An introduction to the general linear model

Application to biomarker discovery with omics data.

Alex Sanchez-Pla
2023-05-06

Outline

1) The general linear models

2) Some examples of simple linear models

3) Fitting linear models

3) Linear models with R

4) Linear models for omics data

5) Exercises

A recap on linear models

What is a linear model (in statistics)

- Linear models appear when we assume that the relation between:
  - one variable,
  - and another (set of) variable(s)

  can be represented through a linear relation, that is, one of the form

$$y \simeq a + b \times x \quad \text{or:} \quad y \simeq a + b \times x + c \times z + \dots$$

- Linear models have been thoroughly used because of their flexibility and many of the important techniques in statistics are "just" linear models.

Regression or ANOVA are linear models

- Linear models provide a flexible way to modelling relations and building explanatory or predictive models, as seen in **linear regression**.

$$Y_i = \beta_0 + \beta_1 \times X_{1i} + \beta_2 \times X_{2i} + \cdots + \epsilon_i$$

- But they also provide a convenient setting to
  - **describe** experimental designs and
  - to **analyze data** that has been obtained from experiments performed according to the design described, as seen in **ANOVA**.

$$Y_{ij} = \mu_i + e_{ij} = \mu + \tau_i + e_{ij}, \quad i = 1 \ldots k, \quad j = 1 \ldots r.$$

The General Linear model

- This capability of linear models to embrace *apparently distinct* models leads to introduce the notion o General Linear Models*.

- GLMs constitute a versatile statistical framework used to analyze relationships between *dependent* (or *response* or *predicor*) variables and one or more independent (or *explanatory* or *predictive*) variables.

- A GLM can be written as:

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + \epsilon_i$$

- where, if there are $i = 1, \ldots, n$ observations,
  - Y_i represents the dependent variable for the ith observation,
  - X_{ij} represents the jth independent variable for the ith observation,
  - β_0 represents the intercept,
  - $\beta_1, \beta_2, \ldots, \beta_p$ represent the coefficients for each independent variable, and
  - ϵ_i represents the error term for the ith observation.

Why is the GLM so "General"?

- The *generality* of the GLM comes from the fact that it can be used to describe, and analyze in a somehow unified form, a variety of problems.
- Although one may think that, assuming that a relation is linear, is an excessive simplification, this approach
  - is thoroughly used,
  - is very successful.
- Indeed, it can be shown that many common statistical models can be re-written as linear models:
  - Common statistical tests are linear models

The GLM has multiple extensions

- Apart of its extensive use "as is",
- The general linear model can be extended in multiple directions
  - Generalized linear models
  - Mixed linear and non linear models
  - etc.

Matrix form for the general linear model

- The equation expressing the linear relationship between the dependent variable and multiple independent variables can be written in a *compact form*, that is, for all individuals at once, using matrix notation as:

$$Y = X\beta + \epsilon$$

- This notation will be useful later when we discuss linear models for omics data analysis.

Some examples of linear models

t-test and ANOVA as GLMs

- Consider a study comparing **two diets** in mice, a "standard" vs a "high-fat" one.
- The main goal of the study are:
  - To *estimate* the mean weight of mice after having been nurrished with each diet
  - To *compare* the difference in weight between the standard and "high fat" diets.
- The usual approach for this problem can be:
  - consider two variables, representing the weight of mice nurrished by ech diet
    - $X_1 \sim N(\mu_1, \sigma_1)$, $X_2 \sim N(\mu_2, \sigma_2)$, or, without assuming normality
    - X_1, X_2 belong to some continuous distribution.
  - And test the hypothesis $H_0 : \mu_1 = \mu_2$ with a $t-test$ or a non parametric one, such as Wilcoxon's test.

Re-write the problem as a linear model

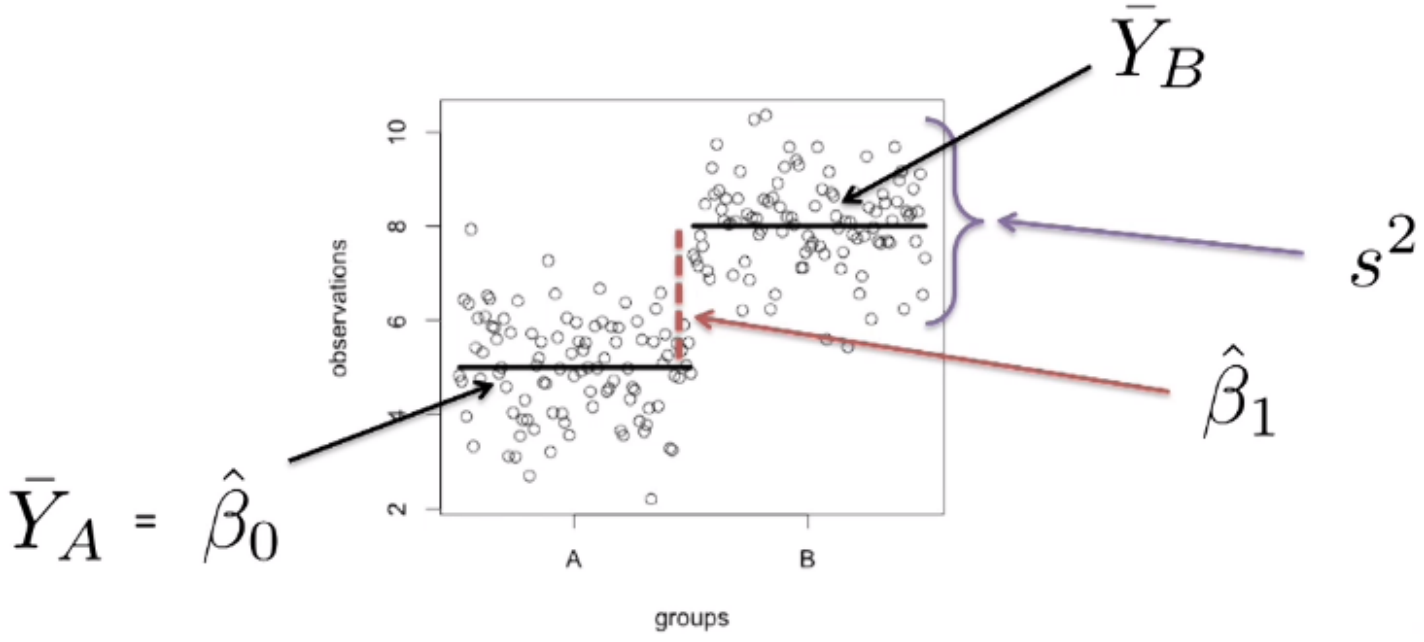
- An alternative approach may come by re-writing the problem as a linear model.
- If *we consider that the weight of the animals is a linear function of the diets*, the following linear model can be written:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, i = 1, \ldots, N$$

- where:
  - Y_i represents the weights of the i-th experimental unit-
  - $x_i = 1$ when mouse i receives the high fat diet and $x_i = 0$ when it receives the standard diet.

Comparison of 2 groups (t-test)

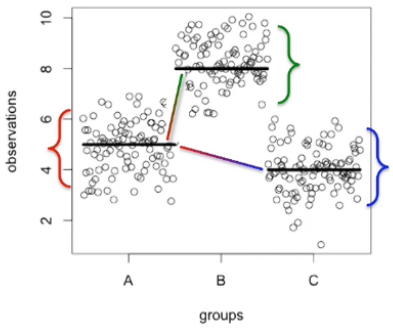
- The coefficients represent the (differential) effect of diet
- Comparison between diets can be based on the means or on the coefficients.



$$\bar{Y}_A = 1 \times \hat{\beta}_0 + 0 \times \hat{\beta}_1$$

$$\bar{Y}_B = 1 \times \hat{\beta}_0 + 1 \times \hat{\beta}_1$$

Comparison of three groups (ANOVA)



$$\bar{Y}_A = \hat{\beta}_0$$
$$\bar{Y}_B = \hat{\beta}_0 + \hat{\beta}_1$$
$$\bar{Y}_C = \hat{\beta}_0 + \hat{\beta}_2$$

- With more than two diets the scenario is similar
  - β_0 represents the "control" or standard
  - β_1 , β_2 the effect of each diet added to the standard
- Comparisons can be based on
  - The means
  - The coefficients

- Notice that, although the linear model postulates a relation between *parameters* we will work with their *estimates* so, there is an error that we need to control in order for our conclusions (inferences) to be reliable.

Matrix notation for linear models

- A *linear model* can be written in a compact way using matrix notation:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

- The matrix $\mathbf{X}$ is called the *design matrix*
- Using this notation any linear model can be described by the right choice of
  - the vector of parameters $\boldsymbol{\beta}$ and
  - the *design matrix* $\mathbf{X}$, which describes the way that way independent variables are weighted by the parameters to give the response variable \mathbf{Y}.

Two groups in matrix notation

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \end{pmatrix}$$

Three groups in matrix notation

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \end{pmatrix}$$

Estimation and testing with linear modesl

Fitting linear models

- For linear models to be useful, we have to *estimate* the parameters, β_i, whose values are unknown.
- A common approach is to use as estimates, $\hat{\beta}_i$ the values that minimize the difference between the "true values" and the model fitted to the data:

$$RSS = \sum_{i=1}^{n} \{y_i - (\beta_0 + \beta_1 x_i)\}^2$$

- This is called the least-squares method, that can be solved in different ways.

The normal equations

- One of the ways to estimate the parameters using least-squares yields to solving a set of equations (one per parameter), called the *normal equations*
- The solution of the normal equations are the *least square estimates* of the parameters:

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

- Solving the normal equations also probvides error estimates for the model coefficients: $$ \operatorname{se}\left(\hat{\beta}*i\right)=\sqrt{s^2\left(X^T X\right)*{i i}^{-1}} $$

Significance testing with linear models.

- Assuming a series of assumptions hold
  - Variance homogeneity
  - Linearity of relations
  - Independence and *normality* of error terms
- A test can be built to test the significance of the model coefficients

$$\frac{\text{signal}}{\text{noise}} = \frac{\hat{\beta}_i}{\text{se}\left(\hat{\beta}_i\right)}$$

How is a linear models usually applied?

- Once one has built and estimated a linear model it can be used for different purposes. For example:
  - Estimating the parameters values to know the mean effect of one factor or of each level of this factor.
  - Comparing the parameters' values to decide if different treatments have the same effect, or have any effect at all.
  - Predicting the observed value of the response variable for a fixed set of values of the independent variable.

Fitting a models to adjust for covariates

- A common quote in papers is something like: *"The study of ... was controlled for age, race, and sex ..."*, $$ Y{{32 \text { yo,Caucasian,female }}}=\beta_0+\beta{25-35 \text { yo }}+\beta{\text {Causasian }}+\beta{\text {female }}+\varepsilon $$
  - A common -and discussable- application of linear models is adjusting the effect of distinct covariables.
  - These covariables:
  - may not be of direct interest in the study ...
  - but they may be considered to have a "confusion effect" on the relation between the response variable and the explanatory ones.
  - For some reason there is often believed (not by statisticians) that adding these variables to the model will automagically remove their effect from the relation between the explanatory and the independent variables.

Linear models in R

Fitting linear models in R

- In practice, when using R, *we rarely fit a model by solving the normal equations*
- The usual, and most practical way to do it, is to use `lm` function.
- The `lm` function requires
  - Either a `formula` relating the variables to be included in a linear model
  - Or a design matrix, that we can create using the `model.matrix` function, and which, implicitly, defines that model.

The importance of the design matrix

- The choice of design matrix is a *critical step* in linear modeling as
    - it encodes which coefficients will be fit in the model,
    - and the inter-relationship between the samples.
- Defining which design matrix we use is equivalent to defining the parameters of the model (or the model's *parametrization*).
    - Same data can be modelled differently, if parameters receive different meanings.
    - In practice this represents using distinct design matrices
- A typical example: How does the meaning of a one-way factor ANOVA change if we consider a model with or without an intercept?
    - How is it reflected in the design matrix?

Model matrix for two groups

- Suppose we have two groups, 1 and 2, with two samples each.
- We might start to encode this experimental design like so:

```
x ← c(1,1,2,2)
f ← formula(~ x)
model.matrix(f)
```

```
##   (Intercept) x
## 1           1 1
## 2           1 1
## 3           1 2
## 4           1 2
## attr(,"assign")
## [1] 0 1
```

Model matrix for two groups (2)

- Note that an intercept will be included by default, so the formula could equivalently be written: `~ x + 1`.
- We can then inspect the design matrix which is formed by this:

```
model.matrix(f)
```

```
##   (Intercept) x
## 1           1 1
## 2           1 1
## 3           1 2
## 4           1 2
## attr(,"assign")
## [1] 0 1
```

model.matrix requires factors

- Note, this is not the design matrix we wanted.
- We should instead first tell R that these values should not be interpreted numerically, but as different levels of a factor variable:

```
x ← factor(c(1,1,2,2))
model.matrix(~ x)
```

```
##   (Intercept) x2
## 1           1  0
## 2           1  0
## 3           1  1
## 4           1  1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

- Now we have achieved the correct design matrix.
- Or have we?

The role of the intercept term

- Note that the previous matrix has one intercept column and one group column although there are two groups indeed.
    - The first group's values are represented by the basal or "overall mean".
    - The second group's are represented by one column.
- An alternative representation is possible setting the intercept to zero.
- Both representations are equivalent and for one-factor designs it's up to you which one to choose

```
x ← factor(c(1,1,2,2))
model.matrix(~ x + 0)
```

```
##   x1 x2
## 1  1  0
## 2  1  0
## 3  0  1
## 4  0  1
## attr(,"assign")
## [1] 1 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

Design matrix for more than 2 groups

- How is the design matrix for an experiment with 3 groups?
- We proceed like in the previous case

```
x ← factor(c(1,1,2,2,3,3))
model.matrix(~ x)
```

```
##   (Intercept) x2 x3
## 1           1  0  0
## 2           1  0  0
## 3           1  1  0
## 4           1  1  0
## 5           1  0  1
## 6           1  0  1
## attr(,"assign")
## [1] 0 1 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

- Again the first group is implicit in the intercept but it can be set explicitly by setting the intercept to zero.

An alternative parametrization

- An alternate formulation of design matrix is possible by specifying `+0` in the formula:

```
x ← factor(c(1,1,2,2,3,3))
model.matrix(~ x + 0)
```

```
##   x1 x2 x3
## 1  1  0  0
## 2  1  0  0
## 3  0  1  0
## 4  0  1  0
## 5  0  0  1
## 6  0  0  1
## attr(,"assign")
## [1] 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

- This representation allows fitting a separate coefficient for each group.

Linear Models for Omics Data

Linear models for omics data

- For the analysis of omics data a very popular option is the `limma` package.
- `limma` extends some R functionalities to make them easy to use in the analysis of omics data using linear models.
- Besides this it includes extensions to the standard linear model to improve analysis capabilities.
- In the following we show
    - How to create a design matrix from a "targets" file containing information on groups.
    - How to create a contrasts matrix to define the comparisons to be done.
    - How to do the comparisons and how to interpret the resulting analysis tables.

Example: Comparing 3 types of tumors

- This example study is based on a paper published in http://www.ncbi.nlm.nih.gov/pubmed/15897907 whose data are available in GEO as series GSE1561 series on the following link http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE1561
- The researchers investigated three types of breast cancer tumors: apocrine (APO), basal (BAS) and luminal (LUMI).
- The classification is based on the resistance of tumors to estrogen and androgen receptors.
    - Tumors classified as "APO" are negative for estrogen receptor (ER-) and positive for the androgen receptor (AR+).
    - Those classified as "LUMI" are ER + and AR + and
    - Those classified as "BAS" are ER- and AR.

Identifying groups and comparisons

- The assignment of each sample to an experimental group can be obtained from this link: http://www.ncbi.nlm.nih.gov/geo/gds/profileGraph.cgi?gds=1329

- Obviously this is an observational study but its analysis can be done using a linear model approach as well.

- We will usually proceed in three steps:

1. Identify the experimental factors and their levels.
2. Write the design matrix associated with this study design.
3. Build the contrast matrix that defines the comparisons we are interested in.

- In this example we have identified three groups and we wish to compare each tumor type with the oher two

    1. "APO" vs “LUMI”
    2. “APO" vs “BAS”
    3. “LUMI" vas "BAS"

Defining the groups

- An easy way to define the study groups consists of preparing a text file where these are established.

```
url ← "https://raw.githubusercontent.com/ASPteaching"
repo ← "Introduction_to_Design_of_Experiments"
folder ← "main/omicsData/dataset_2_Breast_cancer_GSE1561/data"
dataFile ← "BreastCancerGSE1561.csv"
targetsFile ← "targets.txt"
targetsFileName ← paste(url, repo, folder, targetsFile, sep="/")
```

The `targets` file

```
library(magrittr)
targets ←read.table(targetsFileName, row.names=1, head=T)
```

```
kableExtra::kable(head(targets[,1:6]), n=7) %>% kableExtra::kable_styling()
```

	fileName	Sample	Ids	SampleIDs	Group	Apocrine.grade
GSM26878.CEL	GSM26878	PF14	EnPnT2N1G2	PF14	A	3
GSM26883.CEL	GSM26883	PF19	EnPuT4N0Gu	PF19	A	2
GSM26887.CEL	GSM26887	PF23	EnPnT2N0G2	PF23	A	3
GSM26903.CEL	GSM26903	PF39	EnPuT4N0Gu	PF39	A	3
GSM26910.CEL	GSM26910	PF46	EnPnT4N1G3	PF46	A	3
GSM26888.CEL	GSM26888	PF24	EnPnTiN0G3	PF24	B	2

Showing only first 7 rows

Creating the design matrix

```
design←matrix(
  c(1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,
    0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,
    0,0,0,0,0,0,0,0,0,0,1,1,1,1,1),
  nrow=15,byrow=F)
colnames(design) ←c("A", "B", "L")
rownames(design)←  targets$Sample
```

```
print(design)
```

```
##       A B L
## PF14  1 0 0
## PF19  1 0 0
## PF23  1 0 0
## PF39  1 0 0
## PF46  1 0 0
## PF24  0 1 0
## PF25  0 1 0
## PF28  0 1 0
## PF34  0 1 0
## PF42  0 1 0
## PF15  0 0 1
## PF32  0 0 1
## PF33  0 0 1
## PF43  0 0 1
## PF47  0 0 1
```

Another way to create the design matrix

```
design2 <-model.matrix(~ 0+targets$Group)
colnames(design2)<-c("A", "B", "L")
rownames(design2)<-  targets$Sample
```

```
print(design2)
```

```
##       A B L
## PF14 1 0 0
## PF19 1 0 0
## PF23 1 0 0
## PF39 1 0 0
## PF46 1 0 0
## PF24 0 1 0
## PF25 0 1 0
## PF28 0 1 0
## PF34 0 1 0
## PF42 0 1 0
## PF15 0 0 1
## PF32 0 0 1
## PF33 0 0 1
## PF43 0 0 1
## PF47 0 0 1
## attr(,"assign")
## [1] 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$`targets$Group`
## [1] "contr.treatment"
```

Defining the questions: The contrast matrix

```
library(limma)
cont.matrix ← makeContrasts (
  AvsB = B-A,
  AvsL = L-A,
  BvsL = L-B,
  levels=design)
cont.matrix
```

```
##        Contrasts
## Levels AvsB AvsL BvsL
##      A   -1   -1    0
##      B    1    0   -1
##      L    0    1    1
```

Data for the analysis

- Data can be obtained from the open repository Gene Expression Omnibus but, for simplicity, it has been downloaded and prepared for the analysis.
- They are also available remotely at the sub-directory "omicsData"

```
dataFile ← "BreastCancerGSE1561.csv"
dataFileName ← paste(url, repo, folder, dataFile, sep="/")

dataMatrix ← read.csv(dataFileName, row.names=1)

colnames(dataMatrix)==rownames(targets)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
dim(dataMatrix)
```

```
## [1] 22283    15
```

Fit the model and the contrasts

- Once we have the data, the design matrix and the contrast matrix we can proceed to estimate the model, fit the contrasts and check the results

```
fit←lmFit(dataMatrix, design)
fit.main←contrasts.fit(fit, cont.matrix)
fit.main←eBayes(fit.main)
```

- This creates an object, which is called here `fit.main` from where distinct results can be extracted.

Results are in "topTables"

- For each comparison in the contrast matrix a "top Table" can be generated showing features sorted from most to least differentially expressed, based on the test p-value.

```
topTab_AvsB ← topTable (fit.main, number=nrow(fit.main), coef="AvsB", adjust="fdr")
kableExtra::kable(head(topTab_AvsB, n=5)) %>% kableExtra::kable_styling()
```

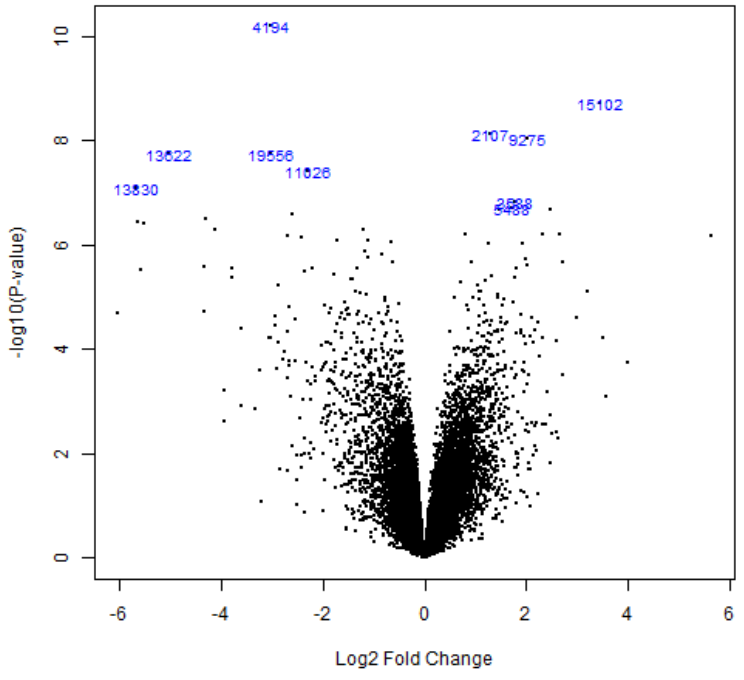
	logFC	AveExpr	t	P.Value	adj.P.Val	B
204667_at	-3.038344	8.651157	-17.65663	0	1.40e-06	13.610350
215729_s_at	3.452290	6.137595	13.67095	0	2.08e-05	11.214963
202579_x_at	1.293975	10.226164	12.26570	0	5.24e-05	10.115252
209787_s_at	2.023113	9.871298	12.05888	0	5.24e-05	9.939472
214243_s_at	-5.015130	9.278980	-11.47539	0	6.75e-05	9.422507

```
# topTab_AvsL ← topTable (fit.main, number=nrow(fit.main), coef="AvsL", adjust="fdr"
# topTab_BvsL  ← topTable (fit.main, number=nrow(fit.main) , coef="BvsL", adjust="fd
```

Volcano plots provide visualization

- A volcano plot shows, for each comparison
  - the magnitude of the change ("biological significance") vs
  - the "statistical significance" (-log p-value)

```
volcanoplot(fit.main, coef="AvsB",
                      highlight=10)
```



Exercises

Exercises on linear models for omics

- You can go through the exercises in the document Exercises on linear models for microarrays
    - Start by reading the experiment description
    - Postulate the type of experimental design
    - Write the linear model
    - Create the design and the contrast matrix
    - Get the data
    - Fit the model
    - Examine the results