1 Program 3dFDR

1.1 Purpose

Program `3dFDR` implements the False Discovery Rate (FDR) algorithm for thresholding of voxelwise statistics. Instead of controlling for *alpha* (the probability of a false positive anywhere in the volume) the FDR algorithm controls for the proportion of false positives relative to the number of detections.

Program input consists of a dataset containing one (or more) statistical sub-bricks. Output consists of a bucket dataset with one sub-brick for each input sub-brick. For non-statistical input sub-bricks, the output sub-brick is a copy of the input. However, statistical input sub-bricks are replaced by the corresponding FDR values, as follows:

For each voxel, the minimum value of q is determined such that

$$E(FDR) \leq q$$

leads to rejection of the null hypothesis in that voxel. Only voxels inside the user specified mask (optional) will be considered. These q-values are then mapped to z-scores for compatibility with the *AFNI* statistical threshold display, i.e.,

$$\text{input stat} \Rightarrow p\text{-value} \Rightarrow \text{FDR } q\text{-value} \Rightarrow \text{FDR } z\text{-score}$$

These calculations are performed independently for each statistical sub-brick.

1.2 Theory

This section contains a very brief summary of the theory; for more details, please see References [1] and [2].

1.2.1 Definitions

Let N be the total number of voxels in the analysis (either the number of voxels in the entire volume, or the number of voxels in the user-specified mask). Each voxel can be assigned to one of four categories, depending on whether or not the voxel is declared active, and whether or not the voxel is truly active. This is depicted in the table below.

	Declared Inactive	Declared Active	
Truly Inactive	N_{ii}	N_{ia}	T_i
Truly Active	N_{ai}	N_{aa}	T_a
	D_i	D_a	N

The False Discovery Rate (FDR) is defined as follows:

$$FDR = \frac{N_{ia}}{D_a} = \frac{N_{ia}}{N_{aa} + N_{ia}}$$

That is, the FDR is the proportion of voxels declared active which are truly inactive (i.e., the number of false positives relative to all detections). The objective of the False Discovery Rate algorithm is to control the number of false positives, so that the average FDR (over many repetitions of the experiment) is no larger than the user specified value q ($0 < q < 1$), i.e.,

$$E(FDR) \leq q$$

This is achieved using the following algorithm.

1.2.2 The FDR Algorithm

For each voxel, the minimum value q is determined such that $E(FDR) \leq q$ leads to rejection of the null hypothesis in that voxel. If the user specifies a mask, only voxels inside the mask will be considered. For compatibility with the *AFNI* statistical threshold display, the q-values are then converted to z-scores. The algorithm is outlined below.

The FDR Algorithm

For all voxels ($i = 1, ..., N$)
    Convert the voxel statistic s_i to its corresponding p-value p_i

Sort the voxel p-values: $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(N-1)} \leq p_{(N)}$

$q_{\min} = 1.0$
For $i = N$ downto 1
    $q = c(N) \cdot p_{(i)} \cdot \left(\frac{N}{i}\right)$
    if $q > q_{\min}$ then $q_{(i)} = q_{\min}$
        else $q_{(i)} = q_{\min} = q$

For $i = 1$ to N
    Convert $q_{(i)}$ to z-score $z_{(i)}$

For $i = 1$ to N
    Replace voxel statistic s_i with FDR z-score z_i

■

In the above algorithm, the constant $c(N)$ depends on the assumption about the distribution of p-values across voxels. Specifically, if it is assumed that p-values are independent across voxels, then $c(N) = 1$. For an arbitrary distribution of p-values, $c(N) = \sum_{i=1}^{N} \frac{1}{i}$ (see Reference [2]).

1.3 Usage

1.3.1 Syntax

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

3dFDR [**-input** *fname* | **-input1D** *dname*] [**-mask_file** *mname*] [**-mask_thr** m_{thr}] [**-cind** | **-cdep**] [**-quiet**] [**-list**] **-prefix** *pname*

The different command line options are explained below.

1.3.2 Options

-input *fname*

Use file *fname* as the input 3d dataset. A dataset is specified using one of these forms:

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

-input1D *dname*

Read a column of p-values from the .1D file *dname*. The -input and -input1D options are mutually exclusive.

-mask_file *mname*

Use file *mname* as a mask dataset. Only voxels that are inside the mask will be considered. If file *mname* contains more than one sub-brick, the specific mask sub-brick must be specfied. (Default: no mask)

-mask_thr m_{thr}

Only voxels whose corresponding mask value is greater than, or equal to, m_{thr} in absolute value will be considered. (Default: $m_{thr} = 1.0$)

The constant $c(N)$ depends on the assumption about the p-values:

-cind

Set $c(N) = 1$. This assumes that p-values are independent across voxels.

-cdep

Set $c(N) = \sum_{i=1}^{N} \frac{1}{i}$. This applies for an arbitrary distribution of p-values.

Default: $c(N) = 1$.

-quiet

The optional -quiet command is used to suppress screen output as the program proceeds.

-list

Use this option to write sorted list of voxel FDR q-values and z-scores to the screen.

-prefix *pname*

or

-output *pname*

Use *pname* for the output dataset prefix name. Note: If this option is not used, then the only program output will be written to the screen.

1.3.3 Sub-brick selection

You can also add a sub-brick selection list after the end of the dataset name. 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]'.

1.4 Examples

Example 1. Input column listing of p-values

We consider the rt-PA treatment example from Ref. [1], where 15 hypotheses comparing two treatments are to be tested. The (ordered) p-values from the 15 comparisons are listed below:

0.0001	0.0004	0.0019	0.0095	0.0201	0.0278	0.0298	0.0344
0.0459	0.3240	0.4262	0.5719	0.6528	0.7590	1.0000	

Using the Bonferroni approach, for an overall $\alpha = 0.05$, we have:

$$p_{thr} = \frac{0.05}{15} = 0.0033.$$

Therefore, using the Bonferroni procedure, we would reject the null hypotheses corresponding to the 3 smallest p-values listed above.

Now, suppose that we wish to use the FDR controlling procedure with $q^* = 0.05$. First, save the above p-values as a single column of numbers in file rt-PA.1D. (Note that it is *not* necessary to sort the p-values when they are stored in the .1D file.) Now, use the following command line to calculate the FDR q-values:

Command Line for Example 1

```
3dFDR  -input1D rt-PA.1D  -list
```

■

Screen output for this example is depicted below.

Program 3dFDR Screen Output from Example 1

```
Program:          3dFDR
Author:           B. Douglas Ward
Initial Release:  25 January 2002
Latest Revision:  25 January 2002
```

Index	p-value	q-value	z-score
1	0.000100	0.001500	3.174684
2	0.000400	0.003000	2.967738
3	0.001900	0.009500	2.593516
4	0.009500	0.035625	2.101182
5	0.020100	0.060300	1.878594
6	0.027800	0.063857	1.853176
7	0.029800	0.063857	1.853176
8	0.034400	0.064500	1.848708
9	0.045900	0.076500	1.771365
10	0.324000	0.486000	0.696685
11	0.426200	0.581182	0.551659
12	0.571900	0.714875	0.365317
13	0.652800	0.753231	0.314382
14	0.759000	0.813214	0.236281
15	1.000000	1.000000	0.000000

■

In this example, we see that only the first 4 p-values satisfy:

$$p_{(i)} \leq \frac{i}{15}(0.05)$$

Therefore, controlling FDR at $q^* = 0.05$, we would reject the null hypotheses corresponding to the 4 smallest p-values.

Example 2. Input prefix+orig bucket dataset

In this example, we apply the FDR algorithm to a "bucket" dataset produced by program 3dDeconvolve. The contents of the input dataset are listed below:

Contents of Input Bucket Dataset: myData.bucket+orig

Brick #	Label	Contents
0	Base t^0 Coef	least squares est. of b_0(constant term)
1	Base t^1 Coef	least squares est. of b_1(linear trend)
2	Stim01[0] Coef	least squares est. of h_0 (impulse response at time lag 0)
⋮	⋮	⋮
12	Stim01[10] Coef	least squares est of h_{10} (impulse response at time lag 10)
13	Stim01 F-stat	F-statistic for significance of Stim01
14	Area LC[0]	Sum of IRF coefficients
15	Area F-stat	F-statistic for significance of Area
16	Full F-stat	F-statistic for significance of the overall regression

We see that 3 of the 17 sub-bricks contain F-statistics. To convert these F-statistics to FDR z-scores, use the following command line:

Program 3dFDR Command Line for Example 2

```
3dFDR \
-input myData.bucket+orig \
-mask_file 'myData.bucket+orig[0]' \
-mask_thr 1000.0 \
-cind \
-prefix myFDR.bucket
```

■

The new dataset, myFDR.bucket+orig, also contains 17 sub-bricks. The non-statistical sub-bricks are direct copies of the corresponding sub-bricks from myData.bucket+orig. However, each of the 3 F-statistic sub-bricks has been replaced with the corresponding FDR z-scores.

Contents of Output Bucket Dataset: myFDR.bucket+orig

Brick #	Label	Contents
0	Base t^0 Coef	least squares est. of b_0(constant term)
1	Base t^1 Coef	least squares est. of b_1(linear trend)
2	Stim01[0] Coef	least squares est. of h_0 (impulse response at time lag 0)
⋮	⋮	⋮
12	Stim01[10] Coef	least squares est of h_{10} (impulse response at time lag 10)
13	FDRz Stim01 F-stat	FDR z-score for significance of Stim01
14	Area LC[0]	Sum of IRF coefficients
15	FDRz Area F-stat	FDR z-score for significance of Area
16	FDRz Full F-stat	FDR z-score for significance of the overall regression

It is not necessary to duplicate the entire dataset. The following command line reads in just the 2 sub-bricks corresponding to the "Area" GLT, i.e., sub-bricks #14 and #15.

Program 3dFDR Command Line for Example 2

```
3dFDR \
-input 'myData.bucket+orig[14,15]' \
-mask_file 'myData.bucket+orig[0]' \
-mask_thr 1000.0 \
-cind \
-prefix myFDR.ftest
```

■

In this case, the output dataset contains just 2 sub-bricks: sub-brick #0 is a copy of sub-brick #14 (containing the sum of the IRF coefficients); sub-brick #1 contains the corresponding FDR z-scores.

Contents of Output Bucket Dataset: myFDR.ftest+orig

Brick #	Label	Contents
0	Area LC[0]	Sum of IRF coefficients
1	FDRz Area F-stat	FDR z-score for significance of Area

Example 3. Input prefix+tlrc fitt dataset

In this example, we consider an "fitt" Tlrc dataset. This dataset has 2 sub-bricks, as described below:

Contents of Input "fitt" Dataset: myTtest+tlrc

Brick #	Label	Contents
0	#0	difference in sample means
1	#1	t-statistic for significance of difference

To convert the t-stats to FDR z-scores, execute the following command line:

Program 3dFDR Command Line for Example 3

```
3dFDR \
-input myTtest+tlrc \
-mask_file myMask+tlrc \
-cdep \
-prefix myFDR.ttest
```

■

Note that, since the mask threshold is not specified, the default value ($m_{thr} = 1$) is used. Also, note the -cdep command, since the user does not wish to assume that the Tlrc voxels have independent p-values. The output 2 sub-brick bucket dataset is described below:

Contents of Output Bucket Dataset: myFDR.ttest+tlrc

Brick #	Label	Contents
0	#0	difference in sample means
1	FDRz #1	FDR z-score for significance of difference

1.5 References

[1] Benjamini, Y. and Hochberg, Y. (1995) Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, Series B, **57**, 289-300.

[2] Genovese, C.R., Lazar, N.A., and Nichols, T. Thresholding of Statistical Maps in Functional Neuroimaging Using the False Discovery Rate.