Gene Set Enrichment Analysis with LIGER

Jean Fan

September 27, 2015

Gene Set Enrichment Analysis (GSEA) is a computational method that determines whether an a priori defined set of genes shows statistically significant, concordant differences between two biological states. The original algorithm is detailed in Subramanian, Tamayo, et al. with Java implementations available through the Broad Institute.

The `liger` package provides a lightweight R implementation of this enrichment test on a list of values. Given a list of values, such as p-values or log-fold changes derived from differential expression analysis or other analyses comparing biological states, this package enables you to test a priori defined set of genes for enrichment to enable interpretability of highly significant or high fold-change genes.

Examples

Consider an example, simulated dataset.

```r
library(liger)
# load gene set
data("org.Hs.GO2Symbol.list")
# get universe
universe <- unique(unlist(org.Hs.GO2Symbol.list))
# get a gene set
gs <- org.Hs.GO2Symbol.list[[1]]
# fake dummy example where everything in gene set is perfectly enriched
vals <- rnorm(length(universe), 0, 10)
names(vals) <- universe
vals[gs] <- rnorm(length(gs), 100, 10)

head(vals)  # look at vals
```

```
##      AKT3   C10orf2      DNA2      LIG3     MEF2A     MGME1
##  93.34684 108.12757  92.53642  98.72407  96.09053  88.69796
```

Here, `vals` can be seen as representing a list of log-fold changes derived from differential expression analysis on samples in two biological states. We want to interpret the set of differentially expressed genes with high positive fold changes using gene set enrichment analysis.

Testing individual gene sets

To test for enrichment of a particular gene set:

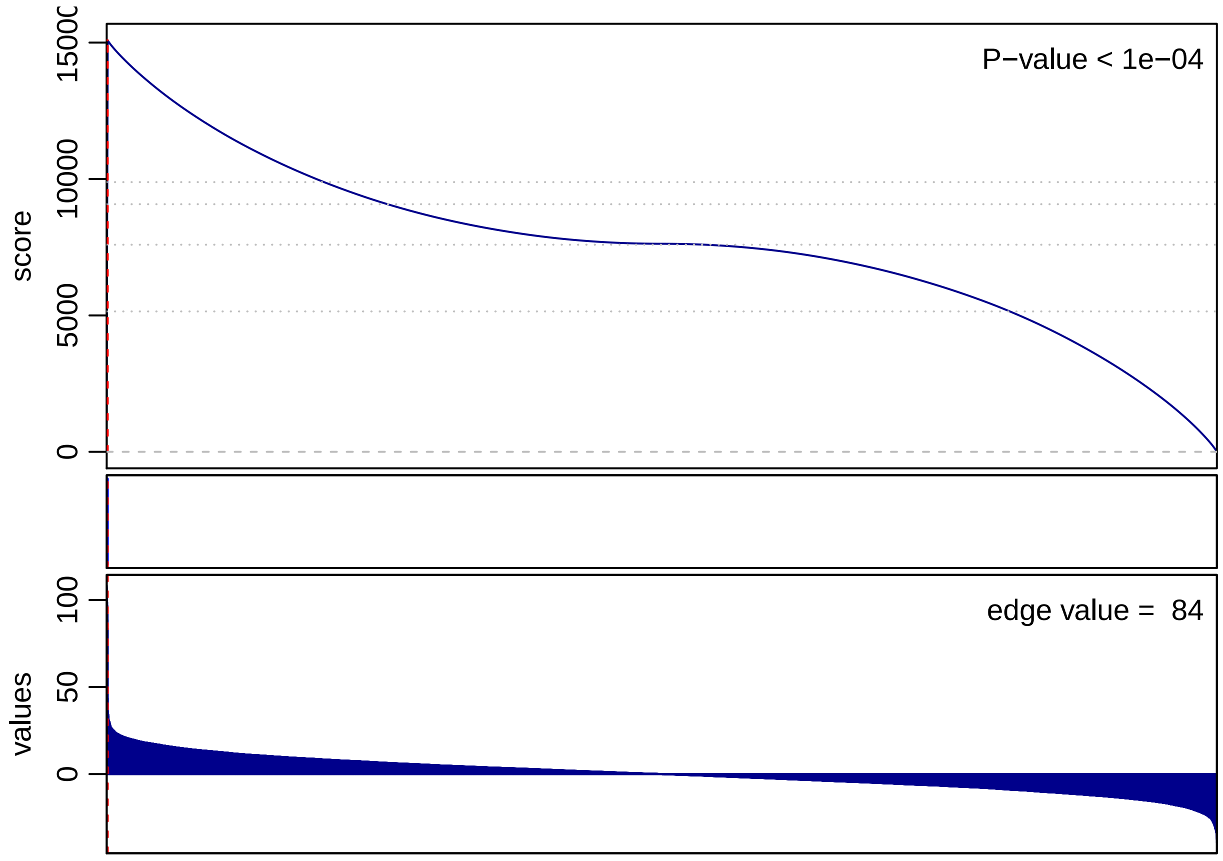
```r
names(org.Hs.GO2Symbol.list)[[1]]
```

```
## [1] "GO:0000002"
```

```r
gs  # look at gs
```

```
##  [1] "AKT3"     "C10orf2"  "DNA2"     "LIG3"     "MEF2A"    "MGME1"
##  [7] "MPV17"    "OPA1"     "PID1"     "PRIMPOL"  "SLC25A33" "SLC25A36"
## [13] "SLC25A4"  "STOML2"   "TYMP"
```

```r
gsea(values=vals, geneset=gs, mc.cores=1, plot=TRUE)
```



```
## [1] 1e-04
```

In this simulation, we created `vals` such that `gs` was obviously enriched. And indeed, we see that this gene set exhibits significant enrichment.

Now to test for enrichment of another gene set:

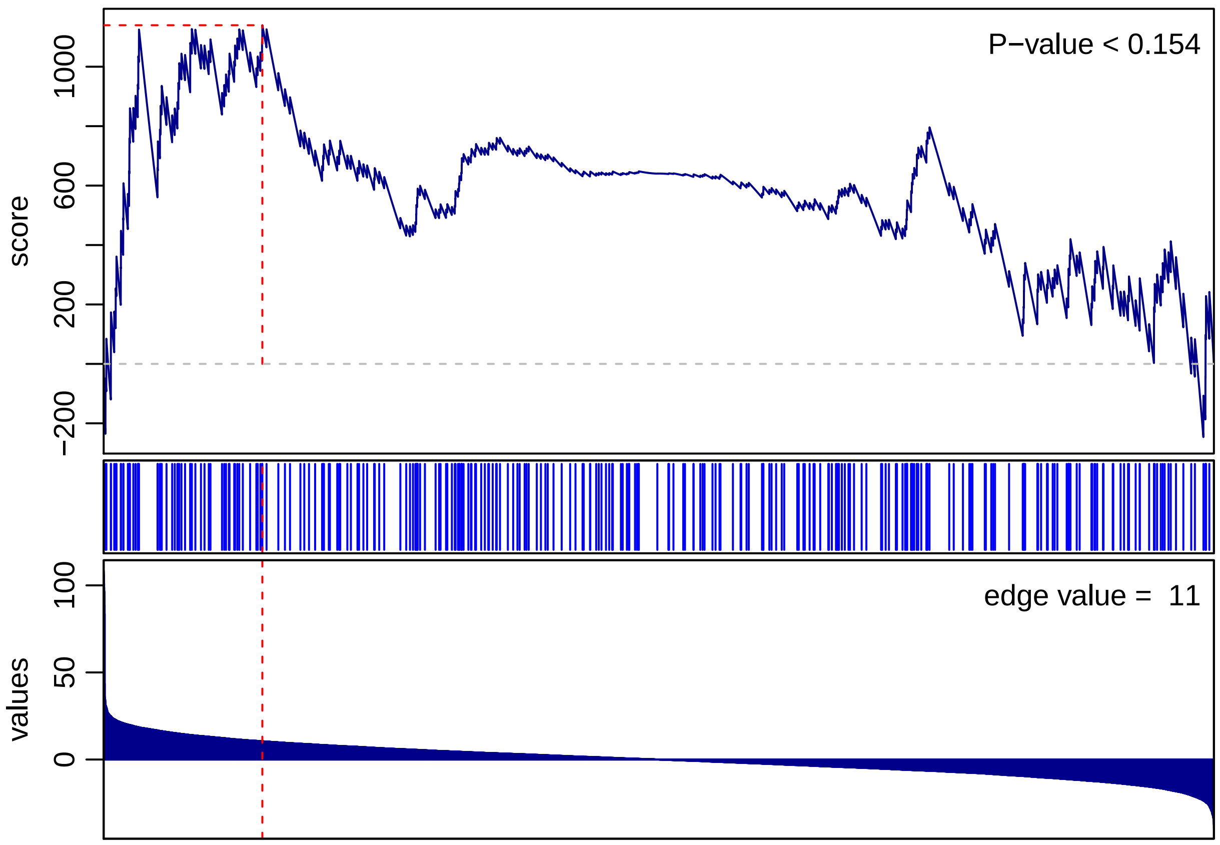
```r
gs.new <- org.Hs.GO2Symbol.list[[2]]
names(org.Hs.GO2Symbol.list)[[2]]
```

```
## [1] "GO:0000003"
```

```r
head(gs.new)  # look at gs.new
```

```
## [1] "ACE"    "ACR"    "ADAM2"  "ADAM20" "ADAM21" "ADAM28"
```

```r
gsea(values=vals, geneset=gs.new)
```

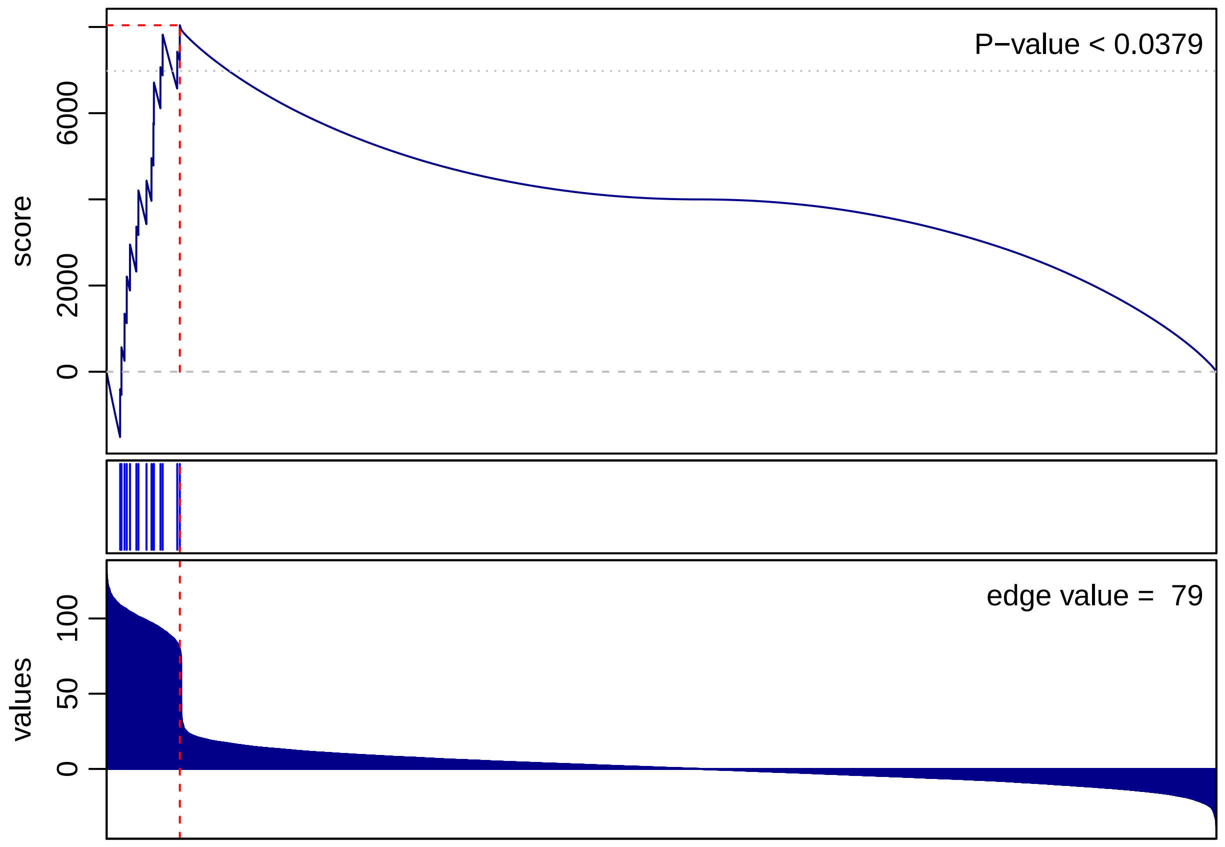


```
## [1] 0.1541
```

In this simulation, we created `vals` such that `gs.new` was obviously not enriched. And indeed, we see that this gene set does not exhibit significant enrichment.

If we simulate a more ambiguous case:

```
# add some noise
vals[sample(1:length(universe), 1000)] <-  rnorm(1000, 100, 10)
# test previously perfectly enriched gene set again
gs <- org.Hs.GO2Symbol.list[[1]]
gsea(values=vals, geneset=gs)
```



```
## [1] 0.0379
```

The enrichment plots and p-values are affected as expected.

Testing multiple gene sets

We can also test a number of gene sets:

```
bulk.gsea(values=vals, set.list=org.Hs.GO2Symbol.list[1:10])
```

```
##                 p.val      q.val     sscore        edge
## GO:0000002 0.00029997 0.00040000  1.4068440  79.4923973
## GO:0000003 0.43205679 0.68043333  0.4203315 111.8244167
## GO:0000012 0.00349965 0.01120000 -1.0710749   7.7765343
## GO:0000014 0.00629937 0.02613333 -1.0256543   0.2520751
## GO:0000018 0.00009999 0.00000000 -1.6749295  23.4978537
## GO:0000022 0.43425657 0.68043333  0.5555959  98.2081752
```

To save on computation time, we can also iterative assess significance:

```
iterative.bulk.gsea(values=vals, set.list=org.Hs.GO2Symbol.list[1:10])
```

```
## initial: [1e+02 - 4] [1e+03 - 4] [1e+04 - 2] done
```

```
##                 p.val       q.val     sscore        edge
## GO:0000002 0.00029997 0.000899910  1.4068440  79.4923973
## GO:0000003 0.44554455 0.504950495  0.6078311 111.8244167
## GO:0000012 0.00349965 0.006999300 -1.0710749   7.7765343
## GO:0000014 0.00629937 0.009449055 -1.0256543   0.2520751
## GO:0000018 0.00009999 0.000599940 -1.6749295  23.4978537
## GO:0000022 0.50495050 0.504950495  0.6296185  98.2081752
```

R Session Info

```
sessionInfo()
```

```
## R version 3.4.4 (2018-03-15)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS High Sierra 10.13.2
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/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] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] liger_0.1
##
## loaded via a namespace (and not attached):
##  [1] compiler_3.4.4     backports_1.1.2    magrittr_1.5
##  [4] rprojroot_1.3-2    parallel_3.4.4     tools_3.4.4
##  [7] htmltools_0.3.6    yaml_2.1.18        Rcpp_0.12.16
## [10] stringi_1.1.7      rmarkdown_1.9      knitr_1.20
## [13] stringr_1.3.0      digest_0.6.15      matrixStats_0.53.1
## [16] evaluate_0.10.1
```