Proteus: a simple R package for downstream analysis of *MaxQuant* data

Marek Gierlinski[*]
Francesco Gastaldello[†]

Abstract

Proteus is a package for quick and easy downstream analysis of *MaxQuant* evidence data in R environment. It provides a variety of tools for peptide and protein aggregation, quality checks, data exploration and visualisation. Differential expression is done with *limma*, offering more robust treatment of data gaps than random imputation. Availability and implementation: The open-source R package is available to install from GitHub (https://github.com/bartongroup/Proteus).

1 Introduction

MaxQuant is one of the most popular tools for analyzing mass spectrometry (MS) quantitative proteomics data (J. Cox and Mann 2008). The output of a *MaxQuant* run usually consists of several tables, including the evidence data and summarized peptide and protein intensities. The downstream analysis and understanding of these data are essential for interpreting peptide and protein quantification. The standalone *Perseus* software package (Tyanova et al. 2016) is often used in conjunction with *MaxQuant*.

Proteus offers simple but comprehensive downstream analysis of *MaxQuant* output in R environment (R Core Team 2018). The package is built with simplicity and flexibility of analysis in mind. On one hand, a user unfamiliar with R can obtain differential expression results with a few lines code following the tutorial. On the other hand, a more experienced R programmer can perform advanced analysis using a plethora of R and Bioconductor packages (Huber et al. 2015).

2 Methods

Proteus analysis begins with reading the evidence file. To conserve memory only essential columns are retained. Reverse sequences and contaminants are rejected by default. In the current version unlabeled, TMT and SILAC data can be used.

Peptide measurements (intensities or SILAC ratios) are aggregated from individual peptide entries with the same sequence or modified sequence. Quantification is carried out as the sum (unlabeled or TMT) or median (SILAC) of individual measurements. A user-defined function for peptide aggregation can be provided.

Protein intensities for unlabeled and TMT data are aggregated, by default, using the high-flyer method, where protein intensity is the mean of the three top-intensity peptides (Silva et al. 2006), though the sum of intensities can be used as well. For SILAC experiments, median ratio is calculated. Again, a user-provided function for protein aggregation can be used instead.

The ability to aggregate peptide and protein data according to any prescription gives the package flexibility. On the other hand, the default, predefined aggregation functions make the package very easy to use. The full analysis can be done in just a few lines of R code.

[*]The Barton Group, School of Life Sciences, University of Dundee, Dundee, UK
[†]Biological Chemistry and Drug Discovery, University of Dundee, Dundee, UK

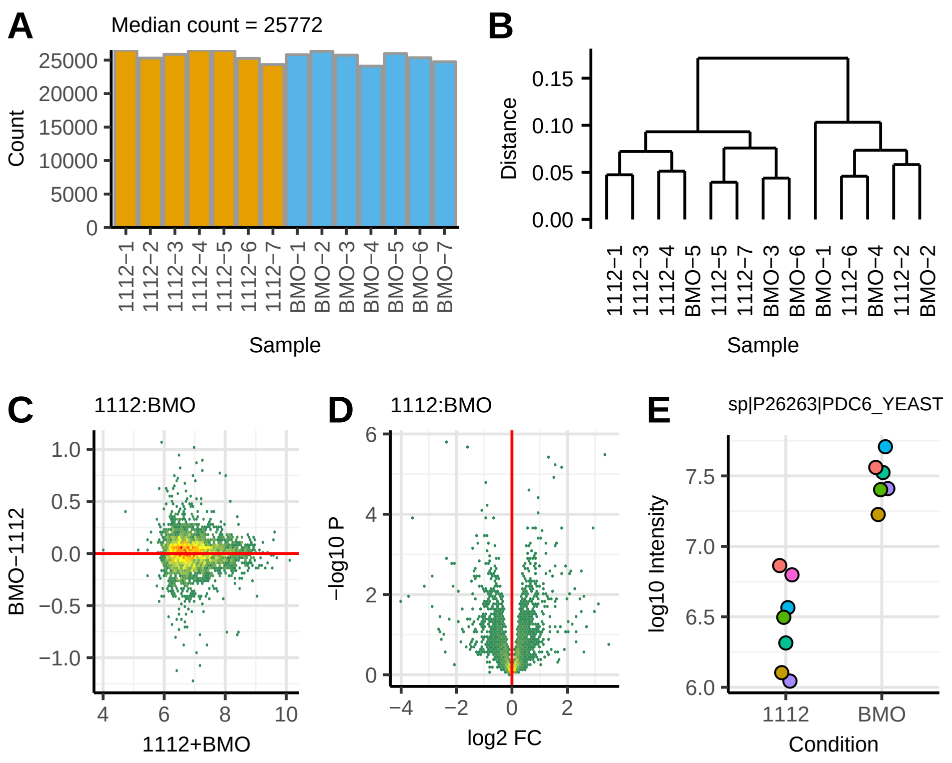

Figure 1: Visualization in Proteus. A. Peptide count per sample. B. Clustering of samples at protein level. C. Fold-change versus intensity for protein data. D. Volcano plot following differential expression analysis for protein data. E. Log-intensities of replicates in two conditions for a selected protein.

Peptide or protein data are encapsulated in an R object together with essential information about the experiment design, processing steps and some useful statistics. Either object can be used for further processing, that is, analysis can be done on peptide or protein level, using the same functions. The basic analysis and visualization includes peptide/protein count (Fig. 1A), sample comparison, correlation and clustering (Fig. 1B). Measurements can be normalized between samples using any arbitrary function, e.g., to the median or quantiles. A pair of conditions can be compared in a fold-change/intensity plot (Fig. 1C). The package provides functions to fetch protein annotations from UniProt servers.

Bioconductor package *limma* (Ritchie et al. 2015) is used for differential expression analysis. It is a well-established and robust differential expression tool. Since MS experiments usually create a lot of gaps in data it is crucial how these gaps are treated before differential expression tests. *limma* offers an advantage over random imputation methods by borrowing information across peptides or proteins and using the mean-variance relationship to estimate variance where data are missing. The results can be visualised as a volcano plot (Fig. 1D). Intensities or ratios from individual proteins can be plotted across all conditions (Fig. 1E), including breakdown into constituent peptides. The package offers interactive data explorer based on the Shiny framework.

2.1 A minimal example

Here we present an example of data processing from the evidence file to the differential expression analysis. The input data consists of the *MaxQuant*'s evidence file and a manually-created simple metadata text file with information about sample names and biological conditions.

```
evi <- readEvidenceFile("evidence.txt")
```

```
meta <- read.delim("metadata.txt", header=TRUE, sep="\t")
pepdat <- makePeptideTable(evi, meta)
prodat <- makeProteinTable(pepdat)
prodat.med <- normalizeData(prodat)
res <- limmaDE(prodat.med)
```

The final object `res` is a data frame containing log-fold-changes and p-values (both raw and multiple-test adjusted) for each protein.

3 Results

Here we compare performance of *Perseus* and *Proteus*. *Perseus* is a commonly used *MaxQuant* data analysis tool with a graphical user interface, available in MS Windows. We chose two cases for this comparison. First, we analyse a label-free proteomics data set in two conditions and three replicates each. Second, we create a simulated data set based on a large real data set to investigate power and false positives from both tools. We focus on performance of differential expression, which is done by a t-test in *Perseus* and *limma* in *Proteus*.

3.1 Simple data set

First we focused on the differential expression offered by both packages using the same protein data set. We used a subset of a large data set from Gierlinski et al. (in preparation). The set was prepared in *Proteus*. We read the evidence file and filtered out reverse sequences and contaminants. Then, we created peptide table based on the tree randomly selected replicates in each condition (samples 1083-7, 1083-28, 1083-34, WT-8, WT-13, WT-25). Peptides were aggregated by summing multiple evidence entries for a given (unmodified) sequence. Next, we aggregated peptides into proteins using the high-flyer method and peptide-to-protein mapping based on the leading razor protein. We filtered protein intensities in these replicates, so at least one data point was present in each condition. These data were normalized to median (that is, after normalization median intensity in each sample was the same). The resulting intensity table containing 3338 proteins in two conditions in three replicates each was processed in *Perseus* and *Proteus*.

In *Perseus* we imported data from a file and log-2-transformed. Missing data were filled with random imputation, using the default parameters, width = 0.3 and down shift = 1.8. Then, we performed two-sample t-test. In *Proteus* data were log-2-transformed and differential expression was performed by *limma*. Since the protein intensities were the same, we compare the difference between imputation plus t-test versus *limma* (without imputation).

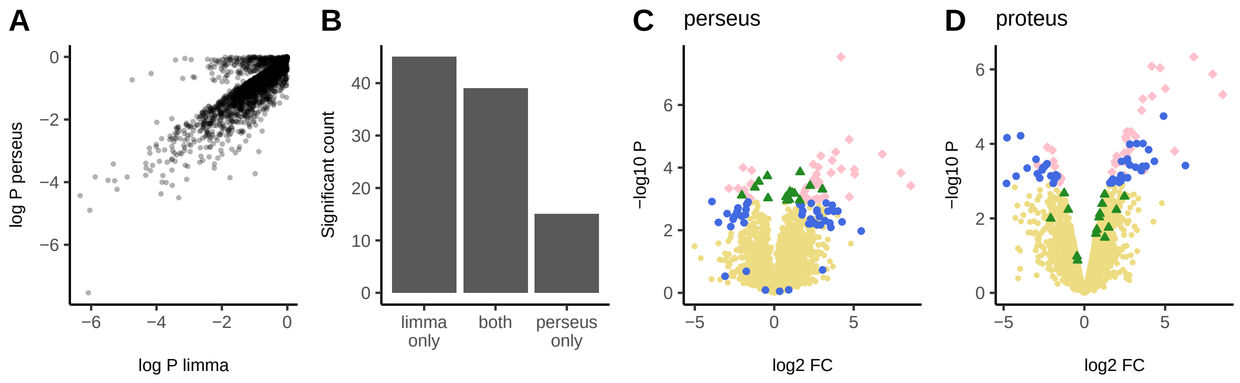

Figure 2: Perseus vs limma for a selection of 3 vs 3 replicates. A. P-value (not adjusted) comparison. B. Number of significant proteins. C. Volcano plot using fold-change and p-values from Perseus. All data are in yellow background. The limma-significant-only proteins are as blue circles, the perseus-significant-only proteins are as green triangles. Proteins significant in both tools are as pink diamonds. D. Volcano plot using fold-change and p-values from Proteus. Symbols are the same as in C.

t-test results from *Perseus* were exported as a generic table and loaded into R environment for comparison to *Proteus* results. Figure 2 shows the comparison of p-values, significantly differentially expressed proteins and volcano plots for both approaches. There are 39 proteins called as significant by both methods, 15 only by *Perseus* and 45 only by *limma* (in *Proteus*). We can see in Figure 2C a small group of proteins called by *limma* only where *Perseus* reported large p-values (6 blue circles at the bottom of the plot). These proteins have missing data and rather large intensity. Imputation in *Perseus* filled the gaps with low intensities, inflating variance and missing what otherwise would be differentially expressed. An example of such protein is shown in Figure 3A. Another group of blue circles in Figure 2C indicates that the permutation FDR method used in *Perseus* is slightly more conservative that that in *limma*. All these proteins have adjusted p-values near the limit 0.05. An example of such protein is shown in Figure 3B. The proteins plotted as green triangles (see Figure 2 C and D) are marked as differentially expressed by *Perseus* but not *limma*. They typically have small variance and small fold change. They are called significant by a simple t-test but no by *limma*, which moderates variance and avoids cases of unusually small variability. An example of such protein is shown in Figure 3C. We note that these data can be easily eliminated from *Perseus* by setting a fold-change limit in t-test.

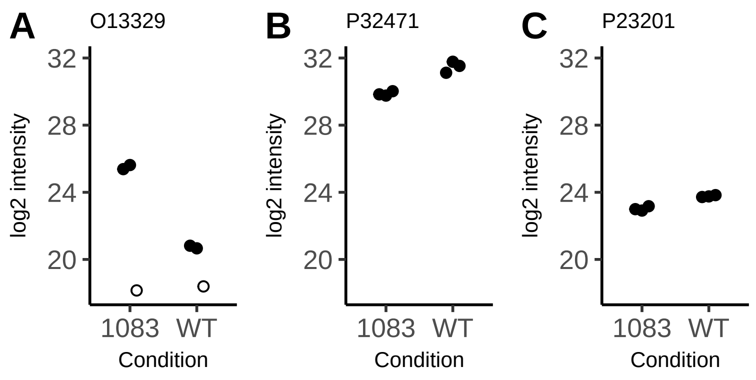

Figure 3: Selected examples of proteins called as significant by one tool only. A. Called by limma only. Imputation in Perseus inflated variance creating a false negative. B. Called by limma only. Perseus FDR is more conservative than that in limma at the same limit of 0.05. C. Called by Perseus only. An example of very low variance, which is moderated and called negative by limma. Data imputed by Perseus are marked with open circles.

The imputation in *Perseus* is designed to fill missing low-intensity data with a randomly generated Gaussian numbers (see supplemental figure 3 in Tyanova et al. (2016)). However, on some occasions data from a replicate can be missing even for high-intensity data. In such cases variance is dramatically inflated and the protein is not called as differentially expressed. We warn against using data imputation. *limma* offers a better approach to missing data, by modelling mean-intensity variance and using moderated variance for the test. Certainly, the imputation step can be omitted in *Perseus*, but this reduces power and makes analysis of data with only one replicate impossible in one condition. Again, *limma* can estimate variance and make a decision about differential expression even in such extreme cases (at an increased risk of a false positive).

3.2 Simulated data

Next, we compared performance of differential expression in both tools using simulated data. We generate the simulated set based on real data. Since we have a good data set of two conditions in 35 replicates each (Gierlinski at al. in preparation), we used it to find the mean-variance relationship and the rate of missing values as a function of the intensity.

Figure 4A shows the relation between the logarithm of the variance and logarithm of the mean calculated across all 35 replicates (data from both conditions overlapped). It is well approximated by a straight line. Figure 4B shows the distribution of the number of “good” replicates as a function of the logarithm of the mean intensity. The good replicates are those with signal detection, as opposed to missing data. We see that in our data set all measurements with $\log_2$ mean below ~19 contain only one good replicate.

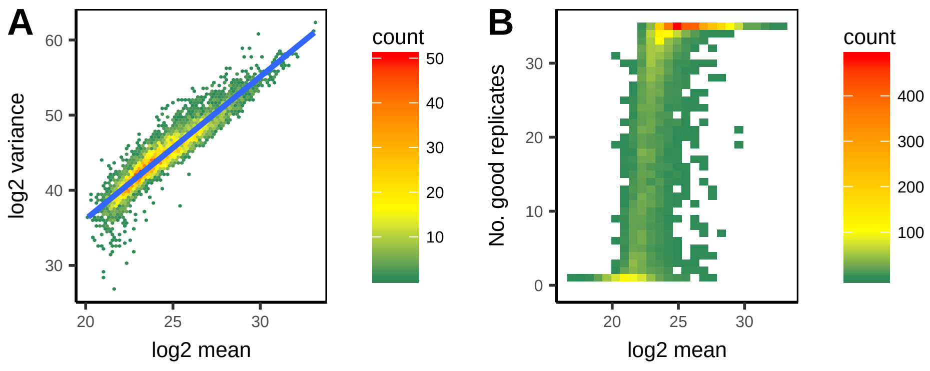

Figure 4: Properties of the full data set with 35 replicates in two conditions. A. Logarithm of the variance versus logarithm of the mean is very well approximated by a linear function. B. Number of good replicates as a function of the loarithm of the mean intensity (that is not missing data). Data from both conditions were aggregated.

We used this information to create a simulated data set. We generated data in two conditions in three replicates each and allow for missing data in each condition. We chose 7 values of $\log_2 M$ (mean) between 17 and 29 and 15 values of $\log_2 FC$ (fold change) between 0 and 2.8. For each combination of $\log_2 M$ and $\log_2 FC$ we generated two random samples of up to 3 data points from the log-normal distribution with the given mean and variance estimated from the linear function found from real data. The first sample has the mean M, the second sample has the mean $M * FC$. For each sample the number of good replicates is generated based on data in Figure 4B. First, for the given M, we use the cumulative distribution of the number of good replicates to generate a number between 1 and 35. This is then sub-sampled to the 3 replicates generated (for example, if 10 is generated in the first step, we create a vector of 10 good and 25 bad replicates and draw a random sample of 3). Since we are not interested in samples with no data, we enforce at least one good replicate in each sample. This means that data with only one good replicate will be over-represented for very low intensities. This is not an issue as our aim is to assess tool performance at each intensity level and low intensities will invariably contain a lot of missing data. For each combination of

$\log_2 M$ and $\log_2 FC$ we generated 1000 samples in two conditions, using this technique. This gives us a large set of 105,500 "proteins" covering a wide range of intensities and fold changes.

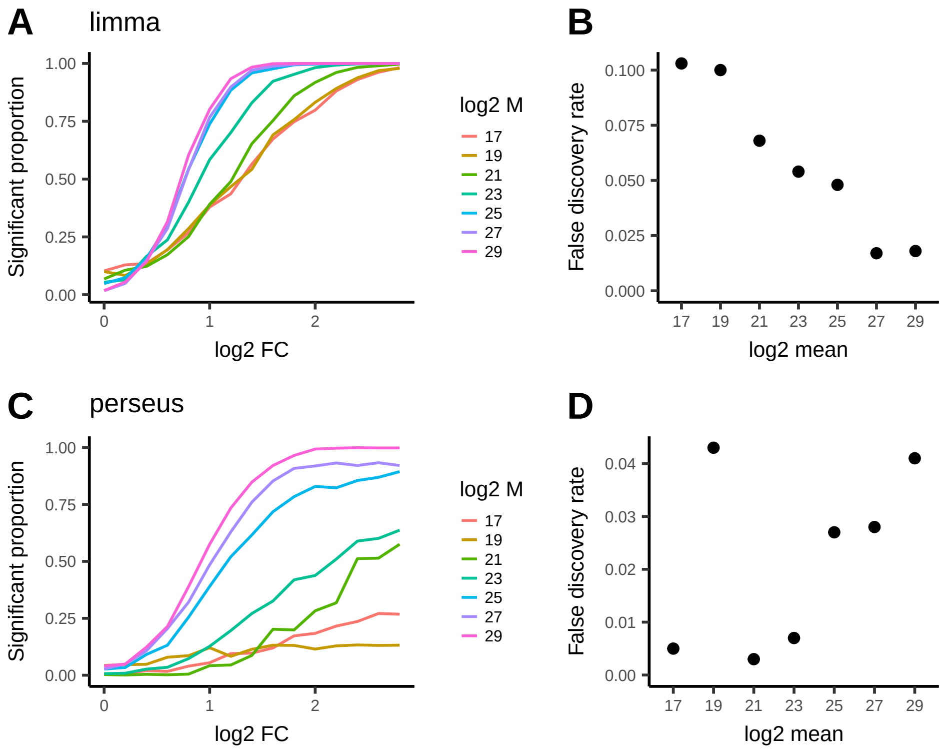

Figure 5: Results for the full set of simulated data with 3 replicates. Top panels show limma, bottom panles show Perseus results. A and C show the proportion of tests called as signficant as a function of the simulated fold change (FC) and mean (M). B and D show the false discovery rate, that is the proportion of tests for simulated log FC = 0 called as significant.

Then, we performed the differential expression on the simulated data using *Perseus* and *limma*. In *Perseus* we imported simulated data from a file, log2-transformed, applied default imputation and used a two-sample t-test. In *limma* log2-transformed data were used directly. The results are shown in Figure 5. Panels A and C show the proportion of proteins called significant in a group of 1000 proteins for each combination of fold change and mean intensity. We can see that *limma* performs well across all intensities, discovering almost all positives for the highest $\log_2 FC = 2.8$ used here. In contrast, the sensitivity of *Perseus* drops dramatically at low intensities. Even at medium intensities of $\log_2 M = 19$ only about half of the changing proteins are discovered at large fold changes of $\log_2 FC = 2$. See also Figure 7A.

The main reason for this behaviour is imputation of missing replicates. We notice that due to the way simulated data were generated, all proteins for the lowest intensity $\log_2 M = 17$ contain only one good replicate in each condition. As t-test cannot deal with samples of one, imputation is necessary and the result is randomized. On the other hand, *limma* borrows information across the entire set and builds a reliable model of variance which works for any sample size. As we can see from the bottom curve in Figure 5A (corresponding to $\log_2 M = 17$) *limma* performs really well even in tests of one versus one replicate.

The price to pay for increased sensitivity of *limma* is the increased false discovery rate (FDR). We can estimate FDR as a proportion of proteins called significant at $\log_2 FC = 0$. Figure 5B shows that FDR for *limma* exceeds the assumed limit of 0.05 at the three lowest intensities. In contrast, *Perseus* controls the FDR well (Figure 5D).

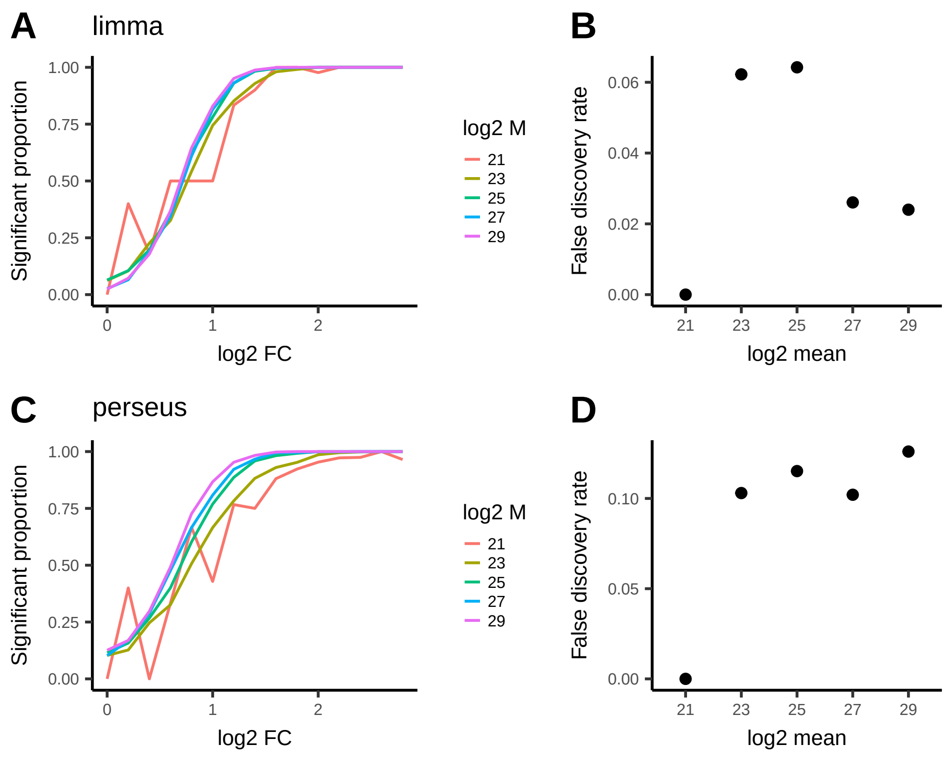

Figure 6: Results for the filtered set of simulated data with 3 replicates. Only data with at least 2 replicates in each condition were used. Panels are the same as in Figure 5.

Because imputation is clearly an issue we decided to compare *limma* and *Perseus* using data that do not require imputation. We used the same simulated data set, but filtered out all proteins with only one good replicate in either condition. Filtering low-replicate data would reflect a more realistic workflow for a researcher who doesn't want to apply imputation. After filtering we processed data in *Perseus* and *limma* as before, but skipped the imputation step in *Perseus*. Results are shown in Figure 6. Not surprisingly, the significant proportion of *limma* and *Perseus* are now more similar, though *limma* still offers slight advantage (see also Figure 7B). The false discovery rate is now better in *limma* than in *Perseus* where 4 out of 5 intensity groups result in $FDR \sim 0.1$.

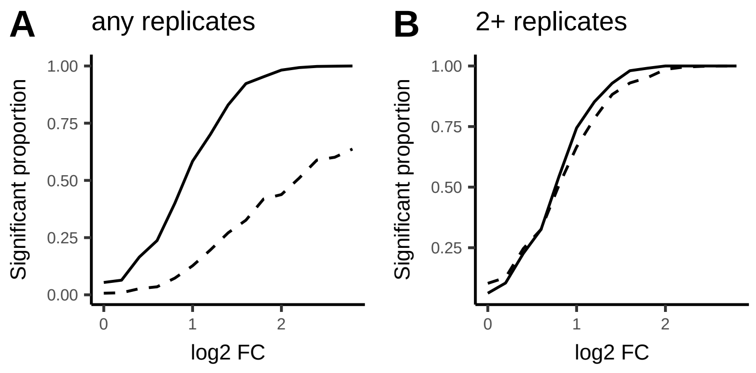

Figure 7: Direct comparison of significance curves from Figs. 5 (A) and 6 (B) for *Perseus* (dahsed curves) and *limma* (solid curves).

4 Conclusions

Proteus allows easy and flexible analysis of *MaxQuant* output in R environment. It uses a powerful package *limma* for differential analysis. It offers an alternative to a popular package *Perseus* for researchers willing to use R.

References

Cox, Jürgen, and Matthias Mann. 2008. "MaxQuant Enables High Peptide Identification Rates, Individualized P.p.b.-Range Mass Accuracies and Proteome-Wide Protein Quantification." *Nature Biotechnology* 26 (12): 1367–72. doi:10.1038/nbt.1511.

Huber, W., Carey, V. J., Gentleman, R., Anders, et al. 2015. "Orchestrating High-Throughput Genomic Analysis with Bioconductor." *Nature Methods* 12 (2): 115–21. http://www.nature.com/nmeth/journal/v12/n2/full/nmeth.3252.html.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. http://www.R-project.org/.

Ritchie, Matthew E., Belinda Phipson, Di Wu, Yifang Hu, Charity W. Law, Wei Shi, and Gordon K. Smyth. 2015. "Limma Powers Differential Expression Analyses for RNA-Sequencing and Microarray Studies." *Nucleic Acids Research* 43 (7). Oxford University Press (OUP): e47–e47. doi:10.1093/nar/gkv007.

Silva, J. C., M. V. Gorenstein, G. Z. Li, J. P. Vissers, and S. J. Geromanos. 2006. "Absolute quantification of proteins by LCMSE: a virtue of parallel MS acquisition." *Mol. Cell Proteomics* 5 (1): 144–56.

Tyanova, Stefka, Tikira Temu, Pavel Sinitcyn, Arthur Carlson, Marco Y Hein, Tamar Geiger, Matthias Mann, and Jürgen Cox. 2016. "The Perseus Computational Platform for Comprehensive Analysis of (Prote) Omics Data." *Nature Methods.* Nature Research.