Mixed Models for TMT: Mouse Diet Case Study

Lieven Clement

Solution of the TMT exercise in the design tutorial.

Load packages

```
library("msqrob2")
library("dplyr")
library("ggplot2")
library("patchwork")
library("BiocParallel")
register(SerialParam())
```

Experimental context

The data used in this exercise has been published by Plubell et al. (2017) (PXD005953). The objective of the experiment was to explore the impact of low-fat and high-fat diets on the proteomic content of adipose tissue in mice. It also assesses whether the duration of the diet may impact the results. The authors assigned twenty mice into four groups (5 mice per group) based on their diet, either low-fat (LF) or high-fat (HF), and the duration of the diet, which was classified as short (8 weeks) or long (18 weeks). Samples from the epididymal adipose tissue were extracted from each mice. The samples were then randomly distributed across three TMT 10-plex mixtures for analysis. In each mixture, two reference labels were used, each containing pooled samples that included a range of peptides from all the samples. Not all labels were used, leading to an unbalanced design. Each TMT 10-plex mixture was fractionated into nine parts, resulting in a total of 27 MS runs.

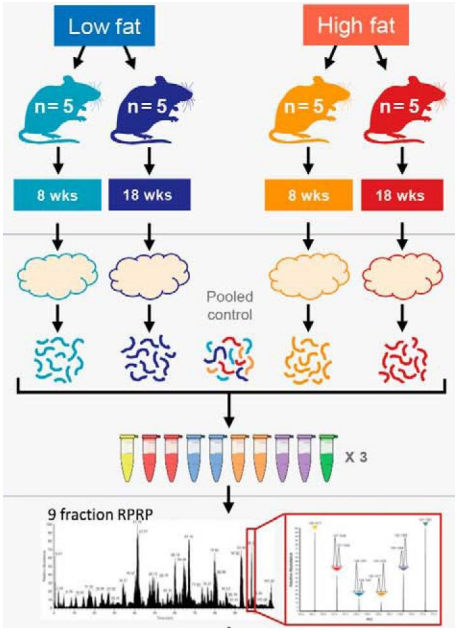


Figure 1: Overview of the experimental design. Taken from Plubell et al. 2017.

Data

The data were reanalyzed by Huang et al. (2020) and have been deposited in the `MSV000084264` MASSiVE repository, but we will retrieve the timestamped data from our Zenodo repository. We need 2 files: the Skyline identification and quantification table generated by the authors and the sample annotation files.

```
library("BiocFileCache")
bfc <- BiocFileCache()
psmFile <- bfcrpath(bfc, "https://zenodo.org/records/14767905/files/mouse_psms.txt?download=
annotFile <- bfcrpath(bfc, "https://zenodo.org/records/14767905/files/mouse_annotations.csv?
```

Let's prepare the data. We will also subset the data set to reduce computational costs. We here randomly sample 500 proteins from the experiment.

```
## Reading data
psms <- read.delim(psmFile)
coldata <- read.csv(annotFile)
coldata$Duration <- gsub("_.*", "", coldata$Condition) ## 1.
coldata$Diet <- gsub(".*_", "", coldata$Condition) ## 2.
```

```
colnames(coldata)[1] <- "Label" ## 3.
## Subsetting data
proteinIds <- unique(psms$Protein.Accessions)
set.seed(1234)
psms <- psms[psms$Protein.Accessions %in% sample(proteinIds, 500), ]
## Converting data
coldata$runCol <- coldata$Run
coldata$quantCols <- paste0("Abundance..", coldata$Label)
mouse <- readQFeatures(psms, colData = coldata,
                       quantCols = unique(coldata$quantCols),
                       runCol = "Spectrum.File", name = "psms")
```

```
  |
  |                                                                      |   0%
  |
  |===                                                                   |   4%
  |
  |=====                                                                 |   7%
  |
  |========                                                              |  11%
  |
  |==========                                                            |  15%
  |
  |=============                                                         |  19%
  |
  |================                                                      |  22%
  |
  |==================                                                    |  26%
  |
  |=====================                                                 |  30%
  |
  |========================                                              |  33%
  |
  |============================                                          |  37%
  |
  |============================                                          |  41%
  |
  |================================                                      |  44%
  |
  |===================================                                   |  48%
  |
```

```
  |=================================                                     |  52%
  |
  |==================================                                    |  56%
  |
  |====================================                                  |  59%
  |
  |======================================                                |  63%
  |
  |=======================================                               |  67%
  |
  |=========================================                             |  70%
  |
  |==========================================                            |  74%
  |
  |============================================                          |  78%
  |
  |==============================================                        |  81%
  |
  |================================================                      |  85%
  |
  |==================================================                    |  89%
  |
  |====================================================                  |  93%
  |
  |======================================================                |  96%
  |
  |========================================================================| 100%
```

```
names(mouse) <- sub("^.*(Mouse.*ACN).*raw", "\\1", names(mouse))
```

Data preprocessing

We use the same data processing workflow as for the spike-in study in the TMT lecture: median of ratios normalisation and maxLFQ summarisation, both performed run by run.

```
inputNames <- names(mouse)
logNames <- paste0(inputNames, "_log")
normNames <- paste0(inputNames, "_norm")
summNames <- paste0(inputNames, "_proteins")
## Encode missing values
mouse <- zeroIsNA(mouse, inputNames)
```

```r
## Log transformation
mouse <- logTransform(mouse, inputNames, name = logNames, base = 2)
## Sample filtering
mouse <- subsetByColData(
    mouse, mouse$Condition != "Norm" & mouse$Condition != "Long_M"
)
## PSM filtering
mouse <- filterFeatures(
    mouse, ~ Protein.Accessions != "" & ## Remove failed protein inference
        !grepl(";", Protein.Accessions)) ## Remove protein groups
for (i in logNames) { ## Remove duplicated PSMs
    rowdata <- rowData(mouse[[i]])
    rowdata$ionID <- paste0(rowdata$Annotated.Sequence, rowdata$Charge)
    rowdata$rowSums <- rowSums(assay(mouse[[i]]), na.rm = TRUE)
    rowdata <- data.frame(rowdata) |>
        group_by(ionID) |>
        mutate(psmRank = rank(-rowSums))
    rowData(mouse[[i]]) <- DataFrame(rowdata)
}
mouse <- filterFeatures(mouse, ~ psmRank == 1, keep = TRUE)
mouse <- filterNA(mouse, logNames, pNA = 0.7) ## Remove highly missing PSMs
## Normalisation: median of ratios, run by run
for (k in seq_along(logNames)) {
    mouse <- sweep(
        mouse,
        MARGIN = 2,
        STATS = nfLogMedianOfRatios(mouse, i = logNames[k]),
        i = logNames[k],
        name = normNames[k]
    )
}
## Protein summarisation: maxLFQ, run by run
mouse <- aggregateFeatures(
    mouse, i = normNames, name = summNames,
    fcol = "Protein.Accessions",
    fun = function(X) iq::maxLFQ(X)$estimate
)
```

```
  |
  |                                                                      |   0%
  |
```

```
|===                                                                   |   4%
|
|=====                                                                 |   7%
|
|========                                                              |  11%
|
|==========                                                            |  15%
|
|=============                                                         |  19%
|
|================                                                      |  22%
|
|==================                                                    |  26%
|
|=====================                                                 |  30%
|
|=========================                                             |  33%
|
|============================                                          |  37%
|
|==============================                                        |  41%
|
|================================                                      |  44%
|
|==================================                                    |  48%
|
|====================================                                  |  52%
|
|=======================================                               |  56%
|
|=========================================                             |  59%
|
|============================================                          |  63%
|
|==============================================                        |  67%
|
|================================================                      |  70%
|
|==================================================                    |  74%
|
|====================================================                  |  78%
|
|======================================================                |  81%
```

```
  |
  |=============================================================          |  85%
  |
  |==============================================================         |  89%
  |
  |================================================================       |  93%
  |
  |==================================================================     |  96%
  |
  |====================================================================== | 100%
```

```
## Join sets
mouse <- joinAssays(mouse, summNames, "proteins")
```

Why do the normalisation and the summarisation have to be performed run by run?

- In a TMT experiment, the samples of a mixture are labelled, pooled and measured together. Each MS run therefore contains all labels (samples) of one mixture, but only for one fraction. Within a run, every PSM is quantified in all labels from the same spectrum, so intensities are directly comparable between the samples of that run.
- Between runs, this no longer holds. Each run measures a different fraction of a different mixture, so the runs contain different sets of peptides and PSMs, with different loading, ionisation and identification. The run effect is the strongest source of variation in the data (see the MDS plots below).
- **Normalisation**: the median of ratios normalisation factors are computed within each run, across the labels that share the same PSMs. This corrects for differences in the amount of sample material between the labels. If we normalised across runs, the factors would be computed on PSMs that were measured in different runs and fractions, so differences in peptide composition between fractions would be confused with differences in sample loading.
- **Summarisation**: within a run, all labels are quantified by exactly the same PSMs, so the protein summary of every sample is built from the same features and is comparable between samples. A protein is quantified by different PSMs in different runs (fractions), so summarising across runs would mix PSM-specific effects with run effects.
- The remaining differences between runs are not removed by the preprocessing. Instead, they are modelled explicitly in the statistical model with a random effect for run. As a consequence, every mouse sample appears once in every run of its mixture (9 times) in the protein-level data, which we account for with a random effect for mouse.

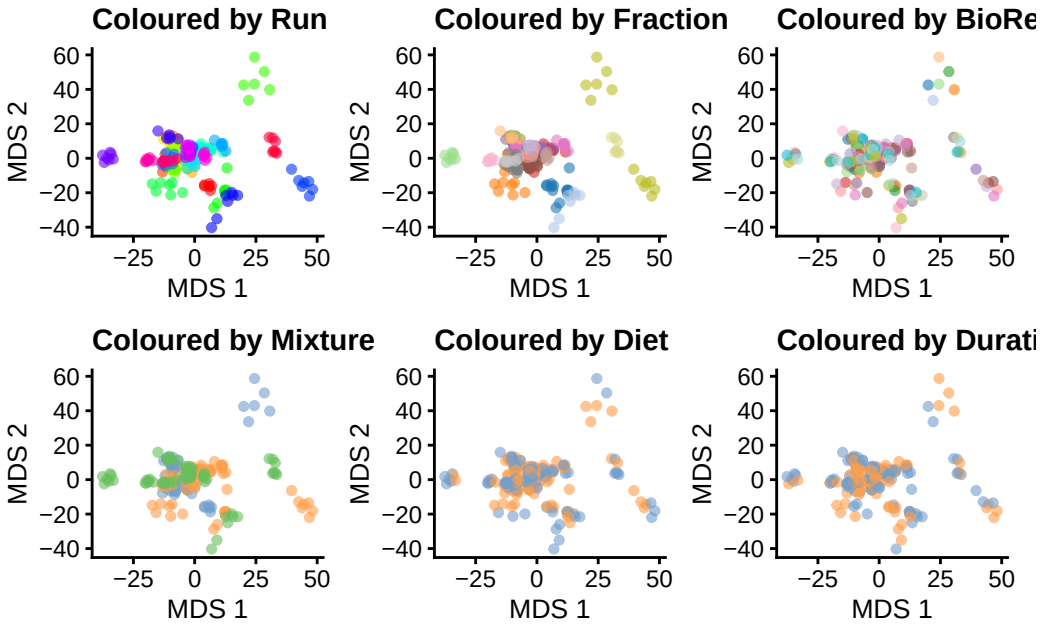
Data exploration and modelling

Let's explore the data.

```
library("scater")
se <- getWithColData(mouse, "proteins") |>
  as("SingleCellExperiment") |>
  runMDS(exprs_values = 1)
```

We plot the MDS and colour each sample based on different potential sources of variation.

```
plotMDS(se, colour_by = "Run") + ggtitle("Coloured by Run") +
  scale_colour_manual(values = rainbow(27)) +
  plotMDS(se, colour_by = "Fraction") + ggtitle("Coloured by Fraction") +
  plotMDS(se, colour_by = "BioReplicate") + ggtitle("Coloured by BioReplicate") +
  plotMDS(se, colour_by = "Mixture") + ggtitle("Coloured by Mixture") +
  plotMDS(se, colour_by = "Diet") + ggtitle("Coloured by Diet") +
  plotMDS(se, colour_by = "Duration") + ggtitle("Coloured by Duration") &
  theme(legend.position = "none")
```



The data exploration leads to several observations:

- The strongest source of variation is associated with the MS acquisition run.

- Part of this run effect is influenced by which fraction it contains since samples from the same fraction tend to be closer than samples from different fractions.
- It is difficult to identify an effect of mouse (biological replicate) because every run contains distinct mice. However, this does not exclude an effect of mice which has been identified as a potential source of variation and hence should still be modelled.
- There is potentially also an effect of TMT mixture since samples from the same mixtures tend to cluster together (in the center of the plot). However, this effect are more subtle to detect and difficult to disentangle from the run and fraction effects.
- Although again very subtle, we can see within each run that samples from the mice with the same diet tends to group together. However, these effects are overwhelmed by the run effects. An effect of duration is to subtle to pinpoint from the current data exploration.

Data modelling disentangles the different sources of variation, given they are properly defined in the model.

Proteomics data contain several sources of variation that need to be accounted for by the model:

1. **Treatment of interest**: we model the source of variation induced by the experimental treatment of interest as a **fixed effect**. Fixed effects are effect that are considered non-random, i.e. the treatment effect is assumed to be the same and reproducible across repeated experiments, but it is unknown and has to be estimated. We will include `Diet` as a fixed effect that models the fact that a change in diet type can induce changes in protein abundance. Similarly, we also include `Duration` as a fixed effect to model the change in protein abundance induces by the diet duration. Finally, we will also include an interaction between the two variables allowing that the changes in protein abundance induced by diet type can be different whether the mice were fed for a short or long duration.

2. **Biological replicate effect**: the experiment involves biological replication as the adipose tissue extracts were sampled from 20 mice (5 mice per Diet x Duration combination). Replicate-specific effects occurs due to uncontrollable factors, such as social behavior, feeding behavior, physical activity, occasional injury,… Two mice will never provide exactly the same sample material, even were they genetically identical and manipulated identically. These effects are typically modelled as random effects which are considered as a random sample from the population of all possible mice and are assumed to be i.i.d normally distributed with mean 0 and constant variance, $b_{\text{mouse}} \sim N(0, \sigma^2_{\text{mouse}})$. The use of random effects thus models the correlation in the data, explicitly. We expect that intensities from the same mouse are more alike than intensities between mice. Note that every mouse corresponds to one sample, i.e. one label in one mixture, that has been measured in 9 runs (one for each fraction). The mouse effect thus plays the same role as the sample effect `(1 | Mixture:Label)` in the spike-in study: without it, the 9 runs of a mouse would be treated as independent samples.

```
length(unique(mouse$BioReplicate))
```

```
[1] 20
```

3. **Labelling effects**: the 20 mouse adipose tissue samples have been labelled using 18-plex TMT. We can expect that samples measured within the same TMT label may be more similar than samples measured within different TMT labels. Since these effects may not be reproducible from one experiment to another, for instance because each TMT kit may potentially contain different impurity ratios, we can account for this correlation using a random effect for TMT label.

```
length(unique(mouse$Label))
```

```
[1] 10
```

4. **Mixture effects**: the 20 mouse samples were assigned to one out of 3 mixtures. Again, we expect protein intensities from the same mixture will be more alike than those of different mixtures. Hence, we will add a random effect for mixture.

```
table(mouse$Mixture)
```

```
 PAMI-176_Mouse_A-J  PAMI-176_Mouse_K-T PAMI-194_Mouse_U-Dd
                 54                  63                  63
```

5. **Run effects**: protein intensities that are measured within the same run will be more similar than protein intensities between runs. We will use a random effect for run to explicitly model this correlation in the data. Note that each sample has been acquired in 9 fractions, each fraction being measured in a separate run. Accounting for the effects of run will also absorb the effects of fraction.

```
length(unique(mouse$Run))
```

```
[1] 27
```

We will model the main effects for `Diet` and `Duration`, and a `Diet:Duration` interaction, to account for proteins for which the `Diet` effect changes according to `Duration`, and vice versa, which can be written as `Diet + Duration + Diet:Duration`, shortened into `Diet * Duration` (recall the heart example in a previous course). Adding the technical sources of variation, the model becomes.

```
model <- ~ Diet * Duration + ## (1) fixed effect for Diet and Duration with interaction
  (1 | BioReplicate) +  ## (2) random effect for biological replicate (mouse)
  (1 | Label) + ## (3) random effect for label
  (1 | Mixture) + ## (4) random effect for mixture
  (1 | Run) ## (5) random effect for MS run
```

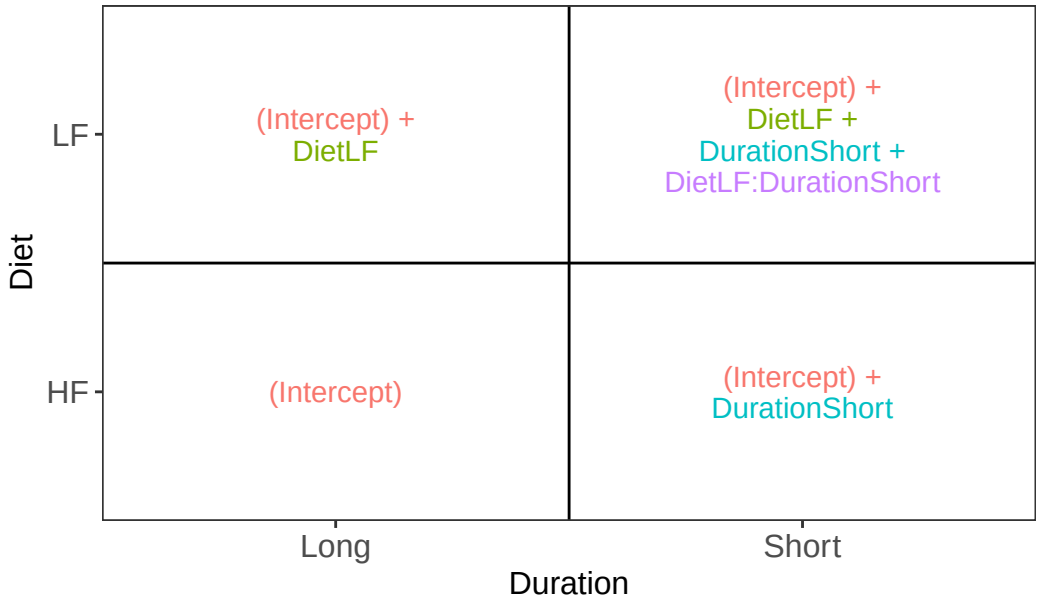
We estimate the model with `msqrob()`.

```
mouse <- msqrob(mouse, i = "proteins", formula = model)
```

Difference between low fat and high fat diet after short duration

We need to convert this question in a combination of the model parameters. We guide the contrast definition using the `ExploreModelMatrix` package. Since we are not interested in technical effects, we will only focus on the variables of interest, here `Diet * Duration`.

```
library("ExploreModelMatrix")
vd <- VisualizeDesign(
    sampleData =  colData(mouse),
    designFormula = ~ Diet * Duration,
    textSizeFitted = 4
)
vd$plotlist[[1]]
```



Assessing the difference between low-fat and high-fat diets for short duration boils down to assessing the difference between the `Short_LF` and `Short_HF`. The mean for the short low-fat diet group is defined by `(Intercept) + DietLF + DurationShort + DietLF:DurationShort`. The mean for the short high-fat diet group is defined by `(Intercept) + DurationShort`. The difference between the two results in the contrast below:

```
(L <- makeContrast(
    "DietLF + DietLF:DurationShort = 0",
    parameterNames = c("DietLF", "DietLF:DurationShort")
))
```

```
                     DietLF + DietLF:DurationShort
DietLF                                           1
DietLF:DurationShort                             1
```

```
mouse <- hypothesisTest(mouse, i = "proteins", L)
```

Let us retrieve the result table from the `rowData`. Note that the model column is named after the column names of the contrast matrix `L`.

```
inference <- rowData(mouse[["proteins"]])[[colnames(L)]]
head(inference)
```

```
             logFC        se        df          t        pval    adjPval
P02468 -0.03195188 0.1497508 127.77518 -0.2133670 0.831380793 0.92045731
Q61598 -0.35037677 0.1465970 127.47139 -2.3900677 0.018309085 0.11074764
P08003  0.18638918 0.4689975 121.63120  0.3974204 0.691754015 0.90769839
Q03265  0.35558692 0.1977116 125.12751  1.7985129 0.074506510 0.28427099
Q62000  0.72872636 0.2395232  89.11931  3.0424042 0.003083212 0.03475621
P17182  0.03465565 0.1434052 130.10451  0.2416625 0.809422168 0.91794950
```

The table contains the hypothesis testing results for every protein. We report the differentially abundant proteins at the 5% FDR level in a table, sorted by their p-value. We define a small helper function so that we can reuse it for the other research questions.

```
sigTable <- function(inference, alpha = 0.05) {
  inference |>
    as.data.frame() |>
    tibble::rownames_to_column("protein") |>
    filter(adjPval < alpha) |>
```

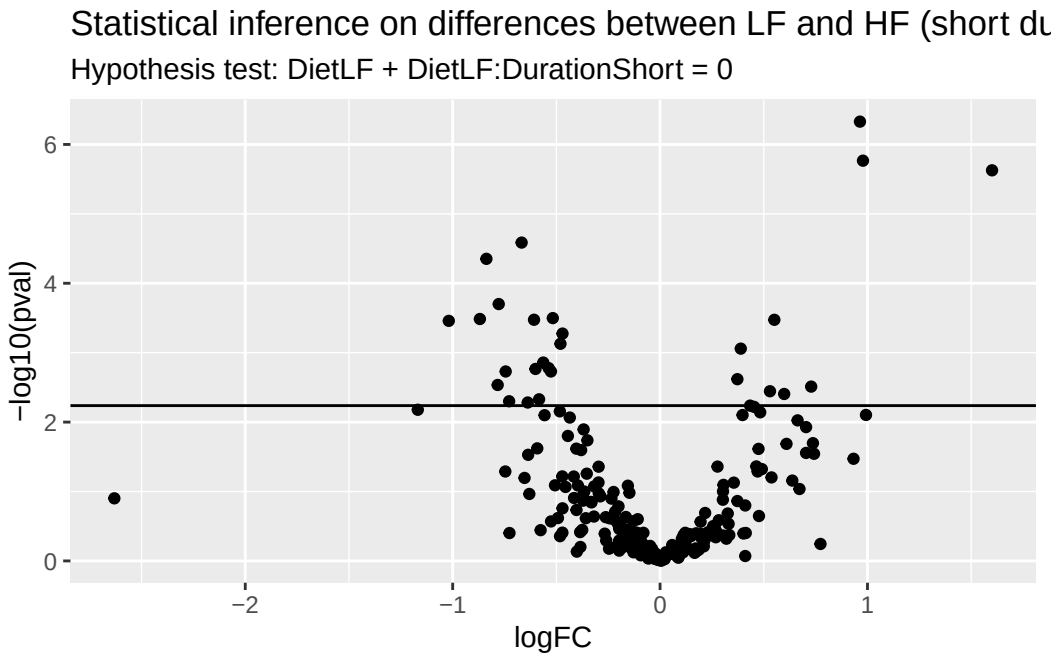
```
    arrange(pval) |>
    knitr::kable(digits = 4)
}
sigTable(inference)
```

protein	logFC	se	df	t	pval	adjPval
O08528	0.9638	0.1774	89.6527	5.4341	0.0000	0.0001
P47738	0.9784	0.1933	108.5100	5.0610	0.0000	0.0002
Q5SWU9	1.6007	0.3234	124.9618	4.9489	0.0000	0.0002
Q61233	-0.6675	0.1526	120.7108	-4.3750	0.0000	0.0016
O54931	-0.8373	0.1968	108.3551	-4.2557	0.0000	0.0022
Q9Z1T1	-0.7779	0.1878	35.8608	-4.1430	0.0002	0.0078
P10126	-0.5171	0.1393	116.2325	-3.7115	0.0003	0.0078
Q3TWL2	-0.8688	0.2067	23.4635	-4.2037	0.0003	0.0078
Q3TIV5	-0.6077	0.1599	61.0581	-3.8007	0.0003	0.0078
Q8K2B3	0.5509	0.1479	92.5480	3.7247	0.0003	0.0078
Q6P9Q4	-1.0187	0.2488	26.8479	-4.0941	0.0003	0.0078
Q9CR00	-0.4705	0.1301	76.7912	-3.6172	0.0005	0.0110
Q9WVA4	-0.4802	0.1378	97.0439	-3.4836	0.0007	0.0142
Q8R164	0.3893	0.1118	67.5952	3.4829	0.0009	0.0155
Q9D1M0	-0.5631	0.1699	79.2198	-3.3142	0.0014	0.0229
O89079	-0.5373	0.1637	64.1541	-3.2819	0.0017	0.0244
Q99K85	-0.6007	0.1842	70.0912	-3.2611	0.0017	0.0244
Q9D832	-0.5262	0.1570	36.9266	-3.3514	0.0019	0.0244
O88456	-0.7443	0.2170	28.4068	-3.4299	0.0019	0.0244
P28271	0.3728	0.1202	119.6760	3.1007	0.0024	0.0299
Q61792	-0.7830	0.2568	103.1546	-3.0488	0.0029	0.0345
Q62000	0.7287	0.2395	89.1193	3.0424	0.0031	0.0348
Q60739	0.5299	0.1727	46.4263	3.0679	0.0036	0.0387
O08586	0.5985	0.1804	17.5730	3.3169	0.0039	0.0407
Q8C1B7	-0.5833	0.1991	63.5916	-2.9305	0.0047	0.0466
Q8BUR4	-0.7277	0.2327	21.4505	-3.1266	0.0050	0.0478
P63001	-0.6380	0.2126	31.4756	-3.0015	0.0052	0.0480

We can use the table above directly to build a volcano plot using `ggplot2` functionality.

```
ggplot(inference) +
    aes(x = logFC, y = -log10(pval)) +
    geom_hline(yintercept = -log10(inference$pval[which.min(abs(inference$adjPval - 0.05))] )
    geom_point() +
```

```
    ggtitle("Statistical inference on differences between LF and HF (short duration)",
            paste("Hypothesis test:", colnames(L), "= 0"))
```



In this example (remember this is a subset of the complete data set), only a few proteins pass the significance threshold of 5% FDR.

Difference between low fat and high fat diet after long duration

Following the same approach as above, the hypothesis test becomes:

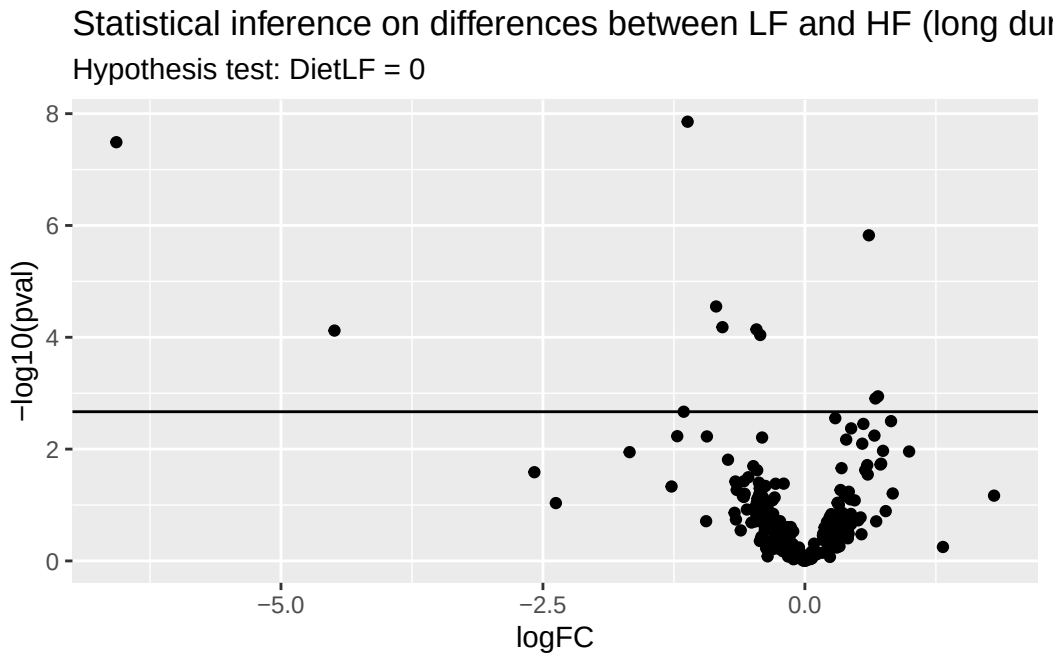
```
L <- makeContrast("DietLF = 0", parameterNames = "DietLF")
mouse <- hypothesisTest(mouse, i = "proteins", L)
inference <- rowData(mouse[["proteins"]])[[colnames(L)]]
sigTable(inference)
```

protein	logFC	se	df	t	pval	adjPval
P61924	-1.1192	0.1720	63.5456	-6.5063	0.0000	0.0000
Q924P3	-6.5712	0.6115	14.2327	-10.7455	0.0000	0.0000
Q8R164	0.6107	0.1157	67.5952	5.2767	0.0000	0.0001
O89079	-0.8472	0.1878	64.1541	-4.5113	0.0000	0.0017
Q920E5	-0.7868	0.1792	45.9298	-4.3901	0.0001	0.0027
Q76MZ3	-0.4623	0.1096	70.5912	-4.2169	0.0001	0.0027

protein	logFC	se	df	t	pval	adjPval
Q8CJ12	-4.4895	0.8944	19.0222	-5.0193	0.0001	0.0027
Q9WUA2	-0.4263	0.1014	58.4971	-4.2041	0.0001	0.0028
P32261	0.6978	0.2090	113.1754	3.3393	0.0011	0.0308
Q8K1Z0	0.6733	0.1914	34.4777	3.5176	0.0012	0.0308
O35598	-1.1563	0.3276	19.6868	-3.5298	0.0021	0.0484

And we plot the results.

```
ggplot(inference) +
    aes(x = logFC, y = -log10(pval)) +
    geom_hline(yintercept = -log10(inference$pval[which.min(abs(inference$adjPval - 0.05))] )
    geom_point() +
    ggtitle("Statistical inference on differences between LF and HF (long duration)",
            paste("Hypothesis test:", colnames(L), "= 0"))
```



Again, only a few proteins come out differentially abundant between the two diets, but after a long diet duration.

Average difference between low fat and high fat diet

One may want to identify the set of proteins that are systematically differentially abundant between diets, irrespective of the duration. To answer this question, we want to infer on the average

difference between group `LF` and group `HF`. The average low-fat diet is defined by `((Intercept) + DietLF + DurationShort + DietLF:DurationShort + (Intercept) + DietLF)/2`. The average high-fat diet group is defined by `((Intercept) + DurationShort + (Intercept))/2`. The difference between the two results in the hypothesis test below:

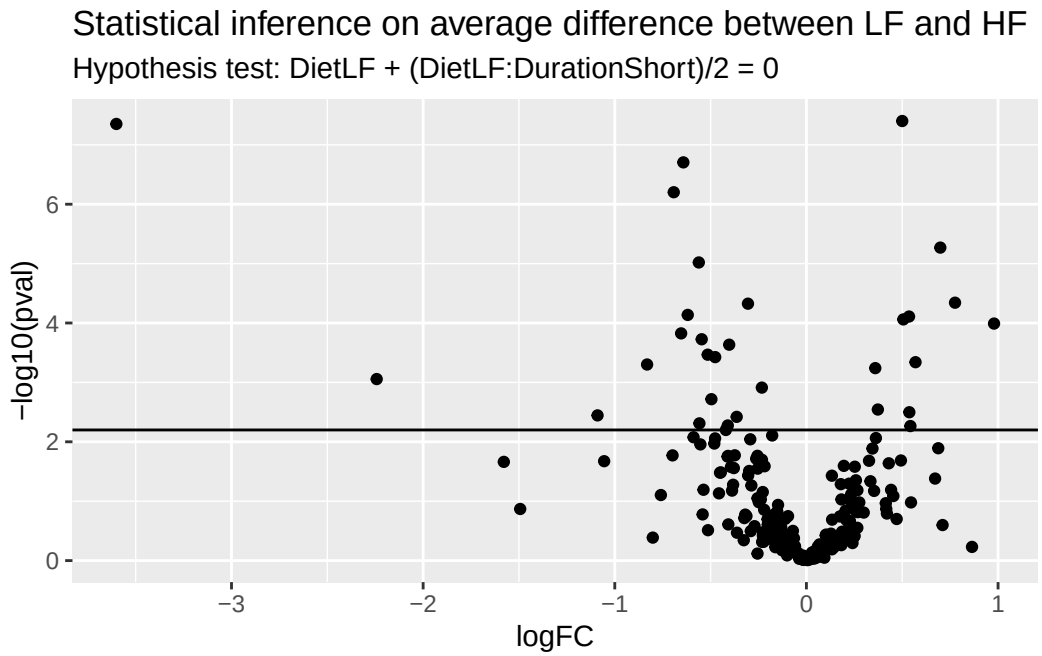
```
L <- makeContrast(
    "DietLF + (DietLF:DurationShort)/2 = 0",
    parameterNames = c("DietLF", "DietLF:DurationShort")
)
mouse <- hypothesisTest(mouse, i = "proteins", L)
inference <- rowData(mouse[["proteins"]])[[colnames(L)]]
sigTable(inference)
```

protein	logFC	se	df	t	pval	adjPval
Q8R164	0.5000	0.0807	67.5952	6.1929	0.0000	0.0000
Q924P3	-3.6008	0.3436	14.2327	-10.4803	0.0000	0.0000
P61924	-0.6421	0.1100	63.5456	-5.8349	0.0000	0.0000
O89079	-0.6923	0.1252	64.1541	-5.5306	0.0000	0.0000
P47738	0.6988	0.1460	108.5100	4.7874	0.0000	0.0003
Q61233	-0.5614	0.1214	120.7108	-4.6230	0.0000	0.0004
Q62000	0.7755	0.1808	89.1193	4.2884	0.0000	0.0015
Q76MZ3	-0.3050	0.0703	70.5912	-4.3361	0.0000	0.0015
O54931	-0.6192	0.1501	108.3551	-4.1246	0.0001	0.0019
Q8K1Z0	0.5355	0.1195	34.4777	4.4822	0.0001	0.0019
P07759	0.5058	0.1237	101.8980	4.0880	0.0001	0.0020
Q5SWU9	0.9791	0.2439	124.9618	4.0139	0.0001	0.0021
Q3TWL2	-0.6537	0.1448	23.4635	-4.5150	0.0001	0.0029
Q99K85	-0.5464	0.1385	70.0912	-3.9439	0.0002	0.0033
P10126	-0.4029	0.1060	116.2325	-3.8000	0.0002	0.0038
Q3TIV5	-0.5151	0.1357	61.0581	-3.7951	0.0003	0.0053
Q9D1M0	-0.4760	0.1281	79.2198	-3.7162	0.0004	0.0055
Q8BH35	0.5693	0.1564	89.5586	3.6394	0.0005	0.0063
Q8BUR4	-0.8314	0.2030	21.4505	-4.0947	0.0005	0.0065
Q9R069	0.3593	0.1006	88.8423	3.5718	0.0006	0.0071
Q8CJ12	-2.2423	0.5692	19.0222	-3.9391	0.0009	0.0104
Q9WUA2	-0.2325	0.0684	58.4971	-3.3991	0.0012	0.0138
O54774	-0.4957	0.1542	76.5716	-3.2139	0.0019	0.0207
Q8CI59	0.3728	0.1173	39.6654	3.1794	0.0029	0.0296
O08586	0.5370	0.1573	17.5730	3.4143	0.0032	0.0315
Q99J45	-1.0898	0.3314	20.4802	-3.2882	0.0036	0.0342
Q920E5	-0.3638	0.1193	45.9298	-3.0492	0.0038	0.0349

protein	logFC	se	df	t	pval	adjPval
Q61792	-0.5583	0.1941	103.1546	-2.8760	0.0049	0.0433
Q9Z1T1	-0.4103	0.1382	35.8608	-2.9683	0.0053	0.0449
Q9Z1X4	0.5426	0.1836	36.7539	2.9547	0.0054	0.0449

And we plot the results.

```
ggplot(inference) +
    aes(x = logFC, y = -log10(pval)) +
    geom_hline(yintercept = -log10(inference$pval[which.min(abs(inference$adjPval - 0.05))] )
    geom_point() +
    ggtitle("Statistical inference on average difference between LF and HF",
            paste("Hypothesis test:", colnames(L), "= 0"))
```



Interaction: does the diet effect change according to duration?

We will now explore whether the effect of diet on protein abundance may be affected by duration, i.e. we want to infer on the difference of differences. The difference between hypothesis 1 and 2 is `(DietLF + DietLF:DurationShort) - (DietLF)` and results in the hypothesis below:

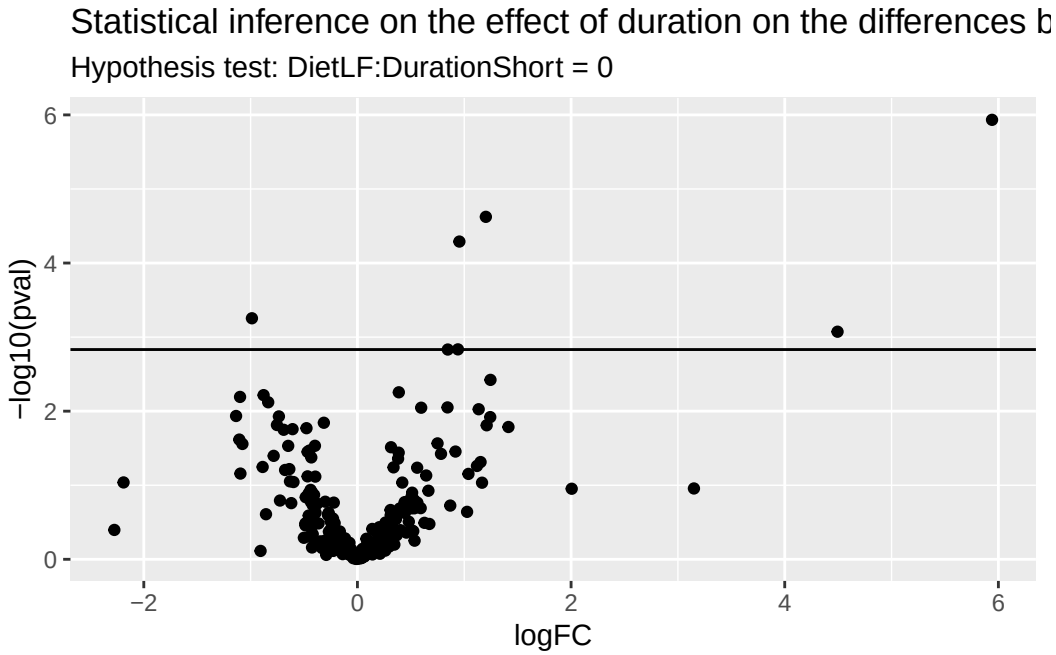
We can proceed with the same statistical pipeline.

```
L <- makeContrast(
    "DietLF:DurationShort = 0",
    parameterNames = "DietLF:DurationShort"
)
mouse <- hypothesisTest(mouse, i = "proteins", L)
inference <- rowData(mouse[["proteins"]])[[colnames(L)]]
sigTable(inference)
```

protein	logFC	se	df	t	pval	adjPval
Q924P3	5.9407	0.7394	14.2327	8.0348	0e+00	0.0003
O08528	1.2022	0.2696	89.6527	4.4589	0e+00	0.0029
P61924	0.9543	0.2197	63.5456	4.3444	1e-04	0.0042
P32261	-0.9876	0.2780	113.1754	-3.5528	6e-04	0.0345
Q8CJ12	4.4944	1.1361	19.0222	3.9561	8e-04	0.0419

And we plot the results.

```
ggplot(inference) +
    aes(x = logFC, y = -log10(pval)) +
    geom_hline(yintercept = -log10(inference$pval[which.min(abs(inference$adjPval - 0.05))] )
    geom_point() +
    ggtitle("Statistical inference on the effect of duration on the differences between diets
            paste("Hypothesis test:", colnames(L), "= 0"))
```



Session Info

```
sessionInfo()
```

```
R version 4.6.1 (2026-06-24)
Platform: aarch64-apple-darwin23
Running under: macOS Tahoe 26.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.6/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.6/Resources/lib/libRlapack.dylib;  LAPACK

locale:
[1] nl_BE.UTF-8/nl_BE.UTF-8/nl_BE.UTF-8/C/nl_BE.UTF-8/nl_BE.UTF-8

time zone: Europe/Brussels
tzcode source: internal

attached base packages:
[1] stats4    stats     graphics  grDevices utils     datasets  methods
```

```
[8] base

other attached packages:
 [1] ExploreModelMatrix_1.24.0   scater_1.40.1
 [3] scuttle_1.22.0              SingleCellExperiment_1.34.0
 [5] BiocFileCache_3.2.0         dbplyr_2.6.0
 [7] BiocParallel_1.46.0         patchwork_1.3.2
 [9] ggplot2_4.0.3               dplyr_1.2.1
[11] msqrob2_1.21.1              QFeatures_1.22.0
[13] MultiAssayExperiment_1.38.0 SummarizedExperiment_1.42.0
[15] Biobase_2.72.0              GenomicRanges_1.64.0
[17] Seqinfo_1.2.0               IRanges_2.46.0
[19] S4Vectors_0.50.1            BiocGenerics_0.58.1
[21] generics_0.1.4              MatrixGenerics_1.24.0
[23] matrixStats_1.5.0

loaded via a namespace (and not attached):
  [1] RColorBrewer_1.1-3      jsonlite_2.0.0          magrittr_2.0.5
  [4] estimability_2.0.0      ggbeeswarm_0.7.3        farver_2.1.2
  [7] nloptr_2.2.1            rmarkdown_2.31          vctrs_0.7.3
 [10] memoise_2.0.1           minqa_1.2.8             htmltools_0.5.9
 [13] S4Arrays_1.12.0         BiocBaseUtils_1.14.2    curl_7.1.0
 [16] BiocNeighbors_2.6.0     SparseArray_1.12.2      htmlwidgets_1.6.4
 [19] plyr_1.8.9              httr2_1.2.3             cachem_1.1.0
 [22] igraph_2.3.3            mime_0.13               lifecycle_1.0.5
 [25] pkgconfig_2.0.3         rsvd_1.0.5              Matrix_1.7-5
 [28] R6_2.6.1                fastmap_1.2.0           rbibutils_2.4.1
 [31] shiny_1.14.0            clue_0.3-68             digest_0.6.39
 [34] irlba_2.3.7             RSQLite_3.53.2          beachmat_2.28.0
 [37] filelock_1.0.3          labeling_0.4.3          abind_1.4-8
 [40] compiler_4.6.1          bit64_4.8.2             withr_3.0.3
 [43] S7_0.2.2                viridis_0.6.5           DBI_1.3.0
 [46] MASS_7.3-65             iq_2.0.1                rappdirs_0.3.4
 [49] DelayedArray_0.38.2     tools_4.6.1             vipor_0.4.7
 [52] otel_0.2.0              beeswarm_0.4.0          httpuv_1.6.17
 [55] glue_1.8.1              promises_1.5.0          nlme_3.1-169
 [58] grid_4.6.1              cluster_2.1.8.2         reshape2_1.4.5
 [61] gtable_0.3.6            tidyr_1.3.2             BiocSingular_1.28.0
 [64] ScaledMatrix_1.20.0     XVector_0.52.0          ggrepel_0.9.8
 [67] pillar_1.11.1           stringr_1.6.0           limma_3.68.4
 [70] later_1.4.8             rintrojs_0.3.4          splines_4.6.1
 [73] lattice_0.22-9          bit_4.6.0               tidyselect_1.2.1
 [76] knitr_1.51              reformulas_0.4.4        gridExtra_2.3.1
```

```
 [79] ProtGenerics_1.44.0     shinydashboard_0.7.3    xfun_0.59
 [82] statmod_1.5.2           DT_0.34.0               stringi_1.8.7
 [85] lazyeval_0.2.3          yaml_2.3.12             boot_1.3-32
 [88] evaluate_1.0.5          codetools_0.2-20        MsCoreUtils_1.24.0
 [91] tibble_3.3.1            cli_3.6.6               xtable_1.8-8
 [94] Rdpack_2.6.6            Rcpp_1.1.1-1.1          png_0.1-9
 [97] parallel_4.6.1          assertthat_0.2.1        blob_1.3.0
[100] AnnotationFilter_1.36.0 lme4_2.0-1              viridisLite_0.4.3
[103] scales_1.4.0            purrr_1.2.2             rlang_1.2.0
[106] cowplot_1.2.0           shinyjs_2.1.1
```

References

Huang, Ting, Meena Choi, Manuel Tzouros, et al. 2020. "MSstatsTMT: Statistical Detection of Differentially Abundant Proteins in Experiments with Isobaric Labeling and Multiple Mixtures." *Mol. Cell. Proteomics* 19 (10): 1706–23.

Plubell, Deanna L, Phillip A Wilmarth, Yuqi Zhao, et al. 2017. "Extended Multiplexing of Tandem Mass Tags (TMT) Labeling Reveals Age and High Fat Diet Specific Proteome Changes in Mouse Epididymal Adipose Tissue." *Mol. Cell. Proteomics* 16 (5): 873–90.