QCB 508 – Week 12

John D. Storey

Spring 2020

Contents

HD Latent Variable Models

Definition

Latent variables (or hidden variables) are random variables that are present in the underlying probabilistic model of the data, but they are unobserved.

In high-dimensional data, there may be latent variables present that affect many variables simultaneously.

These are latent variables that induce **systematic variation**. A topic of much interest is how to estimate these and incorporate them into further HD inference procedures.

Model

Suppose we have observed data $\boldsymbol{Y}_{m\times n}$ of m variables with n observations each. Suppose there are r latent variables contained in the r rows of $\boldsymbol{Z}_{r\times n}$ where

$$\mathrm{E}\left[\boldsymbol{Y}_{m\times n} \mid \boldsymbol{Z}_{r\times n}\right] = \boldsymbol{\Phi}_{m\times r}\boldsymbol{Z}_{r\times n}.$$

Let's also assume that $m \gg n > r$. The latent variables $\boldsymbol{Z}$ induce systematic variation in variable $\boldsymbol{y}_i$ parameterized by $\boldsymbol{\phi}_i$ for $i = 1, 2, \ldots, m$.

Estimation

There exist methods for estimating the row space of $\boldsymbol{Z}$ with probability 1 as $m \to \infty$ for a fixed n in two scenarios.

Leek (2011) shows how to do this when $\boldsymbol{y}_i|\boldsymbol{Z} \sim \text{MVN}(\boldsymbol{\phi}_i\boldsymbol{Z}, \sigma_i^2\boldsymbol{I})$, and the $\boldsymbol{y}_i|\boldsymbol{Z}$ are jointly independent.

Chen and Storey (2015) show how to do this when the $\boldsymbol{y}_i|\boldsymbol{Z}$ are distributed according to a single parameter exponential family distribution with mean $\boldsymbol{\phi}_i\boldsymbol{Z}$, and the $\boldsymbol{y}_i|\boldsymbol{Z}$ are jointly independent.

Jackstraw

Suppose we have a reasonable method for estimating $\boldsymbol{Z}$ in the model

$$\text{E}\left[\boldsymbol{Y} \mid \boldsymbol{Z}\right] = \boldsymbol{\Phi}\boldsymbol{Z}.$$

The **jackstraw** method allows us to perform hypothesis tests of the form

$$H_0 : \boldsymbol{\phi}_i = \boldsymbol{0} \text{ vs } H_1 : \boldsymbol{\phi}_i \neq \boldsymbol{0}.$$

We can also perform this hypothesis test on any subset of the columns of $\boldsymbol{\Phi}$.

This is a challening problem because we have to "double dip" in the data $\boldsymbol{Y}$, first to estimate $\boldsymbol{Z}$, and second to perform significance tests on $\boldsymbol{\Phi}$.

Procedure

The first step is to form estimate $\hat{\boldsymbol{Z}}$ and then test statistic t_i that performs the hypothesis test for each $\boldsymbol{\phi}_i$ from $\boldsymbol{y}_i$ and $\hat{\boldsymbol{Z}}$ $(i = 1, \ldots, m)$. Assume that the larger t_i is, the more evidence there is against the null hypothesis in favor of the alternative.

Next we randomly select s rows of $\boldsymbol{Y}$ and permute them to create data set $\boldsymbol{Y}^0$. Let this set of s variables be indexed by $\mathcal{S}$. This breaks the relationship between $\boldsymbol{y}_i$ and $\boldsymbol{Z}$, thereby inducing a true H_0, for each $i \in \mathcal{S}$.

We estimate $\hat{\boldsymbol{Z}}^0$ from $\boldsymbol{Y}^0$ and again obtain test statistics t_i^0. Specifically, the test statistics t_i^0 for $i \in \mathcal{S}$ are saved as draws from the null distribution.

We repeat permutation procedure B times, and then utilize all saved sB permutation null statistics to calculate empirical p-values:

$$p_i = \frac{1}{sB} \sum_{b=1}^{B} \sum_{k \in \mathcal{S}_b} 1\left(t_k^{0b} \geq t_i\right).$$

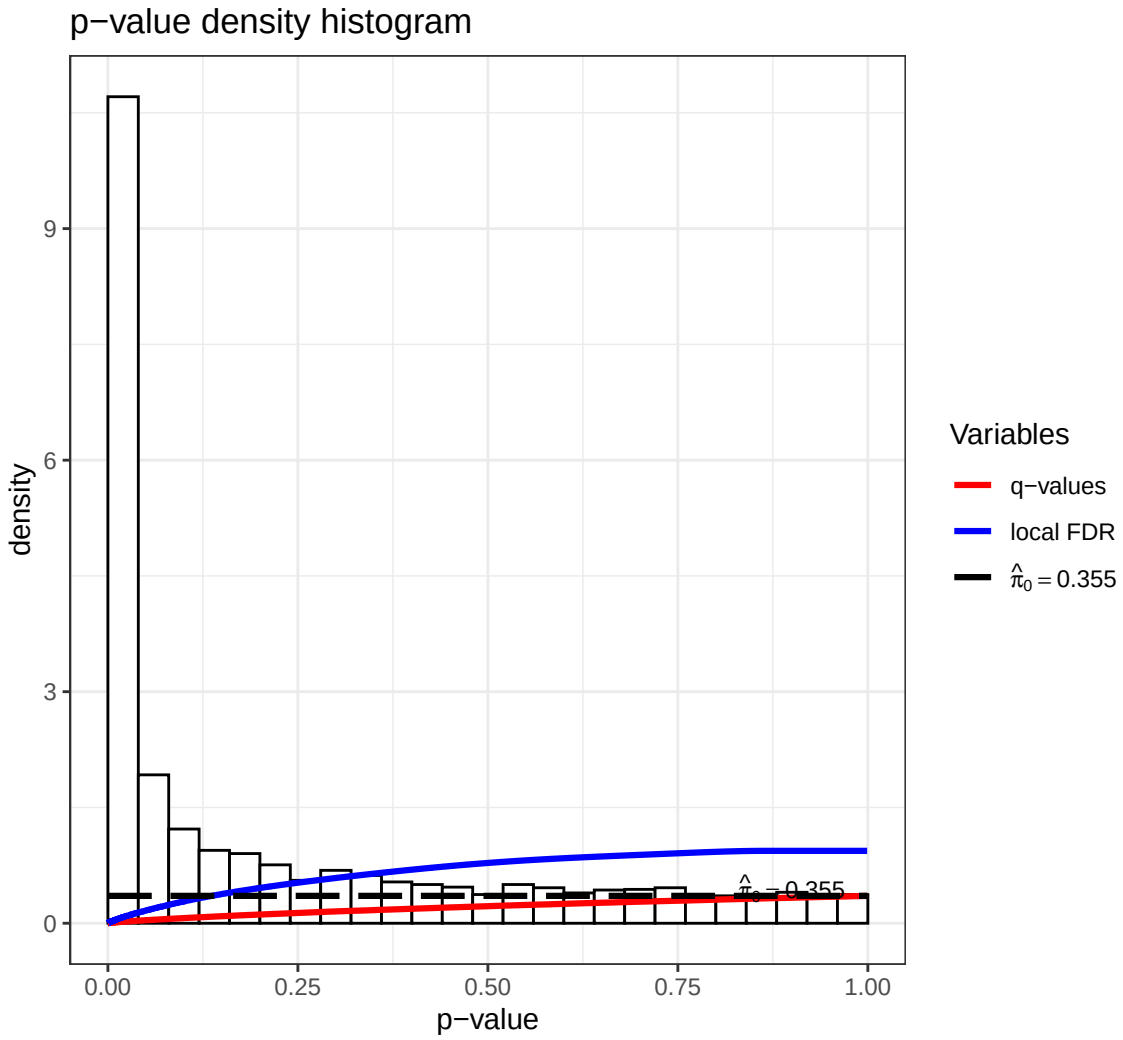
Example: Yeast Cell Cycle

Recall the yeast cell cycle data from earlier. We will test which genes have expression significantly associated with PC1 and PC2 since these both capture cell cycle regulation.

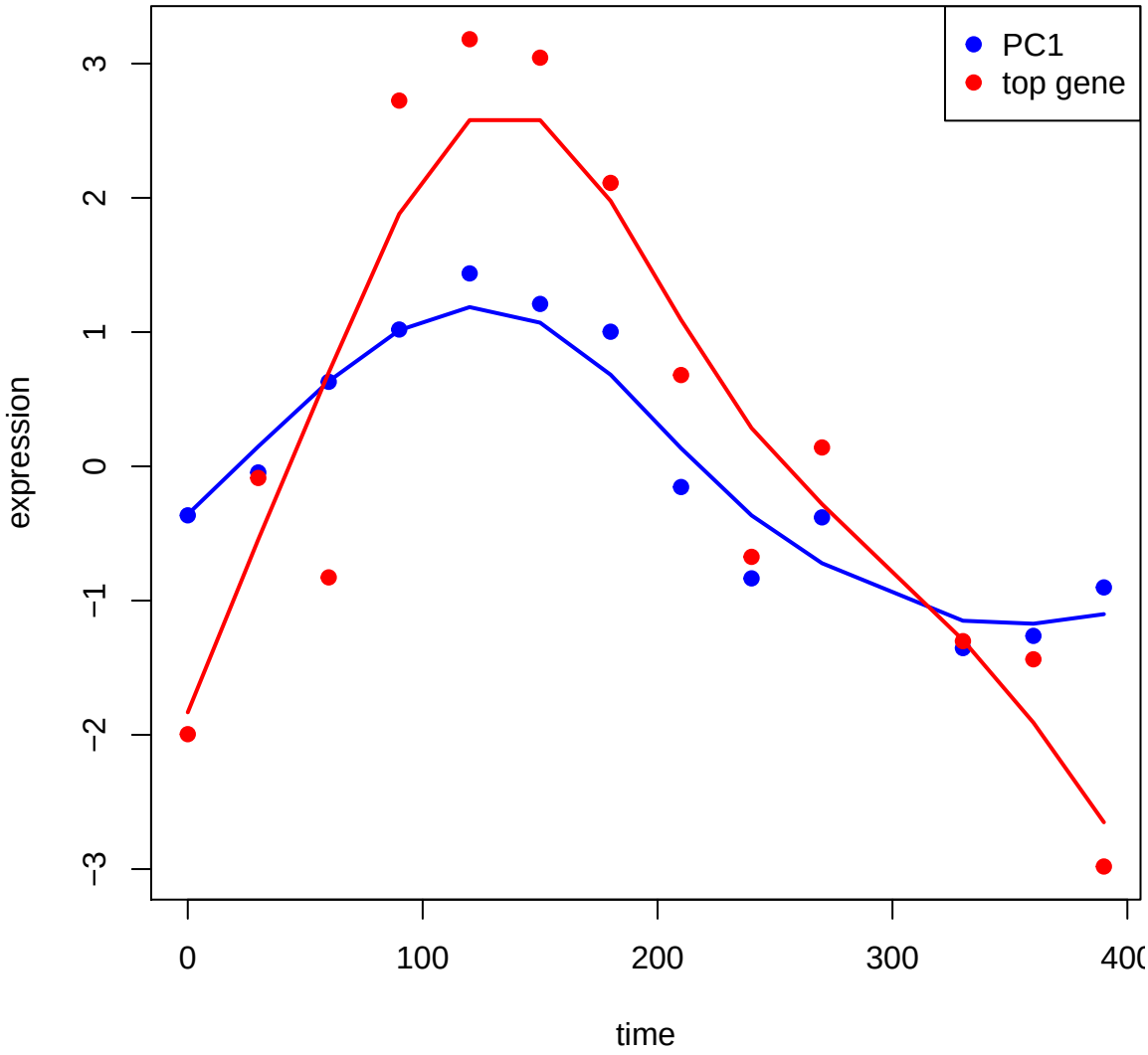
```
> library(jackstraw)
> load("./data/spellman.RData")
> time
 [1]   0  30  60  90 120 150 180 210 240 270 330 360 390
> dim(gene_expression)
[1] 5981   13
> dat <- t(scale(t(gene_expression), center=TRUE, scale=FALSE))
```

Test for associations between PC1 and each gene, conditioning on PC1 and PC2 being relevant sources of systematic variation.

```
> jsobj <- jackstraw_pca(dat, r1=1, r=2, B=500, s=50, verbose=FALSE)
> jsobj$p.value %>% qvalue() %>% hist()
```

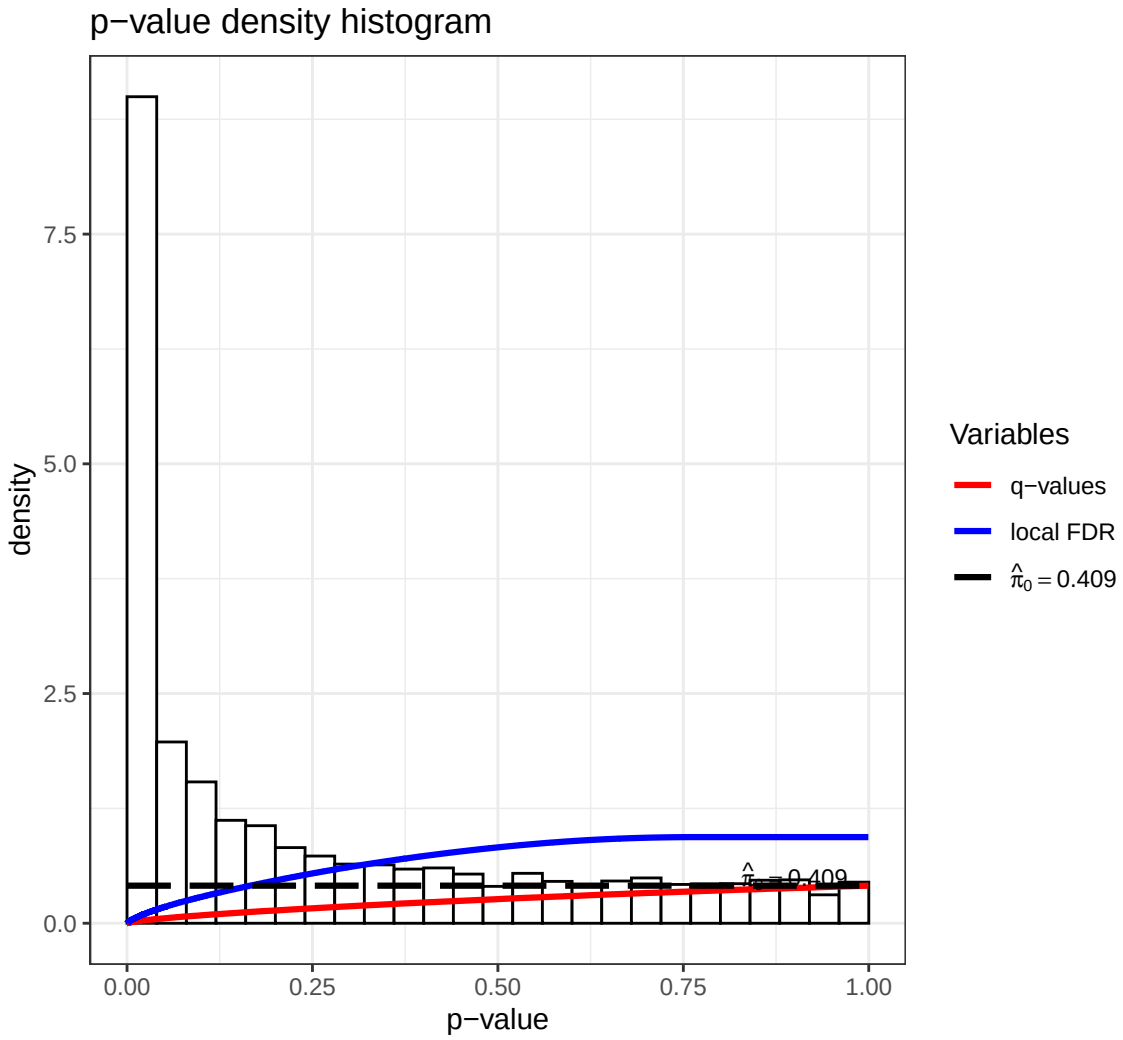


This is the most significant gene plotted with PC1.

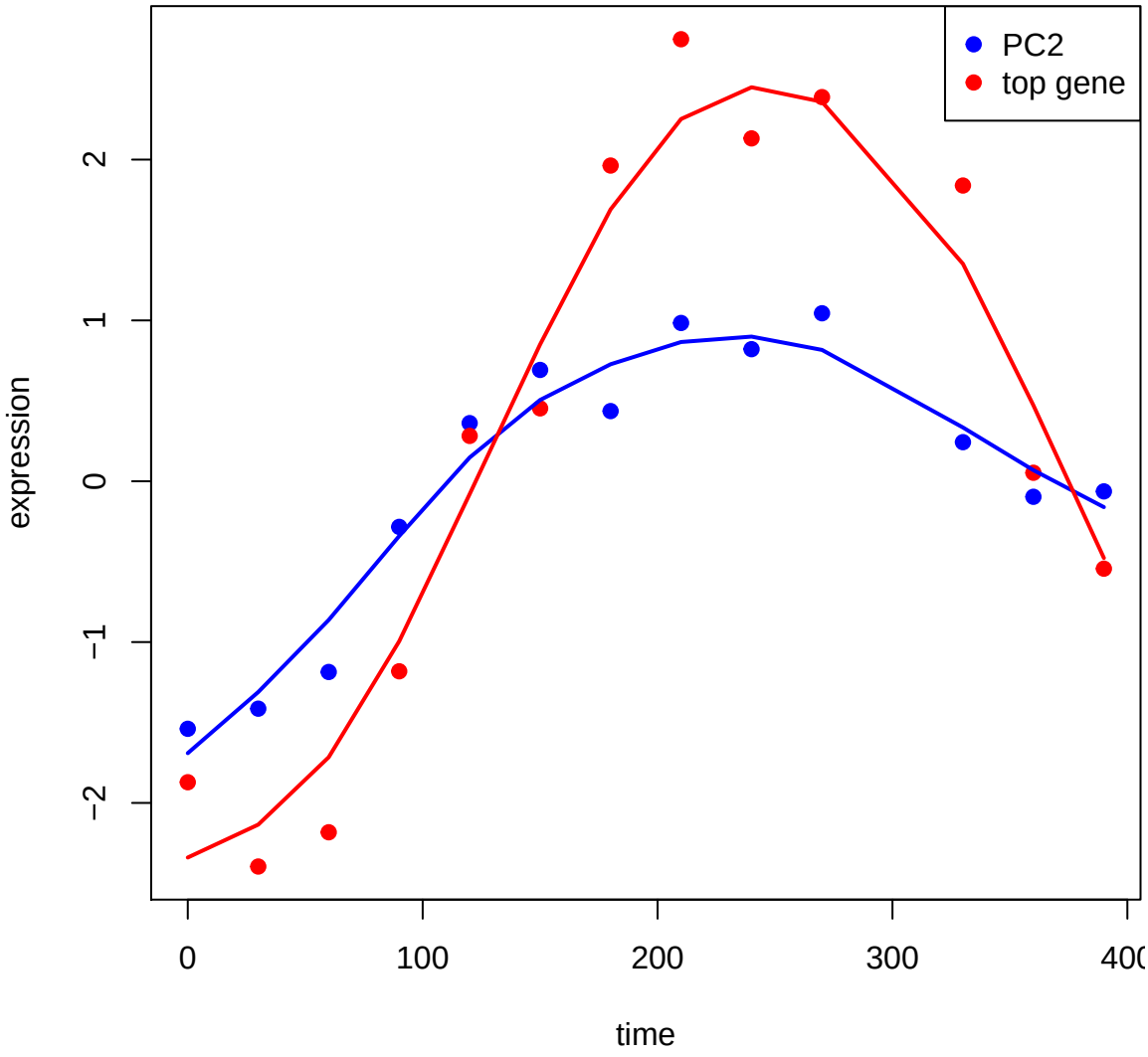


Test for associations between PC2 and each gene, conditioning on PC1 and PC2 being relevant sources of systematic variation.

```
> jsobj <- jackstraw_pca(dat, r1=2, r=2, B=500, s=50, verbose=FALSE)
> jsobj$p.value %>% qvalue() %>% hist()
```



This is the most significant gene plotted with PC2.



Surrogate Variable Analysis

The **surrogate variable analysis** (SVA) model combines the many responses model with the latent variable model introduced above:

$$\boldsymbol{Y}_{m\times n} = \boldsymbol{B}_{m\times d}\boldsymbol{X}_{d\times n} + \boldsymbol{\Phi}_{m\times r}\boldsymbol{Z}_{r\times n} + \boldsymbol{E}_{m\times n}$$

where $m \gg n > d + r$.

Here, only $\boldsymbol{Y}$ and $\boldsymbol{X}$ are observed, so we must combine many regressors model fitting techniques with latent variable estimation.

The variables $\boldsymbol{Z}$ are called **surrogate variables** for what would be a complete model of all systematic variation.

Procedure

The main challenge is that the row spaces of $\boldsymbol{X}$ and $\boldsymbol{Z}$ may overlap. Even when $\boldsymbol{X}$ is the result of a randomized experiment, there will be a high probability that the row spaces of $\boldsymbol{X}$ and $\boldsymbol{Z}$ have some overlap.

Therefore, one cannot simply estimate $\boldsymbol{Z}$ by applying a latent variable esitmation method on the residuals $\boldsymbol{Y} - \hat{\boldsymbol{B}}\boldsymbol{X}$ or on the observed response data $\boldsymbol{Y}$. In the former case, we will only estimate $\boldsymbol{Z}$ in the space orthogonal to $\hat{\boldsymbol{B}}\boldsymbol{X}$. In the latter case, the estimate of $\boldsymbol{Z}$ may modify the signal we can estimate in $\boldsymbol{B}\boldsymbol{X}$.

A recent method, takes an EM approach to esitmating $\boldsymbol{Z}$ in the model

$$\boldsymbol{Y}_{m\times n} = \boldsymbol{B}_{m\times d}\boldsymbol{X}_{d\times n} + \boldsymbol{\Phi}_{m\times r}\boldsymbol{Z}_{r\times n} + \boldsymbol{E}_{m\times n}.$$

It is shown to be necessary to penalize the likelihood in the estimation of $\boldsymbol{B}$ — i.e., form shrinkage estimates of $\boldsymbol{B}$ — in order to properly balance the row spaces of $\boldsymbol{X}$ and $\boldsymbol{Z}$.

The regularized EM algorithm, called **cross-dimensonal inference** (CDI) iterates between

1. Estimate $\boldsymbol{Z}$ from $\boldsymbol{Y} - \hat{\boldsymbol{B}}^{\text{Reg}}\boldsymbol{X}$
2. Estimate $\boldsymbol{B}$ from $\boldsymbol{Y} - \hat{\boldsymbol{\Phi}}\hat{\boldsymbol{Z}}$

where $\hat{\boldsymbol{B}}^{\text{Reg}}$ is a regularized or shrunken estimate of $\boldsymbol{B}$.

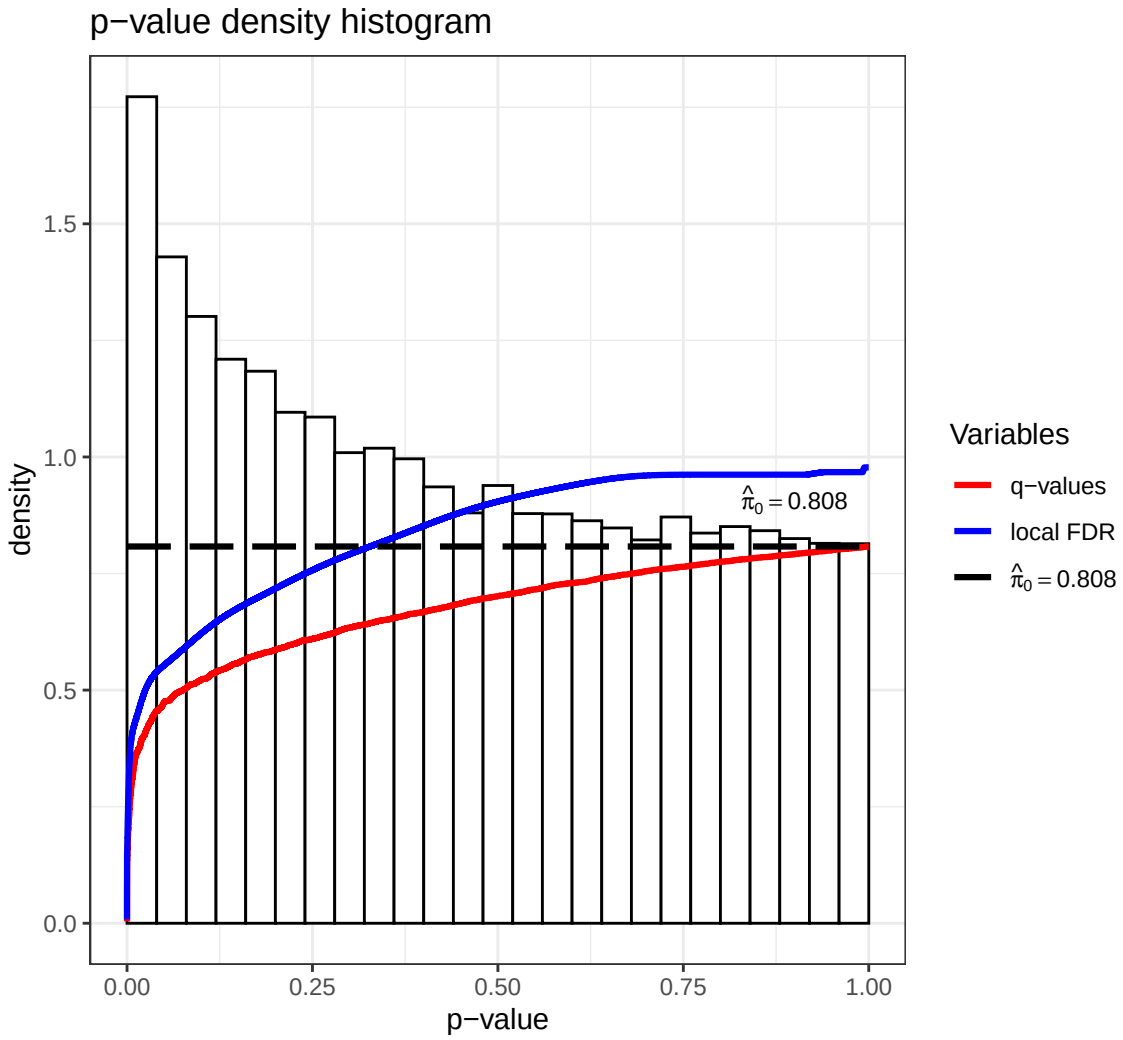
It can be shown that when the regularization can be represented by a prior distribution on $\boldsymbol{B}$ then this algorithm achieves the MAP.

Example: Kidney Expr by Age

In Storey et al. (2005), we considered a study where kidney samples were obtained on individuals across a range of ages. The goal was to identify genes with expression associated with age.

```
> library(edge)
> library(splines)
> load("./data/kidney.RData")
> age <- kidcov$age
> sex <- kidcov$sex
> dim(kidexpr)
[1] 34061    72
> cov <- data.frame(sex = sex, age = age)
> null_model <- ~sex
> full_model <- ~sex + ns(age, df = 3)
```

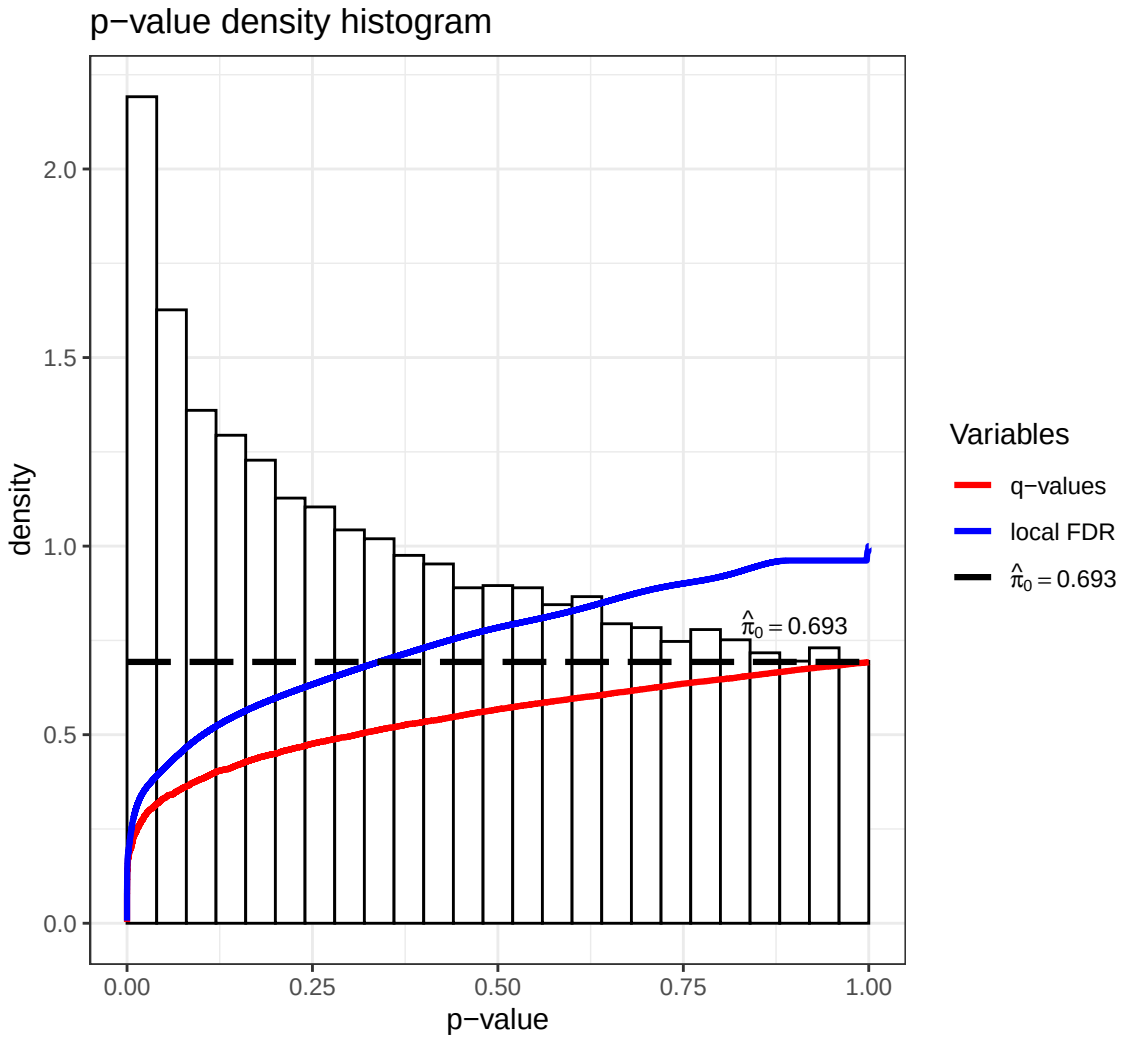
```
> de_obj <- build_models(data = kidexpr, cov = cov,
+                        null.model = null_model,
+                        full.model = full_model)
> de_lrt <- lrt(de_obj, nullDistn = "bootstrap", bs.its = 100, verbose=FALSE)
> qobj1 <- qvalueObj(de_lrt)
> hist(qobj1)
```



Now that we have completed a standard generalized LRT, let's estimate $\boldsymbol{Z}$ (the surrogate variables) using the `sva` package as accessed via the `edge` package.

```
> dim(nullMatrix(de_obj))
[1] 72  2
> de_sva <- apply_sva(de_obj, n.sv=4, method="irw", B=10)
Number of significant surrogate variables is:  4
Iteration (out of 10 ):1  2  3  4  5  6  7  8  9  10
> dim(nullMatrix(de_sva))
[1] 72  6
> de_svalrt <- lrt(de_sva, nullDistn = "bootstrap", bs.its = 100, verbose=FALSE)
```

```
> qobj2 <- qvalueObj(de_svalrt)
> hist(qobj2)
```



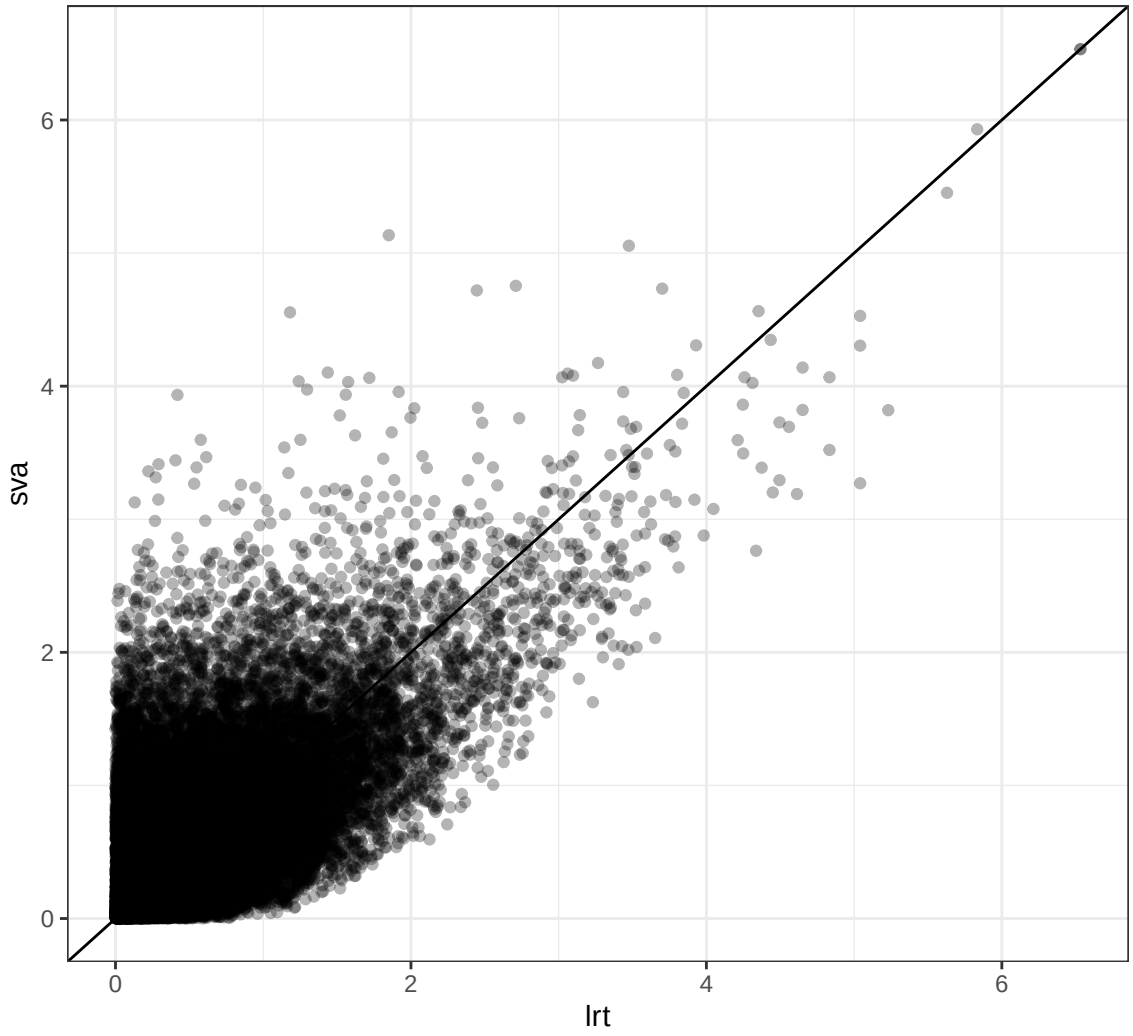

```
> summary(qobj1)

Call:
qvalue(p = pval)

pi0:    0.8081212

Cumulative number of significant calls:

          <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1    <1
p-value       27    161   798   1676  2906 5271 34061
q-value        0      0     2      4    10   27 34061
local FDR      0      0     2      2     5   18 34061
```

```
> summary(qobj2)

Call:
qvalue(p = pval)

pi0:    0.6925105
```

```
Cumulative number of significant calls:

          <1e-04 <0.001 <0.01 <0.025 <0.05 <0.1    <1
p-value       28    151  1001   2051  3549 6168 34061
q-value        0      0     3      4     6   51 34061
local FDR      0      0     2      2     3   28 34053
```

P-values from two analyses are fairly different.

```
> data.frame(lrt=-log10(qobj1$pval), sva=-log10(qobj2$pval)) %>%
+   ggplot() + geom_point(aes(x=lrt, y=sva), alpha=0.3) + geom_abline()
```



Extras

Source

License

Source Code

Session Information

```
> sessionInfo()
R version 3.6.0 (2019-04-26)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS  10.15.3

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] splines   parallel  stats     graphics  grDevices utils
[7] datasets  methods   base

other attached packages:
 [1] edge_2.14.0         Biobase_2.42.0
 [3] BiocGenerics_0.28.0 jackstraw_1.3
 [5] qvalue_2.15.0       MASS_7.3-51.5
 [7] broom_0.5.2         forcats_0.5.0
 [9] stringr_1.4.0       dplyr_0.8.4
[11] purrr_0.3.3         readr_1.3.1
[13] tidyr_1.0.2         tibble_2.1.3
[15] ggplot2_3.2.1       tidyverse_1.3.0
[17] knitr_1.28

loaded via a namespace (and not attached):
 [1] nlme_3.1-144         matrixStats_0.55.0
 [3] bitops_1.0-6         fs_1.3.1
 [5] bit64_0.9-7          lubridate_1.7.4
 [7] httr_1.4.1           tools_3.6.0
 [9] backports_1.1.5      R6_2.4.1
[11] irlba_2.3.3          mgcv_1.8-31
[13] DBI_1.1.0            lazyeval_0.2.2
[15] colorspace_1.4-1     withr_2.1.2
[17] tidyselect_1.0.0     bit_1.1-15.2
[19] compiler_3.6.0       cli_2.0.2
[21] rvest_0.3.5          xml2_1.2.2
[23] labeling_0.3         scales_1.1.0
[25] genefilter_1.64.0    digest_0.6.25
[27] minqa_1.2.4          rmarkdown_2.1
[29] pkgconfig_2.0.3      htmltools_0.4.0
[31] lme4_1.1-21          limma_3.38.3
[33] dbplyr_1.4.2         rlang_0.4.5
[35] readxl_1.3.1         RSQLite_2.2.0
[37] rstudioapi_0.11      farver_2.0.3
[39] generics_0.0.2       jsonlite_1.6.1
[41] BiocParallel_1.16.5  gtools_3.8.1
[43] RCurl_1.98-1.1       magrittr_1.5
[45] Matrix_1.2-18        Rcpp_1.0.3
[47] munsell_0.5.0        S4Vectors_0.20.1
```

```
[49] fansi_0.4.1          lifecycle_0.1.0
[51] stringi_1.4.6        yaml_2.2.1
[53] ClusterR_1.2.1       plyr_1.8.5
[55] blob_1.2.1           grid_3.6.0
[57] crayon_1.3.4         lattice_0.20-40
[59] haven_2.2.0          annotate_1.60.0
[61] hms_0.5.3            pillar_1.4.3
[63] boot_1.3-24          corpcor_1.6.9
[65] codetools_0.2-16     stats4_3.6.0
[67] reshape2_1.4.3       XML_3.99-0.3
[69] reprex_0.3.0         glue_1.3.1
[71] evaluate_0.14        snm_1.30.0
[73] modelr_0.1.6         vctrs_0.2.3
[75] nloptr_1.2.1         cellranger_1.1.0
[77] gtable_0.3.0         assertthat_0.2.1
[79] xfun_0.12            rsvd_1.0.3
[81] lfa_1.12.0           xtable_1.8-4
[83] survival_3.1-8       IRanges_2.16.0
[85] memoise_1.1.0        AnnotationDbi_1.44.0
[87] cluster_2.1.0        sva_3.30.1
[89] gmp_0.5-13.6
```