Continuous models and intro to `limma`

Keegan Korthauer

February 7, 2023

Laying the foundation of linear models

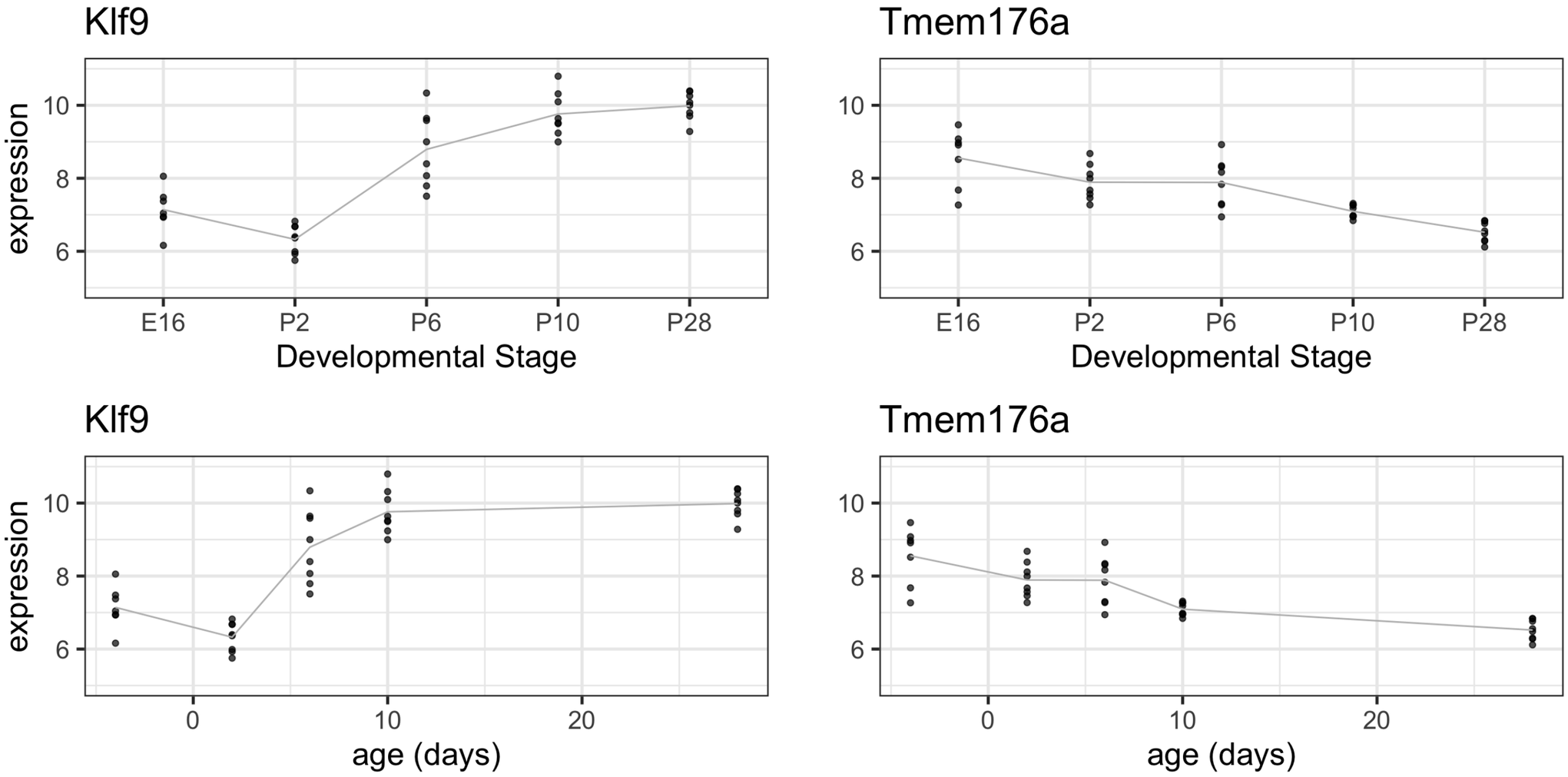
- ***t* -tests** can be used to test the equality of 2 population means
- **ANOVA** can be used to test the equality of more than 2 population means
- **Linear regression** provides a general framework for modeling the relationship between a response variable and different types of explanatory variables
  - *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**)

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`

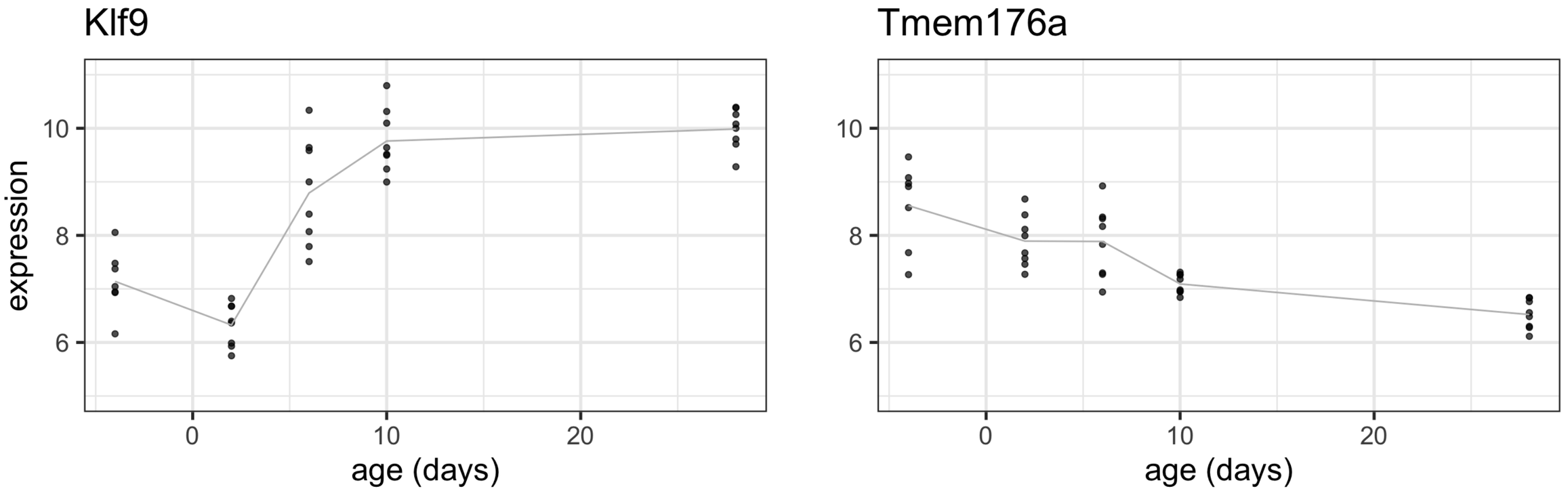
What if we treat age as a continuous variable?

▶ Code



Linear model with age as continuous covariate

▶ Code



- 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:

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

- α_0 = the **intercept** (expected value of y when x is equal to zero)
- α_1 = the **slope** (expected change in y for every one-unit increase in x)

Simple Linear Regression (Matrix form)

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

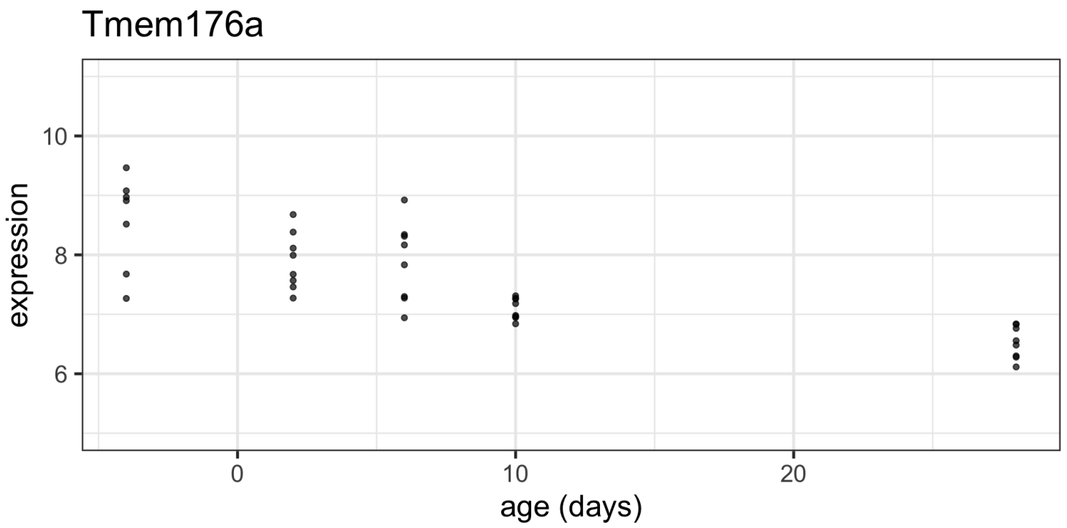
Remember / convince yourself that the matrix algebra yields simple linear equations:

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

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

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

SLR with continuous age covariate



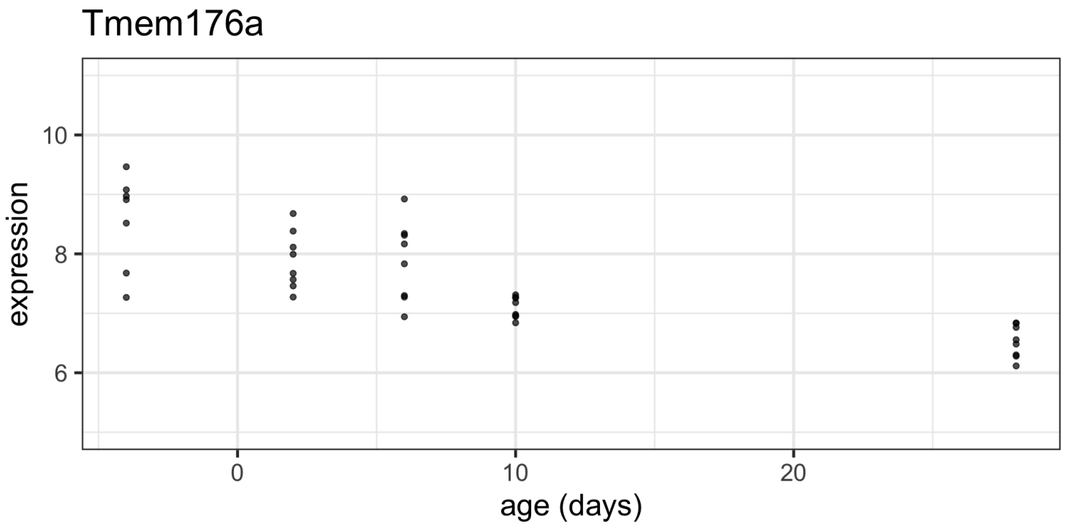

```
1 Tmem_fit <- filter(twoGenes, gene == "Tmem176a") %>%
2   lm(expression ~ age, data = .)
3 tidy(Tmem_fit)
```

```
# A tibble: 2 × 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   8.10      0.114       71.1  3.58e-41
2 age          -0.0614    0.00821     -7.47 6.74e- 9
```

Interpretation of **intercept**:

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

SLR with continuous age covariate




```
1 tidy(Tmem_fit)
```

```
# A tibble: 2 × 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   8.10      0.114       71.1  3.58e-41
2 age          -0.0614    0.00821     -7.47 6.74e- 9
```

Interpretation of **slope**:

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

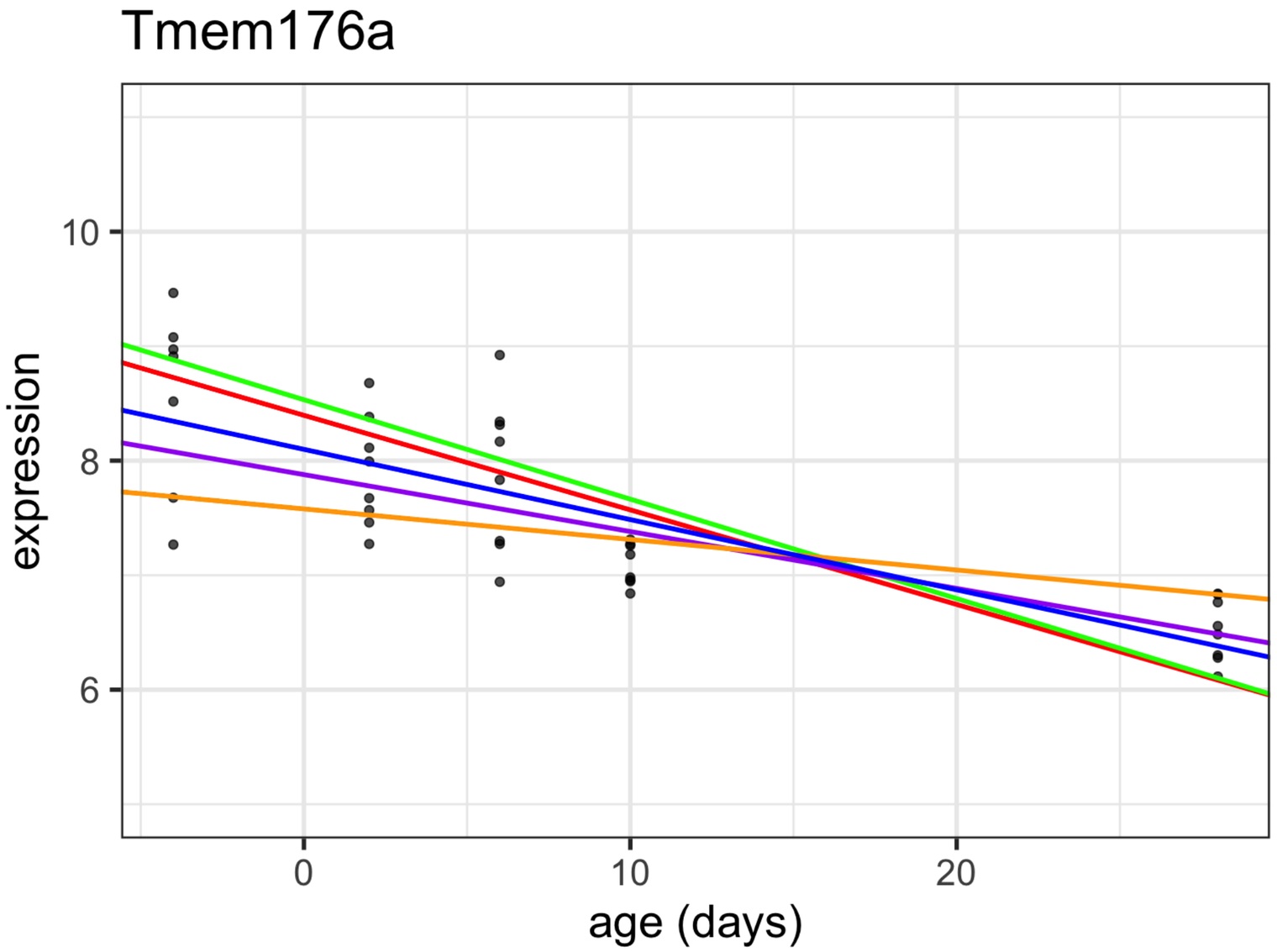
How do we estimate the intercept and slope?

Why is this the **optimal** line?

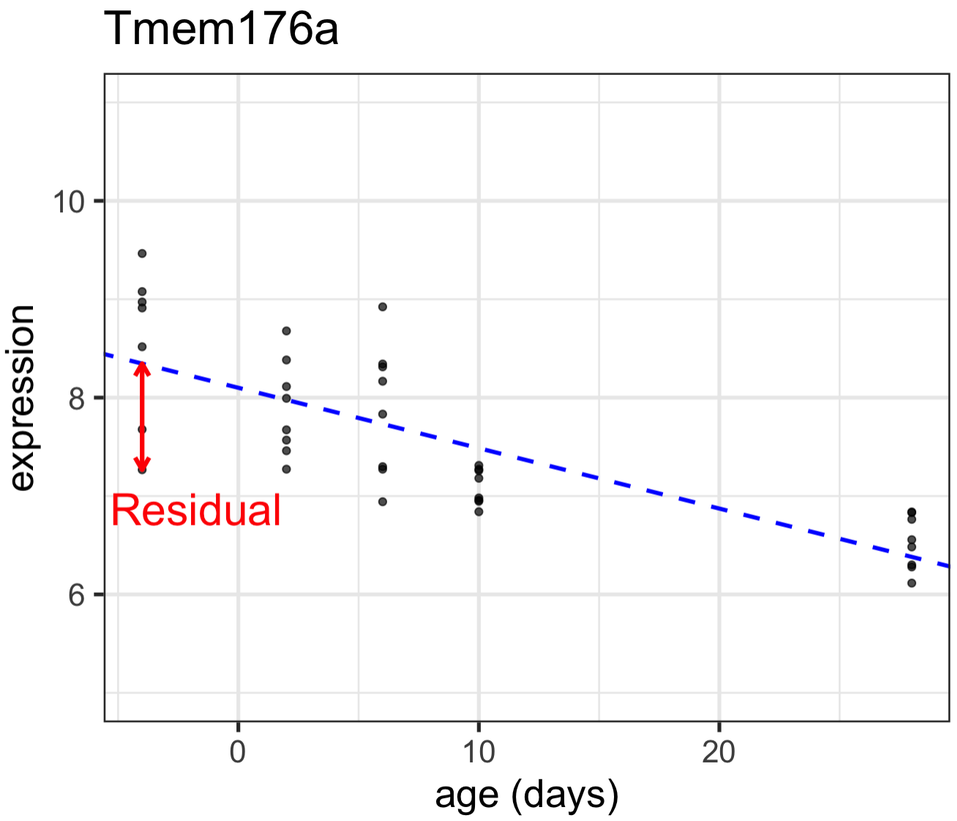
```
1 tidy(Tmem_fit)
```

```
# A tibble: 2 × 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   8.10     0.114       71.1  3.58e-41
2 age          -0.0614   0.00821     -7.47 6.74e- 9
```

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 Estimator for Simple Linear Regression (1 covariate)

- Mathematically: ε_i represents the error: $e_i = y_i - \alpha_0 - \alpha_1 x_i, i = 1, \ldots, n$
- 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*
  - the sum of squared errors is also referred to as **Residual Sum of Squares (RSS)**
- 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)

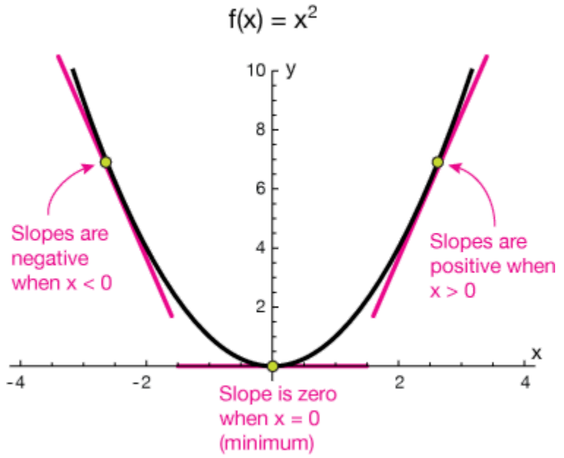
$$S(\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_p) = \sum_{i=1}^{n} (y_i - \alpha_0 - \alpha_1 x_{1i} - \alpha_2 x_{2i} - \ldots - \alpha_p x_{pi})^2$$

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

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

- Take partial derivatives with respect to each coeff, set to zero, solve system of equations:

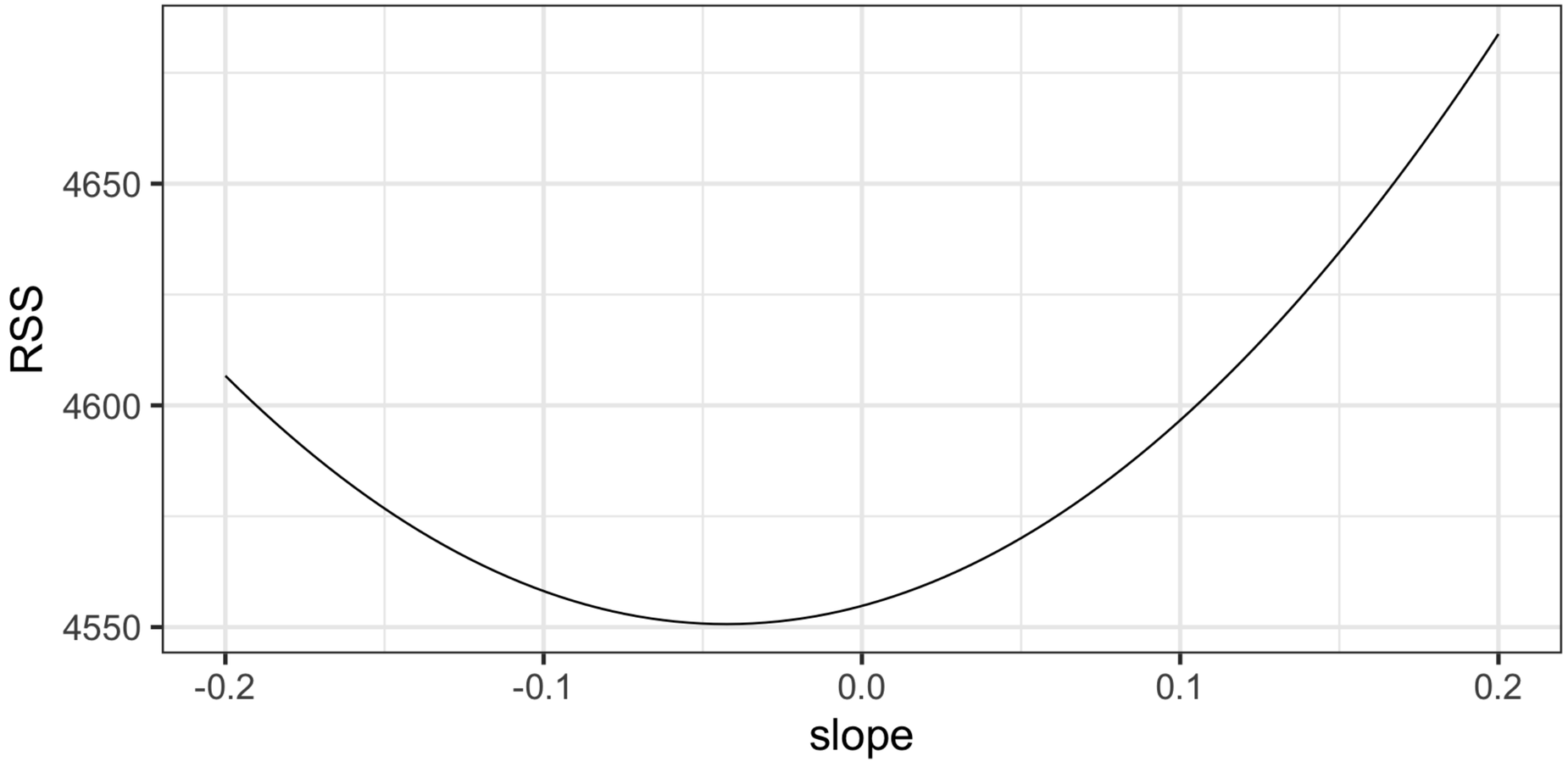
$$\frac{\partial S}{\partial \alpha_0} = \begin{bmatrix} \frac{\partial S}{\partial \alpha_0} \\ \frac{\partial S}{\partial \alpha_1} \\ \vdots \\ \frac{\partial S}{\partial \alpha_p} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$



Sums of squares for Tmem176a

Fixing the intercept at 8.1000793, let's plot the RSS values for a range of possible slope values.

▶ Code



OLS interactive demo

Launch demo

Explore

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 - what happens to the total area of the squares in the second plot when you modify the slope and intercept from the default values?
    - Note that you can reset to default values by refreshing the page

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"

Assumptions of OLS Regression

1. $\boldsymbol{\varepsilon}$ have mean zero
2. $\boldsymbol{\varepsilon}$ are iid (implies constant variance)
3. *(only required for hypothesis testing in small sample settings)* $\boldsymbol{\varepsilon}$ are Normally distributed

Connection to other estimators

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?

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 plots

Linear regression

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

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

- If 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

`lm` output | Plot

```
oneGene <- toLongerMeta(eset) %>%
  filter(gene %in% c("1427275_at")) %>%
  mutate(gene = "Smc4")
lm(expression ~ age + I(age^2), data = oneGene) %>% tidy()
```

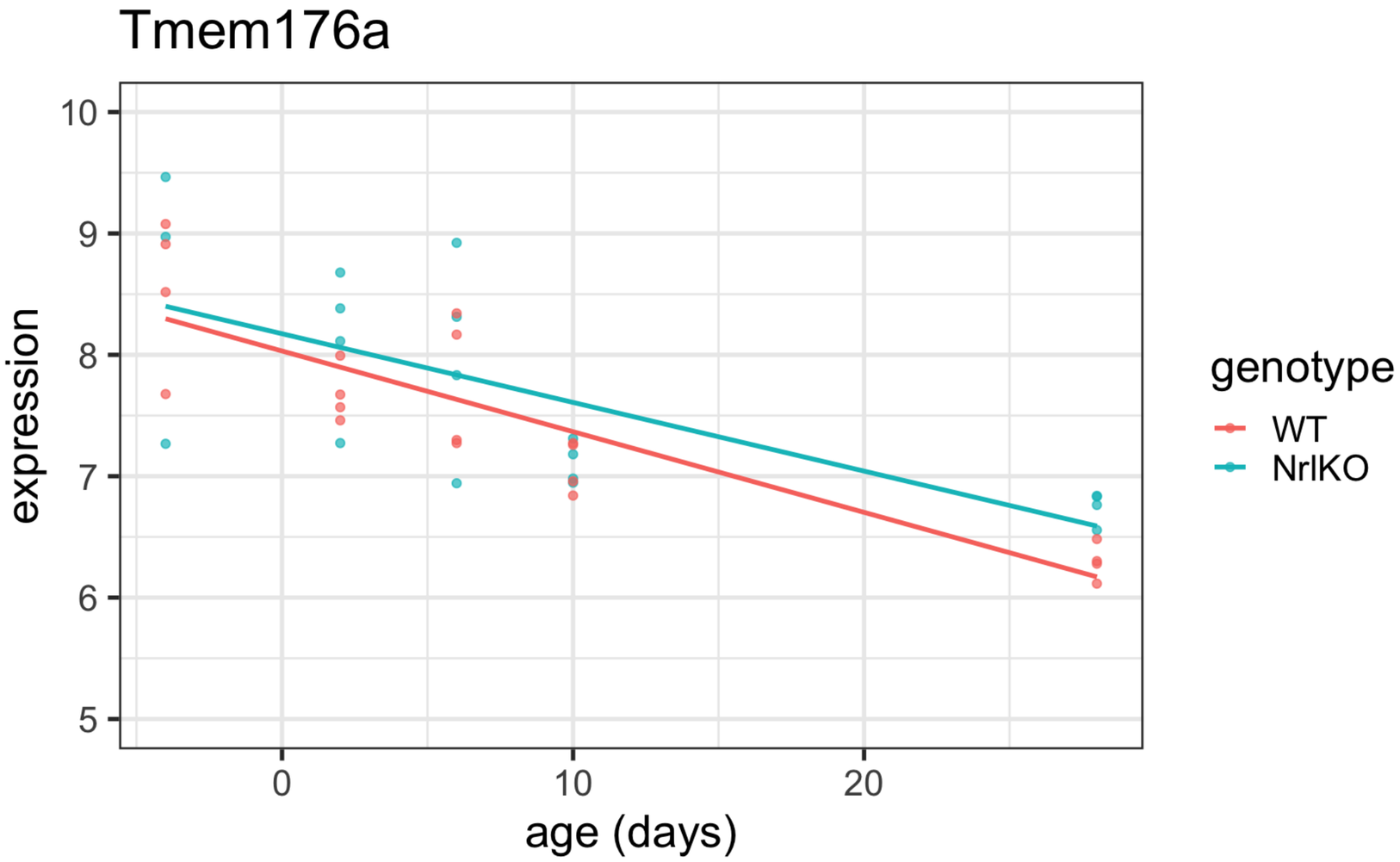
```
# A tibble: 3 × 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)  8.48       0.161       52.7  1.11e-35
2 age         -0.147      0.0326      -4.52 6.52e- 5
3 I(age^2)     0.00501    0.00116      4.30 1.23e- 4
```

Important

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

Putting it all together (continuous + categorical variables)

▶ Code



Interaction between continuous and categorical variables

```
1 lm(expression ~ age*genotype, data = filter(twoGenes, gene=="Tmem176a")) %>%
2   tidy()
```

```
# A tibble: 4 × 5
  term               estimate std.error statistic  p.value
  <chr>                 <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)          8.03       0.157     51.3   1.57e-34
2 age                 -0.0665     0.0114    -5.82  1.33e- 6
3 genotypeNrlKO        0.142      0.228      0.623 5.37e- 1
4 age:genotypeNrlKO    0.00987    0.0164     0.600 5.52e- 1
```

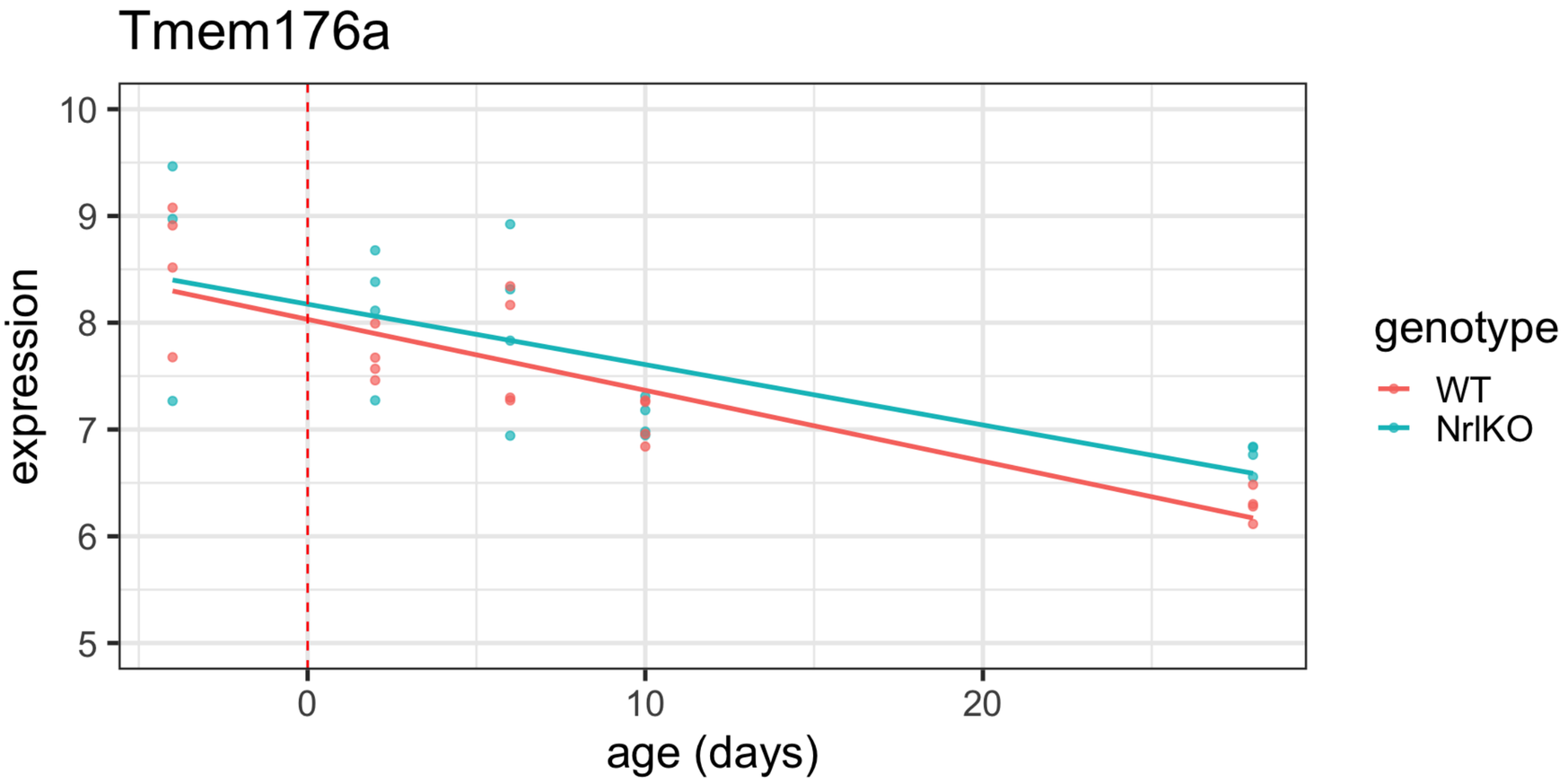
`(Intercept)`: Intercept of WT line

`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



Note

Intercept terms refer to the estimates when the continuous covariate is equal to zero. 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 (e.g. everything involving genotype) with an ***F*-test**:

```
1 Klf9dat <- filter(twoGenes, gene=="Klf9")
2 anova(lm(expression ~ age*genotype, data = Klf9dat),
3       lm(expression ~ age, data = Klf9dat))
```

```
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
```

Conclusion

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.

Collinearity & confounding

- If there are problems in the experimental design, we may not be able to estimate effects of interest

Danger

The technical definition of **collinearity** is that a column of the design matrix can be obtained (or *accurately approximated*) as a linear combination of other columns.

You can think of this as one column (or variable) not containing *unique information*

- This phenomenon can occur if our design is confounded
- As an example, let's pretend we know that all the E16 and P2 mice are female and the rest are male

Example: E16/P2 mice are female and the rest are male

Dataset | Design matrix | Linear combination | `lm` output

```
Klf9dat_conf <- Klf9dat %>%
  mutate(sex = ifelse(dev_stage %in% c("E16", "P2"), "female", "male"))
table(Klf9dat_conf$sex, Klf9dat_conf$dev_stage)
```

```
        E16 P2 P6 P10 P28
female    7  8  0   0   0
male      0  0  8   8   8
```

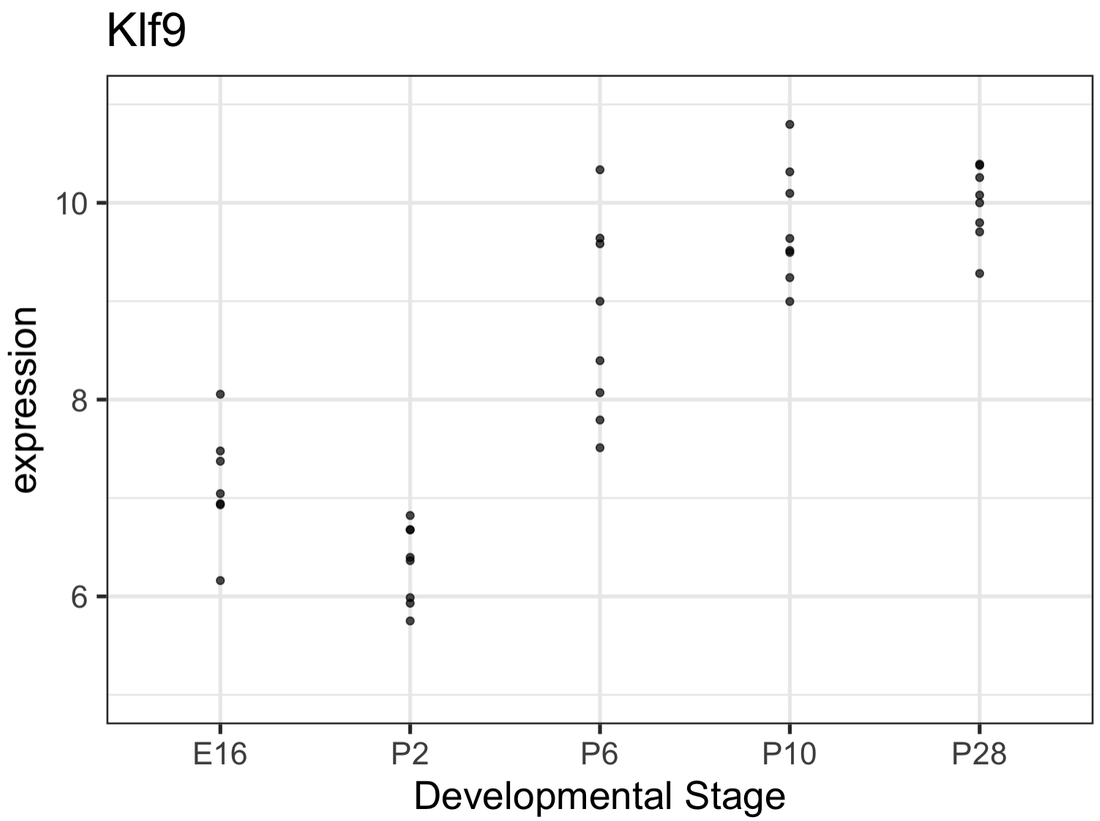
Hidden confounding

Danger

Hidden **confounders** are variables with associations to both the outcome and predictor variable(s), but are not measured (or not included in the model).

They can cause spurious correlations, and illustrate why *correlation does not imply causation*.

Hidden | Revealed | Simpson's paradox



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 predictor? 👍
  - categorical predictor? 👍
  - quantitative predictor? 👍
  - various combinations of the above? 👍
- Don't be afraid to build models with more than 1 covariate
- later, we'll talk about extensions to discrete outcomes (e.g. dichotomous or counts) via *generalized linear models*

What about the other 45 thousand probesets??

```
1 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

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

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: $\hat{Var}(\hat{\boldsymbol{\alpha}}_g) = s_g^2(\mathbf{X}^T\mathbf{X})^{-1} = s_g^2\mathbf{V}$
  - $\mathbf{V}$ 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 $s_g\sqrt{v_{jj}}$

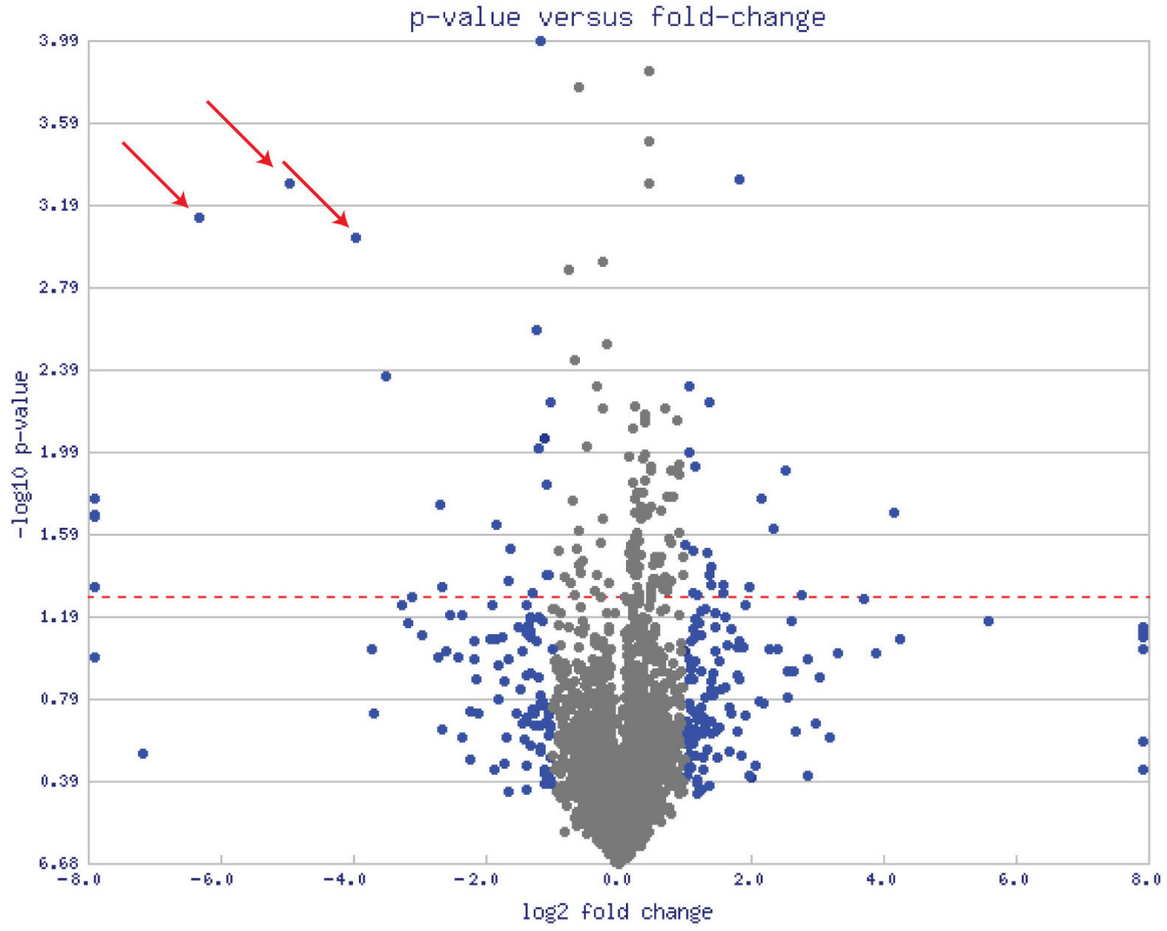
What's the big deal?

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



Important

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?

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

- Small variance estimate s_g leads to large *t* statistic → 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

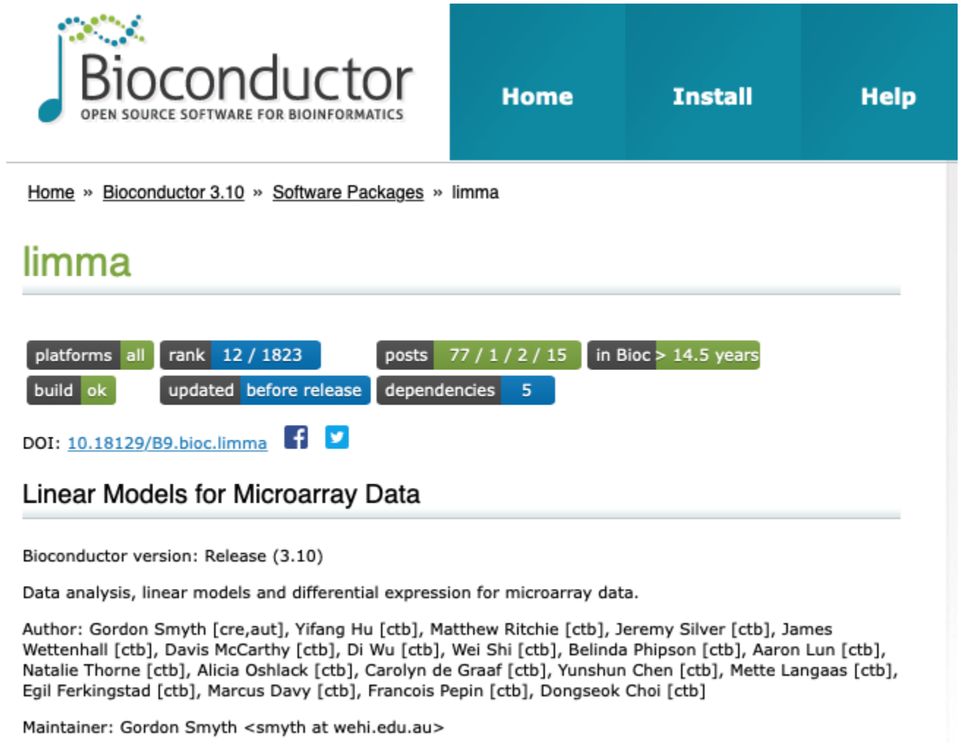
Empirical Bayesian techniques: `limma`

› 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



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`
- 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:**

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

- **Fitted/predicted values:**

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

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

```
allGenes <- toLongerMeta(eset)
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 P28          28 NrlKO
 2 1415670_at GSM92611         7.32 P28          28 NrlKO
 3 1415670_at GSM92612         7.42 P28          28 NrlKO
 4 1415670_at GSM92613         7.35 P28          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
  4.686   0.051   4.788
```

```
lmfits %>% head()
```

```
# A tibble: 6 × 4
# Groups:   gene [6]
  gene         `(Intercept)`       age genotypeNrlKO
  <chr>                <dbl>     <dbl>         <dbl>
1 1415670_at            7.22  0.000623     -0.000286
2 1415671_at            9.32 -0.00184       0.145
3 1415672_at            9.76 -0.00393      -0.0422
4 1415673_at            8.40  0.00398      -0.0436
5 1415674_a_at          8.52 -0.00598       0.0192
6 1415675_at            9.67 -0.00642       0.133
```

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

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

```
   user  system elapsed
  0.101   0.027   0.128
```

```
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.

Tip

Avoiding repetitive calculations involving $\mathbf{X}$ leads to ~35X faster computation in this example

How can we better estimate the SE?

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

Important

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

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

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

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 for how the gene-wise α_{gj} and σ_g^2 vary **across genes**

> **Important**
>
> Under the limma framework, we are no longer considering genes in isolation. We will leverage information across genes to obtain improved estimates.

- this is done by assuming a **prior distribution** for those quantities
- Prior distribution for **gene-specific variances** σ_g^2: inverse χ^2 with mean s_0^2, d_0 df:

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

- 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:

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

- A weighted mean of the *prior* and the *observed* 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 $\tilde{t}_{gj}$ 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[1]:	$s_g^2 = \frac{1}{n-p}\hat{\boldsymbol{\varepsilon}}^T\hat{\boldsymbol{\varepsilon}}$	$\tilde{s}_g^2 = \frac{d_0 s_0^2 + d s_g^2}{d_0 + d}$
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:	$t_{gj} \sim t_d$	$\tilde{t}_{gj} \sim t_{d_0+d}$

1. 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

- **degrees of freedom** for null distribution *increase* relative to default (d 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