Basics of Statistical Inference and Some Experimental Design Concepts

Lieven Clement

Outline

- Francisella tularensis Example
- Hypothesis testing
- Multiple testing
- Moderated statistics
- Experimental design

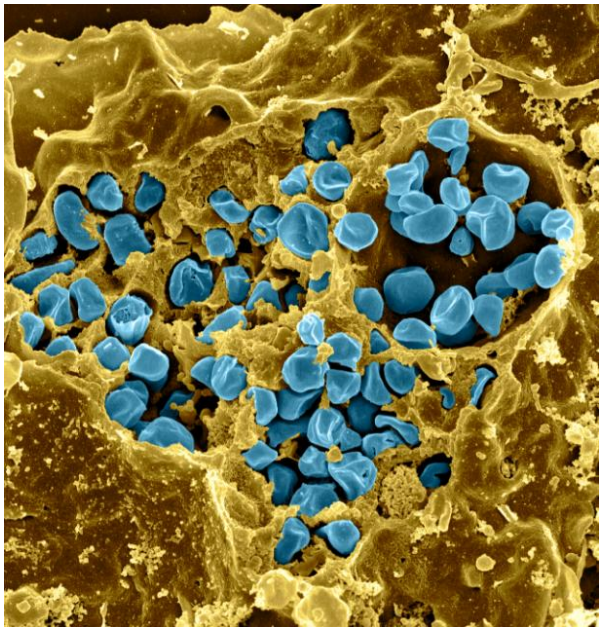
Note, that this dataset has been acquired with DDA (data dependent acquisition). It was searched with MaxQuant. The analysis starts from the peptides.txt file, which is a table in the output directory of MaxQuant with the data summarized at the peptide-level.

The preprocessing steps are slightly different. The modeling concepts, however, are also applicable to DIA data.

Movie clip

Francisella tularensis experiment

Movie clip



- Pathogen: causes tularemia
- Metabolic adaptation key for intracellular life cycle of pathogenic microorganisms.
- Upon entry into host cells quick phagosomal escape and active multiplication in cytosolic compartment.
- Francisella is auxotrophic for several amino acids, including arginine.
- Inactivation of arginine transporter delayed bacterial phagosomal escape and intracellular multiplication.
- Experiment to assess difference in proteome using 3 WT vs 3 ArgP KO mutants

Import the data in R

Click to see code

1. Load libraries

```
library(tidyverse)
library(limma)
library(QFeatures)
library(msqrob2)
library(plotly)
library(ggplot2)
```

2. We use a peptides.txt file from MS-data quantified with maxquant that contains MS1 intensities summarized at the peptide level.

```
peptidesFile <- "https://raw.githubusercontent.com/statOmics/PDA22GTPB/data/quantification/f
```

3. Maxquant stores the intensity data for the different samples in columns that start with Intensity. We can retrieve the column names with the intensity data with the code below:

```
ecols <- grep("Intensity\\.", names(read.delim(peptidesFile)))
```

4. Read the data and store it in QFeatures object

```
qf <- readQFeatures(
  assayData = read.delim(peptidesFile),
  fnames = 1,
  quantCols =  ecols,
  name = "peptideRaw")
```

5. Update data with information on design

```
colData(qf)$genotype <- qf[[1]] |>
  colnames() |>
  substr(12,13) |>
  as.factor() |>
  relevel("WT")
colData(qf)
```

```
DataFrame with 6 rows and 1 column
                         genotype
                         <factor>
Intensity.1WT_20_2h_n3_3       WT
Intensity.1WT_20_2h_n4_3       WT
Intensity.1WT_20_2h_n5_3       WT
Intensity.3D8_20_2h_n3_3       D8
Intensity.3D8_20_2h_n4_3       D8
Intensity.3D8_20_2h_n5_3       D8
```

Preprocessing

Click to see code to log-transfrom the data

1. Log transform

- peptides with zero intensities are missing peptides and should be represented with a `NA` value rather than `0`.

```
qf <- zeroIsNA(qf, "peptideRaw") # convert 0 to NA
```

- Logtransform data with base 2

```
qf <- logTransform(qf, base = 2, i = "peptideRaw", name = "peptides_log")
```

2. Filtering

Only use proteotypic proteins

```
qf <- filterFeatures(qf, ~ !grepl(pattern = ";", Proteins))
```

- Remove reverse sequences (decoys) and contaminants. Note that this is indicated by the column names Reverse and depending on the version of maxQuant with Potential.contaminants or Contaminants.

```
qf <- filterFeatures(qf,~Reverse != "+")
qf <- filterFeatures(qf,~ Contaminant != "+")
```

- We keep peptides that were observed at least 4 times out of the $n = 6$ samples, so that we can estimate the peptide characteristics.

We tolerate the following proportion of NAs: $\text{pNA} = \frac{(n-4)}{n} = 0.33$, so we keep peptides that are observed in at least 66.7% of the samples. This is an arbitrary value that may need to be adjusted depending on the experiment and the data set.

```
nObs <- 4
n <- ncol(qf[["peptides_log"]])

(qf <- filterNA(qf, i = "peptides_log", pNA = (n - nObs) / n))
```

```
An instance of class QFeatures (type: bulk) with 2 sets:

 [1] peptideRaw: SummarizedExperiment with 10461 rows and 6 columns
 [2] peptides_log: SummarizedExperiment with 5052 rows and 6 columns
```

We keep 5052 peptides upon filtering.

3. Normalization by median centering

```
qf <-  sweep( #4. Subtract log2 norm factor column-by-column (MARGIN = 2)
  qf,
  MARGIN = 2,
  STATS = nfLogMedianOfRatios(qf, i="peptides_log"), #1.
  i = "peptides_log",
  name = "peptides_norm"
)
```

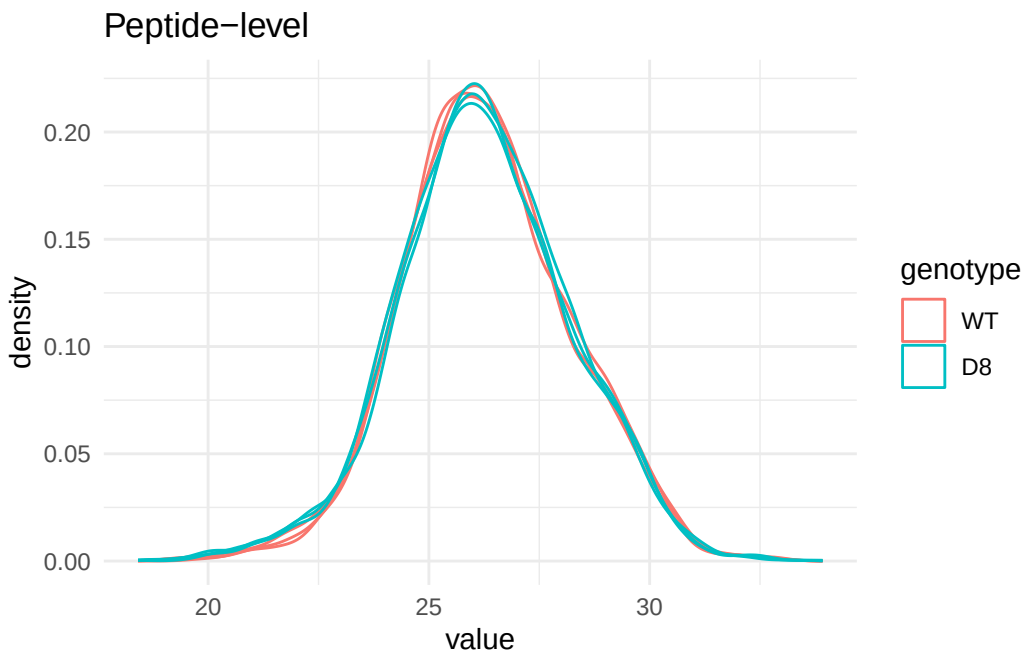
4. Summarization. We use maxLFQ from the iq package.

```
qf <- aggregateFeatures(qf,
    i = "peptides_norm",
    fcol = "Proteins",
    name = "proteins",
    fun = function(X) iq::maxLFQ(X)$estimate
)
```

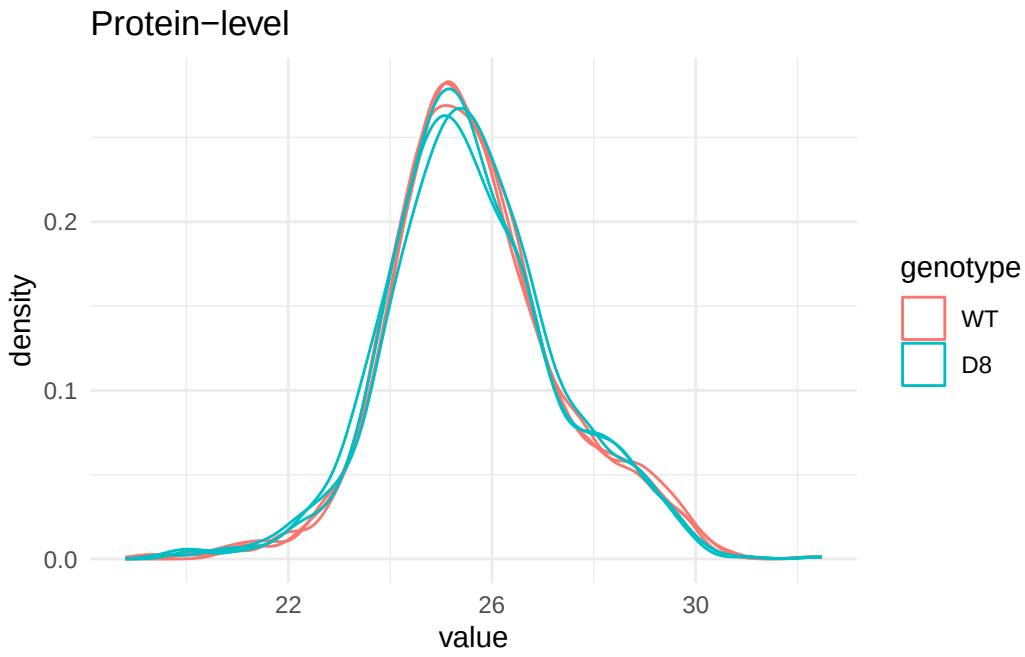
```
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

Plot of preprocessed data

```
qf[, , "peptides_norm"] |> #1.
  longForm(colvars = c( "genotype")) |>  #2.
  data.frame() |>
  filter(!is.na(value)) |>
  ggplot() + #3.
  aes(x = value,
      colour = genotype,
      group = colname) +
  geom_density() +
  theme_minimal() +
  ggtitle("Peptide-level")
```




```
qf[, , "proteins"] |> #1.
  longForm(colvars = c( "genotype")) |>  #2.
  data.frame() |>
  filter(!is.na(value)) |>
  ggplot() + #3.
  aes(x = value,
      colour = genotype,
      group = colname) +
  geom_density() +
  theme_minimal() +
    ggtitle("Protein-level")
```



Summarized data structure

Design

```
qf |>
  colData() |>
  knitr::kable()
```

	genotype
Intensity.1WT_20_2h_n3_3	WT
Intensity.1WT_20_2h_n4_3	WT
Intensity.1WT_20_2h_n5_3	WT
Intensity.3D8_20_2h_n3_3	D8
Intensity.3D8_20_2h_n4_3	D8
Intensity.3D8_20_2h_n5_3	D8

- WT vs KO
- 3 vs 3 repeats

Summarized intensity matrix

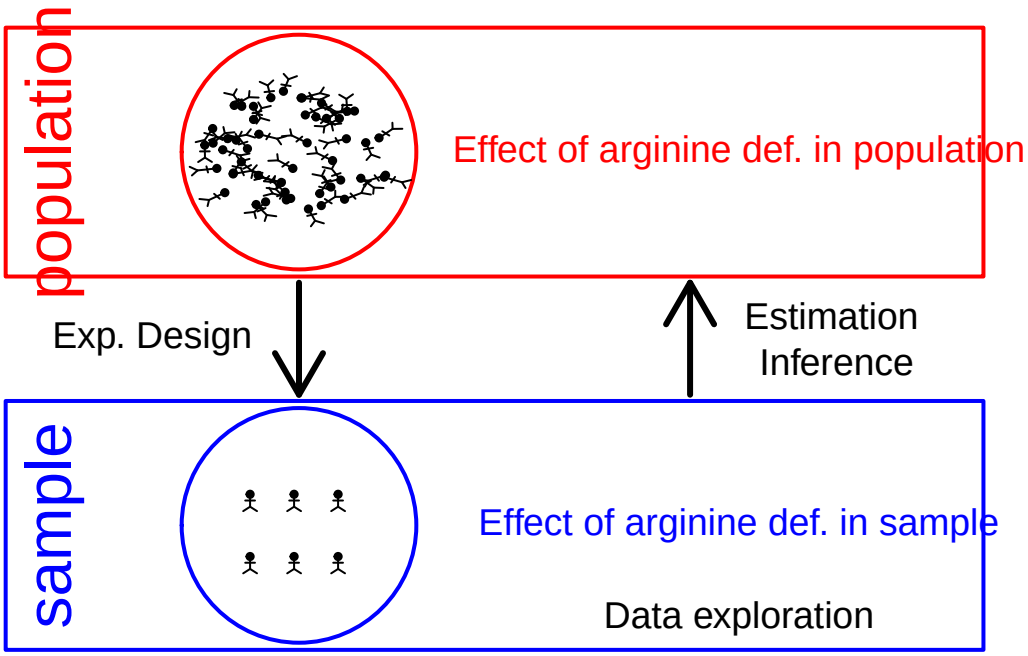
```
qf[["proteins"]] |>
  assay() |>
  head() |>
  knitr::kable()
```

	Intensity.1WT_20_2h_n5_1	Intensity.1WT_20_2h_n5_2	Intensity.1WT_20_2h_n5_3	Intensity.3D8_20_2h_n5_1	Intensity.3D8_20_2h_n5_2	Intensity.3D8_20_2h_n5_3
WP_003013731	26.91971	26.03992	26.27133	25.93598	26.19855	26.22295
WP_003013959	25.87033	25.65042	25.74652	26.07970	25.90919	26.03624
WP_003014268	26.79067	26.76619	27.08102	26.62292	26.80336	26.77253
WP_003014232	24.80922	24.96019	25.18033	24.92020	24.80622	24.86137
WP_003014233	25.86714	25.66709	25.77431	25.70902	25.71849	25.53778
WP_003014207	28.05088	27.93109	27.87884	28.01439	28.02191	28.05122

- 1020 proteins

Hypothesis testing: a single protein

Movie clip



T-test

$$\log_2 \text{FC} = \bar{y}_{p1} - \bar{y}_{p2}$$

$$T_g = \frac{\widehat{\log_2 \text{FC}}}{\text{se}_{\widehat{\log_2 \text{FC}}}}$$

$$T_g = \frac{\widehat{\text{signal}}}{\widehat{\text{Noise}}}$$

If we can assume equal variance in both treatment groups:

$$\text{se}_{\log_2 \text{FC}} = \text{SD}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

Linear model

$$\begin{aligned} y_i &= \beta_0 + \beta_{D8} x_{i,D8} + \epsilon_i \\ \epsilon_i &\sim N(0, \sigma^2) \end{aligned}$$

- $x_{i,D8}$ is dummy variable

$$\begin{aligned} x_{i,D8} &= 0 \text{ (WT)} \\ x_{i,D8} &= 1 \text{ (D8)} \end{aligned}$$

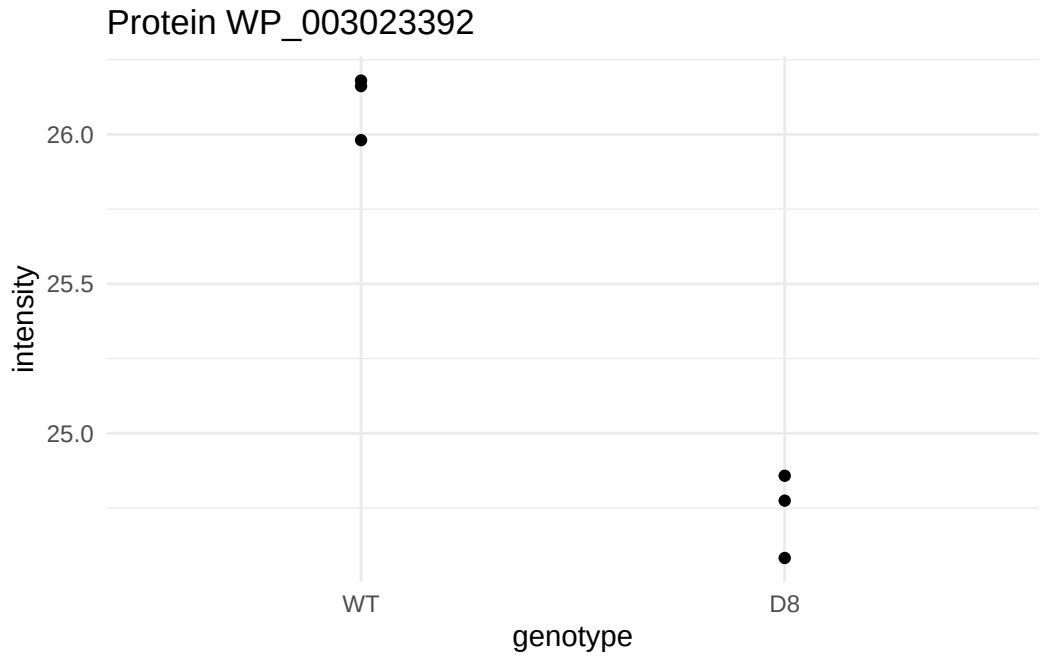
- $\log_2$ FC?

$$\begin{aligned} \hat{\mu}_{WT} &= \hat{\beta}_0 \\ \hat{\mu}_{D8} &= \hat{\beta}_0 + \hat{\beta}_{D8} \\ \widehat{\log_2 \text{FC}} &= \hat{\beta}_{D8} \end{aligned}$$

In R

```
lm(intensity ~ genotype, data = WP_003023392)
```

```r
WP_003023392 <- data.frame(
    intensity = assay(qf[["proteins"]]["WP_003023392",]) |> c(),
    genotype = colData(qf)[,1])

WP_003023392 |>
  ggplot(aes(x=genotype,y=intensity)) +
  geom_point() +
  ggtitle("Protein WP_003023392") +
  theme_minimal()
```



$$t = \frac{\log_2 \widehat{\text{FC}}}{\text{se}_{\log_2 \widehat{\text{FC}}}} = \frac{-1.37}{0.103} = -13.2$$

- Is t = -13.2 indicating that there is an effect?
- How likely is it to observe t = -13.2 when there is no effect of the argP KO on the protein expression?

Null hypothesis (H_0) and alternative hypothesis (H_1)

- With data we can never prove a hypothesis (falsification principle of Popper)

- With data we can only reject a hypothesis
- In general we start from *alternative hypothesis* H_1: we want to show an effect of the KO on a protein

H_1: On average the protein abundance in WT is different from that in KO

- But, we will assess this by falsifying the opposite:
  H_0: On average the protein abundance in WT is equal to that in KO

```
lm1 <- lm(intensity ~ genotype, data = WP_003023392)
summary(lm1)
```

```
Call:
lm(formula = intensity ~ genotype, data = WP_003023392)

Residuals:
       1        2        3        4        5        6
 0.07266 -0.12677  0.05411  0.03622 -0.15565  0.11943

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 26.10746    0.07308  357.27 3.68e-10 ***
genotypeD8  -1.36902    0.10334  -13.25 0.000188 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1266 on 4 degrees of freedom
Multiple R-squared:  0.9777,    Adjusted R-squared:  0.9721
F-statistic: 175.5 on 1 and 4 DF,  p-value: 0.0001877
```

- How likely is it to observe an equal or more extreme effect than the one observed in the sample when the null hypothesis is true?
- When we make assumptions about the distribution of our test statistic we can quantify this probability: *p-value*. The p-value will only be calculated correctly if the underlying assumptions hold!
- When we repeat the experiment, the probability to observe a fold change for this gene that is more extreme than a 2.58 fold ($\log_2 FC = -1.37$) down or up regulation by random chance (if H_0 is true) is 2 out of 10000.

- If the p-value is below a significance threshold α we reject the null hypothesis. *We control the probability of a false positive result at the α-level (type I error)*
- Note, that the p-values are uniform under the null hypothesis, i.e. when H_0 is true all p-values are equally likely.

Multiple hypothesis testing

Movie clip

- Consider testing DA for all $m = 1066$ proteins simultaneously
- What if we assess each individual test at level α? $\rightarrow$ Probability to have a false positive (FP) among all m simultaneous tests $>>> \alpha = 0.05$
- Indeed for each non DA protein we have a probability of 5% to return a FP.
- In a typical experiment the majority of the proteins are non DA.
- So an upper bound of the expected FP is $m \times \alpha$ or $1066 \times 0.05 = 53$.

$\rightarrow$ Hence, we are bound to call many false positive proteins each time we run the experiment.

Multiple testing

Family-wise error rate

Movie clip

The family-wise error rate (FWER) addresses the multiple testing issue by no longer controlling the individual type I error for each protein, instead it controls:

$$\text{FWER} = \text{P}\left[FP \geq 1\right]$$

The Bonferroni method is widely used to control the type I error:

- assess each test at
$$\alpha_{\text{adj}} = \frac{\alpha}{m}$$
- or use adjusted p-values and compare them to α:
$$p_{\text{adj}} = \min\left(p \times m, 1\right)$$

Problem, the method is very conservative!

False discovery rate

Movie clip

- FDR: Expected proportion of false positives on the total number of positives you return.
- An FDR of 1% means that on average we expect 1% false positive proteins in the list of proteins that are called significant.
- Defined by Benjamini and Hochberg in their seminal paper Benjamini, Y. and Hochberg, Y. (1995). "Controlling the false discovery rate: a practical and powerful approach to multiple testing". Journal of the Royal Statistical Society Series B, 57 (1): 289–300.

The **False Discovery Proportion (FDP)** is the fraction of false positives that are returned, i.e.

$$FDP = \frac{FP}{R}$$

- However, this quantity cannot be observed because in practice we only know the number of proteins for which we rejected H_0, R.
- But, we do not know the number of false positives, FP.

Therefore, Benjamini and Hochberg, 1995, defined The **False Discovery Rate (FDR)** as

$$\text{FDR} = \text{E}\left[\frac{FP}{R}\right] = \text{E}\left[\text{FDP}\right]$$

the expected FDP.

- Controlling the FDR allows for more discoveries (i.e. longer lists with significant results), while the fraction of false discoveries among the significant results is well controlled on average. As a consequence, more of the true positive hypotheses will be detected.

Intuition of BH-FDR procedure

Consider $m = 1000$ tests

- Suppose that a researcher rejects all null hypotheses for which $p < 0.01$.
- If we use $p < 0.01$, we expect $0.01 \times m_0$ tests to return false positives.
- A conservative estimate of the number of false positives that we can expect can be obtained by considering that the null hypotheses are true for all features, $m_0 = m = 1000$.
- We then would expect $0.01 \times 1000 = 10$ false positives ($FP = 10$).
- Suppose that the researcher found 200 genes with $p < 0.01$ ($R = 200$).

- The proportion of false positive results (FDP = false positive proportion) among the list of $R = 200$ genes can then be estimated as

$$\widehat{\text{FDP}} = \frac{\widehat{FP}}{R} = \frac{10}{200} = \frac{0.01 \times 1000}{200} = 0.05.$$

Benjamini and Hochberg (1995) procedure for controlling the FDR at α

1. Let $p_{(1)} \leq ... \leq p_{(m)}$ denote the ordered p-values.
2. Find the largest integer k so that

$$\frac{p_{(k)} \times m}{k} \leq \alpha$$

or

$$p_{(k)} \leq k \times \alpha/m$$

3. If such a k exists, reject the k null hypotheses associated with $p_{(1)}, ..., p_{(k)}$. Otherwise none of the null hypotheses is rejected.

The adjusted p-value (also known as the q-value in FDR literature):

$$q_{(i)} = \tilde{p}_{(i)} = \min\left[\min_{j=i,...,m}\left(mp_{(j)}/j\right), 1\right].$$

In the hypothetical example above: $k = 200$, $p_{(k)} = 0.01$, $m = 1000$ and $\alpha = 0.05$.

Francisella Example

Movie clip

Click to see code

```
ttestMx <- function(y,group) {
    test <- try(t.test(y[group],y[!group],var.equal=TRUE),silent=TRUE)
    if(is(test,"try-error")) {
      return(c(log2FC=NA,se=NA,tstat=NA,p=NA))
      } else {
      return(c(log2FC= (test$estimate%*%c(1,-1)),se=test$stderr,tstat=test$statistic,pval=tes
      }
 }

 res <- apply(
    assay(qf[["proteins"]]),
```

```
    1,
    ttestMx,
    group = colData(qf)$genotype=="D8") |>
  t()
 colnames(res) <- c("logFC","se","tstat","pval")
 res <- res |>
   as.data.frame() |>
   na.exclude() |>
   arrange(pval)
 res$adjPval <- p.adjust(res$pval, "fdr")
 alpha <- 0.05
res$adjAlphaForm <- paste0(1:nrow(res)," x ",alpha,"/",nrow(res))
res$adjAlpha <- alpha * (1:nrow(res))/nrow(res)
res$"pval < adjAlpha" <- res$pval < res$adjAlpha
res$"adjPval < alpha" <- res$adjPval < alpha
```

Below you can find a results table illustrating the BH-FDR correction. For your reference we also give the Bonferroni adjusted significance level that can be applied to the raw p-values, so that you can also assess how many features we would return when controlling the FWER instead: $\alpha_{\text{adj}}^{\text{Bonferroni}} = \alpha/m = 0.05/1020 = 5 \times 10^{-5}$

	logFC	pval	adjPval	adjAlphaForm	adjAlpha	pval < adjAlpha	adjPval < alpha
WP_003034781	1.1224142	0.0000022	0.0022721	1 x 0.05/1020	0.0000490	TRUE	TRUE
WP_003040849	-1.2263612	0.0000076	0.0026390	2 x 0.05/1020	0.0000980	TRUE	TRUE
WP_011733723	-0.3258931	0.0000078	0.0026390	3 x 0.05/1020	0.0001471	TRUE	TRUE
WP_003041244	1.2057070	0.0000791	0.0201736	4 x 0.05/1020	0.0001961	TRUE	TRUE
WP_003038430	-0.3836478	0.0001838	0.0280097	5 x 0.05/1020	0.0002451	TRUE	TRUE
WP_003023392	-1.3690155	0.0001877	0.0280097	6 x 0.05/1020	0.0002941	TRUE	TRUE
WP_011733588	-0.3823744	0.0001922	0.0280097	7 x 0.05/1020	0.0003431	TRUE	TRUE
WP_011733045	1.3040827	0.0002362	0.0301195	8 x 0.05/1020	0.0003922	TRUE	TRUE
WP_004339068	1.8975424	0.0006535	0.0730468	9 x 0.05/1020	0.0004412	FALSE	FALSE

	logFC	pval	adjPval	adjAlphaForm	adjAlpha	pval < adjAlpha	adjPval < alpha
WP_003018840	-0.5013825	0.0007161	0.0730468	10 x 0.05/1020	0.0004902	FALSE	FALSE

...

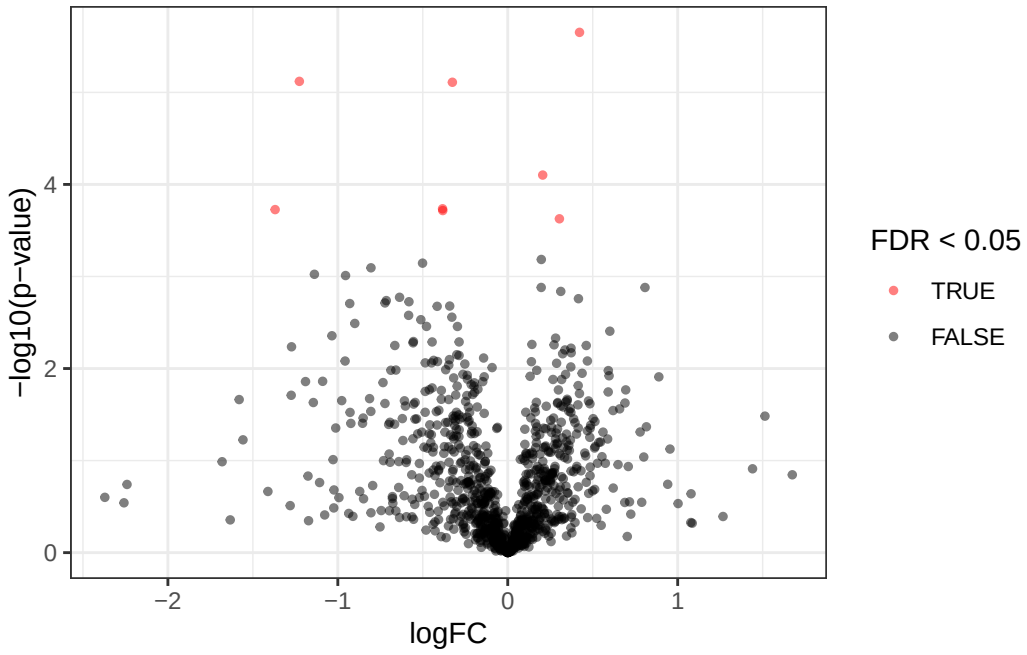
WP_003022463	-0.0003682	0.9989893	0.9999324	1019 x 0.05/1020	0.049951	FALSE	FALSE
WP_003033335	-0.0000299	0.9999324	0.9999324	1020 x 0.05/1020	0.050000	FALSE	FALSE

Results

Click to see code

```
volcanoT <- plotVolcano(res)
```

```
volcanoT
```



Moderated Statistics

Movie clip

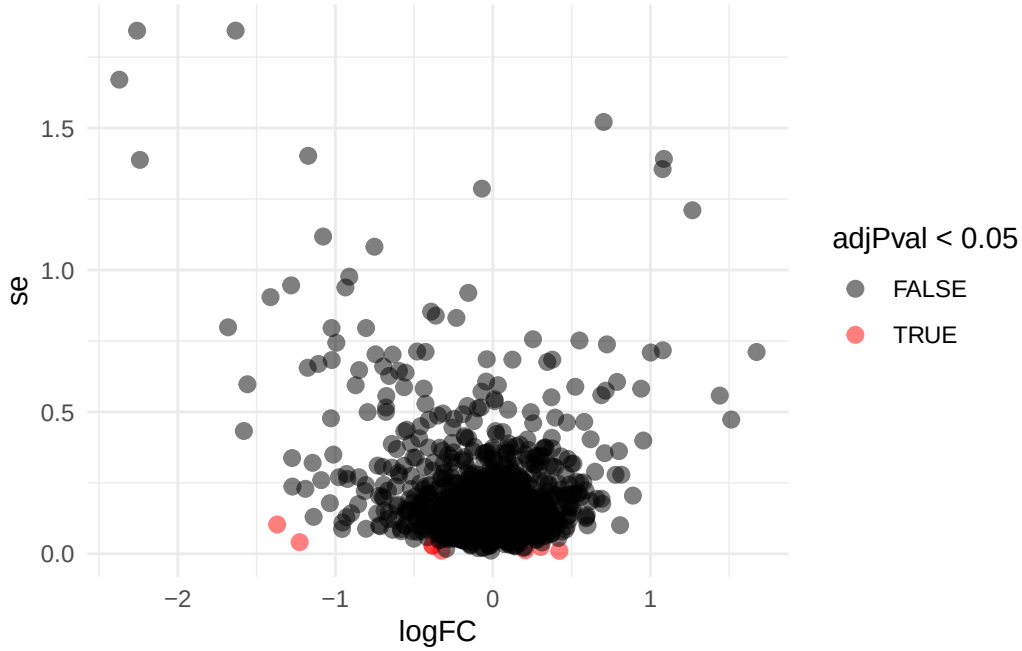
Problems with ordinary t-test

Click to see code

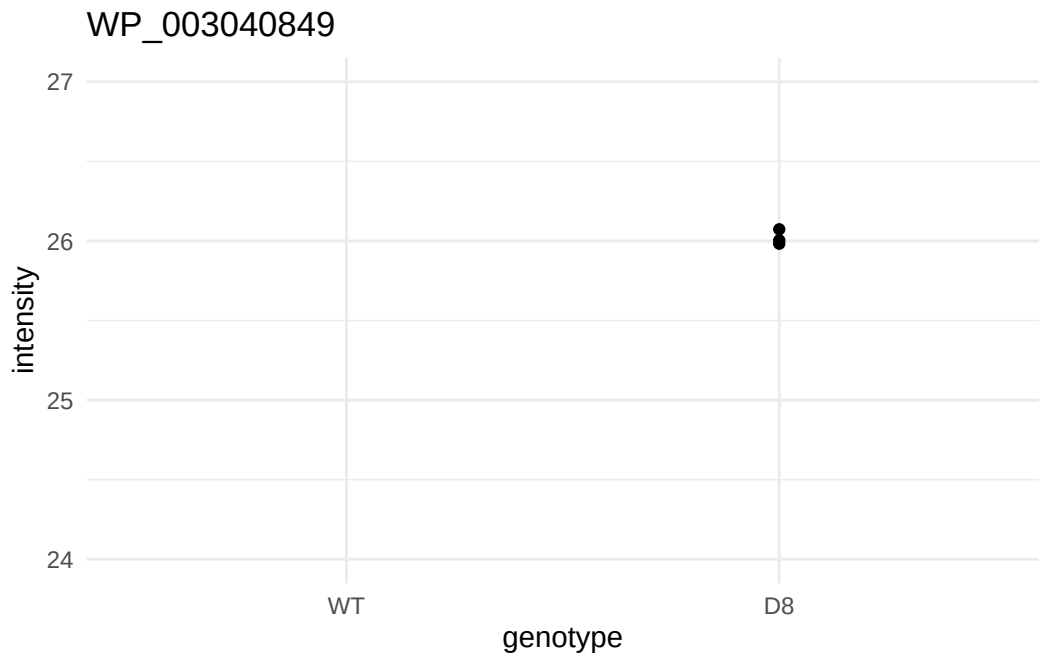
```
problemPlots <- list()
problemPlots[[1]] <- res |>
  ggplot(aes(x = logFC, y = se, color = adjPval < 0.05)) +
    geom_point(cex = 2.5) +
    scale_color_manual(values = alpha(c("black", "red"), 0.5)) +
    theme_minimal()

for (i in 2:3)
{
problemPlots[[i]] <- colData(qf) |>
    as.data.frame() |>
    mutate(intensity = qf[["proteins"]][rownames(res)[i],] |>
             assay() |>
             c()) |>
    ggplot(aes(x=genotype,y=intensity)) +
    geom_point() +
    ggtitle(rownames(res)[i]) +
    ylim(c(24,27)) +
      theme_minimal()
}
```

```
problemPlots
```

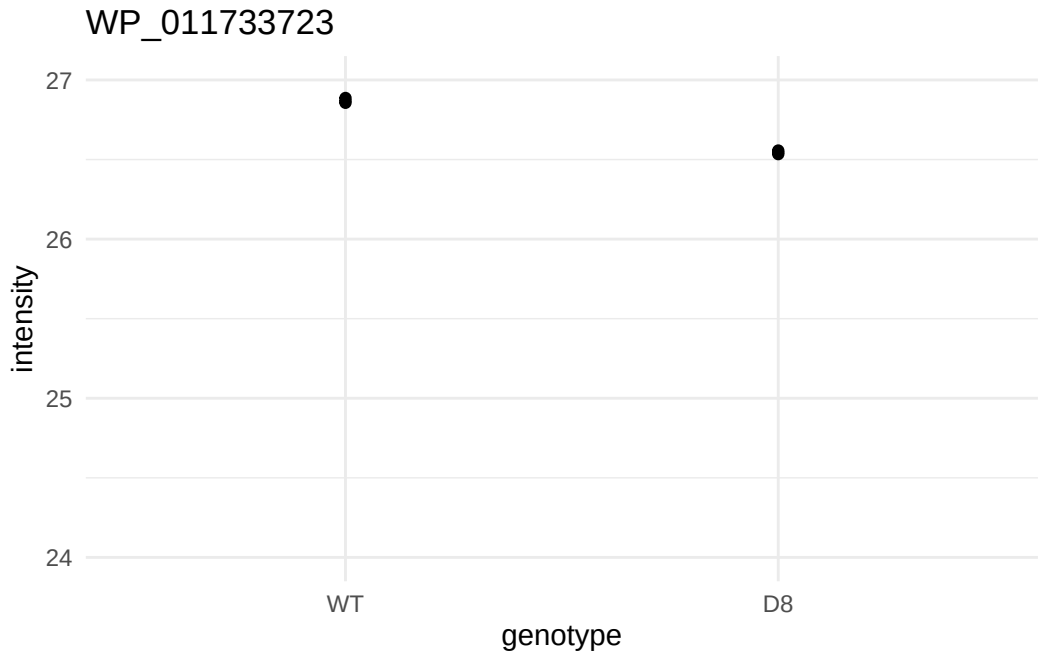
```
[[1]]
```


1.5
1.0
0.5
0.0
se
−2
−1
0
1
logFC
adjPval < 0.05
FALSE
TRUE

[[2]]


WP_003040849
27
26
25
24
intensity
WT
D8
genotype

```
[[3]]
```



A general class of moderated test statistics is given by

$$T_g^{mod} = \frac{\bar{Y}_{g1} - \bar{Y}_{g2}}{C \quad \tilde{S}_g},$$

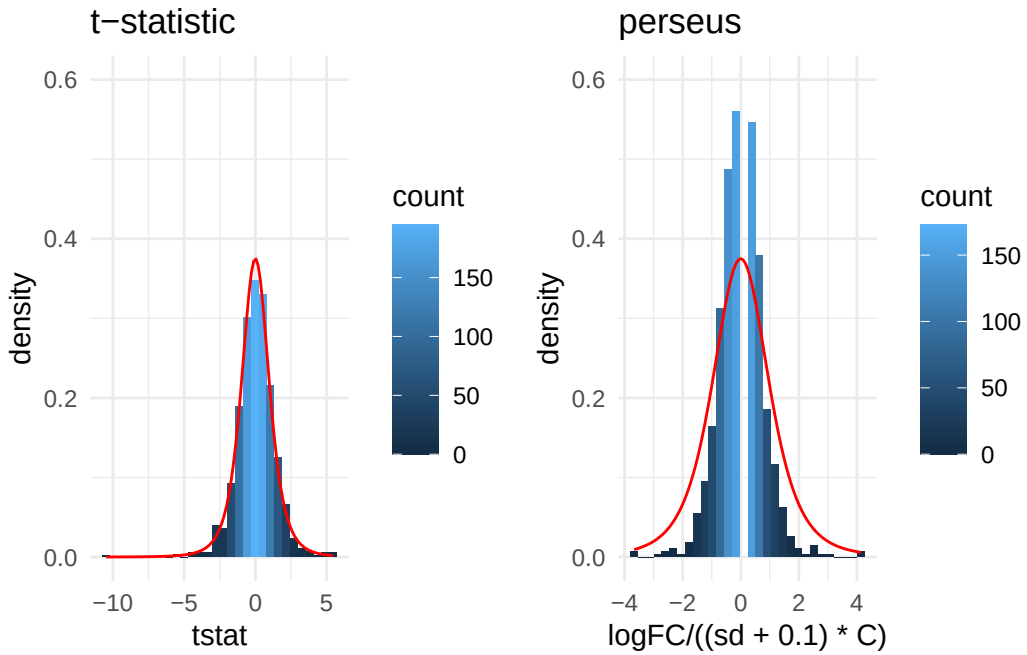
where $\tilde{S}_g$ is a moderated standard deviation estimate.

- C is a constant depending on the design e.g. $\sqrt{1/n_1 + 1/n_2}$ for a t-test and of another form for linear models.
- $\tilde{S}_g = S_g + S_0$: add small positive constant to denominator of t-statistic.
- This can be adopted in perseus.

Click to see code

```
simI<-sapply(res$se/sqrt(1/3+1/3),function(n,mean,sd) rnorm(n,mean,sd),n=6,mean=0) |> t()
resSim <- apply(
    simI,
    1,
    ttestMx,
    group = colData(qf)$genotype=="D8") |>
```

```
 t()
colnames(resSim) <- c("logFC","se","tstat","pval")
resSim <- as.data.frame(resSim)
tstatSimPlot <- resSim |>
  ggplot(aes(x=tstat)) +
    geom_histogram(aes(y=..density.., fill=..count..),bins=30) +
    stat_function(fun=dt,
   color="red",
   args=list(df=4)) +
  ylim(0,.6) +
  ggtitle("t-statistic") +
  theme_minimal()


resSim$C <- sqrt(1/3+1/3)
resSim$sd <- resSim$se/resSim$C
tstatSimperseus <- resSim |>
  ggplot(aes(x=logFC/((sd+.1)*C))) +
    geom_histogram(aes(y=..density.., fill=..count..),bins=30) +
    stat_function(fun=dt,
                  color="red",
                 args=list(df=4)) +
    ylim(0,.6) +
   ggtitle("perseus") +
  theme_minimal()
```



- The choice of S_0 in perseus is ad hoc and the t-statistic is no-longer t-distributed.
- permutation test, but is difficult for more complex designs.
- Allows for Data Dredging because user can choose S_0

Empirical Bayes

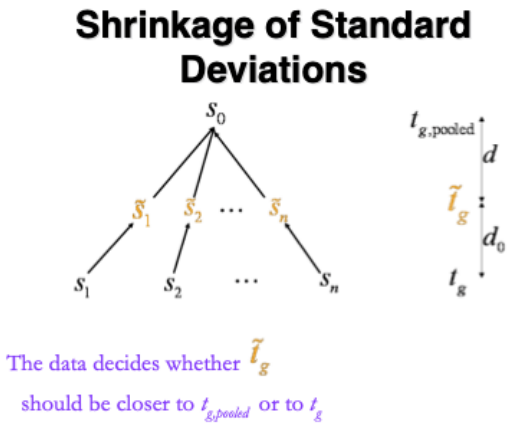


Figure courtesy to Rafael Irizarry

$$T_g^{mod} = \frac{\bar{Y}_{g1} - \bar{Y}_{g2}}{C \quad \tilde{S}_g},$$

- **empirical Bayes** theory provides a formal framework for borrowing strength across proteins,
- Implemented in popular bioconductor packages **limma** and **msqrob2**

$$\tilde{S}_g = \sqrt{\frac{d_g S_g^2 + d_0 S_0^2}{d_g + d_0}},$$

- S_0^2: common variance (over all proteins)
- Moderated t-statistic is t-distributed with $d_0 + d_g$ degrees of freedom.
- Note that the degrees of freedom increase by borrowing strength across proteins!

Click to see the code

1. We model the protein level expression values using the `msqrob` function. By default `msqrob2` estimates the model parameters using robust regression.

We will model the data with a different group mean for every genotype. The group is incoded in the variable `genotype` of the colData. We can specify this model by using a formula with the factor `genotype` as its predictor: `formula = ~genotype`.

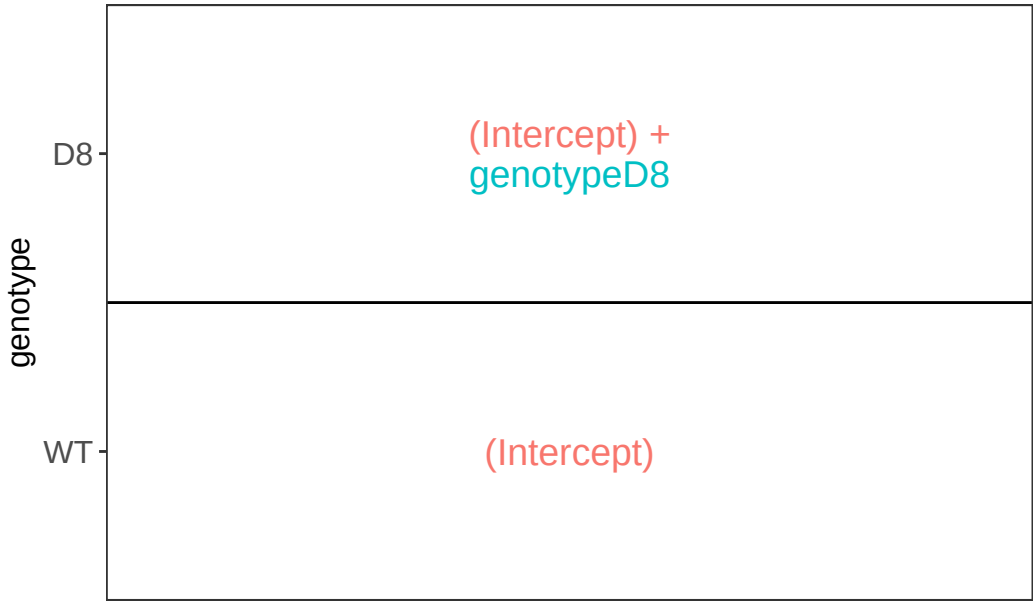
Note, that a formula always starts with a symbol '~'.

```
qf <- msqrob(object = qf, i = "proteins", formula = ~genotype)
```

2. Inference

We first explore the design of the model that we specified using the the package `ExploreModelMatrix`

```
library(ExploreModelMatrix)
VisualizeDesign(colData(qf),~genotype)$plotlist[[1]]
```



We have two model parameters, the (Intercept) and genotypeD8. This results in a model with two group means:

1. For the wild type (WT) the expected value (mean) of the log2 transformed intensity y for a protein will be modelled using

$$\mathrm{E}[Y|\text{genotype} = \text{WT}] = (\text{Intercept})$$

2. For the knockout genotype D8 the expected value (mean) of the log2 transformed intensity y for a protein will be modelled using

$$\mathrm{E}[Y|\text{genotype} = \text{D8}] = (\text{Intercept}) + \text{genotypeD8}$$

The average log2FC between D8 and WT is thus

$$\log_2 \mathrm{FC}_{D8-WT} = \mathrm{E}[Y|\text{genotype} = \text{D8}] - \mathrm{E}[Y|\text{genotype} = \text{WT}] = \text{genotypeD8}$$

Hence, assessing the null hypothesis that there is no differential abundance between D8 and WT can be reformulated as

$$H_0 : \text{genotypeD8} = 0$$

We can implement a hypothesis test for each protein in msqrob2 using the code below:

```
L <- makeContrast("genotypeD8 = 0", parameterNames = c("genotypeD8"))
qf <- hypothesisTest(object = qf, i = "proteins", contrast = L)
```

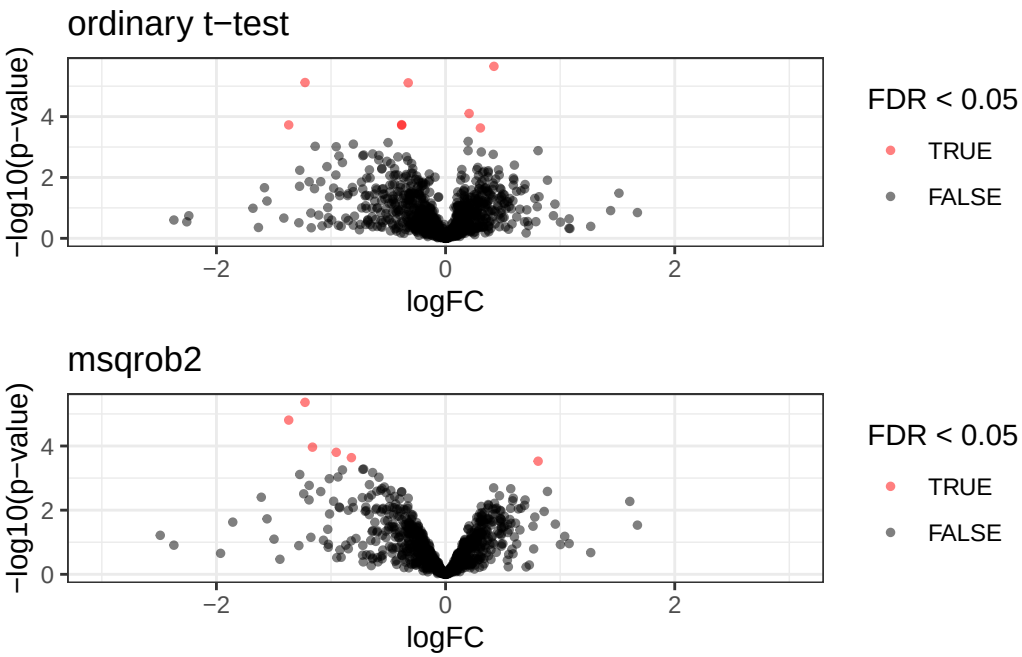
We can show the list with all significant DE proteins at the 5% FDR using

```
msqrobCollect(qf[["proteins"]],L) |>
  arrange(pval) |>
  filter(adjPval<0.05)
```

```
                            logFC         se       df          t         pval
genotypeD8.WP_003040849 -1.2263612 0.08020151 6.087933 -15.290999 4.358069e-
06
genotypeD8.WP_003023392 -1.3690155 0.11105902 6.087933 -12.326919 1.560512e-
05
genotypeD8.WP_003033719 -1.1615474 0.12602389 5.828253  -9.216882 1.087304e-
04
genotypeD8.WP_003026016 -0.9540456 0.11530658 6.080135  -8.273991 1.573910e-
04
genotypeD8.WP_003033046 -0.8219170 0.10214460 5.815009  -8.046603 2.307079e-
04
genotypeD8.WP_003038350  0.8077325 0.10952806 6.087933   7.374663 2.978545e-
04
                            adjPval   contrast      feature
genotypeD8.WP_003040849 0.004301414 genotypeD8 WP_003040849
genotypeD8.WP_003023392 0.007701128 genotypeD8 WP_003023392
genotypeD8.WP_003033719 0.035772317 genotypeD8 WP_003033719
genotypeD8.WP_003026016 0.038836240 genotypeD8 WP_003026016
genotypeD8.WP_003033046 0.045541744 genotypeD8 WP_003033046
genotypeD8.WP_003038350 0.048997070 genotypeD8 WP_003038350
```

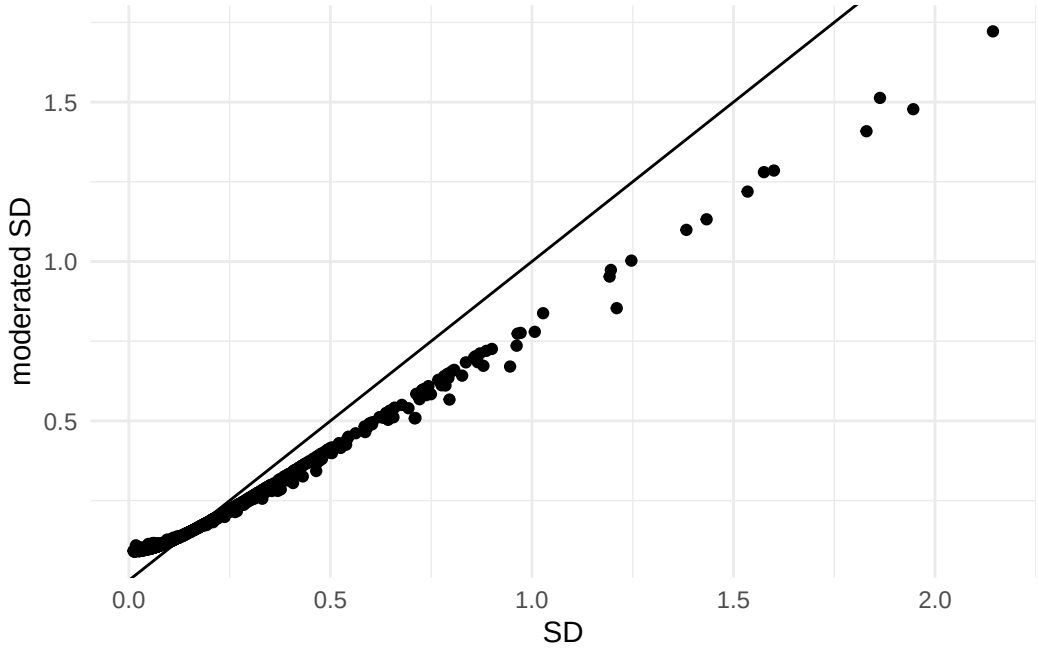
We can also visualise the results using a volcanoplot

```
volcano <- plotVolcano(
    rowData(qf[["proteins"]])$genotypeD8) +
  ggtitle("msqrob2")
```



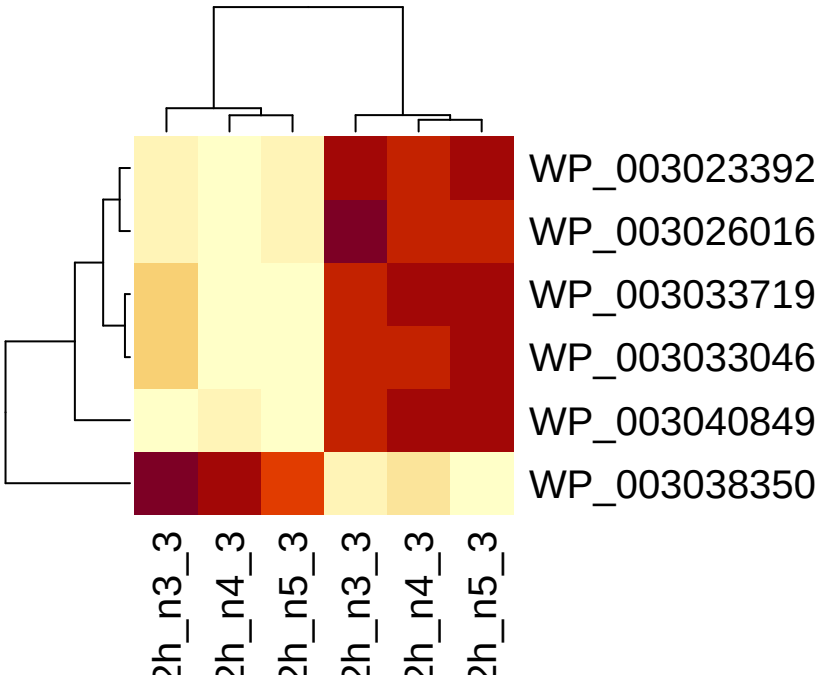
- The volcano plot opens up when using the EB variance estimator
- Borrowing strength to estimate the variance using empirical Bayes solves the issue of returning proteins with a low fold change as significant due to a low variance.

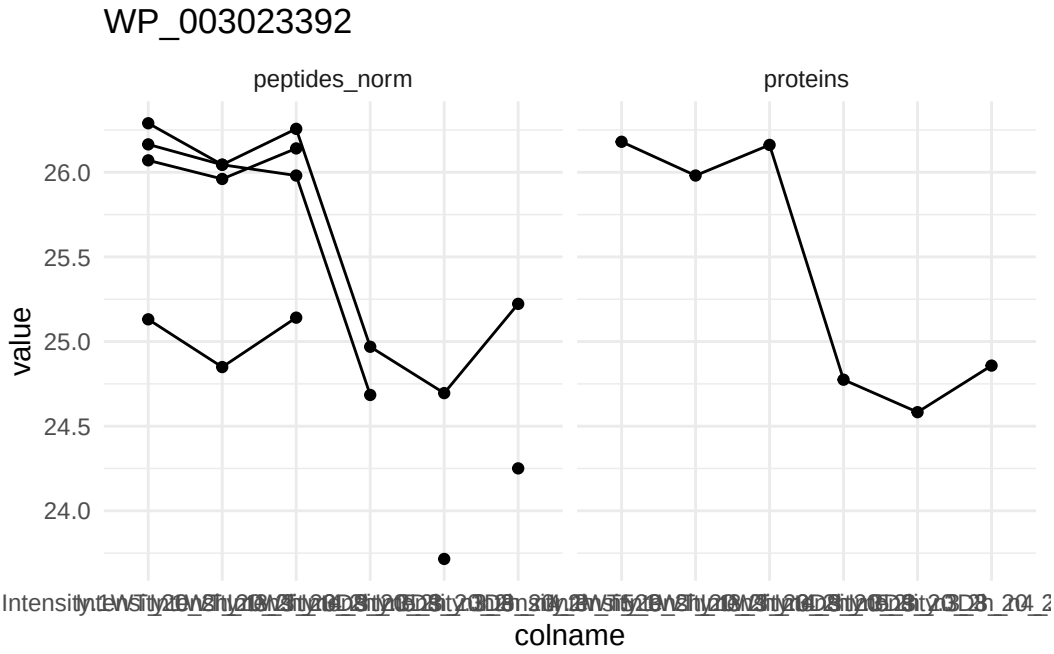
Shrinkage of the variance and moderated t-statistics

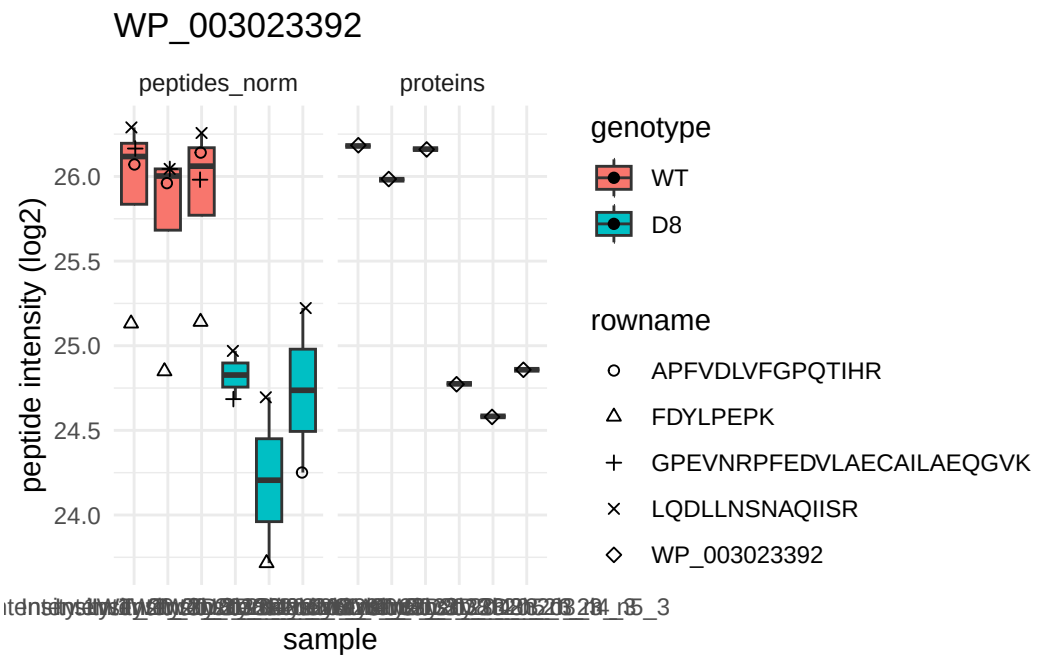


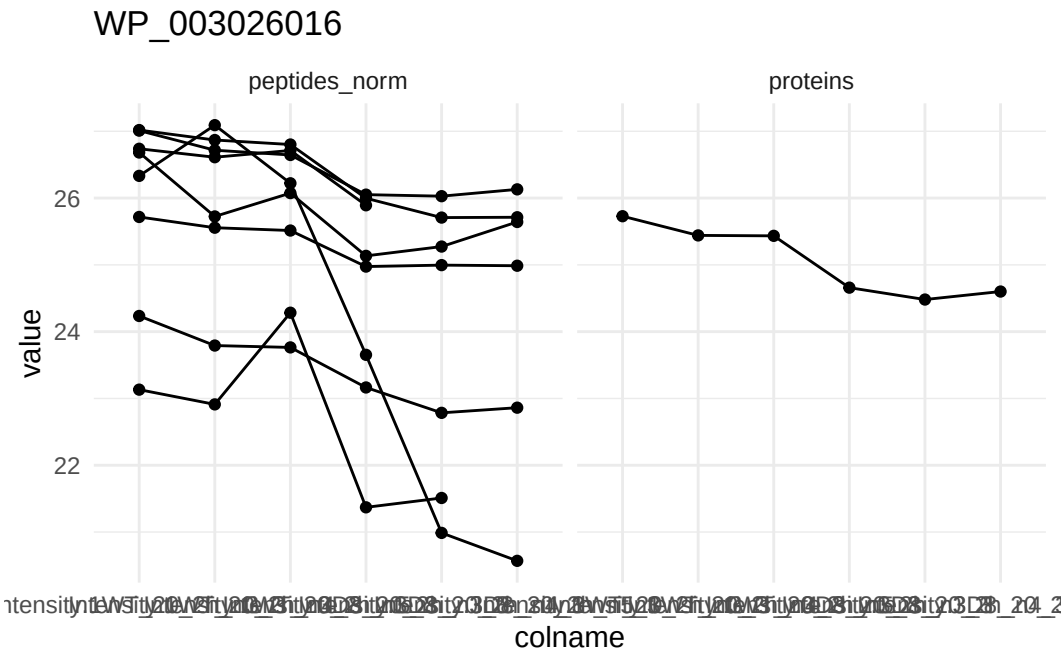
- Small variances are shrunken towards the common variance resulting in large EB variance estimates
- Large variances are shrunken towards the common variance resulting in smaller EB variance estimates
- Pooled degrees of freedom of the EB variance estimator are larger because information is borrowed across proteins to estimate the variance

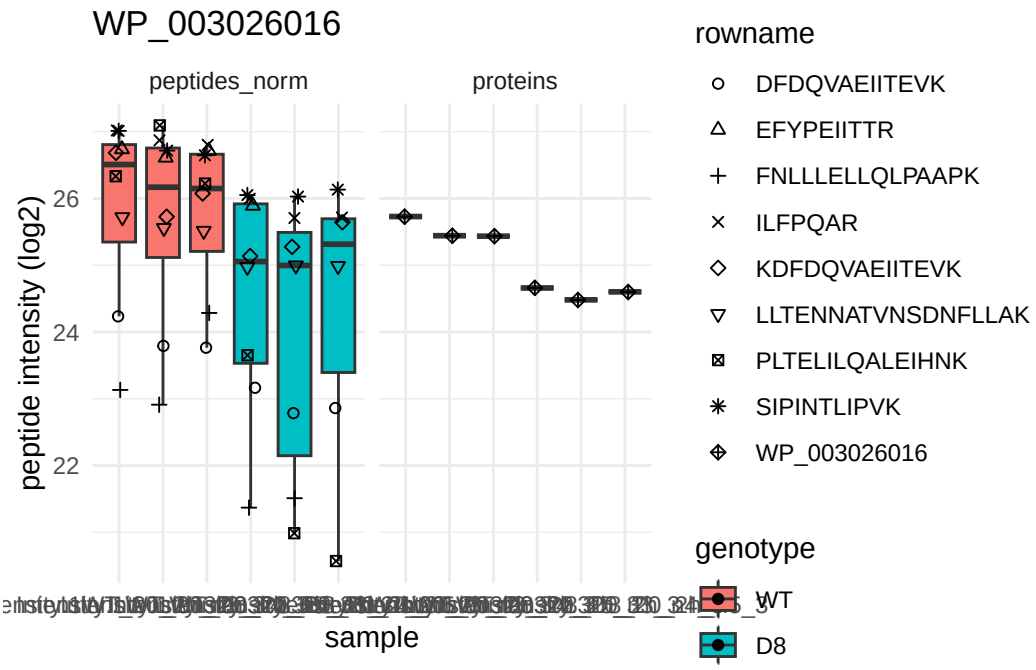
Plots

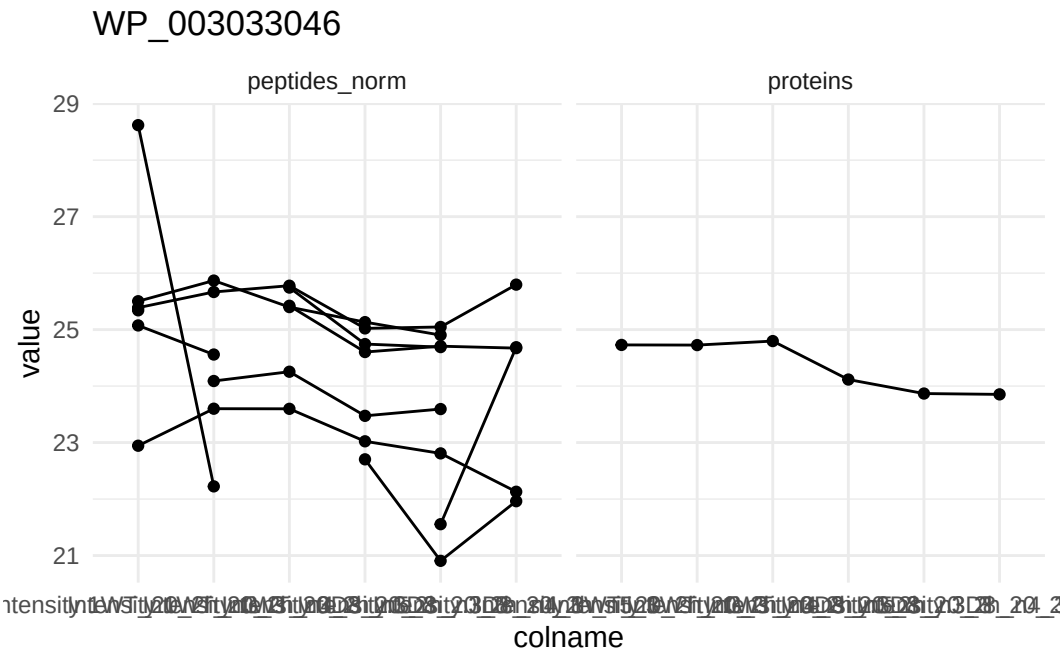

WP_003023392
WP_003026016
WP_003033719
WP_003033046
WP_003040849
WP_003038350

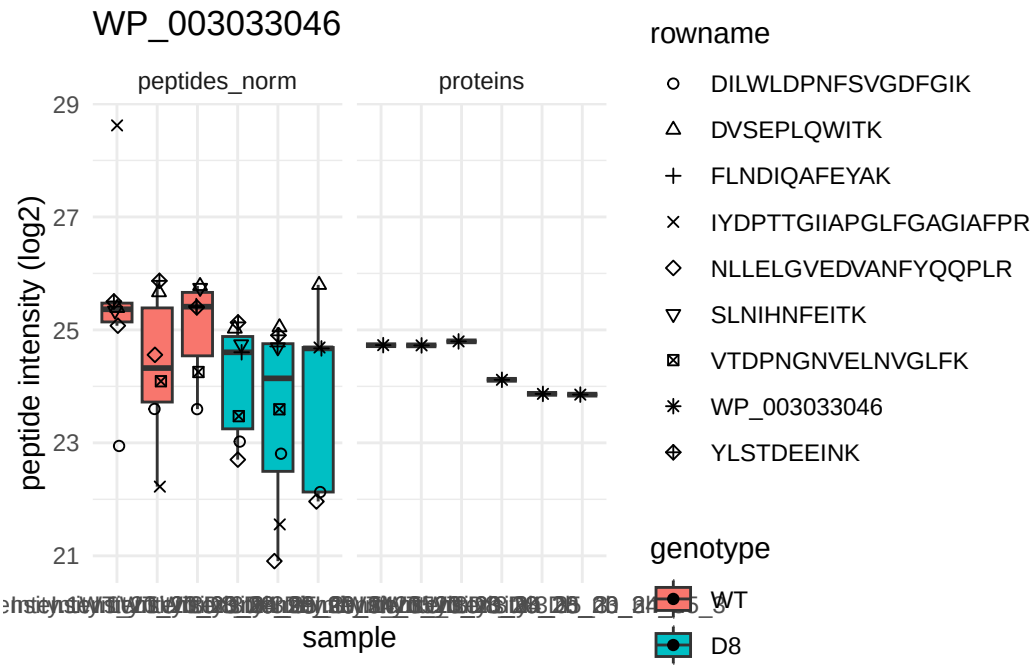

WP_003023392
peptides_norm
proteins
value
26.0
25.5
25.0
24.5
24.0
colname

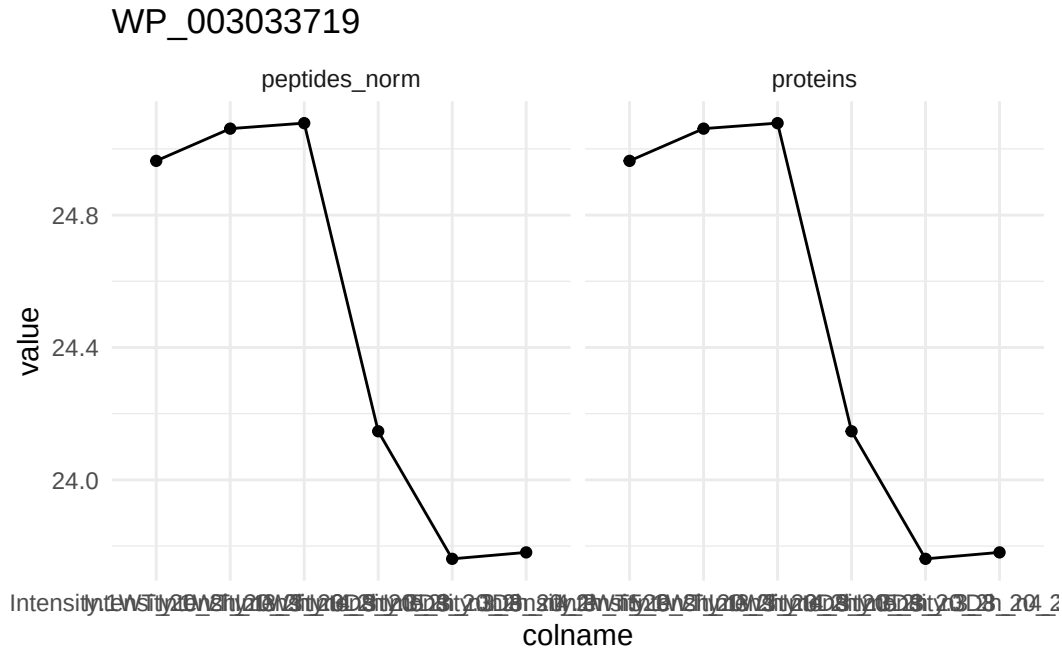

WP_003023392
peptides_norm
proteins
peptide intensity (log2)
26.0
25.5
25.0
24.5
24.0
sample
genotype
WT
D8
rowname
APFVDLVFGPQTIHR
FDYLPEPK
GPEVNRPFEDVLAECAILAEQGVK
LQDLLNSNAQIISR
WP_003023392

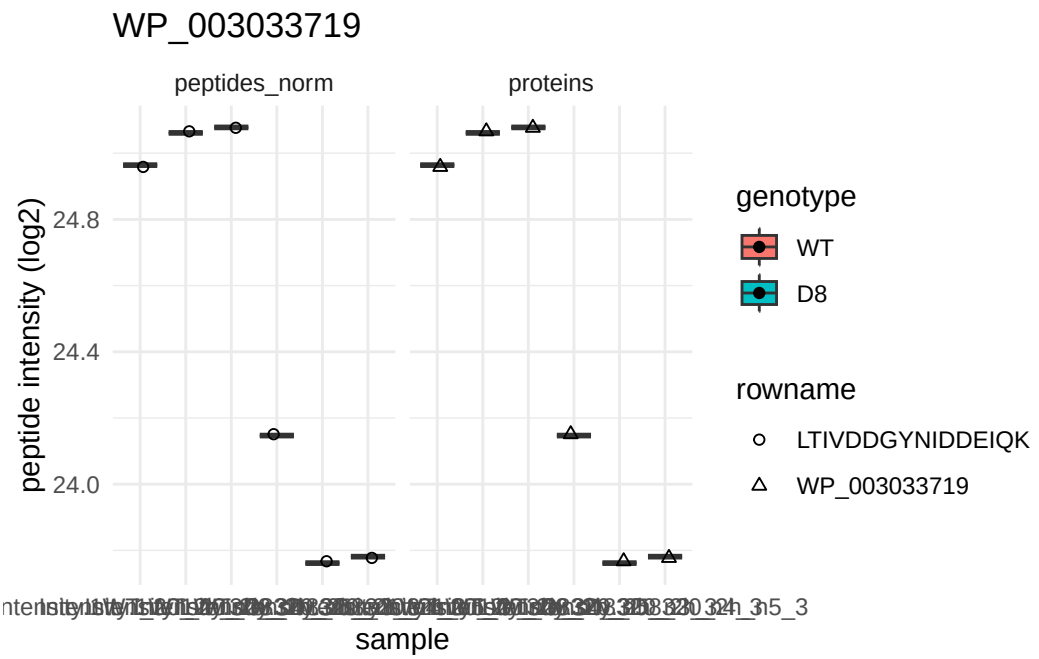

WP_003026016
peptides_norm
proteins
value
26
24
22
colname

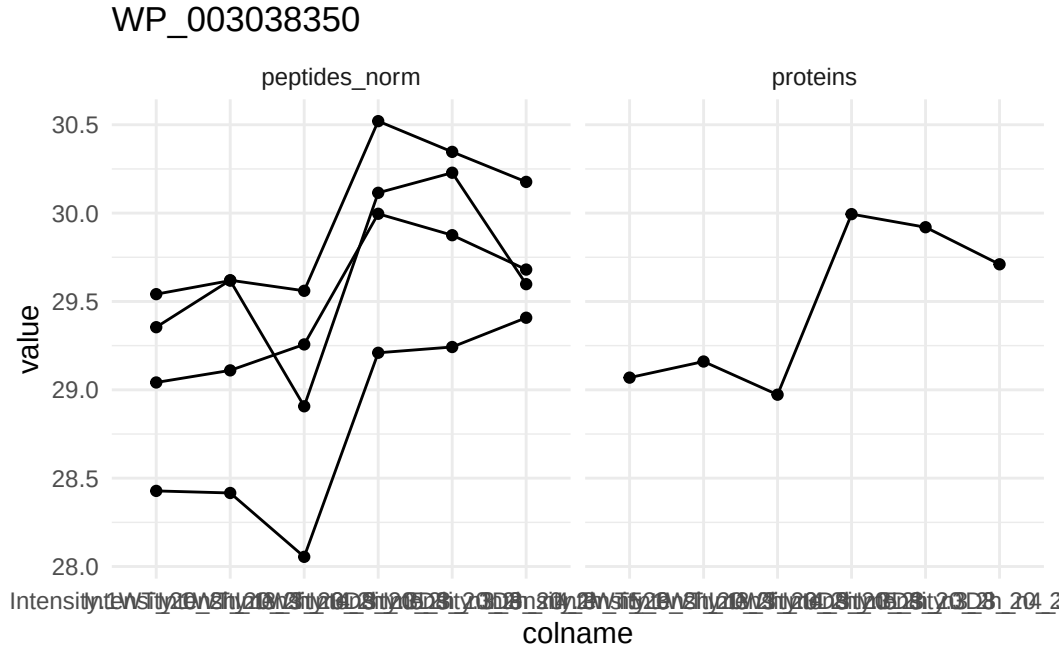

WP_003026016
peptides_norm
proteins
peptide intensity (log2)
26
24
22
sample
rowname
DFDQVAEIITEVK
EFYPEIITTR
FNLLLELLQLPAAPK
ILFPQAR
KDFDQVAEIITEVK
LLTENNATVNSDNFLLAK
PLTELILQALEIHNK
SIPINTLIPVK
WP_003026016
genotype
WT
D8

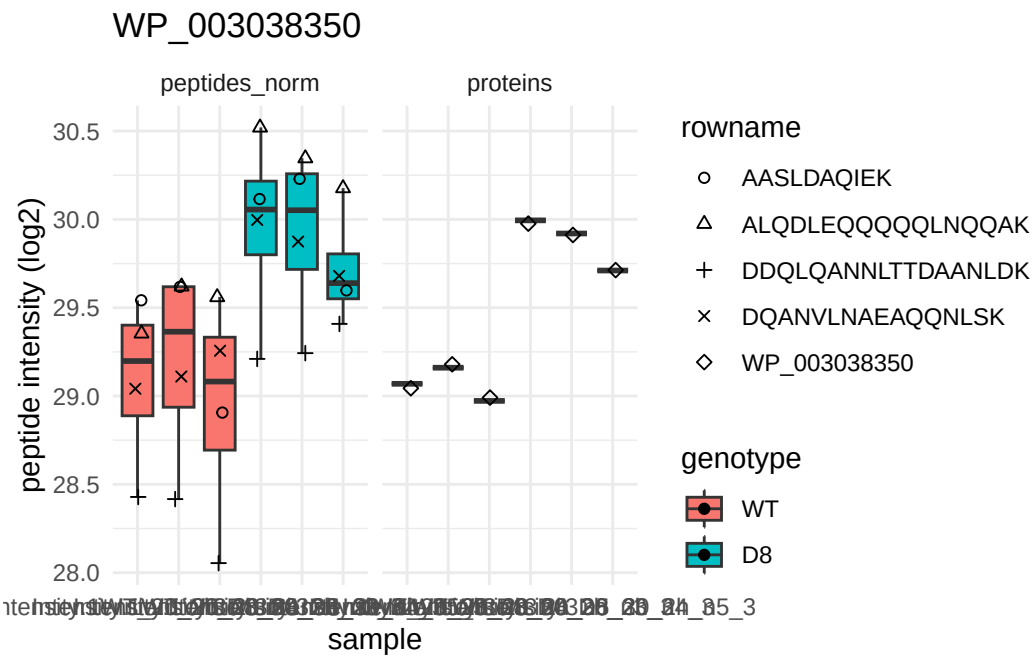

WP_003033046
peptides_norm
proteins
value
29
27
25
23
21
colname

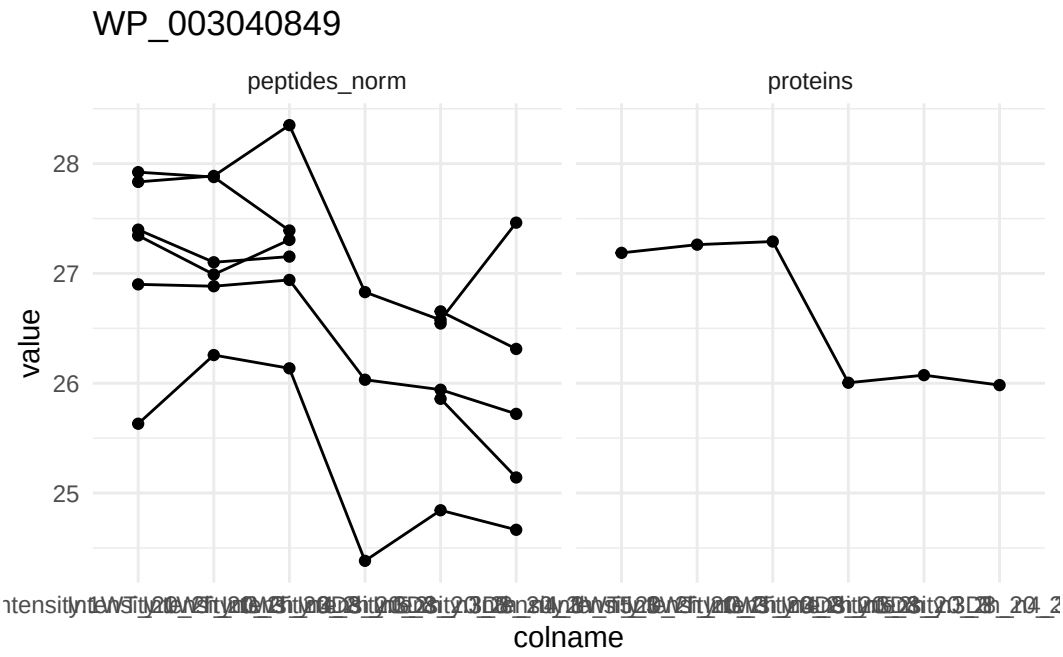

WP_003033046
peptides_norm
proteins
peptide intensity (log2)
29
27
25
23
21
sample
rowname
DILWLDPNFSVGDFGIK
DVSEPLQWITK
FLNDIQAFEYAK
IYDPTTGIIAPGLFGAGIAFPR
NLLELGVEDVANFYQQPLR
SLNIHNFEITK
VTDPNGNVELNVGLFK
WP_003033046
YLSTDEEINK
genotype
WT
D8

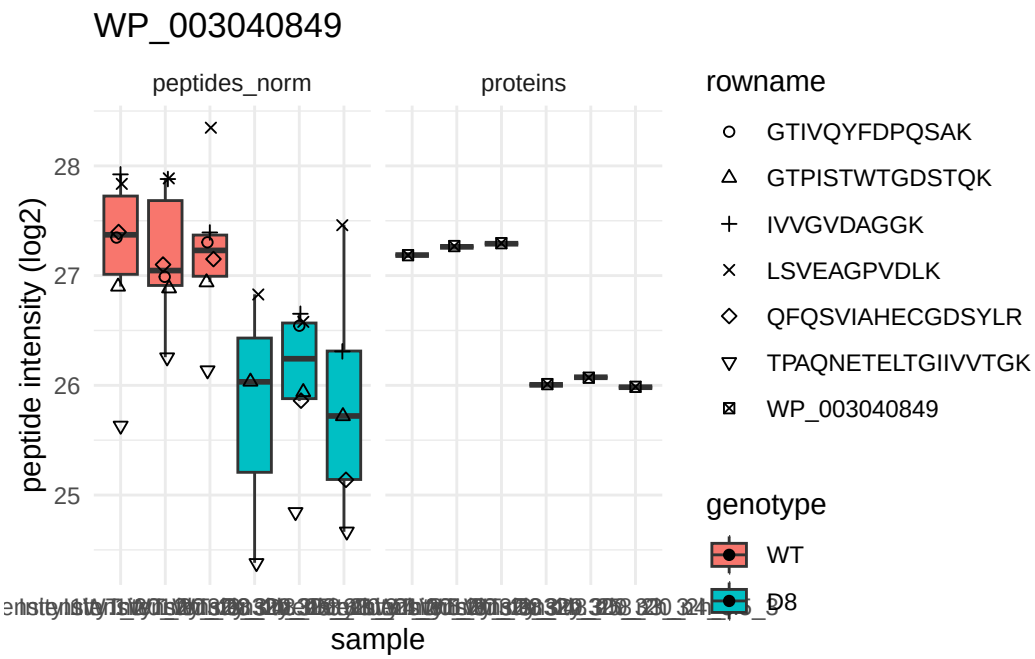

WP_003033719
peptides_norm
proteins
value
24.8
24.4
24.0
colname


WP_003033719
peptides_norm
proteins
peptide intensity (log2)
24.8
24.4
24.0
genotype
WT
D8
rowname
LTIVDDGYNIDDEIQK
WP_003033719
sample


WP_003038350
peptides_norm
proteins
value
30.5
30.0
29.5
29.0
28.5
28.0
colname


WP_003038350
peptides_norm
proteins
peptide intensity (log2)
30.5
30.0
29.5
29.0
28.5
28.0
sample
rowname
AASLDAQIEK
ALQDLEQQQQQLNQQAK
DDQLQANNLTTDAANLDK
DQANVLNAEAQQNLSK
WP_003038350
genotype
WT
D8


WP_003040849
peptides_norm
proteins
value
28
27
26
25
colname



Experimental Design

Random Sampling

- Random sampling is closely related to the concept of the population or the scope of the study.
- Based on a sample of subjects, the researchers want to come to conclusions that hold for
  - all kinds of people
  - people with hypertension
- Scope of the study should be well specified before the start of the study.
- Representative sample: For the statistical analysis to be valid, it is required that the subjects are selected completely at random from the population to which we want to generalize our conclusions.
- Selecting completely at random from a population implies:
  - all subjects in the population should have the same probability of being selected in the sample,
  - the selection of a subject in the sample should be independent from the selection of the other subjects in the sample.

Randomisation

- Make sure that groups are comparable / Avoid systematic differences between groups
- Randomisation: treatments of interest are attributed at random to the experimental units (subjects, animals, plants, cages, …)

Control

Good control is necessary

→ Placebo controlled double blind experiments
→ Injection of control animal with blank containing same solvents, etc.

Replication / Sample size

Within an experiment: Enables to estimate uncertainty / biological variability

- Replication is essential for quantifying the noise
- Noise: biological and technical in nature

Between experiments: Any true finding should be reproducible

- Genuine replicates include all sources of variability: technical + biological
- Technical replicates are important if you assess new technologies

Movie clip

$$\log_2 \mathrm{FC} = \bar{y}_{p1} - \bar{y}_{p2}$$

$$T_g = \frac{\log_2 \mathrm{FC}}{\mathrm{se}_{\log_2 \mathrm{FC}}}$$

$$T_g = \frac{\widehat{\mathrm{signal}}}{\widehat{\mathrm{Noise}}}$$

If we can assume equal variance in both treatment groups:

$$\mathrm{se}_{\log_2 \mathrm{FC}} = \mathrm{SD}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

→ if number of bio-repeats increases we have a higher power!

- cfr. Study of tamoxifen treated Estrogen Receptor (ER) positive breast cancer patients (tutorial)

Randomized complete block designs

$$\sigma^2 = \sigma^2_{bio} + \sigma^2_{\text{lab}} + \sigma^2_{\text{extraction}} + \sigma^2_{\text{run}} + \ldots$$

- Biological: fluctuations in protein level between mice, fluctuations in protein level between cells, ...
- Technical: cage effect, lab effect, week effect, plasma extraction, MS-run, ...

Nature methods: Points of significance - Blocking

https://www.nature.com/articles/nmeth.3005.pdf

Mouse example

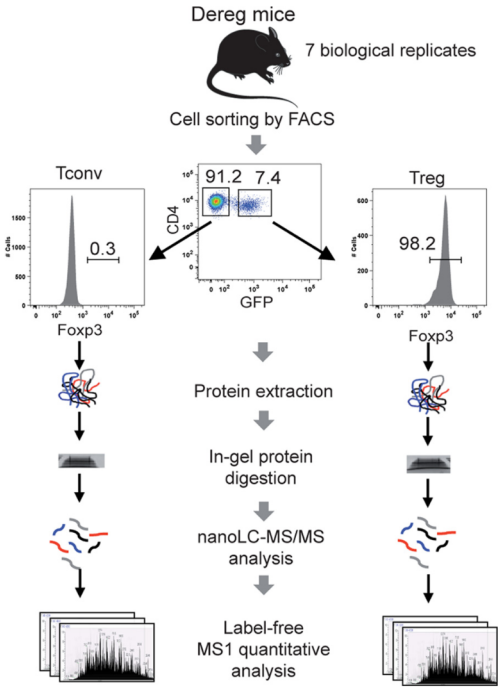


FIG. 1. **Label-free quantitative analysis of conventional and regulatory T cell proteomes.** General analytical workflow based on cell sorting by flow cytometry using the DEREG mouse model and parallel proteomic analysis of Tconv and Treg cell populations by nanoLC-MS/MS and label-free relative quantification.

Duguet et al. (2017) MCP 16(8):1416-1432. doi: 10.1074/mcp.m116.062745

- All treatments of interest are present within block!
- We can estimate the effect of the treatment within block!
- We can isolate the between block variability from the analysis using linear model:

$$\sim \text{type} + \text{mouse}$$

- Not possible with perseus!

Assess the impact of blocking in the tutorial session!

- Completely randomized design with only one cell type per mouse (Treg and Tconv)

$$\updownarrow$$

- Randomized complete block design assessing Treg and Tconv in each mouse

References