FUNNEL-GSEA R Package

Yun Zhang, Juilee Thakar, Xing Qiu

2017-01-28

Introduction

```
## The package can be installed using the following code
# library(devtools)
# install_github("yunzhang813/FUNNEL-GSEA-R-Package", build_vignettes=TRUE)

## Load the package
library(FUNNEL)
```

This R package is built for FUNNEL-GSEA. It is a statistical inference framework for Gene Set Enrichment Analysis (GSEA) based on FUNctioNal ELastic-net regression (FUNNEL), which utilizes the temporal information based on functional principal component analysis (FPCA), and disentangles the effects of **overlapping** genes by a functional extension of the elastic-net regression. It then performs hypothesis testing for gene sets by an extension of Mann-Whitney U test which is based on weighted rank sums computed from correlated observations.

In this vignette, we will introduce the one-step function for carrying out the FUNNEL analysis, and also some useful functions contained in the FUNNEL framework.

- The one-step function, `FUNNEL.GSEA()`, takes pre-processed data and a list of gene sets as inputs and runs the whole testing procedure automatically. There are also some post-analysis tools such as `weightPerGene()` and `plotWeight()` that faciliates the interpretation and visulization of the FUNNEL result.
- For advanced users, we will introduce functions for individual steps, such as `FPCA.Fstats()`, `equiv.regression()` and `wMWUTest()`. They are the key building blocks of the FUNNEL framework, and also useful by themselves. The users may find more flexibity by using these functions for their own analyses.

FUNNEL one-step function

Sample data

For convenience, a set of real temporal gene expression data is included in the `FUNNEL` package. Gene expression data of human influenza infection (Huang et al. (2011)) were downloaded from Gene Expression Omnibus series GSE52428. 17 healthy adults with live influenza (H3N2/Wisconsin) were examined and changes in host peripheral blood gene expression were collected at 16 time points over 132 hours. Among the 17 subjects, nine developed symptomatic infection and eight had asymptomatic infection. We include subject 1 (symptomatic) as sample data for illustration purpose. These data were log2-transformed and pre-filtered by the IQR criterion (>=0.3).

We also include 186 CP:KEGG pathways with a focus on human immuno-response to infectious diseases in the `FUNNEL` package. These pathways are available from the Broad Institute. For studies that do not focus on immunology, an alternative list of pathways should be used instead.

The sample data and pathways can be loaded by the following command. The expression data is called `X`, which is a 11189x16 dimensional matrix. A vector of ordered length 16, `tt`, is also provided with this package.

It indicates the sampling time points of these data. The 186 CP:KEGG pahtways are formatted as a data list called `genesets`.

```
data("H3N2-Subj1")
```

More details about the sample data can be found in `help("H3N2-Subj1")`.

One-step analysis

The most user-friendly function provided by `FUNNEL` is `FUNNEL.GSEA()`. It inputs pre-processed data and a list of gene sets and runs several steps of the FUNNEL procedure automatically.

It takes about 5~10 minutes to run the following code on a modern laptop computer.

```
system.time(result <- FUNNEL.GSEA(X, tt, genesets))
```

```
## 26 real eigenvalues are negative or zero and are removed!
## Weight calculation...
## Gene set test...
```

```
##    user  system elapsed
## 313.594   2.396 316.146
```

Once the computation is done, we will have a result list that contains the following useful objects.

1. `pvals` is a vector of unadjusted p-values for pre-defined gene sets.
2. `weight.list` is a list of weights (a.k.a. empirical gene set membership) computed from the elastic-net regression.
3. `correlation` is the mean intergene correlation coefficient estimated from genes within each gene set.
4. `sig.genesets` is a vector of names (or indices) of significant gene sets at 5% significance level (default).
5. `Fstats` is a vector of F-statistics, which are gene-level summary statistics used in extended Mann-Whitney U test.

For Subject 1 (symptomatic), `FUNNEL.GSEA` identified the following significant gene sets that responded to the influenza infection. Many of these pathways listed here, such as the B cell receptor signaling, T cell receptor signaling, NOD-like signaling, RIG-I-like receptor signaling, Toll-like receptor signaling, and FC-Epsilon-RI signaling are well documented immune signaling pathways that send signals that lead to the activation of various cell-specific immune activities.

```
result$sig.genesets
```

```
##  [1] "B_CELL_RECEPTOR_SIGNALING_PATHWAY"
##  [2] "APOPTOSIS"
##  [3] "PHOSPHATIDYLINOSITOL_SIGNALING_SYSTEM"
##  [4] "NATURAL_KILLER_CELL_MEDIATED_CYTOTOXICITY"
##  [5] "LEISHMANIA_INFECTION"
##  [6] "PENTOSE_PHOSPHATE_PATHWAY"
##  [7] "NON_SMALL_CELL_LUNG_CANCER"
##  [8] "TOLL_LIKE_RECEPTOR_SIGNALING_PATHWAY"
##  [9] "CYTOSOLIC_DNA_SENSING_PATHWAY"
## [10] "RIG_I_LIKE_RECEPTOR_SIGNALING_PATHWAY"
## [11] "PYRUVATE_METABOLISM"
## [12] "HEMATOPOIETIC_CELL_LINEAGE"
## [13] "NOD_LIKE_RECEPTOR_SIGNALING_PATHWAY"
## [14] "NUCLEOTIDE_EXCISION_REPAIR"
## [15] "FC_EPSILON_RI_SIGNALING_PATHWAY"
## [16] "EPITHELIAL_CELL_SIGNALING_IN_HELICOBACTER_PYLORI_INFECTION"
## [17] "RENAL_CELL_CARCINOMA"
```

```
## [18] "T_CELL_RECEPTOR_SIGNALING_PATHWAY"
## [19] "PROGESTERONE_MEDIATED_OOCYTE_MATURATION"
## [20] "PEROXISOME"
## [21] "CIRCADIAN_RHYTHM_MAMMAL"
## [22] "UBIQUITIN_MEDIATED_PROTEOLYSIS"
## [23] "ANTIGEN_PROCESSING_AND_PRESENTATION"
## [24] "CITRATE_CYCLE_TCA_CYCLE"
## [25] "PATHOGENIC_ESCHERICHIA_COLI_INFECTION"
## [26] "PRION_DISEASES"
## [27] "INOSITOL_PHOSPHATE_METABOLISM"
## [28] "ONE_CARBON_POOL_BY_FOLATE"
## [29] "N_GLYCAN_BIOSYNTHESIS"
```

Altering default parameter settings

Users may customize their own studies by varing the default settings in `FUNNEL.GSEA()`. Details about the default parameter settings can be found in `help("FUNNEL.GSEA")`. The following code is a simple example, which uses 2 eigenfunctions and different panelty values for the elastic-net regression. Many of the immune signaling pathways remain significant. (Default: `nharm=3, lam1=0.4, lam2=0.01`.)

```
result2 <- FUNNEL.GSEA(X, tt, genesets, nharm=2, lam1=0.3, lam2=0.1)
```

```
## 26 real eigenvalues are negative or zero and are removed!
## Weight calculation...
## Gene set test...
```

```
result2$sig.genesets
```

```
##  [1] "B_CELL_RECEPTOR_SIGNALING_PATHWAY"
##  [2] "APOPTOSIS"
##  [3] "CYTOSOLIC_DNA_SENSING_PATHWAY"
##  [4] "LEISHMANIA_INFECTION"
##  [5] "TOLL_LIKE_RECEPTOR_SIGNALING_PATHWAY"
##  [6] "PENTOSE_PHOSPHATE_PATHWAY"
##  [7] "RIG_I_LIKE_RECEPTOR_SIGNALING_PATHWAY"
##  [8] "NATURAL_KILLER_CELL_MEDIATED_CYTOTOXICITY"
##  [9] "PHOSPHATIDYLINOSITOL_SIGNALING_SYSTEM"
## [10] "NOD_LIKE_RECEPTOR_SIGNALING_PATHWAY"
## [11] "NUCLEOTIDE_EXCISION_REPAIR"
## [12] "T_CELL_RECEPTOR_SIGNALING_PATHWAY"
## [13] "ACUTE_MYELOID_LEUKEMIA"
## [14] "INOSITOL_PHOSPHATE_METABOLISM"
## [15] "FC_EPSILON_RI_SIGNALING_PATHWAY"
## [16] "EPITHELIAL_CELL_SIGNALING_IN_HELICOBACTER_PYLORI_INFECTION"
## [17] "CITRATE_CYCLE_TCA_CYCLE"
## [18] "PYRIMIDINE_METABOLISM"
## [19] "PEROXISOME"
## [20] "CIRCADIAN_RHYTHM_MAMMAL"
## [21] "GLUTATHIONE_METABOLISM"
## [22] "HEMATOPOIETIC_CELL_LINEAGE"
## [23] "CYTOKINE_CYTOKINE_RECEPTOR_INTERACTION"
## [24] "UBIQUITIN_MEDIATED_PROTEOLYSIS"
## [25] "PATHOGENIC_ESCHERICHIA_COLI_INFECTION"
## [26] "N_GLYCAN_BIOSYNTHESIS"
```

Some post-analysis tools

Empirical gene set membership

We developed the concept of "empirical gene set membership" (quantified by weights) for individual genes. It reflects the empirical evidence trained from the data and illustrates the practical membership for the (overlapping) genes with regard to their belonging gene sets.

If users are interested in some specific genes (for example, all the genes in the Primary Immunodeficiency pathway), the function `weightPerGene()` exracts the estimated weights (a.k.a. empirical gene set membership) from the `FUNNEL.GSEA` result.

```
est.weights <- weightPerGene(result$weight.list,
                             genesOfInterest=genesets[["PRIMARY_IMMUNODEFICIENCY"]])
```

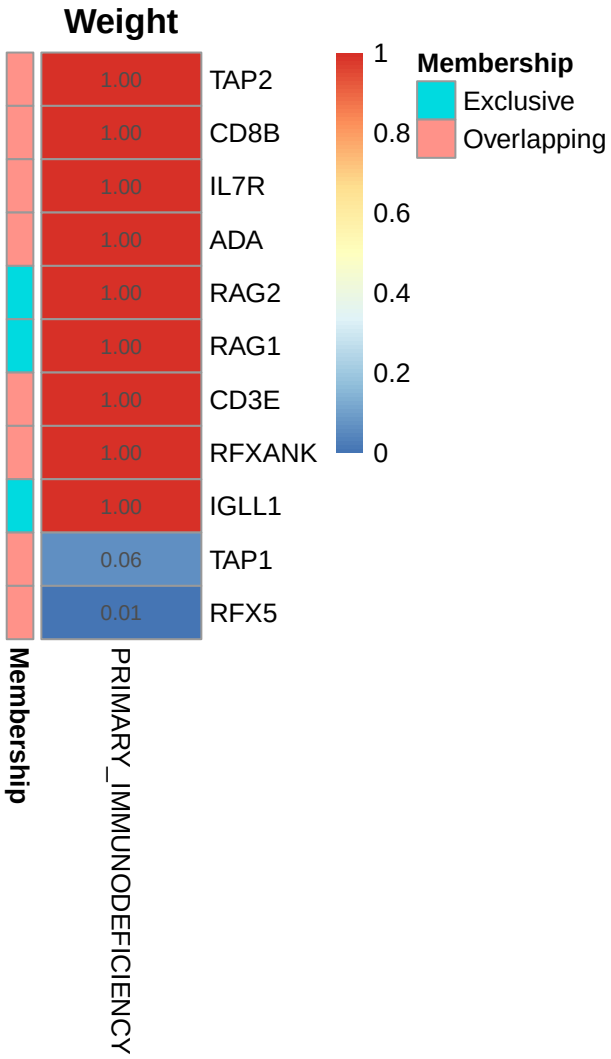
This function returns a list of length of `length(genesOfInterest)`.

1. If a gene belongs to multiple pathways, it returns a vector of estimated weights (usually, sum to 1) for each of the overlapping pathways. In some cases, the estimated weights can be a zero vector due to LASSO penalty, which means that this gene didn't show any similar expression pattern to other genes in the same pathway.
2. If a gene is exclusive to one pathway, it returns integer `1`.
3. `NA` means that this gene is not presented in the expression data.

Weight plot

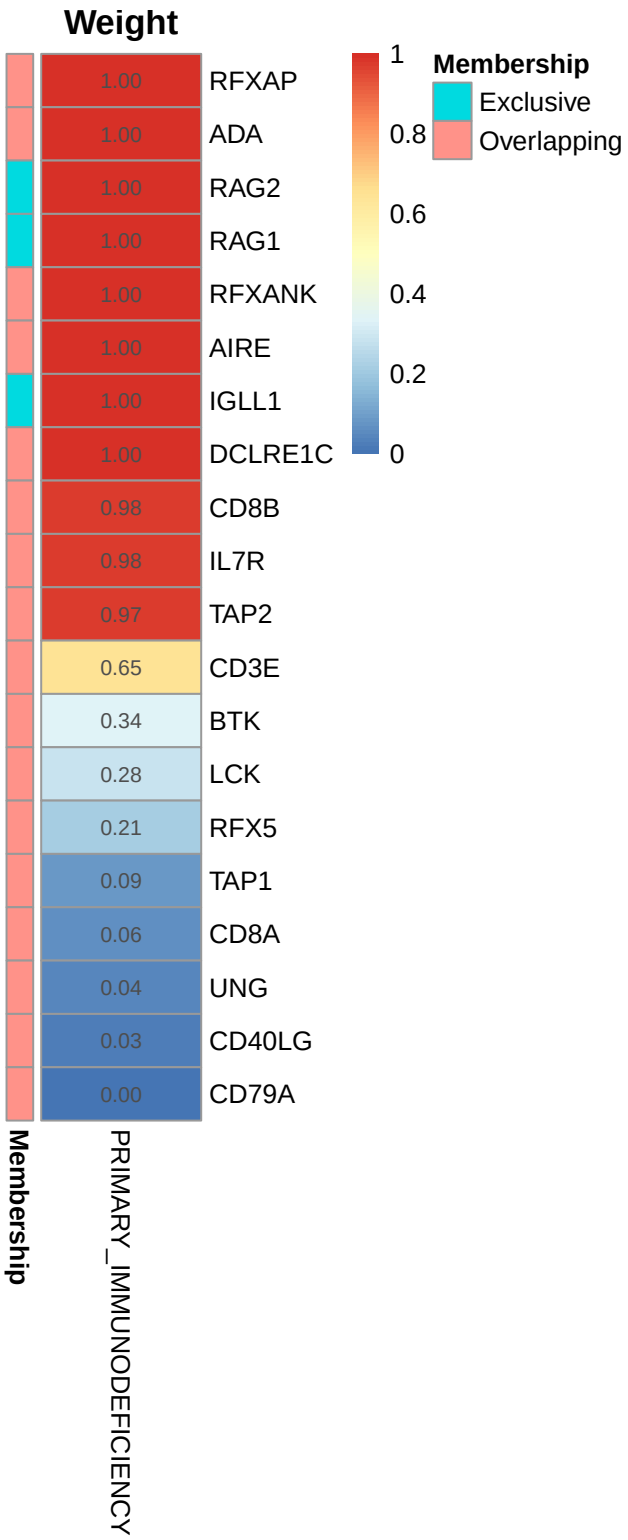
Users can also plot the (non-zero) weights (a.k.a. empirical gene set membership) for a specific gene set. The function `plotWeight()` produces a heatmap of weights based on the `FUNNEL.GSEA` result. For example, the following code plots the (non-trivial) empirical membership for the Primary Immunodeficiency pathway.

```
plotWeight(result$weight.list, geneset.index="PRIMARY_IMMUNODEFICIENCY")
```



We may see how the estimated weights are changed by varying the default setting.

```
plotWeight(result2$weight.list, geneset.index="PRIMARY_IMMUNODEFICIENCY")
```


Weight
1.00
RFXAP
1.00
ADA
1.00
RAG2
1.00
RAG1
1.00
RFXANK
1.00
AIRE
1.00
IGLL1
1.00
DCLRE1C
0.98
CD8B
0.98
IL7R
0.97
TAP2
0.65
CD3E
0.34
BTK
0.28
LCK
0.21
RFX5
0.09
TAP1
0.06
CD8A
0.04
UNG
0.03
CD40LG
0.00
CD79A
1
0.8
0.6
0.4
0.2
0
Membership
Exclusive
Overlapping
Membership
PRIMARY_IMMUNODEFICIENCY

Advanced functions

In this package, we also provide functions for performing the key parts of the FUNNEL-GSEA framework. Advanced users may find these functions helpful for conducting their customized analyses.

Additional data processing

First of all, the following additional data processing is performed internally in `FUNNEL.GSEA()`.

```
library(fda)

## Remove genes in predefined gene sets that are not present in X the filtered input data
genenames <- rownames(X)
newGenesets <- lapply(genesets, function(z) { intersect(z, genenames) } )

## Standardize time and X so that the smoothing penality is applicable to any real data
## with comparable signal-to-noise ratio
tt2 <- (tt - min(tt))/diff(range(tt))
X2 <- t(scale(t(X)))

## Smoothing
mybasis <- create.bspline.basis(range(tt2), length(tt2)+4-2, 4, tt2)
mypar <- fdPar(mybasis, 2, lambda=10^-3.5)
fdexpr <- smooth.basis(tt2, t(X2), mypar)$fd
```

1. We recommend to standardize time (into [0,1] interval), so that the smoothing penalty is comparable across different data sets and the default value (`lambda=10^-3.5`) makes sense in these situations.
2. We recommend the use of standardized (z-transformation) expression data for the gene set analysis because usually the "relationship" between genes is defined through the similarity of the shape, not the magnitude, of their expression patterns.
3. We use functional data analysis framework (R package `fda`) to turn discrete expression data into smooth temporal expression functions (the R object `fdexpr`) that are needed for subsequent analyses.

Functional F-statistics for time-course gene expression data

The function `FPCA.Fstats()` takes time-course expression data and time points as input values. By using Functional Principal Component Analysis (FPCA) techniques, it returns the functional F-statistic for each gene. For more details, please refer to S. Wu and Wu (2013).

```
## Get functional F-statistics
Fstats <- FPCA.Fstats(X2, tt2)
```

```
## 26 real eigenvalues are negative or zero and are removed!
```

```
## Sort the F-statistics from the largest to the smallest.
## The top genes are more "significant".
sorted.genes <- names(sort(Fstats, decreasing = TRUE))
```

For time-course gene expression data, a temporally differentially expressed gene (TDEG) can be defined as a gene with significant non-constant expression pattern across time points. In other words, we want to test the following hypotheses, for gene i

$$H_{i0} : x_i(t) = \mu_i \quad \text{vs.} \quad H_{i1} : x_i(t) \neq \mu_i \quad \text{for} \quad t \in [t_1, t_n].$$

Under H_{i0}, $x_i(t)$ is estimated by a function with constant value $\hat{\mu}_i$ (the sample mean expression for the ith gene); under H_{i1} , $\hat{x}_i(t)$ defined in Equation (1) is used instead. We use the following functional F-statistic

to summarize the information contained by each gene over time:

$$F_i = \frac{RSS_i^0 - RSS_i^1}{RSS_i^1},$$

where RSS_i^0 and RSS_i^1 are the residual sum of squares under the null and the alternative, respectively. This summary statistic can be viewed as a "signal-to-noise" ratio for the functional data. The larger F_i is, the more significant the ith gene is.

1. Note that we use these test statistics as gene-level summary statistics for gene set analysis in `FUNNEL`.
2. Although this F-statistic is defined as a ratio of variance; due to the nature of functional data analysis, F_i may not follow a standard F-distribution under H_0. To select significant TDEGs, we must resort to a resampling method. Please consult S. Wu and Wu (2013) for more details.

An equivalence between functional and multivariate regression

One important computation shortcut we used in `FUNNEL` is the fact that there is a way to turn a (penalized) concurrent functional linear regression analysis into an **equivalent** multivariate linear regression. Technical details can be found in Zhang, Thakar, and Qiu (2016), Supplementary Text, Section 2.

The function `equiv.regression()` takes functional covariates `xfd` and response `yfd` as inputs, and returns a list of equivalent multivariate covariates `Xmat` and response `y`. Running penalized (or ordinary least square) regression on `y` and `Xmat` is equivalent to the corresponding concurrent functional regression.

```
library(quadrupen)

## Take genes C5AR1 and C3AR1 as two examples
gene.i <- c("C5AR1", "C3AR1")

## They belong to the following two pathways
newGeneset.i <- newGenesets[c("NEUROACTIVE_LIGAND_RECEPTOR_INTERACTION",
                              "COMPLEMENT_AND_COAGULATION_CASCADES")]

## The response is just the smoothed curves of these two genes
yfd <- fdexpr[gene.i]

###############
## Method 1 ##
###############

## Let us use the first 3 eigen-functions of both pathways as covariates
xfd <- FUNNEL:::PCA.genesets(fdexpr, newGeneset.i, nharm = 3, centerfns = FALSE)$harmonics

## Calculate the equivalent multivariate regression datasets
equiv <- equiv.regression(yfd, xfd, threshold = 0.01)
equiv.Y <- equiv$y; colnames(equiv.Y) <- gene.i     #3x2 matrix
equiv.X <- equiv$Xmat                               #3x6 matrix
colnames(equiv.X) <- paste(rep(names(newGeneset.i), each=3),
                           rep(paste0("eigfun", 1:3), length(newGeneset.i)), sep=".")

## Now we can run multivariate elastinet regression on the equivalent X and Y, as
## implemented in package quadrupen
en <- elastic.net(equiv.X, equiv.Y[, "C3AR1"],
                  lambda1 = 0.4, lambda2 = 0.01, intercept = FALSE, normalize = FALSE)
```

```
## beta.en are the regression coefficients
beta.en <- as.numeric(attributes(en)$coef)
names(beta.en) <- colnames(equiv.X)

##############
## Method 2 ##
##############

## Alternatively, let us try using the first 2 eigen-functions and increase the variance
## threshold in the equivalence rotated regression to 0.1.
xfd2 <- FUNNEL:::PCA.genesets(fdexpr, newGeneset.i, nharm = 2, centerfns = FALSE)$harmonics

## Calculate the equivalent multivariate regression datasets
equiv2 <- equiv.regression(yfd, xfd2, threshold = 0.1)
equiv2.Y <- equiv2$y; colnames(equiv2.Y) <- gene.i     #3x2 matrix
equiv2.X <- equiv2$Xmat                                #3x6 matrix
colnames(equiv2.X) <- paste(rep(names(newGeneset.i), each=2),
                            rep(paste0("eigfun", 1:2), length(newGeneset.i)), sep=".")

## Now we can run multivariate elastinet regression on the equivalent X and Y, as
## implemented in package quadrupen
en2 <- elastic.net(equiv2.X, equiv2.Y[, "C3AR1"],
                   lambda1 = 0.4, lambda2 = 0.01, intercept = FALSE, normalize = FALSE)

## beta.en are the regression coefficients
beta.en2 <- as.numeric(attributes(en2)$coef)
names(beta.en2) <- colnames(equiv2.X)
```

An extended Mann-Whitney U test that incorporates pre-computed weights and correlation

The function `wMWUTest()` performs an extension of the two-sample Mann-Whitney U test (a.k.a. rank sum test) which incorporates pre-calculated weights for the correlated observations in the test group. Note that the pre-calculated correlation only applies to the test group (the gene set of interest). The correlation of the background genes is assumed to be zero. In the context of gene set analysis, these weights are called "empirical gene set membership" which is calculated by function `FUNNEL.GSEA()`. Users can provide customized weight vectors to `wMWUTest()` to make inference about two-group comparisons.

```
gg1 <- newGenesets[["GLYCOLYSIS_GLUCONEOGENESIS"]]
ww1 <- result$weight.list[["GLYCOLYSIS_GLUCONEOGENESIS"]]
rho <- result$correlation

## The test
test1 <- wMWUTest(gg1, result$Fstats, ww1, rho, df=length(tt2)-1)
## p-value for the gene set test
test1["greater"]
```

```
##   greater
## 0.1935949
```

```
## Should be the same as the p-value below
result$pvals["GLYCOLYSIS_GLUCONEOGENESIS"]
```

```
## GLYCOLYSIS_GLUCONEOGENESIS
##                  0.1935949
```

Session Info

```
sessionInfo()
```

```
## R version 3.3.1 (2016-06-21)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.12.2 (Sierra)
##
## 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] FUNNEL_1.0.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.7         knitr_1.15.1        magrittr_1.5
##  [4] MASS_7.3-45         splines_3.3.1       munsell_0.4.3
##  [7] colorspace_1.2-6    lattice_0.20-34     quadrupen_0.2-4
## [10] fda_2.4.4           stringr_1.1.0       plyr_1.8.4
## [13] tools_3.3.1         parallel_3.3.1      grid_3.3.1
## [16] gtable_0.2.0        htmltools_0.3.5     assertthat_0.1
## [19] yaml_2.1.14         lazyeval_0.2.0      rprojroot_1.2
## [22] digest_0.6.12       tibble_1.2          Matrix_1.2-7.1
## [25] akima_0.6-2         RColorBrewer_1.1-2  reshape2_1.4.2
## [28] ggplot2_2.2.0       evaluate_0.10       rmarkdown_1.3
## [31] sp_1.2-4            pheatmap_1.0.8      stringi_1.1.1
## [34] scales_0.4.1        backports_1.0.5
```

Reference

Huang, Yongsheng, Aimee K Zaas, Arvind Rao, Nicolas Dobigeon, Peter J Woolf, Timothy Veldman, N Christine Øien, et al. 2011. "Temporal Dynamics of Host Molecular Responses Differentiate Symptomatic and Asymptomatic Influenza a Infection." *PLoS Genet* 7 (8). Public Library of Science: e1002234.

Wu, Shuang, and Hulin Wu. 2013. "More Powerful Significant Testing for Time Course Gene Expression Data Using Functional Principal Component Analysis Approaches." *BMC Bioinformatics* 14 (1). BioMed Central: 1.

Zhang, Yun, Juilee Thakar, and Xing Qiu. 2016. "FUNNEL-Gsea: FUNctioNal Elastic-Net Regression in Time-Course Gene Set Enrichment Analysis." *Submitted to Bioinformatics*.