Lecture 8 – Continuous models and intro to limma

STAT/BIOF/GSAT 540: Statistical Methods for High Dimensional Biology

Keegan Korthauer

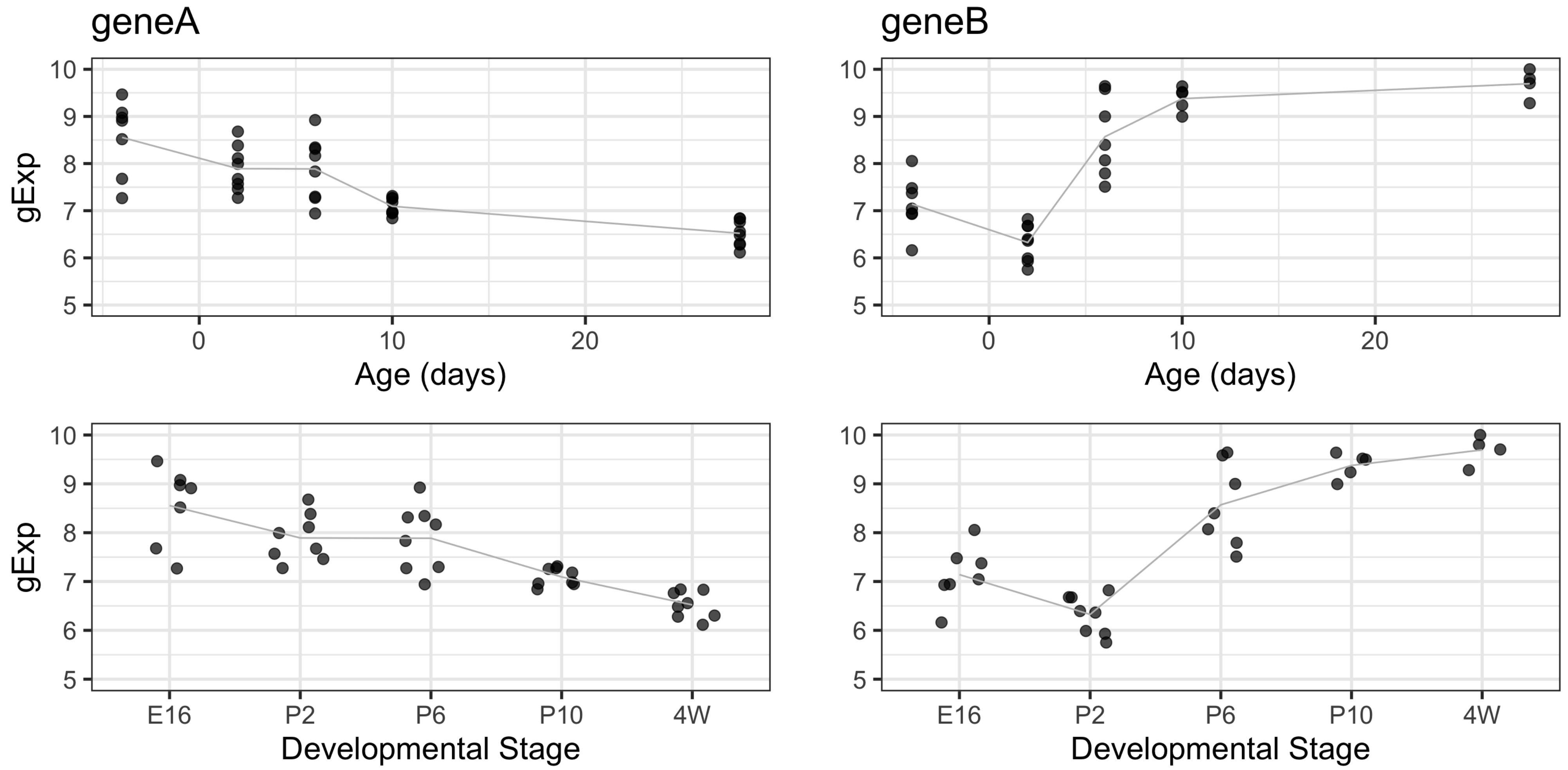
2020/01/29

Slides by: Gabriela Cohen Freue with contributions from Jenny Bryan, Keegan Korthauer, and Sara Mostafavi

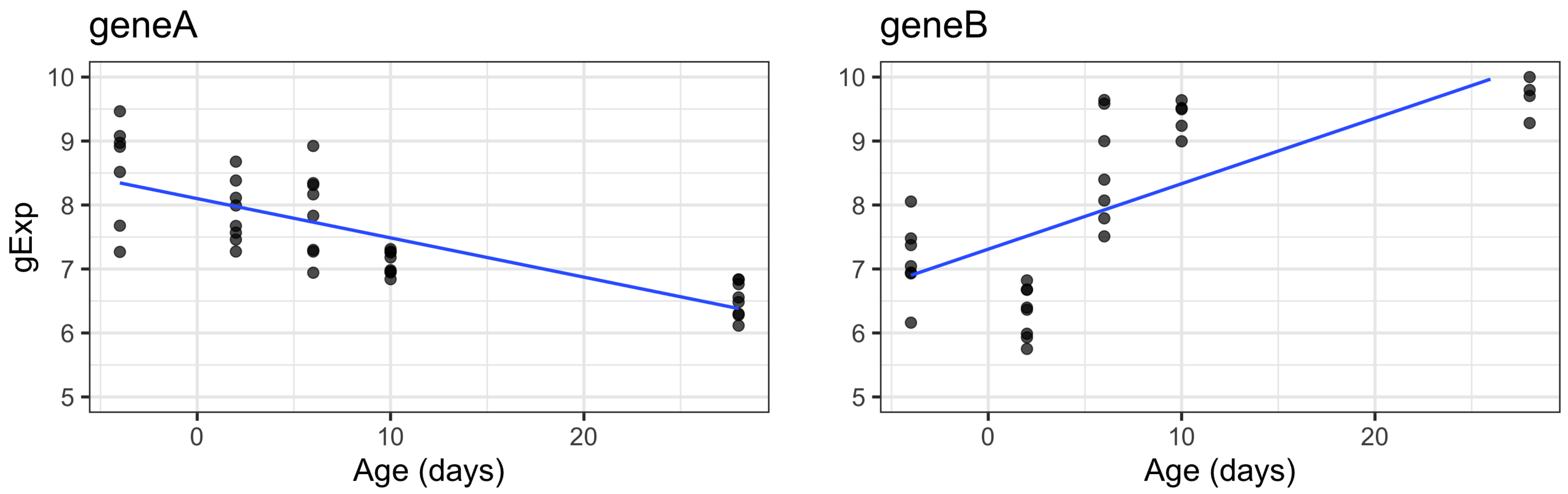
Summary so far

- 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)

What if we represent Age as a continuous variable?



Linear model with Age as continuous covariate



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

Plain vanilla linear model (Matrix formulation)

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

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)

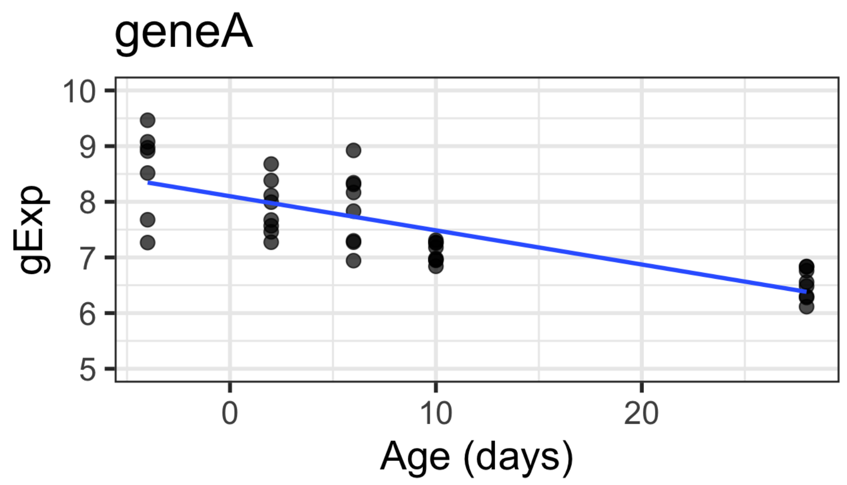
$$\mathbf{Y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$$

Remember / convince yourself that the matrix algebra does indeed reproduce simple linear regression:

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

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

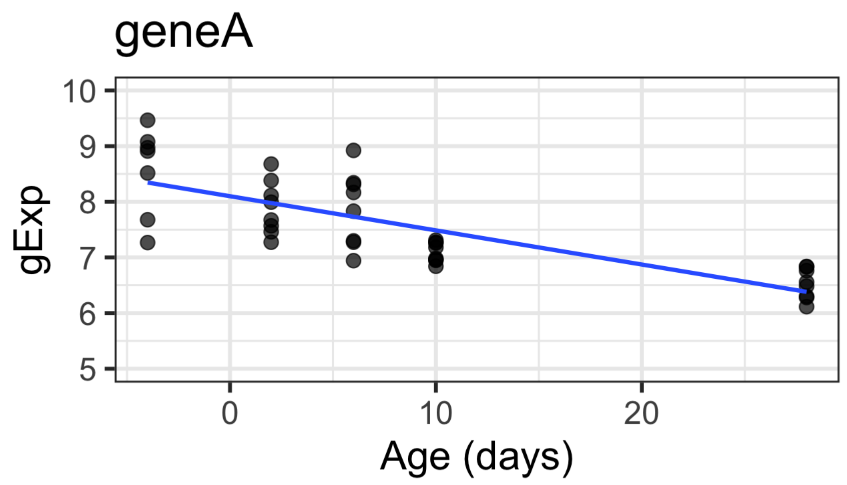
$$\Rightarrow y_i = \alpha_0 + x_i\alpha_1 + \varepsilon_i$$



```
summary(lm(gExp ~ Age, data=devDat %>% filter(gene == "geneA")))
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.100126   0.113959   71.08  < 2e-16 ***
## Age         -0.061373   0.008216   -7.47 6.76e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5536 on 37 degrees of freedom
## Multiple R-squared:  0.6013,    Adjusted R-squared:  0.5905
## F-statistic: 55.81 on 1 and 37 DF,  p-value: 6.757e-09
```

$H_0 : \alpha_0 = 0$ (whether intercept is zero - usually, not of interest)



```r
summary(lm(gExp ~ Age, data=devDat %>% filter(gene == "geneA")))
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  8.100126   0.113959   71.08  < 2e-16 ***
## Age         -0.061373   0.008216   -7.47 6.76e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5536 on 37 degrees of freedom
## Multiple R-squared:  0.6013,    Adjusted R-squared:  0.5905 
## F-statistic: 55.81 on 1 and 37 DF,  p-value: 6.757e-09
```

$H_0 : \alpha_1 = 0$ (tests association between gene expression and age)

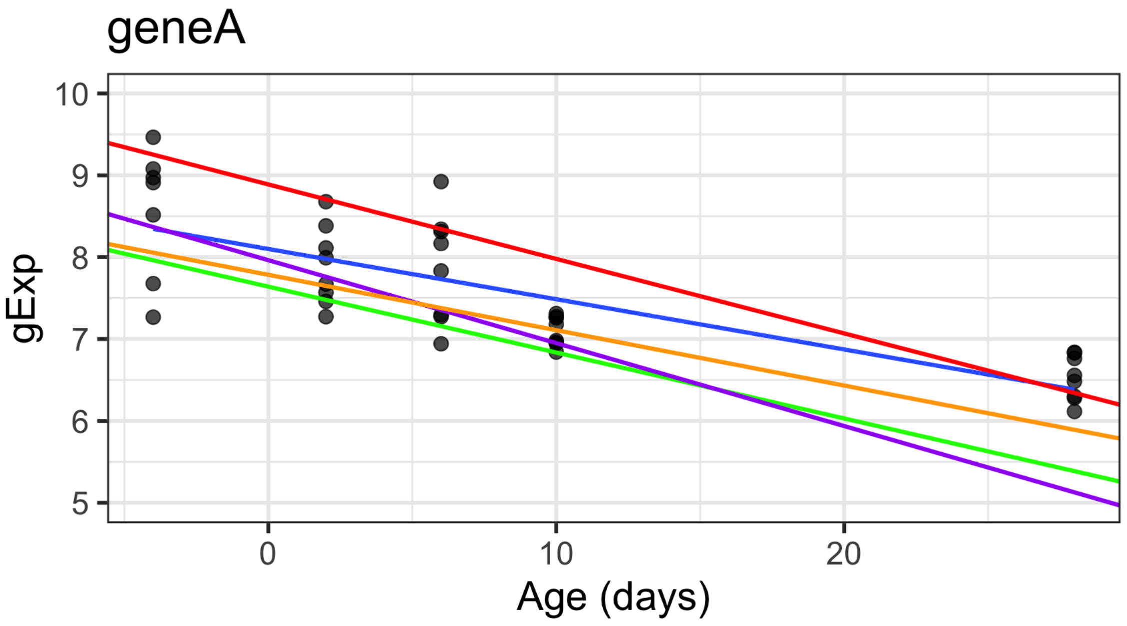
How do we estimate the intercept and slope?

Is there an *optimal* line?

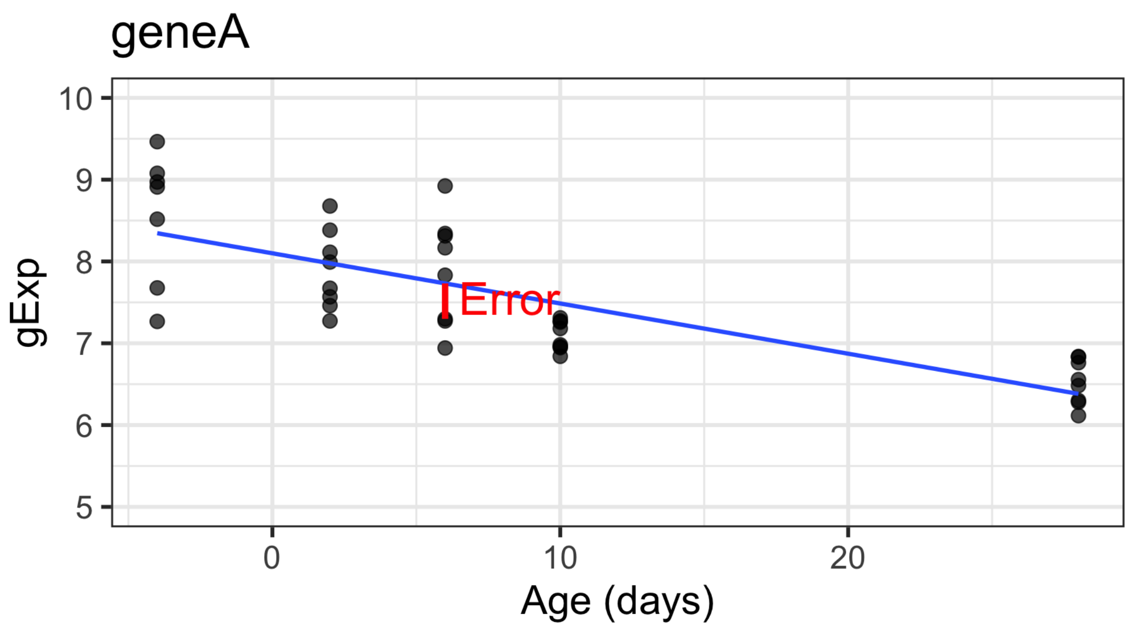
```r
summary(lm(gExp ~ Age, data=devDat %>% filter(gene == "geneA")))
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.100126   0.113959   71.08  < 2e-16 ***
## Age         -0.061373   0.008216   -7.47 6.76e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5536 on 37 degrees of freedom
## Multiple R-squared:  0.6013,    Adjusted R-squared:  0.5905
## F-statistic: 55.81 on 1 and 37 DF,  p-value: 6.757e-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 fitted line and the real observation

OLS interactive demo

- Visual representation of the squared errors in OLS: http://setosa.io/ev/ordinary-least-squares-regression/
- The squares of the errors are represented by the square areas in the second plot
    - select different lines by changing the intercept and slope
    - see how the squares of the errors change
    - which line minimizes the sum of these areas? OLS answers this question
- Move a point in the first plot; observe how sensitive the OLS estimation is

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

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

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

OLS Estimator 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:

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

Properties of OLS regression

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

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

Assumptions:

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

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

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 (cont'd)

Residuals: (note NOT the same as errors ε)

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

Estimated error variance:

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

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

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

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

Inference in Regression (normal iid errors)

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

With a *t* statistic!

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 Normality of errors?

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

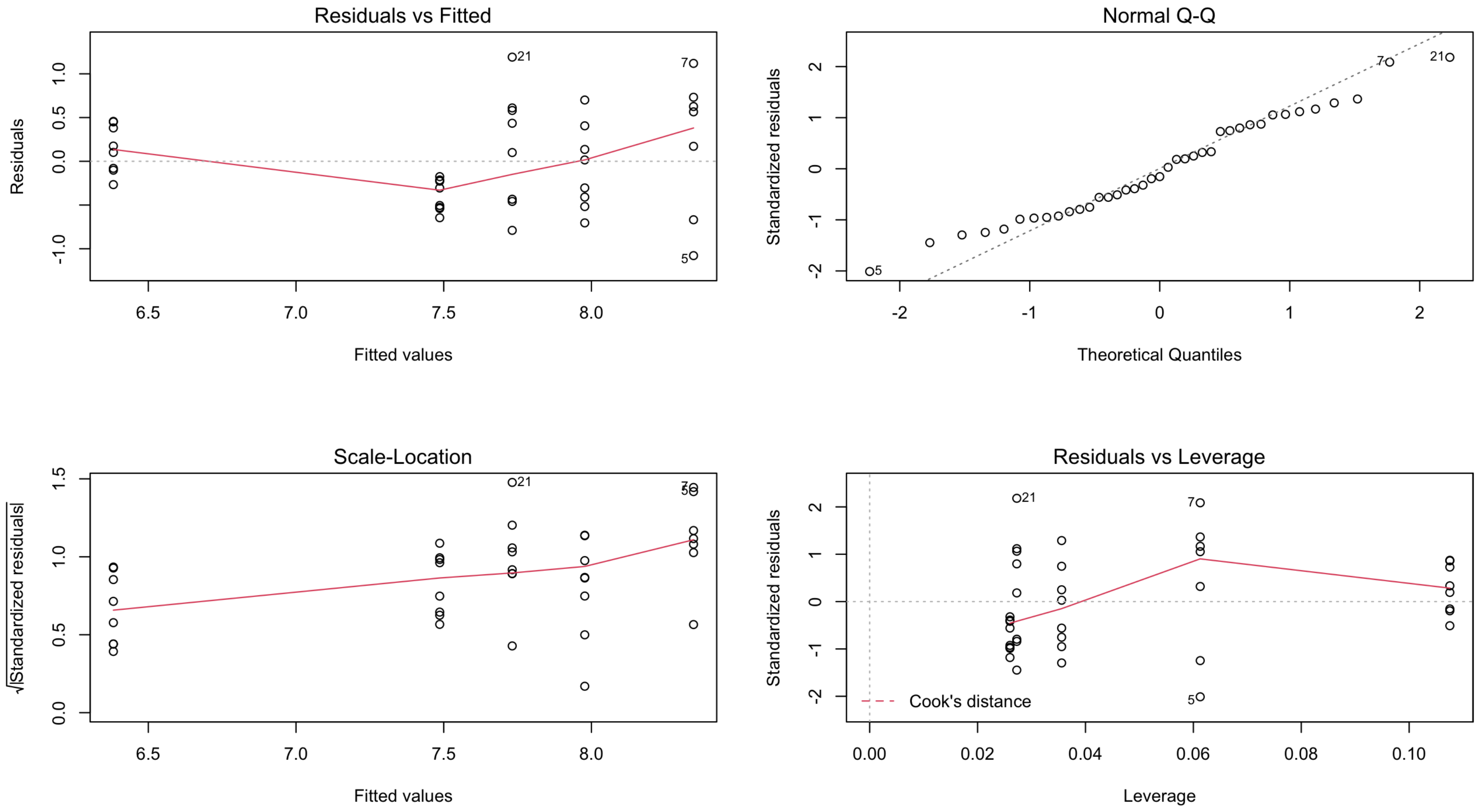
With a *t* statistic!

Under H_0, asymptotically (by CLT)

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

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

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



Linear regression

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

$$y_i = \frac{1}{1 + e^{(\phi - x_i)/\eta}} + \varepsilon$$

- We just did 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)

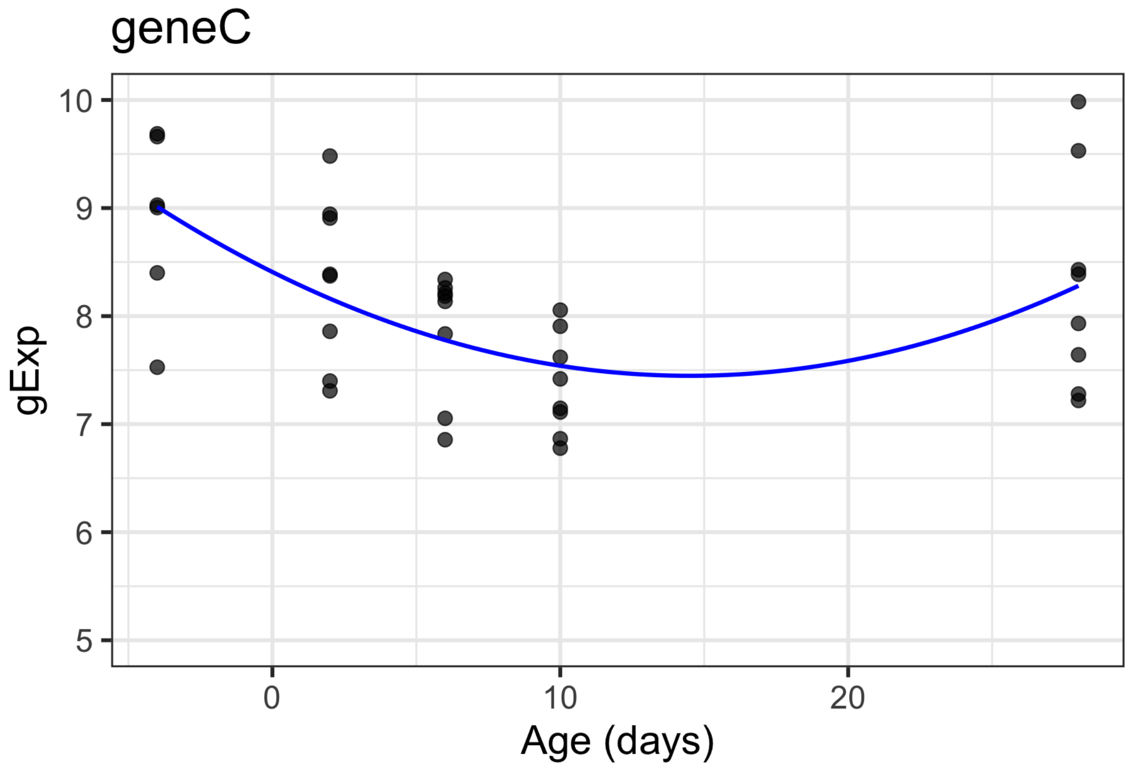
$$y_i = \alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \varepsilon_i$$

Polynomial regression

```
quadfit <- lm(gExp ~ Age + I(Age^2), data=geneC)
summary(quadfit)
```

```
##
## Call:
## lm(formula = gExp ~ Age + I(Age^2), data = geneC)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6250 -0.6437  0.1027  0.4956  1.6997
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.482528   0.160882  52.725  < 2e-16 ***
## Age         -0.147335   0.032626  -4.516 6.53e-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: 0.000307
```

Polynomial regression



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

Putting it all together (continuous + categorical variables)

```
summary(lm(gExp ~ Age*gType, data=devDat %>% filter(gene=="geneA")))
```

```
##
## Call:
## lm(formula = gExp ~ Age * gType, data = devDat %>% filter(gene ==
##     "geneA"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13324 -0.38398 -0.00233  0.31710  1.08867
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     8.031528   0.156571  51.297  < 2e-16 ***
## Age            -0.066444   0.011419  -5.819 1.34e-06 ***
## gTypeNrlKO      0.142349   0.228278   0.624    0.537
## Age:gTypeNrlKO  0.009853   0.016445   0.599    0.553
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5534 on 35 degrees of freedom
## Multiple R-squared:  0.6231,    Adjusted R-squared:  0.5907
```

Interaction between continuous and categorical variables

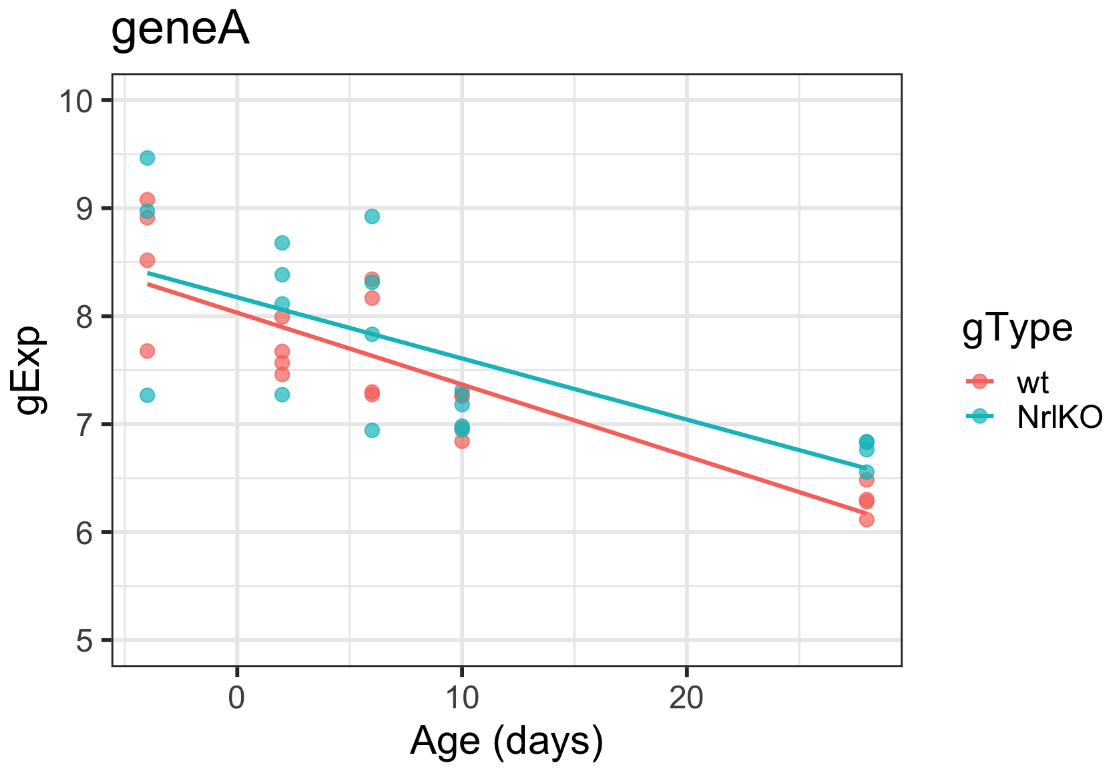
$$y_{ij} = \alpha_0 + \tau_1 x_{ij}^{dummy} + \tau_2 x_{ij}^{Age} + \tau_3 x_{ij}^{dummy} x_{ij}^{Age}$$

where $j \in \{wt, NrlKO\}, i = 1, 2, \ldots, n_j$, and x_{ij}^{dummy} is 1 for $j = NrlKO$ and 0 for $j = wt$

The "intercept" for the knockouts is: $\alpha_0 + \tau_1$

The slope for the knockouts is: $\tau_2 + \tau_3$

Interaction between continuous and categorical variables



Interaction between continuous and categorical variables

```
summary(lm(gExp ~ Age*gType, data=devDat %>% filter(gene=="geneA")))$coef
```

```
##                     Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)      8.031527929 0.15657060 51.2965249 1.574727e-34
## Age             -0.066443801 0.01141908 -5.8186625 1.338596e-06
## gTypeNrlKO       0.142348659 0.22827782  0.6235764 5.369488e-01
## Age:gTypeNrlKO   0.009852783 0.01644510  0.5991317 5.529437e-01
```

`(Intercept)`: Intercept of wt line

`Age`: slope of wt line

`gTypeNrlKO`: difference in intercepts (KO vs wt)

`Age:gTypeNrlKO`: difference in slopes (KO vs wt)

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(gExp ~ Age*gType, data=devDat %>% filter(gene=="geneA")),
      lm(gExp ~ Age, data=devDat %>% filter(gene=="geneA")))
```

```
## Analysis of Variance Table
##
## Model 1: gExp ~ Age * gType
## Model 2: gExp ~ Age
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     35 10.720
## 2     37 11.338 -2  -0.61795 1.0088  0.375
```

It's not clear that genotype affects the intercept or the slope

F tests in regression

Model	Example	# params (df)	RSS
small	gExp ~ Age	$p_{small} = 2$	RSS_{small}
big	gExp ~ Age * gType	$p_{big} = 4$	RSS_{big}

$$\text{big: } y_{ij} = \alpha_0 + \tau_1 x_{ij}^{dummy} + \tau_2 x_{ij}^{Age} + \tau_3 x_{ij}^{dummy} x_{ij}^{Age}$$

$$\text{small: } y_{ij} = \alpha_0 + \tau_2 x_{ij}^{Age}$$

$$F = \frac{\frac{RSS_{small} - RSS_{big}}{p_{big} - p_{small}}}{\frac{RSS_{big}}{n - p_{big}}} \sim_{H_0} F_{p_{big} - p_{small},\, n - p_{big}}$$

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? OK!
    - categorical variable? OK!
    - quantitative variable? OK!
    - various combinations of the above? OK!
- Don't be afraid to build models with more than 1 covariate

What about the other 29 thousand probesets??

```
str(prDat)
```

```
## 'data.frame':    29949 obs. of  39 variables:
##  $ Sample_20: num  7.24 9.48 10.01 8.36 8.59 ...
##  $ Sample_21: num  7.41 10.02 10.04 8.37 8.62 ...
##  $ Sample_22: num  7.17 9.85 9.91 8.4 8.52 ...
##  $ Sample_23: num  7.07 10.13 9.91 8.49 8.64 ...
##  $ Sample_16: num  7.38 7.64 8.42 8.36 8.51 ...
##  $ Sample_17: num  7.34 10.03 10.24 8.37 8.89 ...
##  $ Sample_6 : num  7.24 9.71 10.17 8.84 8.54 ...
##  $ Sample_24: num  7.11 9.75 9.39 8.37 8.36 ...
##  $ Sample_25: num  7.19 9.16 10.11 8.2 8.5 ...
##  $ Sample_26: num  7.18 9.49 9.41 8.73 8.39 ...
##  $ Sample_27: num  7.21 8.64 9.43 8.33 8.43 ...
##  $ Sample_14: num  7.09 9.56 9.88 8.57 8.59 ...
##  $ Sample_3 : num  7.16 9.55 9.84 8.33 8.5 ...
##  $ Sample_5 : num  7.08 9.32 9.24 8.3 8.48 ...
##  $ Sample_8 : num  7.11 8.24 9.13 8.13 8.33 ...
##  $ Sample_28: num  7.34 8.27 9.47 8.38 8.4 ...
##  $ Sample_29: num  7.66 10.03 9.88 8.56 8.69 ...
##  $ Sample_30: num  7.26 9.27 10.54 8.15 8.55 ...
##  $ Sample_31: num  7.31 9.26 10.1 8.37 8.49 ...
```

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*
- 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 the residual degrees of freedom

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

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:

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

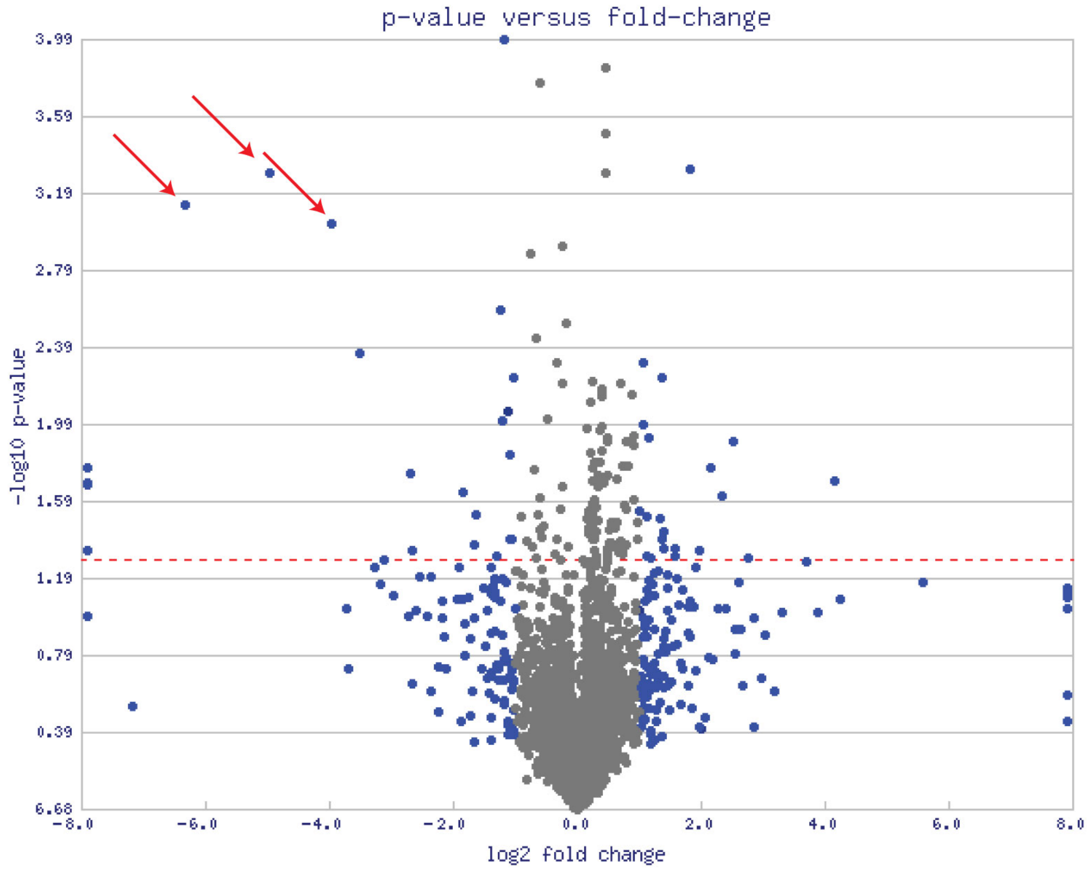
V is the "unscaled covariance", and is the same for all genes!

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

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

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

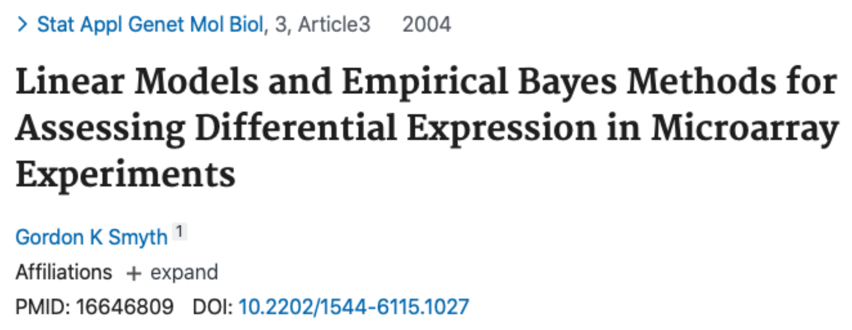
Some genes with very **small p-values** (large -log10 p-values) are not **biologically meaningful** (small effect size)

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

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

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

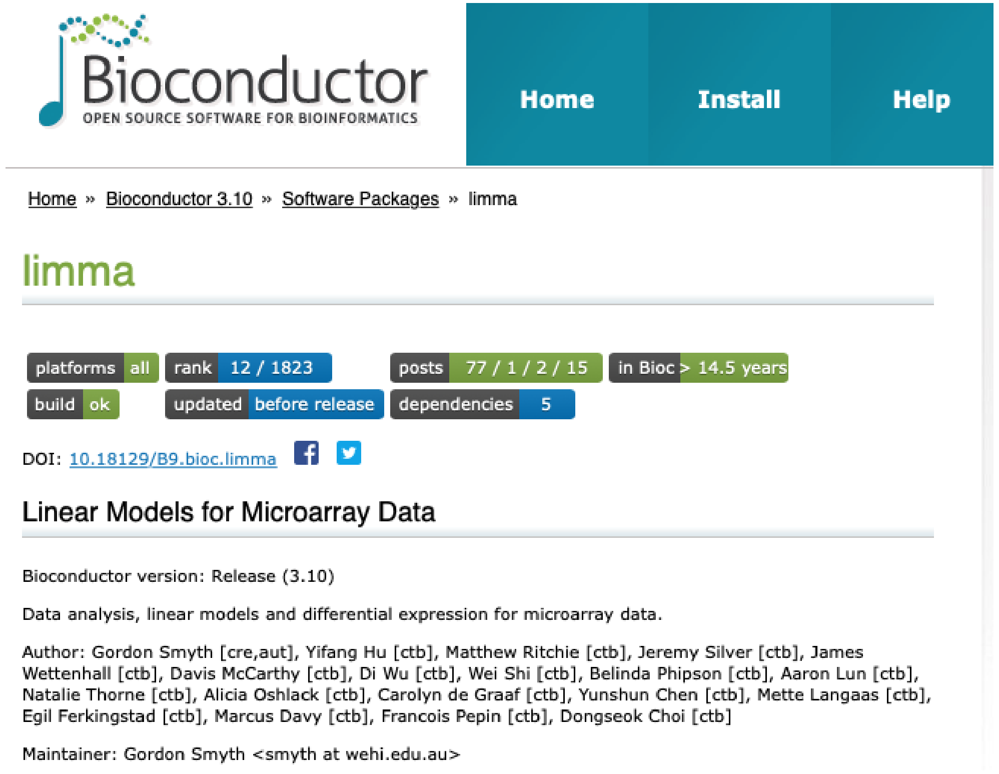
Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments

Gordon K Smyth [1]

Affiliations + expand

PMID: 16646809 DOI: 10.2202/1544-6115.1027

Smyth 2004



Home » Bioconductor 3.10 » Software Packages » limma

limma

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>

eBayes: limma

- **Borrows information** from all genes to get a better estimate of the variance
- Efficiently fits many regression models **without replicating unnecessary calculations**!
- Arranges output in a convenient way to ease further analysis, visualization, and interpretation

Empirical Bayes

Shrinkage = borrowing information across all genes

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

Practically

- Gene by gene (no shrinkage):
    - `lm(y ~ x)` for each gene
    - For example, `by(myDat, gene, lm(y ~ x))`
- All genes at once, using `limma`:
    - `lmFit(myDat, desMat)`
    - `desMat` is a specially formated design matrix (more on this later)

'Industrial scale' model fitting is good, because computations involving just the design matrix $\mathbf{X}$ are not repeated 30K unnecessarily

- **OLS estimator**:

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

- **Fitted/predicted values**:

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

How can we better estimate the SE?

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

Small variance estimate leads to large t statistic, which leads to small p-value

Modeling in `limma`

limma assumes that

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

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

which assumes the usual result about ordinary t-statistics:

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

So far, nothing new…

Modeling in `limma` (cont'd)

- limma imposes a hierarchical model, which describes how the gene-wise α_{gj}'s and σ_g^2's vary *across the genes*
- this is done by assuming a *prior distribution* for those quantities
- **gene-specific variances** σ_g^2: an inverse Chi-square prior with mean s_0^2 and d_0 degrees of freedom

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

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

OK, but how does this help us get a better estimate of the variance?

- The *posterior* (updated based on prior) mean for gene-specific variance:

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

where d_0 and s_0^2 need to be estimated

- 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:

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

- Under limma assumptions, we know the null distribution for the moderated *t*-statistic:

$$\tilde{t}_{gj} \sim t_{d_0+d} \text{ under } H_0$$

- This is how limma is a hybrid of frequentist (*t*-statistic) and Bayesian (hierarchical model) approaches

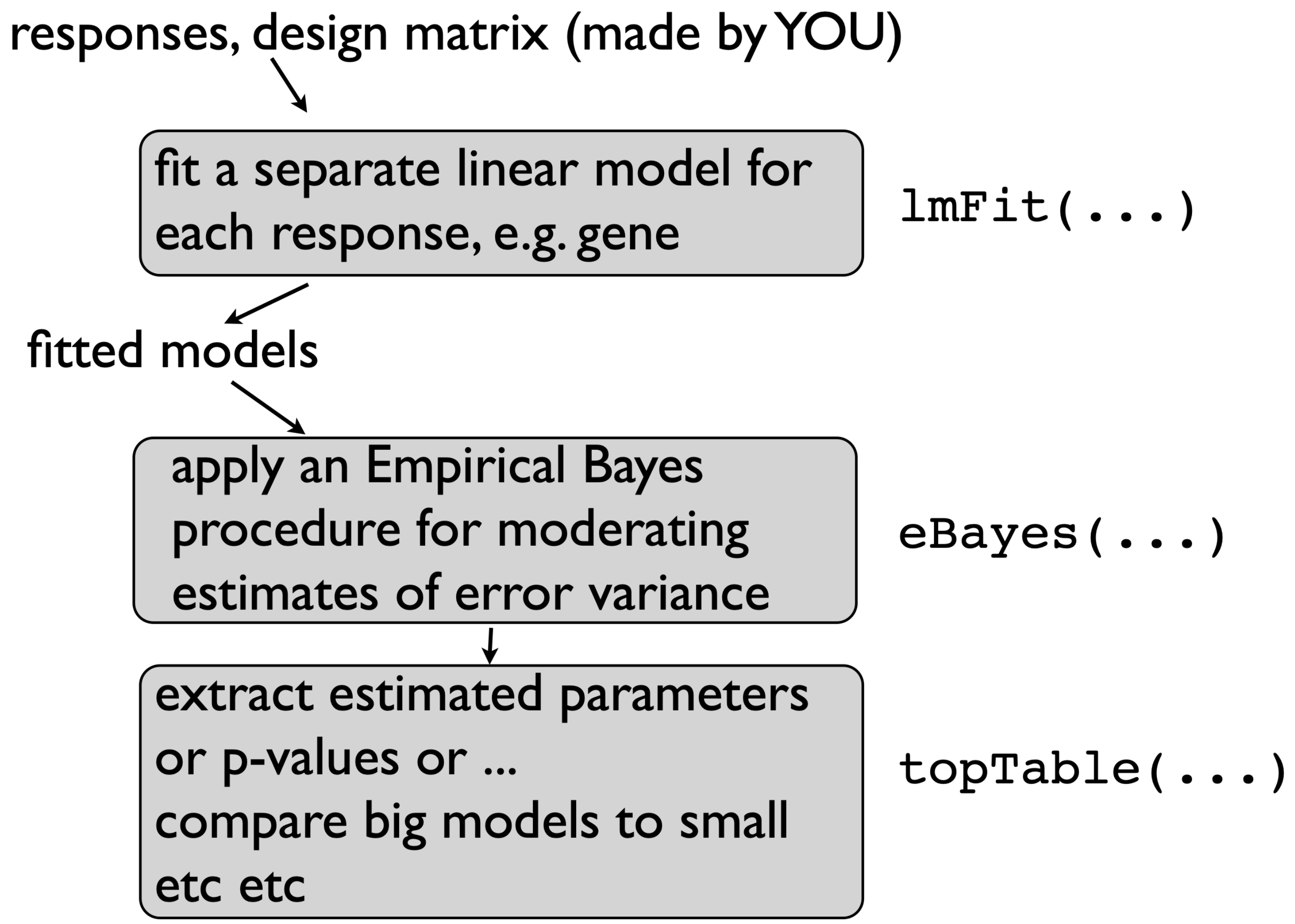
Side-by-side comparison of key quantities and results

	"plain vanilla"	**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 $H_0 : \alpha_{gj} = 0$:	$t_{gj} = \frac{\hat{\alpha}_{gj}}{s_g \sqrt{v_j}}$	$\tilde{t}_{gj} = \frac{\hat{\alpha}_{gj}}{\tilde{s}_g \sqrt{v_j}}$
distribution of the t-statistic under H_0:	$t_{gj} \sim t_d$	$\tilde{t}_{gj} \sim t_{d_0+d}$

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 goes **up** relative to default $d = n - p \rightarrow$ 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*

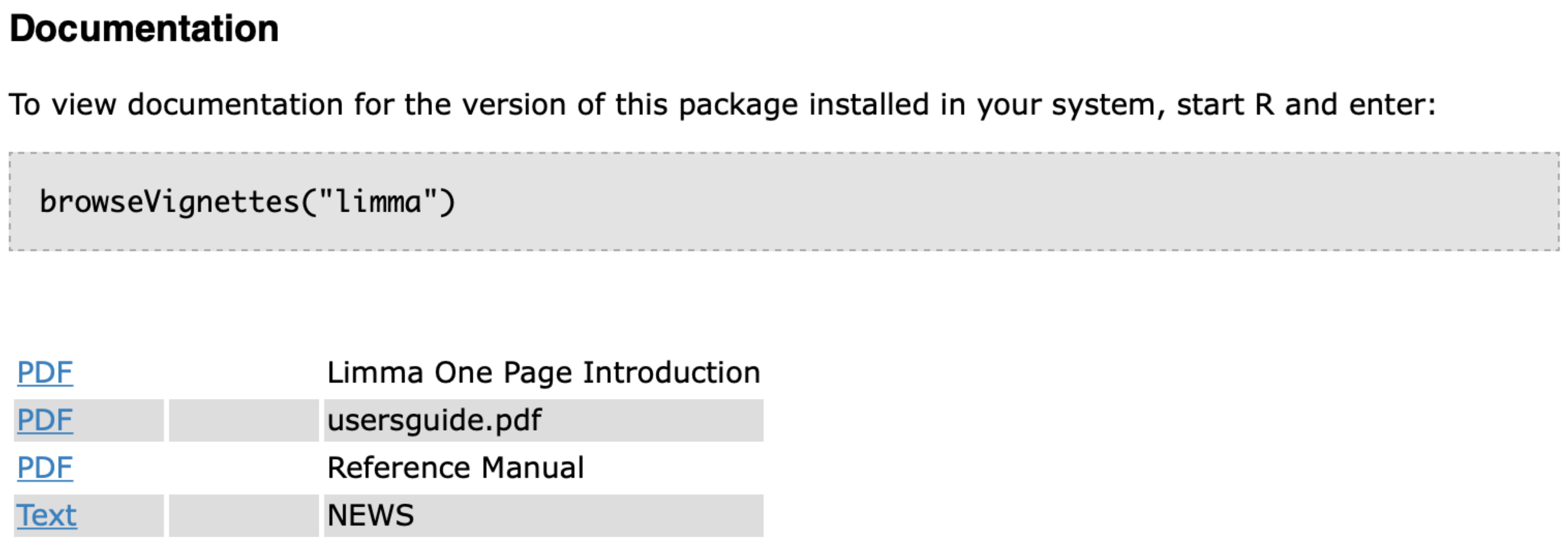
`limma` workflow


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(...)

Functions that make your life easier

Function	Description
`model.matrix`	Takes in your data frame and makes a design matrix
`limma::lmFit`	Fits the linear model to all genes (each gene separately) – replace gene with “feature” depending on your data
`limma::makeContrasts`	Create the contrast matrix C that you desire
`limma::contrast.fit`	Apply a contrast to your estimates
`limma::eBayes`	Use output of linear regression to compute moderated *t* statistics
`limma::topTable`	Query your results; sort your p-values; sort genes; Adjust for multiple comparisons

Getting help



Documentation

To view documentation for the version of this package installed in your system, start R and enter:

```
browseVignettes("limma")
```

PDF		Limma One Page Introduction
PDF		usersguide.pdf
PDF		Reference Manual
Text		NEWS

Bioconductor homepage for limma