PeCorA workflow

Maria Dermit & Jesse G. Meyer

Last update: 19 October, 2020

1. Introduction

`PeCorA` (peptide correlation analysis) is an open-source R-based package for detection of quantitative disagreements between peptides mapped to the same protein. This document describes the most recent version of PeCorA package.

2. Prerequisites for PeCorA analysis

You can install `PeCorA` from github downloading the package by cloning the repository.

```
$ git clone https://github.com/jessegmeyerlab/PeCorA.git
```

```
$ R CMD INSTALL PeCorA-master
```

Alternatively you can install `PeCorA` directly from R using devtools:

```
library(devtools)
install_github("jessegmeyerlab/PeCorA")
```

Or you can install`PeCorA` from CRAN by typing in R: `install.packages("PeCorA")`

Once you have the package installed, load `PeCorA` into R.

```
library(PeCorA)
```

3. Importing data into PeCorA format

You can import mouse microglia proteomics data example included in PeCorA package using `import_processed_data` function. This csv file is in a PeCorA-ready format.

```
t<-import_processed_data("PeCorA_noZ.csv")
```

You can also import Covid LFQ data ouput of MaxQuant included in the the PeCorA package using the function `import_processed_data_for_PeCorA_LFQ`.

```
t2 <- import_processed_data_for_PeCorA_LFQ( LFQfile = "peptides.txt",
                                            condition1="CONTROL",
                                            condition2="_COVID",
                                            condition3="NON.COVID")
```

Checking the names of PeCorA-ready microglia data.frame

```
names(t)
#> [1] "Peptide"                   "Protein"
#> [3] "Peptide.Modified.Sequence" "Begin.Pos"
```

```
#> [5] "End.Pos"                    "Condition"
#> [7] "BioReplicate"               "Normalized.Area"
```

4. Scaling and centering peptides

`PeCorA_preprocessing` initially filters the values to include only precursors with measured MS1 areas in all samples. Next, the peak areas are log2 transformed, and the global distribution of all peak areas was scaled to have the same center. Finally, each peptide is center relative to the mean of the control group's peak area.

```
scaled_peptides <- PeCorA_preprocessing(t,
                                        area_column_name=8,
                                        threshold_to_filter=100,
                                        control_name="cntrl")
```

5. Running PeCorA analysis

`PeCorA` loops through proteins with >2 peptides, and records a linear model on the peptide precursors for each of those protein recording a adjust pvalue within each protein. It makes a dataframe with of the peptides that disagree, sorting smaller adj_pval values at the top of table.

$$\mathrm{y} = \beta_0 + \beta_1(\mathrm{X1}) + \beta_2(\mathrm{X2}) + \beta_3(\mathrm{X1})(\mathrm{X2}) \quad Where\text{:X1= peptide group, X2= treatment group}$$

```
disagree_peptides <-  PeCorA (scaled_peptides)
```

6. Plotting PeCorA results

Example plot of the the peptide with the most significant adjusted p-value from PeCorA in the microglia dataset.

```
PeCorA_plotting_plot<-PeCorA_plotting(disagree_peptides,disagree_peptides[1,],scaled_peptides)
PeCorA_plotting_plot
```

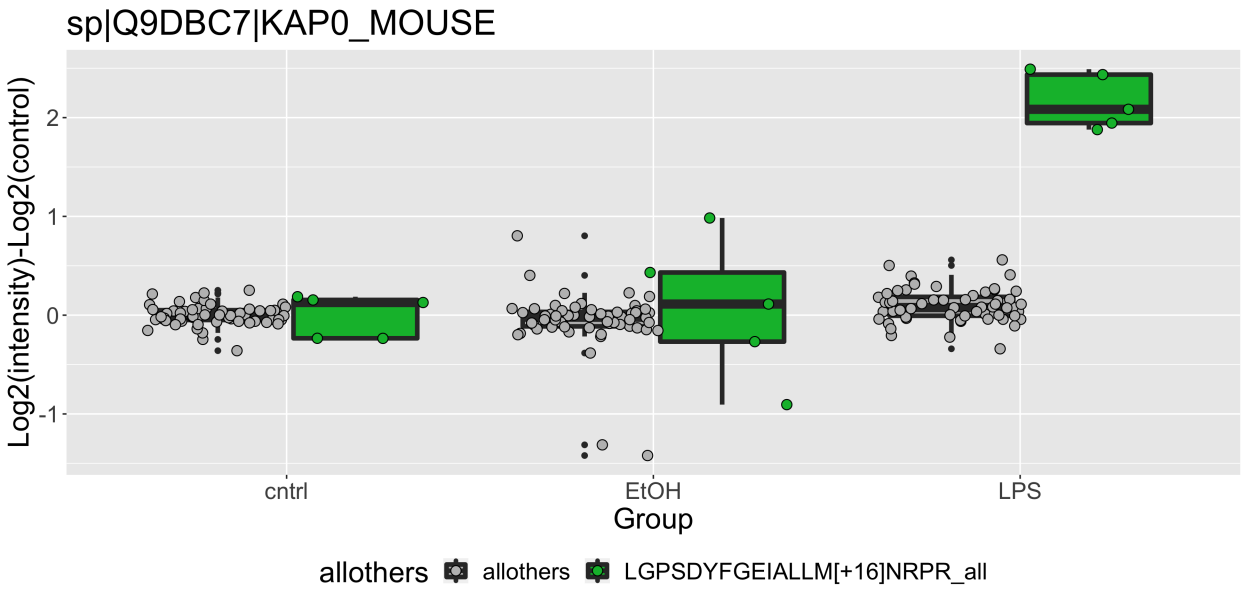

Figure 1: Example of interesting peptide revealed by PeCorA.

```
PeCorA can also be run in the command line using 'PeCorA_wrapper.R' like this:

$ Rscript PeCorA_wrapper.R PeCorA_noZ.csv out_TESTING.txt
cntrl Normalized.Area /Users/username01/Documents 0.01

The six arguments in order are:

Input file - example: PeCorA_noZ.csv
Output file - example out_TESTING.txt
Name of the control group within the "Conditions" column- example: cntrl
Name of column with peptide peak areas- example: Normalized.Area
Directory to write output table and images - example:
/Users/username01/Documents in mac
D:\output\directory in Windows
Threshold to use for significant p-value when printing plots
and printing summary results to console numbers -example:
0.01
```

Session information

```
sessionInfo(package = NULL)
#> R version 3.6.3 (2020-02-29)
#> Platform: x86_64-apple-darwin15.6.0 (64-bit)
#> Running under: macOS Sierra 10.12.6
#>
#> 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_GB.UTF-8/en_GB.UTF-8/en_GB.UTF-8/C/en_GB.UTF-8/en_GB.UTF-8
#>
```

```
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#> [1] PeCorA_0.0.0.9000 reshape_0.8.8     standardize_0.2.1 ggplot2_3.3.2
#>
#> loaded via a namespace (and not attached):
#>  [1] zip_2.1.1         Rcpp_1.0.5        cellranger_1.1.0  plyr_1.8.6
#>  [5] nloptr_1.2.2.2    pillar_1.4.6      compiler_3.6.3    forcats_0.5.0
#>  [9] tools_3.6.3       boot_1.3-25       digest_0.6.25     lme4_1.1-23
#> [13] statmod_1.4.34    evaluate_0.14     lifecycle_0.2.0   tibble_3.0.3
#> [17] gtable_0.3.0      nlme_3.1-149      lattice_0.20-41   pkgconfig_2.0.3
#> [21] rlang_0.4.8       openxlsx_4.2.2    Matrix_1.2-18     curl_4.3
#> [25] yaml_2.2.1        haven_2.3.1       xfun_0.18         rio_0.5.16
#> [29] withr_2.3.0       dplyr_1.0.2       stringr_1.4.0     knitr_1.30
#> [33] hms_0.5.3         generics_0.0.2    vctrs_0.3.4       grid_3.6.3
#> [37] tidyselect_1.1.0  data.table_1.13.0 glue_1.4.2        R6_2.4.1
#> [41] readxl_1.3.1      foreign_0.8-75    rmarkdown_2.4     carData_3.0-4
#> [45] minqa_1.2.4       car_3.0-10        purrr_0.3.4       magrittr_1.5
#> [49] scales_1.1.1      ellipsis_0.3.1    htmltools_0.5.0   MASS_7.3-53
#> [53] splines_3.6.3     abind_1.4-5       colorspace_1.4-1  stringi_1.5.3
#> [57] munsell_0.5.0     crayon_1.3.4
```