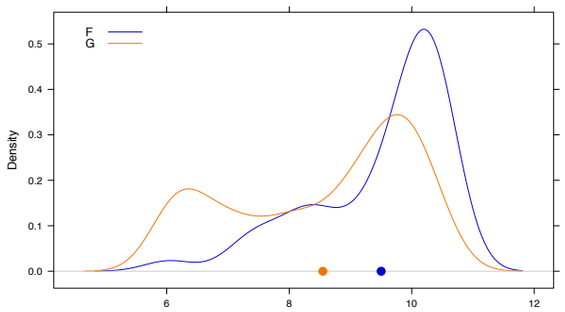
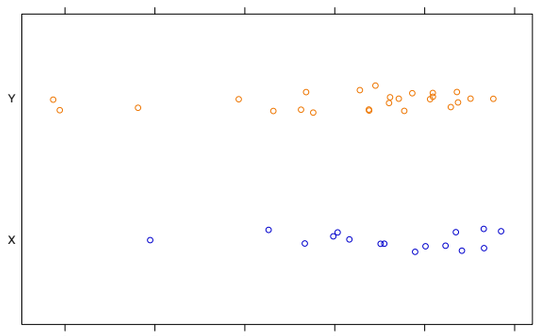
# STAT 540
# Supplementary Slides for Lecture 09

Created by Dr. Jennifer (Jenny) Bryan
Revised by Dr. Gabriela Cohen Freue

Large scale inference -- multiple testing

| True data-generating model | Status w.r.t. null hypothesis $H_0: F = G$ | Observed data | Inferential results | "call" |
|---|---|---|---|---|
| ⋮ | ⋮ | ⋮ | ⋮ | |
|  | $H_0$ false |  | $T = \frac{\bar{X}_{n_x} - \bar{Y}_{n_y}}{s_{\bar{X}-\bar{Y}}} \cong 1$<br>p-value $\cong 0.3$ | not a hit |
| ⋮ | ⋮ | ⋮ | ⋮ | |

<u>*unique*</u> hypothesis testing problem of <u>*same structure*</u> for thousands of genes

| | | | | |
|---|---|---|---|---|
| ⋮ | ⋮ | ⋮ | ⋮ | |

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | true negatives | false positives<br>Type I errors | # nulls |
| $H_A$ holds<br>"interesting" | false negatives<br>Type II errors | true positives | # alts |
| | | discoveries | # genes |

Using notation of Storey 2003 (but not layout!).

Everything interior is a random variable!

You know m = # genes.
You observe S = # discoveries.
Everything else ... who knows?

$$\pi_0 = \frac{m_0}{m} = \text{ proportion of truly null genes}$$

(often assumed to be close to 1)

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | $m_0$ - F | F | $m_0$ |
| $H_A$ holds<br>"interesting" | $m_1$ - T | T | $m_1$ |
| | | S | m |

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | true negatives | false positives<br>Type I errors | # nulls |
| $H_A$ holds<br>"interesting" | false negatives<br>Type II errors | true positives | # alts |
| | | discoveries | # genes |

false positive rate $= P$(stat sig test stat or pvalue) for a truly null gene, i.e. one of the $m_0$

if you threshhold at a p-value $\leq \alpha$, for example $\alpha = 0.05$, then you control the false positive rate

then the expected number of false positives is $\alpha m_0$ .... which can be quite large!
example: 0.05 * 5000 = 250

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | $m_0$ - F | F | $m_0$ |
| $H_A$ holds<br>"interesting" | $m_1$ - T | T | $m_1$ |
| | | S | m |

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | true negatives | false positives<br>Type I errors | # nulls |
| $H_A$ holds<br>"interesting" | false negatives<br>Type II errors | true positives | # alts |
| | | discoveries | # genes |

family-wise error rate (FWER) = $P(F > 1)$

probability at least one null gene is called a hit

if you use Bonferroni, then you control FWER

very very conservative approach

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | $m_0$ - F | F | $m_0$ |
| $H_A$ holds<br>"interesting" | $m_1$ - T | T | $m_1$ |
| | | S | m |

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | true negatives | false positives<br>Type I errors | # nulls |
| $H_A$ holds<br>"interesting" | false negatives<br>Type II errors | true positives | # alts |
| | | discoveries | # genes |

false discovery rate (FDR) = $E\left(\frac{F}{S}\right)$

expected proportion of false positives among the hits

if you use q-values, you control FDR

| "call" based on obs. data<br>true state of nature | "not hit" | reject $H_0$<br>"hit" | |
|---|---|---|---|
| $H_0$ holds | $m_0$ - F | F | $m_0$ |
| $H_A$ holds<br>"interesting" | $m_1$ - T | T | $m_1$ |
| | | S | m |

*Statistical Science*
2003, Vol. 18, No. 1, 71–103

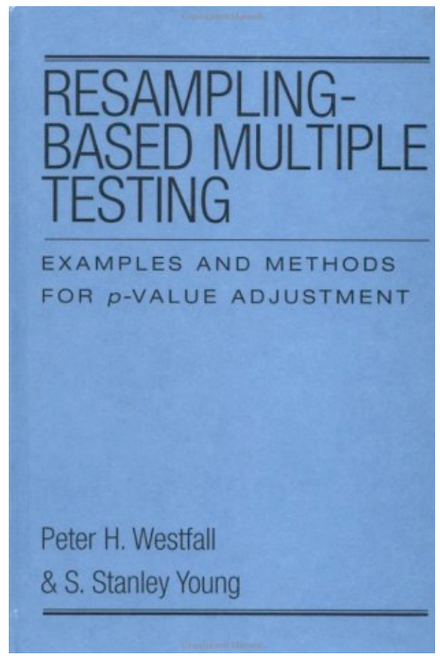

# Multiple Hypothesis Testing in Microarray Experiments

**Sandrine Dudoit, Juliet Popper Shaffer and Jennifer C. Boldrick**

Dudoit S, Shaffer JP, Boldrick JC (2003) Multiple hypothesis testing in microarray experiments. Stat Sci 71–103.

PDF via Project Euclid

Westfall PH, Young SS (1993) Resampling-based multiple testing: examples and methods for p-value adjustment (John Wiley and Sons, New York).

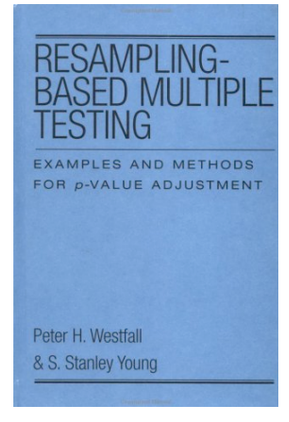


better alternatives exist for Type I error rate control

single-step (e.g. Bonferroni) vs. stepwise (e.g. Holm, Westfall & Young's maxT and minP)

notion of a global adjustment to the p-values vs. a customized adjustment for each p-value, i.e. using the observed data, based on ranked p-values

also advantageous to account for dependence structure in the test statistics, i.e. employ a resampling based multiple testing procedure



but let's abandon the effort to control false positive rates and focus on false discovery rate

In 1995, Benjamini and Hochberg proposed *the idea* of controlling the false discovery rate and also provided *a procedure* for doing so.

The procedure was presented in a frequentist style, using ranked p-values.

But more recently, a Bayesian or empirical Bayesian backstory has also been created and gained traction.

*J. R. Statist. Soc.* B (1995)
**57**, *No.* 1, *pp.* 289–300

**Controlling the False Discovery Rate: a Practical and Powerful Approach to Multiple Testing**

By YOAV BENJAMINI† and YOSEF HOCHBERG

*Tel Aviv University, Israel*

[Received January 1993. Revised March 1994]

## 3. FALSE DISCOVERY RATE CONTROLLING PROCEDURE

### 3.1. *The Procedure*

Consider testing $H_1, H_2, \ldots, H_m$ based on the corresponding $p$-values $P_1, P_2, \ldots, P_m$. Let $P_{(1)} \leqslant P_{(2)} \leqslant \ldots \leqslant P_{(m)}$ be the ordered $p$-values, and denote by $H_{(i)}$ the null hypothesis corresponding to $P_{(i)}$. Define the following Bonferroni-type multiple-testing procedure:

$$\text{let } k \text{ be the largest } i \text{ for which } P_{(i)} \leqslant \frac{i}{m} q^*;$$

$$\text{then reject all } H_{(i)}\ i = 1, 2, \ldots, k. \tag{1}$$

*Theorem 1.* For independent test statistics and for any configuration of false null hypotheses, the above procedure controls the FDR at $q^*$.

*Proof.* The theorem follows from the following lemma, whose proof is given in Appendix A.

*Lemma.* For any $0 \leqslant m_0 \leqslant m$ independent $p$-values corresponding to true null hypotheses, and for any values that the $m_1 = m - m_0$ $p$-values corresponding to the false null hypotheses can take, the multiple-testing procedure defined by procedure (1) above satisfies the inequality

$$E(\mathbf{Q} \mid P_{m_0+1} = p_1, \ldots, P_m = p_{m_1}) \leqslant \frac{m_0}{m} q^*. \tag{2}$$

Now, suppose that $m_1 = m - m_0$ of the hypotheses are false. Whatever the joint distribution of $P_1'', \ldots, P_{m_1}''$ which corresponds to these false hypotheses is, integrating inequality (2) above we obtain

$$E(\mathbf{Q}) \leqslant \frac{m_0}{m} q^* \leqslant q^*,$$

and the FDR is controlled.

*Remark.* Note that the independence of the test statistics corresponding to the false null hypotheses is not needed for the proof of the theorem.

From Benjamini and Hochberg 1995. Q is the false discovery proportion. E(Q)=FDR.

John Storey has played a huge role in popularizing FDR control in genomics.

He has made substantial theoretical / methodological contributions as well.

He also provided the q-value package to implement a version of BH's procedure that produces suitably adjusted p-values ("q-values").

**Statistical significance for genomewide studies**

John D. Storey*† and Robert Tibshirani‡

9440–9445 | PNAS | August 5, 2003 | vol. 100 | no. 16 www.pnas.org/cgi/doi/10.1073/pnas.1530509100

Statistical significance for genomewide studies. Storey JD, Tibshirani R. Proc Natl Acad Sci USA 2003 Aug 5100(16):9440-5

Storey coined the term q-value.

q-value (feature) = expected proportion of false positives if this feature is called significant

= expected proportion of false positives among all features as or more extreme than this feature

compare / contrast w/ p-value = probability of a null feature being as or more extreme as this feature

# Let's examine the distribution of p-values (needed in Lecture 09 for Storey's estimation of FDR)

# Under $H_0$, p-values are distributed $U[0,1]$

Let $X \sim F$.

Define a new random variable:
$Z = F(X)$

Then $Z \sim U[0,1]$.

Proof:

$P(Z \le z) = P(F(X) \le z) = P(X \le F^{-1}(z)) = F(F^{-1}(z)) = z,$
which is the CDF of a $U[0,1]$ random variable.

general fact

connection to p-values

Let $X \sim F$ and denote $E(X) = \mu$.

Consider the one-sided hypothesis test:
$H_0 : \mu \ge 0 \quad$ vs. $\quad H_1 : \mu < 0$

Let $T$ be the test statistic.
Then the p-value, as a random variable, is $F(T)$.
So the p-value is just like $Z$ in the general statement.
Apply the result and get that the p-values
are $U[0,1]$.

# Same result arises in the other one-sided test

Let $X \sim F$ and denote $E(X) = \mu$.

Consider the other one-sided hypothesis test:

$H_0 : \mu \leq 0$ vs. $H_1 : \mu > 0$

Let $T$ be the test statistic.

Then the p-value, as a random variable, is $Z = 1 - F(T)$.

$P(Z \leq z) = P(1 - F(X) \leq z) = P(F(X) \geq 1 - z)$

$= P(X \geq F^{-1}(1 - z)) = 1 - P(X \leq F^{-1}(1 - z))$

$= 1 - F(F^{-1}(1 - z)) = 1 - (1 - z) = z,$

which is the CDF of a $U[0,1]$ random variable.

For the keeners: show same result for a two-sided test!

Important sidebar:
Under $H_0$, p-values are distributed $U[0,1]$

Conversely, what does your intuition tell you about the distribution of p-values when $H_0$ is NOT true?

Important sidebar:
Under $H_0$, p-values are distributed $U[0,1]$

Conversely, what does your intuition tell you about the distribution of p-values when $H_0$ is NOT true?

It will be some distribution on the interval [0,1] but with more mass on tiny numbers, i.e. near zero, than on large numbers, i.e. near one.

Let's do a simulation study to help understand this.

Imagine a huge collection of test statistics ... huge as in 10,000, e.g. one for every gene.

If the gene-wise null hypothesis is true, then all test statistic come from N(0, 1).

If the gene-wise null hypothesis is false ("some genes are exciting!"), then let's say some test statistics come from N(0.75, 1).

Let's say 15% of the genes are exciting. This implies that pi0 = 0.85.

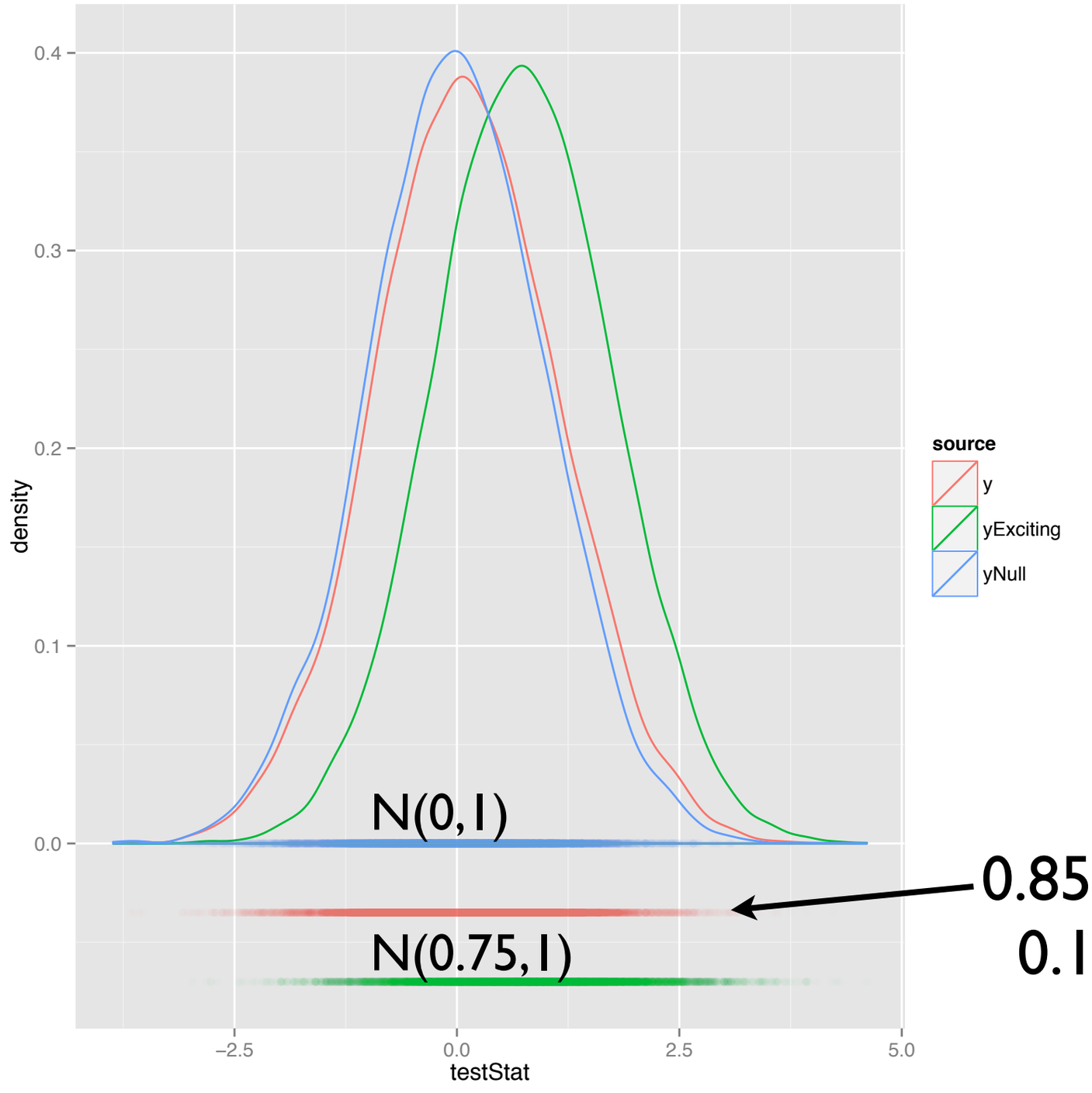
```
> nStats <- 10000

> set.seed(111503)

> pi0 <- 0.85                                  # proportion of truly null

> jDat <- data.frame(tNull = rnorm(nStats),
+ tExciting = rnorm(nStats, mean = 0.75),
+ isNull = rbinom(n = nStats, size = 1, prob = pi0))

> jDat$tObs <- with(jDat, ifelse(isNull, tNull, tExciting))

> table(jDat$isNull)

   0    1
1521 8479

> head(jDat)
        tNull  tExciting isNull        tObs
1  1.09389685  1.8291803      1  1.09389685
2  2.32292958  1.4196443      1  2.32292958
3  0.33944963  0.2820950      1  0.33944963
4  0.01991748 -0.8193972      1  0.01991748
5 -0.05414787  0.5916755      0  0.59167549
6 -0.10646835 -0.9703231      1 -0.10646835
```

mixture:
0.85 * N(0,1) +
0.15 * N(0.75, 1)

N(0,1)  N(0.75,1)

Bern(pi0 = 0.85)

# Think of observed test statistics as a mixture dominated by nulls, with a dash of "exciting" thrown in.

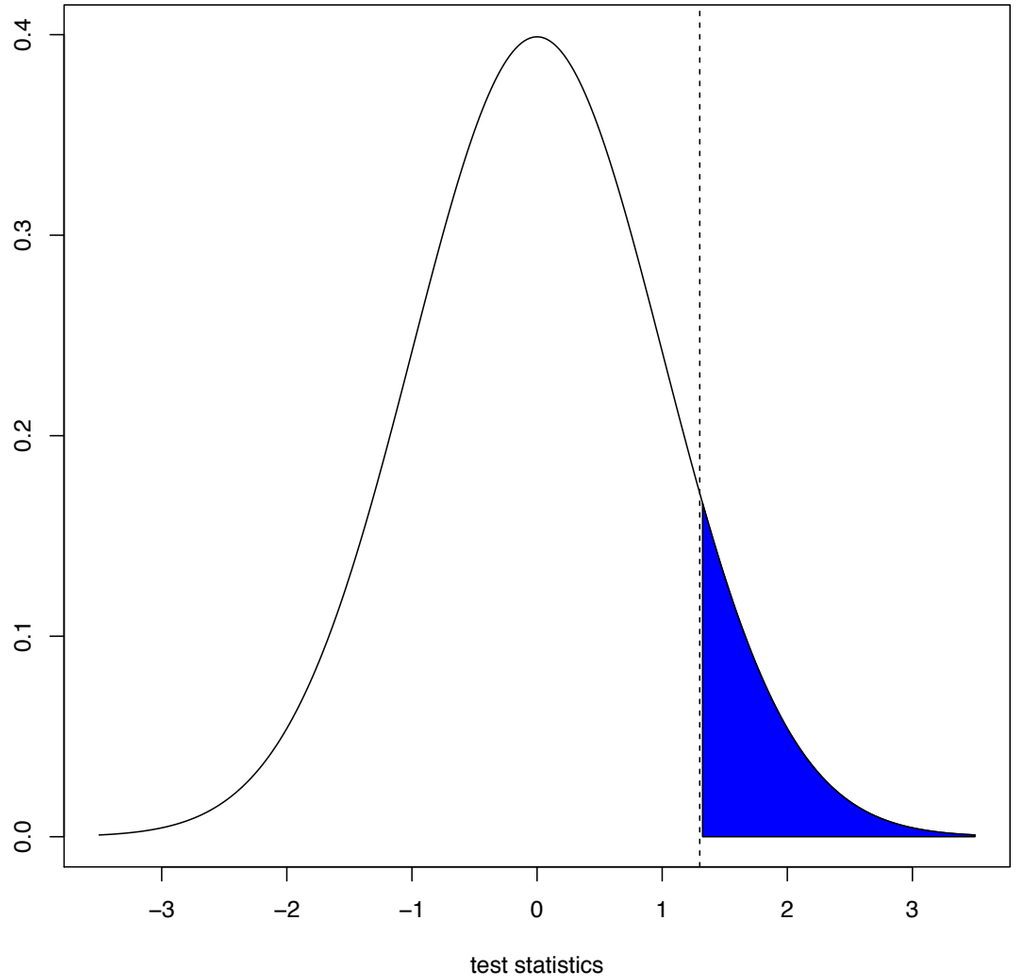



0.85 * N(0,1) +
0.15 * N(0.75, 1)

Imagine we want to conduct this one-sided test:

$$H_0 : \mu_g \leq 0 \qquad H_A : \mu_g > 0$$

p-value = prob. under H0 of a test stat "as or more extreme" than that observed
in this case, big positive numbers are "extreme" and so p-values will be right-tail probabilities

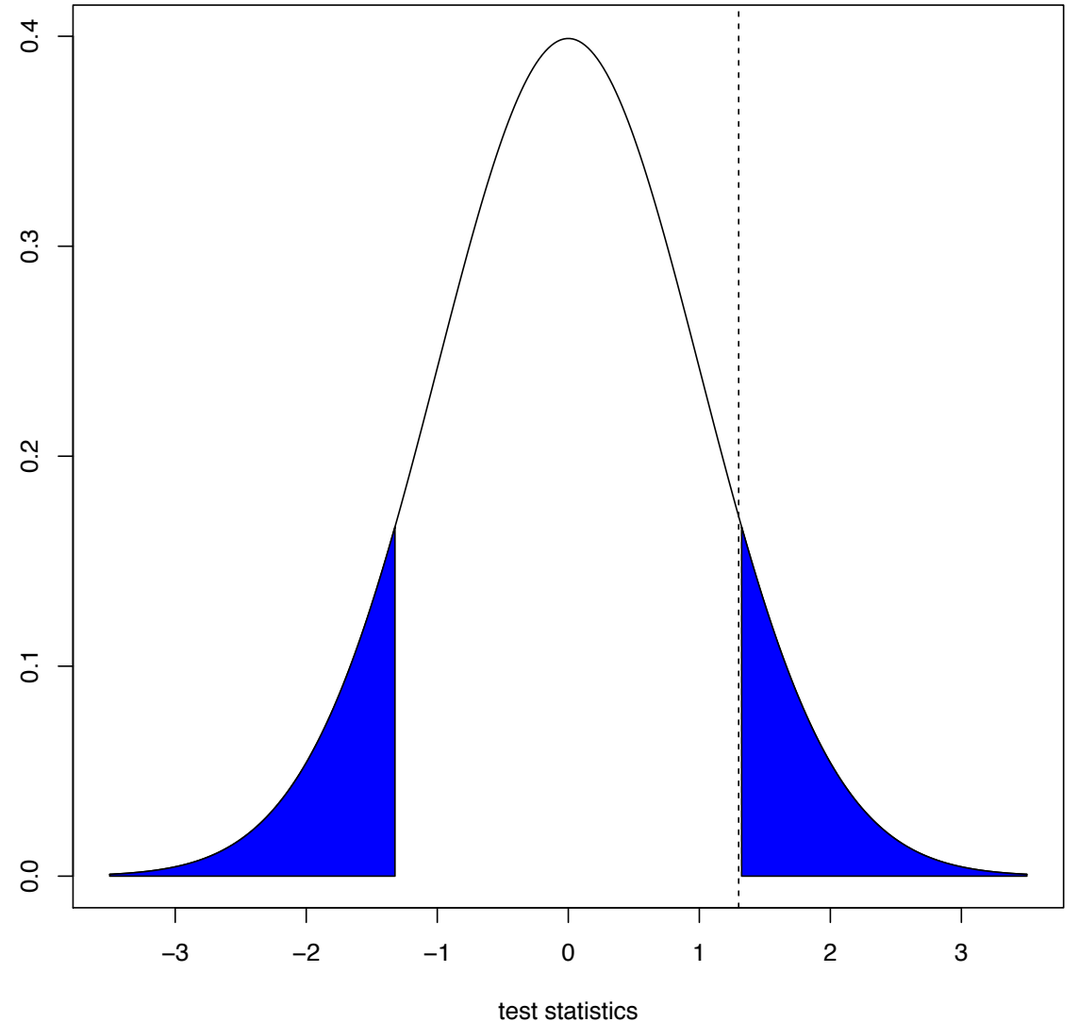


Imagine we want to conduct this two-sided test:

$$H_0 : \mu_g = 0 \qquad H_A : \mu_g \neq 0$$

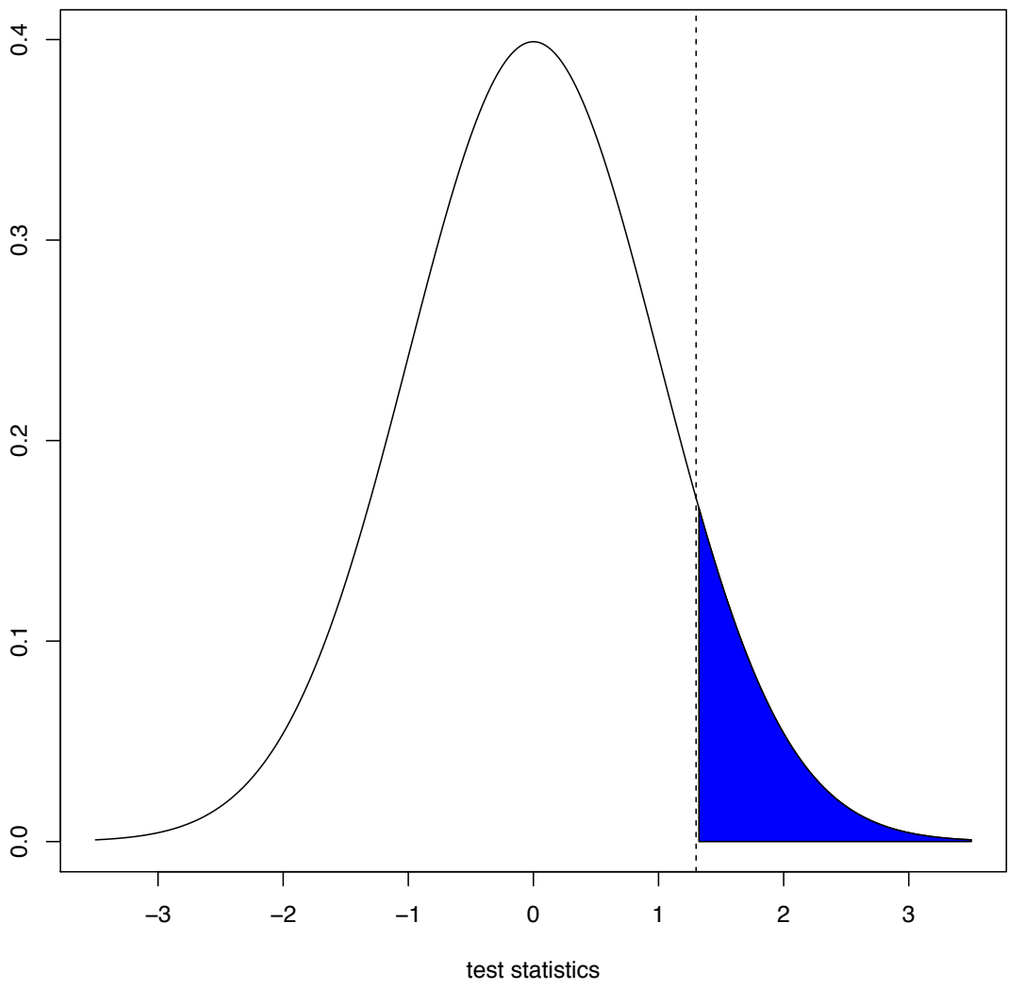
p-value = prob. under H0 of a test stat "as or more extreme" than that observed

in this case, big positive **or** negative numbers are "extreme" and so p-values will be two-tail probabilities

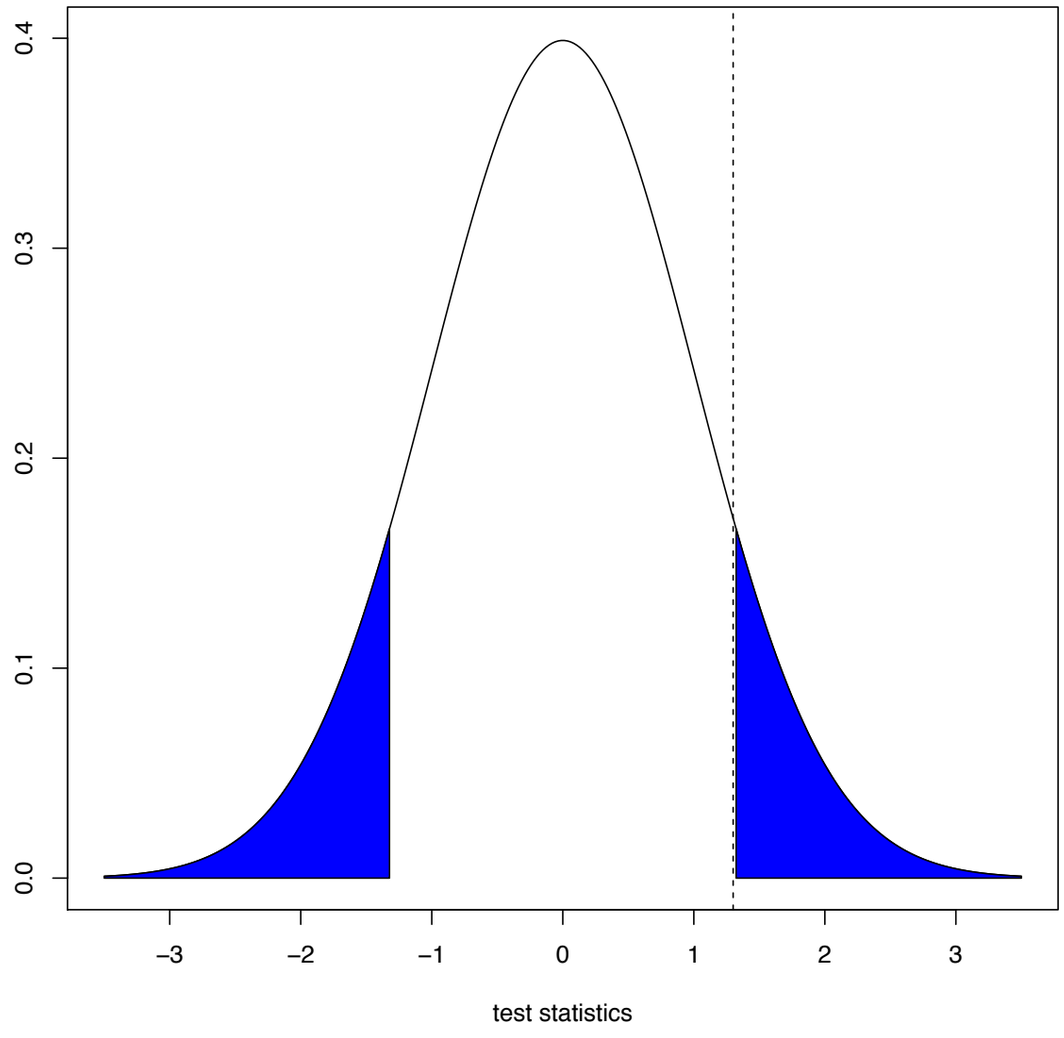


# Convert our null and exciting and observed (mix of 85% null and 15% "exciting") test statistics into both right-tail and two-tail probabilities or p-values.

```
pnorm(z, lower.tail = FALSE)
```




```
pnorm(abs(z), lower.tail = FALSE) * 2
```

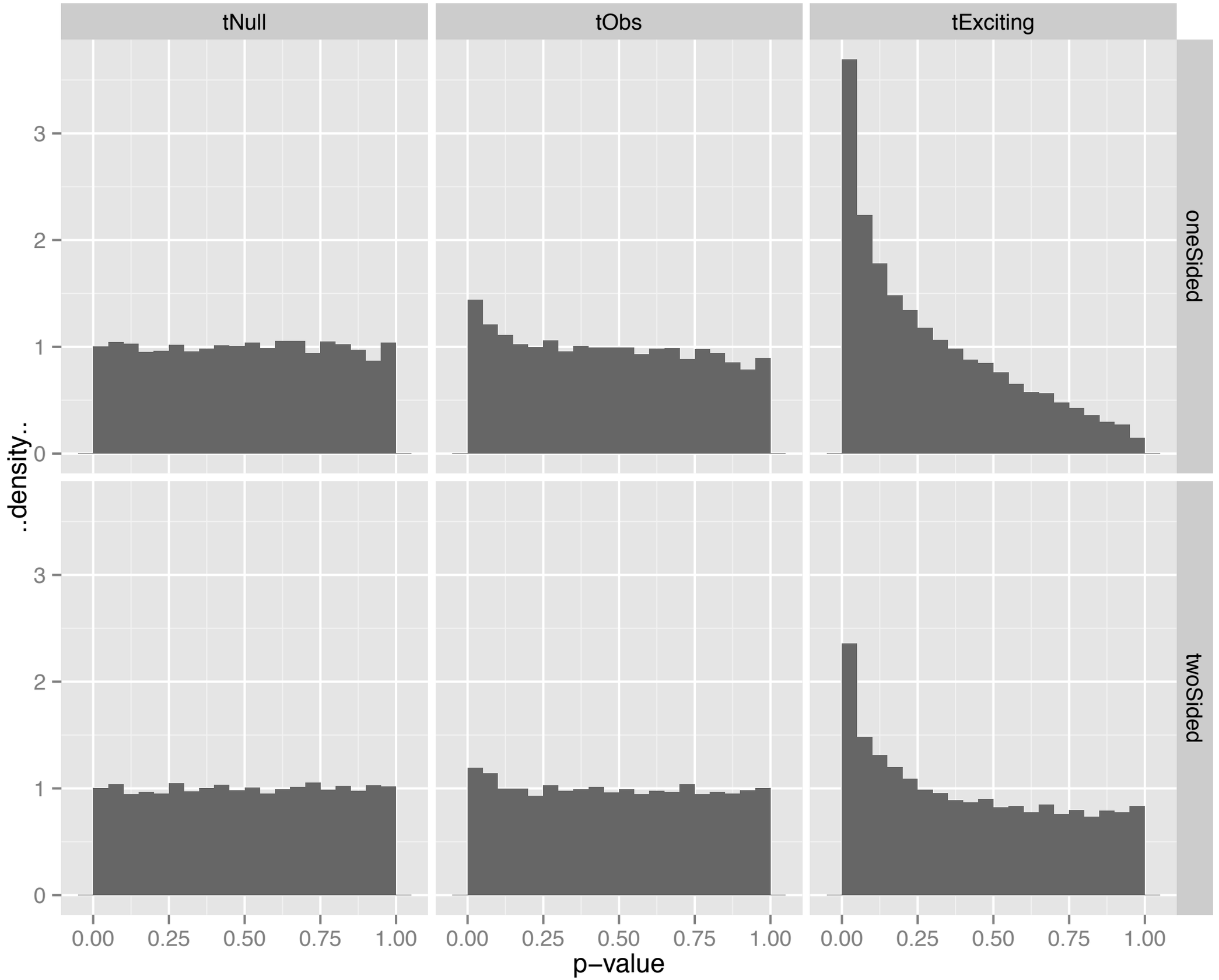


Let's inspect the empirical distribution of p-values.

Distribution of p-values from truly null genes should be U[0,1].

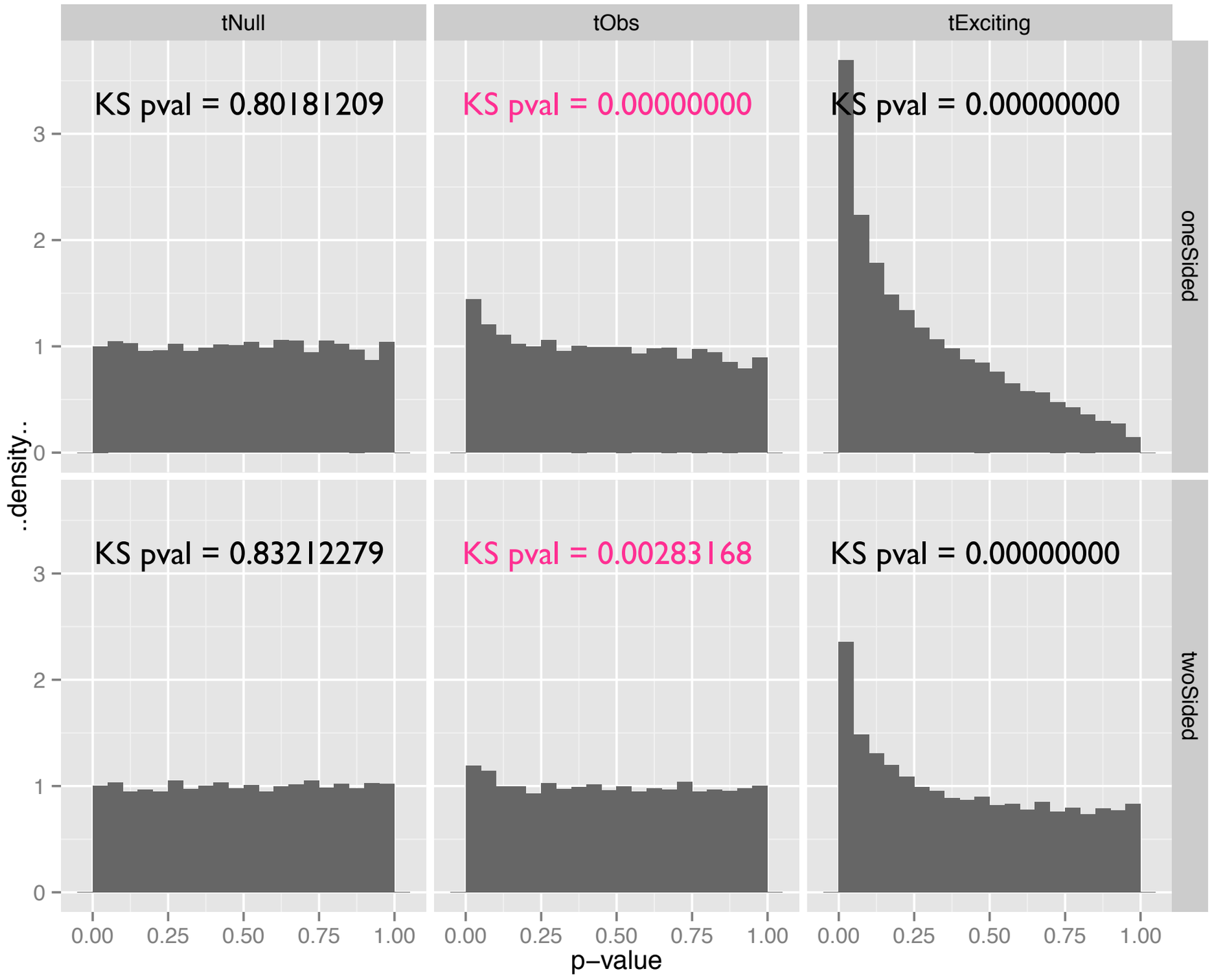
Distribution of p-values from truly exciting genes should ... put more mass on tiny numbers near zero and less mass on numbers near one.

Distribution of observed p-values (85% null + 15% exciting) should look like a mix, but dominated by the U[0,1].


tNull
tObs
tExciting
oneSided
twoSided
..density..
0
1
2
3
0.00
0.25
0.50
0.75
1.00
p−value

tNull
tObs
tExciting
oneSided
twoSided
..density..
p−value
0
1
2
3
0.00
0.25
0.50
0.75
1.00
see these as pi0 of U[0,1] + (1 - pi0) of sthg else

# Use Kolmogorov-Smirnov to test whether p-values come from U[0,1].




With thousands of p-values, even small departures from U[0,1] are detectable.

end rant about p-values being U[0,1] under the null (assuming independence!) and that departures from that help you see what fraction of your genes might be truly exciting

how to estimate the proportion of truly null genes $\hat{\pi}_0$ ?

regard the $p_i$ as a mixture:

$\pi_0 U[0,1] + (1 - \pi_0)$(some dist'n w/ lots of mass near 0)

Note we have now assumed independence of $p_i$ among the null genes. Can be relaxed a little but not completely.

most "large" p-values, i.e. those "near" 1, should originate from truly null genes

define "large" as $\lambda$

restate: most pvals in $[\lambda,1]$ are from $U[0,1]$

$$P(\text{null } p_i > \lambda) = 1 - \lambda$$

$$\#\{p_i > \lambda\} \approx \#\{\text{null } p_i > \lambda\}$$
$$\approx m_0(1-\lambda) = \pi_0 m(1-\lambda)$$

Rearrange to get an estimator for $\pi_0$ :

$$\hat{\pi}_0(\lambda) = \frac{\#\{p_i > \lambda\}}{m(1-\lambda)}$$

We've got an estimator for $\pi$ :

$$\hat{\pi}_0(\lambda) = \frac{\#\{p_i > \lambda\}}{m(1-\lambda)}$$

But we still have a tuning parameter $\lambda$.

Basic idea: compute $\hat{\pi}_0(\lambda)$ many values of $\lambda$.
Use a smooth regression of those to predict $\hat{\pi}_0(\lambda = 1)$.

That's what 'pi0.method = "smoother"' is about in the R package q-value.

FINALLY, the estimated q-value is basically this:

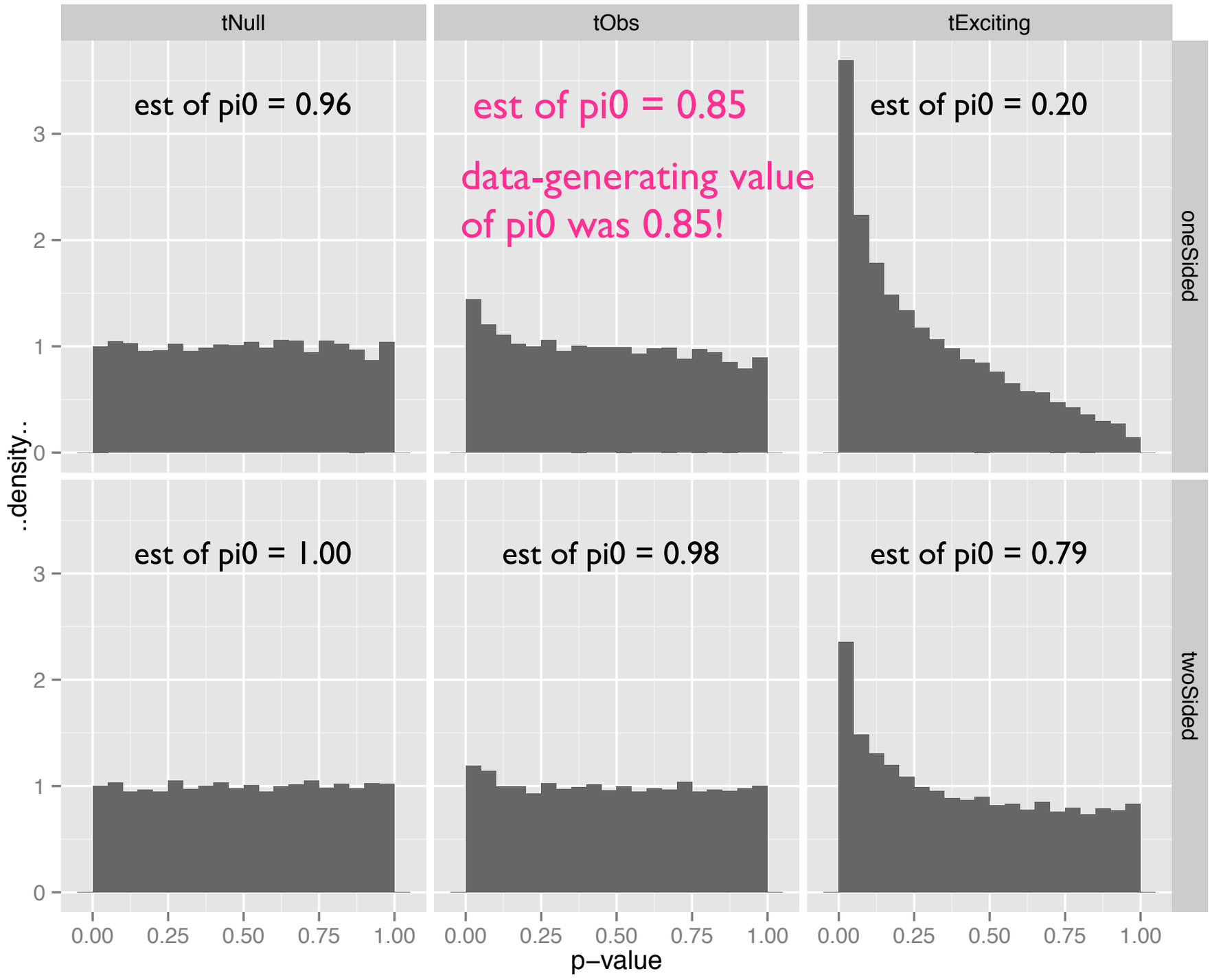
$$\hat{q}(p_i) = \frac{\hat{\pi}_0 m p_i}{\#\{p_j \leq p_i\}} = \frac{\hat{\pi}_0 m}{\text{rank of } p_i} p_i$$

Efron's recent book "Large Scale Inference" has some different approaches, esp. for separating the mixture of null and alternative test statistics. Also, he focuses on "local FDR" which is related but different. We'll save that for another year!

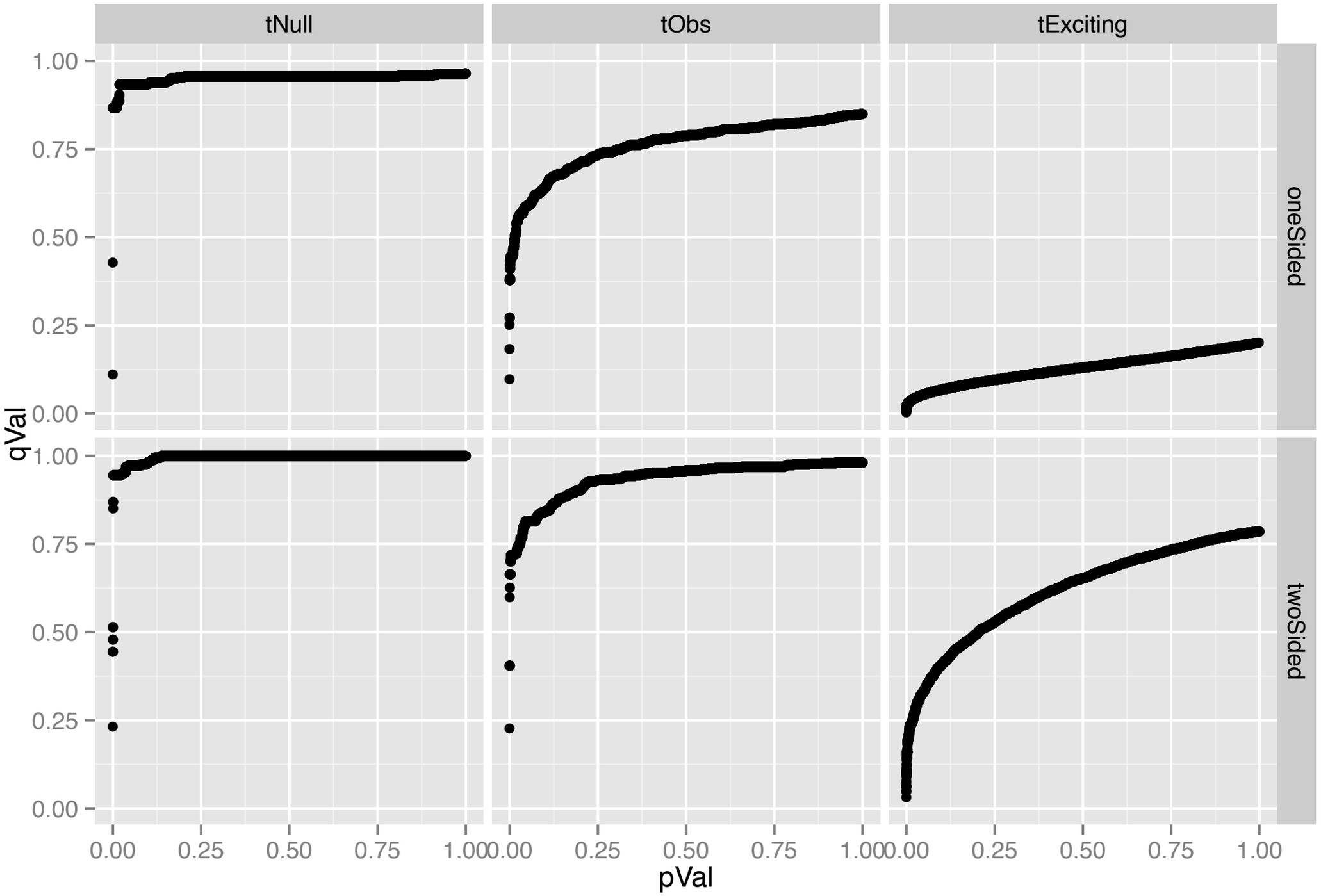
See, for example, the R package fdrtool to take a more fully realized empirical Bayes approach (?).

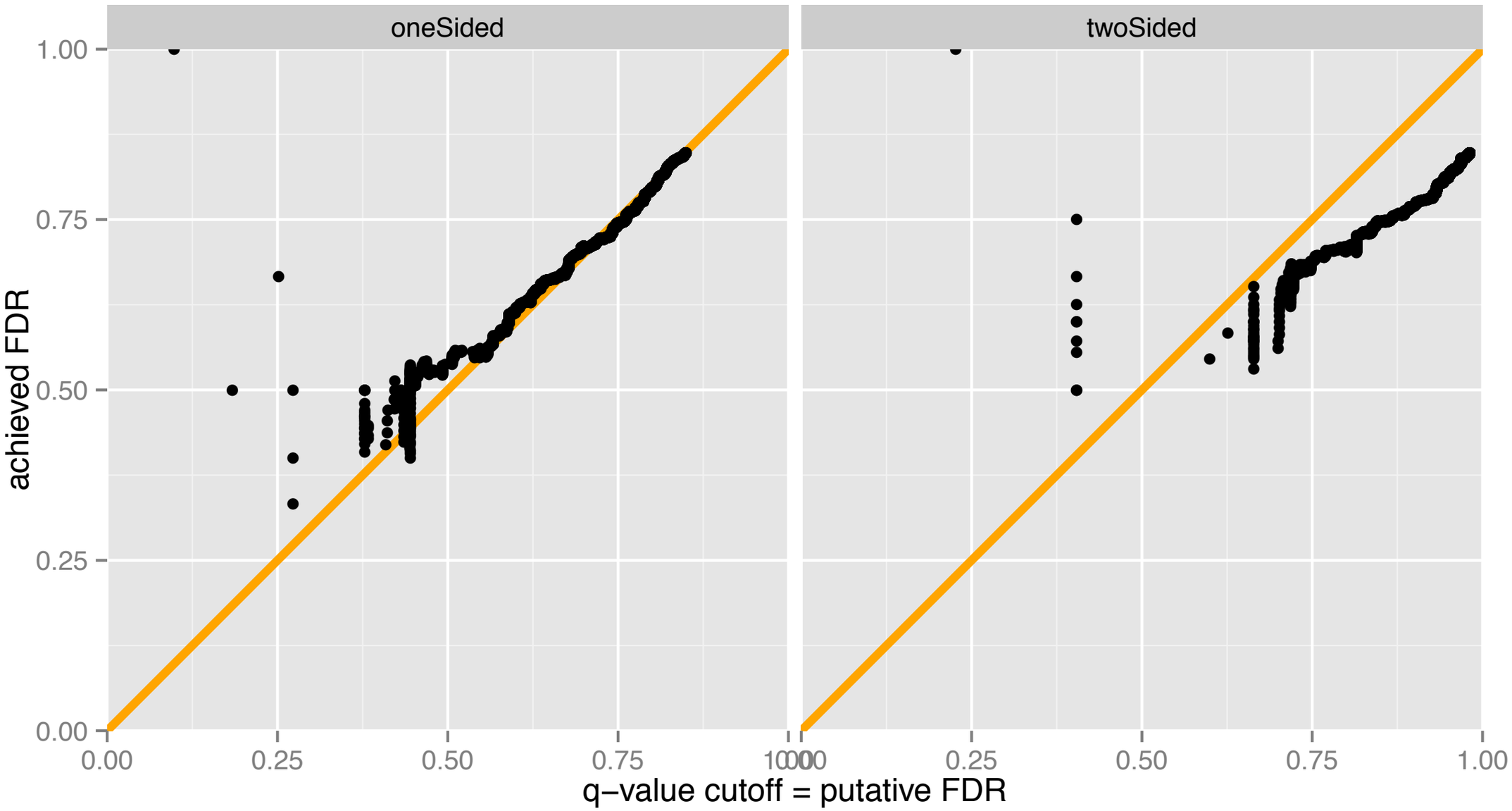
I will use the qvalue Bioconductor package here.

# What are the estimates of pi0 returned by `qvalue()`?



# q-values vs. p-values






Remember: these procedures work on average or in the long-run; no guarantee you hit the performance target in any particular dataset.

let's revisit the photoRec dataset and limma, now that we've talked about multiple testing adjustments ....

```
topTable(fit, coef=NULL, number=10, genelist=fit$genes, adjust.method="BH",
         sort.by="B", resort.by=NULL, p.value=1, lfc=0, confint=FALSE)

topTableF(fit, number=10, genelist=fit$genes, adjust.method="BH",
          sort.by="F", p.value=1, lfc=0)
```

```
 adjust.method: method used to adjust the p-values for multiple testing.
          Options, in increasing conservatism, include '"none"',
          '"BH"', '"BY"' and '"holm"'.  See 'p.adjust' for the complete
          list of options. A 'NULL' value will result in the default
          adjustment method, which is '"BH"'.

     The p-values for the coefficient/contrast of interest are adjusted
     for multiple testing by a call to 'p.adjust'. The '"BH"' method,
     which controls the expected false discovery rate (FDR) below the
     specified value, is the default adjustment method because it is
     the most likely to be appropriate for microarray studies. Note
     that the adjusted p-values from this method are bounds on the FDR
     rather than p-values in the usual sense. Because they relate to
     FDRs rather than rejection probabilities, they are sometimes
     called q-values. See 'help("p.adjust")' for more information.

     Note, if there is no good evidence for differential expression in
     the experiment, that it is quite possible for all the adjusted
     p-values to be large, even for all of them to be equal to one. It
     is quite possible for all the adjusted p-values to be equal to one
     if the smallest p-value is no smaller than '1/ngenes' where
     'ngenes' is the number of genes with non-missing p-values.
```

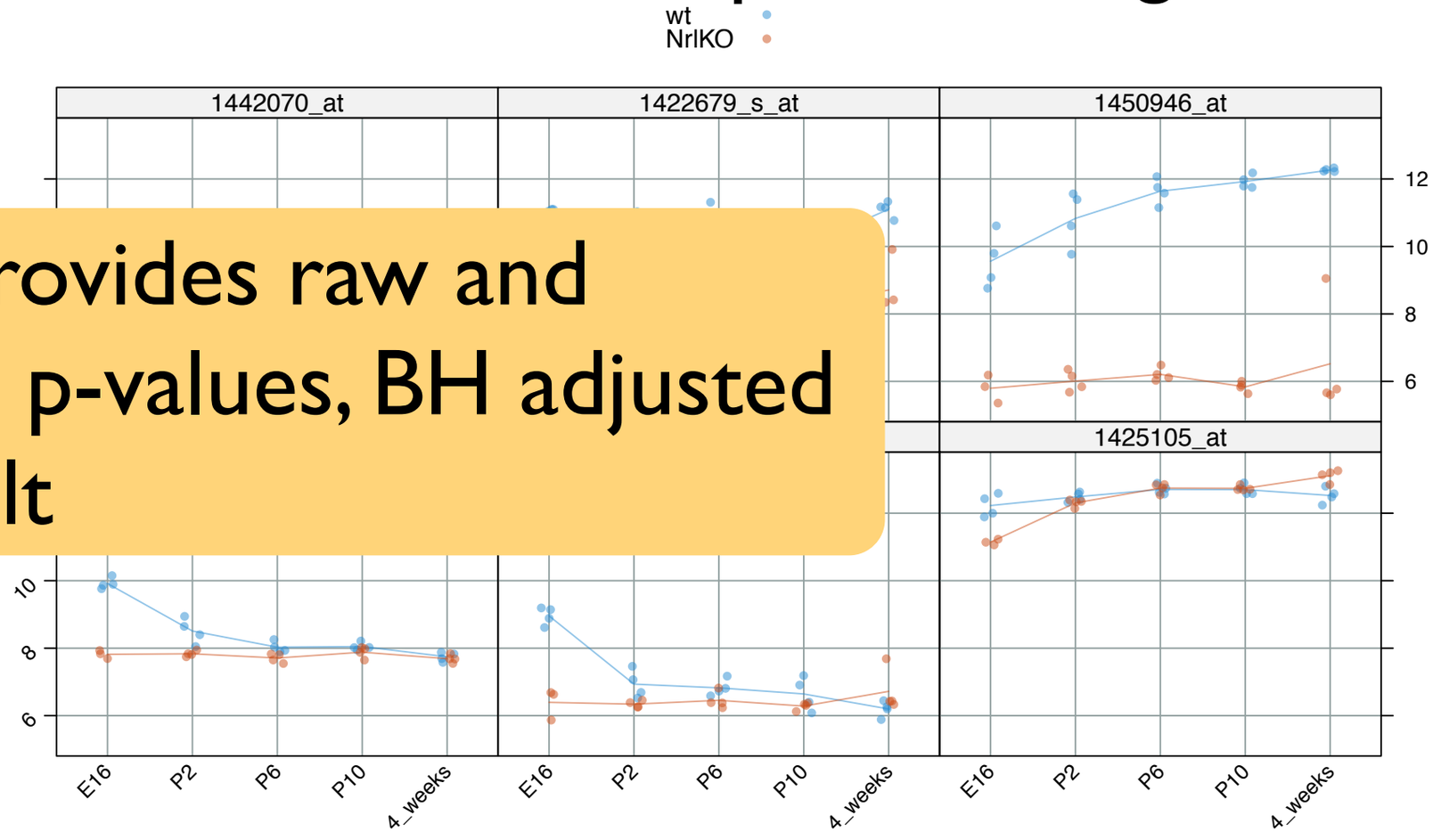
`adjust.method` argument is where you specify how to adjust your p-values for multiple testing

```
> ## this is equivalent: topTable(ebFit, coef = 2)
> ## but much more self-documenting; USE NAMES!
> ## you will be glad you did when you revisit/read your code

> topTable(ebFit, coef = "gTypeNrlKO")
                logFC   AveExpr         t      P.Value    adj.P.Val        B
1455965_at   -2.099167  8.125436 -14.808462 6.702431e-16 2.007311e-11 23.349723
1427256_at   -2.563917  6.784538  -9.572804 6.316787e-11 9.459072e-07 14.131751
1425105_at   -1.084167 12.502308  -7.619877 1.084129e-08 8.070340e-05  9.622431
1442070_at   -1.149417  8.268231  -7.546188 1.328641e-08 8.070340e-05  9.440861
1422679_s_at -2.810333  9.495128  -7.484259 1.577053e-08 8.070340e-05  9.287671
1450946_at   -3.764833  8.733000  -7.421078 1.879238e-08 8.070340e-05  9.130832
1422643_at   -1.813083  7.965949  -7.419730 1.886286e-08 8.070340e-05  9.127482
1452114_s_at -1.702000  6.519538  -7.135296 4.175503e-08 1.563152e-04  8.414647
1424894_at    1.166500  6.961487   6.926615 7.519817e-08 2.384014e-04  7.884984
1448076_at    1.140500  6.453897   6.906506 7.960246e-08 2.384014e-04  7.833658
```

| devStage / gType | E16 | P2 | P6 | P10 | 4_weeks |
|---|---|---|---|---|---|
| wt | $\theta$ | $\beta_{P2}$ | $\beta_{P6}$ | $\beta_{P10}$ | $\beta_{4\_weeks}$ |
| NrlKO | $\tau_{NrlKO}$ | $(\tau\beta)_{NrlKO,P2}$ | $(\tau\beta)_{NrlKO,P6}$ | $(\tau\beta)_{NrlKO,P10}$ | $(\tau\beta)_{NrlKO,4\_weeks}$ |

this finds genes where the knockouts differ from the wild types at the reference developmental stage level E16

limma provides raw and adjusted p-values, BH adjusted by default

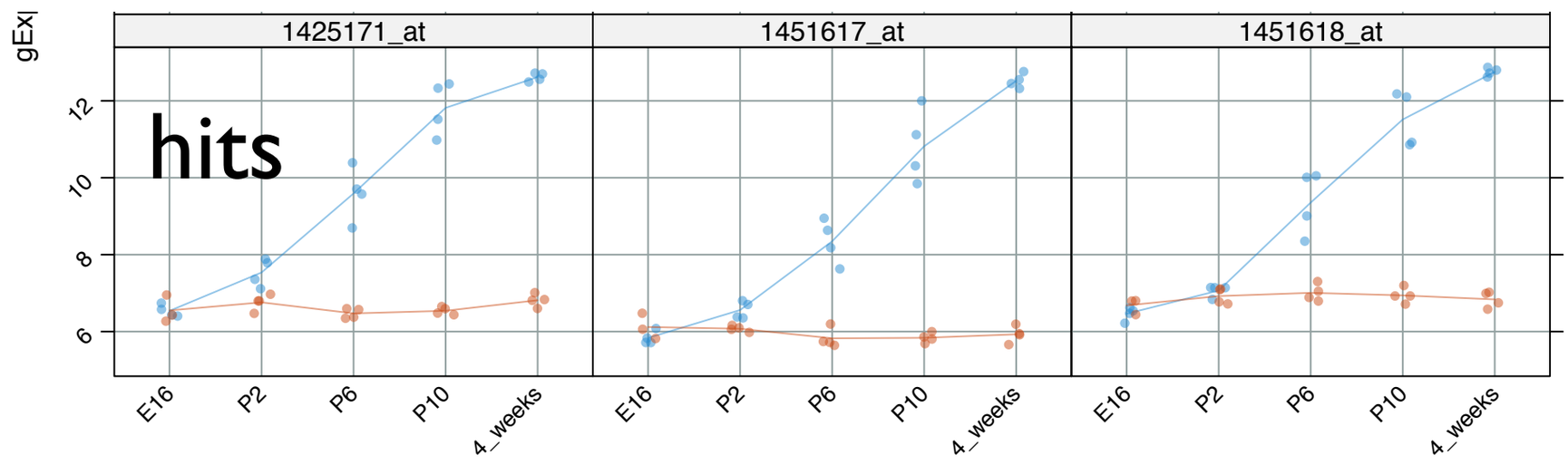


| devStage / gType | E16 | P2 | P6 | P10 | 4_weeks |
|---|---|---|---|---|---|
| wt | $\theta$ | $\beta_{P2}$ | $\beta_{P6}$ | $\beta_{P10}$ | $\beta_{4\_weeks}$ |
| NrlKO | $\tau_{NrlKO}$ | $(\tau\beta)_{NrlKO,P2}$ | $(\tau\beta)_{NrlKO,P6}$ | $(\tau\beta)_{NrlKO,P10}$ | $(\tau\beta)_{NrlKO,4\_weeks}$ |

```
> topTable(ebFit, coef = grep("gType", colnames(coef(ebFit))))
                    ID  gTypeNrlKO gTypeNrlKO.devStageP2 gTypeNrlKO.devStageP6 gTypeNrlKO.devS
1425171_at 1425171_at  0.01225000           -0.78900000           -3.1307500          -5.
1451617_at 1451617_at  0.28408333           -0.77058333           -2.8063333          -5.
1451618_at 1451618_at  0.21608333           -0.35983333           -2.5625833          -4.7895833
1419740_at 1419740_at -0.29666667           -2.07033333           -4.2415833          -4.2063333
1422643_at 1422643_at -1.81308333            3.30833333            5.1363333           5.4715833
1416306_at 1416306_at  0.04941667           -0.02466667            1.0283333           3.1835833
1441809_at 1441809_at  0.24008333           -0.32283333           -0.1543333           0.5314167
1450946_at 1450946_at -3.76483333           -1.05741667           -1.6646667          -2.3219167
1460212_at 1460212_at  0.30625000           -0.39175000           -1.4495000          -4.0167500
1424699_at 1424699_at -0.34500000            0.71750000            1.5500000           3.1162500
           gTypeNrlKO.devStage4_weeks   AveExpr         F      P.Value    adj.P.Val
1425171_at                  -5.814750  8.164641 222.97654 6.137568e-24 1.838140e-19
1451617_at                  -6.874333  7.419538 199.05565 3.577813e-23 5.357596e-19
1451618_at                  -6.132083  8.192077 171.45610 3.597857e-22 3.591741e-18
1419740_at                  -4.214583  8.979256 142.32860 6.282888e-21 4.704155e-17
1422643_at                   4.201083  7.965949 137.48965 1.066043e-20 6.385387e-17
1416306_at                   4.914083  6.559769 127.08218 3.538678e-20 1.766331e-16
1441809_at                   3.734167  6.649615 123.46056 5.491596e-20 2.349540e-16
1450946_at                  -1.977917  8.733000 122.31075 6.329981e-20 2.369707e-16
1460212_at                  -5.762500  8.927436  96.87021 2.129715e-18 6.972448e-15
1424699_at                   2.915000 12.245000  96.29530 2.328107e-18 6.972448e-15
```

this finds genes where genotype makes a difference, somehow, somewhere

# primary p-value adjustment is on the column of ~30K p-values for testing if all these coefficients = 0



but what if you are investigating multiple coefficients contrasts separately but at the same time?

now you have a ‘row-wise’ multiple testing problem on top of the ‘column-wise’ problem posed by the 30K probesets

the ‘row-wise’ multiple testing problem is quite classical -- probably the one most people encounter first in their statistical education

if you tested each of these 6 contrasts for equality with zero for each of your ~30K probesets, should you enact p-value adjustment for all 6 * 30K tests at once? or do you do this separately for the 30K p-values associated with each contrast?

$$\beta_g = C^T \alpha_g$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} \mu_{g1} \\ \mu_{g2} \\ \mu_{g3} \\ \mu_{g4} \end{bmatrix} = \begin{bmatrix} \mu_{g2} - \mu_{g1} \\ \mu_{g3} - \mu_{g1} \\ \mu_{g4} - \mu_{g1} \\ \mu_{g3} - \mu_{g2} \\ \mu_{g4} - \mu_{g2} \\ \mu_{g4} - \mu_{g3} \end{bmatrix}$$

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4 groups
"cell means" or "groups means" parametrization
4 parameters in linear model

$$C^T \begin{bmatrix} \mu_{g1} \\ \mu_{g2} \\ \mu_{g3} \\ \mu_{g4} \end{bmatrix} = \beta_g$$

## Less silly example that shows why one might want to form contrasts.

Imagine you are interested in all six pairwise comparisons between groups.
No valid parametrization of the linear model will deliver that (too many parameters; linear dependence of those parameters).
Therefore, forming contrasts is unavoidable.

classical procedures for this include ...
Tukey Multiple Comparison Procedure
Scheffe Multiple Comparison Procedure
Bonferroni, Holm also used here
.... those aren't frequently used in our setting

The Path of Least Resistance is to use limma's decideTests() function or, more generically, the qvalue package

basic idea of limma's decideTests(..., method = "global"): all the p-values, across the various coefficients or contrasts, are combined, adjusted globally, then separated back out and re-sorted, then threshholded

I have worked an example using the photoRec data but, upon revisiting, I've decided it's cruel to inflict it on you in lecture. Insight : ugliness ratio is too low.

You can find the code here:
https://github.com/jennybc/stat540_2014/blob/master/examples/photoRec/code/48_lectSupp-multiple-testing.r

and I've left some slides about it here, in case you want to go through this on your own.

last in-lecture example:

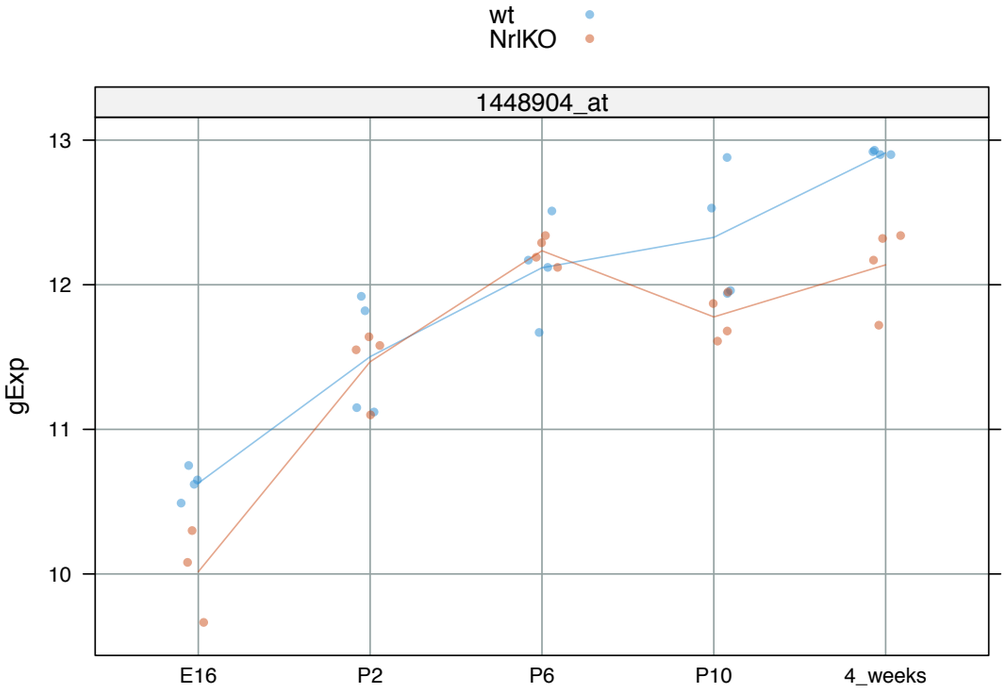
photoRec dataset

two-way ANOVA
parametrization: “cell means”

form these contrasts:
 wt.E16 - wt.P10
 NrlKO.E16 - NrlKO.P10

do inference (i.e. test for equality with zero) on each separately with topTable() or decideTests(..., method = “separate”)

do inference on them jointly with decideTests(..., method = “global”)




# two-way ANOVA parametrization: "cell means"

# lm() for one probeset, just for concreteness

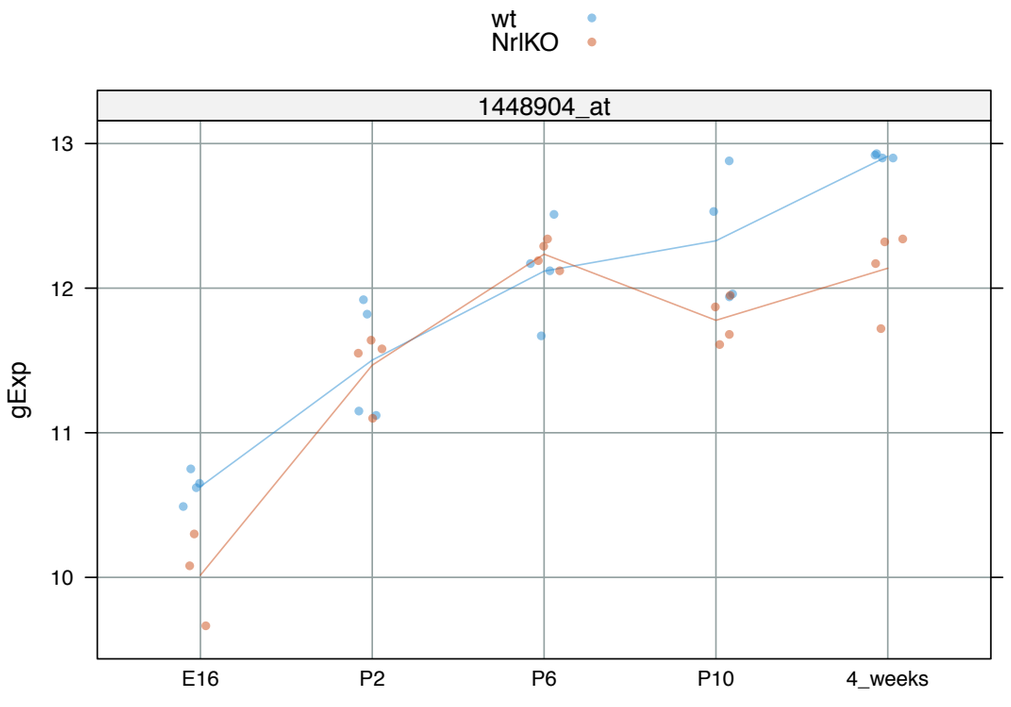
```
> stripplotIt(tinyDat <- prepareData("1448904_at"))
> tinyDat$grp <- with(tinyDat, interaction(gType, devStage))
> summary(lm(gExp ~ grp + 0, tinyDat))

Call:
lm(formula = gExp ~ grp + 0, data = tinyDat)

Residuals:
     Min       1Q   Median       3Q      Max
-0.4475 -0.1263  0.0225  0.1475  0.5525

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
grpwt.E16         10.6275     0.1413   75.20   <2e-16 ***
grpNrlKO.E16      10.0150     0.1632   61.38   <2e-16 ***
grpwt.P2          11.5025     0.1413   81.40   <2e-16 ***
grpNrlKO.P2       11.4675     0.1413   81.15   <2e-16 ***
grpwt.P6          12.1175     0.1413   85.75   <2e-16 ***
grpNrlKO.P6       12.2350     0.1413   86.58   <2e-16 ***
grpwt.P10         12.3275     0.1413   87.23   <2e-16 ***
grpNrlKO.P10      11.7775     0.1413   83.34   <2e-16 ***
grpwt.4_weeks     12.9125     0.1413   91.38   <2e-16 ***
grpNrlKO.4_weeks  12.1375     0.1413   85.89   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2826 on 29 degrees of freedom
Multiple R-squared: 0.9996,  Adjusted R-squared: 0.9994
F-statistic:  6777 on 10 and 29 DF,  p-value: < 2.2e-16
```

```
> prDes$grp <- with(prDes, interaction(gType, devStage))
> kDesMat <- model.matrix(~ grp + 0, prDes)
> kFit <- lmFit(prDat, kDesMat)
```

fits same model, with same parametrization,
but for each of the 30K probesets individually

$$Y_g = X\alpha_g + \varepsilon_g$$

$$\begin{bmatrix} y_{g1} \\ y_{g2} \\ \vdots \\ y_{gn_g} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \vdots & 0 & 0 \\ 0 & 1 & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \vdots & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \vdots & 1 & 0 \\ 0 & 0 & \vdots & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \vdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \mu_{g,wt,E16} \\ \mu_{g,\Delta Nrl,E16} \\ \mu_{g,wt,P2} \\ \mu_{g,\Delta Nrl,P2} \\ \mu_{g,wt,P6} \\ \mu_{g,\Delta Nrl,P6} \\ \mu_{g,wt,P10} \\ \mu_{g,\Delta Nrl,P10} \\ \mu_{g,wt,4\_weeks} \\ \mu_{g,\Delta Nrl,4\_weeks} \end{bmatrix} + \begin{bmatrix} \varepsilon_{g1} \\ \varepsilon_{g2} \\ \vdots \\ \varepsilon_{gn_g} \end{bmatrix}$$



```
> colnames(coef(kFit))                    # parameters in the linear model
 [1] "grpwt.E16"         "grpNrlKO.E16"      "grpwt.P2"          "grpNrlKO.P2"       "grpwt.P6"
 [6] "grpNrlKO.P6"       "grpwt.P10"         "grpNrlKO.P10"      "grpwt.4_weeks"     "grpNrlKO.4_weeks"
> (cont.matrix <- makeContrasts(
+     wt.P10vsE16 = grpwt.P10 - grpwt.E16,
+     NrlKO.P10vsE16 = grpNrlKO.P10 - grpNrlKO.E16,
+     levels = kDesMat))
                 Contrasts
Levels            wt.P10vsE16 NrlKO.P10vsE16
  grpwt.E16                -1              0
  grpNrlKO.E16              0             -1
  grpwt.P2                  0              0
  grpNrlKO.P2               0              0
  grpwt.P6                  0              0
  grpNrlKO.P6               0              0
  grpwt.P10                 1              0
  grpNrlKO.P10              0              1
  grpwt.4_weeks             0              0
  grpNrlKO.4_weeks          0              0
> kFitCont <- contrasts.fit(kFit, cont.matrix)
```

form these contrasts:
wt.E16 - wt.P10
NrlKO.E16 - NrlKO.P10

$$C^T \alpha_g = \beta_g$$

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mu_{g,wt,E16} \\ \mu_{g,\Delta Nrl,E16} \\ \mu_{g,wt,P2} \\ \mu_{g,\Delta Nrl,P2} \\ \mu_{g,wt,P6} \\ \mu_{g,\Delta Nrl,P6} \\ \mu_{g,wt,P10} \\ \mu_{g,\Delta Nrl,P10} \\ \mu_{g,wt,4\_weeks} \\ \mu_{g,\Delta Nrl,4\_weeks} \end{bmatrix} = \begin{bmatrix} \mu_{g,wt,P10} - \mu_{g,wt,E16} \\ \mu_{g,\Delta Nrl,P10} - \mu_{g,\Delta Nrl,E16} \end{bmatrix}$$

```
> kebFit <- eBayes(kFitCont)

> cutoff <- 1e-4

> ## use topTable to get separate hits for each of the contrasts
> wtHits <-
+     topTable(kebFit, coef = "wt.P10vsE16", p.value = cutoff, n = Inf)

> NrlKOHits <-
+     topTable(kebFit, coef = "NrlKO.P10vsE16", p.value = cutoff, n = Inf)

> nrow(wtHits)                                  # 349
[1] 349

> nrow(NrlKOHits)                               # 196
[1] 196

> nrow(wtHits) + nrow(NrlKOHits)                # 545 total if no overlap
[1] 545
```

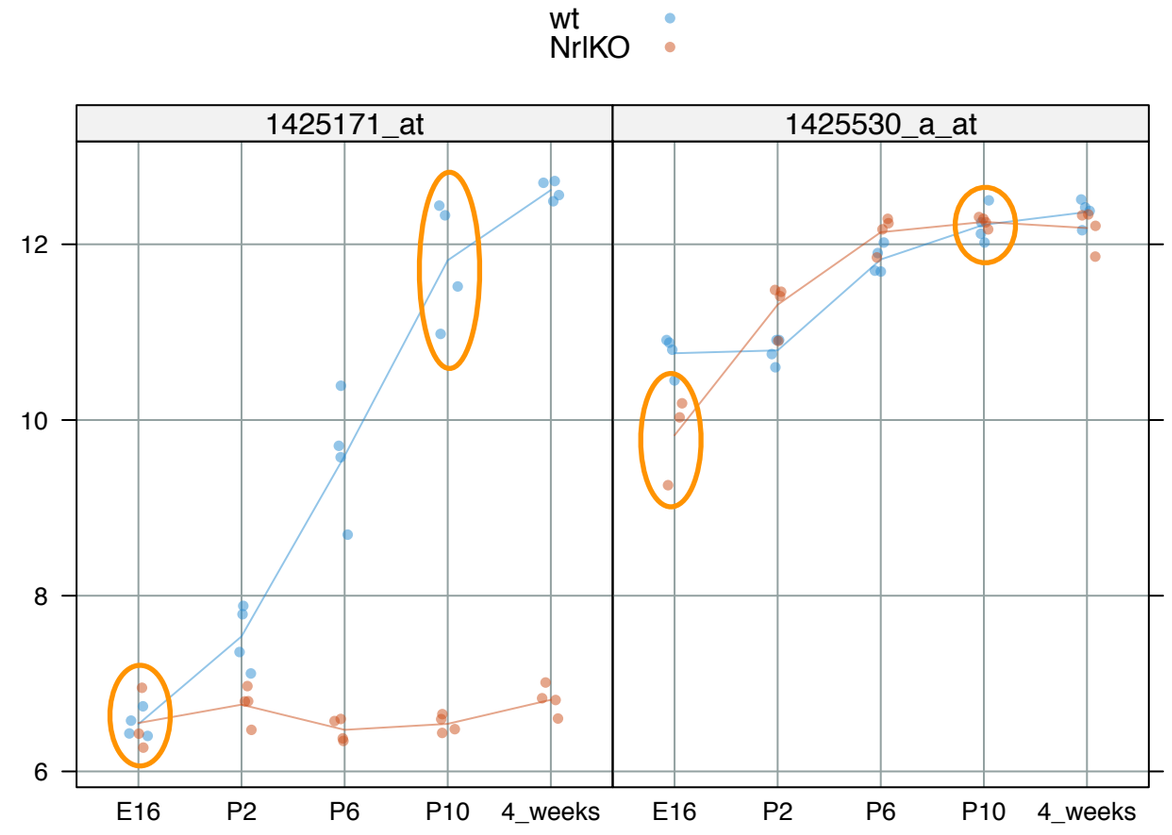
do inference (i.e. test for equality with zero) on each separately with topTable()

```
> sepHits <- union(wtHits$ID, NrlKOHits$ID)

> length(sepHits)                               # 482 actual total
[1] 482

> head(wtHits, 4)
             ID   logFC  AveExpr        t      P.Value    adj.P.Val        B
6793  1425171_at 5.27875 8.164641 20.79608 3.404254e-20 1.019540e-15 33.99699
24047 1451618_at 5.05075 8.192077 18.85996 6.227101e-19 9.324772e-15 31.59672
24046 1451617_at 4.98375 7.419538 18.16635 1.876032e-18 1.872842e-14 30.66463
3117  1419740_at 5.09150 8.979256 16.60568 2.561785e-17 1.918073e-13 28.41252
> head(NrlKOHits, 4)
              ID    logFC   AveExpr        t      P.Value    adj.P.Val        B
7034 1425530_a_at 2.429000 11.613282 13.93584 3.649869e-15 9.173265e-11 23.84590
4843   1422643_at 3.342083  7.965949 13.67690 6.125924e-15 9.173265e-11 23.38211
3966   1421084_at 5.095000 10.022641 13.14200 1.826153e-14 1.428258e-10 22.39805
550    1416306_at 3.358833  6.559769 13.01881 2.358851e-14 1.428258e-10 22.16634
```

do inference (i.e. test for equality with zero) on each separately with topTable()

```
> head(wtHits, 4)
              ID   logFC  AveExpr         t      P.Value    adj.P.Val        B
6793  1425171_at 5.27875 8.164641 20.79608 3.404254e-20 1.019540e-15 33.99699
24047 1451618_at 5.05075 8.192077 18.85996 6.227101e-19 9.324772e-15 31.59672
24046 1451617_at 4.98375 7.419538 18.16635 1.876032e-18 1.872842e-14 30.66463
3117  1419740_at 5.09150 8.979256 16.60568 2.561785e-17 1.918073e-13 28.41252
> head(NrlKOHits, 4)
               ID    logFC   AveExpr         t      P.Value    adj.P.Val        B
7034 1425530_a_at 2.429000 11.613282 13.93584 3.649869e-15 9.173265e-11 23.84590
4843   1422643_at 3.342083  7.965949 13.67690 6.125924e-15 9.173265e-11 23.38211
3966   1421084_at 5.095000 10.022641 13.14200 1.826153e-14 1.428258e-10 22.39805
550    1416306_at 3.358833  6.559769 13.01881 2.358851e-14 1.428258e-10 22.16634
```

top hit for P10 - E16 in wt




top hit for P10 - E16 in NrlKO

# do inference on them jointly with decideTests()

```
> ## use decideTests, optionally with method = "global"
> kRes <- decideTests(kebFit, p.value = cutoff)

> summary(kRes)
   wt.P10vsE16 NrlKO.P10vsE16
-1         256             52
0        29600          29753
1           93            144

> ## verifying that the separate hits obtained via two topTable calls
> ## and via one decideTests call are the same
> length(kHitsSep <- which(rowSums(abs(kRes)) > 0)) # 482
[1] 482

> all(sort(rownames(kRes[kHitsSep, ])) == sort(sepHits)) # TRUE
[1] TRUE

> peek(kRes[kHitsSep, ])
             Contrasts
              wt.P10vsE16 NrlKO.P10vsE16
1421301_at             -1              0
1425100_a_at            1              1
1437588_at             -1              0
1440256_at              0              1
1441693_at              0              1
1448157_s_at           -1              0
1457558_at              1              0
```

method = ‘separate’ is the default

does p-value adjustment for each coefficient or contrast separately

same hits as topTable()

# do inference on them jointly with decideTests()

```
> ## use decideTests, optionally with method = "global"
> kResGlobal <- decideTests(kebFit, p.value = cutoff, method = "global")

> summary(kResGlobal)
   wt.P10vsE16 NrlKO.P10vsE16
-1         246             61
0        29615          29735
1           88            153

> ## how many "global" hits are there?
> length(kHitsGlobal <- which(rowSums(abs(kResGlobal)) > 0)) # 485
[1] 485

> peek(kResGlobal[kHitsGlobal, ])
           Contrasts
            wt.P10vsE16 NrlKO.P10vsE16
1422605_at            0             -1
1426858_at           -1              0
1429269_at           -1              0
1433761_at            0              1
1436465_at           -1             -1
1455515_at           -1              0
1460368_at            0              1
```

method = ‘global’ does p-value adjustment for all coefficients and/or contrast jointly

in general, not the same hits

in fact, no guarantee whether this will yield more or fewer hits!

# do inference on them jointly with decideTests()

```
> nrow(wtHits)                                  # 349
[1] 349

> sum(abs(kResGlobal[ ,"wt.P10vsE16"]))
[1] 334

> vennCounts(cbind(separate = wtAllUnsrtd$adj.P.Val < cutoff,
+                  global = kResGlobal[ ,"wt.P10vsE16"]),
+             include = "both")
     separate global Counts
[1,]        0      0  29600
[2,]        0      1      0
[3,]        1      0     15
[4,]        1      1    334
```

for the P10 vs E16 contrast for wild type ....
separate inference finds more hits,
i.e. 349 vs. 334 probes have an adjusted p-value less than 1e-4

# do inference on them jointly with decideTests()

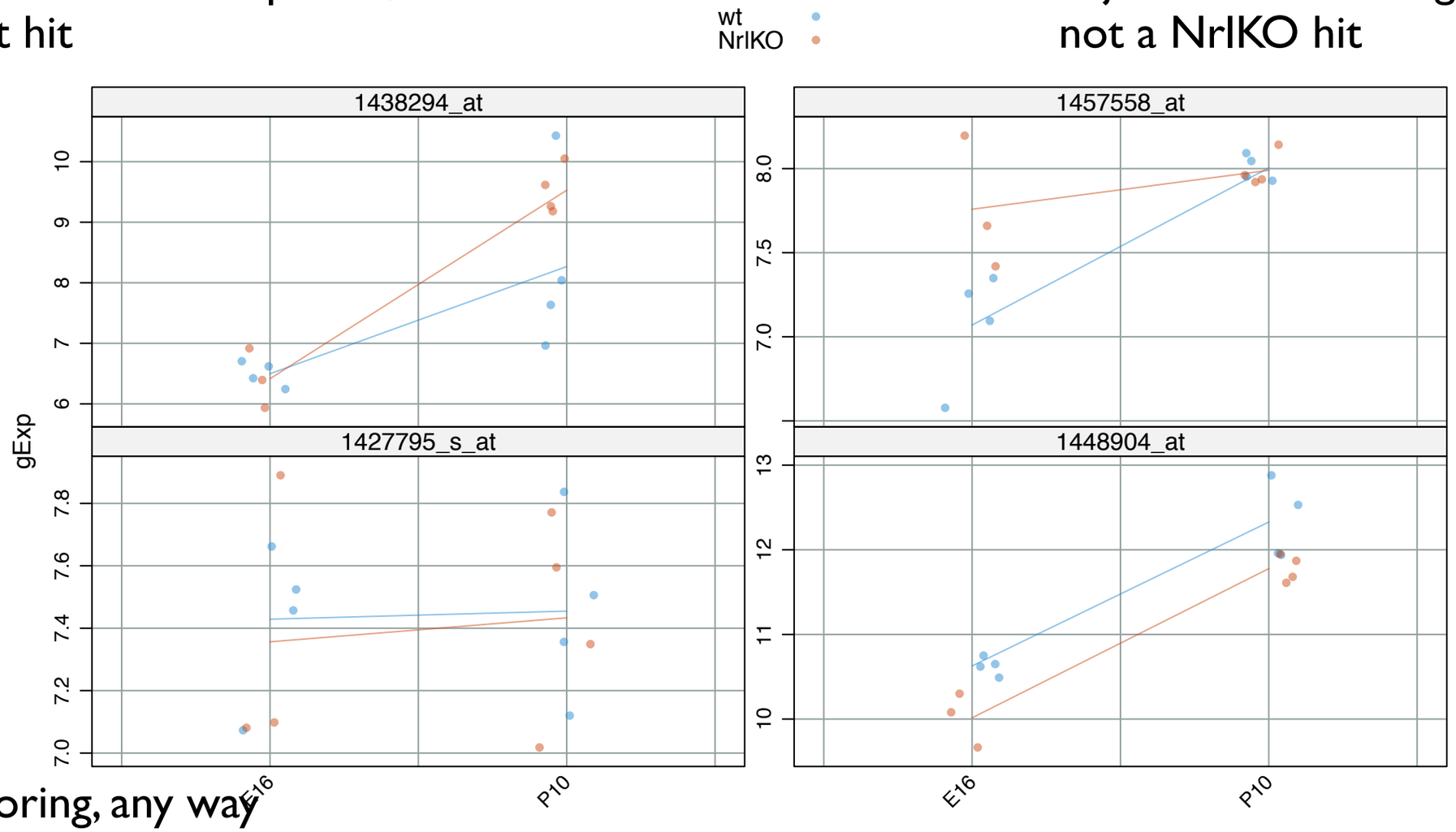
```
> nrow(NrlKOHits)                                  # 196
[1] 196

> sum(abs(kResGlobal[ ,"NrlKO.P10vsE16"]))
[1] 214

> vennCounts(cbind(separate = NrlKOAllUnsrtd$adj.P.Val < cutoff,
+                  global = kResGlobal[ ,"NrlKO.P10vsE16"]),
+             include = "both")
     separate global Counts
[1,]        0      0  29735
[2,]        0      1     18
[3,]        1      0      0
[4,]        1      1    196
```

on the other hand,

for the P10 vs E16 contrast for knockouts ....
global inference finds more hits, i.e. 214 vs. 196 probes have an adjusted p-value less than 1e-4

go figure!

hit for NrlKO with global adjustment but not separate, not a wt hit




boring, any way you look at

hit for wt with separate adjustment but not global, not a NrlKO hit

hit for wt, hit for NrlKO, any way you look at it

## 13.3 Multiple Testing Across Contrasts

The output from `topTable` includes adjusted $p$-values, i.e., it performs multiple testing for the contrast being considered. If several contrasts are being tested simultaneously, then the issue arises of multiple testing for the entire set of hypotheses being considered, across contrasts as well as probes. The function `decideTests()` offers a number of strategies for doing this.

The simplest multiple testing method is `method="separate"`. This method does multiple testing for each contrast separately. This method is the default because it is equivalent to using `topTable()`. Using this method, testing a set of contrasts together will give the same results as when each contrast is tested on its own. The great advantage of this method is that it gives the same results regardless of which set of contrasts are tested together. The disadvantage of this method is that it does not do any multiple testing adjustment between contrasts. Another disadvantage is that the raw $p$-value cutoff corresponding to significance can be very different for different contrasts, depending on the number of DE probes. This method is recommended when different contrasts are being analysed to answer more or less independent questions.

`method="global"` is recommended when a set of closely related contrasts are being tested. This method simply appends all the tests together into one long vector of tests, i.e., it treats all the tests as equivalent regardless of which probe or contrast they relate to. An advantage is that the raw $p$-value cutoff is consistent across all contrasts. For this reason, `method="global"` is recommended if you want to compare the number of DE genes found for different contrasts, for example interpreting the number of DE genes as representing the strength of the contrast. However users need to be aware that the number of DE genes for any particular contrasts will depend on which other contrasts are tested at the same time. Hence one should include only those contrasts which are closely related to the question at hand. Unnecessary contrasts should be excluded as these would affect the results for the contrasts of interest. Another more theoretical issue is that there is no theorem which proves that `adjust.method="BH"` in combination with `method="global"` will correctly control the false discovery rate for combinations of negatively correlated contrasts, however simulations, experience and some theory suggest that the method is safe in practice.

The `"hierarchical"` method offers power advantages when used with `adjust.method="holm"`

for further information, read 13.3 Multiple Testing Across Contrasts in limma User's Guide (p. 62-3)

gives less conservative results when finding probes which respond to several different contrasts at once. However this method should still be viewed as experimental. It provides formal false discovery rate control at the probe level only, not at the contrast level.

```
# find genes where expression changes over development different
for wild type and knockout

> jDesMat <- model.matrix(~ gType * devStage, prDes)

> jFit <- lmFit(prDat, jDesMat)

> ebFit <- eBayes(jFit)

> hits <- topTable(ebFit, coef = grep(":", colnames(coef(ebFit))))
```

**the code above takes a couple of second to execute**

**computation is NOT your problem**

**your time and energy will be devoted to choosing the model, choosing how to parametrize it and digesting the results**