Continuous models and intro to `limma`

Keegan Korthauer

2 February 2022

with slide contributions from Gabriela Cohen Freue and Jenny Bryan

STAT 540

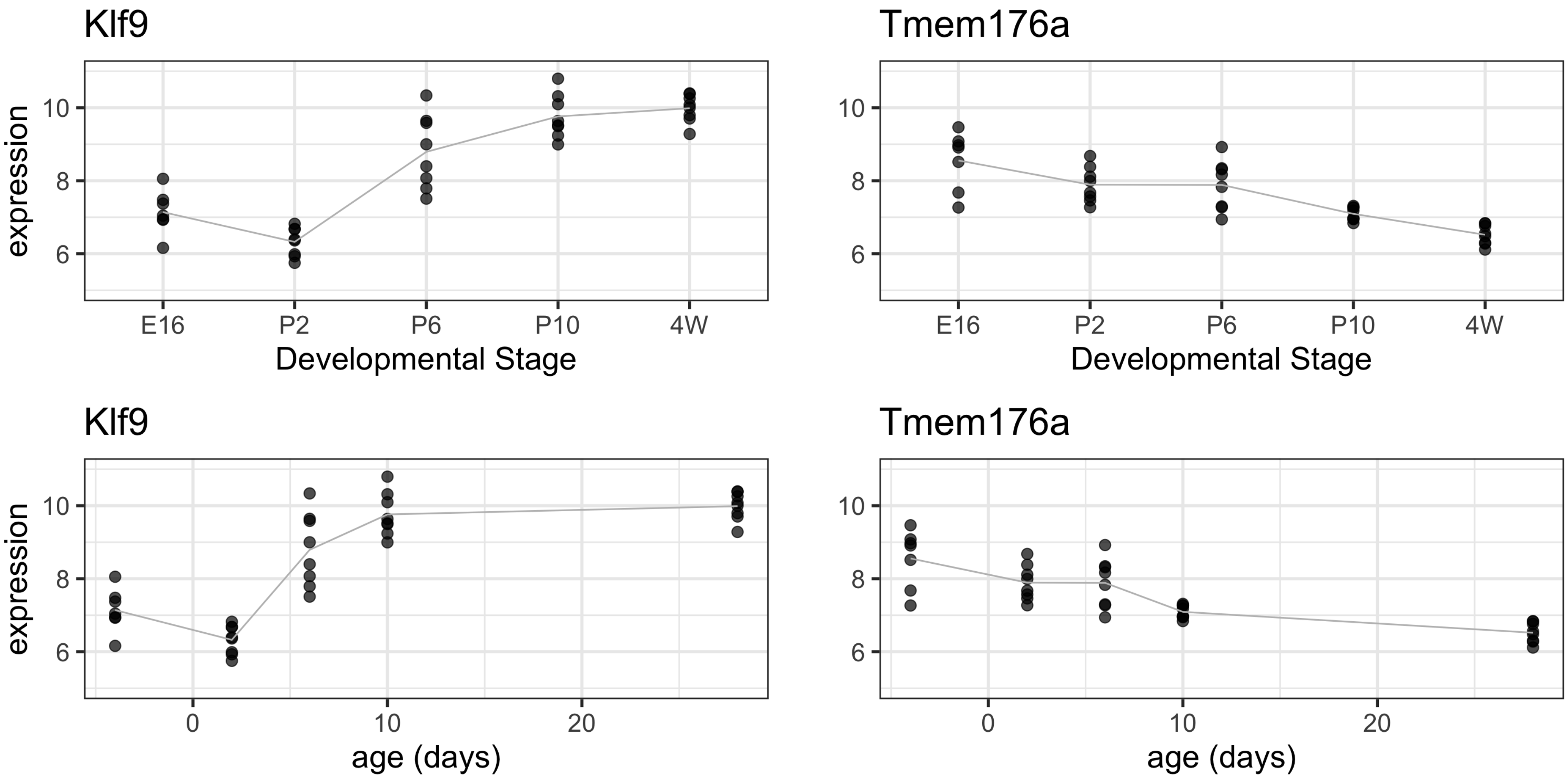
Summary so far

- *t*-tests can be used to test the equality of 2 population means
- ANOVA can be used to test the equality of more than 2 population means
- Linear regression provides a general framework for modeling the relationship between a response variable and different types of explanatory variables
  - *t*-tests can be used to test the significance of *individual* coefficients
  - *F*-tests can be used to test the simultaneous significance of *multiple* coefficients (e.g. multiple levels of a single categorical factor, or multiple factors at once)
  - *F*-tests are used to compare nested models (overall effects or goodness of fit)
- Next up: continuous explanatory variables! Multiple genes!

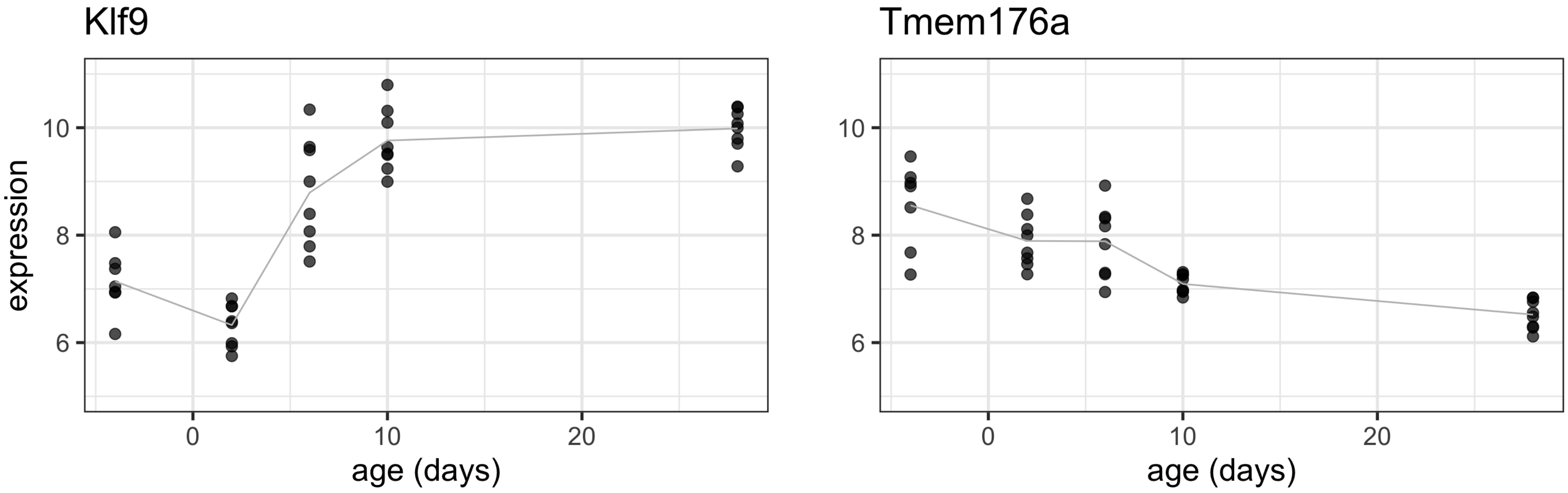
Learning objectives for today

- Understand how linear regression represents continuous variables
  - Be familiar with the intuition behind how the regression line is estimated (Ordinary Least Squares)
  - Interpret parameters in a multiple linear regression model with continuous and factor variables
- Explain the motivation behind specialized methods regression models in high-dimensional settings
  - e.g. Empirical Bayes techniques in `limma`

What if we treat age as a continuous variable?



Linear model with age as continuous covariate



- Linear looks reasonable for gene Tmem176a, but not so much for Klf9
- For now, assume linear is reasonable

Simple Linear Regression (Matrix form)

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

For 1 continuous/quantitative covariate:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}, \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

- $\alpha_0 =$ the intercept (expected value of y when x is equal to zero)
- $\alpha_1 =$ the slope (expected change in y for every one-unit increase in x)

Simple Linear Regression (Matrix form)

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

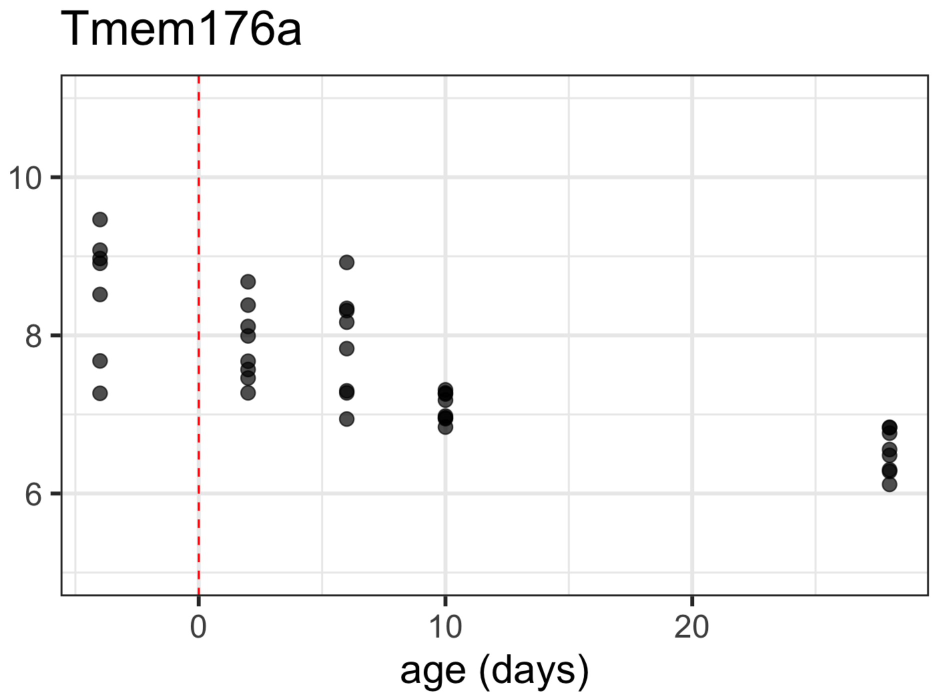
Remember / convince yourself that the matrix algebra yields simple linear equations:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} = \begin{bmatrix} 1 * \alpha_0 + x_1\alpha_1 \\ 1 * \alpha_0 + x_2\alpha_1 \\ \vdots \\ 1 * \alpha_0 + x_n\alpha_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

$$= \begin{bmatrix} \alpha_0 + x_1\alpha_1 + \varepsilon_1 \\ \alpha_0 + x_2\alpha_1 + \varepsilon_2 \\ \vdots \\ \alpha_0 + x_n\alpha_1 + \varepsilon_n \end{bmatrix}$$

$$\Rightarrow y_i = \alpha_0 + x_i\alpha_1 + \varepsilon_i$$

SLR with continuous age covariate



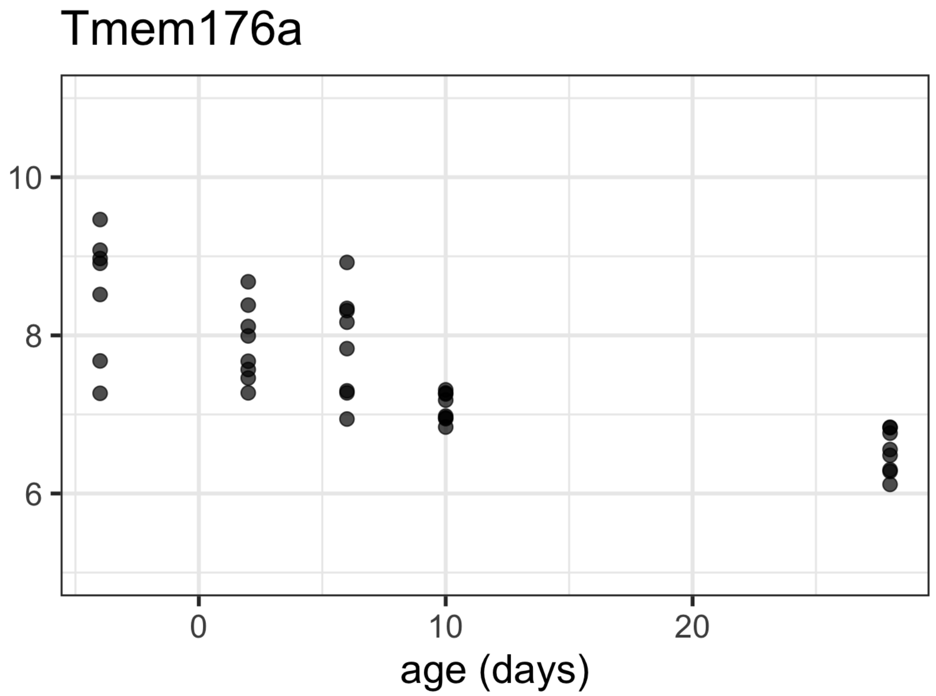
Interpretation of intercept:

```
filter(twoGenes, gene == "Tmem176a") %>%
lm(expression ~ age, data = .) %>%
  summary() %>% .$coeff
```

```
##                Estimate  Std. Error  t value
Pr(>|t|)
## (Intercept)  8.10007931 0.113949630 71.08474 3.579302e-
41
## age         -0.06137385 0.008214834 -7.47110 6.742526e-
09
```

$H_0 : \alpha_0 = 0$ tests the null hypothesis that the intercept is zero - usually, not of interest

SLR with continuous age covariate



Interpretation of slope:

```
filter(twoGenes, gene == "Tmem176a") %>%
lm(expression ~ age, data = .) %>%
  summary() %>% .$coeff
```

```
##                  Estimate  Std. Error  t value
Pr(>|t|)
## (Intercept)  8.10007931 0.113949630 71.08474 3.579302e-
41
## age         -0.06137385 0.008214834 -7.47110 6.742526e-
09
```

$H_0 : \alpha_1 = 0$ tests the null hypothesis that there is no association between gene expression and age - usually of interest

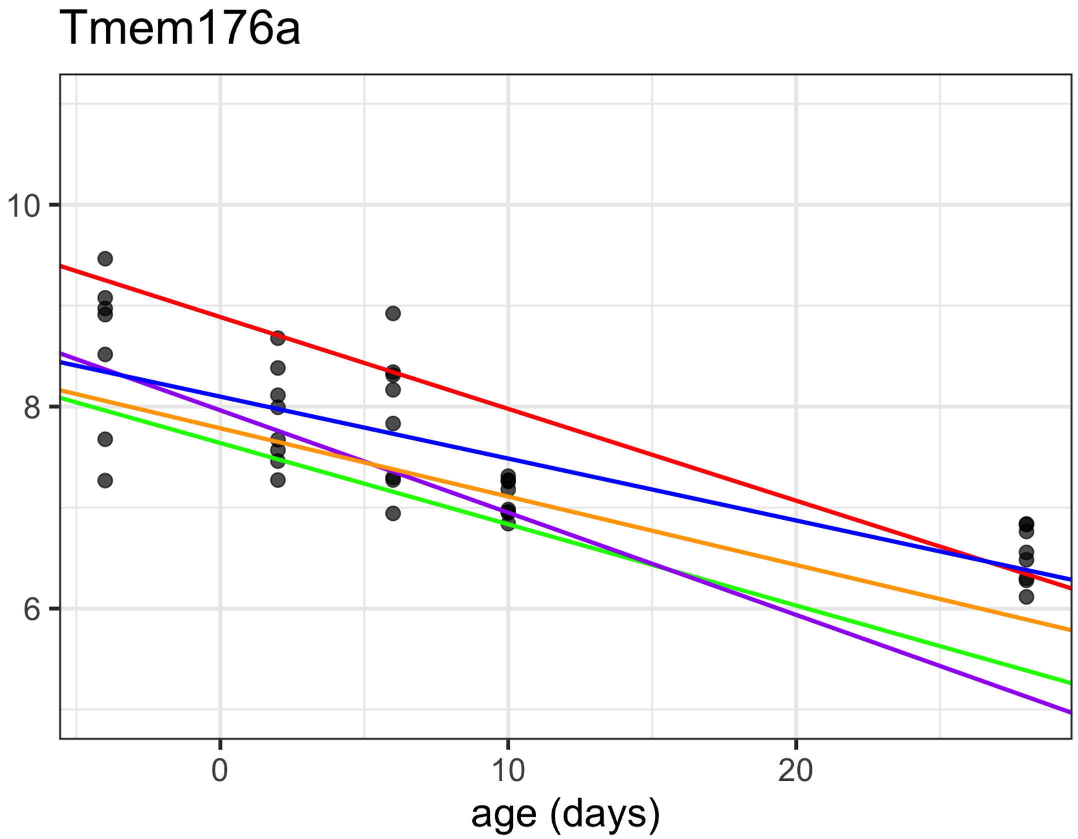
How do we estimate the intercept and slope?

Is there an optimal line?

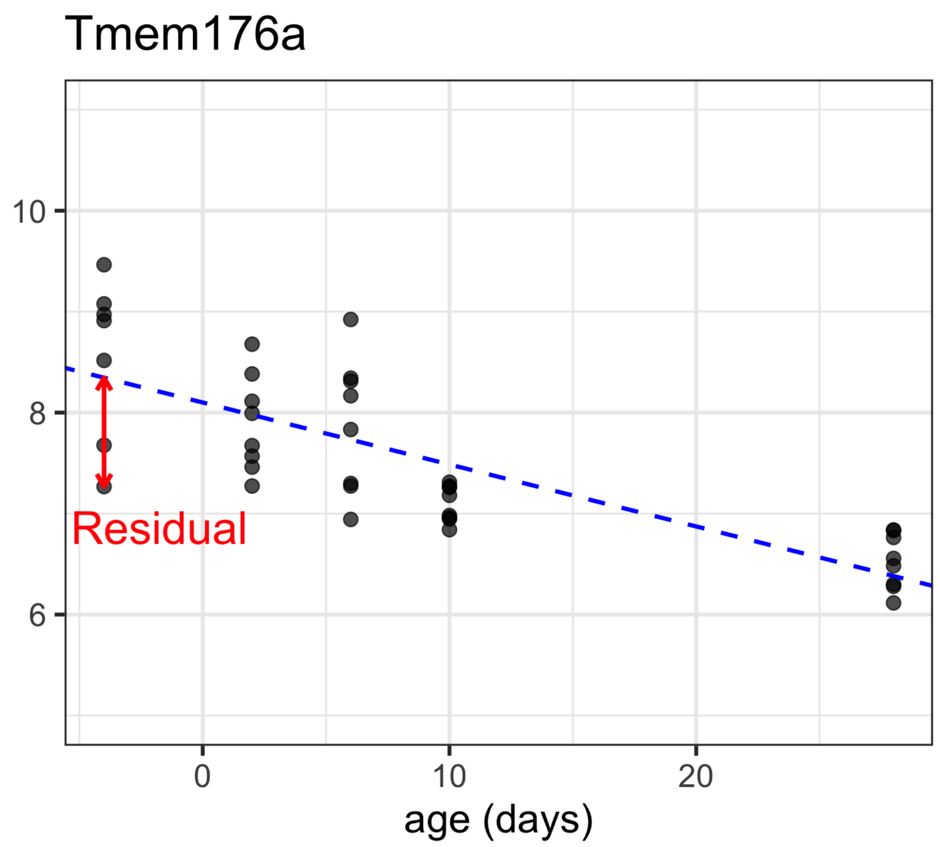
```
filter(twoGenes, gene == "Tmem176a") %>%
lm(expression ~ age, data = .) %>%
  summary()
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.100079   0.113950  71.085  < 2e-16 ***
## age         -0.061374   0.008215  -7.471 6.74e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5535 on 37 degrees of freedom
## Multiple R-squared:  0.6014,    Adjusted R-squared:  0.5906
## F-statistic: 55.82 on 1 and 37 DF,  p-value: 6.743e-09
```

Which one is the *best* line?



Ordinary Least Squares



- Ordinary Least Squares (OLS) regression: parameter estimates minimize the sum of squared errors
- Error: vertical (y) distance between the true regression line (unobserved) and the real observation
- Residual: vertical (y) distance between the fitted regression line and the real observation (estimated error)

OLS interactive demo

Visual representation of the squared errors in OLS: http://setosa.io/ev/ordinary-least-squares-regression/

Visit the link and take a few minutes to explore the first two plots:

1. In the first plot, drag individual points around and observe what changes in second plot
2. In the second plot, adjust the slope and intercept dials from their defaults and observe what changes
    - note: you can reset to default values by refreshing the page

In general, what happens to the total area of the squares in the second plot when you modify the slope and intercept from the default values?

OLS Estimator for Simple Linear Regression (1 covariate)

- Mathematically: ε_i represents the error

$$y_i = \alpha_0 + \alpha_1 x_i + \varepsilon_i, i = 1, \ldots, n$$

- We want to find the line (i.e. an intercept and slope) such that the sum of squared errors is minimized

$$S(\alpha_0, \alpha_1) = \sum_{i=1}^{n} (y_i - \alpha_0 - \alpha_1 x_i)^2$$

  - $\varepsilon_i = y_i - \alpha_0 - \alpha_1 x_i$ is the error
  - $S(\alpha_0, \alpha_1)$ is called an *objective function*

- How to obtain estimates $(\hat{\alpha}_0, \hat{\alpha}_1)$? Let's look at a more general case

OLS for Multiple Linear Regression (p covariates)

- Mathematically:

$$S(\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_p) = \sum_{i=1}^{n} (y_i - \alpha_0 - \alpha_1 x_{1i} - \alpha_2 x_{2i} - \ldots - \alpha_p x_{pi})^2$$

$$= (\mathbf{y} - \mathbf{X}\boldsymbol{\alpha})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\alpha})$$

- We need to find values of $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_p)$ that minimize the sum of squares S

OLS for Multiple Linear Regression (p covariates)

- Mathematically:

$$S(\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_p) = \sum_{i=1}^{n} (y_i - \alpha_0 - \alpha_1 x_{1i} - \alpha_2 x_{2i} - \ldots - \alpha_p x_{pi})^2$$

$$= (\mathbf{y} - \mathbf{X}\boldsymbol{\alpha})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\alpha})$$

- We need to find values of $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_p)$ that minimize the sum of squares S
- Take partial derviatives with respect to each coeff, set to zero, solve system of equations:

$$\frac{\partial S}{\partial \alpha_0} = \begin{bmatrix} \frac{\partial S}{\partial \alpha_0} \\ \frac{\partial S}{\partial \alpha_1} \\ \vdots \\ \frac{\partial S}{\partial \alpha_p} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Properties of OLS regression

Regression model: $\mathbf{Y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$

OLS estimator: $\hat{\boldsymbol{\alpha}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

Fitted/predicted values: $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\alpha}}$

$$= \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$$

where $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is called the "hat matrix"

Properties of OLS regression

Regression model: $\mathbf{Y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$

OLS estimator: $\hat{\boldsymbol{\alpha}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

Fitted/predicted values: $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\alpha}}$

$$= \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$$

where $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is called the "hat matrix"

Additional assumptions (required for results on the next few slides):

1. $\boldsymbol{\varepsilon}$ have mean zero
2. $\boldsymbol{\varepsilon}$ are iid (implies constant variance)

Further, if $\boldsymbol{\varepsilon}$ are iid Normal, then OLS estimator is also MLE (Maximum Likelihood Estimator)

Properties of OLS regression (cont'd)

Residuals: (note NOT the same as errors $\boldsymbol{\varepsilon}$)

$$\hat{\boldsymbol{\varepsilon}} = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\alpha}}$$

Estimated error variance:

$$\hat{\sigma}^2 = \frac{1}{n-p}\hat{\boldsymbol{\varepsilon}}^T\hat{\boldsymbol{\varepsilon}}$$

Estimated covariance matrix of $\hat{\boldsymbol{\alpha}}$:

$$\hat{Var}(\hat{\boldsymbol{\alpha}}) = \hat{\sigma}^2(\mathbf{X}^T\mathbf{X})^{-1}$$

Estimated standard errors for estimated regression coefficients: $\hat{se}(\hat{\alpha}_j)$, obtained by taking the square root of the diagonal elements of $\hat{Var}(\hat{\boldsymbol{\alpha}})$

Inference in Regression (normal iid errors)

How to test $H_0 : \alpha_j = 0$?

With a *t*-test!

Under H_0,

$$\frac{\hat{\alpha}_j}{\hat{se}(\hat{\alpha}_j)} \sim t_{n-p}$$

So a *p*-value is obtained by computing a tail probability for the observed value of $\hat{\alpha}_j$ from a t_{n-p} distribution

Inference - what if we don't assume Normal errors?

Inference - what if we don't assume Normal errors?

How to test $H_0 : \alpha_j = 0$?

Assuming large enough sample size, with a *t*-test!

Under H_0, asymptotically (by CLT)

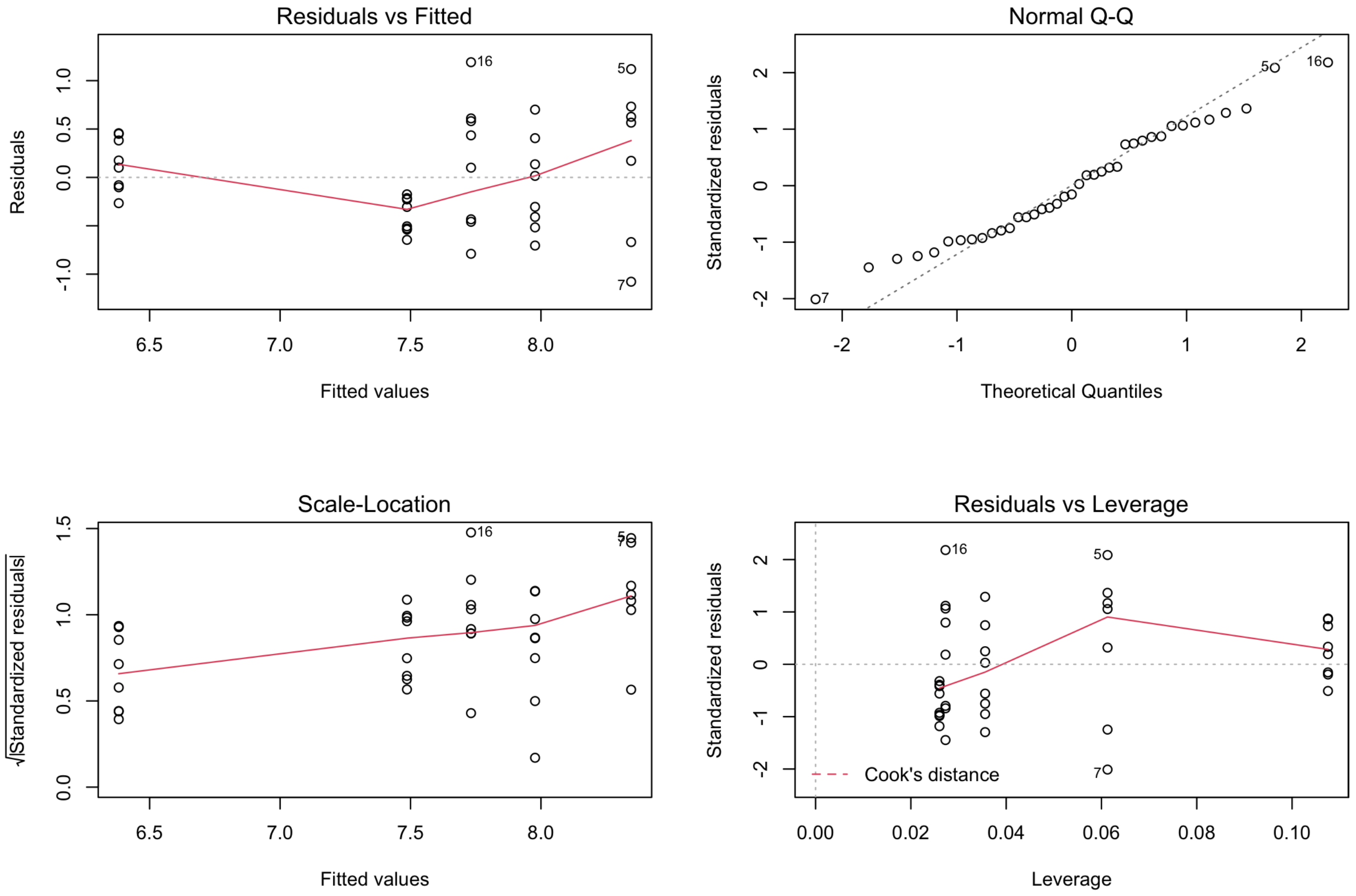
$$\frac{\hat{\alpha}_j}{\hat{se}(\hat{\alpha}_j)} \sim t_{n-p}$$

So with a large enough sample size a *p*-value for this hypothesis test is obtained by computing a tail probability for the observed value of $\hat{\alpha}_j$ from a t_{n-p} distribution

Diagnostics: `plot(lm(y~x))`

Do our assumptions hold?

- Constant variance
- iid errors
- Normality of errors



Linear regression

- The nature of the regression function $y = f(x|\boldsymbol{\alpha})$ is one of the defining characteristics of a regression model
- f is not linear in $\boldsymbol{\alpha}$ $\Rightarrow$ nonlinear model
  - For example, consider nonlinear parametric regression:

$$y_i = \frac{1}{1 + e^{\alpha_0 + \alpha_1 x_i}} + \varepsilon_i$$

- f is linear in $\boldsymbol{\alpha}$ $\Rightarrow$ linear model
  - We just examined simple linear regression (a linear model): $y_i = \alpha_0 + \alpha_1 x_i + \varepsilon_i$
  - What we could do instead: polynomial regression (also a linear model)

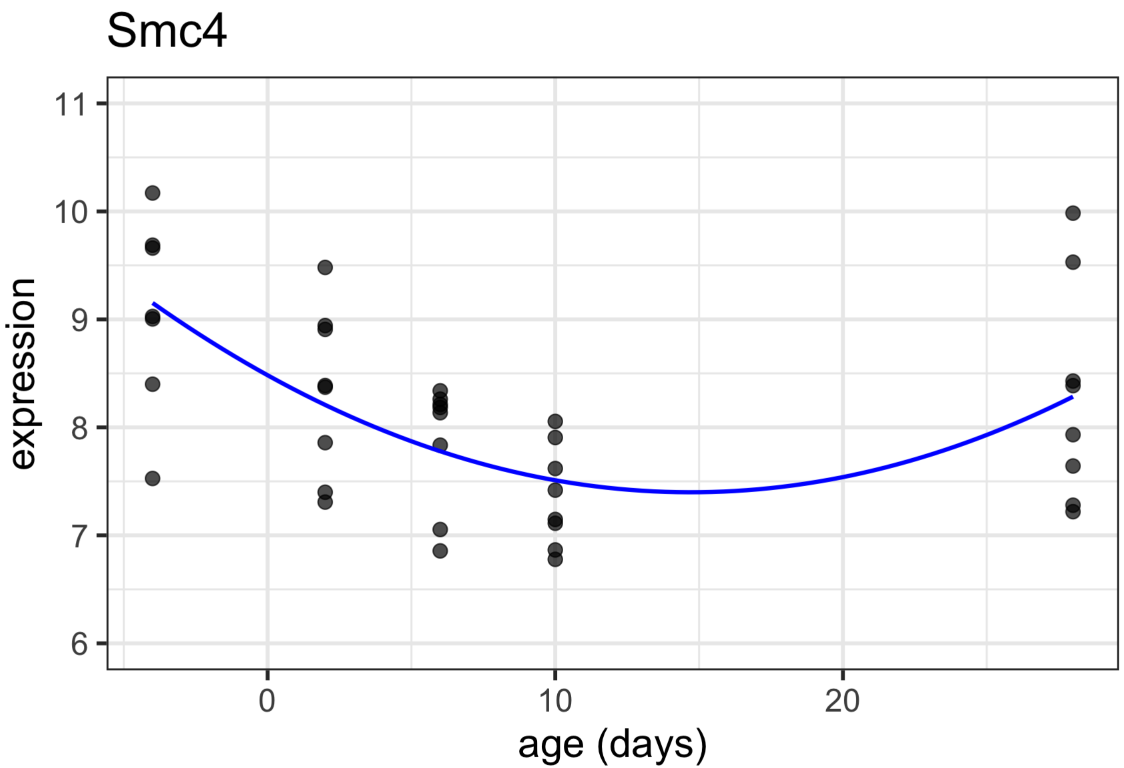
$$y_i = \alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \varepsilon_i$$

Polynomial regression

```
quadfit <- lm(expression ~ age + I(age^2), data = oneGene)
summary(quadfit)
```

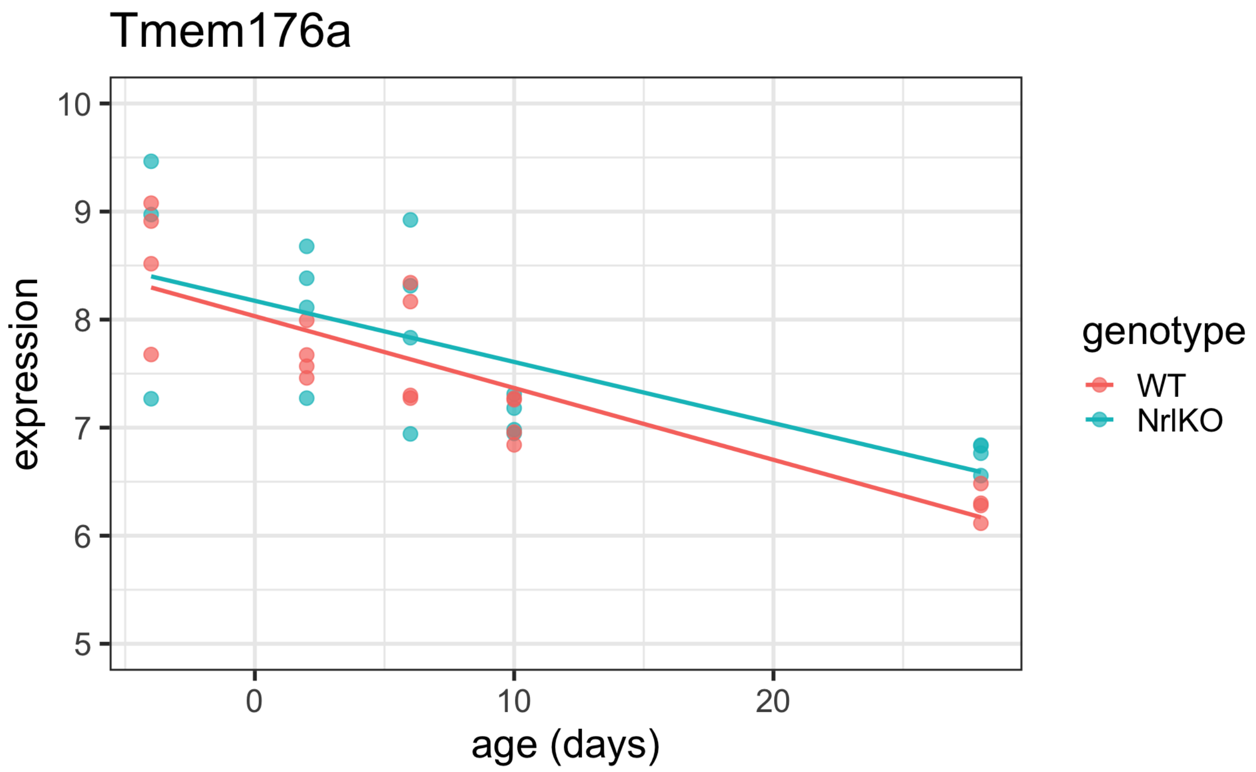
```
##
## Call:
## lm(formula = expression ~ age + I(age^2), data = oneGene)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.6253 -0.6436  0.1023  0.4955  1.6996
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.482542   0.160883  52.725  < 2e-16 ***
## age         -0.147339   0.032626  -4.516 6.52e-05 ***
## I(age^2)     0.005009   0.001164   4.303 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7527 on 36 degrees of freedom
## Multiple R-squared:  0.362,    Adjusted R-squared:  0.3265
## F-statistic: 10.21 on 2 and 36 DF,  p-value: 0.0003069
```

Polynomial regression



Note that this is still a linear model, because it is linear in the α_j

Putting it all together (continuous + categorical variables)



Interaction between continuous and categorical variables

```
lm(expression ~ age*genotype, data = filter(twoGenes, gene=="Tmem176a")) %>%
  summary() %>% .$coeff
```

```
##                        Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)         8.031510398 0.15654982 51.3032221 1.567640e-34
## age                -0.066454446 0.01141757 -5.8203672 1.331685e-06
## genotypeNrlKO       0.142283869 0.22824752  0.6233753 5.370794e-01
## age:genotypeNrlKO   0.009873243 0.01644292  0.6004556 5.520712e-01
```

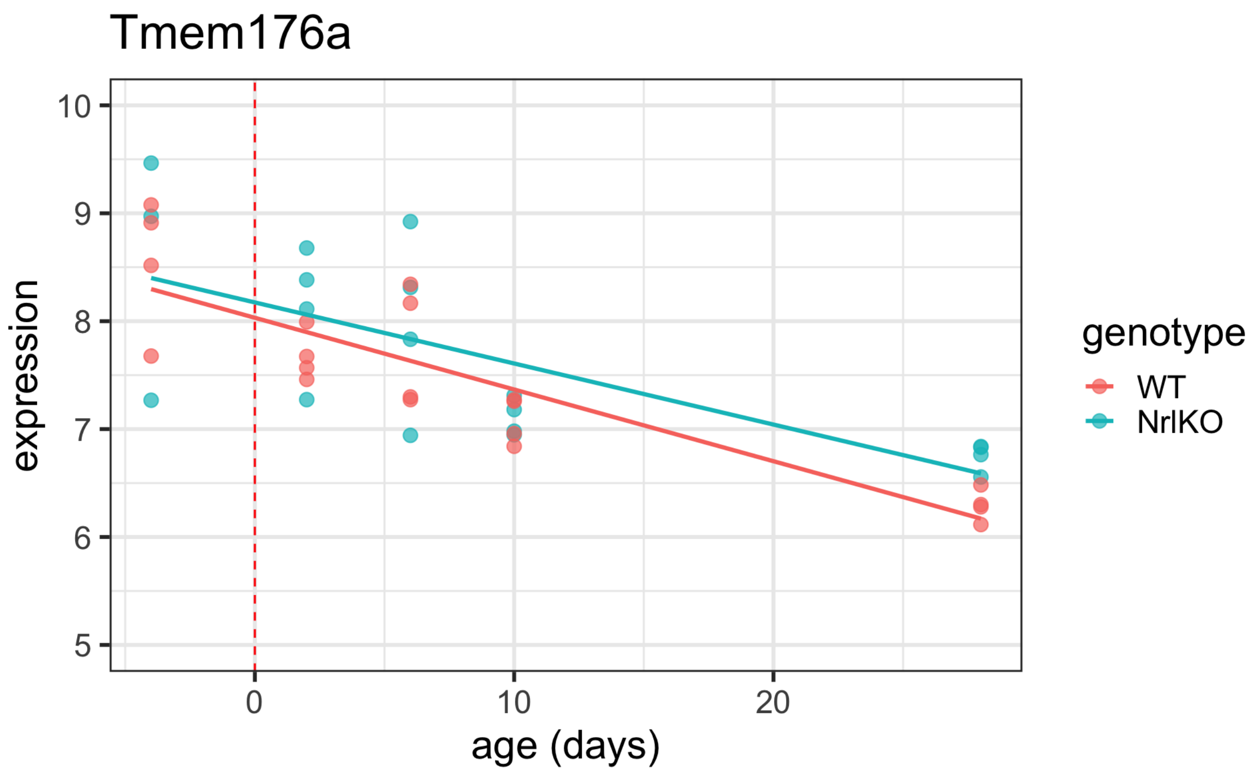
`(Intercept)`: Intercept of WT line

`age`: slope of WT (reference) line

`genotypeNrlKO`: difference in intercepts (KO vs WT)

`age:genotypeNrlKO`: difference in slopes (KO vs WT)

Reminder about the Intercept



Intercept terms refer to the estimates when the continuous covariate is equal to zero. Note that this is not usually very interesting on its own.

Interaction between continuous and categorical variables

$$y_{ij} = \alpha_0 + \tau_{KO} x_{ij,KO} + \tau_{Age} x_{ij,Age} + \tau_{KO:Age} x_{ij,KO} x_{ij,Age}$$

where

- $j \in \{WT, NrlKO\}$, $i = 1, 2, \ldots, n_j$
- $x_{ij,KO}$ is the indicator variable for WT vs KO ($x_{ij,KO} = 1$ for $j = NrlKO$ and 0 for $j = WT$)
- $x_{ij,Age}$ is the continuous age covariate

Interpretation of parameters:

- α_0 is the expected expression in WT for age = 0
- The "intercept" for the knockouts is: $\alpha_0 + \tau_{KO}$
- τ_{Age} is the expected increase in expression in WT for every 1 day increase in age
- The slope for the knockouts is: $\tau_{Age} + \tau_{KO:Age}$

Nested models

As always, you can assess the relevance of several terms at once - such as everything involving genotype - with an *F*-test:

```
anova(lm(expression ~ age*genotype, data = filter(twoGenes, gene=="Klf9")),
      lm(expression ~ age, data = filter(twoGenes, gene=="Klf9")))
```

```
## Analysis of Variance Table
##
## Model 1: expression ~ age * genotype
## Model 2: expression ~ age
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     35 45.948
## 2     37 45.984 -2 -0.036415 0.0139 0.9862
```

We don't have evidence that genotype affects the intercept or the slope

F-tests in regression

Model	Example	# params (df)	RSS
Reduced	expression ~ age	$p_{Red} = 2$	RSS_{Red}
Full	expression ~ age * genotype	$p_{Full} = 4$	RSS_{Full}

Full: $y_{ij} = \alpha_0 + \tau_{KO} x_{ij,KO} + \tau_{Age} x_{ij,Age} + \tau_{KO:Age} x_{ij,KO} x_{ij,Age}$

Reduced: $y_{ij} = \alpha_0 + \tau_{Age} x_{ij,Age}$

Under H_0 : the reduced model explains the same amount variation in the outcome as the full,

$$F = \frac{\frac{RSS_{Red} - RSS_{Full}}{p_{Full} - p_{Red}}}{\frac{RSS_{Full}}{n - p_{Full}}} \sim F_{p_{Fill} - p_{Red},\, n - p_{Full}}$$

A significant F-test means we reject the null; we have evidence that the full model explains significantly more variation in the outcome than the reduced.

Linear regression summary

- linear model framework is extremely general
- one extreme (simple): two-sample common variance t-test
- another extreme (flexible): a polynomial, potentially different for each level of some factor
  - dichotomous variable? 👍
  - categorical variable? 👍
  - quantitative variable? 👍
  - various combinations of the above? 👍
- Don't be afraid to build models with more than 1 covariate

What about the other 45 thousand probesets??

```
eset
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 45101 features, 39 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM92610 GSM92611 ... GSM92648 (39 total)
##   varLabels: title geo_accession ... age (40 total)
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
##   pubMedIds: 16505381
## Annotation: GPL1261
```

Linear regression of many genes

$$\mathbf{Y}_g = \mathbf{X}_g \boldsymbol{\alpha}_g + \boldsymbol{\varepsilon}_g$$

- The g in the subscript reminds us that we'll be fitting a model like this *for each gene g* that we have measured for all samples
- Most of the time, the design matrices $\mathbf{X}_g$ are, in fact, the same for all g. This means we can just use $\mathbf{X}$
- Note this means the residual degrees of freedom are also the same for all g

$$d_g = d = n - \textbf{dimension of } \boldsymbol{\alpha} = n - p$$

Linear regression of many genes (cont'd)

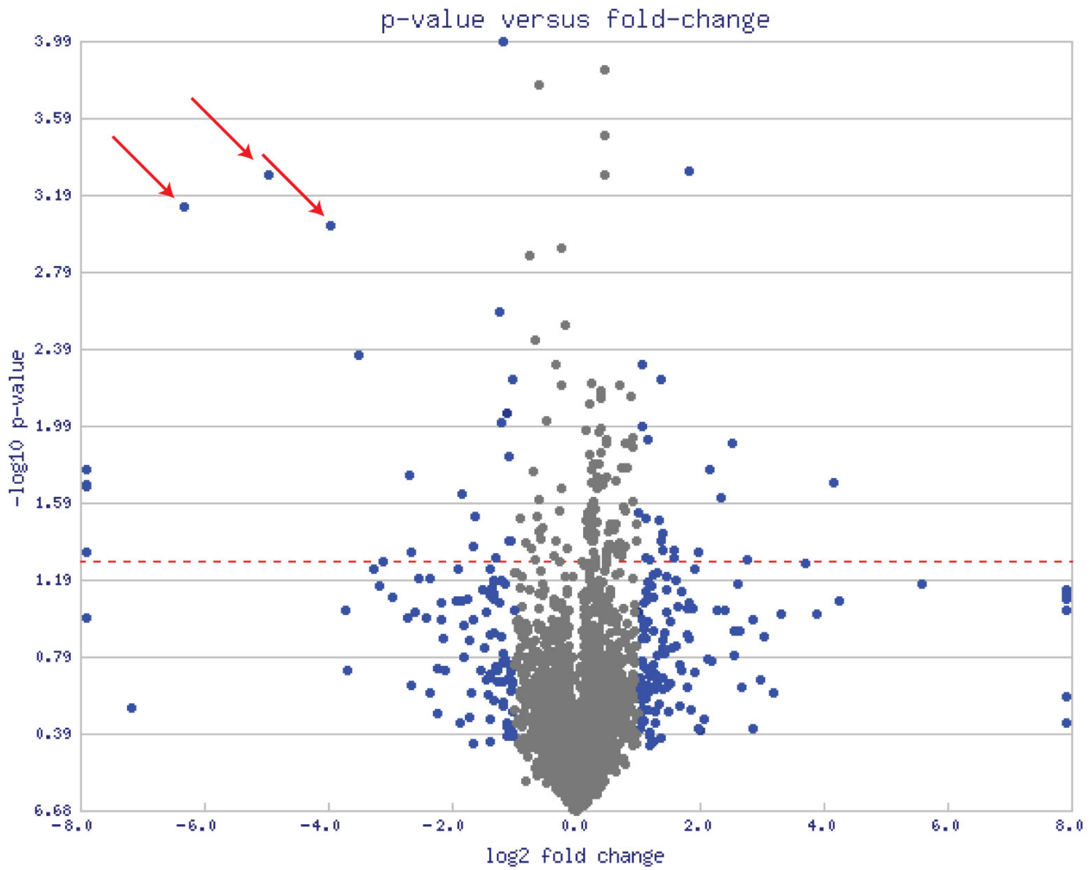
- Data model: $\mathbf{Y}_g = \mathbf{X}\boldsymbol{\alpha}_g + \boldsymbol{\varepsilon}_g$
- Unknown error variance: $Var(\boldsymbol{\varepsilon}_g) = \sigma_g^2$
- Estimated error variance: $\hat{\sigma}_g^2 = s_g^2 = \frac{1}{n-p}\hat{\boldsymbol{\varepsilon}_g}^T\hat{\boldsymbol{\varepsilon}}_g$
- Estimated variance of parameter estimates: $\hat{Var}(\hat{\boldsymbol{\alpha}}_g) = s_g^2(\mathbf{X}^T\mathbf{X})^{-1} = s_g^2\mathbf{V}$
  - $\mathbf{V}$ is the "unscaled covariance" matrix, and is the same for all genes!
  - Estimated standard errors for estimated regression coefficients: $\hat{se}(\hat{\alpha}_{jg})$ obtained by taking the square root of the j^{th} diagonal element of $\hat{Var}(\hat{\boldsymbol{\alpha}}_g)$, which is $s_g\sqrt{v_{jj}}$

So far, nothing is new - these are the "regular" *t* statistics for gene *g* and parameter *j*:

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{s_g \sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

But there are *so* many of them!! 😲

Observed (i.e. empirical) issues with the "standard" *t*-test approach for assessing differential expression



Observed (i.e. empirical) issues with the "standard" *t*-test approach for assessing differential expression

Some genes with very **small p-values** (i.e. large -log10 p-values) are not **biologically meaningful** (small effect size, e.g. fold change)

How do we end up with small p-values but subtle effects?

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g\sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

How do we end up with small p-values but subtle effects?

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g\sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

- Small variance estimate s_g leads to large t statistic $\rightarrow$ small p-value

How do we end up with small p-values but subtle effects?

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g \sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

- Small variance estimate s_g leads to large t statistic $\rightarrow$ small p-value
- Recall: estimates of sample variance from small sample sizes tend to underestimate the true variance!

How do we end up with small p-values but subtle effects?

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g\sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

- Small variance estimate s_g leads to large t statistic $\rightarrow$ small p-value
- Recall: estimates of sample variance from small sample sizes tend to under-estimate the true variance!
- This has led to the development of specialized methodology for assessing genome-wide differential expression

Empirical Bayesian techniques: `limma`

> Stat Appl Genet Mol Biol, 3, Article3 2004

Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments

Gordon K Smyth [1]
Affiliations + expand

PMID: 16646809 DOI: 10.2202/1544-6115.1027

Smyth 2004

Bioconductor
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Home Install Help

Home » Bioconductor 3.10 » Software Packages » limma

limma

platforms all | rank 12 / 1823 | posts 77 / 1 / 2 / 15 | in Bioc > 14.5 years
build ok | updated before release | dependencies 5

DOI: 10.18129/B9.bioc.limma

Linear Models for Microarray Data

Bioconductor version: Release (3.10)

Data analysis, linear models and differential expression for microarray data.

Author: Gordon Smyth [cre,aut], Yifang Hu [ctb], Matthew Ritchie [ctb], Jeremy Silver [ctb], James Wettenhall [ctb], Davis McCarthy [ctb], Di Wu [ctb], Wei Shi [ctb], Belinda Phipson [ctb], Aaron Lun [ctb], Natalie Thorne [ctb], Alicia Oshlack [ctb], Carolyn de Graaf [ctb], Yunshun Chen [ctb], Mette Langaas [ctb], Egil Ferkingstad [ctb], Marcus Davy [ctb], Francois Pepin [ctb], Dongseok Choi [ctb]

Maintainer: Gordon Smyth <smyth at wehi.edu.au>

Why use `limma` instead of regular *t*-tests?

- Borrows information from all genes to get a better estimate of the variance (especially in smaller sample size settings)

Why use `limma` instead of regular *t*-tests?

- Borrows information from all genes to get a better estimate of the variance (especially in smaller sample size settings)
- Efficiently fits many regression models without replicating unnecessary calculations!

Why use `limma` instead of regular *t*-tests?

- Borrows information from all genes to get a better estimate of the variance (especially in smaller sample size settings)

- Efficiently fits many regression models without replicating unnecessary calculations!

- Arranges output in a convenient way to ease further analysis, visualization, and interpretation

How does Empirical Bayes work?

- Empirical: observed
- Bayesian: incorporate 'prior' information
- Intuition: estimate prior information from data; *shrink* (nudge) all estimates toward the consensus

Shrinkage = borrowing information across all genes

Genome-wide OLS fits

- Gene by gene:
  - `lm(y ~ x, data = gene)` for each gene
  - For example, using `dplyr::group_modify` and `broom::tidy`

Genome-wide OLS fits

- Gene by gene:
  - `lm(y ~ x, data = gene)` for each gene
  - For example, using `dplyr::group_modify` and `broom::tidy`
- All genes at once, using `limma`:
  - `lmFit(myDat, desMat)`
  - `myDat` matrix-like object with expression values for all genes
  - `desMat` is a specially formatted design matrix (more on this later)
  - Or, even better, `lmFit(eset, desMat)` where `eset` is an [`ExpressionSet`]() object

'Industrial scale' model fitting is good, because computations involving just the design matrix $\mathbf{X}$ are not repeated tens of thousands of times unnecessarily:

- OLS estimator:

$$\hat{\boldsymbol{\alpha}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Fitted/predicted values:

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$$

OLS of first 2000 genes, using `lm` gene by gene

```
allGenes %>% head(10)
```

```
## # A tibble: 10 × 6
##    gene       sample_id expression dev_stage   age
genotype
##    <chr>      <chr>          <dbl> <fct>     <dbl> <fct>
##  1 1415670_at GSM92610        7.11 4W           28 NrlKO
##  2 1415670_at GSM92611        7.32 4W           28 NrlKO
##  3 1415670_at GSM92612        7.42 4W           28 NrlKO
##  4 1415670_at GSM92613        7.35 4W           28 NrlKO
##  5 1415670_at GSM92614        7.24 E16          -4 NrlKO
##  6 1415670_at GSM92615        7.34 E16          -4 NrlKO
##  7 1415670_at GSM92616        7.38 E16          -4 NrlKO
##  8 1415670_at GSM92617        7.22 P10          10 NrlKO
##  9 1415670_at GSM92618        7.22 P10          10 NrlKO
## 10 1415670_at GSM92619        7.12 P10          10 NrlKO
```

```
system.time(lmfits <- allGenes %>%
    filter(gene %in% unique(allGenes$gene)[1:2000]) %>%
    group_by(gene) %>%
    group_modify(~ tidy(lm(expression ~ age + genotype,
                           data = .x))) %>%
    select(gene, term, estimate) %>%
    pivot_wider(names_from = term, values_from =
estimate))
```

```
##    user  system elapsed
##   8.491   0.064   8.597
```

```
lmfits %>% head() %>% as.data.frame()
```

```
##          gene (Intercept)           age genotypeNrlKO
## 1   1415670_at    7.217851  0.0006225228 -0.0002861005
## 2   1415671_at    9.320083 -0.0018405479  0.1446053811
## 3   1415672_at    9.759959 -0.0039281143 -0.0421705559
## 4   1415673_at    8.404053  0.0039777804 -0.0436443351
## 5 1415674_a_at    8.517675 -0.0059840405  0.0192159017
## 6   1415675_at    9.665691 -0.0064185855  0.1330272055
```

OLS of all genes at once, using `limma`:

```
system.time( limmafits <-
  lmFit(eset, model.matrix(~ age + genotype, data = pData(eset))))
```

```
##    user  system elapsed
##   0.108   0.028   0.136
```

```
limmafits$coefficients %>% head()
```

```
##              (Intercept)           age genotypeNrlKO
## 1415670_at      7.217851  0.0006225228 -0.0002861005
## 1415671_at      9.320083 -0.0018405479  0.1446053811
## 1415672_at      9.759959 -0.0039281143 -0.0421705559
## 1415673_at      8.404053  0.0039777804 -0.0436443351
## 1415674_a_at    8.517675 -0.0059840405  0.0192159017
## 1415675_at      9.665691 -0.0064185855  0.1330272055
```

OLS of all genes at once, using `limma`:

```
system.time( limmafits <-
  lmFit(eset, model.matrix(~ age + genotype, data = pData(eset))))
```

```
##    user  system elapsed
##   0.108   0.028   0.136
```

```
limmafits$coefficients %>% head()
```

```
##              (Intercept)           age genotypeNrlKO
## 1415670_at      7.217851  0.0006225228 -0.0002861005
## 1415671_at      9.320083 -0.0018405479  0.1446053811
## 1415672_at      9.759959 -0.0039281143 -0.0421705559
## 1415673_at      8.404053  0.0039777804 -0.0436443351
## 1415674_a_at    8.517675 -0.0059840405  0.0192159017
## 1415675_at      9.665691 -0.0064185855  0.1330272055
```

So far, no shrinkage.

How can we better estimate the SE?

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g\sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

Under-estimated variance leads to overly large t statistic, which leads to artificially small p-value

Modeling in `limma`

limma assumes that for each gene g

$$\hat{\alpha}_{gj} \,|\, \alpha_{gj}, \sigma_g^2 \sim N(\alpha_{gj}, \sigma_g^2 v_{jj})$$

$$s_g^2 \,|\, \sigma_g^2 \sim \frac{\sigma_g^2}{d} \chi_d^2$$

which are the same as the usual assumptions about ordinary t-statistics:

$$t_{gj} = \frac{\hat{\alpha}_{gj}}{\hat{se}(\hat{\alpha}_{gj})} = \frac{\hat{\alpha}_{gj}}{s_g \sqrt{v_{jj}}} \sim t_d \text{ under } H_0$$

So far, nothing new...

Modeling in `limma` - shrinkage

- limma imposes a hierarchical model, which describes how the gene-wise α_{gj}'s and σ_g^2's vary across the genes
    - We are no longer considering genes in isolation
- this is done by assuming a prior distribution for those quantities
- Prior distribution for gene-specific variances σ_g^2: an inverse Chi-square with mean s_0^2 and d_0 degrees of freedom:

$$\frac{1}{\sigma_g^2} \sim \frac{1}{d_0 s_0^2} \chi^2_{d_0}$$

Modeling in `limma` - shrinkage

- limma imposes a hierarchical model, which describes how the gene-wise α_{gj}'s and σ_g^2's vary across the genes
    - We are no longer considering genes in isolation
- this is done by assuming a prior distribution for those quantities
- Prior distribution for gene-specific variances σ_g^2: an inverse Chi-square with mean s_0^2 and d_0 degrees of freedom:

$$\frac{1}{\sigma_g^2} \sim \frac{1}{d_0 s_0^2} \chi_{d_0}^2$$

- this should feel a bit funny compared to previous lectures - σ_g^2 is no longer a fixed quantity! (i.e. this is Bayesian)

How does this help us better estimate the variance?

- The posterior distribution is an updated version of the observed likelihood based on incorporating the prior information

- The posterior mean for gene-specific variance:

$$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$$

How does this help us better estimate the variance?

- The posterior distribution is an updated version of the observed likelihood based on incorporating the prior information

- The posterior mean for gene-specific variance:

$$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$$

- How to think about it: a weighted mean of the prior (indirect evidence) and the observed (direct evidence) gene-specific variances:

$$\tilde{s}_g^2 = \frac{d_0}{d_0 + d} s_0^2 + \frac{d}{d_0 + d} s_g^2$$

How does this help us better estimate the variance?

- The posterior distribution is an updated version of the observed likelihood based on incorporating the prior information

- The posterior mean for gene-specific variance:

$$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$$

- How to think about it: a weighted mean of the prior (indirect evidence) and the observed (direct evidence) gene-specific variances:

$$\tilde{s}_g^2 = \frac{d_0}{d_0 + d} s_0^2 + \frac{d}{d_0 + d} s_g^2$$

- More simply: "shrinking" the observed gene-specific variance towards the "typical" variance implied by the prior

Moderated *t* statistic

- plug in this posterior mean estimate to obtain a 'moderated' *t* statistic:

$$\tilde{t}_{gj} = \frac{\hat{\alpha}_{gj}}{\tilde{s}_g \sqrt{v_{jj}}}$$

- Under limma's assumptions, we know the null distribution for the moderated *t*-statistic under H_0:

$$\tilde{t}_{gj} \sim t_{d_0+d}$$

- parameters from the prior d_0 and s_0^2 are estimated from the data

Moderated *t* statistic

- plug in this posterior mean estimate to obtain a 'moderated' *t* statistic:

$$\tilde{t}_{gj} = \frac{\hat{\alpha}_{gj}}{\tilde{s}_g \sqrt{v_{jj}}}$$

- Under limma's assumptions, we know the null distribution for the moderated *t*-statistic under H_0:

$$\tilde{t}_{gj} \sim t_{d_0+d}$$

- parameters from the prior d_0 and s_0^2 are estimated from the data

- This is how limma is a hybrid of frequentist (*t*-statistic) and Bayesian (hierarchical model) approaches (i.e. empirical Bayes)

Side-by-side comparison of key quantities and results

	OLS	limma
Estimated gene-wise residual variance:	$s_g^2 = \frac{1}{n-p}\hat{\boldsymbol{\varepsilon}}^T\hat{\boldsymbol{\varepsilon}}$	$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$
t-statistic for $H_0 : \alpha_{gj} = 0$:	$t_{gj} = \frac{\hat{\alpha}_{gj}}{s_g\sqrt{v_{jj}}}$	$\tilde{t}_{gj} = \frac{\hat{\alpha}_{gj}}{\tilde{s}_g\sqrt{v_{jj}}}$
distribution of the t-statistic under H_0:	$t_{gj} \sim t_d$	$\tilde{t}_{gj} \sim t_{d_0+d}$

*Not shown: estimation formulas for prior parameters d_0 and s_0^2

Moderated vs traditional tests

- moderated variances will be *"shrunk" toward the typical gene-wise variance*, relative to to raw sample residual variances

Moderated vs traditional tests

- moderated variances will be *"shrunk" toward the typical gene-wise variance*, relative to to raw sample residual variances

- degrees of freedom for null distribution *increase* relative to default $(d \text{ vs } d_0 + d)$

    - $\rightarrow$ makes it closer to a standard normal
    - $\rightarrow$ makes tail probabilities (p-values) smaller
    - $\rightarrow$ easier to reject the null

Moderated vs traditional tests

- moderated variances will be *"shrunk" toward the typical gene-wise variance*, relative to to raw sample residual variances

- degrees of freedom for null distribution *increase* relative to default $(d \text{ vs } d_0 + d)$

  - $\rightarrow$ makes it closer to a standard normal
  - $\rightarrow$ makes tail probabilities (p-values) smaller
  - $\rightarrow$ easier to reject the null

- overall, when all is well, limma will deliver statistical results that are more stable and more powerful

Preview: `limma` workflow

responses, design matrix (made by YOU)

fit a separate linear model for each response, e.g. gene — `lmFit(...)`

fitted models

apply an Empirical Bayes procedure for moderating estimates of error variance — `eBayes(...)`

extract estimated parameters or p-values or ...
compare big models to small
etc etc — `topTable(...)`