1 Notes on what I might say

When I say distance, I mean a measure of the weighted euclidean distance between the vectorized contact maps of the proteins. When I say that a file is not critical to the program, I mean that comments may be applied to Cluster.cpp and the program will compile and run without problems. Removing all of the non-critical files will reduce the program to one that outputs contact maps of proteins instead of one that clusters them as well. I may indicate that it is not critical to the program but more effort than a simple comment is required to proceed. A simple result that is outputted by an algorithm is legible to the human and makes instant sense. Ability to visually compare is not yet provided.

2 Aurguments

```
-A|a          Runs Cascade, K-means, Hierarchical, Cutoff, Distance, and
              Partition algorithms and sets the number of clusters to be found
              to 0. Overrides the -O|o option, unless Cascade does not find
              enough clusters.
-S|s          Runs the Cascade algorithm and sets the number of clusters to be
              found to 0. Overrides the -O|o option. If Cascade does not find a
              sufficient number of clusters, -O|o will be defaulted to.
-K|k          Runs the K-means algorithm.
-T|t[|a|s|m|c|A|S|M|C] Runs the Hierarchical algorithm. It may be immediately
              followed by a letter to indicate the inter-cluster measure;
              Average, Minimum, Maximum, and centroid in the order presented.
-U|u          Runs the Cutoff algorithm.
-D|d          Runs the Distance algorithm.
-P|p          Runs the Partition algorithm.
-O|o[Number]  Does not run the Cascade algorithm. May be followed by a number.
              That number is the number of clusters the clustering algorithms
              will look for. If no number is given, the program will prompt for
              it. Overrides -S|s and part of -A|a.
-I|i[Number]  Sets the number of iterations that K-means and Cascade uses.
              Cascade will use four times that number. If no number is issued,
              -I|i will be ignored.
*.pdb         The Protein Database File used to pull structures from.
```

3 Files and system requirements

Cluster.cpp - The main program. Critical to the functioning of the program. Contains a global ”constant” on line 27 called Length. Changing it will change the length of protein that the program was designed for. As published it is set to 73, or 73 residues long.
Contact.h - Finds the contact maps of the proteins. The protein may be rotated or translated in any way possible from the origin and the contact map will remain the same. Output the contact

maps of the proteins in the file series CMP#.txt. Also metricizes the contact maps by removing all entries below the diagonal, lining whats left of the row in a single row, and then stacking them into a single matrix. It uses the α-carbon in each residue as the location of the residue. Critical to the functioning of the program.
Protein.h - Header file for Protein.cpp. Critical to the functioning of Contact.h and Protein.cpp. It provides for a dynamically allocated linked list of amino acids. Designed for proteins of arbitrary length. Each residue has the option to include a linked list to alternative locations and handles it by averaging over the alternatives.
Protein.cpp - Implementation for Protein.h. Critical to the functioning of Contact.h
Cascade.h - Runs the k-means clustering algorithm for every possible number of clusters. It then examines the function of squared errors returned from that and estimates the number of clusters based on this. It is possible for too many iterations to cause it to guess low by 1. It can also display the results for feedback from the user. The results are placed in Cascade.txt. Not required to run the program. Modification is required to run the ”All Algorithms” option without it.
Read.r - Reads the results from Cascade.h, makes a graph, and places it in Cascade.png. No required to compile program.
Kmeans.h - Runs the k-means clustering algorithm. It is looking for approximately round clusters. It will find the number of clusters specified. It places the simple results in ClusterK.txt. It also produces a file that contains the metricized cluster average position and places that in CMPK.txt. Not required to run the program.
Tree.h - Runs the hierarchical clustering algorithm. It is looking for approximately round clusters. Provides the option to measure the distance between clusters by the two structures most different from each other, the two structures most similar to each other, the average distance between all structures in the two clusters, or the distance between the centroid of the clusters. It will attempt to find the number of clusters specified where each cluster has two or more structures assigned to it. All other structures which are too dissimilar to be in such a cluster are discarded. If it finds that this is impossible, it will include clusters of one. It places the simple results in ClusterT.txt. It places the metricized cluster average position in CMPT.txt. It also displays the hierarchal connectivity tree in Tree.txt. The nodes of the tree are the distances between the children of the node and the leaves are the numbers of the structures. Not critical to running the program.
Cutoff.h - Runs the cutoff clustering algorithm. The algorithm itself is written in such a way that only dynamic allocations would need to be rewritten to fit well in cluster.c and cluster.h. It finds the first mode of distances between all structures. As mode is generally ill-defined in this case, it uses the first peak density averaged over 2.5% of the number of distance relations for that mode. It then clusters all structures together which are closer than the cutoff distance. This generally results in irregularly shaped and larger clusters. It places the simple results in ClusterC.txt and the metricized cluster average position in CMPC.txt. Not critical to running the program.
Dist.h - Finds the distances between all of the structures. The distance is the weighted squared euclidean distance with common values of 30 given weight 0 and everything else weight 1. This distance measure is the standard used throughout the program. It places the lower triangle of distance matrix in Distance.txt where the first distance given is the distance between the first and second structures. Not critical to running the program.
cluster.h - The header file for cluster.c. It is required for cluster.c, Cascade.h, Kmeans.h, Tree.h, Cutoff.h, and Dist.h.
cluster.c - Provides the implamentation for cluster.h and is required for Cascade.h, Kmeans.h,

Tree.h, Cutoff.h, and Dist.h.
Part.h - It takes CMPT.txt, CMPK.txt, and CMPC.txt and partitions them up into the contact maps of the centroid of the clusters. The contact maps are placed in the series CMPT#.txt, CMPK#.txt, and CMPC#.txt. It presently produces one extra file filled with nothing but zeros, but triggering the EOF on the source files before the creation of the file is difficult. It does not require any or all of the clustering algorithms to have been run. It is not critical to the program.
omp.h - Implementation of OpenMP. It is critical to Cascade.h and Part.h, but not difficult to remove the dependency. It is included with g++, other compilers may not be so forgiving. If it is missing, errors should be generated in other header files.
R - Required for Read.r to work.
Eye of Gnome (eog) - Required to display Cascade.png. Other *.png viewers may be specified but will need to be substituted in the code on line 347 of Cluster.cpp.

4 Functions

Cluster.cpp - All functions are wrapper functions for subordinate functions.
Contact.h - void Input(Protein&) takes the input file and builds the linked list of the protein. void Contact(Protein&, char[], int) builds the contact maps of the proteins and metricizes them.
Protein.h - An otherwise unremarkable object header.
Protein.cpp - double Contact(int, int) finds the euclidean distance between two residues. If the distance is greater than 30, it return 30.
Cascade.h - int Recommend(double[], int) recommends the number of clusters. It takes the second finite difference derivative by two forward first order approximations finite differences. It then finds the average concavity of last tenth of the positions and then finds where that concavity happens for the last time. The idea is that the concavity is approximately zero and that is when increasing the number of clusters is not helpful. void Diff(double[], double[], int) takes the first array and find the first order forward finite difference derivative and places it in the second array.
Read.r - Nothing remarkable.
Kmeans.h - The work horse is not here.
Tree.h - The work horse is not here. void TreeOut(Node*, int, int) recursively builds Tree.txt.
Cutoff.h - void Quicksort(double[], int, int), int Pivot(double[], int, int), void Swap(double&, double&) need to be used together. It is a quick sort algorithm built by Bruce Bolden of the CS department. Quicksort just needs to know the start and end of the array. double Mode(int, double**) takes the number of elements to be clustered and the distances between the proteins and it returns the first peak smoothed to 2.5% of the number of distances between proteins. void Mop(int[], bool**, int, int) takes the graph contact matrix in bool** and unifies the ClusterID in the lowest numbered cluster. It returns which clusters are occupied by the diagonal of bool**. int CutoffCluster(int, int, double**, int**, double[], int, char, double**, int*, double&) returns the number of clusters found, the cutoff value used, and the the cluster assignments. CutoffCluster(nrows, ncols, data, mask, weight, transpose, distance, distancematrix, clusterid, cutoff) is the function call. It approximately conforms to the standards provided in cluster.h and cluster.c.
Dist.h - The work horse is not here.
cluster.h & **cluster.c** - Documentation is mostly provided by Michiel de Hoon, Seiya Imoto, Satoru Miyano. I made a few modifications indicated else where in the documentation. http://bonsai.hgc.jp/ mde-

hoon/software/cluster/software.htm.
Part.h - void Part(char[], char) needs the name of a file to place the contact map and which map it is building.

5 Compilation

I compile by issuing the command:

```
g++ Cluster.cpp Protein.cpp cluster.c -fopenmp -O3 -o Cluster.out
```

-fopenmp is the linking for OpenMP. With the exception of Part.h and Cascade.h, this compile option is not required. -O3 is the maximum optimization option. It will optimize the code as much as possible. I have found that is will optimize out variable passing if it determines that a segmentation fault is going to occur in the function, which is annoying for debugging them. -O3 allows the program to take 114.1 KB on the hard drive instead of 114.2 KB. That may not sound impressive but the speed is improved by a factor of 7.

6 Data

The data I used was 1000 randomly generated p53TAD proteins provided by Stepan. I then cut it up and repeated it as I saw fit. I used at one time or another the first 3, the first 10 repeated 5 times, the first 20, the first 20 repeated 3; 10; or 50 times, the first 50, the first 100, the first 200, or all of them.