Introduction to the OCplus package

Alexander Ploner
Medical Epidemiology & Biostatistics
Karolinska Institutet, Stockholm
email: alexander.ploner@ki.se

April 17, 2024

1 Overview

The `OCplus` package offers a variety of tools for designing and analyzing gene expression microarray experiments. The common underlying statistical concept is the use of the false discovery rate (fdr) to identify differentially expressed (DE) genes.

A commonly underappreciated fact is that in a microarray setting, magical thresholds like 0.05 or 0.01 make even less sense for the fdr than they do for the traditional p-values. The trade-off between fdr and the ability to detect relevant genes can be made much more explicit than the classical trade-off between p-value and statistical power. A central idea of `OCplus` is to allow the user to make up her mind about the trade-off appropriate for her specific situation, based on the operating characteristics of her experimental design or data set (hence the name).

The main functionality of `OCplus` falls into three categories, listed below with their most important functions:

1. Sample size assessment: `TOC`, `samplesize`
2. Data analysis: `EOC`, `fdr1d`, `fdr2d`
3. Estimation of the proportion of non-DE genes: `tMixture`

This short introduction explains the underlying model and demonstrates the main functionality in each category; in-depth descriptions can be found in the individual vignettes.

2 Installation

You need the package `interp`, available from CRAN. In order to run `EOC`, you also need the package `multtest` from Bioconductor.

```
> library(OCplus)
```

3 Sample size calculations

3.1 samplesize

This function allows the user to choose an appropriate number of microarray chips per group for an assumed proportion of regulated genes with a minimum fold change. Specifically, the function calculates the global false discovery rate (FDR) among genes with the absolute largest t-statistics, assuming a given proportion `p0` of non-differentially expressed (nonDE) genes, and a given effect size `D` for the differentially expressed (DE)genes:

```
> ss1 = samplesize(p0=0.95, D=1, crit=0.01)
```

In the example above, we assume that 95% of all genes are nonDE, and that the 5% DE genes have a log2-fold change of `D`$= \pm 1$ (i.e. a fold change of 0.5 and 2, respectively); this produces the following result:

```
> ss1

       FDR_0.01     fdr_0.01
5  6.383062e-01 6.993132e-01
10 2.525227e-01 3.799915e-01
15 7.292893e-02 1.467379e-01
20 1.793387e-02 4.307385e-02
25 4.272392e-03 1.142484e-02
30 1.033049e-03 2.984716e-03
35 2.555867e-04 7.865114e-04
40 6.458286e-05 2.098671e-04
45 1.661413e-05 5.666724e-05
50 4.338704e-06 1.545971e-05
```

The listed FDRs are for the genes with 1% largest t-statistics (or equivalently, the 1% smallest p-values). We find that for $n = 5$ microarray chips per group, these genes have a FDR of 64%, meaning that roughly 2/3 of the top genes can be expected to be false positives; if we invest however in $n = 5$ microarray chips per group, less than 2% of the top genes will be false positives.

As a a side effect, `samplesize` produces a plot of the FDR as a function of the number of chips per group, as shown in Figure 1. It shows that there is little to gain by increasing group sizes beyond $n = 20$.

3.2 TOC

This function calculates the theoretical operating characteristics of a chosen design; for a given group size, proportion of regulated genes and minimum fold changes, the function shows the trade off between FDR and sensitivity for any possible threshold on the t-statistics.

```
> samplesize(p0=0.95, D=1, crit=0.01)
```

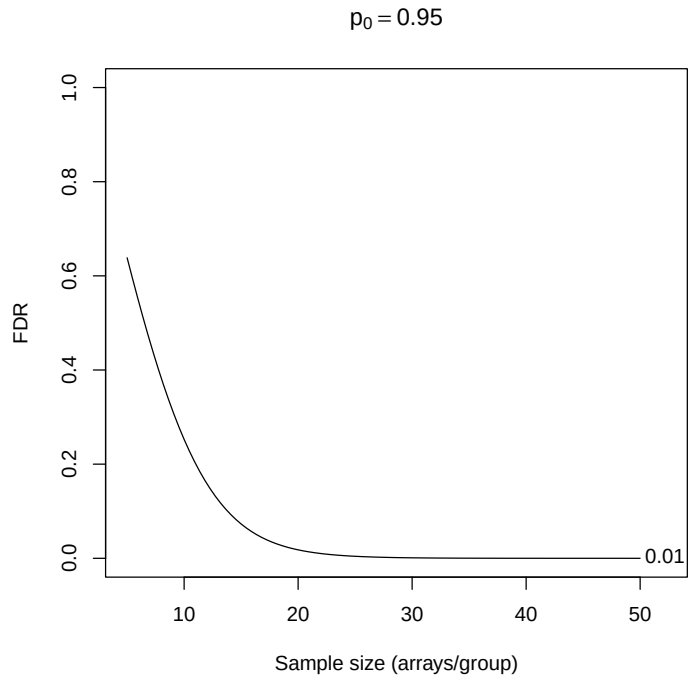

Figure 1: FDR as a function of samplesize, assuming that genes with 1% absolutely largest t-statistics are declared DE.

```
> TOC(n=20, p0=0.95, D=1, alpha=FALSE, legend=TRUE)
```

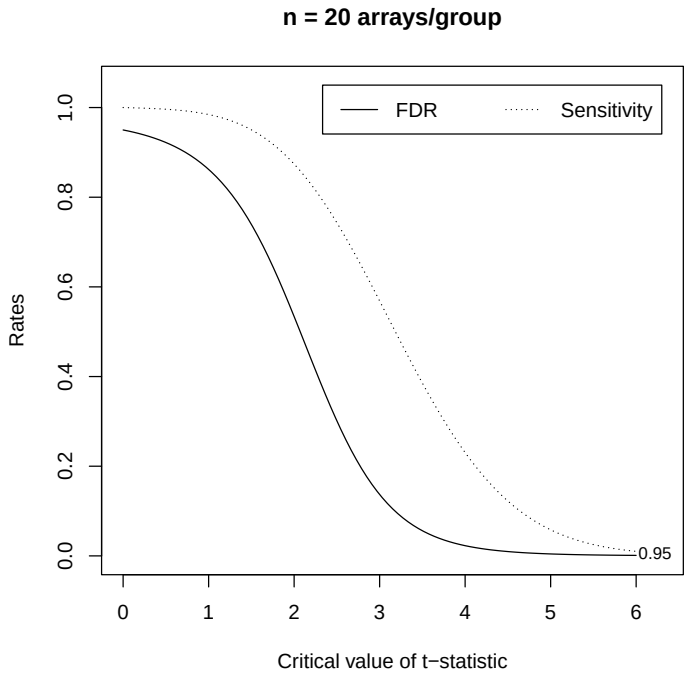

Figure 2: FDR and sensitivity as a function of the threshold for declaring a gene to be DE

4 Identifying DE genes

`OCplus` offers three different functions for identifying differentially expressed genes. All three are based on different variants of the false discovery rate: `EOC` computes the global false discovery rate (FDR) for each gene, `fdr1d` and `fdr2d` compute different variants of the local false discovery rate (fdr). Using the FDR is the conventional and most direct approach and works generally out of the box. The fdr approach is potentially more powerful, because it uses smoothing to combine information across genes, but it may require some experimentation to get the smoothing paramters right: `fdr1d` will often work with the default settings, but `fdr2d` will usually require some modifications (but is proportionally more powerful than `fdr1d`).

We use (unrealistically simple, but convenient) simulated data in the following to demonstrate these approaches:

```
> set.seed(123)
> simdat = MAsim(ng=10000, n=10, p0=0.95, D=1, sigma=1)
> dim(simdat)

[1] 10000    20

> colnames(simdat)

 [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "1" "1" "1" "1"
[16] "1" "1" "1" "1" "1"
```

`simdat` contains the simulated log-expression values for 10,000 genes and two groups of samples with 10 chips per group; the log-expression values are assumed normal and independent, with standard deviation one and mean zero for the 95% non-DE expressed genes, and mean ± 1 for the DE genes in the second group.

4.1 EOC

This function is the counterpart to `TOC` and returns the empirical operating characteristics: for each gene, the associated t-statistic, p-value, FDR and sensitivity.

```
> sim1 = EOC(simdat, colnames(simdat))
> sim1[1:5,]

        tstat   pvalue       FDR      sens
1 -1.08042708 0.294988 0.8883654 0.9791502
2 -0.01011066 0.991940 0.9600976 1.0000000
3  1.07476286 0.297496 0.8893513 0.9791502
4  1.55454200 0.137240 0.7959427 0.9032160
5 -1.53863046 0.141232 0.8022685 0.9032160
```

Note that this function plots the operating characteristics by default, but this can be suppressed by setting the argument `plot=FALSE`, and the output still has its own plotting method, see Figure 3.

The genes with the smallest FDR can be extracted via `topDE`:

```
> topDE(sim1, co=0.1)
```

```
         tstat   pvalue        FDR        sens
7286 -6.923683 0.000000 0.00000000 0.000000000
2418 -6.624755 0.000004 0.01923200 0.002555231
261   5.232498 0.000052 0.03846399 0.024833621
324  -5.100353 0.000056 0.03846399 0.029938743
1357  5.485208 0.000036 0.03846399 0.014734901
1804 -5.061102 0.000060 0.03846399 0.032398626
3480 -5.762388 0.000020 0.03846399 0.004784699
3934 -4.966195 0.000068 0.03846399 0.034778803
4116  4.921912 0.000072 0.03846399 0.037310495
4267 -4.896086 0.000076 0.03846399 0.045041530
4951  4.920252 0.000072 0.03846399 0.039911046
6108 -4.899887 0.000072 0.03846399 0.042471013
7589 -5.697200 0.000020 0.03846399 0.007351062
8115 -5.176199 0.000052 0.03846399 0.027396695
8399 -5.447536 0.000040 0.03846399 0.017307420
8432  5.583280 0.000028 0.03846399 0.012372744
8767  5.595279 0.000028 0.03846399 0.009799388
9195  5.398711 0.000040 0.03846399 0.019879230
9207  5.239546 0.000052 0.03846399 0.022234610
6870 -4.864438 0.000084 0.04038719 0.047414117
1150 -4.788900 0.000100 0.04545745 0.049595297
3687  4.777048 0.000104 0.04545745 0.052126558
7683 -4.737323 0.000116 0.04849808 0.054419226
526   4.658298 0.000164 0.06436831 0.058403926
632   4.658329 0.000164 0.06436831 0.055800096
515   4.584209 0.000216 0.07988675 0.059742039
7731 -4.564034 0.000228 0.08120176 0.062075857
2282  4.506643 0.000252 0.08654398 0.064065811
```

The proportion of non-DE genes `p0` is by default estimated from the data using a variant of Storey's method; the estimate can be extracted from the output:

```
> p0(sim1)
```

```
[1] 0.9615998
```

`p0` can also be specified explicitly in the function call, if an alternative estimate is availabale, see Section 5.

```
> plot(sim1)
```

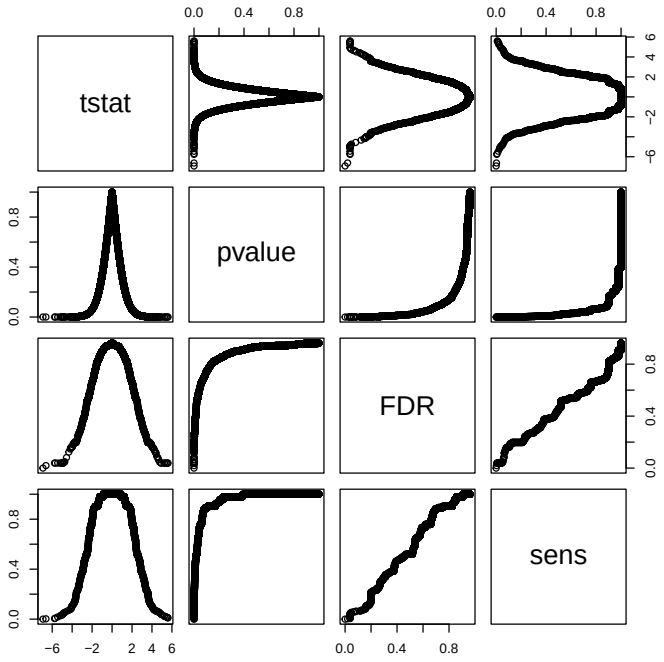

Figure 3: Estimated FDR and sensitivity for the simulated data

4.2 fdr1d

This function returns for each gene the test statistic and the local (univariate) fdr:

```
> sim2 = fdr1d(simdat, colnames(simdat), verb=FALSE)
> sim2[1:5,]

        tstat fdr.local
1  1.08042708 0.9953734
2  0.01011066 0.9869064
3 -1.07476286 0.9398789
4 -1.55454200 0.9264060
5  1.53863046 0.9653268
```

The `verb=FALSE` here just stops the function from reporting the number of the current permutation, which creates too much output for a vignette. `fdr1d` does not plot automatically, but has its own plotting method, see Figure 4.

The proportion of non-DE genes `p0` is by default estimated from the data, using a variant of Efron's method. The estimate can again be extracted from the output via the function `p0`; it is also reported by the summary method:

```
> summary(sim2)

     tstat            fdr.local
 Min.   :-5.59528   Min.   :0.00929
 1st Qu.:-0.74855   1st Qu.:0.94698
 Median :-0.01244   Median :0.96968
 Mean   :-0.01883   Mean   :0.93877
 3rd Qu.: 0.68864   3rd Qu.:0.98488
 Max.   : 6.92368   Max.   :0.99999
```

Note that the value for `p0` is very close to the estimate used in `sim1` above.

The genes with fdr below a specified threshold can again be listed by

```
> topDE(sim2, co=0.1)

         tstat  fdr.local
7286  6.923683 0.00929036
2418  6.624755 0.01451970
3480  5.762388 0.04710459
7589  5.697200 0.05089399
8767 -5.595279 0.05115318
8432 -5.583280 0.05183200
1357 -5.485208 0.05738021
9195 -5.398711 0.06227360
8399  5.447536 0.06788212
9207 -5.239546 0.07277199
```

```
> plot(sim2)
```

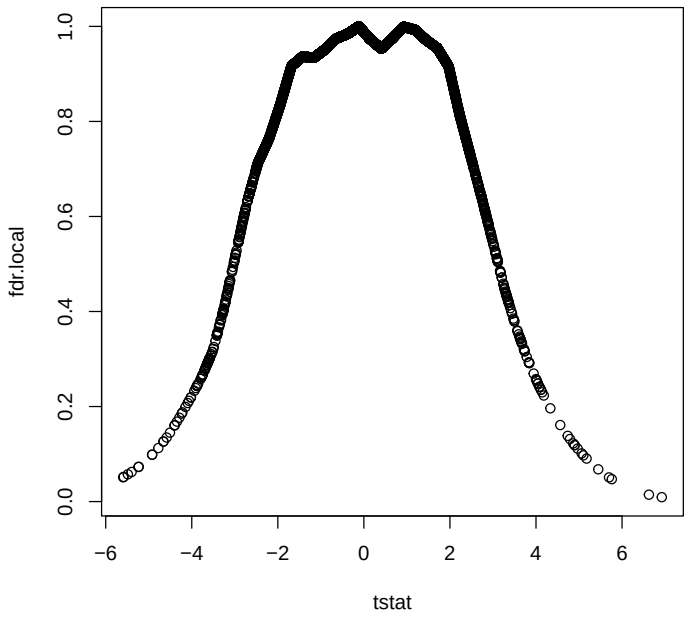

Figure 4: Local fdr as a function of t-statistics for the simulated data; inner ticks on the horizontal axis indicate observed t-statistics.

```
261  -5.232498 0.07328162
8115  5.176199 0.09048230
324   5.100353 0.09719682
4116 -4.921912 0.09861059
4951 -4.920252 0.09876205
```

4.3 fdr2d

This function reports for each the test statistic, the auxiliary test statistic (generally the log of the standard error) and the local fdr based on the two test statistics:

```
> sim3 = fdr2d(simdat, colnames(simdat), p0=p0(sim2), verb=FALSE)
> sim3[1:5,]
        tstat      logse fdr.local
1  1.08042708 -0.6146155 0.8822793
```

```
2  0.01011066 -1.0317545 0.9126043
3 -1.07476286 -0.7420905 0.8926538
4 -1.55454200 -1.0702954 0.8772499
5  1.53863046 -0.9119842 0.8869714
```

Note that here the proportion `p0` is specified explicitly – we take the estimate based on the univariate densities used for `sim2`. The default behavior for `fdr2d` is also to estimate `p0` from the data, but the results can be highly erratic, and we recommend using an external estimate, either from `EOC` or `fdr1d` as above, or from `tMixture`, as described in Section 5.

As mentioned above, the smoothing parameter `smooth` of `fdr2d` will often require adjustment. A useful graphical diagnostic for a suitable value of `smooth` is shown in Figure 5: in theory, the onedimensional fdr is equal to the twodimensional fdr averaged across the log standard errors; in Figure 5, the solid line shows the onedimensional fdr (`sim2`) and the broken line shows the averaged twodimensional fdr; the agreement between the two lines is good, though better for low fdr (in the tail) than for high fdr (in the center). In practice, it is quite hard to achieve perfect agreement throughout, as different degrees of smoothing might be required in the center compared to the tails. We are, however, generally only interested in the genes with low fdr anyway, so it is usually sufficient to achieve a good fit in the tails.

The results can be summarized as above, and the top list extracted with the same method as for `fdr1d`:

```
> summary(sim3)
```

```
     tstat              logse           fdr.local
 Min.   :-5.59528   Min.   :-1.6559   Min.   :0.0000039
 1st Qu.:-0.74855   1st Qu.:-0.9374   1st Qu.:0.8732190
 Median :-0.01244   Median :-0.8200   Median :0.8944727
 Mean   :-0.01883   Mean   :-0.8312   Mean   :0.8714442
 3rd Qu.: 0.68864   3rd Qu.:-0.7127   3rd Qu.:0.9130052
 Max.   : 6.92368   Max.   :-0.2370   Max.   :1.5133774
```

```
> topDE(sim3, co=0.1)
```

```
         tstat      logse    fdr.local
2418  6.624755 -1.1640917 3.881476e-06
7286  6.923683 -1.1099516 3.881476e-06
8432 -5.583280 -1.0391987 3.881476e-06
8767 -5.595279 -1.0040761 3.881476e-06
1357 -5.485208 -0.9759968 1.013406e-02
9195 -5.398711 -0.9326397 1.218408e-02
6108  4.899887 -0.8199713 2.022921e-02
7589  5.697200 -1.1382646 2.068290e-02
6870  4.864438 -0.8269019 2.273411e-02
3480  5.762388 -1.1545476 2.460183e-02
```

```
> plot(sim2)
> lines(average.fdr(sim3), lty=2)
```

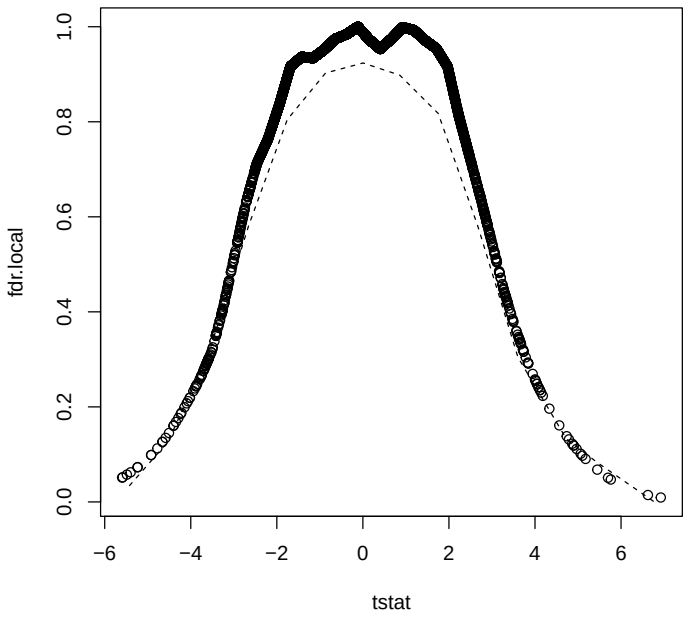

Figure 5: Local onedimensional fdr as a function of the t-statistic (solid line, as in Figure 4) and averaged twodimensional fdr (broken line).

```
2612  4.044563 -0.6664508 2.794586e-02
3934  4.966195 -0.9017693 3.147987e-02
3739 -4.345276 -0.7770245 3.365484e-02
4243 -4.217978 -0.7644934 4.041926e-02
8399  5.447536 -1.0913539 4.048985e-02
526  -4.658298 -0.8758258 4.063613e-02
4116 -4.921912 -0.9357994 4.213936e-02
1150  4.788900 -0.9348992 4.405492e-02
9085  3.383585 -0.5761734 4.472602e-02
1804  5.061102 -1.0147315 5.024718e-02
2946 -3.880855 -0.6957500 5.203410e-02
6711  4.016070 -0.7679300 5.402341e-02
6400  3.739040 -0.7069355 5.445395e-02
632  -4.658329 -0.9241577 5.495515e-02
3425  3.840764 -0.7495248 6.014583e-02
5417  3.834645 -0.7790989 6.834854e-02
7948  2.923848 -0.5316809 7.148949e-02
5901 -3.688175 -0.7389362 7.505865e-02
7683  4.737323 -1.0125250 7.741859e-02
9737 -3.622699 -0.6799167 8.110732e-02
1471 -3.706950 -0.7858991 8.140958e-02
515  -4.584209 -0.9842774 8.525831e-02
9024  3.638140 -0.7548439 8.707996e-02
324   5.100353 -1.0505343 8.930437e-02
9207 -5.239546 -1.1186978 8.944276e-02
2480  4.181968 -0.8827882 8.954586e-02
404   4.080794 -0.8681257 9.275735e-02
261  -5.232498 -1.1232774 9.471108e-02
7257 -3.655494 -0.7834283 9.962233e-02
3687 -4.777048 -1.0352392 9.975418e-02
```

Note that the table of fdrs (`$fdr`) in this output contains fdrs greater than one; this, too, is a consequence of not quite correct smoothing for genes with large fdr.

Figure 6 shows the standard plot for output from `fdr2d`. This is basically a scatterplot of the two contributing statistics, with the estimated fdr overlayed as isolines. Note that the averaged values shown as a broken line in Figure 5 are calculated by averaging along the vertical axis (the log standard errors) in Figure 6.

4.4 Compare performances

We have now three different analyses for the simulated data, one in terms of FDR and two in terms of fdr. The summary functions indicate that `fdr2d` seems to find the most regulated genes, but this is misleading, as FDR and fdr cannot be compared directly. The function `OCshow` compares the output from multiple

```
> plot(sim3)
```

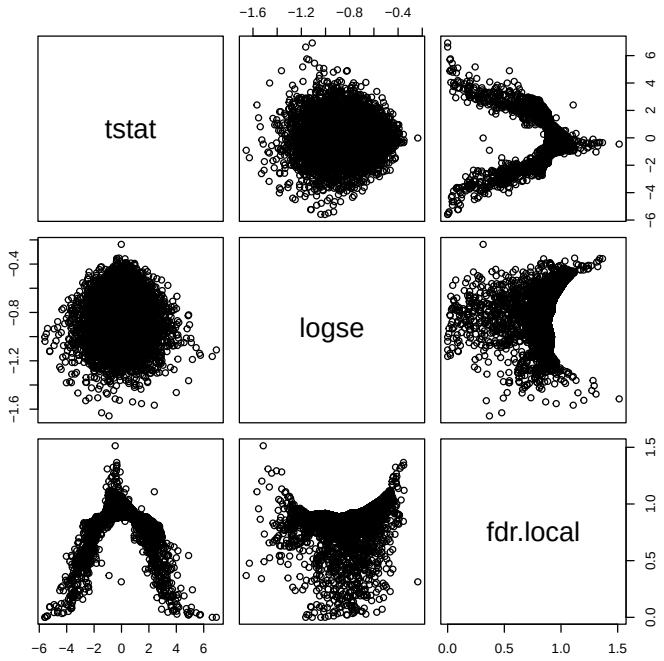

Figure 6: A scatterplot of the log standard errors vs. the t-statistics, with the estimated fdr indicated by isolines.

```
> OCshow(sim1, sim2, sim3, legend=c("FDR","fdr1d","fdr2d"))
```

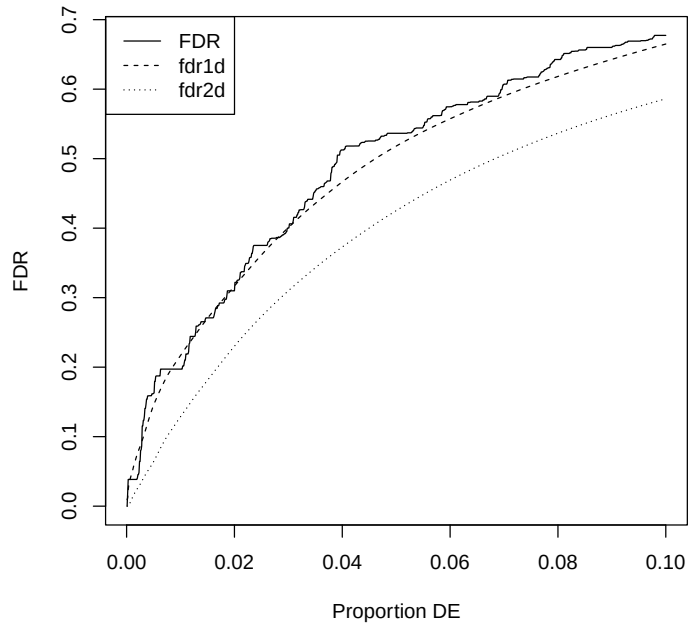

Figure 7: Vertical axis shows the resulting (global) FDR if we declare for each method the proportion of genes with the smallest FDR/fdr shown on the horizontal axis to be differentially expressed.

analyses graphically, in terms of FDR, by averaging across fdrs. The result in Figure 7 shows the resulting FDR if we choose the declare the proportion of genes with the smallest FDR/fdr shown on the horizontal axis to be DE. In this case, the original FDR as provided by `EOC` and the FDR based on `fdr1d` are comparable, but the FDR based on `fdr2d` is clearly lower.

5 Estimating the proportion of non-DE genes

An alternative method for estimating the proportion of non-DE expressed genes in a data set is based on fitting a mixture t-distributions to the vector of observed t-statistics, see Pawitan et al. (2005a). In the simplest case, we just compute the t-statistics and specify the number `nq` of mixture components in the call to `tMixture`:

```
> tt = tstatistics(simdat, colnames(simdat))
> tt[1:10,]

 [1]  1.08042708  0.01011066 -1.07476286 -1.55454200  1.53863046
 [6] -1.52024951 -0.42763371  0.65026341 -0.37170449  0.25691453

> tm = tMixture(tt, nq=3)
```

In this case, we assume three components, corresponding to down-, up-, and non-regulated genes, and the mixture proportion of the non-regulated genes is the desired estimate:

```
> tm$p0.est

[1] 0.9151372
```

The estimate is a bit low compared to what we know is true ($p_0 = 0.95$). This is due to the fact that the numerical optimization used by this routine is fairly sensitive to the choice of starting values; it is therefore good practice to vary the starting values for different parameters:

```
> tMixture(tt, nq=3, p0=0.80)$p0.est

[1] 0.9568759

> tMixture(tt, nq=3, p0=0.60)$p0.est

[1] 0.9567759
```

This is essentially the true value for both starting values.

Note that the specification of too many components can lead to spurious mixture components that cannot be distinguished reliably from the non-regulated genes. In order to get reasonable estimates, these components with small non-centrality parameter `delta` are combined with the non-regulated component (which has by definition `delta=0`). E.g.:

```
> tm2 = tMixture(tt, nq=5)
> tm2$p0.est

[1] 0.9532227

> tm2$p0.raw

[1] 0.03364113
```

The estimated proportion `p0.est` is the same as with three components, but it is really the sum of `p0.raw` and the component with non-centrality parameter absolutely smaller than a critical value (0.75 by default):

```
> tm2$p1

[1] 0.01630578 0.91958160 0.01209428 0.01837721

> tm2$delta

[1] -2.587871223 -0.004285759 -1.584812608  2.558282790
```

References

Y. Pawitan, K. R. Krishna Murthy, S. Michiels, and A. Ploner. Bias in the estimation of false discovery rate in microarray studies. *Bioinformatics*, 21 (20):3865–72, 2005a.

Y. Pawitan, S. Michiels, S. Koscielny, A. Gusnanto, and A. Ploner. False discovery rate, sensitivity and sample size for microarray studies. *Bioinformatics*, 21(13):3017–24, 2005b.

A. Ploner, S. Calza, A. Gusnanto, and Y. Pawitan. Multidimensional local false discovery rate for microarray studies. *Bioinformatics*, page btk013, In press. doi: doi:10.1093/bioinformatics/btk013.