Introduction to Linear Models

FGCZ Protein Informatics Training

Witold Wolski wew@fgcz.ethz.ch
08 March, 2021

linear models (lm)

Overview

- What is a linear model
- How to estimate coefficients
- What are contrasts
- How to determine the error of the coefficients, test statistics and p-values
- What are interactions in linear models
- Example: Yeast data with batches
- limma - Empirical Bayes
- Benchmarking
- Conclusions

lm intro

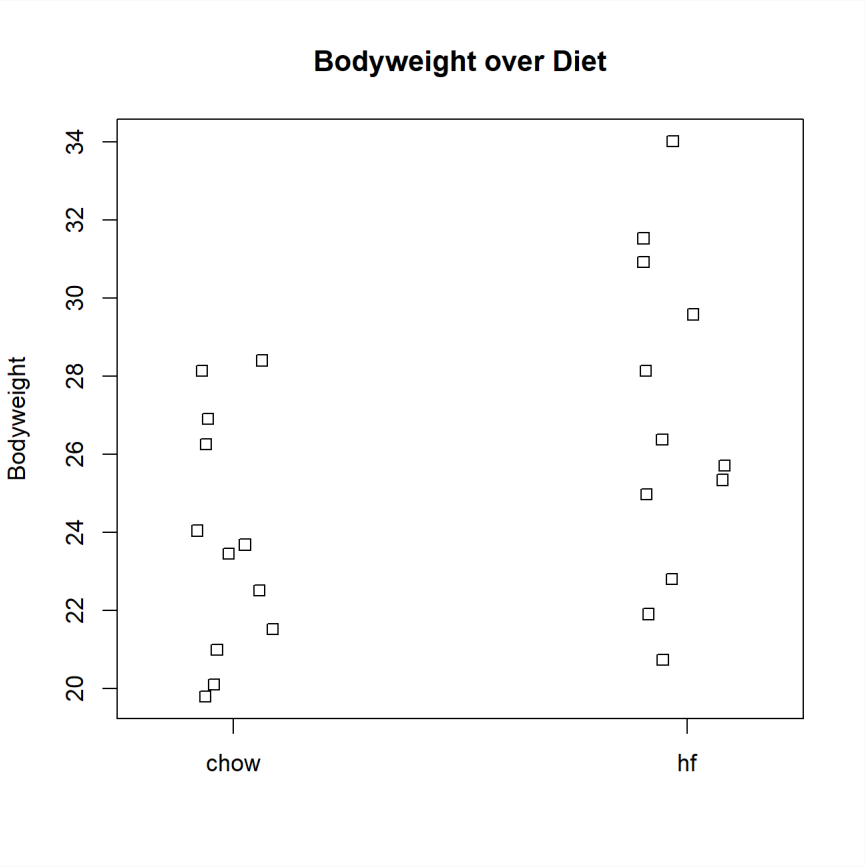
```
dat ← read.csv("femaleMiceWeights.csv")
head(dat)
```

```
##   Diet Bodyweight
## 1 chow      21.51
## 2 chow      28.14
## 3 chow      24.04
## 4 chow      23.45
## 5 chow      23.68
## 6 chow      19.79
```

```
table(dat$Diet)
```

```
##
## chow   hf
##   12   12
```

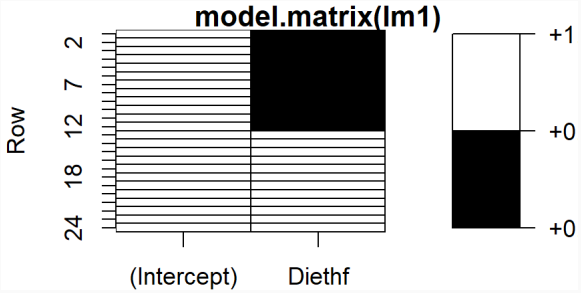
```
stripchart(Bodyweight ~ Diet, # < formula interface
           data= dat,
           vertical=TRUE,
           method="jitter",
           main="Bodyweight over Diet")
```



lm intro

```r
# summarize data
meansum ←
  dat %>%
  group_by(Diet) %>%
  summarise(mean = mean(Bodyweight))
# linear model
lm1 ← lm(Bodyweight ~ Diet,
         data = dat)
coefs ← coef(lm1)
```

$$y = b_0 X_0 + b_1 X_1 + \epsilon$$



group means

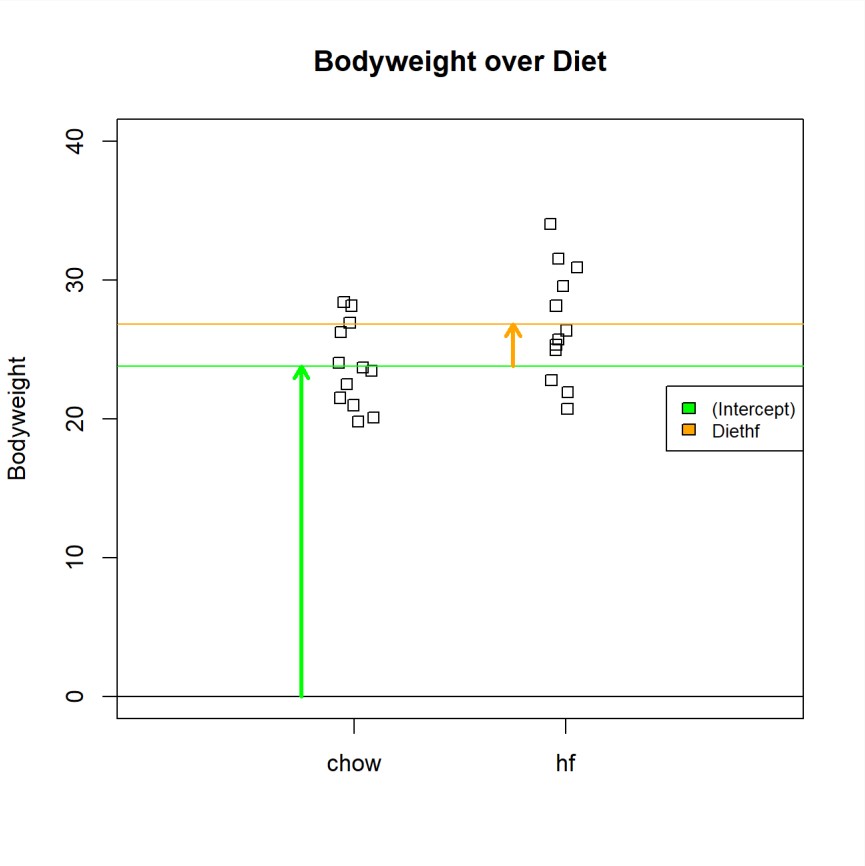
Diet	mean
chow	23.81333
hf	26.83417

coefficients

	x
(Intercept)	23.813333
Diethf	3.020833

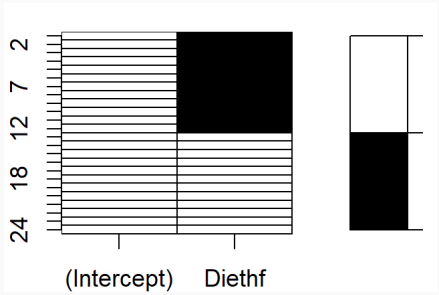
lm intro - examin the coefficients

```
stripchart(Bodyweight ~ Diet,
           data = dat , vertical=TRUE,
           method="jitter",
           main="Bodyweight over Diet",
           ylim=c(0,40), xlim=c(0,3))
a ← -0.25; lgth ← .1
abline(h=0)
arrows(1+a,0,1+a,coefs[1],lwd=3,
       col="green",length=lgth)
abline(h=coefs[1],col="green")
arrows(2+a,coefs[1],2+a,coefs[1]+coefs[2],
       lwd=3,col="orange",length=lgth)
abline(h=coefs[1]+coefs[2],col="orange")
legend("right",names(coefs),
       fill=c("green","orange"),
       cex=.75,bg="white")
```



lm intro - determining the coefficients

```
Y ← dat$Bodyweight
X ← model.matrix(lm1)
par(mar = c(2,2,1,1))
plot(X, col=c("black", "white"), main="")
```




```
beta ← solve(t(X) %*% X) %*% (t(X) %*% Y)
epsilon ← Y - t(beta) %*% t(X)
beta
```

```
##                 [,1]
## (Intercept) 23.813333
## Diethf       3.020833
```

$$\beta = (X^T X)^{-1}(X^T Y)$$

β minimizes

$$\sum (Y - \beta X)^2 = (Y - \beta X)(Y - \beta X)^T$$

.

predicting Y

$$\hat{Y} = X\beta = b_0 X_0 + b_1 X_1$$

residues

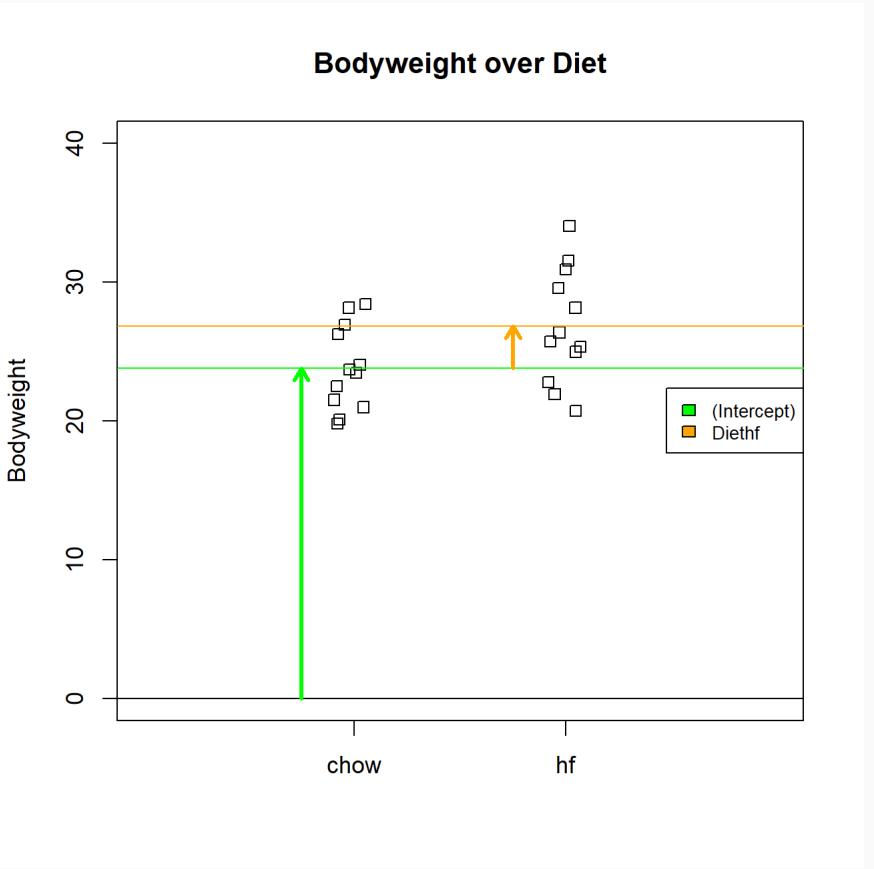
$$e = Y - X\beta$$

lm intro - contrasts

```
linfct ← rbind(
  chow = c(1, 0),
  hf = c(1, 1 )
  )
linfct %*% coef(lm1)
```

```
##          [,1]
## chow 23.81333
## hf   26.83417
```

$$Y_{chow} = b_0 \cdot 1 + b_1 \cdot 0$$
$$Y_{hf} = b_0 \cdot 1 + b_1 \cdot 1$$



lm intro - contrasts

A contrast is a linear combination of variables (parameters or statistics) whose coefficients add up to zero, allowing comparison of different treatments.

$$Y_{chow-hf} = (1) \cdot Y_c + (-1) \cdot Y_h$$

```
contrasts ← rbind(
  "chow - hf" =
    1 * linfct["chow",] + -1 * linfct["hf",]
)
contrasts
```

```
##           [,1] [,2]
## chow - hf    0   -1
```

```
contrasts %*% coef(lm1)
```

```
##                [,1]
## chow - hf -3.020833
```

lm intro - LSE standard error

```
epsilon  ← resid(lm1)
sigma ←
  sqrt(sum(epsilon ^2)/
         (length(epsilon ) -
            length(coef(lm1))) )

X ← model.matrix(lm1)
solve(t(X) %*% X) * sigma^2
```

```
##             (Intercept)    Diethf
## (Intercept)    1.080255 -1.080255
## Diethf        -1.080255  2.160510
```

```
vcov(lm1)
```

```
##             (Intercept)    Diethf
## (Intercept)    1.080255 -1.080255
## Diethf        -1.080255  2.160510
```

$$
\begin{aligned}
var(\hat{\beta}) &= var((X^\top X)^{-1} X^\top Y) \\
&= \cdots \\
&= \sigma^2 (X^\top X)^{-1} \\
with \quad \sigma^2 &= \sum e^2/(n-p)
\end{aligned}
$$

LSE = least square estimate

lm intro - computing the test statistic

```
std.error ← sqrt(diag(
  linfct %*%
  vcov(lm1) %*%
  t(linfct)))
t.statistic ←
  linfct%*%coef(lm1)/std.error
t.statistic
```

```
##          [,1]
## chow 22.91168
## hf   25.81814
```

```
std.error ← sqrt(diag(
  contrasts %*%
    vcov(lm1) %*%
    t(contrasts)))
t.statistic ←
  contrasts %*% coef(lm1) / std.error
t.statistic
```

```
##                [,1]
## chow - hf -2.055174
```

```
head(linfct)
```

```
##       [,1] [,2]
## chow     1    0
## hf       1    1
```

```
head(contrasts)
```

```
##            [,1] [,2]
## chow - hf     0   -1
```

$$t_i = \frac{\beta_i}{se(\beta_i)}$$

lm intro - getting the p-values

```
lfq ←
  prolfqua::my_contrast( lm1,
  rbind(linfct, contrasts)) %>%
  dplyr::select(lhs, estimate,
                std.error,
                statistic, p.value) %>%
  mutate(p.value =
           round(p.value,digits=3))
```

prolfqua

	lhs	estimate	std.error	statistic	p.value
chow	chow	23.81	1.04	22.91	0.00
hf	hf	26.83	1.04	25.82	0.00
chow - hf	chow - hf	-3.02	1.47	-2.06	0.05

lm intro - getting the p-values (adjusted)

```
library(multcomp)
multcomp ← summary(
  glht(lm1, rbind(linfct, contrasts))) %>%
  broom::tidy() %>%
  dplyr::select(contrast, estimate,
                std.error, statistic ,
                adj.p.value)
```

multcomp

contrast	estimate	std.error	statistic	adj.p.value
chow	23.81	1.04	22.91	0.00
hf	26.83	1.04	25.82	0.00
chow - hf	-3.02	1.47	-2.06	0.12

lm intro - interactions

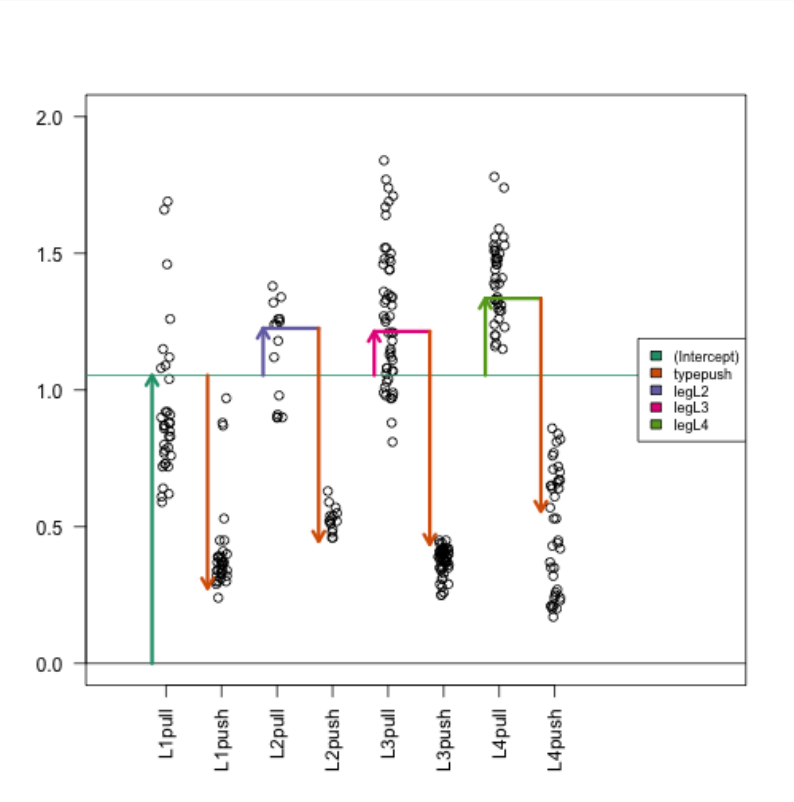
Model without interaction

```
spider ← read.csv("spider_wolff_gorb_2013.csv"
                  , skip = 1)
table(spider$leg, spider$type)
```

```
##
##      pull push
##   L1   34   34
##   L2   15   15
##   L3   52   52
##   L4   40   40
```

```
noI ← lm(friction ~ type + leg, data = spider)
coef(noI)
```

```
## (Intercept)    typepush       legL2       legL3       legL4
##   1.0539153  -0.7790071   0.1719216   0.1604921   0.2813382
```



lm intro - interactions

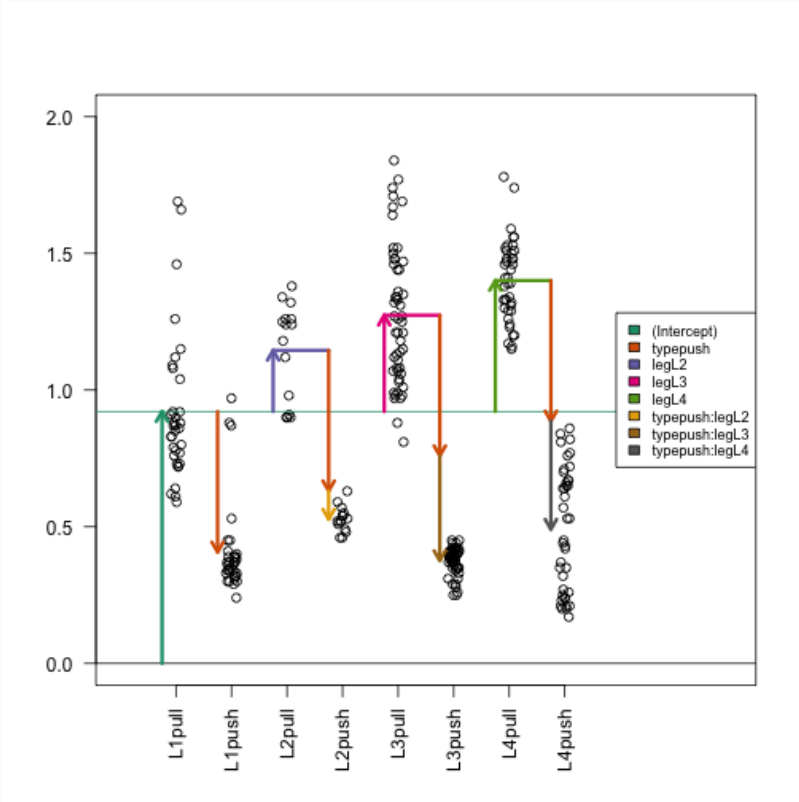
Model with interaction

```
withI ← lm(friction ~ type + leg + type:leg,
           data = spider)
an ← anova(withI)
broom::tidy(an)[1:3,c("term", "p.value")]
```

```
## # A tibble: 3 x 2
##   term       p.value
##   <chr>        <dbl>
## 1 type     2.75e-101
## 2 leg      2.97e- 15
## 3 type:leg 2.26e- 11
```

```
coef(withI)
```

```
##    (Intercept)       typepush          legL2          legL3          legL4 typepush:legL2
##      0.9214706     -0.5141176      0.2238627      0.3523756      0.4792794     -0.1038824
## typepush:legL3 typepush:legL4
##     -0.3837670     -0.3958824
```



Yeast analysis - batches

- Condition : enthanol and glucose
- Batch: p2691 (12 to 16 March 2018) and p2370 (March 2017)

R linear model:

```
lm(normalizedIntensity ~ Condition + Batch + Condition:Batch, data = proteinData)
```

And we are going to compute the following contrasts ($\log_2(FC)$):

$$\begin{aligned}
\text{fc}_{glucose-ethanol} \\
\text{fc}_{p2370-p2691} \\
\text{fc}_{glucose:p2370-ethanol:p2370} \\
\text{fc}_{glucose:p2691-ethanol:p2691} \\
\text{fc}_{interaction} &= \text{fc}_{glucose:p2370-ethanol:p2370} - \text{fc}_{glucose:p2691-ethanol:p2691}
\end{aligned}$$

limma - Empricial Bayes

- In a mass spectrometric LFQ experiment, we measure hundreds of proteins in parallel.
- Hence, these measurements are correlated.
- Also, the analysis has a parallel structure, and we fit the same linear model to all protein.

- Potentially we can transfer information from the measurement of one peptide/protein to the other.
- The empirical Bayes approach is used to improve the test statistic to test the null hypothesis $H_0 : \beta_{pj} = 0$.

"Linear models and empirical bayes methods for assessing differential expression in microarray experiments" Gordon K. Smyth

limma - Empirical Bayes

Define the moderated t-statistic by:

$$\tilde{t}_{pj} = \frac{\hat{\beta}_{pj}}{\tilde{s}_p \sqrt{v_{pj}}}$$

with p protein index j parameter index, v element of the variance covariance matrix, $\hat{\beta}$ model parameter, $\tilde{s}$ posterior standard error.

The posterior values shrink the observed variances towards the prior values with the degree of shrinkage depending on the relative sizes of the observed and prior degrees of freedom.

$$\tilde{s}_p^2 = E(\sigma^2 | s_p^2) = \frac{d_0 s_0^2 + d_p s_p^2}{d_0 + d_p}$$

where d are the degrees of freedom.

This statistic represents a hybrid classical/Bayes approach in which the posterior variance \(\\) has been substituted into the classical t-statistic in place of the usual sample variance.

limma - Empirical bayes

$$\tilde{s}_p^2 = E(\sigma^2|s_p^2) = \frac{d_0 s_0^2 + d_p s_p^2}{d_0 + d_p}$$

For $d_p << d_0$, $\tilde{s} \rightarrow s_0$
For $d_p >> d_0$, $\tilde{s} \rightarrow s_p$

s_0 is the same for all proteins in and experiment.

What happens with the $correlation(T, d)$ for $d_p \rightarrow 0$ sample sizes? Where T is the t-statistics and d is the difference between samples.

Hint: $T \propto d/\tilde{s}$

```
d_0 = 4; s2_0 = 2;
s2_p = 6;
d_p = 4;
(d_0*s2_0 + d_p*s2_p)/(d_0 + d_p)
```

```
## [1] 4
```

```
d_p = 8
(d_0*s2_0 + d_p*s2_p)/(d_0 + d_p)
```

```
## [1] 4.666667
```

```
d_p = 12
(d_0*s2_0 + d_p*s2_p)/(d_0 + d_p)
```

```
## [1] 5
```

```
d_p = 1
(d_0*s2_0 + d_p*s2_p)/(d_0 + d_p)
```

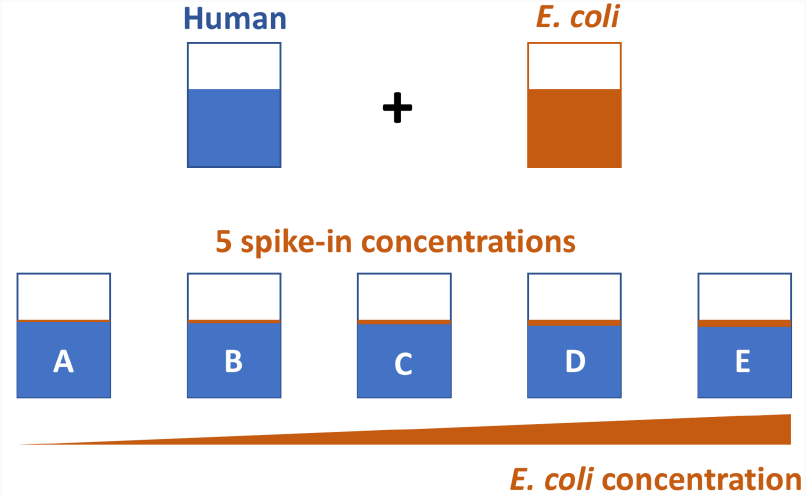
```
## [1] 2.8
```

Benchmarking

The Ionstar dataset.

Table: All possible pairs of E. coli concentrations with the expected fold-changes.

c1	c2	fc
7.5	9.0	1.20
6.0	7.5	1.25
4.5	6.0	1.33
3.0	4.5	1.50
6.0	9.0	1.50
4.5	7.5	1.67
3.0	6.0	2.00
4.5	9.0	2.00



https://www.pnas.org/content/115/21/E4767

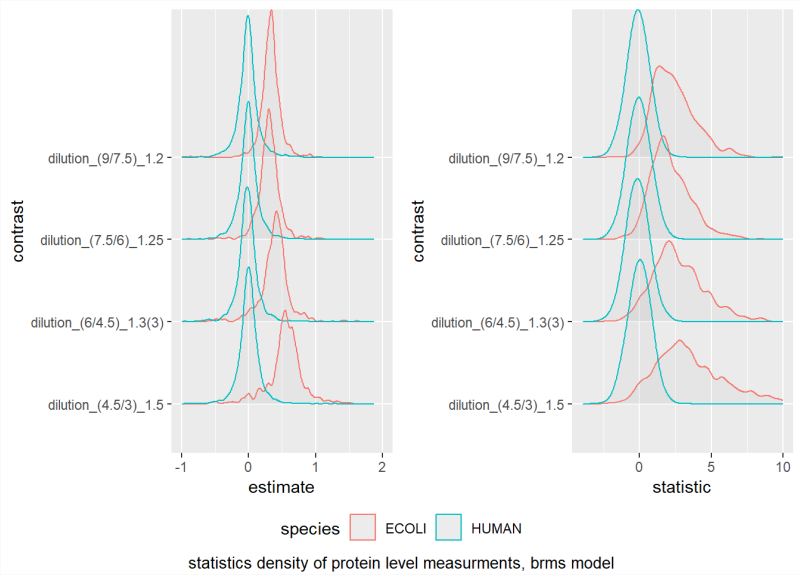
Benchmarking

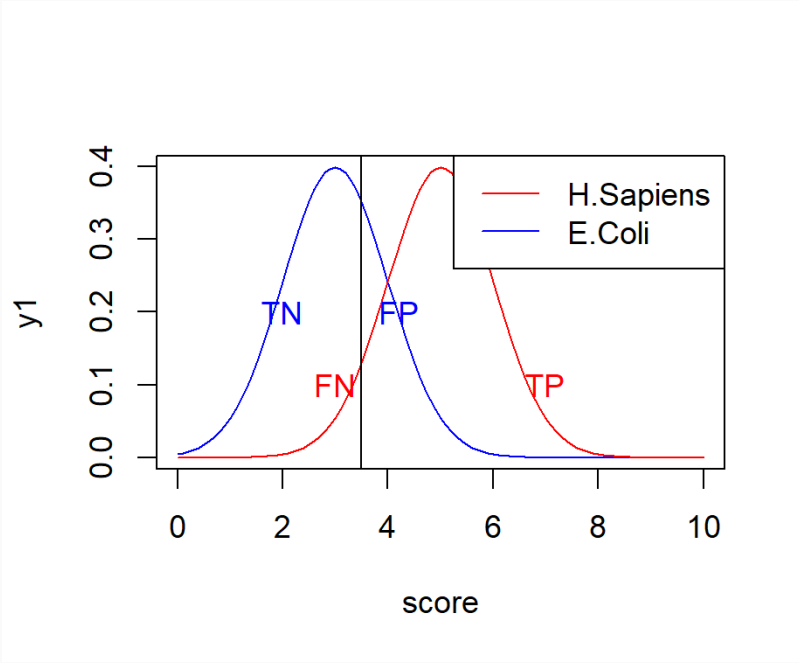
Table: Confusion matrix, TP - true positive, FP - false positive, FN - false negative, P - all positive cases (all E. coli proteins), N - all negative cases (all H. sapiens proteins), m- all proteins.

Prediction \ Truth	**E.coli**	**H.sapiens**	**Total**
beta != 0	TP	FP	R
beta == 0	FN	TN	
Total	P	N	m

$$TPR = \frac{TP}{TP + FN} = \frac{TP}{P}$$
$$FPR = \frac{FP}{FP + TN} = \frac{FP}{N}$$
$$FDP = \frac{FP}{TP + FP} = \frac{FP}{R}$$

Benchmarking





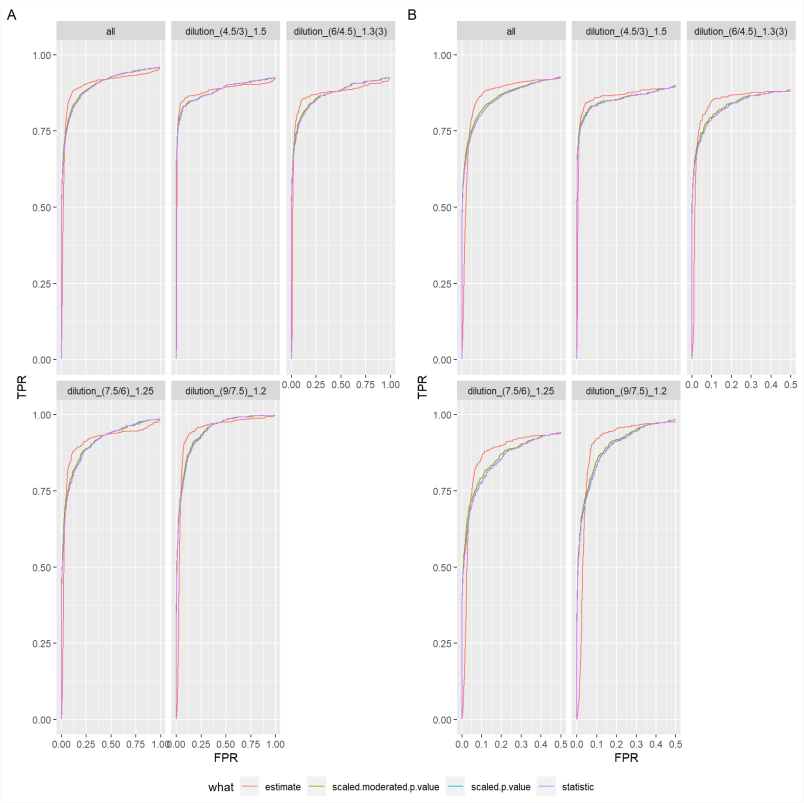
Compute the confusion matrix for each value of the score.

Benchmarking

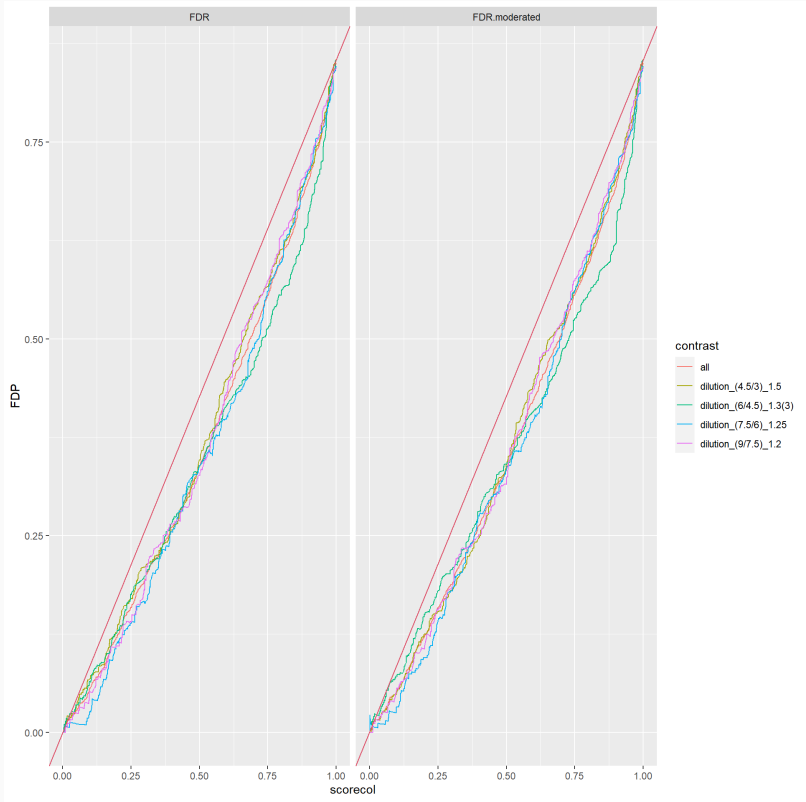
- By plotting the TPR versus the FPR we obtain the receiver operator characteristic curve (ROC curve). The area under the curve (AUC) or partial areas under the curve (pAUC), at various values of the FPR, are further measures of performance.

- A further question we can examine using the benchmark data is, how well the false discovery estimate (FDR) obtained to from the statistical model matches the false discovery proportion (FDP). The FDR is the expected value of the false discovery proportion. Ideally, the FDR should be an unbiased estimate of the FDP. By plotting the FDR gainst the FDP we can asses visually if these assumptions are met.

Benchmarking



ROC curves



FDR vs FDP curves

Conclusions

- Linear models allow to
  - estimate fold changes between condition using contrasts
  - but also test differences of fold changes (interactions).
  - run ANOVA analysis
- If you model more than two conditions
  - Problems because of missing data are more prominent
    (no observations in one of the conditions.)
- p-value moderation improves the protein/peptide variance estimates, the t-statistics and p-values
- Benchmark data is used to test analysis pipelines

Other Software

Other software for modelling fold changes used in Proteomics:

Using linear models

- **limma** - Ritchie, Smyth at al. 2015 PMID: 25605792
- **MSStats** https://www.bioconductor.org/packages/release/bioc/html/MSstats.html
- **ROPECA** Suomi and Elo 2017 PMID: 28724900
- **MSqRob** - Geomine, Gevaert and Clement 2016 PMID: 26566788

Other models

- **mapDIA** - Teo, Kim et al. 2016 PMID: PMID: 26381204
- **tirqler** - https://github.com/statisticalbiotechnology/triqler