Tutorial for
MATLAB Metric GSEA (MrGSEA)

Author:
Michal Marczyk
michal.marczyk@polsl.pl

If you use the software, please cite:
Zyla, J., Marczyk, M., Weiner, J., & Polanska, J. (2017). Ranking metrics in gene set enrichment analysis: do they matter?. *BMC Bioinformatics*, *18*(1), 256

1. Input data

1.1 Expression data

The presented MATLAB Gene Set Enrichment Analysis (GSEA) implementation needs definition of two input data files. The first file has to consist of gene expressions and all descriptive information about them. Gene expression matrix has to be constructed in the following form: rows represent genes, column represent individuals. Next, grouping variable has to be defined, where 0 will represent controls and 1 cases. Finally, the variable with entrez ID or Gene Symbol of investigated genes has to be defined. Please keep the order between gene IDs and their corresponding expression values. The same rule works for individuals grouping variable. Brief idea of variables structure is presented in Figure 1. An exemplary data set is enclosed with the package (GSE6344.mat).

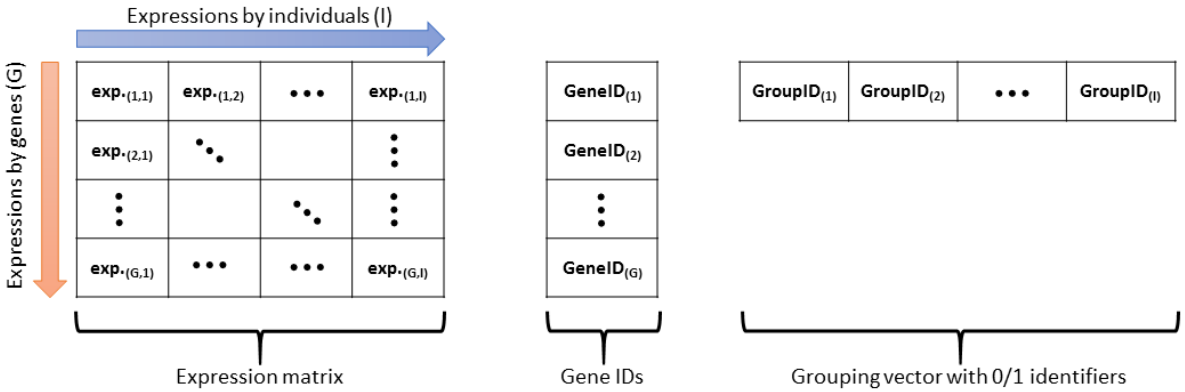

Figure 1 – Input variable structure of expressions, grouping variable and gene IDs.

1.2 Gene set data collection

Second data set needs to include gene sets definitions. The variable must have structure form in MATLAB with following information:

- GS.ID – vector of gene sets IDs,
- GS.descr – cell matrix with details description of gene set (could be fill with NaN),
- GS.nb – variable with information about total number of gene sets,
- GS.entrez – cell matrix with gene entrez IDs for each gene set,
- GS.entrez_nb – vector of gene number in each gene set.

Exemplary .mat file is included in package. What is more, the package includes a function, which allows to transform .xls, .txt etc. files to MATLAB compatible format. The function name is update_GS() and as input argument .xls,.txt etc. file name is needed:

```
update_GS('KEGG_GS.xlsx')
```

The file must have the following structure (exemplary file KEGG_GS.xlsx):

- Column 1 – gene set ID,
- Column 2 – gene set details description (could be fill with NaN),
- Column 3-N – gene entrez IDs for each gene set.

Brief idea of gene set file structure is presented in Figure 2.

GeneSet ID$_{(1)}$	GeneSet Description$_{(1)}$	entrez $_{(1,1)}$	• • •	entrez $_{(1,N)}$
GeneSet ID$_{(2)}$	GeneSet Description$_{(2)}$	entrez $_{(2,1)}$	• • •	entrez $_{(2,N)}$
⋮	⋮	⋮	⋮	⋮
GeneSet ID$_{(GS)}$	GeneSet Description$_{(GS)}$	entrez $_{(GS,1)}$	• • •	entrez $_{(GS,N)}$

Figure 2 – Gene set input variable structure.

2. Run algorithm

After preparing two input data files running the procedure can be started. First of all, please keep all functions in one folder "GSEA_package", where inside also the gene set data collection will be stored. Next, please set the path to folder with functions package and set default options:

```
clear all; close all; clc
addpath('GSEA_package') % add path with package functions folder name
% set options
opts = default_GSEA_opts();
```

The default options can be change and set according to user needs by function editing or overwriting functions output ("opts"). Below the descriptions of settings is presented:

```
%Default parameters for GSEA method.
opts.GS_name = 'KEGG_GS';    %GeneSet database name
opts.GS_filt = [15,500];     %minimum and maximum number of genes in GS
opts.sort_type = 'descend'; %type of ranks sorting
opts.p = 1;                  %0 - Kolmogorov-Smirnov statistic,
                              1-weighting genes by ranking metric
opts.rank_type = 'S2N';%ranking method
{'S2N','ttest','ratio','diff','log2_ratio','SoR','BWS','ReliefF','WAD','FC
ROS','MWT','MSD','external' (dedicated for own rank implementations)}
opts.abs = true;             %if use absolute value of ranking statistic
opts.tied_rank = false;      %if create tied ranks of ranking statistic
opts.perm_type = 'pheno';    %type of permutation {'entrez','pheno'}
opts.perm_nb = 1000;         %number of permutations
opts.perm_ext = false;       %if extend number of permutation till
                              one significant result occur(may be
                              time consuming)
opts.GS_sort_type = 'none'; %options for sorting results
                             {'ES_pval','NES_qval','NES_FWER','none'}
opts.show = true;            %if plot results
opts.save = false;           %if save results
```

If you would like to implement your own ranking metrics please put a script into m_fun.m file and mark `opts.rank_type='external';`.

After setting all options according to user needs GSEA algorithm can be run. The input to the function must have following order: gene expression matrix, grouping variable, gene entrez IDs, output file name, options - output form `default_GSEA_opts()`.

```
[POS,NEG,DESCR,GenePval] =
MrGSEA(expressions,group,gene_IDs,'result_file_name',opts);
```

3. Output information

The GSEA_PreRanked function can return four variables as a result:

```
[POS,NEG,DESCR,GenePval] = MrGSEA(…)
```

POS – results for up-regulated enriched gene sets (Positive value of Enrichment Score),
NEG – results for down-regulated enriched gene sets (Negative value of Enrichment Score),
DESCR – variable with description of each column in POS and NEG matrices.,
GenePval – P-value for each gene (useful for ranking metrics where permutations are needed to assess significance of expression differentiation).

The exemplary results are presented below:

GS NAME	GS size	ES	ES p-value	NES	NES p-value	NES q-value	NES FWER	# genes in LEF
10	60	0,595833	0,006	1,82957	0,000681	0,015217	0,182602	31
20	29	0,659412	0,003155	1,88158	0,000416	0,013921	0,111367	19
30	25	0,580781	0,001009	1,572377	0,005219	0,034965	1	11

Where:

- ES - Enrichment Score,
- NES - Normalised Enrichment Score,
- # genes in LEF – represents number of genes in Leading Edge Fraction

For more details of result metric please see work of Subramanian et al.[1]

The second type of output results are plots for each gene set, which include distribution of enrichment score across genes, cumulative distribution of P_{miss} and P_{hit} and ranking metric value with marked genes in gene set. The plots will be generated only if `opts.show = true`. Exemplary plot for Gene Set ID 20 is presented in Figure 3.

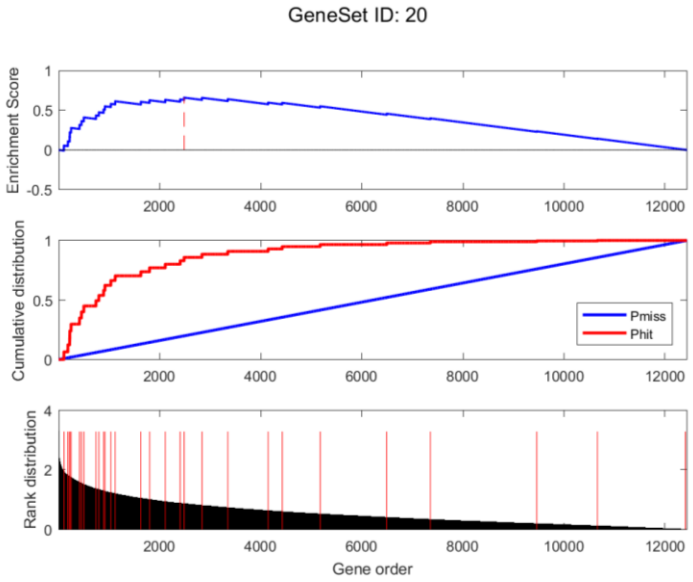

Figure 3 – Output plot generated for each gene set.

[1] Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., 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)

4. Exemplary script for enclosed files in package

The following script will run GSEA procedure based on KEGGs (`opts.GS_name`) for data set GSE6344. The calculated ranking metric will be absolute value (`opts.abs = true`) of signal to noise ratio (`opts.rank_type = 'S2N'`) for 1000 permutations (`opts.perm_nb`) of phenotypes (`opts.perm_type = 'pheno'`). Gene sets with gen no. lower than 15 or higher than 500 will be removed (`opts.GS_filt = [15,500]`).

```
clear all; close all; clc
% add path with package functions folder name
addpath('GSEA_package')

% creat gene set variable
update_GS('KEGG_GS.xlsx')

% set options
opts = default_GSEA_opts();
opts.show = true;          % plot results
opts.save = true;          % save results
opts.perm_nb = 1000;       % number of permutations
opts.perm_type = 'pheno'; % premutation type
opts.rank_type = 'S2N';    % ranking metric
opts.abs = true;           % ranking metric absolut value
opts.GS_filt = [15,500];   % Gene sets filtration


load('data_GSE6344.mat')
[res_pos,res_neg,res_descr,p_gene] =
MrGSEA(dataF,group,prob,'example_results',opts);
```