffnm equil
   ```

B. Metadynamik

1. Intro

In einer Metadynamik Simulation wird zu der Potentialenergiefunktion des Molekülsystems (also zu dem üblichen Kraftfeld) eine zusätzliche, 'künstliche' Energiefunktion addiert (biasing potential). Diese dient dazu, das Molekül aus dem Energieminimum zu bringen, in dem es sich gerade befindet. Die biasing potential Funktion hat die Form einer Summe von vielen Gaussfunktionen, die erst während der Simulation eine nach der anderen addiert werden. Die Gaussfunktionen nehmen als Argument eine oder einige (einfache oder komplizierte) Funktion(en) der Atomkoordinaten; diese wird/werden Collective Variables (CV) oder Reaktionskoordinate genannt. Beispiele von CV: Diederwinkel φ und ψ in einem Peptid; Abstand eines Liganden von der Mitte der Bindungstasche; Gyrationsradius eines Proteinmoleküls.

Eine wichtige Eigenschaft der Metadynamik Methode ist, dass die Summe der Gaussfunktionen zu der negativen freien Energie (ΔF bzw. ΔG) konvergieren, so lange die Simulation länger wird. Die kritische Voraussetzung für Erfolg ist hier die angemessene Wahl der Collective Variables. Eine passende, natürliche Wahl der Reaktionskoordinaten für ein Dipeptid sind die Diederwinkel φ und ψ, so dass man als Ergebnis die freie Energie in Form eines Ramachandrandiagrams bekommen würde.

Das Gromacs Paket an sich kann leider keine Metadynamik Simulation durchführen. Allerdings wurde ein Hilfsprogramm (Plugin) names Plumed entwickelt, das die erwünschte Funktionalität ergänzt. Praktisch sieht es so aus, dass die Parameter der Metadynamik-Methode durch eine zusätzliche Input-Datei übermittelt werden; wir benutzen hier die Datei `wt-metad.dat`, die folgende Daten enthalten soll:[7]

```
phi: TORSION ATOMS=5,7,9,15
psi: TORSION ATOMS=7,9,15,17
METAD ...
  LABEL=metad
  ARG=phi,psi
  PACE=100
  HEIGHT=0.625
  SIGMA=0.349,0.349
  GRID_MIN=-pi,-pi
  GRID_MAX=pi,pi
  FILE=HILLS
  BIASFACTOR=7.
  TEMP=300.
... METAD
PRINT STRIDE=100 ARG=phi,psi,metad.bias FILE=plumed.xvg
```

[7] Zeilen, die mit `#` beginnen, sind Kommentare und werden vom Programm nicht berücksichtigt.

Hier werden also zuerst die Diederwinkel φ und ψ durch Atomnummern der Atome C, N, Cα, C und N definiert. Dann wird schon die Metadynamik aktiviert: Es werden zusätzliche Gaussiane als Funktionen von φ und ψ addiert, und zwar in jedem hundertsten Schritt der MD, und die Höhe v und Breite σ der Gaussiane

$$w \cdot \exp\left[-\frac{(\varphi(t)-\varphi^*)^2+(\psi(t)-\psi^*)^2}{2\sigma^2}\right]$$

betragen 0.625 kJ/mol und 0.349 rad = 20°. Die letzteren beiden Optionen bestellen die sog. well-tempered Variante der Metadynamik: Hier werden die zusätzlichen Gaussiane mit der Zeit kleiner und kleiner (w wird automatisch reduziert), damit eine bessere Konvergenz der freien Energie gewährleistet wird. Der genaue Wert des Biasfaktors soll dem zu simulierenden System angepasst werden, und es kann eine gute Idee sein, mehrere Werte auszuprobieren. Eine Faustregel empfiehlt allerdings einen Wert von der Hälfte der ewarteten Energiebarriere in kT-Einheiten (1 kT beträgt 2.5 kJ/mol bei 300 K). Die letzte Zeile bestimmt, welche Daten während der Simulation ausgeschrieben werden sollen.

2. Durchführung

Zuerst soll eine Gromacs Simulation vorbereitet werden, genau gleich wie für eine übliche MD Simulation:

```
grompp -f md -c equil -p topol -o md
```

Danach kann schon die eigentliche Metadynamik-Simulation durchgeführt werden:

```
mdrun -deffnm md -plumed wt-metad -v
```

Hier wäre es gut, die Anzahl der Schritte in `md.mdp` so zu wählen, dass die Simulation über Nacht laufen wird und am nächsten Tag früh morgens fertig ist.

3. Analyse

Zur Analyse wird die Datei `HILLS` gebraucht, wo die Position sowie Höhe der Gaussiane ausgeschrieben wird. Wie oben erwähnt, enspricht die Summe aller Gaussiane einem Spiegelbild der freien Energie als Funktion der verwendeten Collective Variables. Hier bekommt man die freie Energie also als Funktion der Diederwinkel $\varphi - \psi$ (Ramachandrandiagramm), und zwar mit dem folgenden Befehl:

```
plumed sum_hills --bin 180,180 --min -pi,-pi --max pi,pi --hills HILLS
```

Die resultierende Landschaft der freien Energie, die in die Datei `fes.dat` ausgeschrieben wurde, kann man am besten grafisch darstellen. Ein Bild im PNG-Format (`fes-kcal.png`) wird mit dem gelieferten Skript erzeugt, das einfach mit dem Befehl `./fes-dat-to-png.sh` ausgeführt wird. Die freie Energie in kcal/mol ist farbkodiert.

Welche Konformationen sehen Sie in dem Ramachandrandiagramm? Vergleichen Sie die Diagramme für die verschiedenen Aminosäuren.

C. Hamiltonian Replica Exchange

1. Intro

Diese Methode, auch kurz 'Hrex' genannt, dient dem gleich Zweck – also alle relevanten Strukturen des Molekülsystems innerhalb kürzerer Simulationszeit zu erhalten – aber der Ansatz ist ganz unterschiedlich. Hier wird das Molekülsystem mehrfach simuliert – es laufen also mehrere Simulationen von dem gleichen Molekül gleichzeitig; diese Simulationen heißen Replikas. Die Besonderheiten von Hrex sind:

- In jeder Replika wird ein unterschiedliches Kraftfeld verwendet: die Kraftfeldterme (nichtbindende sowie Diederwinkeln) für ein Teil des Systems werden mit einem Faktor f skaliert, $0 < f \leq 1$. Für jede Replika wird ein unterschiedlicher Faktor verwendet, wobei eine Replika mit dem Faktor von 1 simuliert wird – hier wird also das Molekülsystem ungestört simuliert.

- Während der Simulation wird regelmäßig (z.B. nach jedem 1000. Schritt) versucht, die Koordinaten + Geschwindigkeiten zwischen den Replikas auszutauschen (daher 'replica exchange'). Es wird zuerst die Wahrscheinlichkeit für den Austausch von Replikas i und j bestimmt

$$\mathcal{P} = \exp\left[\frac{-V_i(r_j) + V_i(r_i) - V_j(r_i) + V_j(r_j)}{kT}\right]$$

wobei V_i und V_j die Kraftfeldfunktionen in Simulationen i und j sind, und r_i und r_j die aktuellen Koordinaten der entsprechenden Replikas. Dann wird eine Zufallszahl $0 < \rho < 1$ generiert, und falls $\mathcal{P} > \rho$, werden die Koordinaten + Geschwindigkeiten der Replikas i und j ausgetauscht. Mit diesem Verfahren wird die richtige, kanonische Wahrscheinlichkeitsverteilung der Konformationen gewährleistet. Dabei wird ermöglicht, dass die einzelnen Koordinatensätze zwischen den Replikas wandern (je nach aktueller Potentialenergien), und so über einen Umweg durch die Simulationen mit herunterskalierten Energien auch höhere Energiebarrieren überwinden. Siehe also Bild 5.

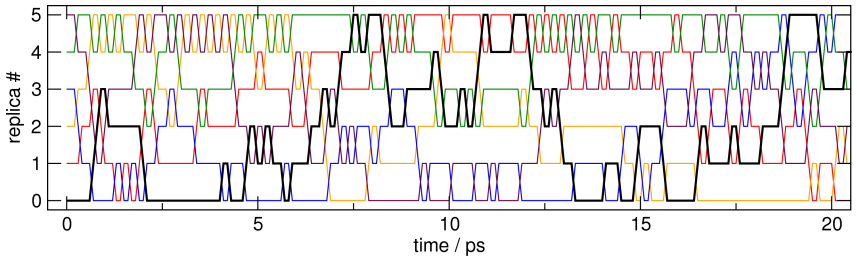

FIG. 5: Der Verlauf einer Hrex Simulation mit sechs Replikas, in der der Austausch der Koordinaten + Geschwindigkeiten alle 0.1 ps versucht wurde.

Das Prinzip des Hrex liegt also darin, dass jegliche Energiebarrieren durch die Skalierung der Kraftfeldterme verringert werden, und deshalb schneller überwunden werden. Schaut man sich die Gleichung für die Rate von einem Prozess an

$$k = A \cdot \exp\left[-\frac{E_A}{kT}\right]$$

so sieht man, dass eine Beschleunigung entweder durch Verringern der Aktivierungsenergie erreicht werden kann – was der Fall bei Hrex ist – oder durch Erhöhung der Temperatur. Deshalb reden wir auch bei Hrex manchmal über 'Aufheizen', auch wenn alle Simulationen bei normaler Temperatur (meist 300 K) durchgeführt werden, und es wird eine 'effektive' Temperatur von T/f angegeben.

Ein wichtiger Unterschied zur Metadynamik ist, dass keine Collective Variables zur Durchführung der Simulation benötigt werden. Erst bei der Analyse kann es dann praktisch sein, welche einzuführen, aber dies hat natürlich keinen Einfluss mehr auf den Verlauf der eigentlichen Simulation. Eine Wahl muss man allerdings vor der Durchfürung machen, und zwar welche Kraftfeldterme skaliert werden sollen. In der Regel konzentrieren sich diese auf einen Teil des Molekülsystems. In dieser Übung werden die nichtbindenden Terme sowie Diederwinkel des Dipeptids skaliert, wobei das Lösungsmittel in allen Replikas die vollen Wechselwirkungen ausübt – also die Kraftfeldterme des Wassers und ggf. des Gegenions bleiben unberührt.

2. Durchführung

Wir werden 4 Replikas simulieren, mit den Skalierungskoeffizienten 1, 0.75, 0.56 und 0.42. Wie sind denn die effektiven Temperaturen?

Für Hrex brauchen wir eine modifizierte Version von Gromacs, die mit den folgenden Befehlen geladen wird:

```
export PATH=/usr/local/run/openmpi-1.10.1/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/run/openmpi-1.10.1/lib
. /usr/local/run/gromacs-4.6.7-plumed-hrex/bin/GMXRC.bash
export PATH=/usr/local/run/plumed2-2.1-hrex/bin:$PATH
```

Zunächst wird die Topologie mit dem `grompp` Befehl post-prozessiert um eine Gesamt-Topologie des Systems zu erhalten:

```
grompp -f md -c npt -p topol -pp processed.top
```

Wie oben erwähnt, sollen sämtliche nichtbindenden Wechselwirkungen des Dipeptids skaliert werden. Hierfür muss die prozessierte Topologie für das Dipeptid zunächst leicht modifiziert werden. Dies geschieht, indem man sämtliche Atomtyp-Einträge des Dipeptids in der `[ atoms ]`-Sektion mit einem Unterstrich (`_`) versieht. Beispielsweise,

```
     1          HC      1    ACE   HH31       1      0.1123       1.008   ; qtot 0.1123
```

wird zu:

```
     1          HC_     1    ACE   HH31       1      0.1123       1.008   ; qtot 0.1123
```

Auf Grundlage der modifizierten, prozessierten Topologie können nun für die oben genannten Skalierungsfaktoren (ersetzt `$scale`) die vier dazugehörigen Topologien (`$i=0..3` ersetzen) erstellt werden:

```
mpirun -np 1 plumed partial_tempering $scale < processed.top > topol$i.top
```

Hierbei erzeugt z.B. der `partial_tempering` Befehl mit `$scale=1` die nichtskalierte Topologie `topol0.top`, und der `partial_tempering` Befehl mit `$scale=0.4` die am meisten skalierte Topologie `topol3.top`. Für die einzelnen Replikas werden daraufhin die zugehörigen `.tpr` Dateien erzeugt:

```
mpirun -np 1 grompp_mpi -maxwarn 1 -f md -c npt -p topol$i -o topol$i
```

Passen Sie auf, dass kein Barostat eingesetzt wird, da dies grundsätzlich in Replica Exchange Simulationen zu Instabilitäten (Crashes) führen kann. Die Hrex Simulation kann nun mit einer leeren `plumed.dat` Datei für die vier Replikas gestartet werden:

```
mpirun -np 4 mdrun_mpi -v -plumed -multi 4 -replex 100 -hrex
```

Dabei beschreibt die `-replex` Option nach wie viel Simulationsschritten ein Austausch zwischen den Replikas versucht werden soll.

3. Analyse

Die Austauschversuche werden in der Ausgabedatei `md0.log` verfolgt. Die entsprechenden Zeilen beginnen mit `Repl`, und es werden die Wahrscheinlichkeiten ausgeschrieben sowie getätigte Austauchaktionen (mit `X`). Sobald die Simulation zu Ende ist, werden die durchschnittlichen Austauschwahrscheinlichkeiten berechnet und ausgeschrieben. Die optimale Effizienz des Samplings wurde für Wahrscheinlichkeiten um 1/3 beobachtet; welchen Werten entspricht dies in Ihrer Simulation?

Gebraucht werden lediglich die Daten aus der ungestörten Simulation ($f = 1$, effektive Temperatur 300 K). Die Analyse ist identisch mit der einer normalen Simulation. Dabei soll bloß beachtet werden, dass keine kontinuierliche Dynamik beobachtet werden kann, denn die Austauschaktionen können zu 'Filmrissen' in der Simulation führen. Die Diederwinkel $\varphi - \psi$ werden mit dem dedizierten Gromacs Programm gemessen

```
g_rama -s topol0 -f traj0.trr
```

und unter `rama.xvg` gespeichert. In dieser Datei muss mit einem Texteditor der Header (wahrscheinlich die ersten 29 Zeilen) gelöscht werden. Dann wird aus den Zeitserien der Winkelpärchen mit einem gelieferten Skript die entsprechende 2D-Verteilung gebildet und unter `rama.histo` gespeichert:

```
./make_histogram.py
```

Das Histogramm, eigentlich ein Ramachandrandiagramm, kann mit einem weiteren Skript visualisiert werden

```
./rama-histo-to-png.sh
```

wo das Bild unter `rama-kcal.png` gespeichert wird. Die Farbe kodiert die (Helmholtz'sche, da NVT) freie Energie in kcal/mol. Falls die Farbskala zu eng (oder zu breit) ist, können Sie sie in der Datei `gnuplot-deltag.in` beliebig anpassen, und dann `./rama-histo-to-png.sh` erneut ausführen.

Welche dominanten Konformationen des Peptids sehen Sie in dem Ramachandrandiagramm? Welche weist die tiefste freie Energie auf und stellt damit das globale Minimum dar? Wie hoch sind die Energiebarrieren zwischen den einzelnen Konformationen? Stimmen die Ergebnisse mit denen aus der Metadynamik-Simulation überein?

VIII. QM/MM SIMULATION

Die offensichtlichste Limitierung der empirischen Kraftelder (MM) liegt darin, dass chemische Reaktionen nicht beschrieben werden können, also Prozesse wo kovalente Bindungen gebrochen werden oder entstehen. Dies kann man mit der Anwendung eines kombinierten Verfahrens umgehen, in dem man die Region, wo Chemie passiert, mit einer quantenchemischen Methode beschreibt, und den großen Rest des Systems berechnet man mit MM, wie sonst üblich. Diese hybriden Schemata werden QM/MM genannt, und der Nobelpreis für Chemie 2013 wurde zum Teil für die Entwicklung dieser Methoden vergeben.

In dieser Übung werden wir eine kleine chemische Reaktion in einem kleinen Molekül untersuchen, nämlich den Protonentransfer in einem Malonaldehydmolekül, siehe Bild 6. Der Vorteil ist dabei, dass die Simulation schnell laufen wird. Dies wird noch mehr beschleunigt, wenn eine effiziente QM Methode eingesetzt wird – hier wird es die semi-empirische Dichtefunktionaltheorie-Methode DFTB3 sein. Die technischen Details zu DFTB3 können Sie nachschauen, sie werden allerdings für die Durchführung sowie Analyse der Simulationen nicht dringend benötigt. DFTB3 ist ein Bestandteil von einer lokalen Version von Gromacs, es wird also kein externes Programm benötigt, um die quantenchmischen Berechnungen durchzuführen.

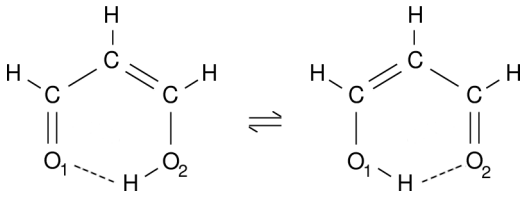

FIG. 6: Malonaldehyd – intramolekularer Protonentransfer.

A. Equilibrierung mit MM

Zuerst soll das Molekülsystem, ein solvatisiertes Malonaldehydmolekül, mithilfe eines üblichen Kraftfeldes, equilibriert werden. Dazu können Sie die gelieferte Topologie benutzen (Datei `mal.top`), sowie die Struktur in `mal.gro`, die Sie noch in eine kubische Wasserbox packen sollen. Der Ablauf wird schematisch wie folgt aussehen:

```
editconf ...
gmx solvate ...
grompp -f steep ...
mdrun -deffnm steep
grompp -f heat ...
mdrun -deffnm heat
grompp -f equil ...
mdrun -deffnm equil
```

wobei Simulationslänge recht kurz (unter 100 ps) gehalten werden kann. Die benötigten `.mdp` Dateien können aus dem Kapitel VII genommen werden.

B. Vorbereitung

Die wichtigste Frage in jeder QM/MM Simulation ist, wie man das Molekülsystem zwischen QM und MM Regionen teilt. Hier ist es sehr einfach: Ein Malonaldehydmolekül ist so klein, dass es ganz mit QM beschrieben werden kann, und das wässrige Lösungsmittel wird dann mit dem MM-Kraftfeld berechnet. Für eine QM/MM Simulation muss dann auch die Topologie des Moleküls modifiziert werden: Es wird keine Energie innerhalb der QM-Region mehr gerechnet, deshalb müssen alle Kraftfeldterme, die die QM-Region beschreiben, wegfallen.

Kopieren Sie die bestehende Topologie unter einem neuen Namen `mal-qmmm.top`. Bearbeiten Sie dann diese Datei mit einem Texteditor (z.B. `gedit`) wie folgt: Löschen Sie die Sektionen *angles*, *dihedrals* sowie *pairs*. Desweiteren ändern Sie den Typ von jeder Bindung von 1 auf 5, und löschen Sie die Parameter. Die Sektion sieht dann wie folgt aus:

```
 [ bonds ]
; ai   aj   funct      b0          kb
  1    2     5
  2    4     5
...
```

Damit ist die Topologie vorbereitet. Es werden gar keine intramolekularen Wechselwirkungen innerhalb des Malonaldehydmoleküls mit dem Kraftfeld berechnet. Die Ladungen der QM-Atome können in der Topologiedatei bleiben, denn sie werden von Gromacs automatisch auf Null gesetzt. Die Wechselwirkungen zwischen der QM und MM Regionen werden geteilt berechnet: die QM-Methode berücksichtigt die elektrostatische (Ladung–Ladung) WW,[8] und das MM-Programm wendet das übliche Lennard-Jones Potential an; deshalb müssen auch den QM-Atomen Atomtypen zugewiesen werden, und LJ-Parameter geliefert werden.

C. Ausführung

Die `.mdp` Datei wird nun einige weitere Optionen enthalten, die zum Großteil das Verhalten der quantenchemischen Methode DFTB3 steuern. Diese sind die folgenden, siehe auch die Datei `long.mdp`:

```
QMMM                    = yes
QMMM-grps               = MAL
QMmethod                = RHF
QMMMscheme              = normal
QMbasis                 = STO-3G
QMcharge                = 0
QMmult                  =
MMChargeScaleFactor     = 1

QMdftbsccmode           = 3
QMdftb-telec            = 10.
QMdftb-slko-path        = /usr/local/run/gromacs-5.0-dftb-v6a-plumed/share/gromacs/top/dftb/3ob/
QMdftb-slko-separator   =
QMdftb-slko-lowercase   = yes
QMdftb-slko-suffix      = -c.spl
QMdftb-partial-pme      = 1
QMdftb-dispersion       = 1
QMdftb-cdko             = 0
```

Zuerst wird festgelegt, dass eine QM/MM Simulation laufen soll, und die QM Region definiert. 'MAL' bezieht sich dabei auf den Namen einer Gruppe in der Indexdatei (hier `index.ndx`), die also mit

```
make_ndx -f equil
```

erzeugt werden muss und dem `grompp_d` anschließend mitgegeben. Die weiteren drei Optionen werden ignoriert (Vorsicht, es wird keine RHF/STO-3G Berechnung gemacht). Wichtig kann allerdings sein, die Ladung (sowie Spinmultiplizität) der QM-Region anzugeben. Die verbleibenden Optionen sind dann wirklich DFTB3-spezifisch, und müssen nicht im Detail verstanden werden. Es ist bloß empfehlenswert zu prüfen, ob der Ordner mit DFTB3 Parametern existiert (`QMdftb-slko-path`).

Genauso wie bei reinen MM-Simulationen muss erst der Gromacs Preprozessor aufgerufen werden, gefolgt von dem eigentlichen Simulieren. Ein wichtiger Unterschied ist hier, dass die double-precision Versionen von allen Gromacs Programmen zu benutzen sind. Dies ist einfach – zu jedem Programmname hängen Sie einfach den Suffix `_d` an:

```
grompp_d -f long -c equil -p mal-qmmm -n index -o long
mdrun_d -deffnm long -nt 1 -v > mdrun.out
```

Dabei ist zu beachten, dass die QM/MM-Topologie benutzt wird. Die freie QM/MM Simulation wird eine etwas längere Zeit in Anspruch nehmen, ca. 40 min für eine Simulation von 100 ps.

---

[8] Die Ladungen der QM Atome werden mithilfe der Mulliken Analyse in der quantenchemischen Berechnung ermittelt.

D. Auswertung

Betrachten Sie zuerst die Trajektorie visuell (`vmd equil.gro long.trr`). Sie können das Wasser ausblenden (*Graphics – Representations* and 'not water' für Selected Atoms), oder direkt die `.xtc` Datei betrachten (`vmd mal.gro long.xtc`). Sehen Sie einen Protonentransfer? Wenn die originale O–H Bindung ungewöhnlich lang wird, kann es bedeuten, dass diese gebrochen ist. Um diesen Darstellungsfehler zu vermeiden, können Sie die Darstellung auf 'Dynamic bonds' ändern.

Um den Protonentransferprozess quantitativ zu verfolgen, sollen wir nun eine passende Reaktionskoordinate wählen – also eine Größe, die wir aus den Koordinaten der Atome berechnen können, und die die Reaktion möglichst ungestört von anderen Einflüssen beschreibt. Bei so einem einfachen Protonentransfer eignet sich sehr gut die Differenz der beiden O–H Abstände.

Die Aufgabe ist nun diese Abstände zu messen und ihre Differenz auszurechnen, das Ganze entlang der Trajektorie aus der QM/MM Simulation. Auch wenn Gromacs selbst natürlich schon Abstände von Atomen messen kann, kann es hier vorteilhaft sein ein externes Tool anzuwenden, nämlich das Plumed Programm. Einer der Zwecke von Plumed ist nämlich das Arbeiten mit verschiedenen Reaktionskoordinaten zu erleichtern.

Erstellen Sie hierfür mittels eines Texteditors eine Eingabedatei für Plumed unter dem Namen `diffdist.dat`:

```
d1: DISTANCE ATOMS=1,9
d2: DISTANCE ATOMS=8,9
d:  COMBINE ARG=d1,d2 COEFFICIENTS=1,-1 PERIODIC=NO
PRINT ARG=d FILE=diffdist.xvg
```

Inhaltlich selbsterklärend, werden hier beide Abstände definiert, dann ihre Differenz, die dann gemessen und in eine Ausgabedatei gedruckt wird. Sie führen den Vorgang wie folgt durch:

```
plumed driver --mf_xtc long.xtc --plumed diffdist.dat --timestep 0.01
```

Die resultierende Datei können sie mit `xmgrace diffdist.xvg` betrachten. Was sehen Sie?

In einer QM/MM Simulation kann uns auch die zeitliche Entwicklung der Elektronendichte interessieren. Diese wird im Rahmen der DFTB Methode durch Partialladungen auf den einzelnen QM Atomen dargestellt. Unser Gromacs Programm speichert die Atomladungen in der Datei `qm_dftb_charges.xvg`, die Sie mit `xmgrace -nxy` betrachten können. Hier ist auf der x-Achse Zeit in ps, und auf der y-Achse die Ladungen; jedem QM-Atom entspricht eine Kurve. Können Sie aus dem Verlauf der Ladungen auf Zusammenhang mit dem Protonentransfer schließen?

Oft interessiert uns nicht so ganz der Verlauf einer Größe, sondern eher die Wahrscheinlichkeit, mit der verschiedene Werte eingenommen werden. Um eine Wahrscheinlichkeitsdichte der Differenz der O–H Abstände zu bilden, benutzen Sie eins der Gromacs-Analysprogramme:

```
g_analyze -f diffdist -dist diffdist-histo -bw 0.005
```

Das Programm meldet den Mittelwert sowie seine Standardabweichung, und gibt das Histogramm in einer neuen Datei aus. Was für einen Mittelwert, und was für eine Form des Histogramms würden Sie erwarten, vorausgesetzt wäre eine ausreichend lange Simulationsdauer? Wird Ihre Erwartung erfüllt?

E. Metadynamik

Um konvergierte freie Energien zu erhalten, kann es notwendig sein lange Simulationen durchzuführen. Das mag insbesondere mit QM/MM Simulationen unmöglich lange Rechenzeiten in Anspruch nehmen. Dieses Problem lässt sich auch hier durch die Anwendung der Extended-Sampling Techniken lösen. Hier werden wir die Protonentransferenergetik mit Metadynamik untersuchen.

Dazu können Sie die bestehende `.tpr` Datei benutzen, die Sie gerade für die freie Simulation vorbereitet hatten. Was zusätzlich gebraucht wird, ist eine Eingabedatei für Plumed (`wt-metad.dat`), in der eine Metadynamiksimulation aufgerufen wird. Diese kann wie folgt aussehen (zu vergleichen mit der Eingabe für das Dipeptid in Sektion VII B), wo wieder die well-tempered Variante der Metadynamik gemacht werden soll:

```
d1: DISTANCE ATOMS=1,9
d2: DISTANCE ATOMS=8,9
d:  COMBINE ARG=d1,d2 COEFFICIENTS=1,-1 PERIODIC=NO
METAD ...
  LABEL=metad
  ARG=d
  PACE=100
  HEIGHT=0.625
  SIGMA=0.02
  FILE=HILLS
  BIASFACTOR=5.
  TEMP=300.
... METAD
PRINT STRIDE=100 ARG=d,metad.bias FILE=plumed.xvg
```

Sobald die Simulation fertig ist, oder sogar noch während sie läuft, kann die `HILLS` Datei analysiert werden, um die Kurve der freien Energie (ΔG) zu konstruieren. Wie vorher, kann dies mit dem Plumed Programm gemacht werden:

```
plumed sum_hills --bin 150 --min -0.15 --max 0.15 --hills HILLS
```

Die resultierende freie Energie in der Datei `fes.dat` kann mit Xmgrace betrachtet werden. Stimmt die Kurve mit der aus der freien Simulation überein? Weist die Kurve die erwartete Symmetrie auf? Wie hoch ist die Barriere? Wenn wir die Protonentransferrate aus der Barrierenhöhe berechnen möchten, würden wir die Rate möglicherweise unterschätzen. Warum?