# Computational Biology: Assignment #4

Due on Monday, Mar 31, 2014

*Jianyang Zeng 1:30pm*

**Weiyi Chen**

# Problem 1

(All of my codes are attached in ps4compbio.py. In this assignment, I would just point out the output and related section of code for each problem.) Following is my related code (python) to extract all pairs of distance restraints between ?CA? atoms when their distance is within 6A.

```python
# read file and save info at following lists
ls_no = []
ls_x = []
ls_y = []
ls_z = []
file = open('1ubq.pdb', 'r')
s_line = file.readline()
while (s_line != 'END\r\n'):
       ls_words = s_line.split()
       if ls_words[2] == 'CA':
              ls_no.append(int(ls_words[1]))
              ls_x.append(float(ls_words[5]))
              ls_y.append(float(ls_words[6]))
              ls_z.append(float(ls_words[7]))
       s_line = file.readline()

# construct dataframe to save position info
d_atom = {'x':pd.Series(ls_x,index=ls_no),
         'y':pd.Series(ls_y,index=ls_no),
         'z':pd.Series(ls_z,index=ls_no)}
df_atom = pd.DataFrame(d_atom)
i_len_atom = len(df_atom.index)

# search pairs with distance within 6A, and delete atoms never in a pair
df_atom['legal'] = False
for i, i_no in enumerate(df_atom.index):
       for j, j_no in enumerate(df_atom.index[i+1:]):
              f_dis_sqr = (df_atom['x'][i_no]-df_atom['x'][j_no])**2
              + (df_atom['y'][i_no]-df_atom['y'][j_no])**2
              + (df_atom['z'][i_no]-df_atom['z'][j_no])**2
              if f_dis_sqr <= 36:
                     df_atom['legal'][i_no] = True
                     df_atom['legal'][j_no] = True
print df_atom
```

The output sample is as follows. I have only copied the first ten lines.

```
          x      y       z legal
3     5.324  0.258 -12.519 True
23    3.644  3.286 -10.943 True
43    0.805  2.798  -8.472 True
65   -1.161  5.381  -6.460 True
89   -1.851  5.184  -2.711 True
108  -4.824  7.381  -1.671 True
135  -4.918  8.362   2.011 True
150  -7.964  9.499   4.039 True
173  -6.779 13.128   4.038 True
188  -6.998 13.508   0.290 True
```

Problem 1 continued on next page. . .

Since all atoms are labeled with 'True', so I would comment the code to search pairs within 6A, to save time in latter problems.

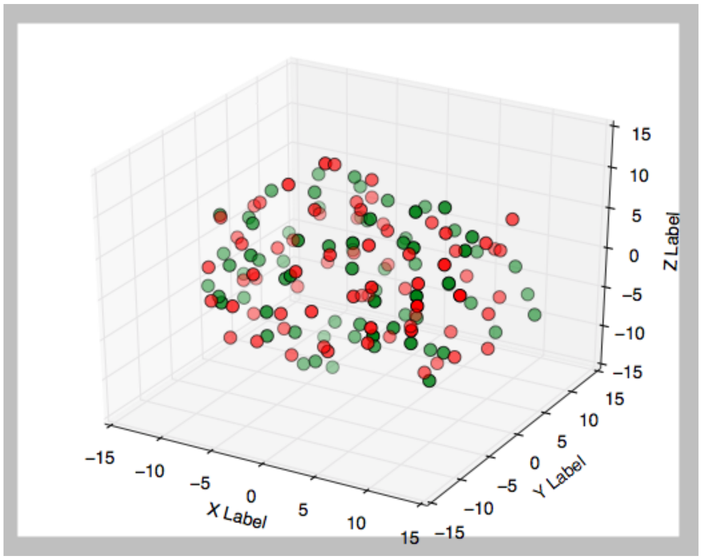
## Problem 2

I apply python MDS library to re-construct the 3D coordinates of corresponding "CA" atoms extracted from problem 1.

```
X_true = df_atom.values
n_samples = len(X_true)
# Center the data
X_true -= X_true.mean()

similarities = euclidean_distances(X_true)

seed = np.random.RandomState(seed=3)
mds = manifold.MDS(n_components=3, max_iter=3000, eps=1e-9, random_state=seed,
                   dissimilarity="precomputed", n_jobs=1)
pos = mds.fit(similarities).embedding_

# Rescale the data
pos *= np.sqrt((X_true ** 2).sum()) / np.sqrt((pos ** 2).sum())

# Rotate the data
clf = PCA(n_components=3)
X_true = clf.fit_transform(X_true)
pos = clf.fit_transform(pos)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

ax.scatter(X_true[:, 0], X_true[:, 1], X_true[:, 2], c='r', s=n_samples)
ax.scatter(pos[:, 0], pos[:, 1], pos[:, 2], c='g', s=n_samples)

ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')

plt.show()
```

It displays a graph, which draws both original positions and positions generated from MDS.

Problem 2 continued on next page. . .



where the red points indicate original position and green ones as MDS's.

## Problem 3

To calculate RMSD, according to its equation

$$RMSD(v,w) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((v_{ix}-w_{ix})^2 + (v_{iy}-w_{iy})^2 + (v_{iz}-w_{iz})^2)}$$

In code,

```
# Calculate RMSD
RMSD = np.sqrt(((X_true - pos)**2).sum() / n_samples)
print RMSD
```

The output value of RMSD is $RMSD = 15.876$

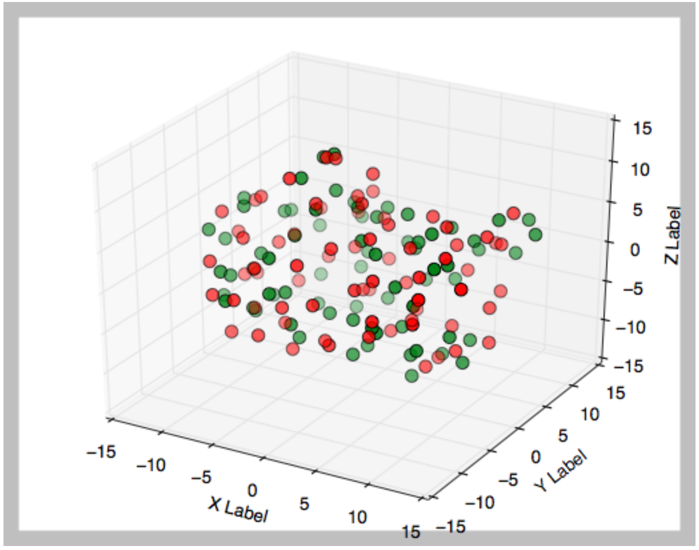
## Problem 4

To add noise, the code is as follows.

```
# Add noise to the similarities
noise = np.random.rand(n_samples, n_samples)
noise = noise + noise.T
noise[np.arange(noise.shape[0]), np.arange(noise.shape[0])] = 0
similarities += noise
```

Problem 4 continued on next page. . .

After that we follow the same process to apply MDS and calculate RMSD. The generated graph is as follows.



$$RMSD = 11.6003$$

(I ran it several times with same seed, the value is always around this value.) The red points indicate original positions of atoms, the green points indicate the generated positions from MDS, given their original difference.