Statistical analysis of proteomics experiments with R and MSstats

Meena Choi & Erik Verschueren

May 21, 2015

Prerequisites

We're setting the working directory to where you saved files for Day 4.

```
setwd('~/Dropbox/neucourse_may2015/Day4-Thursday-1.5hr/')
getwd()
```

```
## [1] "/Users/everschueren/Dropbox/neucourse_may2015/Day4-Thursday-1.5hr"
```

If you didn't have MSstats installed so far. Please install it now.

```
install.packages(pkgs = 'MSstats.daily_3.0.4.tar.gz', repos = NULL, type = 'source')
```

Load MSstats and verify that you have the correct version (3.0.4) loaded.

```
library('MSstats.daily', warn.conflicts = F, quietly = T, verbose = F)
?MSstats.daily
```

SRM analysis with MSstats

1. preparing the data for MSstats input

Let's start by reading in data as it comes out of Skyline.

```
RatPlasmaData <- read.csv('../Day1-Monday-1hr/RatPlasmaMSstatsInput.csv')
head(RatPlasmaData)
```

```
##   ProteinName PeptideSequence PrecursorCharge FragmentIon ProductCharge
## 1   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 2   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 3   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 4   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 5   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 6   NP_036629 CSLPRPWALTFSYGR               2         y10             1
##   IsotopeLabelType Condition BioReplicate       FileName  Area
## 1            light  Diseased          102 D_102_REP1.raw 14516
## 2            light  Diseased          102 D_102_REP2.raw  9607
## 3            light  Diseased          102 D_102_REP3.raw  7480
## 4            light  Diseased          103 D_103_REP1.raw  5692
## 5            light  Diseased          103 D_103_REP2.raw  5953
## 6            light  Diseased          103 D_103_REP3.raw   646
##   StandardType
## 1
## 2
## 3
## 4
## 5
## 6
```

```
?SRMRawData
```

The raw data (input data for MSstats) is required to contain variable of ProteinName, PeptideSequence, PrecursorCharge, FragmentIon, ProductCharge, IsotopeLabelType, Condition, BioReplicate, Run, Intensity. The variable names should be fixed. Now adapt the column scheme of the dataset so that it fits MSstats input format. We're changing `FileName` to `Run` and `Area` to `Intensity`

```
colnames(RatPlasmaData)[9] <- 'Run'
colnames(RatPlasmaData)[10] <- 'Intensity'
head(RatPlasmaData)
```

```
##   ProteinName PeptideSequence PrecursorCharge FragmentIon ProductCharge
## 1   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 2   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 3   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 4   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 5   NP_036629 CSLPRPWALTFSYGR               2         y10             1
## 6   NP_036629 CSLPRPWALTFSYGR               2         y10             1
##   IsotopeLabelType Condition BioReplicate            Run Intensity
## 1            light  Diseased          102 D_102_REP1.raw     14516
## 2            light  Diseased          102 D_102_REP2.raw      9607
## 3            light  Diseased          102 D_102_REP3.raw      7480
## 4            light  Diseased          103 D_103_REP1.raw      5692
## 5            light  Diseased          103 D_103_REP2.raw      5953
## 6            light  Diseased          103 D_103_REP3.raw       646
##   StandardType
## 1
## 2
## 3
## 4
## 5
## 6
```

---

2. Summarizing the data

2.1 Normalizing and summarizing data with dataProcess

```
?dataProcess
```

2.1.1 Default normalization and summarization options ! Always pay attention to the default options

The default option for summarization is `linear`. The default option for normalization is `equalizeMedians`. However, if you have a spiked in standard then you want to set this to `globalStandards` and define the standard with `nameStandards`

```
# `linear` is the default model-based summarization option
Rats.linear <- dataProcess(raw = RatPlasmaData, summaryMethod = 'linear',
                           normalization = 'equalizeMedians',
                           fillIncompleteRows = TRUE, skylineReport = TRUE)
```

```
names(Rats.linear)
```

```
## [1] "ProcessedData" "RunlevelData"  "SummaryMethod" "ModelQC"
```

```
head(Rats.linear$ProcessedData)
```

```
##            PROTEIN         PEPTIDE TRANSITION              FEATURE LABEL
## 23899 NP_001007697 CSSLLWAGAAWLR_2       y3_1 CSSLLWAGAAWLR_2_y3_1     L
## 23857 NP_001007697 CSSLLWAGAAWLR_2       y4_1 CSSLLWAGAAWLR_2_y4_1     L
## 23815 NP_001007697 CSSLLWAGAAWLR_2       y5_1 CSSLLWAGAAWLR_2_y5_1     L
## 23773 NP_001007697 CSSLLWAGAAWLR_2       y6_1 CSSLLWAGAAWLR_2_y6_1     L
## 23731 NP_001007697 CSSLLWAGAAWLR_2       y7_1 CSSLLWAGAAWLR_2_y7_1     L
## 23689 NP_001007697 CSSLLWAGAAWLR_2       y8_1 CSSLLWAGAAWLR_2_y8_1     L
##       GROUP_ORIGINAL SUBJECT_ORIGINAL RUN GROUP SUBJECT SUBJECT_NESTED
## 23899       Diseased              102   1     1       1            1.1
## 23857       Diseased              102   1     1       1            1.1
## 23815       Diseased              102   1     1       1            1.1
## 23773       Diseased              102   1     1       1            1.1
## 23731       Diseased              102   1     1       1            1.1
## 23689       Diseased              102   1     1       1            1.1
##       INTENSITY  ABUNDANCE METHOD
## 23899        24  4.0622458      1
## 23857       182  6.9850780      1
## 23815       782  9.0883081      1
## 23773      1580 10.1029922      1
## 23731         1 -0.5227167      1
## 23689         2  0.4772833      1
```

```
head(Rats.linear$RunlevelData)
```

```
##   RUN      Protein LogIntensities NumFeature NumPeaks GROUP GROUP_ORIGINAL
## 1   1 NP_001007697       10.55270         12       12     1       Diseased
## 2   1    NP_062212       11.76530         19       19     1       Diseased
## 3   1 NP_001008724       17.27211         28       28     1       Diseased
## 4   1    NP_062242       18.10018         18       18     1       Diseased
## 5   1 NP_001010968       14.02051         15       15     1       Diseased
## 6   1 NP_001011908       12.43851         23       23     1       Diseased
##   SUBJECT_ORIGINAL SUBJECT_NESTED SUBJECT
## 1              102            1.1       1
## 2              102            1.1       1
## 3              102          1.1       1
## 4              102          1.1       1
## 5              102          1.1       1
## 6              102          1.1       1
```

```
head(Rats.linear$ModelQC)
```

```
##            PROTEIN         PEPTIDE TRANSITION              FEATURE LABEL
## 23899 NP_001007697 CSSLLWAGAAWLR_2       y3_1 CSSLLWAGAAWLR_2_y3_1     L
## 23857 NP_001007697 CSSLLWAGAAWLR_2       y4_1 CSSLLWAGAAWLR_2_y4_1     L
## 23815 NP_001007697 CSSLLWAGAAWLR_2       y5_1 CSSLLWAGAAWLR_2_y5_1     L
## 23773 NP_001007697 CSSLLWAGAAWLR_2       y6_1 CSSLLWAGAAWLR_2_y6_1     L
## 23731 NP_001007697 CSSLLWAGAAWLR_2       y7_1 CSSLLWAGAAWLR_2_y7_1     L
## 23689 NP_001007697 CSSLLWAGAAWLR_2       y8_1 CSSLLWAGAAWLR_2_y8_1     L
##       GROUP_ORIGINAL SUBJECT_ORIGINAL RUN GROUP SUBJECT SUBJECT_NESTED
## 23899       Diseased              102   1     1       1            1.1
## 23857       Diseased              102   1     1       1            1.1
## 23815       Diseased              102   1     1       1            1.1
## 23773       Diseased              102   1     1       1            1.1
## 23731       Diseased              102   1     1       1            1.1
## 23689       Diseased              102   1     1       1            1.1
##       INTENSITY  ABUNDANCE METHOD  residuals   fitted
## 23899        24  4.0622458      1 -2.5061889 6.568435
## 23857       182  6.9850780      1 -1.7917613 8.776839
## 23815       782  9.0883081      1 -0.3029833 9.391291
## 23773      1580 10.1029922      1  0.2110118 9.891980
## 23731         1 -0.5227167      1 -6.4485170 5.925800
## 23689         2  0.4772833      1 -3.8701028 4.347386
```

```
head(Rats.linear$SummaryMethod)
```

```
## [1] "linear"
```

2.1.2 Different summarization options Besides summarizing observations with linear models MSstats also offers a summarization option using Tukey-Median-Polish (TMP), which is more robust, and as sum of log-intensities, which is the default Skyline behaviour.

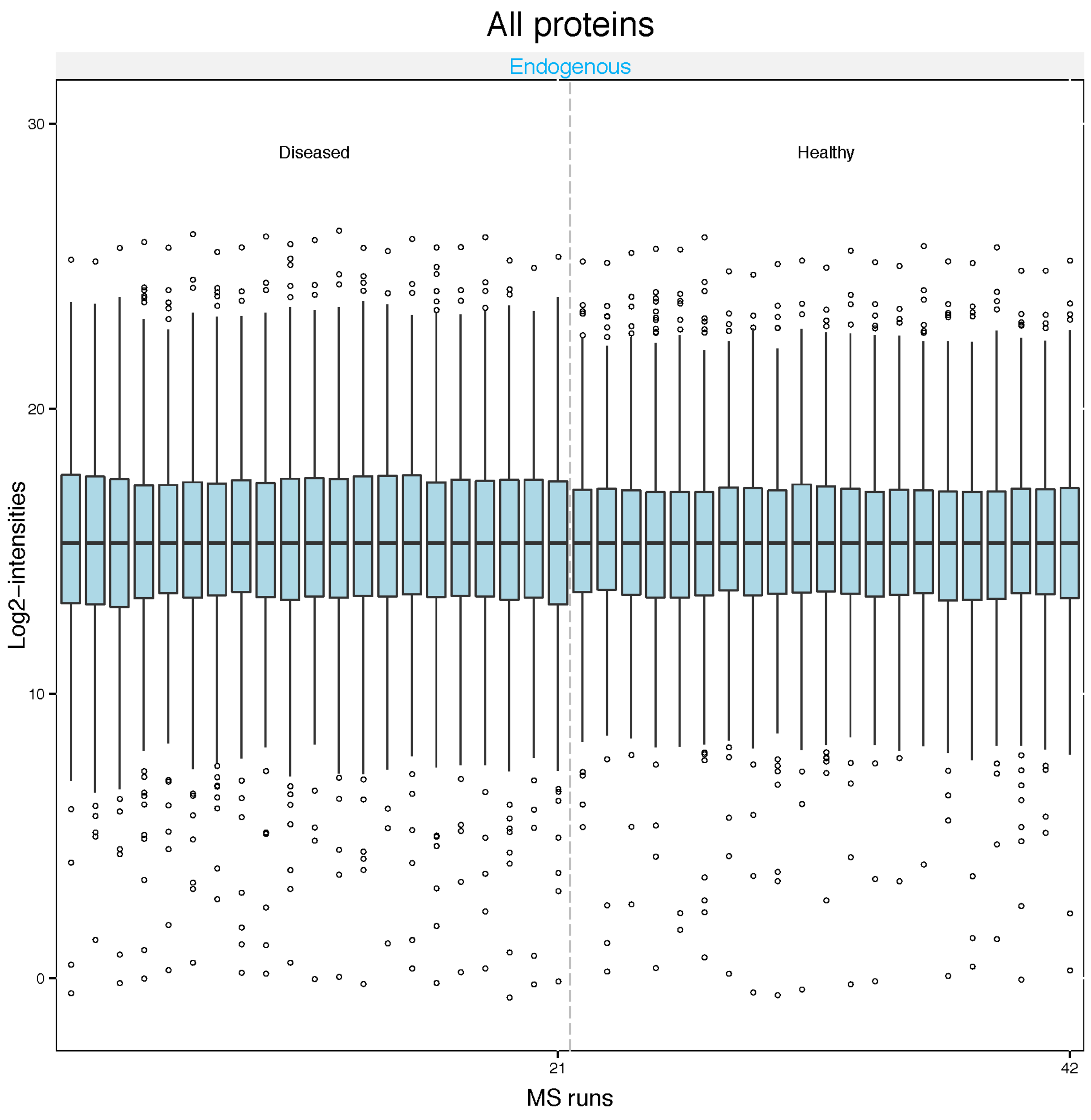
```
# Median comparsion with Tukey-Median-Polish
Rats.TMP <- dataProcess(raw = RatPlasmaData, summaryMethod="TMP",
                        skylineReport=TRUE)
# different method same output..
names(Rats.TMP) == names(Rats.linear)
# but since this is not model-based, no model summary
Rats.TMP$ModelQC
# Different summarization optinons : log-sum-intensities
Rats.logOfSum <- dataProcess(raw = RatPlasmaData, summaryMethod="logOfSum",
                             skylineReport = TRUE)
```

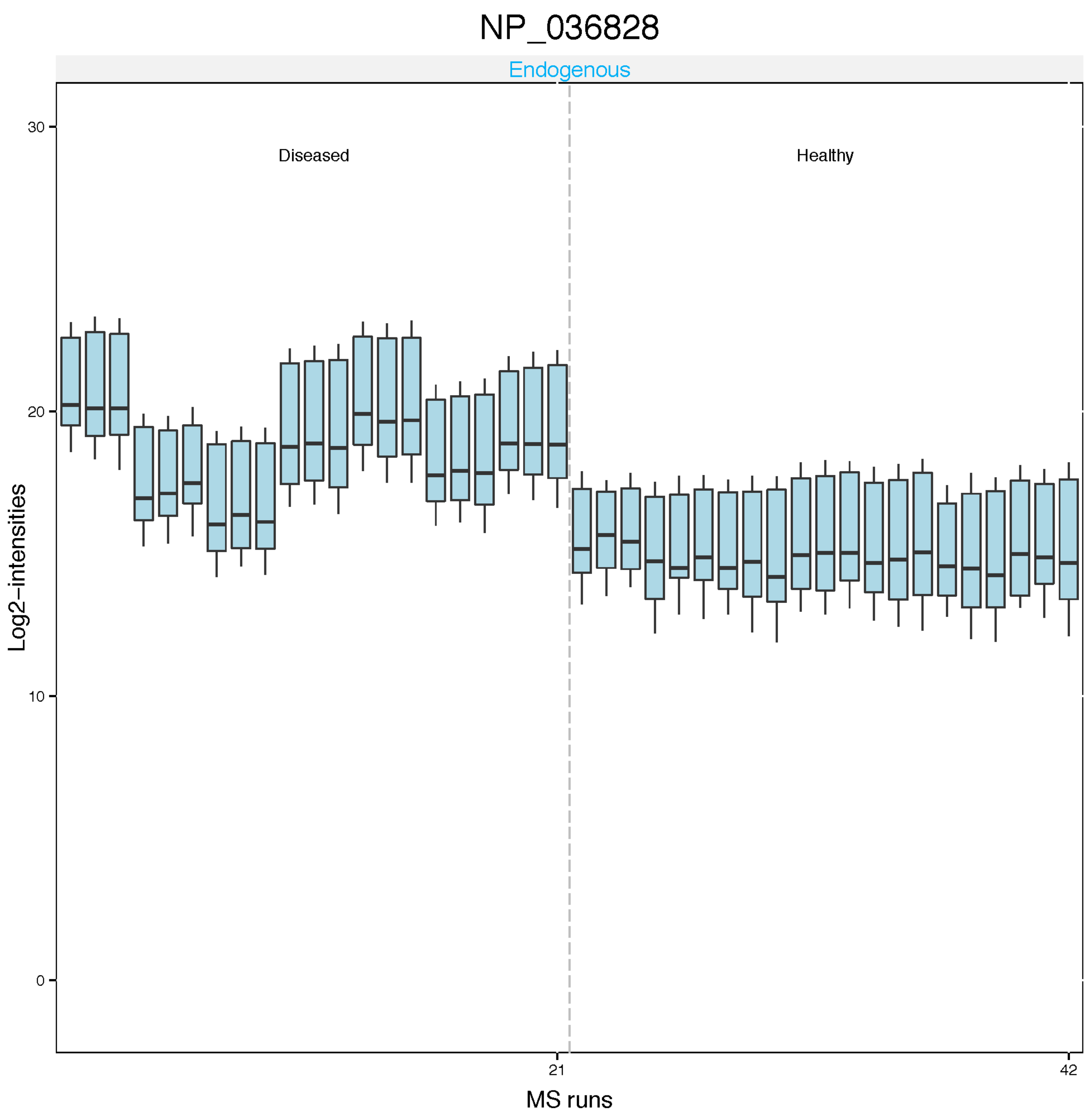
2.2 Visualization of processed data

2.2.1 Quality control and Normalization effects Now let's look at what the equalize medians procedure did to our data. We can generate these for all proteins but also for single Proteins at a time if we have a big dataset. Let's look at the 'Kininogen-1' (Kng1/NP_036828) protein in these example data. Kng1 is upregulated as part of an anti-inflammatory response: it increases vascular permeability and doing so induces hypotension. It is postulated that Kng1 has a cardioprotective effect. As you can appreciate it is upregulated, albeit with soem variability, in many diseased Rats.

```
dataProcessPlots(data = Rats.linear, type="QCplot")
```

```
MYPROTEIN = "NP_036828"
dataProcessPlots(data = Rats.linear, type="QCplot",
                 which.Protein=MYPROTEIN, address="KNG1_eqmedians_")
```

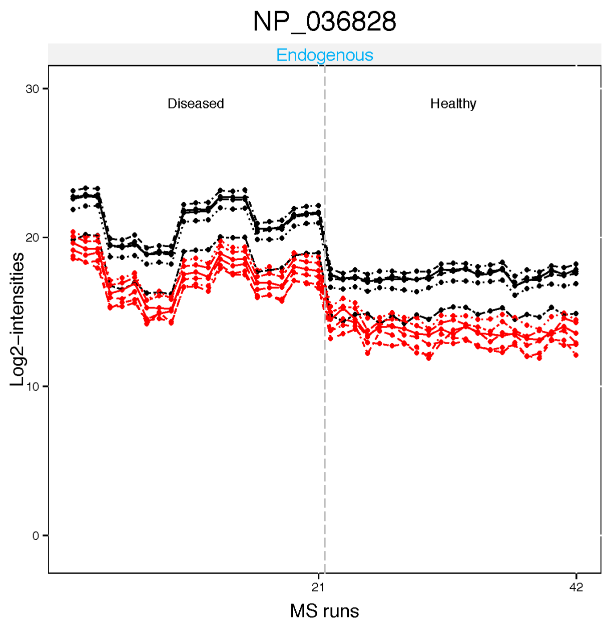


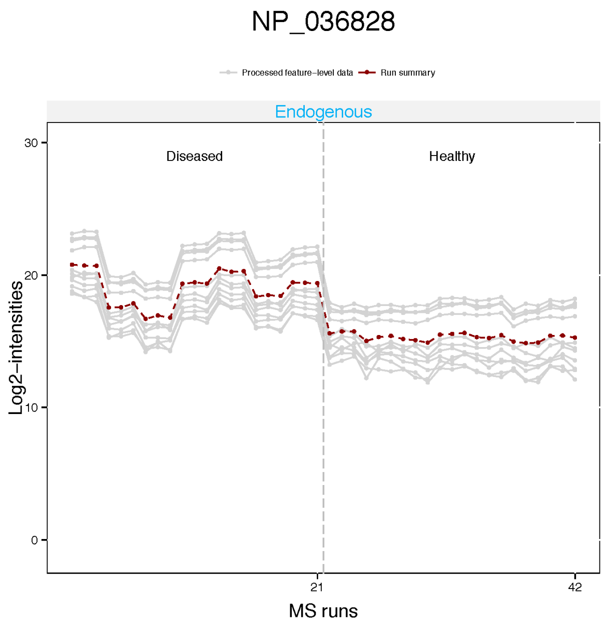


2.2.2 Summarization effects

Profile plots First, let's look at how the linear model summarized the data per protein. The panel left shows each peptide transition across runs, grouped per condition. Ech peptide has a different colour/type layout. The panel on the right shows the same transitions in grey, with the values as summarized by the model overlayed in red.

```
dataProcessPlots(data = Rats.linear, address="Rats_linear_",
                 type="Profileplot", featureName="NA", width=14, height=7)
```

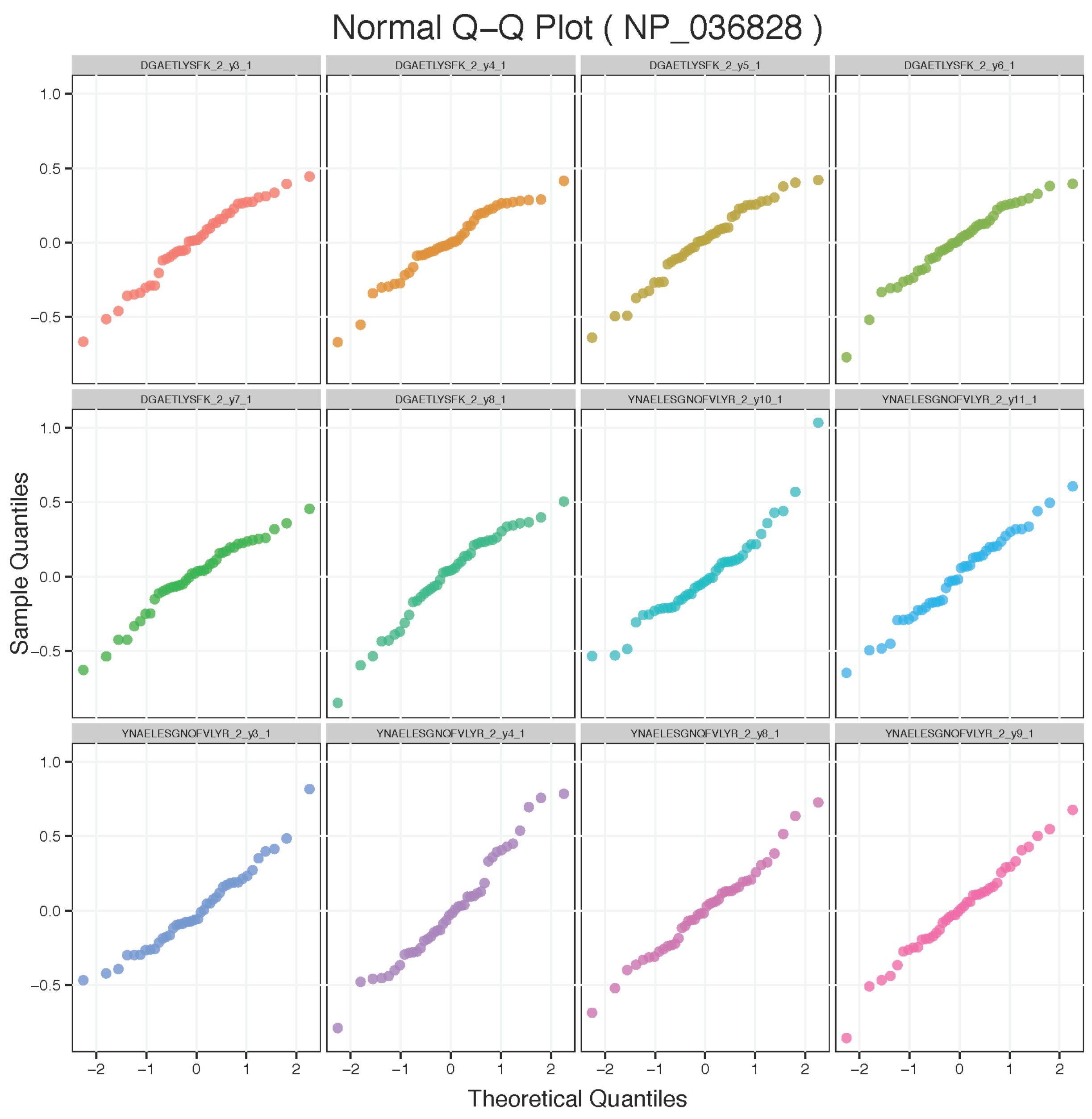




Model based quality plots : quantile-quantile plots Let's inspect how well ***each*** transition is represented by the model. A transition that is modeled well will show a linear relation between the sampled points and the theoretical values (from the model) for every quantile in the quantile-quantile plot.

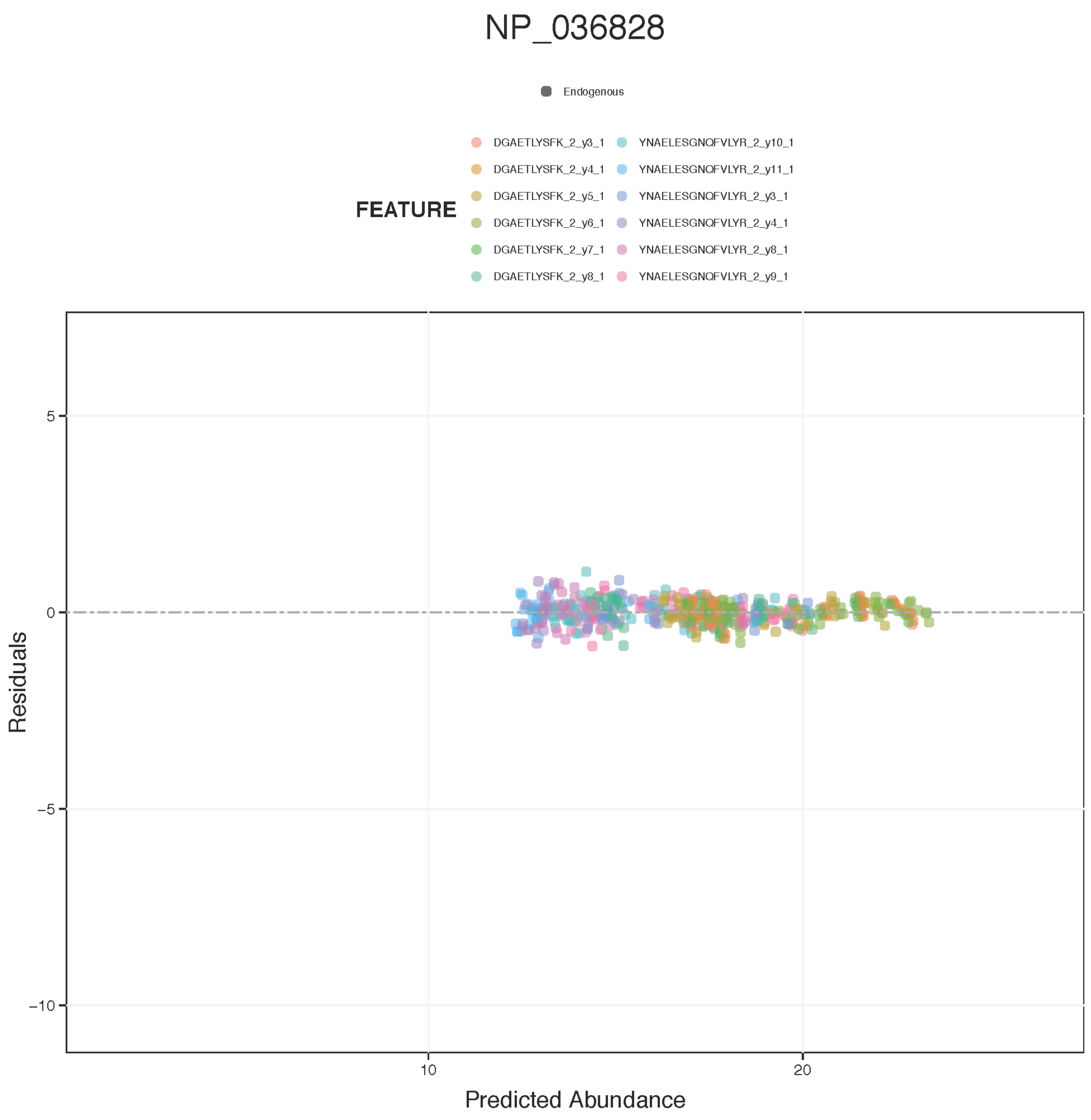
Exercise Inspect the PDF file that contains the quantile-quantile plot. What do you see?

```
modelBasedQCPlots(data = Rats.linear$ModelQC, type="QQPlots", feature.QQPlot="byFeature",
                  address="Rats_linear_")
```



Model based quality plots : residual plots A residual plot is a graph that shows the residuals (or errors between the sampled data point and the model's prediction) on the vertical axis and the independent variable (in this case each prediction) on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, our linear model is appropriate for the data.

```
modelBasedQCPlots(data = Rats.linear$ModelQC, type="ResidualPlots", address="Rats_linear_")
```



---

3. Finding differentially abundant proteins across conditions

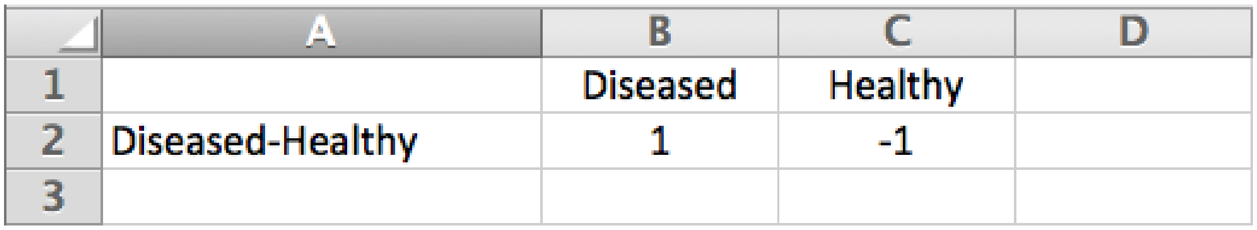
3.1 Comparing conditions with groupComparison

After we normalized the data and summarized each protein's behaviour across conditions with one of the dataProcess summarization methods, we are all set to compare protein changes between groups of conditions. Within MSstats we can do this with the `groupComparison` function, which takes as input the output of the `dataProcess` function.

```
?groupComparison
```

Of course we have to tell `groupComparison` which are the conditions we would like to compare…
You can make your `contrast.matrix` in R in a text editor or even in Excel, if you prefer. We define our contrast matrix by adding a column for every condition, **in alphabetical order**. We add a row for every comparison we would like to make between groups of conditions.

0 is for conditions we would like to ignore. **1** is for conditions we would like to put in the numerator of the ratio or fold-change. **-1** is for conditions we would like to put in the denumerator of the ratio or fold-change.



	A	B	C	D
1		Diseased	Healthy	
2	Diseased-Healthy	1	-1	
3				

NOTE If you make a new contrast matrix save it as a text file, where you saved your script and datasets to read it into R. Make sure to remove the first tab in the matrix header of the text file (before your first condition)!

QUESTION How would you compare a group of conditions versus another group of conditions?

```
Rats.contrasts <- read.delim(file = 'RatPlasmaData-contrasts.txt', sep='\t')
Rats.contrasts <- as.matrix(Rats.contrasts)
```

We're ready to go! let's compare our two populations.

```
Rats.comparisons <- groupComparison(contrast.matrix = Rats.contrasts, data=Rats.linear)
```

Let's inspect the results to see what proteins are changing significantly between Diseased and Healthy.

```
names(Rats.comparisons)
```

```
## [1] "ComparisonResult" "ModelQC"          "fittedmodel"
```

```
names(Rats.comparisons$ComparisonResult)
```

```
## [1] "Protein"    "Label"      "log2FC"     "SE"         "Tvalue"
## [6] "DF"         "pvalue"     "adj.pvalue"
```

```
SignificantProteins =
  Rats.comparisons$ComparisonResult[Rats.comparisons$ComparisonResult$adj.pvalue < 0.05 ,]
nrow(SignificantProteins)
```

```
## [1] 38
```

```
SignificantProteinsUpInDiseased = SignificantProteins[SignificantProteins$log2FC > 2 ,]
nrow(SignificantProteinsUpInDiseased)
```

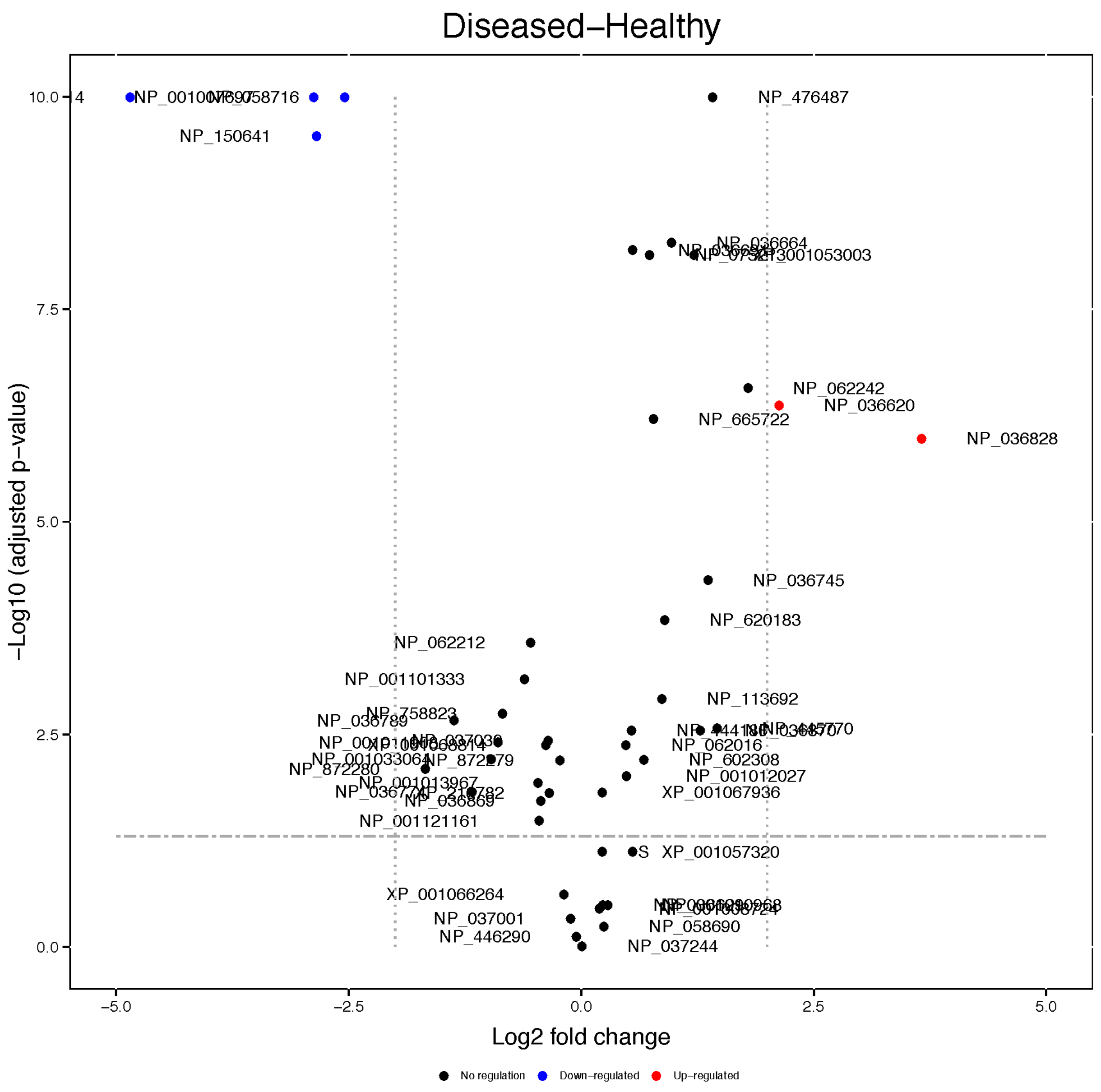
```
## [1] 2
```

3.2 Visualization of differentially abundant proteins

```
?groupComparisonPlots
```

Volcano plots allow us to visually separate strong changes, which are not significant, from strong and significant changes. Look for these subjects in the upper right and upper left quadrants of the plot.

```
groupComparisonPlots(data = Rats.comparisons$ComparisonResult, type = 'VolcanoPlot',
                     sig = 0.05, FCcutoff = 2^2, address = 'Rats_linear_significant_')
```



4. save your models

```
save.image(file = 'RatPlasmaData_MSstats_models.RData')
```

EXTRAS FOR THE ADVANCED USER!

Dealing with missing values

Usually your SRM data should have very little to no missing values.. However, label-free DDA datasets have many missing values. MSstats supports a number of ways to deal with this.
Now, let's revisit how we processed the data with `dataProcess` and fill in missing values with the minimal value per run.

```
Rats.missing.minRun <- dataProcess(raw = RatPlasmaData, cutoffCensored="minFeature",
                                   censoredInt="0", skylineReport=TRUE)
```

Or alternatively, we can censor missing values in the model.

```
Rats.missing.censored <- dataProcess(raw = RatPlasmaData, censoredInt="0",
                                     summaryMethod="linear", normalization=FALSE,
                                     fillIncompleteRows=TRUE, skylineReport=TRUE)
```

Have a look at the profile plots to compare the two missing values options..

```
dataProcessPlots(data = Rats.missing.minRun, address="Rats_missing_minrun_",
                 type="Profileplot", featureName="NA", width=14, height=7)
dataProcessPlots(data = Rats.missing.censored, address="Rats_missing_censored_",
                 type="Profileplot", featureName="NA", width=14, height=7)
```

Planning future experimental designs

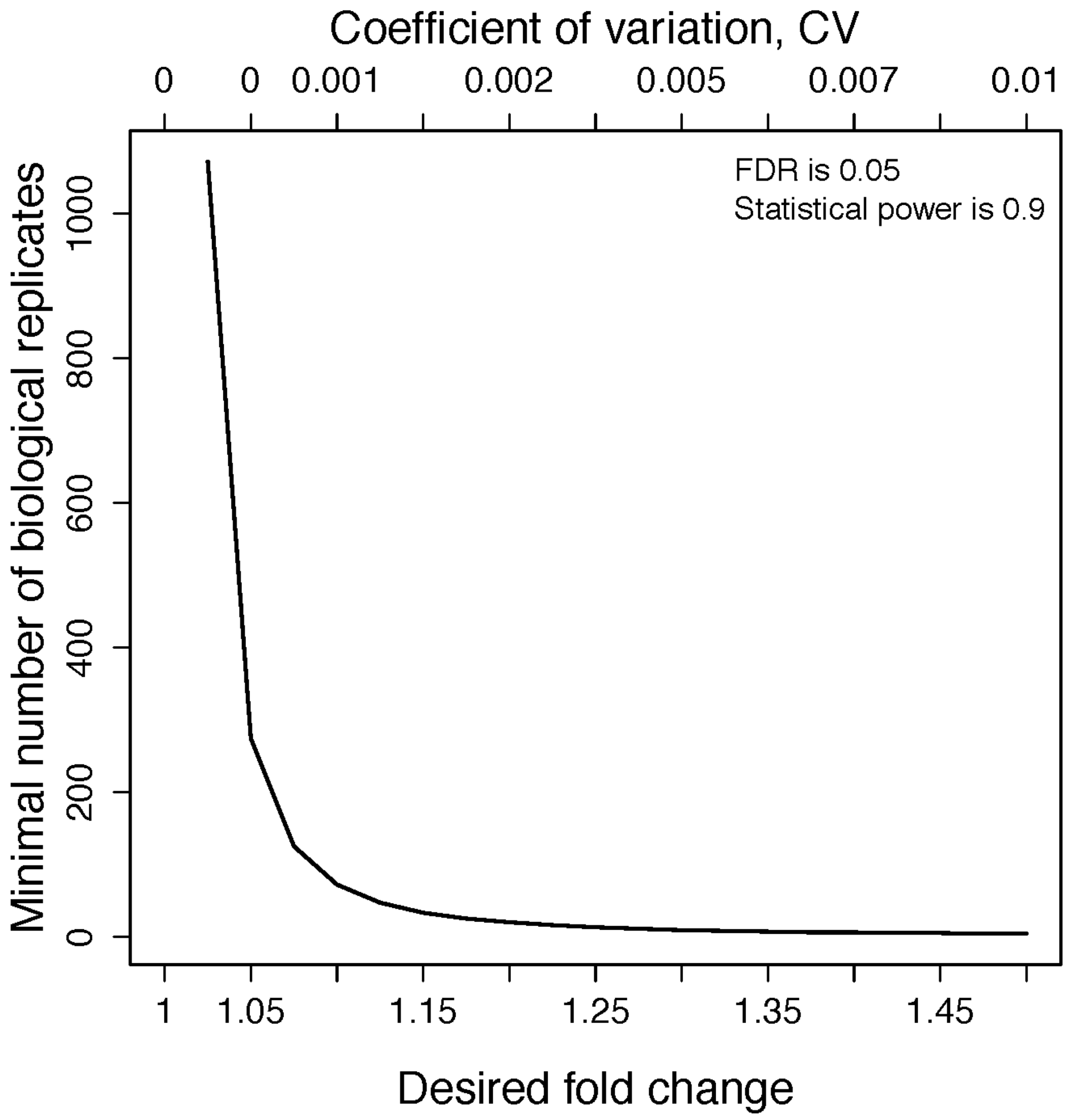
```
?designSampleSize
```

```
Rats.expdesign <- designSampleSize(data = Rats.comparisons$fittedmodel,
                                   desiredFC = c(1,1.5), FDR = 0.05,
                                   numSample = TRUE)
```

Designing sample size for desired fold-change or statistical power

```
pdf(file='Rats_expdesign.pdf', width=7,height=7)
  designSampleSizePlots(data = Rats.expdesign)
dev.off()
```

Visualizing the relationship between desired fold-change and mininum sample size number


Coefficient of variation, CV
FDR is 0.05
Statistical power is 0.9
Minimal number of biological replicates
Desired fold change