Introduction to PepSeq

Derek Sonderegger

2018-12-07

PepSeq is a package developed to facilitate the exploration of PMI’s PepSeq data. Given a pulldown, we want to explore how the cleaved-uncleaved signal varies across different locations in the proteins.

To begin, we will install the package via the following R commands:

library(devtools)  
install\_github('dereksonderegger/PepSeq') # install latest version of Derek's PepSeq tools

library(tidyverse) # load the usual set of data processing tools  
library(PepSeq) # load Derek's library of routines

The first step is to set the working directory to wherever the pulldown .csv file is located. This can be done via the point and click interface (Session -> Set Working Directory) or using the command setwd. Once the working directory is set to where ever the data file lives, you can define the input and output files.

setwd('~/GitHub/PepSeq/inst/extdata')  
input\_file <- 'PepSeq\_vs\_NN.csv'  
output\_file <- 'PepSeq\_vs\_NN.pdf'

Alternatively, you could just list the full pathname to the input and output files and ignore the working directory business.

input\_file <- '~/GitHub/PepSeq/inst/extdata/PepSeq\_vs\_NN.csv'  
output\_file <- '~/GitHub/PepSeq/inst/extdata/PepSeq\_vs\_NN.pdf'

PepSeq assumes that a pulldown .csv file is arrange in a format with one protein/location combination per line and then several sets of cleaved/uncleaved pairs and possibly columns of data that aren’t cleaved/uncleaved but rather just some other signal (e.g. output from another model as to the likelihood of binding).

read.csv(input\_file) %>% colnames()  
#> [1] "Protein\_ID" "Start\_Loc" "wTEV\_Cleaved"   
#> [4] "wTEV\_Uncleaved" "ModelPrediction" "Experimental\_Class"

Notice that in the example pulldown file, there is just one experiment (wTEV) and two other added columns model of neural net model predictions and if the number of literature citations. The orderings of all the columns doesn’t matter, but you need to be able to identify the protein and position columns and each of the cleaved/uncleaved pairs need to have the same prefix followed by an ’\_Cleaved’ or ’\_Uncleaved’. If there is no ‘Cleaved’ or ‘Uncleaved’ suffix on a column then we assume the column is a measurement and will be data to be visualized.

In order to allow there to be many other interesting columns that should be ignored in the visualization, PepSeq requires you to indicate which columns are to be plotted by either having an initial suffix indicating it is data or to indicate the responses by column location (e.g. columns 3 to 7).

plot\_pulldown(  
 file = input\_file, # input file  
 output\_file = output\_file, # output file name  
 protein\_column = 'Protein\_ID', # which is the protein name  
 position\_column = 'Start\_Loc', # which is the starting location  
 read\_indicator = 3:6, # column range to be visualized  
 peaks = FALSE ) # Don't calculate peaks  
  
plot\_pulldown(   
 file = input\_file,   
 output\_file = output\_file,   
 protein\_column = 'Protein\_ID',   
 position\_column = 'Start\_Loc',   
 read\_indicator = c(3,4,5,6), # or specify individual columns  
 peaks = FALSE )

Notice that in each of the calls, we also had to specify which column specified the protein name and which column denotes the position along the protein.

After either of the above are run, a pdf has been created either in the current working directory, or where ever you specified using when creating the output file string.

In all the examples previously, we’ve supressed the peaks highlighting. To turn peak highlighting on, you can just remove the peaks = FALSE, or explicitly ask for it via peaks = TRUE. To change the peak finding method or change the tuning parameter you can use the peak\_method and peak\_param options.

plot\_pulldown(   
 file = input\_file, # input file  
 output\_file = output\_file, # output file name  
 protein\_column = 'Protein\_ID', # which column is the protein name  
 position\_column = 'Start\_Loc', # which column is the starting location  
 read\_indicator = 3:6, # column range to be visualized  
 peaks = TRUE, # Show the peaks  
 peak\_method = 'PoT', # Use "Peaks over Threshold"  
 peak\_param = c(.5, 90, 10) ) # thresholds are by row

Finally if we want to look at the interactive version, all the options are the same, but we don’t need the output\_file.

plot\_pulldown\_Shiny(   
 file = input\_file, # input file  
 protein\_column = 'Protein\_ID', # which is the protein name  
 position\_column = 'Start\_Loc', # which is the starting location  
 read\_indicator = 3:6, # column range to be visualized  
 peaks = TRUE, # Show the peaks  
 peak\_method = 'PoT', # Use "Peaks over Threshold"  
 peak\_param = c(.5, 90, 10) ) # thresholds are by row