Package “"proteoQ"”

Qiang Zhang

2019-05-16

Contents

Introduction to proteoQ 1
Installation 1
Application 1

Introduction to proteoQ

Chemical labeling using tandem mass tag (TMT) has been commonly applied in mass spectrometry (MS)-based quantification of proteins and peptides. The proteoQ tool is designed to aid automated and reproducible analysis of proteomics data. It interacts with an `Excel` spread sheet for dynamic sample selections, aesthetics controls and statistical modelings. The arrangement allows end users to put data behind the scene and quickly address interesting biological questions using various informatic tools. In addition, the entire workflow is documented and can be conveniently reproduced upon revisiting.

The tool currently processes the peptide spectrum matches (PSM) tables from Mascot searches for 6-, 10- or 11-plex TMT experiments. Peptide and protein results are then produced with users' selection of parameters in data filtration, alignment and normalization. The package further offers a suite of tools and functionalities in statistics, informatics and data visualization by creating 'wrappers' around published R functions.

Installation

To install this package, start R (version “3.6”) and enter:

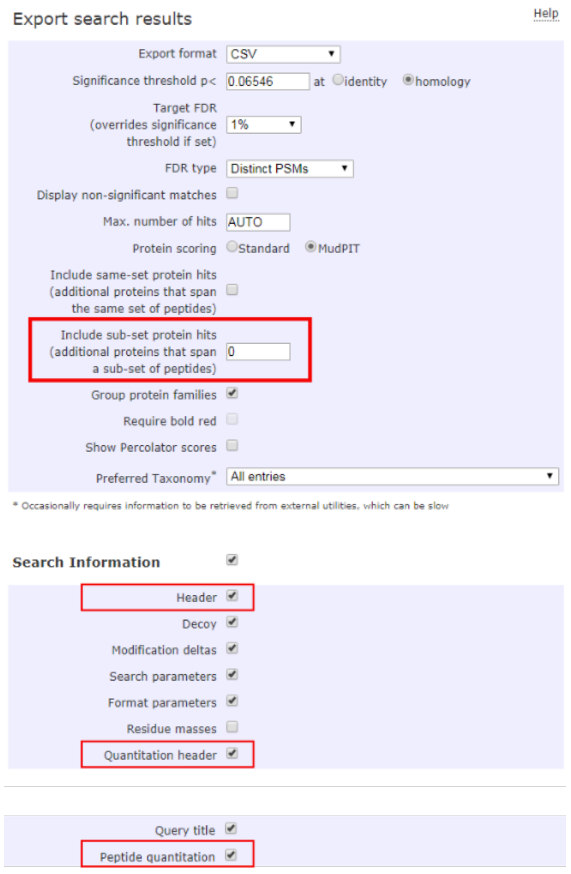
```r
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install(c("Biobase", "GSVA", "Mfuzz", "gage", "limma"))

if (!requireNamespace("devtools", quietly = TRUE))
    install.packages("devtools")
devtools::install_github("qzhang503/proteoQ@master")
```

Application

In this section I illustrate the following applications of `proteoQ`:

- Summarization of PSM data to peptide and protein reports.
- Basic informatic analysis against the peptide and protein data.

Set up the experiments



Export search results

Help

Export format	CSV
Significance threshold p<	0.06546 at ○ identity ● homology
Target FDR (overrides significance threshold if set)	1%
FDR type	Distinct PSMs
Display non-significant matches	☐
Max. number of hits	AUTO
Protein scoring	○ Standard ● MudPIT
Include same-set protein hits (additional proteins that span the same set of peptides)	☐
Include sub-set protein hits (additional proteins that span a sub-set of peptides)	0
Group protein families	✓
Require bold red	☐
Show Percolator scores	☐
Preferred Taxonomy*	All entries

* Occasionally requires information to be retrieved from external utilities, which can be slow

Search Information	✓
Header	✓
Decoy	✓
Modification deltas	✓
Search parameters	✓
Format parameters	✓
Residue masses	☐
Quantitation header	✓
Query title	✓
Peptide quantitation	✓

Figure 1: Mascot_export

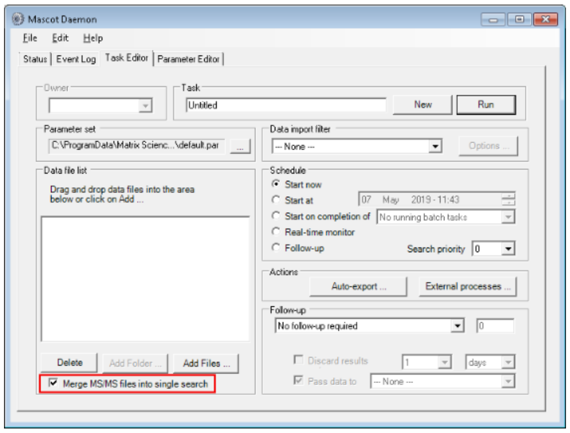
```r
# Load the proteoQ library
library(proteoQ)

# Set up the working directory
dat_dir <- "c:\\The\\First\\Example"
```

PSM table(s) in a `csv` format will be exported by the users from the Mascot search engine. I typically set the option of `Include sub-set protein hits` to `0` with my opinionated choice in satisfying the principle of parsimony. The options of `Header` and `Peptide quantitation` should be checked to include the search parameters and quantitative values. The `filename(s)` of the export(s) will be taken as is.[1]

The same peptide sequence under different PSM files can be assigned to different protein IDs when inferring proteins from peptides using algorithms such as greedy set cover. To avoid such ambiguity in protein inference, I typically enable the option of `Merge MS/MS files into single search` in Mascot Daemon. If the option is disabled, peptide sequences that have been assigned to multiple protein IDs will be removed for now when constructing peptide reports.

The pacakge reads an `Excel` template containing the metadata of multiplex experiment numbers, including TMT channels, LC/MS injection indices, sample IDs, corresponding RAW data file names and addditional fields from the users. The default file name for the experimental summary is `expt_smry.xlsx`. If samples were fractionated off-line prior to `LC/MS`, a second `Excel` template will also be filled out to link multiple `RAW`

[1]The default file names begin with letter `F`, followed by six digits and ends with `.csv` in file name extension.



Owner:
Task: Untitled
Parameter set: C:\ProgramData\Matrix Scienc...\default.par
Data import filter: --- None ---
Data file list: Drag and drop data files into the area below or click on Add ...
[✓] Merge MS/MS files into single search
Schedule: (●) Start now ( ) Start at 07 May 2019 - 11:43 ( ) Start on completion of No running batch tasks ( ) Real-time monitor ( ) Follow-up
Search priority: 0
Follow-up: No follow-up required 0
[ ] Discard results 1 days
[✓] Pass data to --- None ---

Figure 2: Mascot_daemon

file names that are associated to the same sample IDs. The default file name for the fractionation summary is `frac_smry.xlsx`. The function, `extract_raws`, can be used to summarise `.raw` file names under a file folder:

```r
# Supposed the RAW files are under the `raw_dir` folder
extract_raws(raw_dir)
```

Note that the above files should be stored immediately under the the file folder specified by `dat_dir`. Examples of PSM outputs, `expt_smry` and `frac_smry` can be found as the follows:

```r
system.file("extdata", "F012345.csv", package = "proteoQ")
system.file("extdata", "expt_smry.xlsx", package = "proteoQ")
system.file("extdata", "frac_smry.xlsx", package = "proteoQ")
```

and the description of the column keys in the `Excel` files can be found from the help document:

```r
?load_expts
```

As a final step of the setup, we will load the experimental summary and some precomputed results:

```r
# Load the experiment
load_expts()
```

Summarize PSMs to peptides and proteins

Process PSMs — In this section, I demonstrate the summarisation of PSM data to peptides and proteins. The data set I use in this section corresponds to the proteomics data from Mertins et al.(2018). In the study, two different breast cancer subtypes, WHIM2 and WHIM16, from patient-derived xenograft models were assessed by three independent laborotories. Under each location, lysates from WHIM2 and WHIM16 were each split and labeled with 10-plex TMT at equal sample sizes and repeated on a different day. We start by processing PSM data from `Mascot` outputs:

```r
# Generate PSM reports
normPSM(
 rptr_intco = 1000,
 rm_craps = FALSE,
 rm_krts = FALSE,
 rm_outliers = FALSE,
 plot_violins = TRUE
)

# or accept the default in parameters
normPSM()
```

PSM outliers will be assessed at a basis of per peptide and per sample at `rm_outliers = TRUE`, which can be a slow process for large data sets. To circumvent repeated efforts in the assessment of PSM outliers, we may set `rm_outliers = FALSE` and `plot_violins = TRUE` when first executing `normPSM()`. We then visually inspect the violin plots of reporter-ion intensity. Empirically, PSMs with reporter-ion intensity less than 1,000 are trimmed and samples with median intensity that is 2/3 or less to the average of majority samples are removed from further analysis.[2]

Summarize PSMs to peptides — We next summarise PSM to peptides.

```
# Generate peptide reports
normPep(
 id = pep_seq_mod,
 method_align = MGKernel,
 n_comp = 2,
 range_log2r = c(20, 95),
 range_int = c(5, 95)
)
```

At `id = pep_seq_mod`, peptide sequences that are different in variable modificaitons will be treated as different species. The log2FC of peptide data will be aligned by median centering across samples by default. If `method_align = MGKernel` is chosen, log2FC will be aligned under the assumption of multiple Gaussian kernels.[3] The parameter `n_comp` defines the number of Gaussian kernels. The parameters `range_log2r` and `range_int` define the range of log2FC and the range of reporter-ion intensity, respectively, for use in the scaling of standard deviation across samples.

Let's compare the log2FC profiles with and without scaling normalization:[4]

```
# without the scaling of log2FC
pepHist(
 scale_log2r = FALSE,
 ncol = 10
)

# with the scaling of log2FC
pepHist(
 scale_log2r = TRUE,
 ncol = 10
)
```

There are 60 panels of of histograms in each plot, which may not be easy to explore as a whole. In stead, we will break the plots down by their data origins. We begin with modifying the `expt_smry.xlsx` file by adding

[2]The sample removal and PSM re-processing can be achieved by deleting the corresponding entries under the column `Sample_ID` in `expt_smry.xlsx`, followed by the re-load of the experiment, `load_expts()`, and the re-execution of `normPSM()` with desired parameters.

[3]Density kernel estimates can occasionally capture spikes in the profiles of log2FC for data alignment. Users will need to inspect the alignment of ratio histograms and may optimize the data normalization with different combinations of tuning parameters before proceeding to the next steps.

[4]`normPep()` will report log2FC results both before and after the scaling of standard deviations.

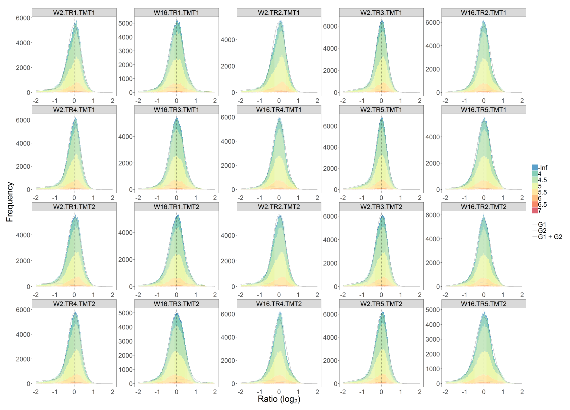
the columns `Select_BI`, `Select_JHU` and `Select_PNNL`. Each of the new columns includes sample entries that are tied to their laboratory origins.

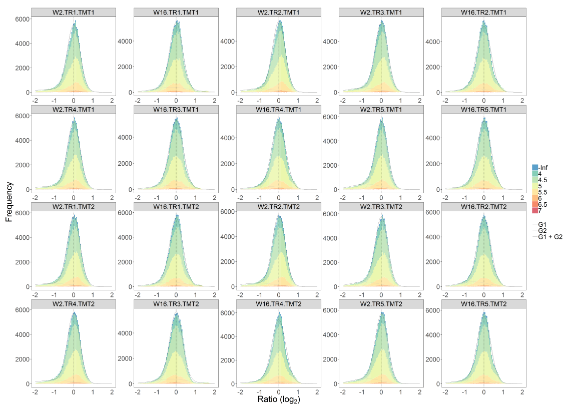
https://www.youtube.com/embed/y0VuWLUpcek

We now are ready to plot histograms for each subset of data.[5] In the tutorial, we only display the plots using the `BI` subset:

```r
# without the scaling of log2FC
pepHist(
 scale_log2r = FALSE,
 col_select = Select_BI,
 filename = "Hist_BI_N.png",
 ncol = 5
)

# with the scaling of log2FC
pepHist(
 col_select = Select_BI,
 filename = "Hist_BI_Z.png",
 ncol = 5
)
```

```
*NB*: We interactively told `pepHist()` that we are interested in sample entries under the newly create
```





As expected, the widths of log2FC profiles are more similar to each other after the scaling normalization. However, such adjustment may cause artifacts when the standard deviaiton across samples are genuinely different. I typically test `scale_log2r` at both `TRUE` and `FALSE`, then make a choice in data scaling together with my a priori knowledge of the characteristics of samples.[6] Alignment of log2FC against housekeeping or normalizer protein(s) is also available. This seems suitable when the quantities of proteins of interest are different across samples where the assumption of constitutive expression for the vast majority of proteins may not hold.

Summarize peptides to proteins — We then summarise peptides to proteins using a two-component Gaussian kernel.

[5] system files will be automatically updated from the modified `expt_smry.xlsx`

[6] The default is `scale_log2r = TRUE` throughout the package. When calling functions involved parameter `scale_log2r`, users will specify explicitly `scale_log2r = FALSE` to overwrite the default. Although the package provides the facility to look for a global setting of `scale_log2`, I don't recommend using it.

```r
# Generate protein reports
normPrn(
 id = gene,
 method_pep_prn = median,
 method_align = MGKernel,
 range_log2r = c(20, 90),
 range_int = c(5, 95),
 n_comp = 2,
 seed = 246,
 fasta = "C:\\Results\\DB\\Refseq\\RefSeq_HM_Frozen_20130727.fasta",
 maxit = 200,
 epsilon = 1e-05
)
```

Similar to the peptide summary, we inspect the alignment and the scaling of ratio profiles, and re-normalize the data if needed.[7]

```r
# without the scaling of log2FC
prnHist(
 scale_log2r = FALSE,
 ncol = 10
)

# with the scaling of log2FC
prnHist(
 scale_log2r = TRUE,
 ncol = 10
)
```

MDS and PCA plots

In this section, we visualize MDS, PCA and Euclidean distance against the peptide data at `scale_log2r = TRUE`. We start with metric MDS for peptide data:

```r
# data from all three laboratories
pepMDS(
    show_ids = FALSE
)
```

It is clear that the WHIM2 and WHIM16 samples are well separated by the Euclidean distance of log2FC (**Figure 2A**). We next take the `JHU` data subset as an example to explore batch effects in the proteomic sample handling:

```r
# `JHU` subset
pepMDS(
  col_select = Select_JHU,
  filename = "MDS_JHU.png",
  show_ids = FALSE
)
```

[7]Prameter `fasta` is solely used for the calculation of protein percent coverage. Precomputed data will be used if no `fasta` database is provided.

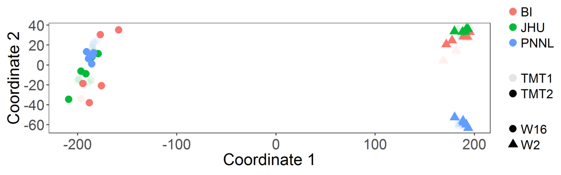
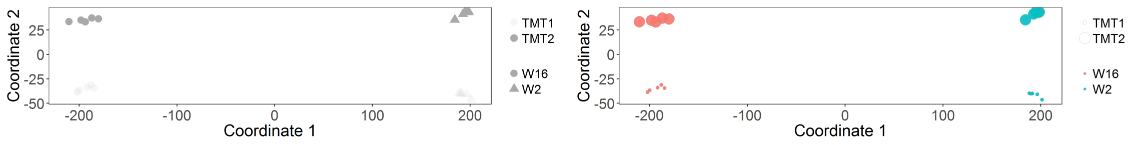


Figure 3: Peptide_MDS



We immediately spot that all samples are coded with the same color (**Figure 2B**). This is not a surprise as the values under column `expt_smry.xlsx::Color` are exclusively `JHU` for the `Select_JHU` subset. For similar reasons, the two different batches of `TMT1` and `TMT2` are distinguished by transparency, which is governed by column `expt_smry.xlsx::Alpha`. We may wish to modify the aesthetics using different keys: e.g., color coding by WHIMs and size coding by batches, without the recourse of writing new R scripts. One solution is to link the attributes and sample IDs by creating additional columns in `expt_smry.xlsx`. Fortunately, in this example, we have coincidentally prepared the column `Shape` and `Alpha` to code WHIMs and batches, respectively. Therefore, we can recycle them directly to make a new plot (**Figure 2C**):

```
# `JHU` subset
pepMDS(
  col_select = Select_JHU,
  col_fill = Shape, # WHIMs
  col_size = Alpha, # batches
  filename = "MDS_JHU_new_aes.png",
  show_ids = FALSE
)
```

The `prnMDS` performs `MDS` for protein data. For `PCA` analysis, the corresponding functions are `pepPCA` and `prnPCA` for peptide and protein data, respectively.

As mentioned at the begining of this section, `MDS` approximates Euclidean distances at a low dimensional space. Sometime it may be useful to have an accurate view of the distance matrix. Functions `pepEucDist` and `prnEucDist` plot the heat maps of Euclidean distance matrix for peptides and proteins, respectively. They are wrappers of (`pheatmap`) and inherit many parameters therein. Supposed that we are interested in visualizing the distance matrix for the `PNNL` subset:

```
# `PNNL` subset
pepEucDist(
    col_select = Select_PNNL,
    annot_cols = c("Shape", "Alpha"),
```

```r
    annot_colnames = c("WHIM", "Batch"),

    # parameters from `pheatmap`
    display_numbers = TRUE,
    number_color = "grey30",
    number_format = "%.2f",

    clustering_distance_rows = "euclidean",
    clustering_distance_cols = "euclidean",

    fontsize = 16,
    fontsize_row = 20,
    fontsize_col = 20,
    fontsize_number = 8,

    cluster_rows = TRUE,
    show_rownames = TRUE,
    show_colnames = TRUE,
    border_color = "grey60",
    cellwidth = 24,
    cellheight = 24,
    width = 16,
    height = 16,
    filename = "EucDist_PNNL.png"
)
```

Parameter `annot_cols` defines the tracks to be displayed on the top of distrance-matrix plots. In this example, we have choosen `expt_smry.xlsx::Shape` and `expt_smry.xlsx::Alpha`, which encodes the WHIM subtypes and the batch numbers, respectively. Parameter `annot_colnames` allows us to rename the tracks for better intuition.

Correlation plots

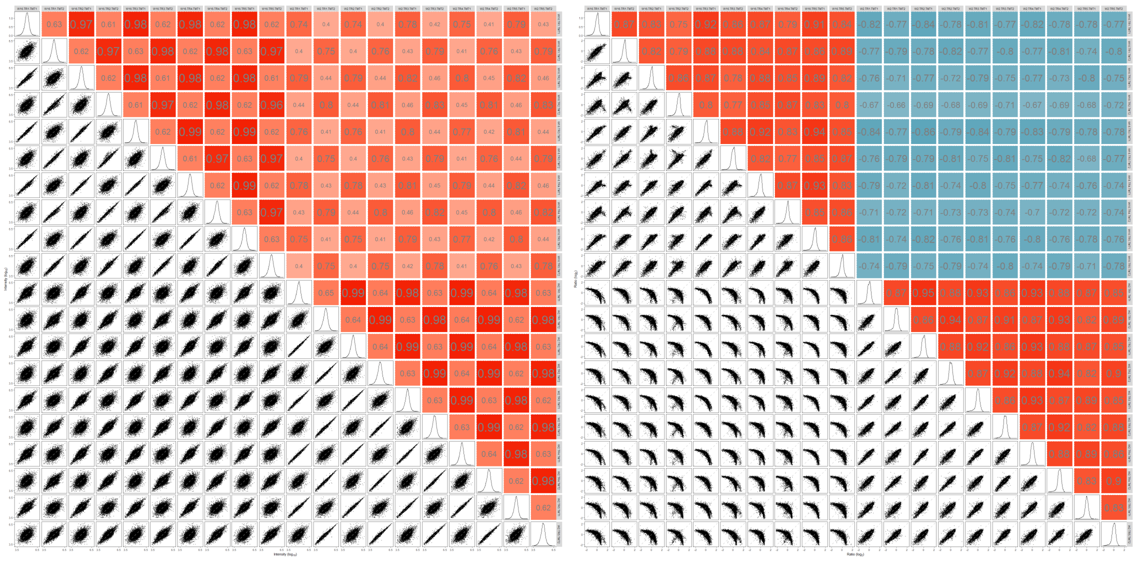
In this section, we compare the correlation between W2 and W16.

The documentation from this point on is under construction; nevertheless, interactive R scripts are made available for now.

Correlations of both intensity and log2FC will be performed.

```r
# Correlation plots of peptide data
pepCorr(
    use_log10 = TRUE,
    scale_log2r = TRUE,
    min_int = 3.5,
    max_int = 6.5,
    min_log2r = -2,
    max_log2r = 2,
    width = 24,
    height = 24
)
```

```r
# Correlation plots of protein data
prnCorr(
    use_log10 = TRUE,
    scale_log2r = TRUE,
    min_int = 3.5,
    max_int = 6.5,
    min_log2r = -2,
    max_log2r = 2,
    width = 24,
    height = 24
)
```



The following shows an example of Euclidean distance matrix against peptide data:

```r
# Euclidean distance plots of peptide data
pepEucDist(
    scale_log2r = TRUE,
    adjEucDist = FALSE,
    show_ids = TRUE,
    annot_cols = c("Peptide_Yield", "Group"),

    display_numbers = TRUE,
    number_color = "grey30",
    number_format = "%.2f",

    clustering_distance_rows = "euclidean",
    clustering_distance_cols = "euclidean",

    fontsize = 16,
    fontsize_row = 20,
    fontsize_col = 20,
    fontsize_number = 8,

    cluster_rows = TRUE,
    show_rownames = TRUE,
    show_colnames = TRUE,
    border_color = "grey60",
    cellwidth = 24,
    cellheight = 24
)
```

The following performs of heat map visualization against protein data:

```
# Protein heat maps
prnHM(
    scale_log2r = TRUE,
    xmin = -.5,
    xmax = .5,
    x_margin = 0.1,
    annot_cols = c("Peptide_Yield", "Group"),
    cluster_rows = TRUE,
    cutree_rows = 6,
    show_rownames = FALSE,
    show_colnames = TRUE,
    fontsize_row = 3,
    cellwidth = 14,
    width = 24,
    height = 12
)
```

The following performs the imputation of peptide and protein data:

```
# Impute missing values
pepImp(m = 5, maxit = 5)
prnImp(m = 5, maxit = 5)
```

The following performs the trend analysis against protein expressions:

```
# Soft clustering in protein expressions by trends
prnTrend(n_clust = 6, scale_log2r = TRUE)
```

The following performs the NMF analysis against protein data:

```
# Protein NMF
library(NMF)
prnNMF(r = 6, xmin = -1, xmax = 1, x_margin = 0.1,
    annot_cols = c("Peptide_Yield", "TMT_Set", "Group"),
    scale_log2r = TRUE)
```

The following performs the significance analysis of peptide and protein data:

```
# Peptide significance tests
# Multiple formulas are allowed
pepSig(
    scale_log2r = TRUE,
    limma_1 = ~ Term["W2.TMT2-W2.TMT1", "W16.TMT2-W16.TMT1", "(W16.TMT2-W16.TMT1)-(W2.TMT2-W2.TMT1)"]
)
```

```r
# Protein significance tests
prnSig(
    scale_log2r = TRUE,
    limma_1 = ~ Term["W2.TMT2-W2.TMT1", "W16.TMT2-W16.TMT1", "(W16.TMT2-W16.TMT1)-(W2.TMT2-W2.TMT1)"]
)
```

The following performs the volcano plot visualization of peptide and protein data:

```r
# Peptide volcano plots
pepVol(scale_log2r = TRUE)

# Protein volcano plots
prnVol(scale_log2r = TRUE)
```

The following performs GSVA:

```r
prnGSVA(gset_nm = c("go_sets", "kegg_sets", "c2_msig"), scale_log2r = TRUE)
```

The following maps gene sets under the environment of volcano plot visualization:

```r
gsvaMap(scale_log2r = TRUE, pval_cutoff = 1E-2, show_sig = "pVal")
```

Philipp, Martins. 2018. "Reproducible Workflow for Multiplexed Deep-Scale Proteome and Phosphoproteome Analysis of Tumor Tissues by Liquid Chromatography-Mass Spectrometry." *Nature Protocols* 13 (7): 1632–61. https://doi.org/10.1038/s41596-018-0006-9.