Data and Molecular Visualization

J. D. Madura

Department of Chemistry and Biochemistry
Center for Computational Chemistry
Duquesne University

June 1, 2016
Schedule of Lectures

Lecture 1
- Visualization (VMD, PyMol, MOE and ngl)
- Introduction to $\mathbb{R}$ and python
- Plotting using $\mathbb{R}$ and python
- Examples

Lecture 2
- IMF
- Force-fields
- Molecular Dynamics (MD)
- CHARMM-GUI
- Example: Simulation of a protein ($in$ $vacuo$ and in solution)

Lecture 3
- MD analysis (pytraj)
- Examples
Lecture Objectives

- Learn how to create scientific plots using \texttt{R} and python
- Learn how to visualize (bio)molecular using VMD, MOE, PyMol, etc.
Introduction

Overview

- Why is visualization important?
- Data versus Molecular visualization.

Data

Molecular

Simple Plot

Understanding

Time/day

VMD
- Input
- Graphics
- Extensions
- Saving and Capturing Graphics
- Movie Making
- MOE
- Input
- Graphics
- Selection
- Compute

Learning Outcomes
An Example

Which of the following is more appropriate?

<table>
<thead>
<tr>
<th>Atom</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>-2.076665</td>
<td>-0.450512</td>
<td>0.113719</td>
<td>0.31</td>
</tr>
<tr>
<td>6</td>
<td>-1.303573</td>
<td>-1.578362</td>
<td>0.018659</td>
<td>-0.25</td>
</tr>
<tr>
<td>6</td>
<td>0.052338</td>
<td>-1.468019</td>
<td>-0.259675</td>
<td>-0.24</td>
</tr>
<tr>
<td>6</td>
<td>0.632158</td>
<td>-0.225992</td>
<td>-0.442758</td>
<td>-0.07</td>
</tr>
<tr>
<td>6</td>
<td>-0.168762</td>
<td>0.907155</td>
<td>-0.343817</td>
<td>-0.23</td>
</tr>
<tr>
<td>6</td>
<td>-1.510177</td>
<td>0.802410</td>
<td>-0.068220</td>
<td>0.38</td>
</tr>
<tr>
<td>1</td>
<td>-1.744472</td>
<td>-2.546830</td>
<td>0.156610</td>
<td>0.24</td>
</tr>
<tr>
<td>1</td>
<td>0.650035</td>
<td>-2.354561</td>
<td>-0.336670</td>
<td>0.24</td>
</tr>
<tr>
<td>1</td>
<td>0.240234</td>
<td>1.886524</td>
<td>-0.485182</td>
<td>0.26</td>
</tr>
<tr>
<td>8</td>
<td>-2.290785</td>
<td>1.925857</td>
<td>0.025961</td>
<td>-0.76</td>
</tr>
<tr>
<td>1</td>
<td>-3.206353</td>
<td>1.679199</td>
<td>0.218002</td>
<td>0.42</td>
</tr>
<tr>
<td>8</td>
<td>-3.438723</td>
<td>-0.428600</td>
<td>0.384651</td>
<td>-0.76</td>
</tr>
<tr>
<td>1</td>
<td>-3.824804</td>
<td>-1.304988</td>
<td>0.485354</td>
<td>0.40</td>
</tr>
<tr>
<td>6</td>
<td>2.118392</td>
<td>-0.091134</td>
<td>-0.700641</td>
<td>-0.42</td>
</tr>
<tr>
<td>1</td>
<td>2.509992</td>
<td>-0.977440</td>
<td>-1.182030</td>
<td>0.24</td>
</tr>
<tr>
<td>1</td>
<td>2.306003</td>
<td>0.750995</td>
<td>-1.358991</td>
<td>0.21</td>
</tr>
<tr>
<td>6</td>
<td>2.894472</td>
<td>0.113603</td>
<td>0.615320</td>
<td>-0.17</td>
</tr>
<tr>
<td>1</td>
<td>2.490833</td>
<td>0.979069</td>
<td>1.138684</td>
<td>0.19</td>
</tr>
<tr>
<td>1</td>
<td>2.737549</td>
<td>-0.751163</td>
<td>1.246935</td>
<td>0.22</td>
</tr>
<tr>
<td>7</td>
<td>4.334095</td>
<td>0.211951</td>
<td>0.325333</td>
<td>-0.80</td>
</tr>
<tr>
<td>1</td>
<td>4.587156</td>
<td>1.050440</td>
<td>-0.167043</td>
<td>0.29</td>
</tr>
<tr>
<td>1</td>
<td>4.922120</td>
<td>0.072153</td>
<td>1.126572</td>
<td>0.30</td>
</tr>
</tbody>
</table>

Explain...
Visualizing Data

“Assessments of change, dynamics, and cause and effect are at the heart of thinking and explanation. To understand is to know what cause provokes what effect, by what means, at what rate. How then is such knowledge to be represented?”¹

The goal is to design “...proper arrangement in space and time images, words, numbers – for presenting information about motion, process, mechanism, cause, and effect.”¹

Therefore visualization, in our case molecular visualization, is extremely important since it is an extremely effective method to convey information.

**Visualizing Data**

“Excellence in statistical graphics consists of complex ideas communicated with clarity, precision, and efficiency.” A quote from Edward R. Tufte

Therefore graphical data should

- show that data
- induce the viewer to think about the substance rather than about the methodology
- avoid distorting what the data have to say
- make large datasets coherent
- encourage the eye to compare different pieces of data
- reveal that data at several levels of detail, from a broad overview to the fine structure
- serve a reasonably clear purpose: description, exploration, tabulation, or decoration
- be closely integrated with the statistical and verbal description of a data set.
Graphical Example

Consider the following data

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th></th>
<th>II</th>
<th></th>
<th>III</th>
<th></th>
<th>IV</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>10.0</td>
<td>Y</td>
<td>8.04</td>
<td></td>
<td>10.0</td>
<td>Y</td>
<td>9.14</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>8.0</td>
<td></td>
<td>6.95</td>
<td>Y</td>
<td>8.0</td>
<td>Y</td>
<td>8.14</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>13.0</td>
<td></td>
<td>7.58</td>
<td>Y</td>
<td>13.0</td>
<td>Y</td>
<td>8.74</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>9.0</td>
<td></td>
<td>8.81</td>
<td>Y</td>
<td>9.0</td>
<td>Y</td>
<td>8.77</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>11.0</td>
<td></td>
<td>8.33</td>
<td>Y</td>
<td>11.0</td>
<td>Y</td>
<td>9.26</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>14.0</td>
<td></td>
<td>9.96</td>
<td>Y</td>
<td>14.0</td>
<td>Y</td>
<td>8.10</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>6.0</td>
<td></td>
<td>7.24</td>
<td>Y</td>
<td>6.0</td>
<td>Y</td>
<td>6.13</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>4.0</td>
<td></td>
<td>4.26</td>
<td>Y</td>
<td>4.0</td>
<td>Y</td>
<td>3.10</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>12.0</td>
<td></td>
<td>10.84</td>
<td>Y</td>
<td>12.0</td>
<td>Y</td>
<td>9.13</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>7.0</td>
<td></td>
<td>4.82</td>
<td>Y</td>
<td>7.0</td>
<td>Y</td>
<td>7.26</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>5.0</td>
<td></td>
<td>5.68</td>
<td>Y</td>
<td>5.0</td>
<td>Y</td>
<td>4.74</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th></th>
<th>Y</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8.0</td>
<td>6.58</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>5.76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>7.71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>8.84</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>8.47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>7.04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>5.25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19.0</td>
<td>12.50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>5.56</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>7.91</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>6.89</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table:** \( N = 11; \) mean of \( X' \)'s = 9.0; mean of \( Y' \)'s = 7.5; equation of regression line: \( Y = 3 + 0.5X \); standard error of estimate of slope = 0.118; \( t = 4.24 \); sum of squares \( X - \bar{X} = 110.0 \); regression sum of square = 27.50; residual sum of squares of \( Y = 13.75 \); correlation coefficient = 0.82; and \( r^2 = 0.67 \).
Graphical Example Visualized

I

II

III

IV
Another Example
One More Example

![Graphs showing potential energy over time for Mg²⁺ in 42 H₂O shell.](image)
Proper Figure Guidelines

Characteristics of a quality graph / plot

- Use complete and consistent axis labels
  - Label each axis with the parameter or variable being measured and the units of measure in parentheses.
  - Use initial capital letters only, not all capitals: Time (min), Reaction Temperature (°C), Thichness (µm).
  - Place all labels outside and parallel to the axes. Numbers and letters on the abscissa and ordinate should read from left to right and from bottom to top, respectively.
  - Do not place arrowheads on the ends of the axis lines.
  - Label the tick marks on an axis in type that is one or two points smaller than the axis labels (but not smaller than 7 points in general, 5 points for ACS journals).
- Use symbols for discrete data points while lines are used for continuous data, e.g. plotting a function.
- Axes should be ±10% larger than the min/max of the data.

Consult the ACS Style Guide, chapter 15, for complete details.
Proper Figure Guidelines, cont.

Every figure must have a caption that includes the figure number and a brief, informative description, preferably in nonsentence format. Good examples of figure captions:

- Figure 2. Mass spectrum obtained when laboratory ambient air containing 2.5 ppm of 1 was introduced into the MS system.
- Figure 4. Change in carotenoid contents during maturation of three varieties of grapes: (A) Concord grapes; (B) Thompson seedless; and (C) Chilean red.
- Figure 7. Reaction rate constants as a function of proton afPnity for the reactions shown in eqs 5–7: $k_{\text{exp}}$, experimental; $k_c$, calculated.

If more information is necessary, use complete sentences and standard punctuation. The caption should be understandable without reference to the text.
Introduction to $\texttt{R}$

$\texttt{R}$ is a statistical and graphics program.

You can download the program from http://cran.r-project.org.

It has a simple language to facilitate the input and output of your data and the results.

A nice introduction to $\texttt{R}$ can be found in the book “Computer Simulation and Data Analysis in Molecular Biology and Biophysics: An Introduction Using $\texttt{R}$” by Victor Bloomfield.

YouTube is an excellent source of videos instructing the viewer on how to install and use $\texttt{R}$. See the YouTube video on installation at http://www.youtube.com/watch?v=WJDrYUqNrHg
```
x=c(1,2,3,4,5)
y=c(1,2,3,4,5)
plot(x,y,xlab="x data",ylab="y data")
```
Improved Simple Plot

Make the x and y axis labels larger and orient y-axis labels.

```r
x = c(1, 2, 3, 4, 5)
y = c(1, 2, 3, 4, 5)
par(cex=1.5, las=1)
plot(x, y, xlab="x data", ylab="y data")
```
Improved Simple Plot, cont.

Change the tick marks to be inside the plot.

```r
x=c(1,2,3,4,5)
y=c(1,2,3,4,5)
par(cex=1.5,las=1,tck=0.025)
plot(x,y,xlab="x data",ylab="y data")
```
Improved Simple Plot, cont.

Add some color to the data points.

```r
x=c(1,2,3,4,5)
y=c(1,2,3,4,5)
par(cex=1.5, las=1, tck=0.025)
plot(x, y, xlab="x data", ylab="y data",
    + col="red")
```
Improved Simple Plot, cont.

Add a second set of data points.

```r
x <- c(1, 2, 3, 4, 5)
y <- c(1, 2, 3, 4, 5)
z <- c(1, 4, 9, 16, 25)
par(cex = 1.5, las = 1, tck = 0.025)
plot(x, y, xlab = "x data", ylab = "y data", 
     col = "red", ylim = c(min(y, z), max(y, z)))
points(x, z, col = "blue")
```
Improved Simple Plot, cont.

Add a legend.

```r
x=c(1,2,3,4,5)
y=c(1,2,3,4,5)
z=c(1,4,9,16,25)
par(cex=1.5,las=1,tck=0.025)
plot(x,y,xlab="x data",ylab="y data",
+  col="red", ylim=c(min(y,z),max(y,z))
points(x,z,col="blue")
legend(2,20,c("red data","blue data"),
+  col=c("red","blue"),pch=c(1,1),bty="n")
```
Improved Simple Plot

Annotations and saving the plot to a file.

```r
x=c(1,2,3,4,5)
y=c(1,2,3,4,5)
z=c(1,4,9,16,25)
png(file="simple07.png")
plot(x,y,xlab="x data",ylab="y data",
+ col="red", ylim=c(min(y,z),max(y,z)),
+ main="Red versus Blue")
points(x,z,col="blue")
legend(1,25,c("red data","blue data"),
+ col=c("red","blue"), pch=c(1,1), bty="n")
axis(3,labels=FALSE)
axis(4,labels=FALSE)
arrows(2,15,3,9)
text(2,16,"This one")
dev.off()
```
Fitting Data

One of the skills you will need is fitting your data to some type of equation. The most common fitting is linear least squares. Everyone is familiar with the ideal gas law $PV = nRT$. Consider the following data collected for 1.0 mol of CO$_2$ at 298.15 K.

<table>
<thead>
<tr>
<th>$P$/bar</th>
<th>$V$/L</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>99.1076</td>
</tr>
<tr>
<td>0.50</td>
<td>49.5538</td>
</tr>
<tr>
<td>0.75</td>
<td>33.0359</td>
</tr>
<tr>
<td>1.00</td>
<td>24.7769</td>
</tr>
<tr>
<td>5.00</td>
<td>4.9554</td>
</tr>
<tr>
<td>10.00</td>
<td>2.4777</td>
</tr>
</tbody>
</table>

Use this data to determine a value for $R$. 
The Solution using R

\begin{verbatim}
P=c(0.25,0.5,0.75,1.0,5.0,10.0)
V=c(99.1076,49.5538,33.0359,24.7769,4.9554,2.4777)
png(file="idgfit.png")
par(cex=1.75,las=1)
plot(1/P,V,
+ xlab=expression(paste("P"^"-1"," / (bar"^"-1","))
+ ylab="V / (L)")
x=1/P
fit=lm(V~x)
abline(fit,col="red")
text(1,80,bquote(paste("slope = ",
+ .(fit$coefficients[2]))))
dev.off()
\end{verbatim}

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{idgfit.png}
\caption{Graph showing the relationship between V and P.}
\end{figure}

slope = 24.7769
Nonlinear Data Fitting

Occasionally one encounters a situation in which the expression you need to fit the data is non-linear. Consider the following scenario. You collect the following data for an enzyme (5 µM) 298.15 K.

<table>
<thead>
<tr>
<th>[Substrate] / mM</th>
<th>Ψ₀ / mM/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.02</td>
<td>10.83</td>
</tr>
<tr>
<td>0.04</td>
<td>18.57</td>
</tr>
<tr>
<td>0.07</td>
<td>26.76</td>
</tr>
<tr>
<td>0.10</td>
<td>32.50</td>
</tr>
<tr>
<td>0.15</td>
<td>39.00</td>
</tr>
<tr>
<td>0.20</td>
<td>43.33</td>
</tr>
<tr>
<td>0.30</td>
<td>48.75</td>
</tr>
<tr>
<td>0.50</td>
<td>54.17</td>
</tr>
<tr>
<td>0.70</td>
<td>56.88</td>
</tr>
</tbody>
</table>

which must be fit to the following expression.

\[
\Psi = \frac{V_{\text{max}}[S]}{K_M + [S]}
\]
The Solution using R

S=c(0.02,0.04,0.07,0.10,0.15,0.20,0.30,0.50,0.70)
v0=c(10.83,18.57,26.76,32.50,39.00,43.33,48.75,54.17,56.88)
png(file="mmenzyme.png")
par(cex=1.75,las=1,tck=0.025)
plot(S,v0,xlab=expression([S] / mM),
+ ylab=expression(paste("v"[o]," / mM s"^"−1")))
axis(3,labels=FALSE)
axis(4,labels=FALSE)
KM=0.5
Vmax=50
fit = nls(v0 ~ ((Vmax*S)/(KM+S)),
+ start=list(KM=KM,Vmax=Vmax))
yfit=fitted(fit)
points(S,yfit,type="l",col="red")
text(0.3,30,bquote(K[M] = .(signif(coef(fit)[1],digits=2))))
text(0.3,25,bquote(V[ max] = .(signif(coef(fit)[2],digits=2))))
dev.off()
A classic example of a hydrogen bond is the water dimer.

Using Gaussian we can generate a potential energy scan for the energy versus distance between the oxygen atoms.

```latex
%chk=dimer.chk

# hf/sto-3g opt=z-matrix

Water dimer potential energy scan

0 1
h
o 1 roh
h 2 rho 1 hoh
o 2 roo 1 ooh 3 0.0
h 4 rho1 3 hoh1 1 d1
h 4 rho1 3 hoh1 1 -d1

roh 0.988
rho 0.9897
roo 2.3 s 50 0.05
rho1 0.9871
hoh 100.3585
ooh 100.4281
hoh1 110.4991
d1 124.5952
```
Water Dimer

After running the Gaussian calculation for the water dimer look in the output file for

<table>
<thead>
<tr>
<th>Summary of Optimized Potential Surface Scan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenvalues — —149.91807 —149.92838 —149.93180 —149.93492 —149.93730</td>
</tr>
<tr>
<td>roh  0.98800  0.98710  0.98660  0.98671  0.98701</td>
</tr>
<tr>
<td>rho  0.96936  0.97592  0.97671  0.97884  0.98162</td>
</tr>
<tr>
<td>roo  2.30000  2.35000  2.40000  2.45000  2.50000</td>
</tr>
<tr>
<td>rho1 0.98471  0.98552  0.98532  0.98549  0.98579</td>
</tr>
<tr>
<td>hoh 102.28153 102.90172 102.38682 101.62750 101.23188</td>
</tr>
<tr>
<td>ooh 123.21440 126.89952 115.20560 106.64471 103.93286</td>
</tr>
<tr>
<td>hoh1 106.38848 105.50372 108.04529 110.58072 111.11540</td>
</tr>
<tr>
<td>d1 126.66192 126.70510 125.41616 124.05844 123.83537</td>
</tr>
<tr>
<td>6   7   8   9   10</td>
</tr>
<tr>
<td>Eigenvalues — —149.93900 —149.94014 —149.94083 —149.94117 —149.94124</td>
</tr>
<tr>
<td>roh  0.98729  0.98755  0.98772  0.98792  0.98810</td>
</tr>
</tbody>
</table>

Using this data create a file, e.g. dimer.dat, with two columns. The first column is the number of the frame and the second column is the Eigenvalue. For example

1, —149.91807
2, —149.92838

.
Water Dimer Plot

The \texttt{R}-code to generate the water dimer plot

```r
data = read.table("dimer.dat", header=F, sep="","n
x = 2.3 + (data[,1] - 1) * 0.05
y = (data[,2] - 2 * (-74.9659012)) * 627.51
png("dimerplt.png")
par(cex=1.5, mgp=c(2, 0.75, 0), tck=0.025, las=1)
plot(x, y, type="l", main="Relative Potential Energy",
+ xlab=expression(r[oo]/A),
+ ylab=expression(paste("Potential Energy / (kcal \cdot mol^{\text{-1}},")
+ axes=F, frame.plot=T)
axis(1, at=seq(min(x), max(x), 0.5))
axis(2, at=seq(-6, 8, 1))
axis(3, at=seq(min(x), max(x), 0.5), labels=F)
axis(4, at=seq(-6, 8, 1), labels=F)
abline(h=0, lty=3, col="red")
dev.off()
```

![Relative Potential Energy](dimerplt.png)
Python

Python is an interpreted programming language popular for its simple and powerful syntax. Python can easily be used with little to no programming experience. Below are examples of plotting using python. The examples include:

1. A simple plot of a function
2. An advance plot of a function
3. Plot of data
Python, cont.

Code for a simple plot of a function

```python
import numpy as np
import matplotlib.pyplot as plt

# generate data
def f(x):
    return np.sin(x)
xminimum = 0
xmaximum = 2*np.pi
numpoints = 10000
yminimum = -1
ymaximum = 1
xvec = np.linspace(xminimum, xmaximum, num=numpoints, dtype=np.float64)
yvec = []
for x in xvec:
yvec.append(f(x))

# plot commands
fig = plt.figure()
plt.plot(xvec, yvec)

# Uncomment to save the figure
# fig.savefig('simpleplot.png')
plt.show()
```

**Figure:** A simple plot of \( \sin(x) \).
Python, cont.

```python
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams({'font', family='serif', size=24})

# generate data

def f(x):
    return np.sin(x)

fvec = np.vectorize(f)
xminimum = 0
xmaximum = 2 * np.pi
numpoints = 10000
yminimum = -1
ymaximum = 1
xvec = np.linspace(xminimum, xmaximum, num=numpoints, dtype=np.float_)
yvec = fvec(xvec)

# plot commands
fig = plt.figure(facecolor='white', figsize=(8, 8))
ax = fig.add_subplot(111)
plt.xlim(xminimum, xmaximum)
plt.ylim(yminimum - 0.1, ymaximum + 0.1)
plt.plot(xvec, yvec, color='red', linewidth=2.1)
plt.plot([xminimum, xmaximum], [0, 0], color='black', linestyle='-', linewidth=2)
for axis in ['top', 'bottom', 'left', 'right']:
    ax.spines[axis].set_linewidth(3)
plt.xticks([0, np.pi, 2 * np.pi], ['0', r'$\pi$', r'$2\pi$'])
plt.xlabel(r'$x$')
plt.ylabel(r'sin($x$)')
plt.title(r'Plot of sin($x$)', y=1.08)
fig.tight_layout()
fig.savefig('advplot.png')
plt.show()
```
**Python, cont.**

```python
def main():
    # Import necessary libraries
    import numpy as np
    import matplotlib.pyplot as plt
    plt.rc('font', family='serif', size=24)

    # Read data
    separationDataToRead = "Spacing.txt"
    waterDimerDataToRead = "WaterDimerEnergy.txt"
    singleWaterDataToRead = "WaterEnergy.txt"

    separation = np.loadtxt(separationDataToRead, dtype=np.float64)
    waterDimer = np.loadtxt(waterDimerDataToRead, dtype=np.float64)
    singleWater = np.loadtxt(singleWaterDataToRead, dtype=np.float64)

    waterDimer = waterDimer - (2.0 * singleWater)

    xminimum = min(separation) - 0.05
    xmaximum = max(separation) + 0.05
    yminimum = min(waterDimer) - 0.005
    ymaximum = max(waterDimer) + 0.005

    # Plot commands
    fig = plt.figure(facecolor='white', figsize=(8, 8))
    ax = fig.add_subplot(111)
    plt.xlim(xminimum, xmaximum)
    plt.ylim(yminimum, ymaximum)
    plt.plot(separation, waterDimer, color='red', linewidth=2.1)
    plt.plot([xminimum, xmaximum], [0, 0], color='black', linestyle='-', linewidth=2)
    for axis in ['top', 'bottom', 'left', 'right']:
        ax.spines[axis].set_linewidth(3)
        plt.xticks([0, np.pi, 2*np.pi], ['0', r'$\pi$', r'$2\pi$'])
    ax.xaxis.set_tick_params(width=2.5, length=7)
    ax.yaxis.set_tick_params(width=2.5, length=7)
    plt.xlabel(r'Separation / ($\AA$)')
    plt.ylabel(r'Energy / (kcal mol$^{-1}$)')
    plt.title(r'Dimer Energy', y=1.08)
    fig.tight_layout()
    fig.savefig('dimerplot.png')
    plt.show()

if __name__ == '__main__':
    main()
```
VMD Overview

- **Visual Molecular Dynamics**
- Open-source program
- Performs visualization, analysis, and interfaces with several external programs.
  - NAMD
  - APBS
  - MSMS
- Extended using Tcl/Tk

The program, VMD, can be freely downloaded from http://www.ks.uiuc.edu/Research/vmd/

There are several VMD tutorials examples on this webpage.
Opening Screen

Dialog Box

Description

- 3D Window
- GUI
  - File
  - Molecule
  - Graphics
  - Display
  - Mouse
- Extensions
  - Help
- Terminal window
Molecule Input Screen

Dialog Box

Description

- File types / extensions
  - PDB
  - COOR
  - DCD
  - PQR
  - MOL2
Representations

Screen

Description

• Coloring Method
• Drawing Method
• Material
Coloring Method

Dialog Box

Description

- Type
- Element
- ResName
- ResType
- Chain
- Conformation
- Secondary Structure
- ColorID
- Volume
Drawing Method

Dialog Box

Description

- Lines
- Bonds
- VDW
- CPK
- Trace
- Ribbons
- Cartoon
- Surf
- isosurface
Representations

Example 1

Example 2

Example 3
Analysis

Description

- APBS Electrostatics
- Contact Map
- NAMD Energy
- NAMD Plot

Screen

- Ramachandran Plot
- RMSD Calculator
- Sequence Viewer

Analysis Description

- APBS Electrostatics
- Contact Map
- NAMD Energy
- NAMD Plot

Screen
Modeling

Dialog Box

Description

- Add Ions
- Add Solvation Box
- Automatic PSF Builder
- Membrane Builder
- Mutate Residue
## Simulation

### Dialog Box

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoIMD</td>
</tr>
<tr>
<td>IMD Connect</td>
</tr>
<tr>
<td>NAMD Graphical Interface</td>
</tr>
<tr>
<td>QM Tool</td>
</tr>
</tbody>
</table>

**Example Dialog Box**

![Example Dialog Box](image-url)
Visualization

Dialog Box

Description

- Clipping Plane Tool
- Color Scale Bar
- Movie Maker
- Multiple Molecule Animation
- Ruler
Under the File Menu

Dialog Box

Description

- Save State
- Render
  - Snapshot
  - Tachyon
  - POV3
  - Postscript
Animation

Protein in Water
MOE Overview

- **Molecular Operating Environment**
- Commercial software
- Integrated visualization, computation and analysis
- Can connect to 3rd party software (e.g. Gaussian, MOPAC, NAMD, APBS, etc.)
- Written in SVL (Scientific Vector Language) and can be extended.

MOE is a commercial program. You can access the program while using the campus network. To install and get the appropriate license file see Scott Boesch.
Opening Screen

MOE 3D Screen

Description

- Menu items across the top
- Buttons down the right side
- 3D viewing
- Bottom right corner
Build

Dialog Box

Description

- Build by atoms
- Build by groups
- Build using SMILES
- Assign ionization state
- Assign stereochemistry
- Set a geometric parameter
Dialog Box

Description

- Numerous file types can be read
  - PDB
  - MOL2
  - moe
  - Fasta
- Text edit the file
Visualization

J. D. Madura

Schedule of Lectures

Lecture Objectives

Introduction

Data Visualization

Scientific Plots

Introduction to R

Water Dimer Example

Python

Molecular Visualization

VMD

Input Graphics

Extensions

Saving and Capturing Graphics

Movie Making

MOE

Input Graphics

Selection

Compute

Learning Outcomes

Render

Screen

Description

- View
- Hide
- Show
- Atoms
- Ribbons
- Draw
Commands

- Invert - inverts the current selection
- Extend - extends the current selection based on selected criteria
- Ligand - selects the ligand
- Pocket - selects the pocket
- Receptor - selects the receptor
- Solvent - selects the solvent
- Potential - selects to force-field to be used
- Atom Selector - selects atoms based on certain criteria
Atom Selection

Dialog Box

Description

- Works on selected subsets of atoms
- Save selection sets
- Pick by Element
- Pick based on proximity
- Extends to complete groups, e.g. residues
- Picks based on connectivity

- Works on selected subsets of atoms
- Save selection sets
- Pick by Element
- Pick based on proximity
- Extends to complete groups, e.g. residues
- Picks based on connectivity

• Works on selected subsets of atoms
• Save selection sets
• Pick by Element
• Pick based on proximity
• Extends to complete groups, e.g. residues
• Picks based on connectivity
Computations Available

- Potential Energy
- Site Preparation
- Protonate 3D
- Partial Charges
- Energy Minimize
- Site Finder
- Surfaces and Maps
- Ligand Interactions
- Conformations (e.g. Conformational Search, Dihedral Energy Plot)
- Biopolymer (e.g. Residue pKa, Align, Superpose, Homology Modeling)
- Simulations (e.g. Dock, SCF Calculations, Dynamics, etc.)
Ligand Interaction

LIE

3D
Stereoview
You should be able to

- create a simple plot of (x,y) pairs of data and properly label the plot using $\mathbb{R}$.
- describe the basic features of VMD and know how to use them.
- describe the basic features of MOE