Statistical Clustering for FMRI 3D Datasets

B. Douglas Ward
Biophysics Research Institute
Medical College of Wisconsin
email: ward@post.its.mcw.edu
Initial Release: October 8, 1999

September 6, 2000

1 Program 3dStatClust

1.1 Purpose

Program 3dStatClust was developed to perform clustering of user specified parameters. Programs such as 3dNLfim and 3dDeconvolve calculate multiple parameters corresponding to the measured time series for each voxel in a 3D+time dataset. The purpose of program 3dStatClust is to provide a convenient method for clustering voxels based upon an arbitrary set of parameters.

One way that program 3dStatClust differs from programs 3dmerge and 3dclust is that the later use the spatial coordinates of the voxels, exclusively, as the criterion for clustering, whereas program 3dStatClust allows clustering on any set of parameters. Other differences are described below.

Section 1.2 discusses the theory underlying statistical clustering. This section is a very brief summary of material that can be found in reference [1]. Section 1.3 explains the batch commands necessary to run program 3dStatClust, and what the various command line options do. Section 1.4 contains examples illustrating the use of program 3dStatClust.

1.2 Theory

1.2.1 Agglomerative Hierarchical Clustering

There are many algorithms for clustering. The algorithm used by program 3dStatClust is referred to as "agglomerative hierarchical". At the beginning, that is, at the lowest level of the cluster hierarchy, every voxel is its own cluster. In other words, every cluster contains one and only one voxel.

At each step in the clustering process, the distance between each cluster and every other cluster is calculated. The two clusters which are closest together are merged into a single cluster. The merged cluster inherits the voxels from the two parent clusters. This process is repeated until, finally, only a single cluster remains; and that cluster contains every voxel.

It is not necessary to specify the number of clusters in advance. Program 3dStatClust stores the results for the top n levels of the cluster hierarchy, where n is specified by the user.

1.2.2 Distance Calculations

As described above, the clustering algorithm uses the distance between clusters for determining which clusters should be merged. However, we have not yet defined what we mean by "distance between clusters".

The spatial clustering programs (3dmerge and 3dclust) use "nearest neighbor" or "minimum distance" for determining if a particular voxel should be added to a cluster. If the distance between that voxel and any voxel in the cluster is less than the specified distance rmm, then the voxel is added to the cluster. Note that it is possible, under this method, for voxels that are very far apart in the physical space to be members of the same cluster. This phenomenon is referred to as "chaining".

Long, threadlike chains may, or may not, correspond to functionally connected voxels in reality. However, since the results of spatial clustering may be visually inspected by the user, the user can guard against absurd results. Note that this is *not* the case with statistical clustering. Statistical clustering occurs in the parameter space, which *cannot* be visually inspected by the user. Any long chains which may develop in the parameter space are not readily apparent to the user. Voxels which are far apart in the parameter space, but which are linked together in the same cluster, may correspond to completely different functional behaviors.

For this reason, program 3dStatClust does *not* use nearest neighbor clustering. Instead, program 3dStatClust calculates the centroid of the parameter vectors of the voxels belonging to each cluster. Two different clusters are merged into a single cluster only if their centroids are close together. This inhibits the formation of chains.

Therefore, the distance between clusters will be defined as the distance between the parameter centroids of the clusters. However, it remains to be determined how distance itself is defined. Program 3dStatClust gives the user 3 options for defining distance: (1) Euclidean distance, (2) statistical distance for independent variables, and (3) statistical distance for correlated variables. These options are defined below.

Remark: Each of the following definitions for distance (not to mention many other possible definitions) is equally valid. Which definition turns out to be the most useful will depend on the particular situation.

Euclidean Distance

Suppose that $\mathbf{X} = (X_1, X_2, \ldots, X_p)^t$ and $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_p)^t$ are two parameter vectors in a $p-$ dimensional parameter space. Then the Euclidean distance between these parameters is defined:

$$d(\mathbf{X}, \mathbf{Y}) = \sqrt{(X_1 - Y_1)^2 + (X_2 - Y_2)^2 + \cdots + (X_p - Y_p)^2}$$

Note that this distance calculation is appropriate only if the p individual parameters are commensurate. This is because the above formula assigns equal weight to each of the parameters.

Statistical Distance for Independent Variables

It may be the case that the individual parameters are not commensurate. This would happen, for example, if one parameter takes on values between 0.01 and 0.02, while another parameter assumes values between -1000 and +1000. Obviously, if the above formula were to be used, the first parameter would essentially be ignored. Therefore, it would be appropriate to apply separate scaling factors to the individual parameters. The user can accomplish this by manually rescaling the individual sub-bricks (using program 3dcalc, for example). Another approach is to have program 3dStatClust perform the scaling internally. In this case, the distance formula becomes:

$$d(\mathbf{X}, \mathbf{Y}) = \sqrt{\frac{(X_1 - Y_1)^2}{s_{11}} + \frac{(X_2 - Y_2)^2}{s_{22}} + \cdots + \frac{(X_p - Y_p)^2}{s_{pp}}}$$

where the scaling factors s_{ii}, $i = 1, \ldots, p$, are calculated as the sample variances for the corresponding parameters:

$$s_{ii} = \frac{1}{N-1} \left[\sum_{k=1}^{N} \left(X_{ik} - \overline{X}_i \right)^2 \right]$$

where the summation is over all voxels which lie above the cutoff threshold.

As is readily apparent, this method assigns less weight to the difference in a particular parameter that has high inherent variability, and assigns more weight to the difference in a parameter which has low inherent variability.

Statistical Distance for Correlated Variables

The approach described above treats the parameters as independent variables. However, this method can be extended to deal with correlated variables. In additional to the sample variances, it is necessary to calculate the covariances between the parameters. The sample covariance matrix for p parameters is defined:

$$\mathbf{S} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & s_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ s_{p1} & s_{p2} & \cdots & s_{pp} \end{bmatrix}$$

where

$$s_{ij} = \frac{1}{N-1} \left[\sum_{k=1}^{N} \left(X_{ik} - \overline{X}_i \right) \left(X_{jk} - \overline{X}_j \right) \right]$$

The statistical distance between parameter vectors $\mathbf{X}$ and $\mathbf{Y}$ is then defined:

$$\begin{aligned} d(\mathbf{X}, \mathbf{Y}) &= \sqrt{(\mathbf{X} - \mathbf{Y})^t \, \mathbf{S}^{-1} \, (\mathbf{X} - \mathbf{Y})} \\ &= \sqrt{\left[\mathbf{S}^{-\frac{1}{2}} (\mathbf{X} - \mathbf{Y}) \right]^t \left[\mathbf{S}^{-\frac{1}{2}} (\mathbf{X} - \mathbf{Y}) \right]} \end{aligned}$$

If we further define

$$\begin{aligned} \mathbf{Z}_X &= \mathbf{S}^{-\frac{1}{2}}\mathbf{X} \\ \mathbf{Z}_Y &= \mathbf{S}^{-\frac{1}{2}}\mathbf{Y} \end{aligned}$$

then the statistical distance between $\mathbf{X}$ and $\mathbf{Y}$ is given by:

$$\begin{aligned} d(\mathbf{X},\mathbf{Y}) &= \sqrt{(\mathbf{Z}_X - \mathbf{Z}_Y)^t\,(\mathbf{Z}_X - \mathbf{Z}_Y)} \\ &= \text{Euclidean distance between } \mathbf{Z}_X \text{ and } \mathbf{Z}_Y \end{aligned}$$

1.3 Usage

1.3.1 Syntax

The syntax for execution of program 3dStatClust is as follows:

3dStatClust [-prefix pname] [-session dir] [-verb]
[-dist_euc | -dist_ind | -dist_cor] -thresh t tname [-nclust n]
filename[brick(s)] [filename[brick(s)]] ... [filename[brick(s)]]

The different command line options are explained below.

1.3.2 Options

-prefix *pname*
or

-output *pname*
Use *pname* for the output dataset prefix name. The default is *pname* = SC.

-session *dir*
Use *dir* for the output dataset session directory. The default is *dir* = ./ = current working directory.

-verb
The optional -verb command is used to print out additional output as the program proceeds.

The following alternative (mutually exclusive) options determine how distance is calculated between parameter vectors:

-dist_euc	=	Calculate Euclidean distance between parameter vectors (Default)
-dist_ind	=	Calculate statistical distance for independent parameter vectors
-dist_cor	=	Calculate statistical distance for correlated parameter vectors

-thresh t ***tname*** Use threshold statistic from file *tname*. Only voxels whose threshold statistic is greater than t in absolute value will be considered. (If file *tname* contains more than one sub-brick, the threshold statistic sub-brick must be specified!)

-nclust n The -nclust command is used to create a single *AFNI* "bucket" type dataset having n sub-bricks. These n sub-bricks correspond to the top n levels of the agglomerative clustering hierarchy. The output is written to the file with the user specified prefix filename. Each of the individual sub-bricks can then be accessed for display within program *afni*.

Sub-brick	Contents
#0	1 cluster
#1	2 clusters
#2	3 clusters
⋮	⋮
#n-1	n clusters

Command line arguments after the above are taken as parameter datasets. A dataset is specified using one of these forms:

prefix+view   prefix+view.HEAD   prefix+view.BRIK

1.3.3 Sub-brick selection

You can also add a sub-brick selection list after the end of the dataset name. This allows only a subset of the sub-bricks to be used for clustering (by default, all of the input dataset sub-bricks are used for clustering). A sub-brick selection list looks like one of the following forms:

fred+orig[5]	==>	use only sub-brick #5
fred+orig[5,9,12]	==>	use #5, #9, and #12
fred+orig[5..8] or [5-8]	==>	use #5, #6, #7, and #8
fred+orig[5..13(2)] or [5-13(2)]	==>	use #5, #7, #9, #11, and #13

Sub-brick indexes start at 0. You can use the character '$' to indicate the last sub-brick in a dataset; for example, you can select every third sub-brick by using the selection list:

fred+orig[0..$(3)]

The '$', '(', ')', '[', and ']' characters are special to the shell, so you will have to escape them. This is most easily done by putting the entire dataset plus selection list inside single quotes, as in 'fred+orig[5..7,9]'.

1.3.4 Notes

- Program 3dStatClust execution time is approximately proportional to the square of the number of voxels above the threshold. Therefore, the user should take care in specifying the cutoff threshold. As a practical matter, clustering of more than about 10000 voxels is *not* recommended.

- There is a practical limit to the number of different clusters that can be displayed. The color coding scale provided by *afni* has only a finite number of values, and some of the colors are difficult to distinguish. Suggestions: Be sure to click on "pos", since cluster identification numbers are always positive. My preference is to set the number of colors to 16, and to display no more than 8 or 10 clusters.

- Within each sub-brick, the numbering of the different clusters corresponds to the relative sizes of the clusters, i.e., the largest cluster has index 1, the next largest has index 2, etc.

- Since the output dataset generated by program 3dStatClust consists primarily of 0's, this dataset can be compressed to a small fraction of its original size.

1.4 Examples

Example 1. Euclidean distance

Suppose that 3D+time dataset Larry+orig has been analyzed using program 3dDeconvolve. There were 3 input stimulus functions, corresponding to the 3 experimental conditions. The system impulse response function (IRF) was estimated individually for each of these 3 conditions, for time lags 0,1,...,6 (TR), and these IRF's were written to 3D+time datasets Larry.cond1.iresp+orig, etc. Also, the multiple regression coefficients and statistics were written to the bucket dataset Larry.buck+orig. Suppose that the F-statistic for significance of the multiple regression is contained in sub-brick #53 of file Larry.buck+orig, and that this statistic will be used for the voxel threshold. The objective is to form clusters of voxels having similar IRF's. This may be accomplished using the following script:

Batch Command File for Example 1

```
3dStatClust \
-verb \
-dist_euc \
-nclust 15 \
-prefix Larry.statclust \
-thresh 5.0 'Larry.buck+orig[53]' \
Larry.cond1.iresp+orig \
Larry.cond2.iresp+orig \
Larry.cond3.iresp+orig
```

■

The command -verb is used to write additional output to the terminal during program execution. Since we are comparing IRF's, and since the IRF coefficients are commensurate (they have approximately the same range of values), the -dist_euc command indicates that Euclidean distance is to be used for distance calculations. The command -nclust 15 indicates that the output is to consist of a 15 sub-brick dataset, with the sub-bricks corresponding to the top 15 levels of the hierarchical clustering (e.g., sub-brick #5 will correspond to level #6 of the hierarchical clustering, i.e., sub-brick #5 will contain 6 clusters).

The command -prefix Larry.statclust indicates that this 15 sub-brick dataset is to be written to file Larry.statclust+orig.

The command -thresh 5.0 'Larry.buck+orig[53]' is used to specify that sub-brick #53 from dataset Larry.buck+orig is to be used as the threshold statistic; only voxels whose threshold statistic is $\geq$ 5.0 will be used in the clustering. The last three lines of the batch command file specify the 3 IRF time series datasets. Since no sub-brick selectors are appended to the file names, *every* sub-brick is used as a parameter. Each IRF time series consists of 7 points, hence the total number of parameters is $3 \times 7 = 21$. Therefore, clustering occurs in a 21 dimensional parameter space.

Example 2. Statistical distance for independent variables

Here we consider an example where the cluster parameters are not commensurate. Suppose that drug response data is contained in the 3D+time dataset Curly+orig. Program 3dNLfim is used to analyze this data, using the differential-exponential drug response model. Therefore, the full model would be:

$$Y_i = \gamma_0 + \gamma_1 t_i + k\left[e^{-\alpha_1(t_i-t_0)} - e^{-\alpha_2(t_i-t_0)}\right]u(t_i-t_0) + \varepsilon_i$$

where

Y_i = measured time series data ($i = 1, \ldots, n$);
γ_0 = constant offset term;
γ_1 = coefficient of linear trend;
k = multiplicative constant;
t_0 = time delay;
α_1 = elimination rate constant;
α_2 = absorption rate constant;
ε_i = Gaussian noise, i.i.d. $N(0, \sigma^2)$;
and $u(t)$ is the unit step function:

$$u(t) = \begin{cases} 0, & \text{for } t < 0, \\ 1, & \text{for } t \geq 0. \end{cases}$$

The signal parameter estimates are constrained to fall within the following limits:

$$\begin{array}{rcccl} 45.0 & \leq & t_0 & \leq & 75.0 \\ -500.0 & \leq & k & \leq & 500.0 \\ 0.00 & \leq & \alpha_1 & \leq & 0.15 \\ 0.15 & \leq & \alpha_2 & \leq & 0.50 \end{array}$$

The bucket dataset Curly.buck+orig, which was generated by program 3dNLfim, has the following structure:

Brick	Label	Contents
#0	constant	$g_n[0]$ = non-linear L.S. est. of γ_0
#1	linear	$g_n[1]$ = non-linear L.S. est. of γ_1
#2	t0	$g_s[0]$ = non-linear L.S. est. of t_0
#3	k	$g_s[1]$ = non-linear L.S. est. of k
#4	alpha1	$g_s[2]$ = non-linear L.S. est. of α_1
#5	alpha2	$g_s[3]$ = non-linear L.S. est. of α_2
#6	Signal TMax	Time of signed maximum of signal
#7	Signal SMax	Signed maximum of signal
#8	Signal % SMax	Signed maximum of signal as a % of baseline
#9	Signal Area	Area under signal (always positive)
#10	Signal % Area	(Signed) area under signal as a % of baseline
#11	F-stat Regression	$F^{**} = \dfrac{MS(\text{Reg})}{MS(\text{Error})}$

Now, suppose that one wishes to form clusters based on parameters t_0, α_1, and α_2. It is obvious, from the above table, that these parameters are not commensurate. Since the range of variation for parameter t_0 is much larger than that for parameters α_1 and α_2, parameter t_0 would dominate the clustering if the parameters are not scaled. In order to scale the parameters, the -dist_ind command is used, as indicated below.

Batch Command File for Example 2

```
3dStatClust  \
-verb \
-dist_ind \
-nclust 20 \
-prefix Curly.statclust \
-thresh 50.0 'Curly.buck+orig[11]' \
'Curly.buck+orig[2]' \
'Curly.buck+orig[4]' \
'Curly.buck+orig[5]'
```

■

Note that the -thresh command refers to sub-brick #11 (the F-statistic sub-brick). Also, note that the clustering parameters are taken from sub-bricks #2, #4, and #5 (the t_0, α_1, and α_2 sub-bricks, respectively).

Example 3. Spatial Clustering

Yes, it is possible to do spatial clustering with program 3dStatClust. However, this requires entering the spatial coordinates for each voxel as the parameters for clustering.

This is easily accomplished using the output from program 3dcalc. Suppose that spatial clustering is to be performed on file Moe.bucket+orig, and that this file also contains the statistic to be used for thresholding. Spatial clustering can be accomplished as follows:

Batch Command File for Example 3

```
3dcalc  -a 'Moe.bucket+orig[0]'  -expr ''x''  -fscale  -prefix Moe.xcoor
3dcalc  -a 'Moe.bucket+orig[0]'  -expr ''y''  -fscale  -prefix Moe.ycoor
3dcalc  -a 'Moe.bucket+orig[0]'  -expr ''z''  -fscale  -prefix Moe.zcoor

3dStatClust \
-verb  -dist_euc  -nclust 10  -prefix Moe.StatClust \
-thresh 5.0 'Moe.bucket+orig[28]' \
Moe.xcoor+orig  Moe.ycoor+orig  Moe.zcoor+orig
```

■

First, three invocations of program 3dcalc are used to create files Moe.xcoor+orig, Moe.ycoor+orig, and Moe.zcoor+orig. These single sub-brick files contain the x-, y-, and z-coordinates, respectively, for every voxel in the dataset. Now, when these same files are input as parameter sub-bricks into program 3dStatClust, that program will form clusters based upon the spatial coordinates of the voxels which lie above the threshold. In this example, the threshold is specified by sub-brick #28 of Moe.bucket+orig. A truncated listing of the screen output follows:

Program 3dStatClust Screen Output from Example 3

```
 Program:          3dStatClust
 Author:           B. Douglas Ward
 Initial Release:  08 October 1999
 Latest Revision:  05 September 2000

Reading threshold dataset:   Moe.bucket+orig[28]
Number of voxels above threshold = 31
Reading parameter dataset:   Moe.xcoor+orig
Reading parameter #1
Reading parameter dataset:   Moe.ycoor+orig
Reading parameter #2
Reading parameter dataset:   Moe.zcoor+orig
Reading parameter #3
Number of parameters = 3
Output dataset will have 10 sub-bricks
```

⋮

etc.

⋮

```
Merging cluster #1 and cluster #4
Distance = 51.909966

# Clusters = 3

 Cluster #1
   # Voxels = 19
   Centroid:
```
$$\begin{array}{r} 1.6773 \\ 68.3882 \\ -1.4803 \end{array}$$

```
 Cluster #2
   # Voxels = 6
   Centroid:
```
$$\begin{array}{r} 33.7500 \\ 5.6250 \\ 1.2500 \end{array}$$

```
 Cluster #3
   # Voxels = 6
   Centroid:
```
$$\begin{array}{r} -44.3751 \\ 1.2497 \\ 1.8750 \end{array}$$

```
Merging cluster #1 and cluster #2
Distance = 70.536003

# Clusters = 2

 Cluster #1
   # Voxels = 25
   Centroid:
```
$$\begin{array}{r} 9.3747 \\ 53.3250 \\ -0.8250 \end{array}$$

```
 Cluster #2
   # Voxels = 6
   Centroid:
```
$$\begin{array}{r} -44.3751 \\ 1.2497 \\ 1.8750 \end{array}$$

```
Merging cluster #1 and cluster #2
Distance = 74.887718

# Clusters = 1

 Cluster #1
   # Voxels = 31
   Centroid:
```
$$\begin{array}{r} -1.0285 \\ 43.2459 \\ -0.3024 \end{array}$$

■

It is instructive to compare the results of spatial clustering using program 3dStatClust, with the results of spatial clustering using program 3dmerge. Remember that these programs use different algorithms for forming clusters.

1.5 References

1. R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis.* Englewood Cliffs, NJ: Prentice-Hall, Inc. (1982).