

# Set package

Simina M. Boca  
Johns Hopkins Bloomberg School of Public Health  
email: sboca@jhsph.edu,

Hector Corrada Bravo  
University of Maryland, College Park  
email: hcorrada@umiacs.umd.edu,

Jeffrey T. Leek  
Johns Hopkins Bloomberg School of Public Health  
email: jleek@jhsph.edu,

Giovanni Parmigiani  
Dana-Farber Cancer Institute and  
Harvard School of Public Health  
email: gp@jimmy.harvard.edu

March 29, 2011

## Contents

<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>Real data example</b>	<b>2</b>
<b>3</b>	<b>The postProb function</b>	<b>3</b>
<b>4</b>	<b>The errDisc function</b>	<b>4</b>
<b>5</b>	<b>The bayesEst function</b>	<b>4</b>
<b>6</b>	<b>The enumAtoms function</b>	<b>5</b>

## 1 Overview

Gene-set analysis generally relies on hypothesis testing and p-values. An alternative approach is to consider a decision-theoretic method which focuses on estimating the fraction of non-null genes in a gene-set, which is more interpretable than traditional approaches [1]. This method uses gene-level statistics and non-overlapping sets (atoms) as input, and returns the sets which are in the Bayes estimator for a given loss function.

The **Set** package implements the decision-theory approach, allowing users to apply the main loss function considered in [1]:

$$L(\tau, U) = (1 - w) * \text{Number of false discoveries} + w * \text{Number of missed discoveries},$$

where  $w$  is a fixed constant between 0 and 1,  $\tau$  is the set consisting of all the genes which have densities from the alternative distribution (for example, the genes which are differentially expressed), and  $U$  represents a union of sets which is a candidate estimator. Alternatively, loss functions which penalize for the number of features or atoms in the Bayes estimator,  $L_f^\lambda$  and  $L_a^\xi$ , can be used, as can loss function which penalize both. Finally, the loss function can use the ratio of false discoveries and missed discoveries:

$$L_r(\tau, U) = (1 - w) * \text{Ratio of false discoveries} + w * \text{Ratio of missed discoveries},$$

Minimizing the Bayes risk is equivalent to minimizing the posterior expected loss, and thus the important quantities which need to be estimated are the posterior expected number of false discoveries (EFD), the posterior expected number of missed discoveries (EMD), the posterior expected ratio of false discoveries (EFDR), and the posterior expected ratio of missed discoveries (EMDR). This vignette represents an introduction to the **Set** package. Four functions are available, **postProb**, **errDisc**, **bayesEst**, and **enumAtoms**. A real data example is also available, taken from [4].

## 2 Real data example

We use one of the datasets from [4]. It compares mRNA expression profiles from 15 males and 17 females from lymphoblastoid cell lines. The gene-sets used represented chromosomal regions. We excluded 40 of the original 212 sets, in order to obtain non-overlapping atoms.

To load the data set type **data(GenderStats)**, and to view a description of this data type ? **GenderStats**. The gene-level statistics are in the vector **gene.stat**. To load the gene-sets type **data(ChromSets)**, and to view a description of this data type ? **ChromSets**. The gene-level statistics are in the vector **sets**.

```
> library(Set)
> data(GenderStats)
> data(ChromSets)
> head(gene.stat)
```

XIST	HDHD1A	EIF1AX	DDX3X	216342_x_at	SMC1L1
2.2625184	1.4953403	0.8762549	0.8493446	0.8317806	0.6815810

```
> head(names(sets))

[1] "chr10q24" "chr5q23" "chr8q24" "chr16q24" "chr7p21" "chr10q23"

> sets[["chr10q24"]]

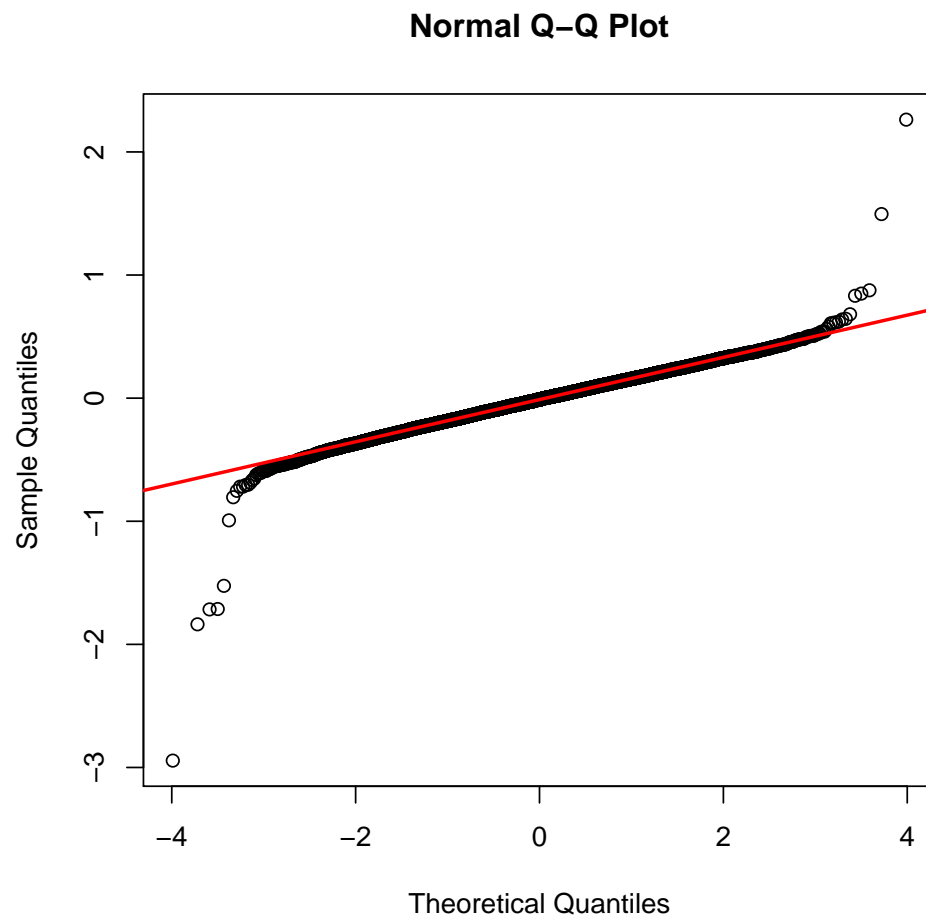
[1] "PITX3" "NEURL" "C10orf12" "NDUFB8" "C10orf95" "DNMT"
[7] "CWF19L1" "OBFC1" "PEO1" "UBTD1" "CUTC" "SEC31L2"
[13] "MGEA5" "NPM3" "PSD" "KAZALD1" "C10orf61" "NOLC1"
[19] "SEMA4G" "ARL3" "MMS19L" "CYP17A1" "GBF1" "ANKRD2"
[25] "ALDH18A1" "WNT8B" "PPRC1" "TLL2" "FRAT1" "FBXL15"
```

[31]	"GOT1"	"CNNM2"	"ENTPD1"	"KCNIP2"	"NT5C2"	"CNNM1"
[37]	"SHFM3"	"INA"	"C10orf76"	"LDB1"	"HPS1"	"C10orf26"
[43]	"HPSE2"	"ABCC2"	"CUEDC2"	"SCD"	"BLNK"	"SLC25A28"
[49]	"FRAT2"	"TRIM8"	"EXOSC1"	"CHUK"	"SORBS1"	"POLL"
[55]	"HIF1AN"	"SFRP5"	"C10orf6"	"AVPI1"	"CPN1"	"CCNJ"
[61]	"ENTPD7"	"PGAM1"	"C10orf77"	"PDZK7"	"NFKB2"	"PDCD11"
[67]	"PKD2L1"	"FGF8"	"SH3MD1"	"KIAA0690"	"PAX2"	"BTRC"
[73]	"FAM26B"	"TAF5"				

### 3 The postProb function

To implement our method, we must obtain an estimate of the posterior probability for each gene that it is differentially expressed. In order to do this, we must make some assumptions about the null distribution of the expression profiles. We note that the data is approximately normal, with the deviations occurring primarily for very high or very low expression values, as seen from the QQ plot (as in [2]):

```
> qqnorm(gene.stat)
> qqline(gene.stat, col = "red", lwd = 2)
```



We standardize the gene-level statistics and assume that the null distribution is normal with mean 0 and standard deviation 1. We now estimate the posterior probabilities, using the method in [3]. Since the function requires null statistics, we simulate 20 datasets using the  $N(0, 1)$  distribution. We choose the number of intervals in which the observed and null statistics are split to be equal to the number of genes.

```
> gene.stat <- (gene.stat - mean(gene.stat))/sd(gene.stat)
> set.seed(70790707)
> post.prob.genes <- postProb(gene.stat, null = rnorm(20 * length(gene.stat),
+      0, 1), K = length(gene.stat))
> head(post.prob.genes)
```

XIST	HDHD1A	EIF1AX	DDX3X	216342_x_at	SMC1L1
0.66370773	0.39100775	0.01661465	0.00000000	0.00000000	0.00000000

## 4 The errDisc function

We now obtain estimates of the following quantities: the posterior expected number of false discoveries (EFD), the posterior expected number of missed discoveries (EMD), the posterior expected fraction of false discoveries (EFDR), and the posterior expected fraction of missed discoveries (EMDR) for each atom. The result is a matrix.

```
> EFD.EMD.res <- errDisc(sets, post.prob.genes)
> head(EFD.EMD.res)
```

	EFD	EMD	EFDR	EMDR
chr10q24	71.87490	166.1797	0.9712824	0.02481776
chr5q23	25.35882	167.6636	0.9753390	0.02486116
chr8q24	69.49881	166.8036	0.9788564	0.02489978
chr16q24	37.18980	167.4946	0.9786790	0.02488037
chr7p21	19.46946	167.7743	0.9734729	0.02485545
chr10q23	46.88215	167.1870	0.9767114	0.02487161

## 5 The bayesEst function

In the case of the loss function used throughout the majority of [1],  $L$ , it is easy to find the Bayes estimator, by just seeing which atoms have the EFDR below the desired threshold  $w$ . Alternatively, the `BayesEst` function can be used. We try for the values  $w = 0.75$  and  $w = 0.97$ .

```
> bayesEst(EFD.EMD.res, w = 0.75, type = "count", atom.sizes = sapply(sets,
+      length))
```

```
[1] "chrYq11"
```

```
> bayesEst(EFD.EMD.res, w = 0.97, type = "count", atom.sizes = sapply(sets,
+      length))
```

```
[1] "chr10q21" "chr6q16" "chr1p35" "chr9q21" "chr4q21" "chr3q13"
[7] "chr12p12" "chr7p22" "chrYq11" "chrXq26"
```

The user may desire to penalize for the number of features or the number of atoms, corresponding to the loss functions  $L_f^\lambda$  and  $L_a^\xi$ , or for both atoms and features. For  $L_f^\lambda$ , we take  $\lambda = 0.20$ :

```
> bayesEst(EFD.EMD.res, w = 0.75, type = "count", atom.sizes = sapply(sets,
+   length), pen.for.feats = 0.2)

character(0)

> bayesEst(EFD.EMD.res, w = 0.97, type = "count", atom.sizes = sapply(sets,
+   length), pen.for.feats = 0.2)

[1] "chrYq11"
```

For  $L_a^\xi$ , we take  $\xi = 5$ :

```
> bayesEst(EFD.EMD.res, w = 0.75, type = "count", atom.sizes = sapply(sets,
+   length), pen.for.atoms = 5)

character(0)

> bayesEst(EFD.EMD.res, w = 0.97, type = "count", atom.sizes = sapply(sets,
+   length), pen.for.atoms = 5)

[1] "chrYq11"
```

We can also get the Bayes estimator for the loss function  $L_r$ , which considers the fraction of false discoveries and missed discoveries, instead of the number of false discoveries and missed discoveries:

```
> bayesEst(EFD.EMD.res, w = 0.75, type = "ratio", atom.sizes = sapply(sets,
+   length))

[1] "chrYq11"
```

## 6 The enumAtoms function

In the example above, the sets considered are not overlapping. In the case of overlapping sets, there are multiple ways of obtaining atoms. One of them is to consider all intersections and set differences of existing sets. The `enumAtoms` function can be used for this purpose, as seen in the following toy example:

```
> set1 <- 1:50
> set2 <- 31:130
> set3 <- 131:180
> set4 <- 161:260
> sets <- list(set1, set2, set3, set4)
> atoms.list <- enumAtoms(sets)
> length(atoms.list)

[1] 6

> sapply(atoms.list, length)

 1 1,2   2   3 3,4   4
30 20  80 30 20  80
```

## References

- [1] S.M. Boca, H. Corrada Bravo, B. Caffo, J.T. Leek, and G. Parmigiani. A decision-theory approach to interpretable set analysis for high-dimensional data. *JHU Biostat Working Paper 211*, 2010.
- [2] R.A. Irizarry, C. Wang, Y. Zhou, and T.P. Speed. Gene set enrichment analysis made simple. *JHU Biostat Working Paper 185*, 2009.
- [3] J.D. Storey, J.M. Akey, and L. Kruglyak. Multiple locus linkage analysis of genomewide expression in yeast. *PLoS Biology*, 3, 2005.
- [4] A. Subramanian, P. Tamayo, V.K. Mootha, S. Mukherjee, B.L. Ebert, M.A. Gillette, A. Paulovich, S.L. Pomeroy, T.R. Golub, E.S. Lander, et al. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.