STAT 540
Class meeting 09
Monday, February 2, 2015

Dr. Gabriela Cohen Freue
Department of Statistics

Based on previous preparation by Dr. Jennifer (Jenny) Bryan

Quantitative covariates ... for real this time
Industrial Revolution for fitting linear models ... how to massively scale up production

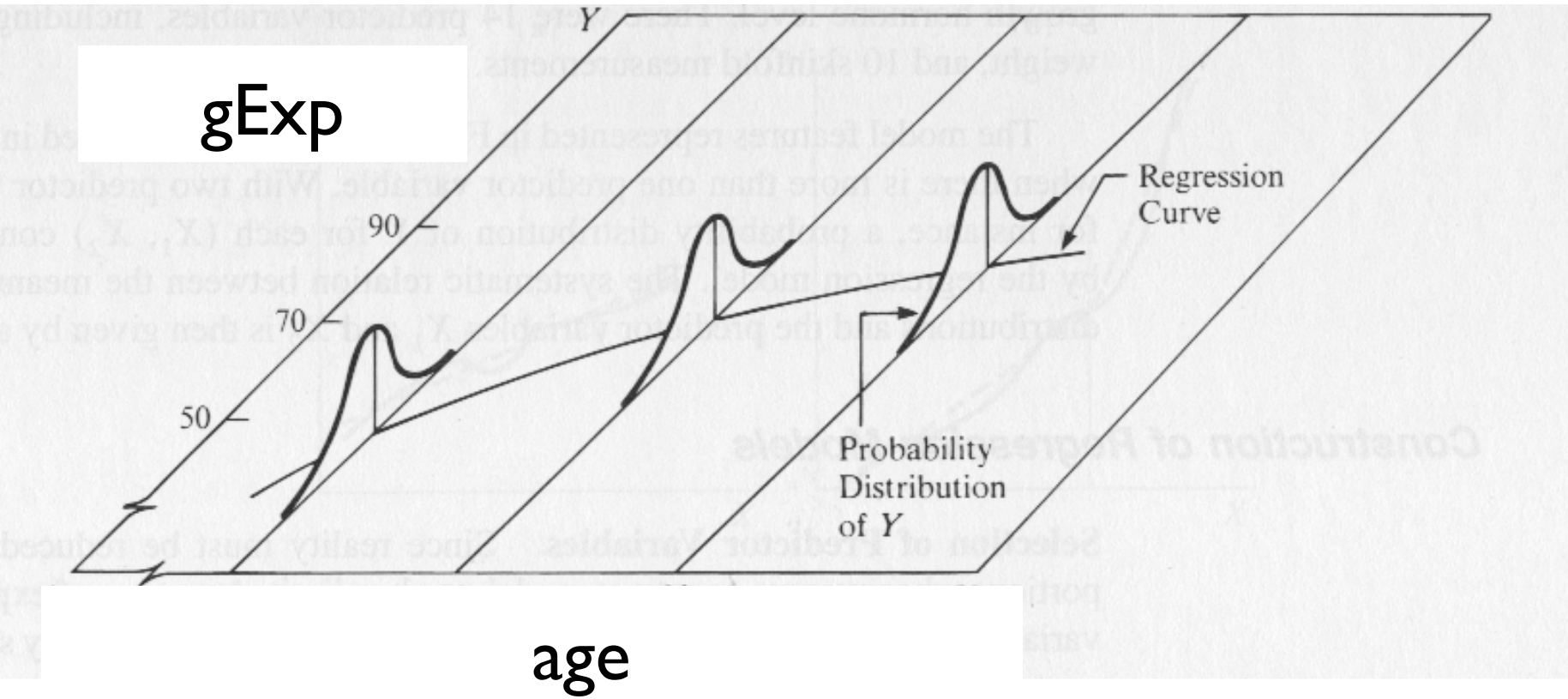
Factors that take on many levels can be unwieldy to deal with ... do you care about the effect of each level and all of it's potential interactions? or do you only care about the factor in a big picture way?

If it represents something like time or dose or temperature ... factor treatment makes it awkward to pull out natural classes of "hits", e.g. things that go up

consider making a quantitative covariate, age in days, and use that to explain changes in gene expression

```
> ## recode() is from add-on package 'car'

> prDes$age <-
+     recode(prDes$devStage,
+            "'E16'=-2; 'P2'=2; 'P6'=6; 'P10'=10; '4_weeks'=28",
+            as.factor.result = FALSE)

> peek(prDes)
          sample devStage gType age
Sample_22     22      E16    wt  -2
Sample_16     16      E16 NrlKO  -2
Sample_5       5       P2 NrlKO   2
Sample_31     31       P6    wt   6
Sample_15     15      P10 NrlKO  10
Sample_36     36  4_weeks    wt  28
Sample_2       2  4_weeks NrlKO  28

> str(prDes)
'data.frame':  39 obs. of  4 variables:
 $ sample  : num  20 21 22 23 16 17 6 24 25 26 ...
 $ devStage: Factor w/ 5 levels "E16","P2","P6",..: 1 1 1 1 1 1 1 2 2 2 ...
 $ gType   : Factor w/ 2 levels "wt","NrlKO": 1 1 1 1 2 2 2 1 1 1 ...
 $ age     : num  -2 -2 -2 -2 -2 -2 -2 2 2 2 ...
```

meet our new quantitative covariate or predictor ...
age, which is a new version of the factor devStage

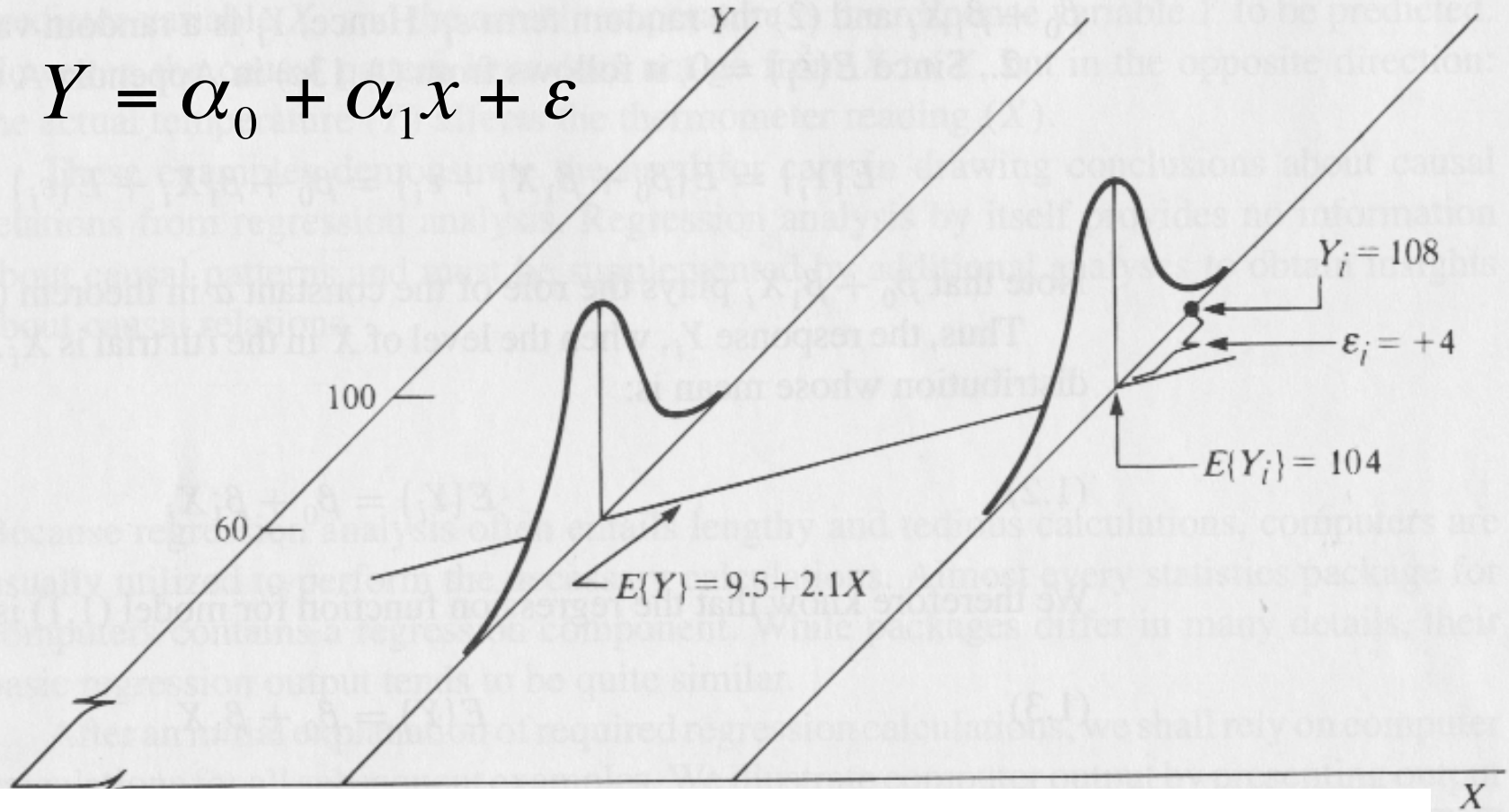


$$Y_{X=x} = f(x;\alpha) + \varepsilon_x, E(\varepsilon_x) = 0$$

FIGURE 1.6 Illustration of Sin[illegible] Model (1.1).

gExp

$$Y = \alpha_0 + \alpha_1 x + \varepsilon$$



age

Regression function is *linear* ... <u>*linear model*</u>.

Plain vanilla linear model, matrix formulation

$$Y = X\alpha + \varepsilon$$

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

Here's what a design matrix would look like with 1 quantitative covariate.

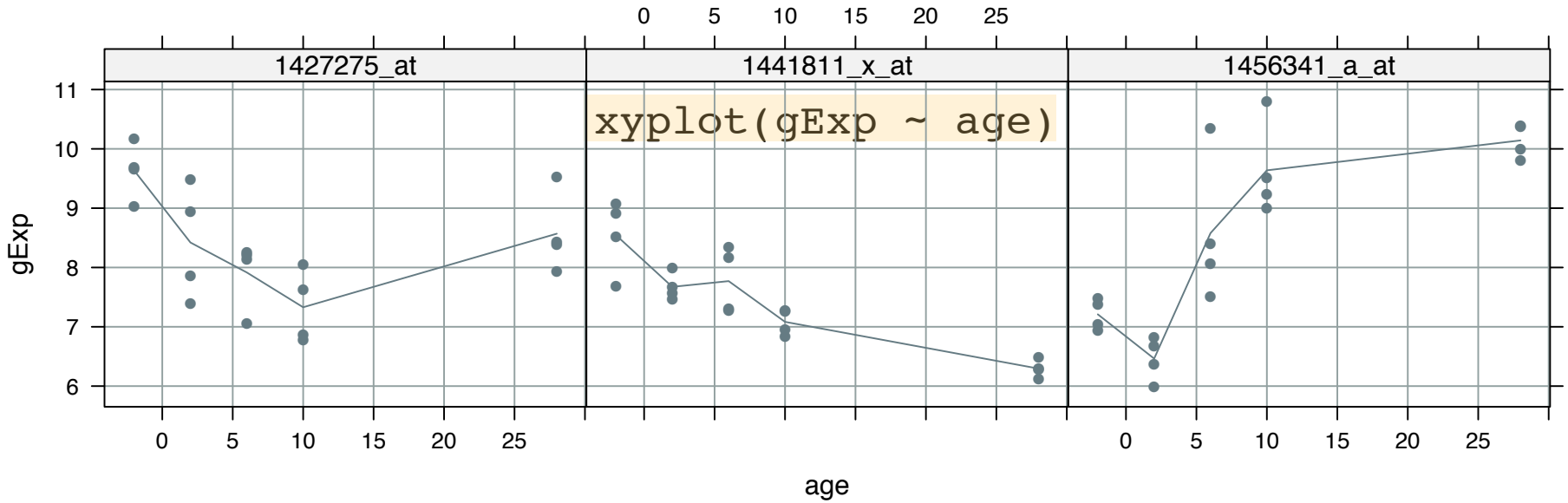
$$Y = X\alpha + \varepsilon$$

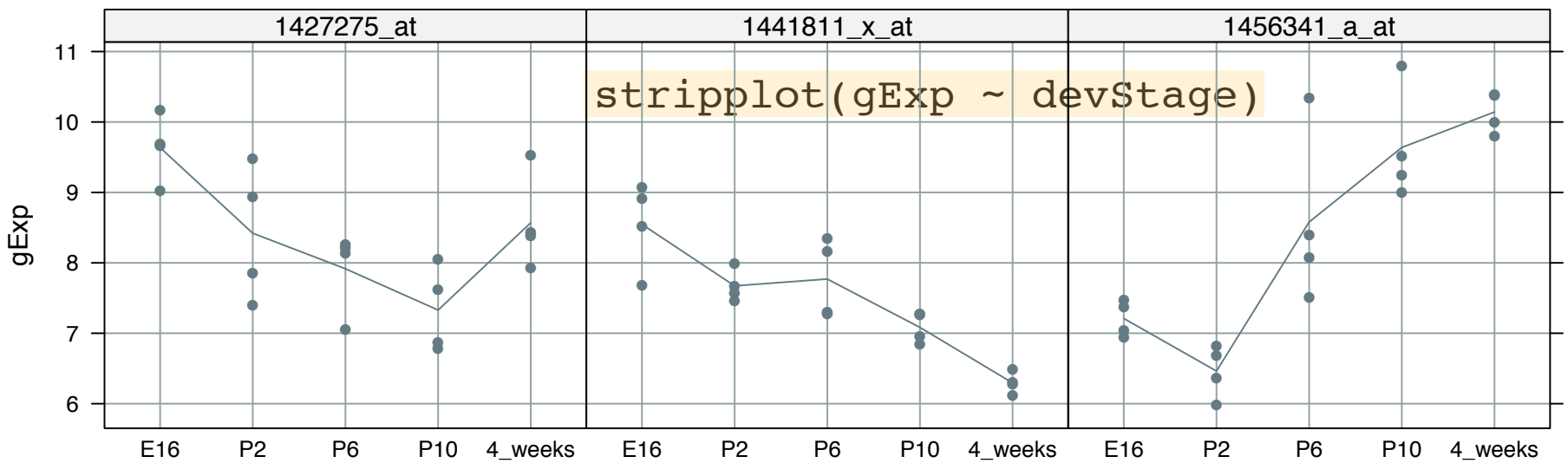
$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} = \begin{bmatrix} \alpha_0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

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

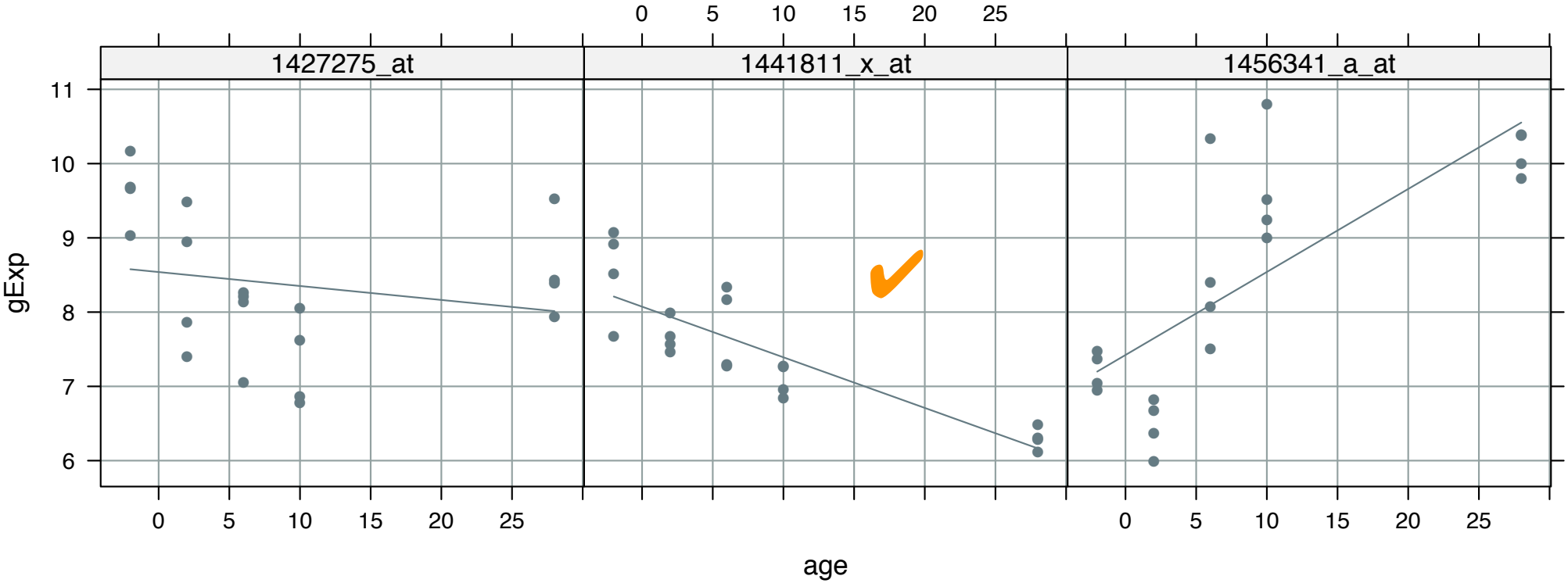
Remember / convince yourself that the matrix algebra does indeed reproduce simple linear regression.

for starters, let's just work with wild type data for 3 example probesets

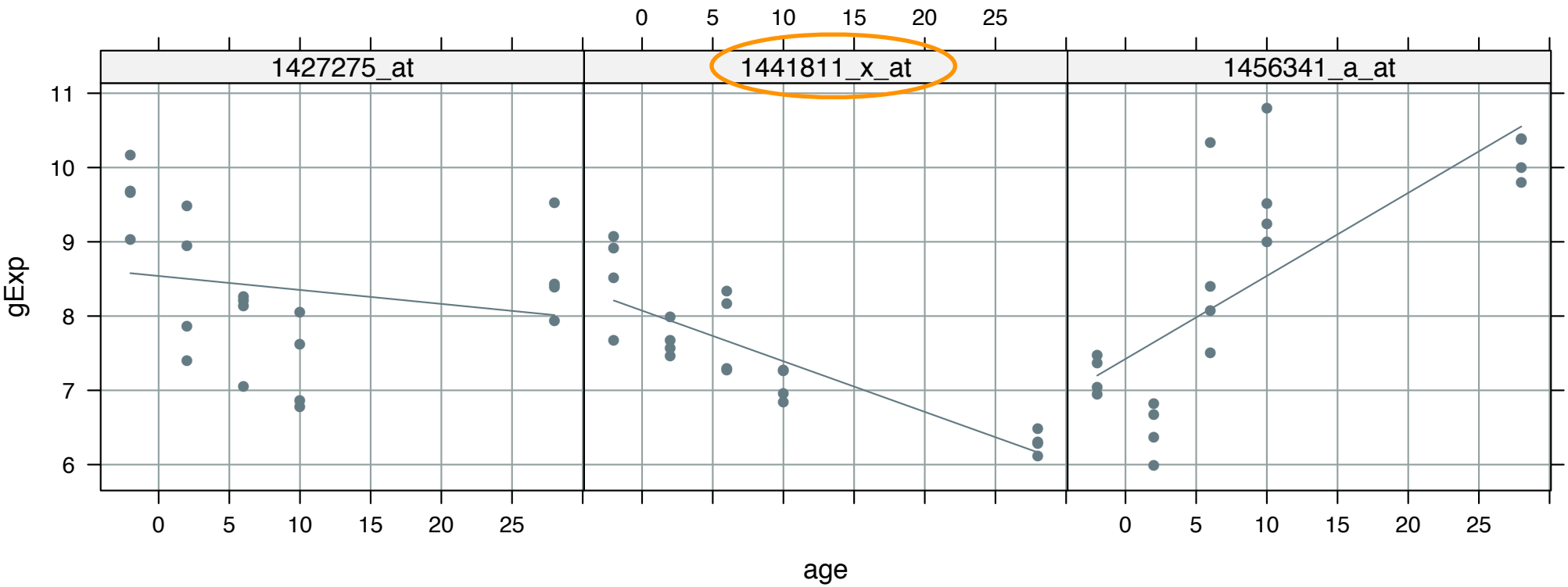




Kind of a different look to the data, no?



linear looks reasonable for 1, but not the other two




```
> summary(linFits[["1441811_x_at"]])

Call:
lm(formula = gExp ~ age, data = z)

Residuals:
     Min       1Q   Median       3Q      Max
-0.55059 -0.37459 -0.08398  0.31011  0.86827

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.073374   0.133118  60.648  < 2e-16 ***
age         -0.068179   0.009771  -6.978 1.62e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4545 on 18 degrees of freedom
Multiple R-squared: 0.7301,    Adjusted R-squared: 0.7151
F-statistic: 48.69 on 1 and 18 DF,  p-value: 1.622e-06
```

- The nature of the regression function $f(x; \alpha)$ is one of the defining characteristics of a regression model
  - f linear in $\alpha \Rightarrow$ linear model
  - f not linear in $\alpha \Rightarrow$ nonlinear model

nonlinear parametric regression

$$Y = \frac{1}{1 + e^{(\phi - x)/\xi}} + \varepsilon$$

simple linear regression (a linear model)

$$Y = \alpha_0 + \alpha_1 x + \varepsilon$$

What we just did.

polynomial regression (also a linear model)

$$Y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \varepsilon$$

What we're about to do.

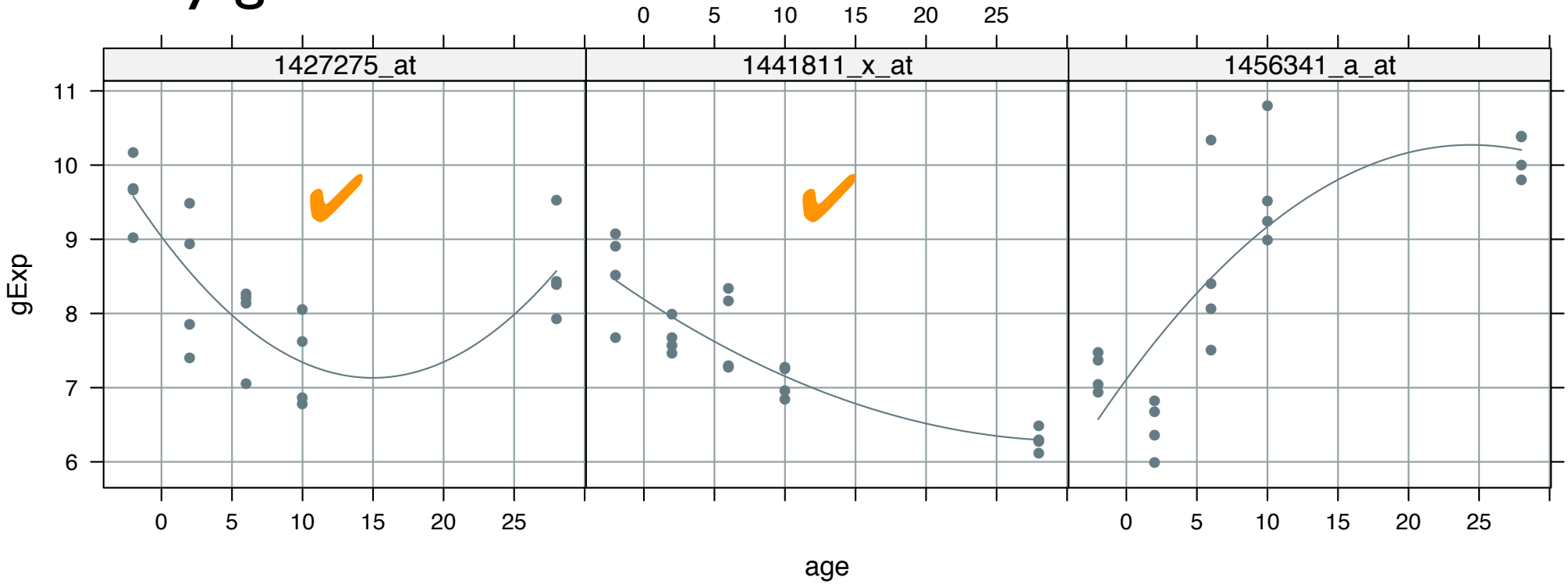
- The nature of the regression function $f(x; \alpha)$ is one of the defining characteristics of a regression model
  - f linear in $\alpha \Rightarrow$ linear model
  - f not linear in $\alpha \Rightarrow$ nonlinear model

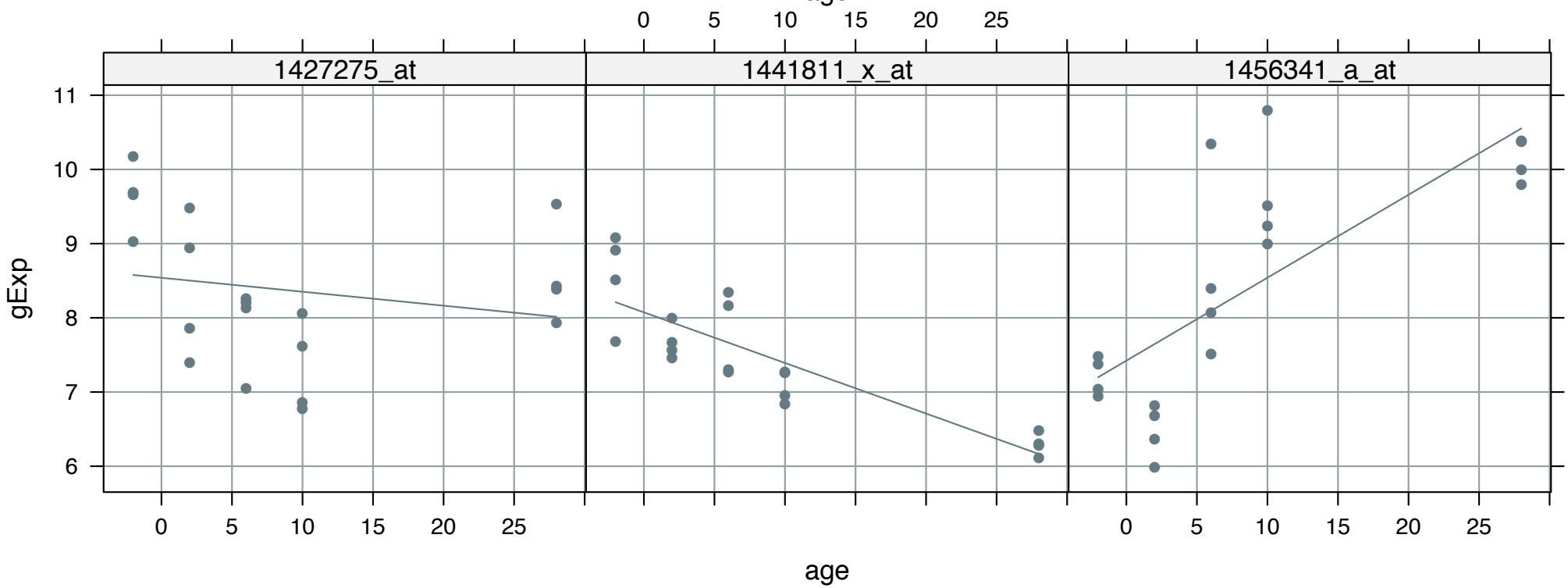
polynomial regression (also a linear model)

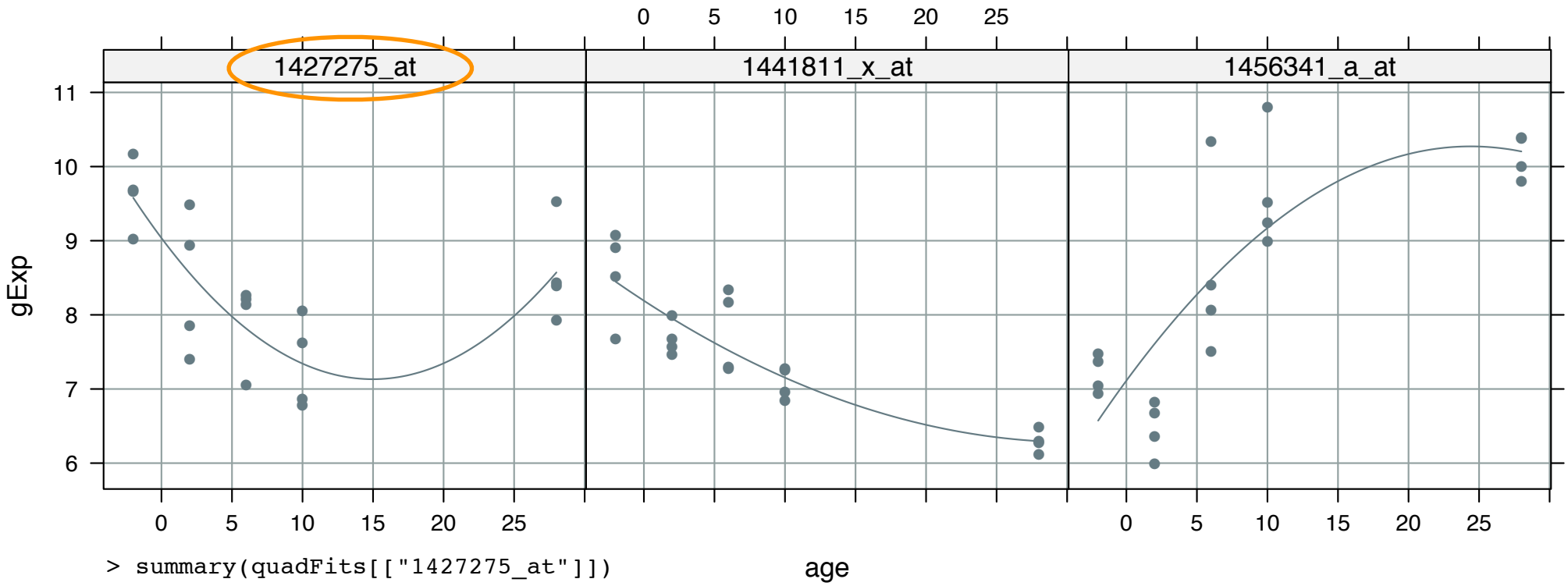
$$Y = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \varepsilon$$

NOTE: **This is a linear model**, because it is linear in the alphas. It is easy but wrong to focus on the x's and mistake this for a nonlinear model.

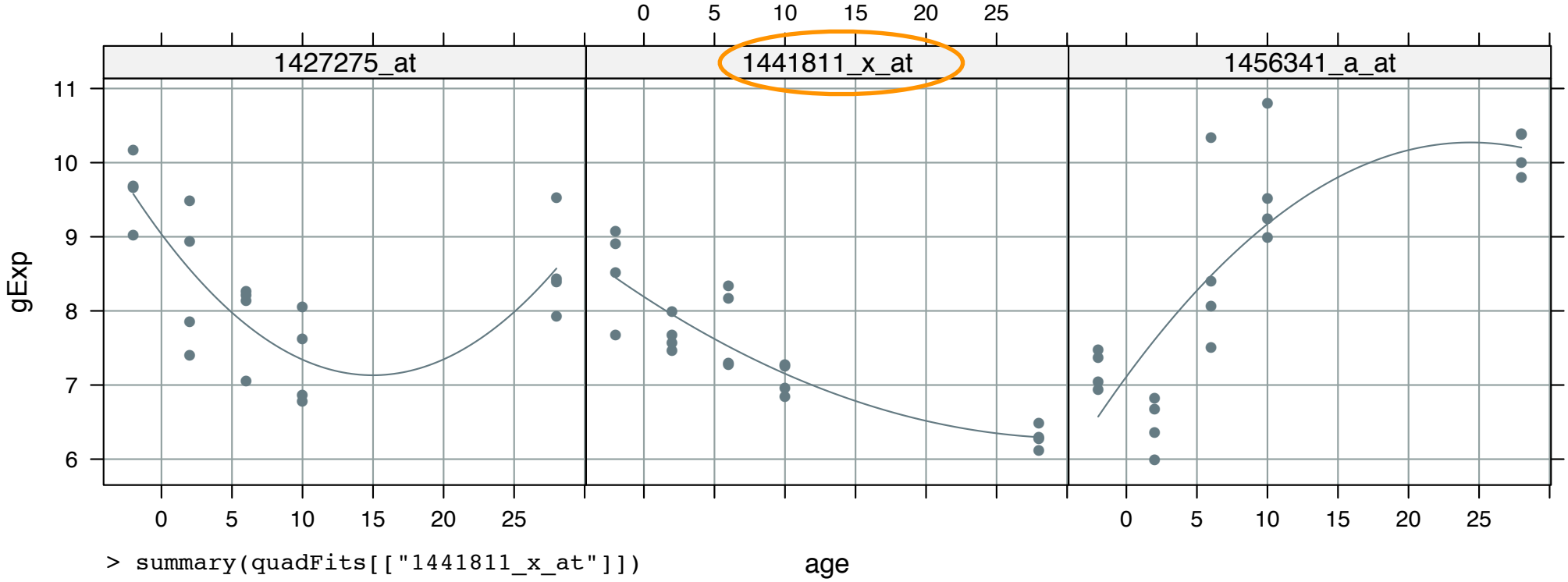
fairly good fit for 2 of 3 now!



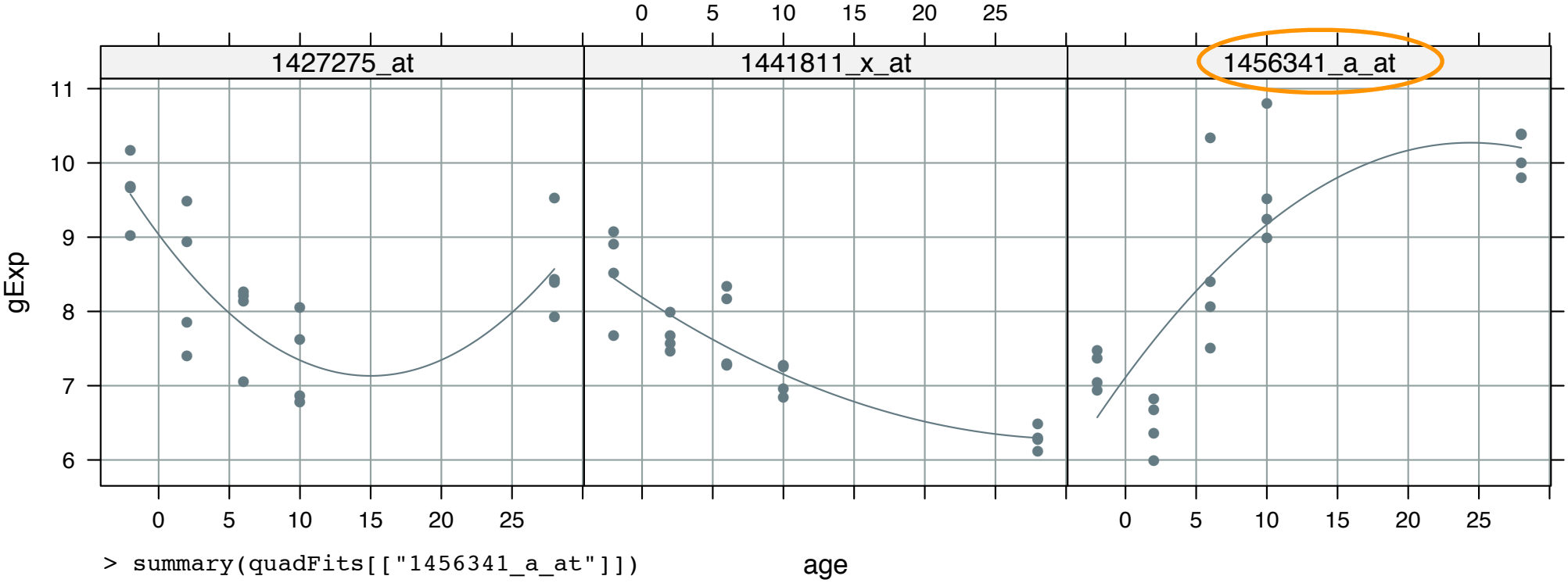





```
> summary(quadFits[["1427275_at"]])

Call:
lm(formula = gExp ~ age + I(age^2), data = z)

Residuals:
     Min       1Q   Median       3Q      Max
-1.16275 -0.55506  0.09503  0.40804  0.95803

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.036401   0.212313  42.562  < 2e-16 ***
age         -0.254305   0.048125  -5.284 6.07e-05 ***
I(age^2)     0.008490   0.001661   5.110 8.71e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6444 on 17 degrees of freedom
Multiple R-squared: 0.6218,    Adjusted R-squared: 0.5773
F-statistic: 13.98 on 2 and 17 DF,  p-value: 0.0002572
```




```
> summary(quadFits[["1441811_x_at"]])

Call:
lm(formula = gExp ~ age + I(age^2), data = z)

Residuals:
     Min       1Q   Median       3Q      Max
-0.76946 -0.25477 -0.00589  0.13662  0.82202

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.190766   0.140969  58.103  < 2e-16 ***
age         -0.123836   0.031953  -3.876  0.00121 **
I(age^2)     0.002006   0.001103   1.819  0.08660 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4279 on 17 degrees of freedom
Multiple R-squared: 0.774,     Adjusted R-squared: 0.7475
F-statistic: 29.12 on 2 and 17 DF,  p-value: 3.23e-06
```




```
> summary(quadFits[["1456341_a_at"]])

Call:
lm(formula = gExp ~ age + I(age^2), data = z)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.6211 -0.5010 -0.0050  0.3955  1.8651 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  7.112481   0.310922  22.875  3.3e-14 ***
age          0.258892   0.070477   3.673  0.00188 ** 
I(age^2)    -0.005303   0.002433  -2.180  0.04363 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9437 on 17 degrees of freedom
Multiple R-squared: 0.6737,	Adjusted R-squared: 0.6353 
F-statistic: 17.55 on 2 and 17 DF,  p-value: 7.337e-05
```

F tests in regression

Remember this?

small model is nested within big, e.g., it's a special case where some parameters are equal to zero

model	example	# params = DF	RSS
small	lm(y ~ gType + devStage)	$p_{small} = 6$	RSS_{small}
big	lm(y ~ gType * devStage)	$p_{big} = 10$	RSS_{big}

$$y_{ijk} = \theta + \tau_j + \beta_k + (\tau\beta)_{jk} + \varepsilon_{ijk} \text{ “big”}$$

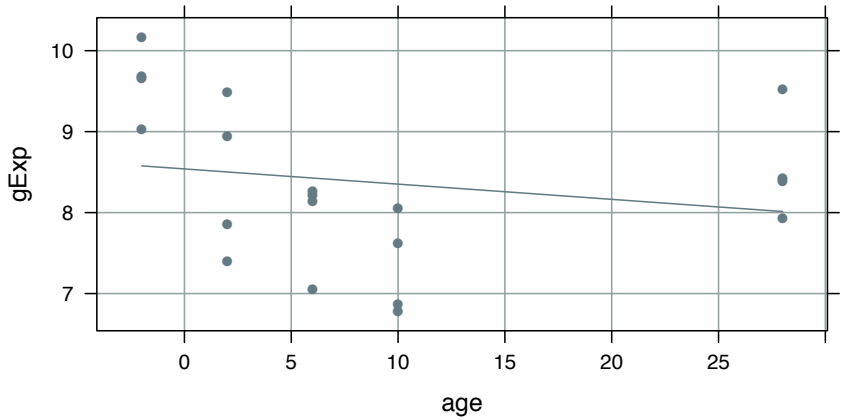
$$y_{ijk} = \theta + \tau_j + \beta_k + \cancel{(\tau\beta)}_{jk} + \varepsilon_{ijk} \text{ “small”}$$

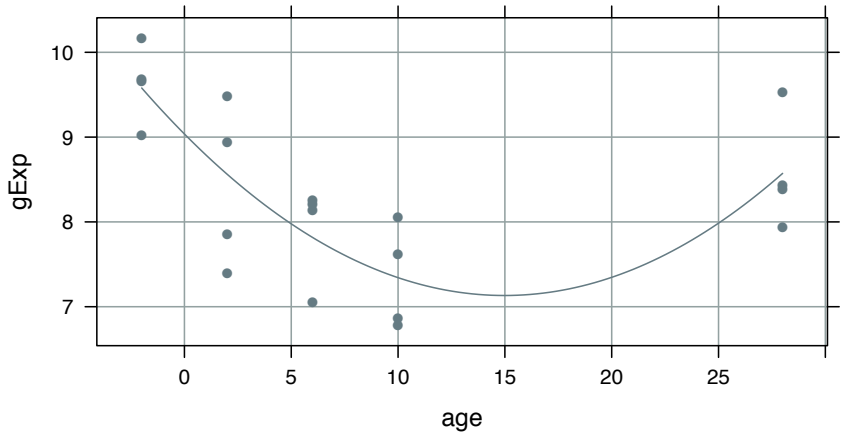
by definition:

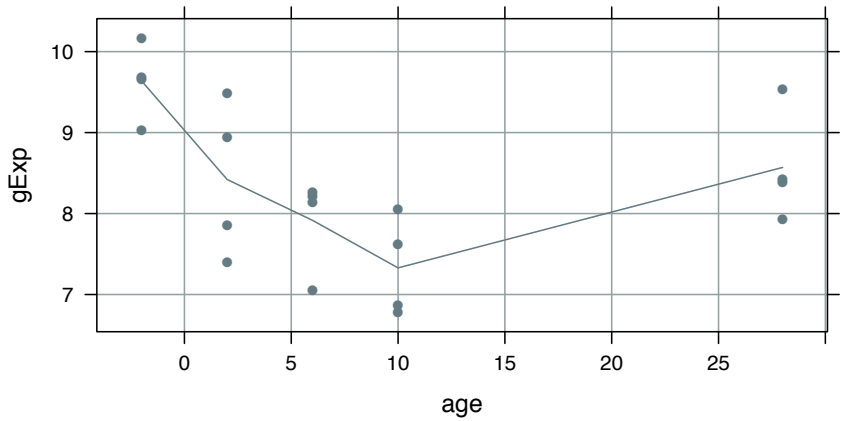
$p_{small} < p_{big}$

$RSS_{small} \geq RSS_{big}$

$$F = \frac{\left(\dfrac{RSS_{small} - RSS_{big}}{p_{big} - p_{small}}\right)}{\dfrac{RSS_{big}}{n - p_{big}}} \sim_{H_0} F_{(p_{big} - p_{small}, n - p_{big})}$$








```
> (jGene <- luckyGenes[1])
[1] "1427275_at"
```

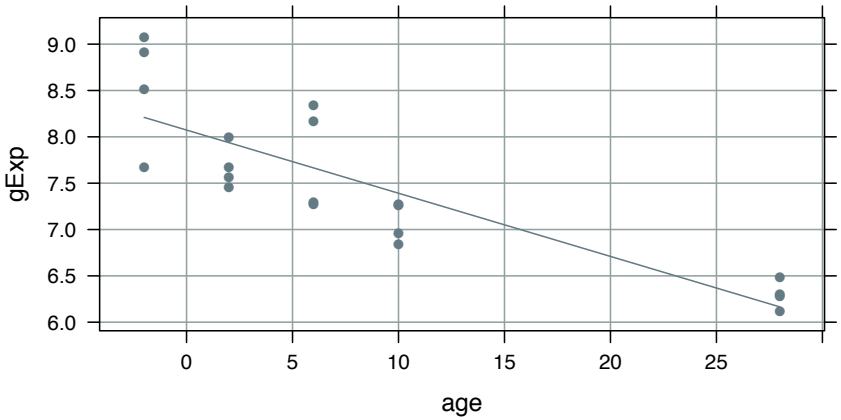
small big

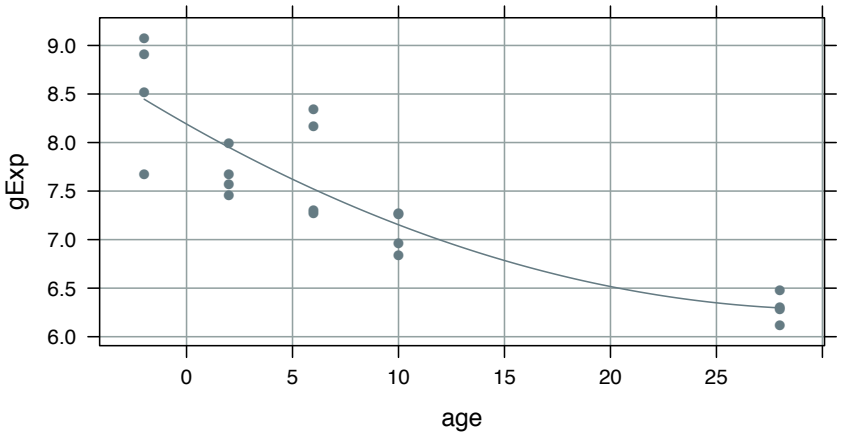
```
> anova(linFits[[jGene]], quadFits[[jGene]])

Analysis of Variance Table

Model 1: gExp ~ age
Model 2: gExp ~ age + I(age^2)
  Res.Df     RSS Df Sum of Sq      F   Pr(>F)
1     18 17.9021
2     17  7.0591  1    10.843 26.113 8.71e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> AIC(linFits[[jGene]], quadFits[[jGene]], factFits[[jGene]])
                  df      AIC
linFits[[jGene]]   3 60.54129
quadFits[[jGene]]  4 43.92930
factFits[[jGene]]  6 47.54810
```

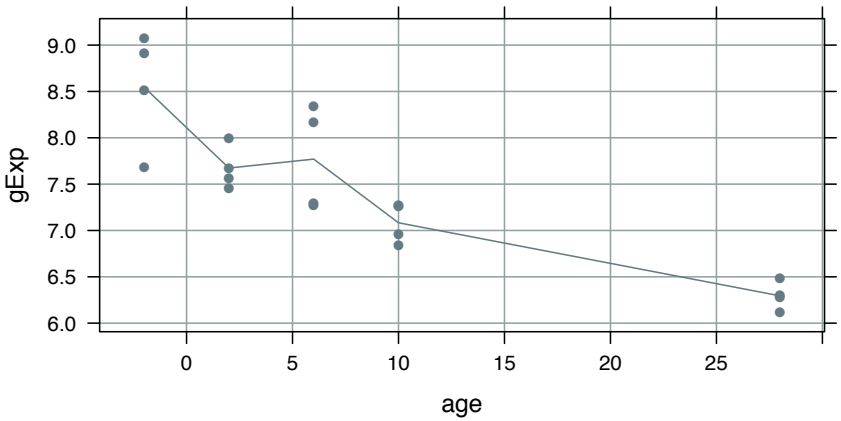
it's "worth it" to go from linear to quadratic here

but hard to justify going from quadratic to one-way ANOVA

possible links to read more about using AIC to compare non-nested models: stackexchange and Wikipedia








```
> (jGene <- luckyGenes[3])
[1] "1441811_x_at"
```

small big

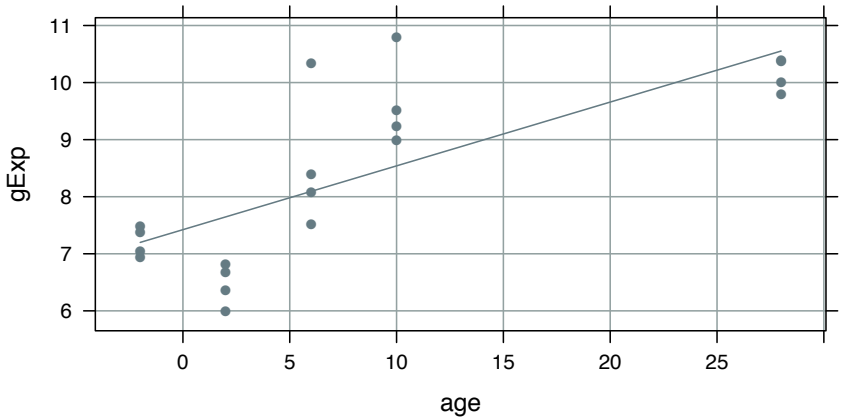
```
> anova(linFits[[jGene]], quadFits[[jGene]])

Analysis of Variance Table

Model 1: gExp ~ age
Model 2: gExp ~ age + I(age^2)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     18 3.7176
2     17 3.1120  1   0.60559 3.3081 0.0866 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> AIC(linFits[[jGene]], quadFits[[jGene]], factFits[[jGene]])
                  df      AIC
linFits[[jGene]]   3 29.10466
quadFits[[jGene]]  4 27.54851
factFits[[jGene]]  6 27.12587
```

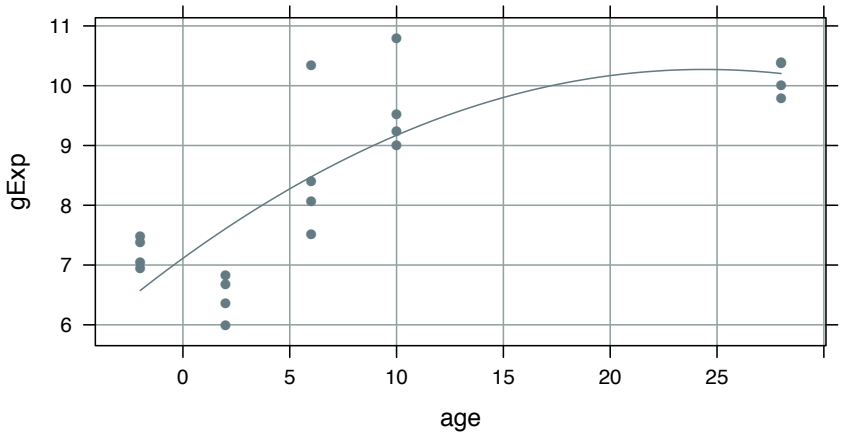
meh

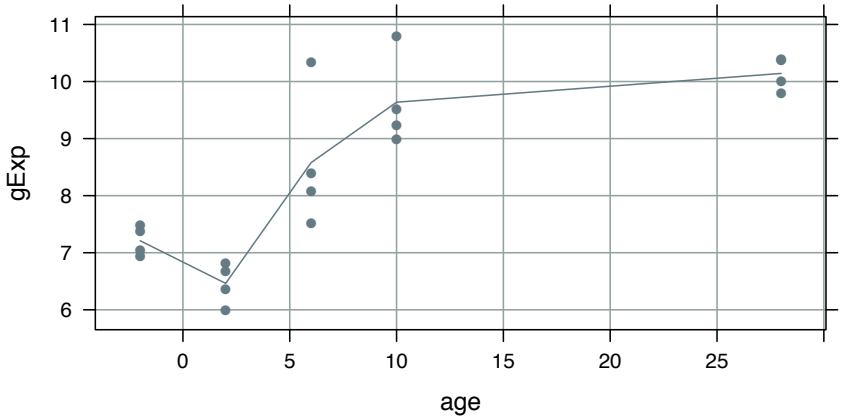
not clear it's "worth it" to go from linear to quadratic here

even less payoff to go from quadratic to one-way ANOVA

<u>Occam's Razor</u> and the <u>KISS principle</u> → stick w/ simple linear model








```
> (jGene <- luckyGenes[2])

[1] "1456341_a_at"
```

small big

```
> anova(linFits[[jGene]], quadFits[[jGene]])

Analysis of Variance Table

Model 1: gExp ~ age
Model 2: gExp ~ age + I(age^2)
  Res.Df    RSS Df Sum of Sq      F  Pr(>F)
1     18 19.370
2     17 15.139  1    4.2308 4.7509 0.04363 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> AIC(linFits[[jGene]], quadFits[[jGene]], factFits[[jGene]])
                  df      AIC
linFits[[jGene]]   3 62.11743
quadFits[[jGene]]  4 59.18864
factFits[[jGene]]  6 48.70210
```

it's probably "worth it" to go from linear to quadratic here (?)

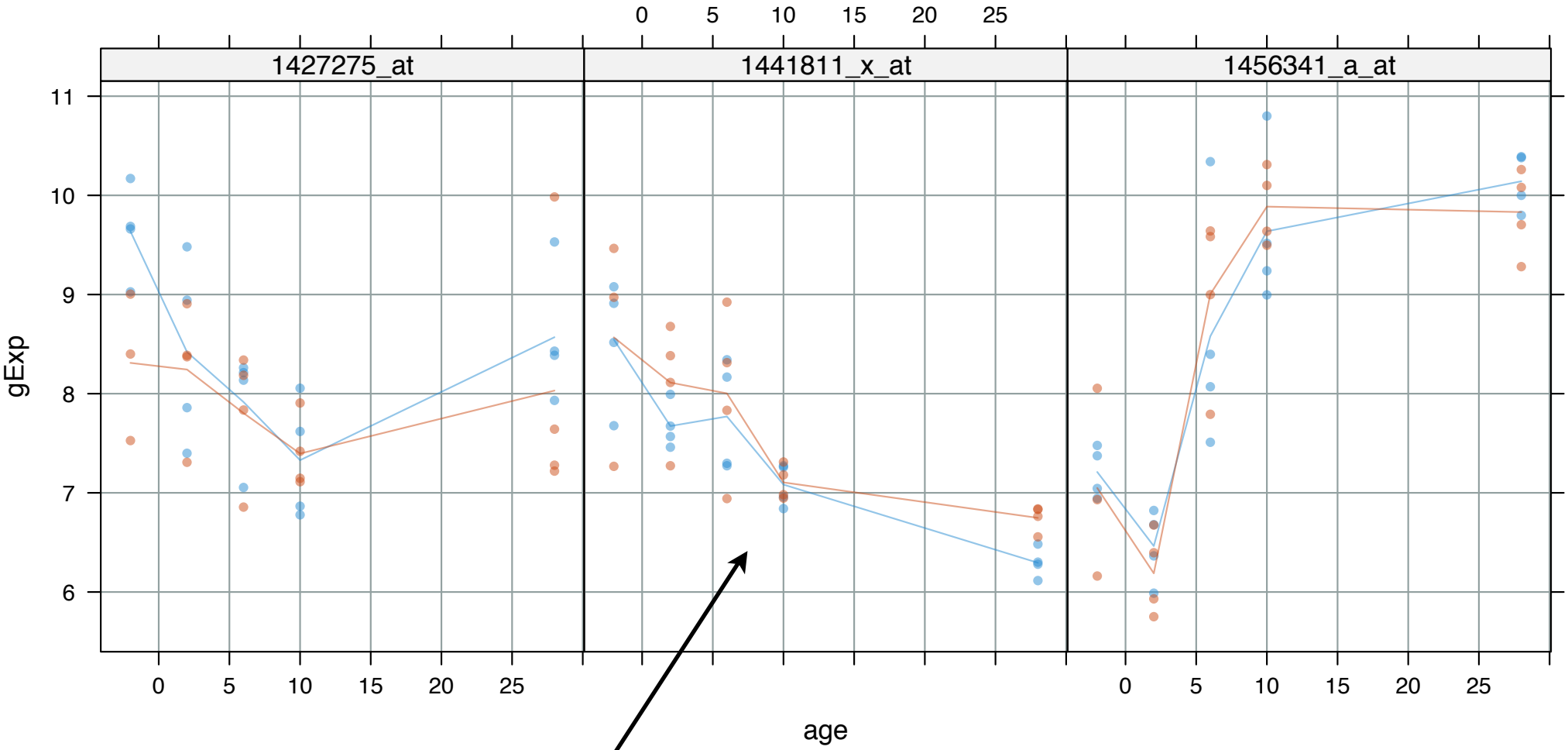
going from quadratic to one-way ANOVA seems justified

increase the complexity ...

1 quantitative covariate: age

AND 1 categorical covariate:

genotype = wt vs. Nrl knockout



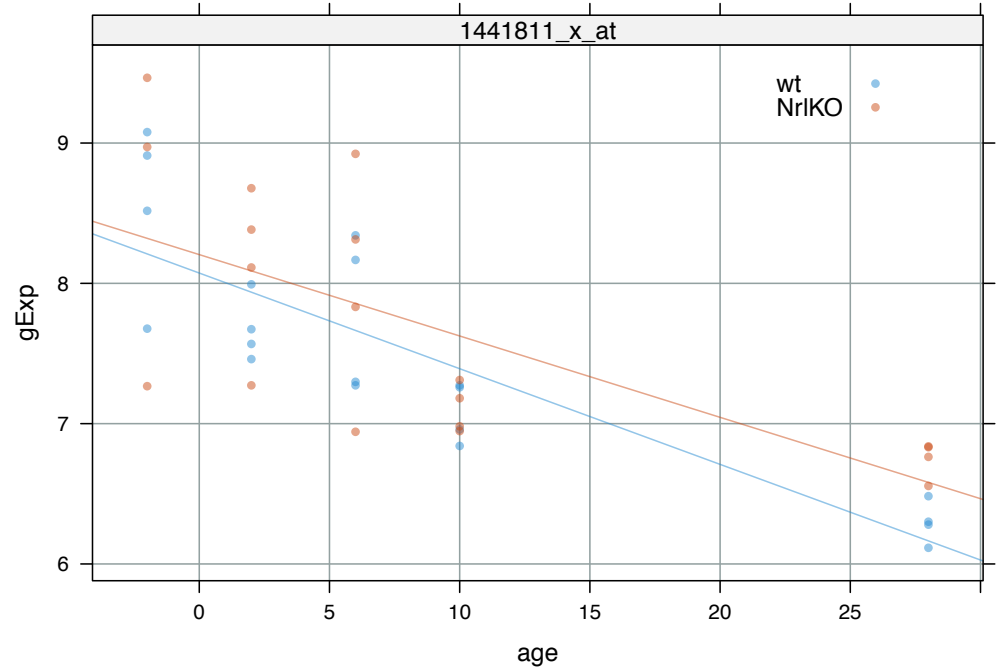
let's focus on this one for a model with just intercept and slope, possibly different for wt and NrlKO

$$y_{ij} = \alpha_{0,wt} + \tau_{0,j} + (\alpha_{1,wt} + \tau_{1,j})age_i + \varepsilon_{ij}$$

where $j \in \{wt, NrlKO\}$

$$i = 1,2,\ldots,n_j$$

$$\tau_{0,wt} = \tau_{1,wt} \equiv 0$$




```
> jFit <- lm(gExp ~ gType * age, jDat)
> summary(jFit)

Call:
lm(formula = gExp ~ gType * age, data = jDat)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.05383 -0.41194 -0.02491  0.31295  1.14417 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)     8.07337    0.16552  48.776  < 2e-16 ***
gTypeNrlKO      0.13148    0.24070   0.546    0.588    
age            -0.06818    0.01215  -5.612 2.51e-06 ***
gTypeNrlKO:age  0.01019    0.01744   0.584    0.563    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5651 on 35 degrees of freedom
Multiple R-squared: 0.607,	Adjusted R-squared: 0.5733 
F-statistic: 18.02 on 3 and 35 DF,  p-value: 3.047e-07
```

The intercept for the knockouts is:

$$\alpha_{0,wt} + \tau_{0,\Delta Nrl}$$

and the slope for knockouts is:

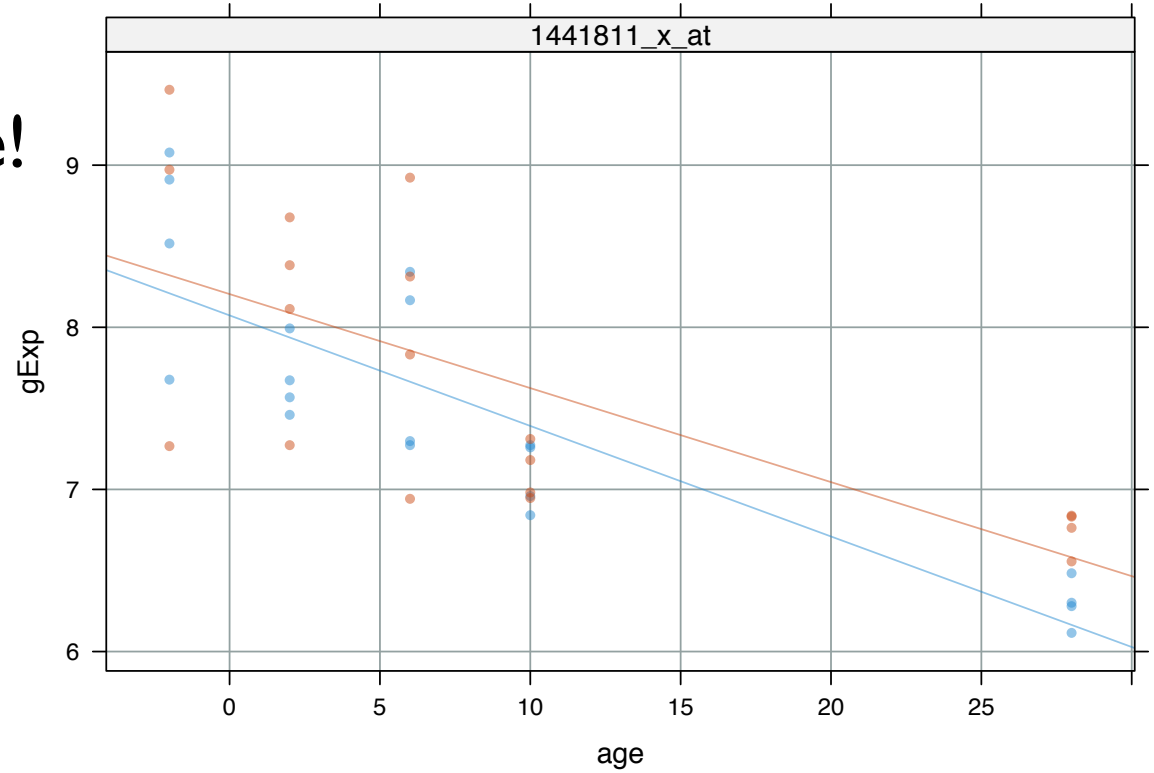
$$\alpha_{1,wt} + \tau_{1,\Delta Nrl}$$

as always, different parametrizations are possible!

$$y_{ij} = \alpha_{0,j} + \alpha_{1,j} age_i + \varepsilon_{ij}$$

where $j \in \{wt, NrlKO\}$

$i = 1,2,\ldots,n_j$




```
> jFitAlt <- lm(gExp ~ gType/age - 1, jDat)
> summary(jFitAlt)

Call:
lm(formula = gExp ~ gType/age - 1, data = jDat)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.05383 -0.41194 -0.02491  0.31295  1.14417 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
gTypewt         8.07337    0.16552  48.776  < 2e-16 ***
gTypeNrlKO      8.20485    0.17476  46.949  < 2e-16 ***
gTypewt:age    -0.06818    0.01215  -5.612 2.51e-06 ***
gTypeNrlKO:age -0.05799    0.01251  -4.636 4.80e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5651 on 35 degrees of freedom
Multiple R-squared: 0.9951,  Adjusted R-squared: 0.9945 
F-statistic:  1761 on 4 and 35 DF,  p-value: < 2.2e-16
```

(intercept, slope) for wild type:

$(\alpha_{0,wt}, \alpha_{1,wt})$

(intercept, slope) for the knockouts:

$(\alpha_{0,\Delta Nrl}, \alpha_{1,\Delta Nrl})$

as always, you can switch between parametrizations via multiplication by an appropriate contrast matrix!

$$y_{ij} = \alpha_{0,wt} + \tau_{0,j} + (\alpha_{1,wt} + \tau_{1,j})age_i + \varepsilon_{ij}$$

where $j \in \{wt, NrlKO\}$

$i = 1,2,\ldots,n_j$

$\tau_{0,wt} = \tau_{1,wt} \equiv 0$

$$\Leftrightarrow$$

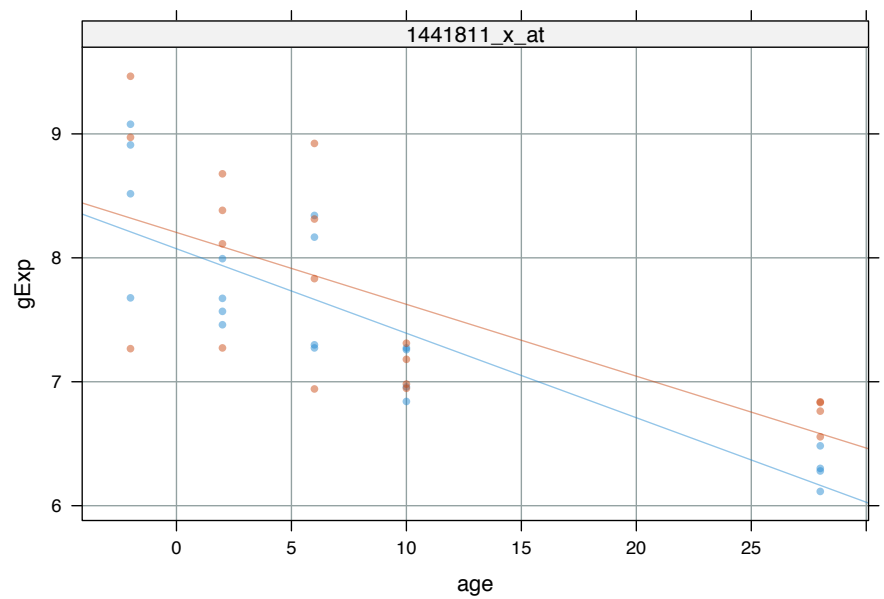
$$y_{ij} = \alpha_{0,j} + \alpha_{1,j}age_i + \varepsilon_{ij}$$

where $j \in \{wt, NrlKO\}$

$i = 1,2,\ldots,n_j$

```
> (contMat <- rbind(c(1, 0, 0, 0),
+                   c(1, 1, 0, 0),
+                   c(0, 0, 1, 0),
+                   c(0, 0, 1, 1)))
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    1    1    0    0
[3,]    0    0    1    0
[4,]    0    0    1    1
```

```
> cbind(coefDefault = coef(jFit),
+       coefAlt = coef(jFitAlt),
+       matrixResult = as.vector(contMat %*% coef(jFit)))
               coefDefault     coefAlt matrixResult
(Intercept)     8.07337352  8.07337352   8.07337352
gTypeNrlKO      0.13147574  8.20484926   8.20484926
age            -0.06817881 -0.06817881  -0.06817881
gTypeNrlKO:age  0.01018928 -0.05798953  -0.05798953
```

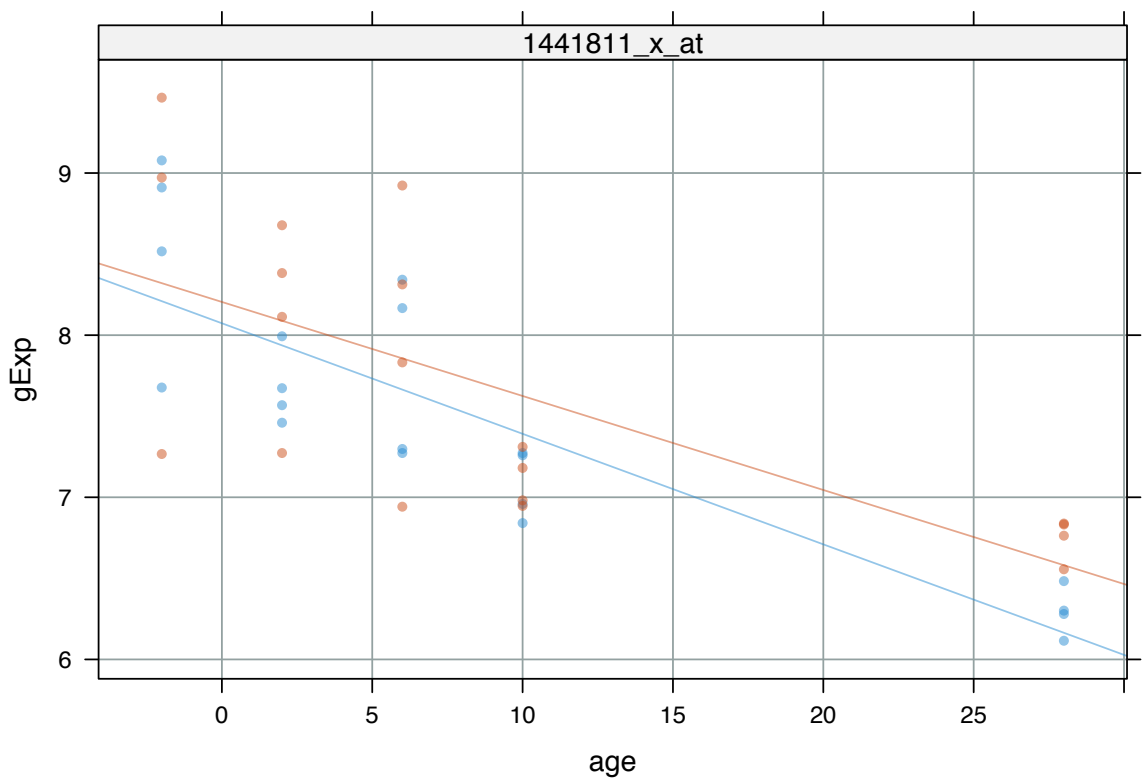


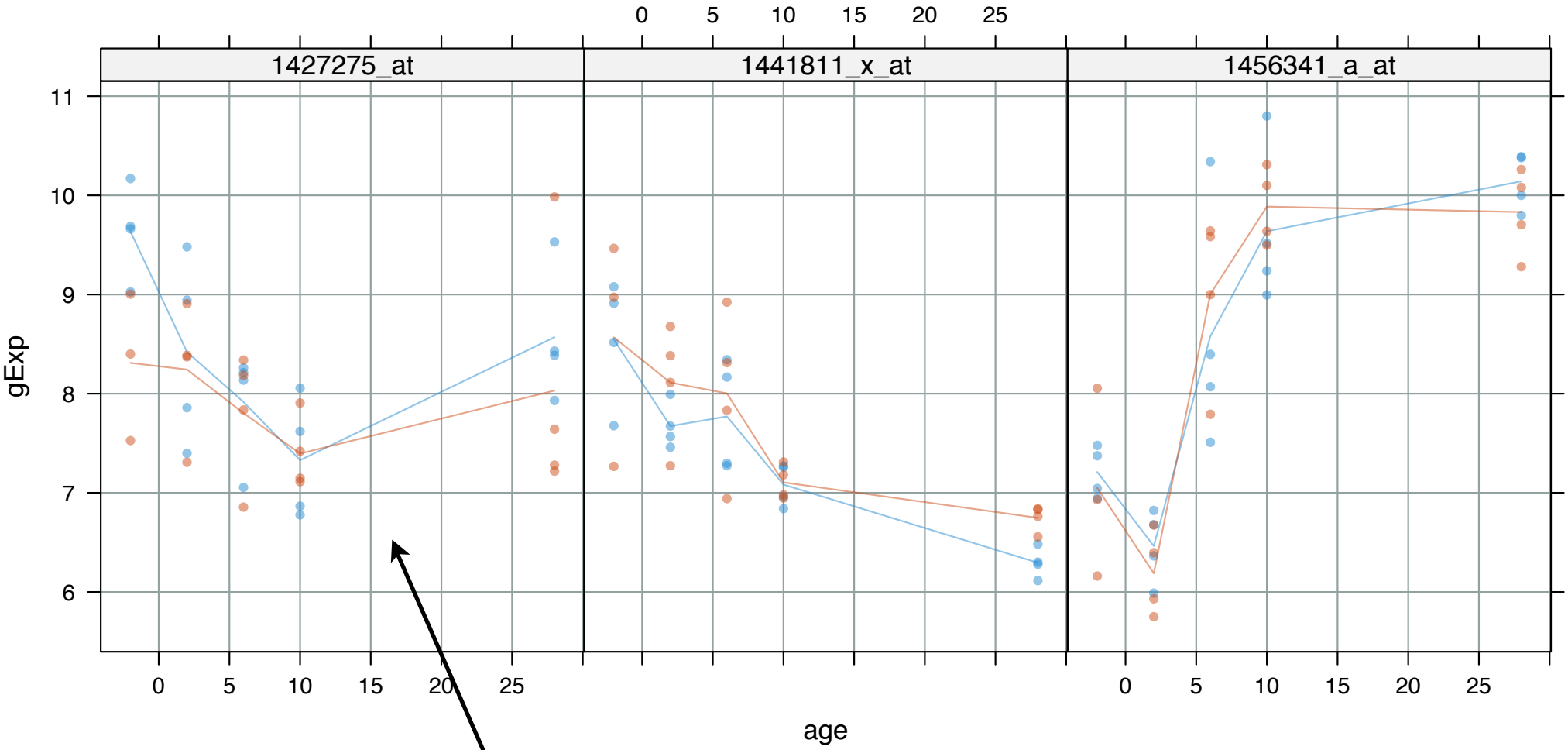
as always, you can assess the relevance of several terms at once -- such as everything involving genotype -- with an F test

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

small big

```
> anova(lm(gExp ~ age, jDat), jFit)

Analysis of Variance Table

Model 1: gExp ~ age
Model 2: gExp ~ gType * age
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     37 11.774
2     35 11.176  2   0.59807 0.9365 0.4016
```





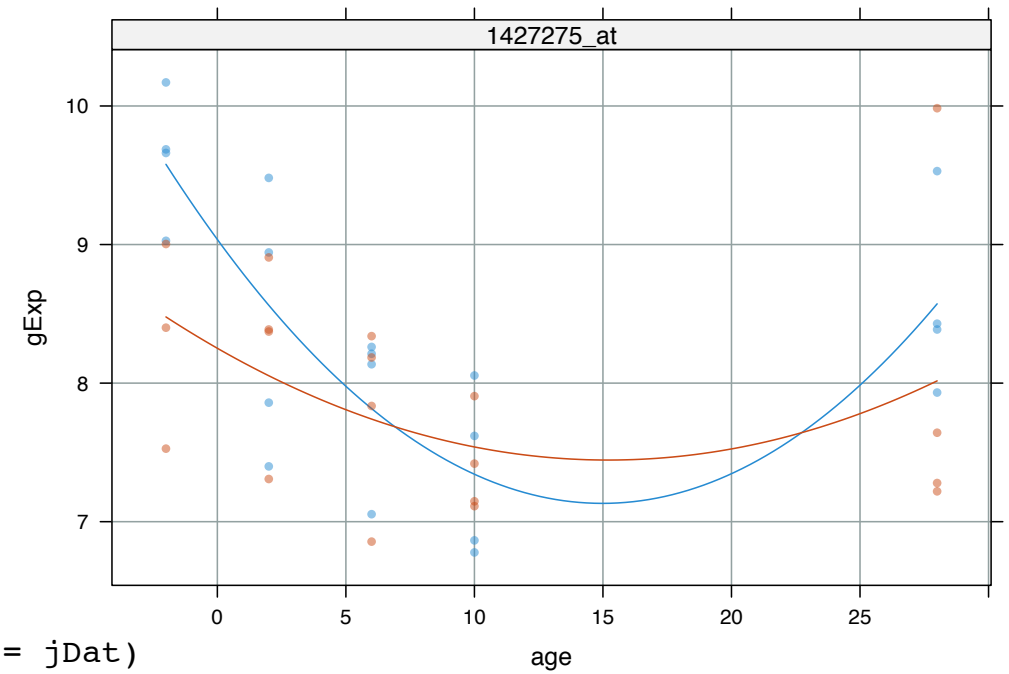
let's focus here for a model including a quadratic age term

$$y_{ij} = \alpha_{0,wt} + \tau_{0,j} + (\alpha_{1,wt} + \tau_{1,j})age_i + (\alpha_{2,wt} + \tau_{2,j})age_i^2 + \varepsilon_{ij}$$

where $j \in \{wt, NrlKO\}$

$i = 1,2,\ldots,n_j$

$\tau_{0,wt} = \tau_{1,wt} = \tau_{2,wt} \equiv 0$




```
> summary(jFit)

Call:
lm(formula = gExp ~ gType * (age + I(age^2)), data = jDat)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.16275 -0.55816  0.08203  0.42020  1.96803 

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)    
(Intercept)          9.036401   0.234853  38.477  < 2e-16 ***
gTypeNrlKO          -0.784969   0.350249  -2.241   0.0319 *  
age                 -0.254305   0.053234  -4.777 3.55e-05 ***
I(age^2)             0.008490   0.001838   4.620 5.63e-05 ***
gTypeNrlKO:age       0.148195   0.078232   1.894   0.0670 .  
gTypeNrlKO:I(age^2) -0.005001   0.002673  -1.871   0.0702 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7128 on 33 degrees of freedom
Multiple R-squared: 0.4755,  Adjusted R-squared: 0.3961 
F-statistic: 5.984 on 5 and 33 DF,  p-value: 0.0004804
```

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

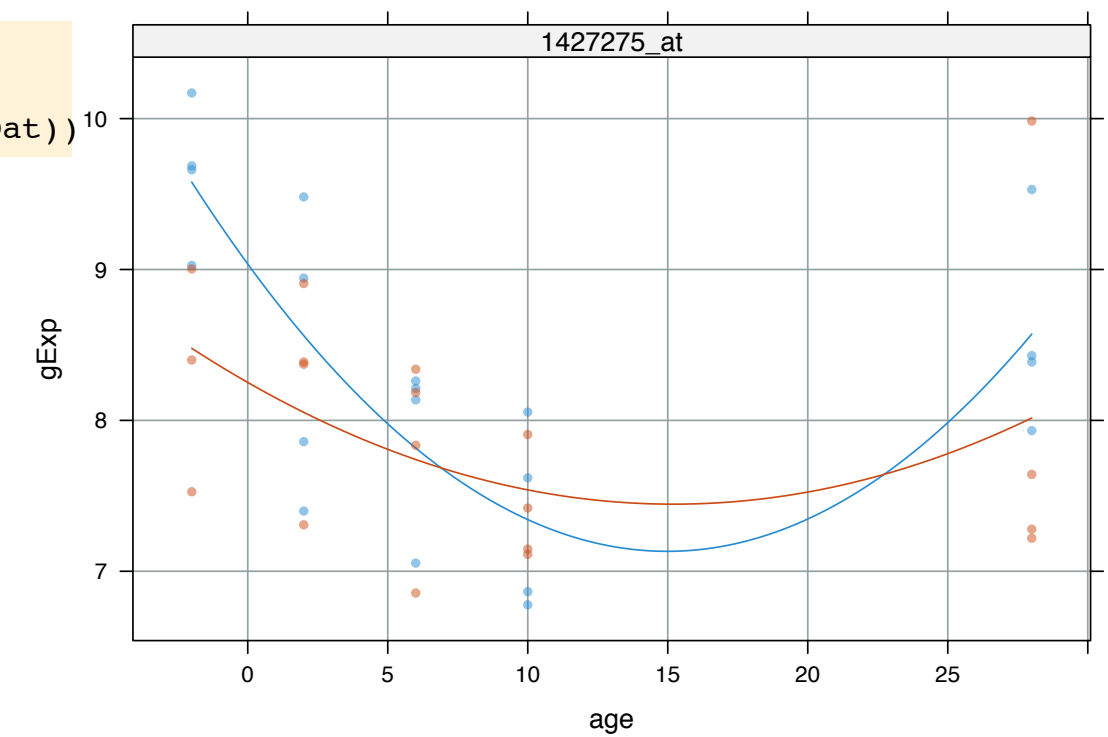
borderline evidence that genotype affects something about the parabola (location or shape)

small

```
> anova(lm(gExp ~ age + I(age^2), jDat),
+       lm(gExp ~ gType * (age + I(age^2)), jDat))
```

big

```
Analysis of Variance Table

Model 1: gExp ~ age + I(age^2)
Model 2: gExp ~ gType * (age + I(age^2))
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     36 20.081
2     33 16.767  3    3.3144 2.1744 0.1097
```



linear model framework is extremely general!

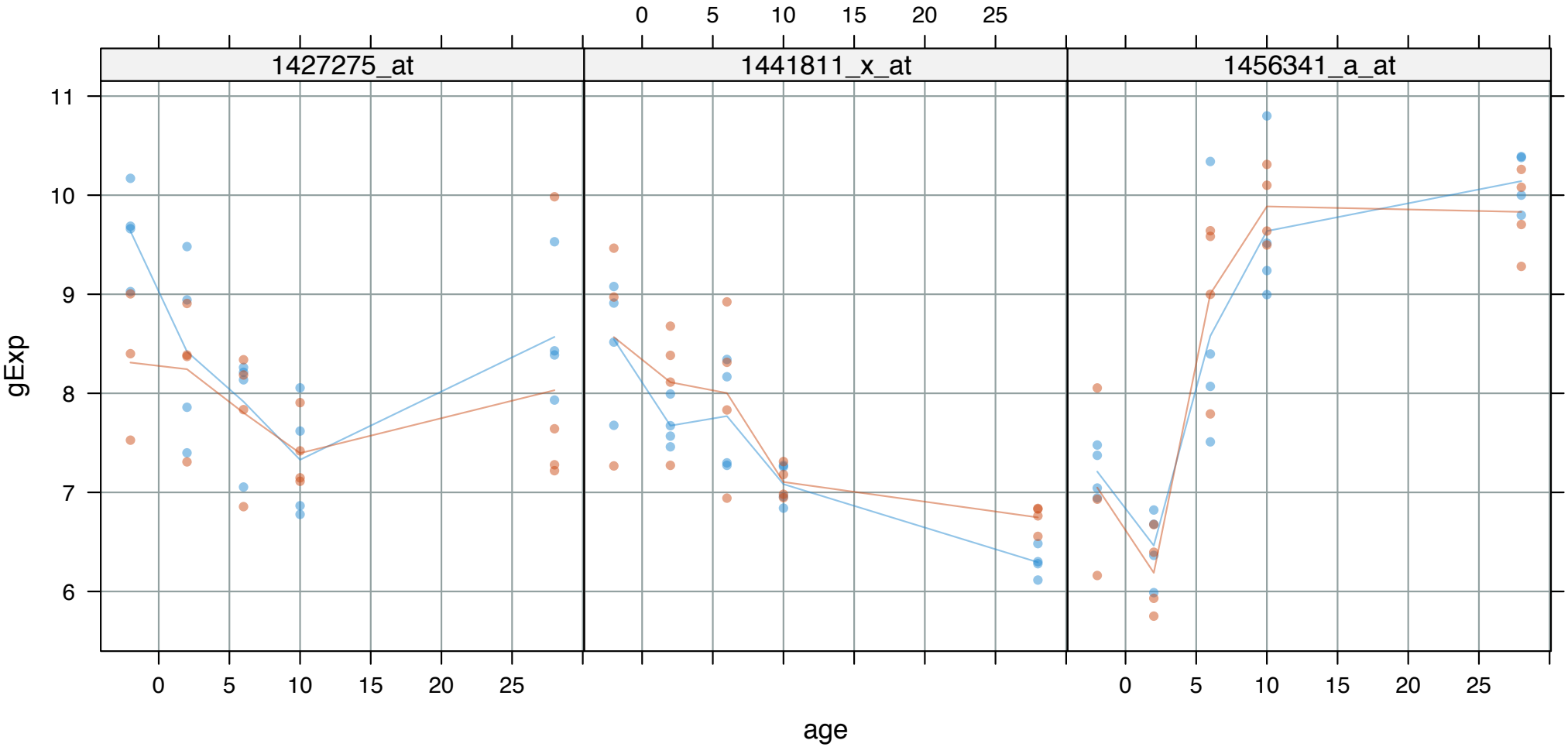
one extreme (simple): two-sample common variance t-test

another extreme (flexible): a polynomial, potentially different for each level of some factor

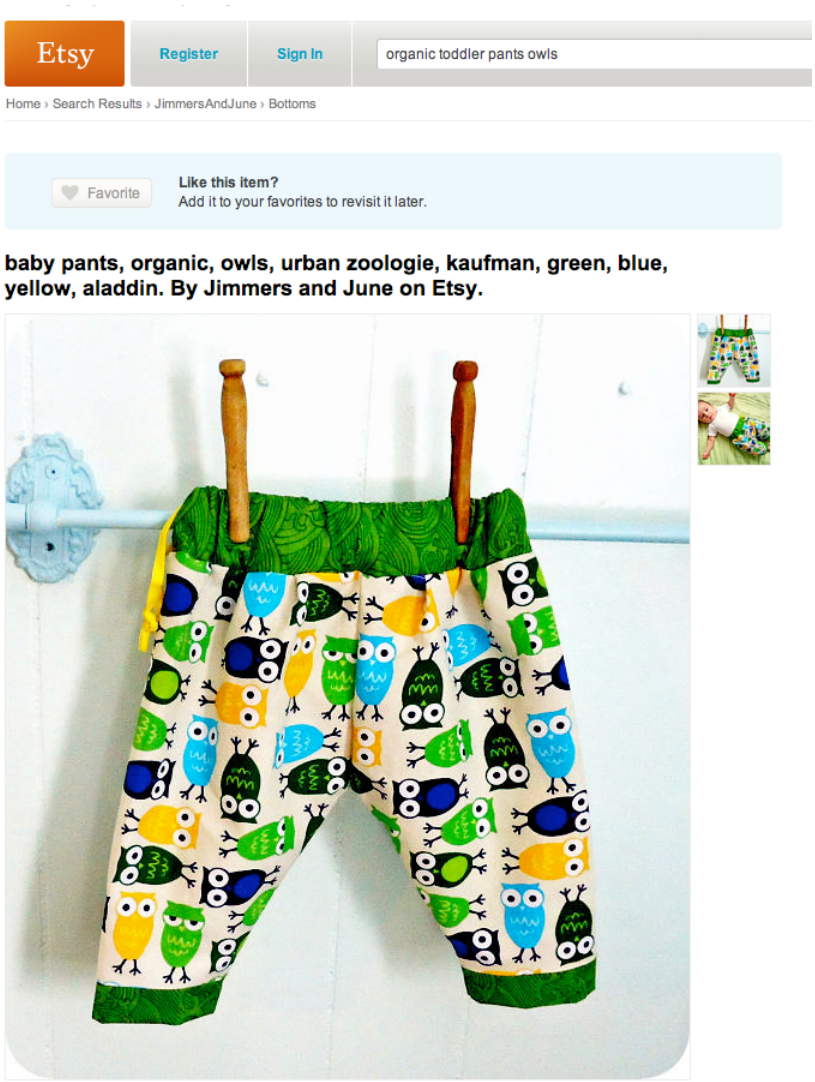
dichotomous variable? OK!
categorical variable? OK!
quantitative variable? OK!
various combinations of the above? OK!

don't be afraid to build models with more than 1 covariate
don't be intimidated by all the "contrast" talk

that's truly all I have to say
about linear models *per se*



What about the other 29,946 probesets?



```
lm(yMat ~ x)
lmFit(...) # from limma
```

```
lm(y ~ x)
```

```
by(myDat,
   gene,
   lm(y ~ x))
# or any other apply-ish approach
```

`lm(yMat ~ x)`

$$Y = X\alpha + \varepsilon$$

$$\begin{bmatrix} y_{11} & & y_{1G} \\ y_{21} & & y_{2G} \\ \vdots & \cdots & \\ & & \\ y_{n1} & & y_{nG} \end{bmatrix} = X \begin{bmatrix} \alpha_1 & \cdots & \alpha_G \end{bmatrix} + \begin{bmatrix} \varepsilon_{11} & & \varepsilon_{1G} \\ \varepsilon_{21} & & \varepsilon_{2G} \\ \vdots & \cdots & \\ & & \\ \varepsilon_{n1} & & \varepsilon_{nG} \end{bmatrix}$$

built-in function lm() can do “multivariate regression” = many dependent vars (“responses”)
aka “multivariate multiple regression”

```
From lm() documentation:

If response is a matrix a linear model is fitted separately by least-squares to each
column of the matrix.

lm returns an object of class "lm" or for multiple responses of class c("mlm", "lm").
```

Industrial scale model fitting is good because things like this are not recomputed 30K times unnecessarily*

$Y = X\alpha + \varepsilon$ regression model

$\hat{\alpha} = (X^T X)^{-1} X^T Y$ the MLE and OLS estimator of α

$\hat{\sigma}^2 = \frac{1}{n-p} \hat{\varepsilon}^T \hat{\varepsilon}$ the estimated error variance

$\hat{V}(\hat{\alpha}) = \hat{\sigma}^2 (X^T X)^{-1}$ the estimated covariance matrix of $\hat{\alpha}$

How test $H_0 : \alpha_j = 0$?

With a t-statistic. Under H_0, we have (at least approximately) that:

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

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.

* under the hood, lm() is doing something more clever and numerically stable than this

The problem, in a nutshell

responses ~ model formula, data

fit a separate linear model for each response, e.g. gene

```
lm(yMat ~ x)
lmFit(...)
```

fitted models

extract estimated parameters
or p-values or ...
compare big models to small
etc etc

```
?????????
```

```
> methods(class = "lm")
 [1] Anova.lm*               Boot.lm*                add1.lm*
 [4] addterm.lm*             alias.lm*               anova.lm
 [7] avPlot.lm*              bootCase.lm*            boxCox.lm*
[10] boxcox.lm*              case.names.lm*          ceresPlot.lm*
[13] confidenceEllipse.lm*   confint.lm*             cooks.distance.lm*
[16] crPlot.lm*              deltaMethod.lm*         deviance.lm*
[19] dfbeta.lm*              dfbetaPlots.lm*         dfbetas.lm*
[22] dfbetasPlots.lm*        drop1.lm*               dropterm.lm*
[25] dummy.coef.lm*          durbinWatsonTest.lm*    effects.lm*
[28] extractAIC.lm*          family.lm*              formula.lm*
[31] hatvalues.lm            hccm.lm*                infIndexPlot.lm*
[34] influence.lm*           influencePlot.lm*       inverseResponsePlot.lm*
[37] kappa.lm                labels.lm*              leveneTest.lm*
[40] leveragePlot.lm*        linearHypothesis.lm*    logLik.lm*
[43] logtrans.lm*            mmp.lm*                 model.frame.lm
[46] model.matrix.lm         ncvTest.lm*             nextBoot.lm*
[49] nobs.lm*                outlierTest.lm*         plot.lm
[52] powerTransform.lm*      predict.lm              print.lm
[55] proj.lm*                qqPlot.lm*              qr.lm*
[58] residualPlot.lm*        residualPlots.lm*       residuals.lm
[61] rstandard.lm            rstudent.lm             sigmaHat.lm*
[64] simulate.lm*            spreadLevelPlot.lm*     summary.lm
[67] variable.names.lm*      vcov.lm*                vif.lm*

> methods(class = "mlm")
 [1] SSD.mlm*          add1.mlm*         anova.mlm         deviance.mlm*
 [5] drop1.mlm*        estVar.mlm*       mauchly.test.mlm* plot.mlm
 [9] predict.mlm       summary.mlm       vcov.mlm*

   Non-visible functions are asterisked
```

Precious little support for working with objects of class mlm. **This is sad.***

* I have MacGyvered some of this stuff for myself but I can't inflict it on you.

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(...)`

limma workflow

limma is designed to help you out **AFTER** you've applied eBayes()

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(...)`

You will probably settle for plain vanilla modelling on a small scale, using lm()...

OR

Empirical Bayes flavored linear modelling on a large scale, using limma.

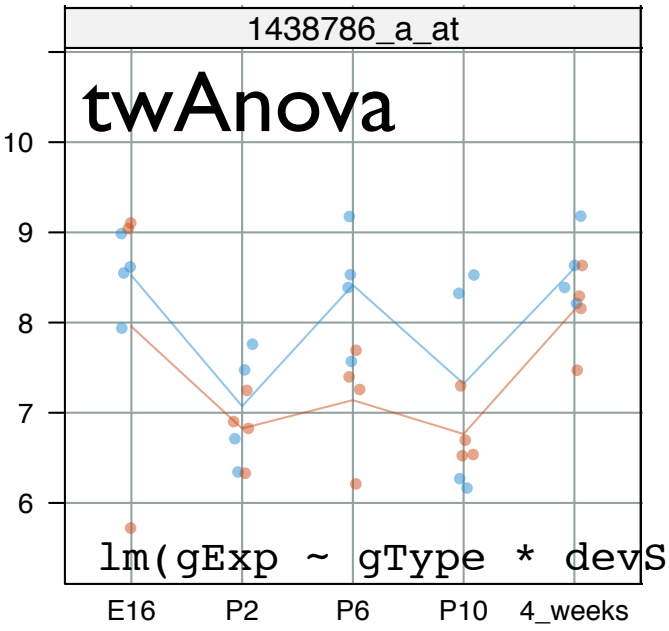
I have fit all the models we've considered to all ~30K probesets.

Let's examine some of the results *en masse*.

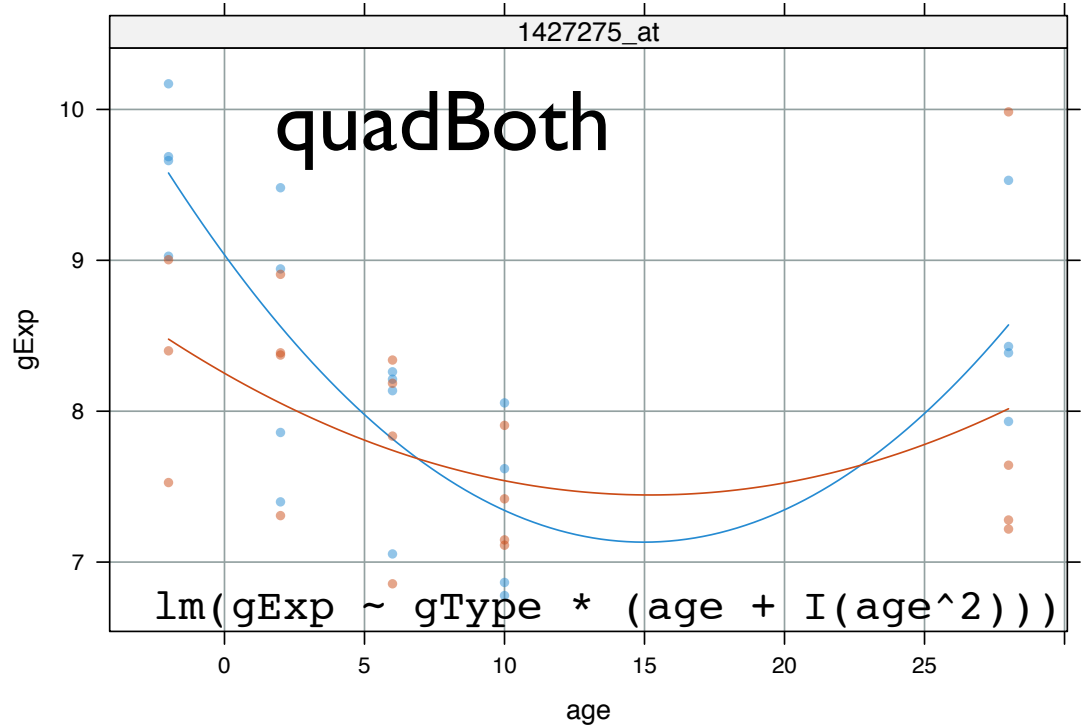
Let this drive home the point that ...

- background variability
- intercepts
- Nrl knockout effects
- devStage effects
- age effects, both linear and quadratic
- and interactions of all the above

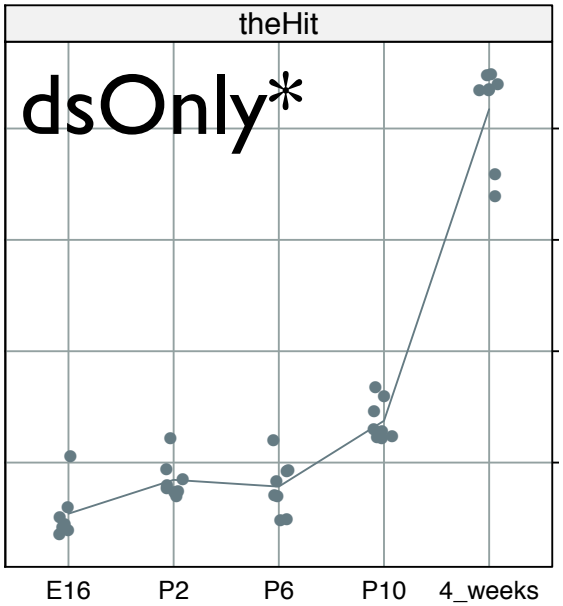
differ for each gene.



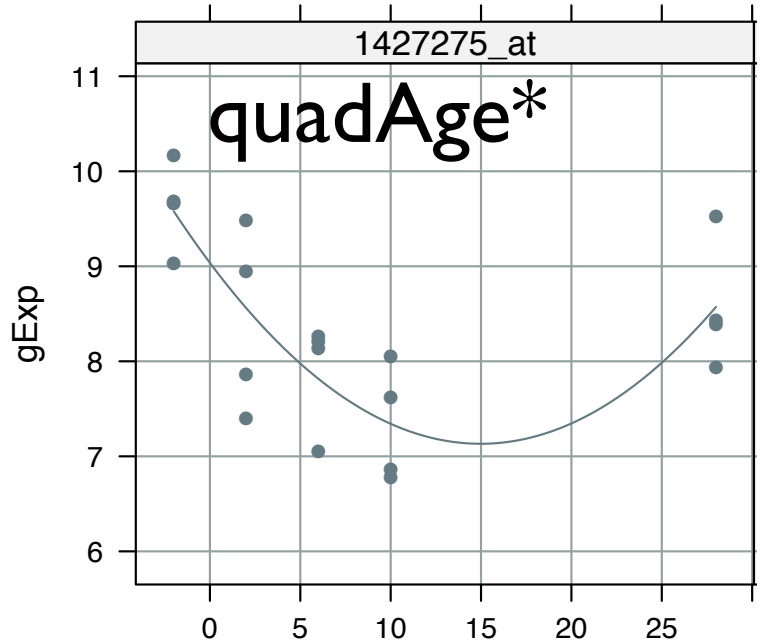
`lm(gExp ~ gType * devStage)`



`lm(gExp ~ gType * (age + I(age^2)))`

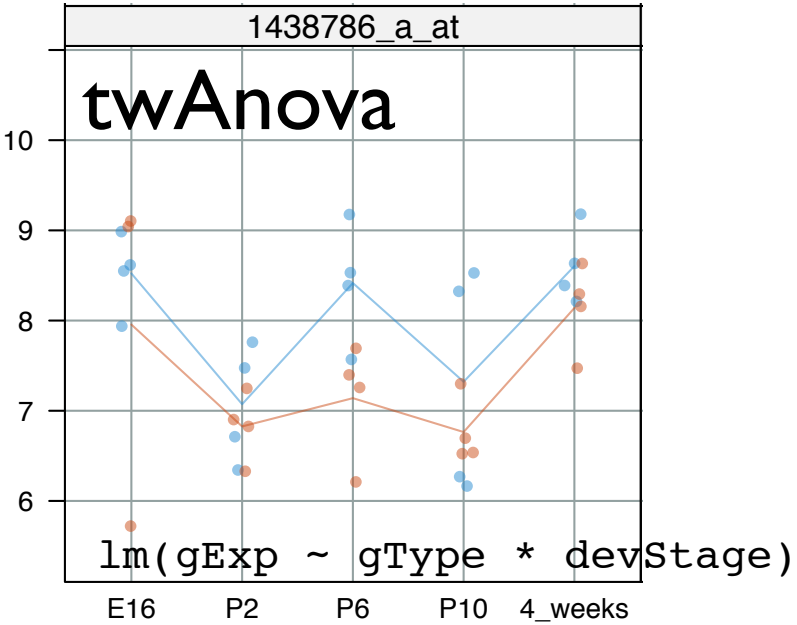


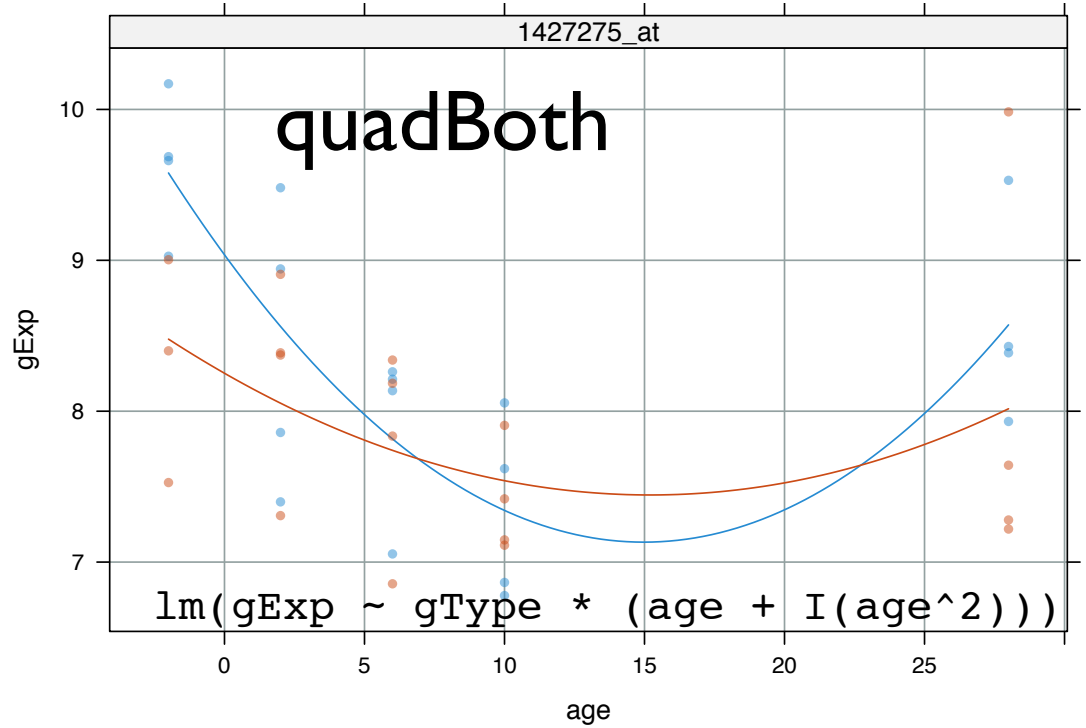
`lm(gExp ~ devStage)`

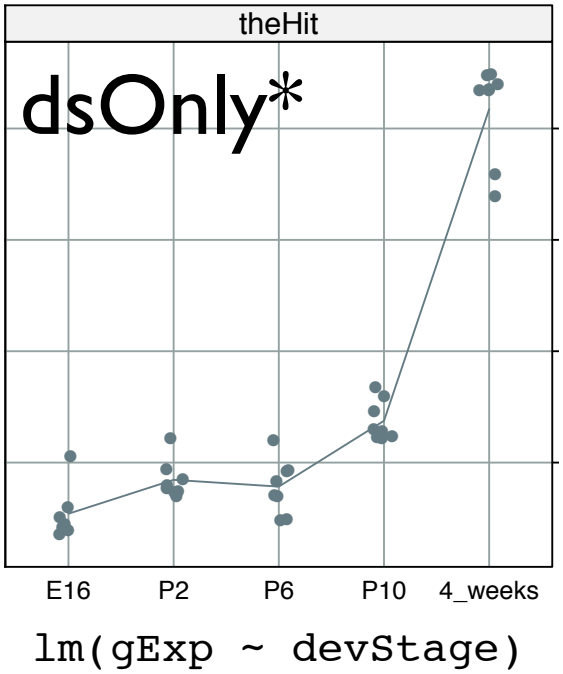


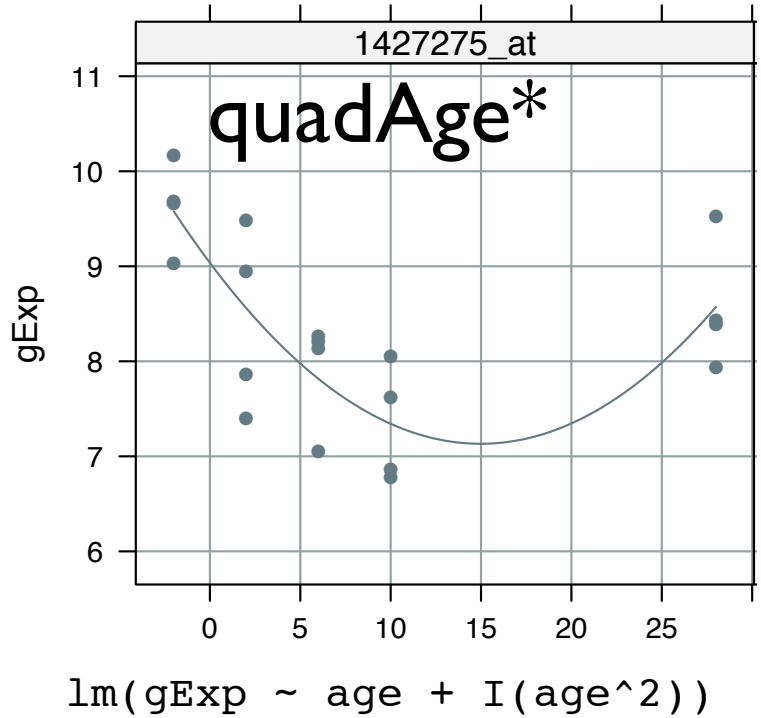
`lm(gExp ~ age + I(age^2))`

* Figures slightly misleading. Model is fit to all the data, wild type and Nrl knockout, but gType is not used as a covariate.

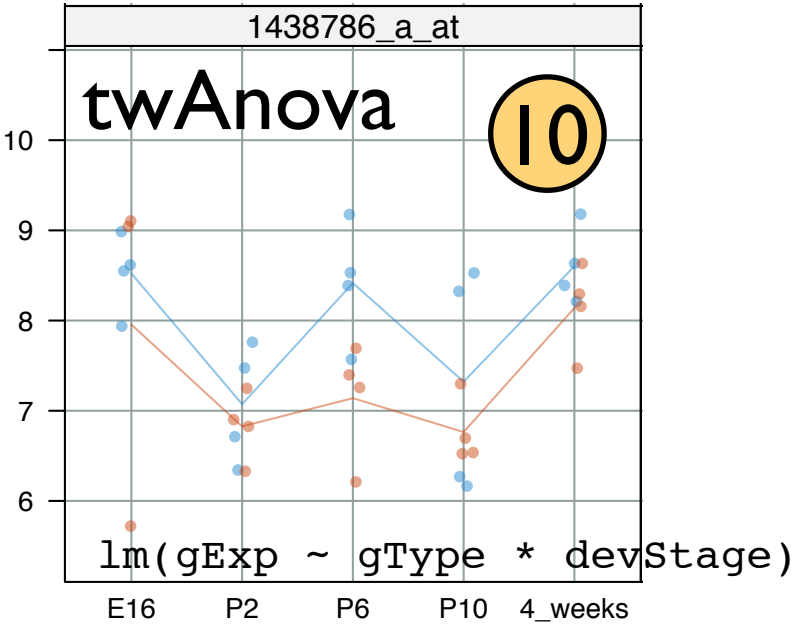


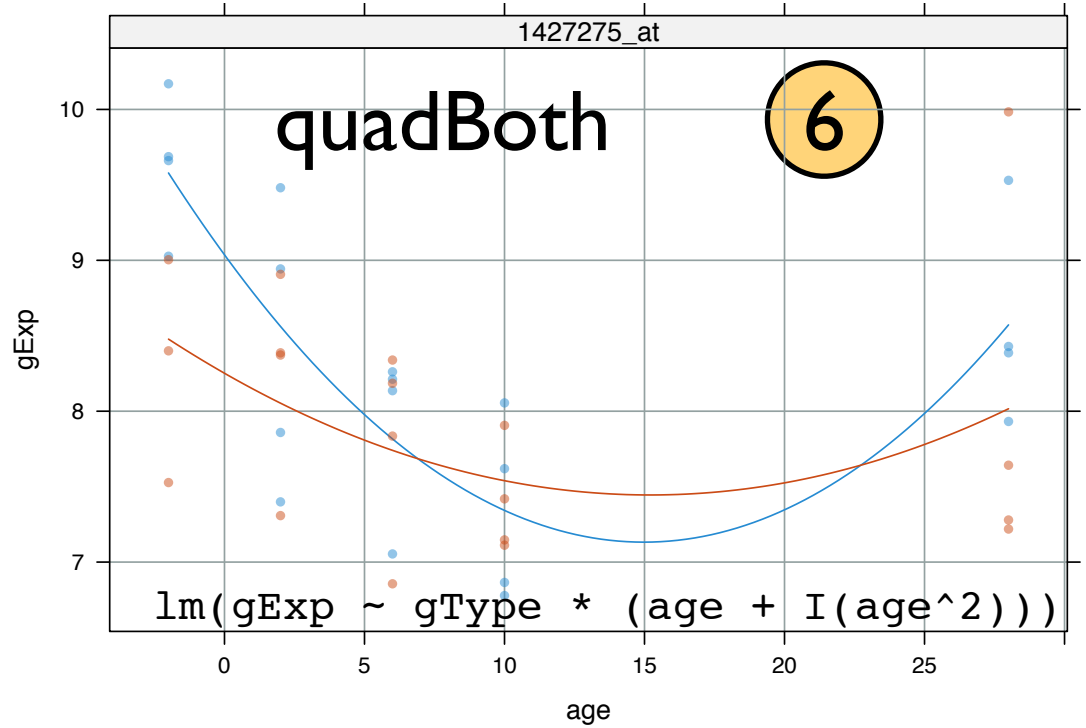


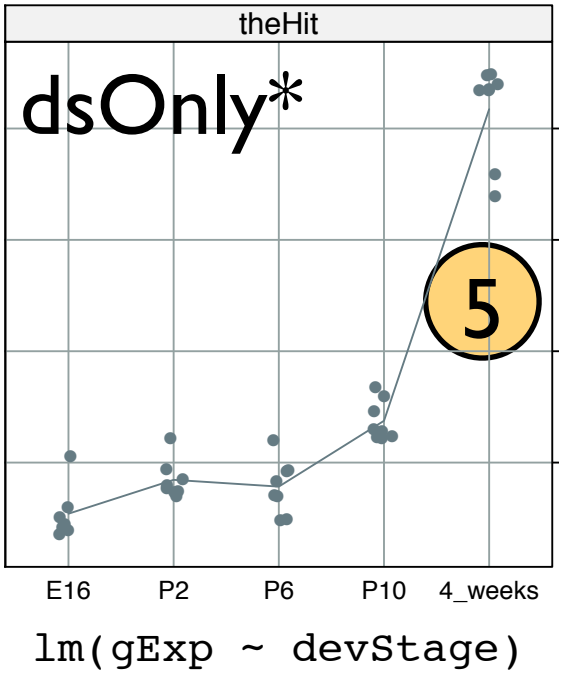


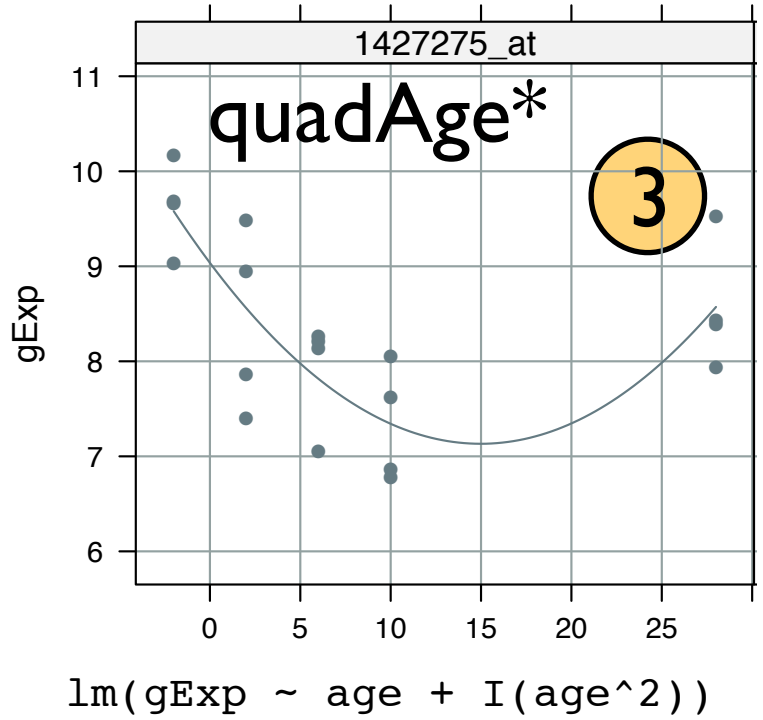


How "big" are these models? How many parameters are we using to specify the mean structure?





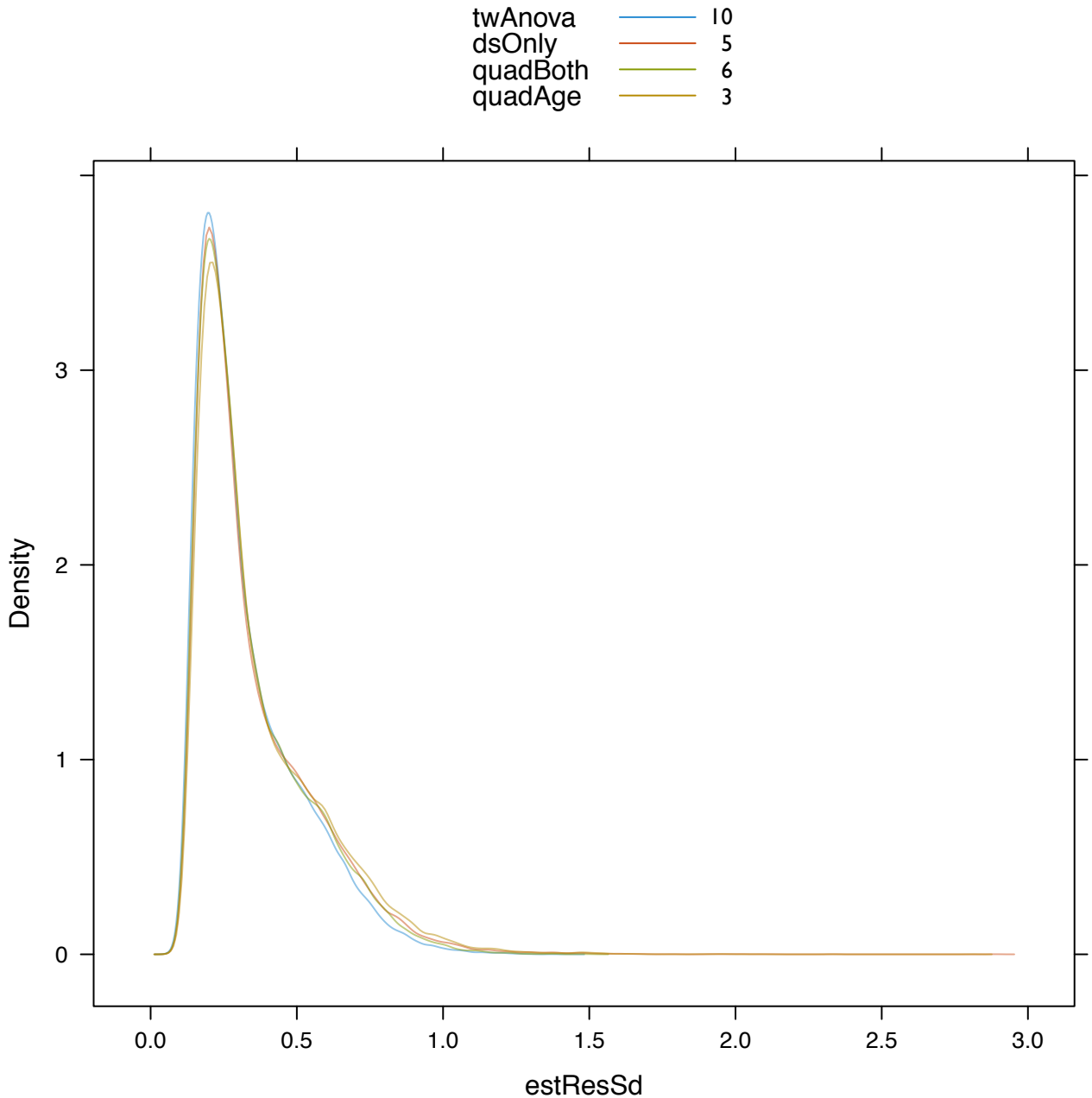




How "big" are these models? How many parameters are we using to specify the mean structure?

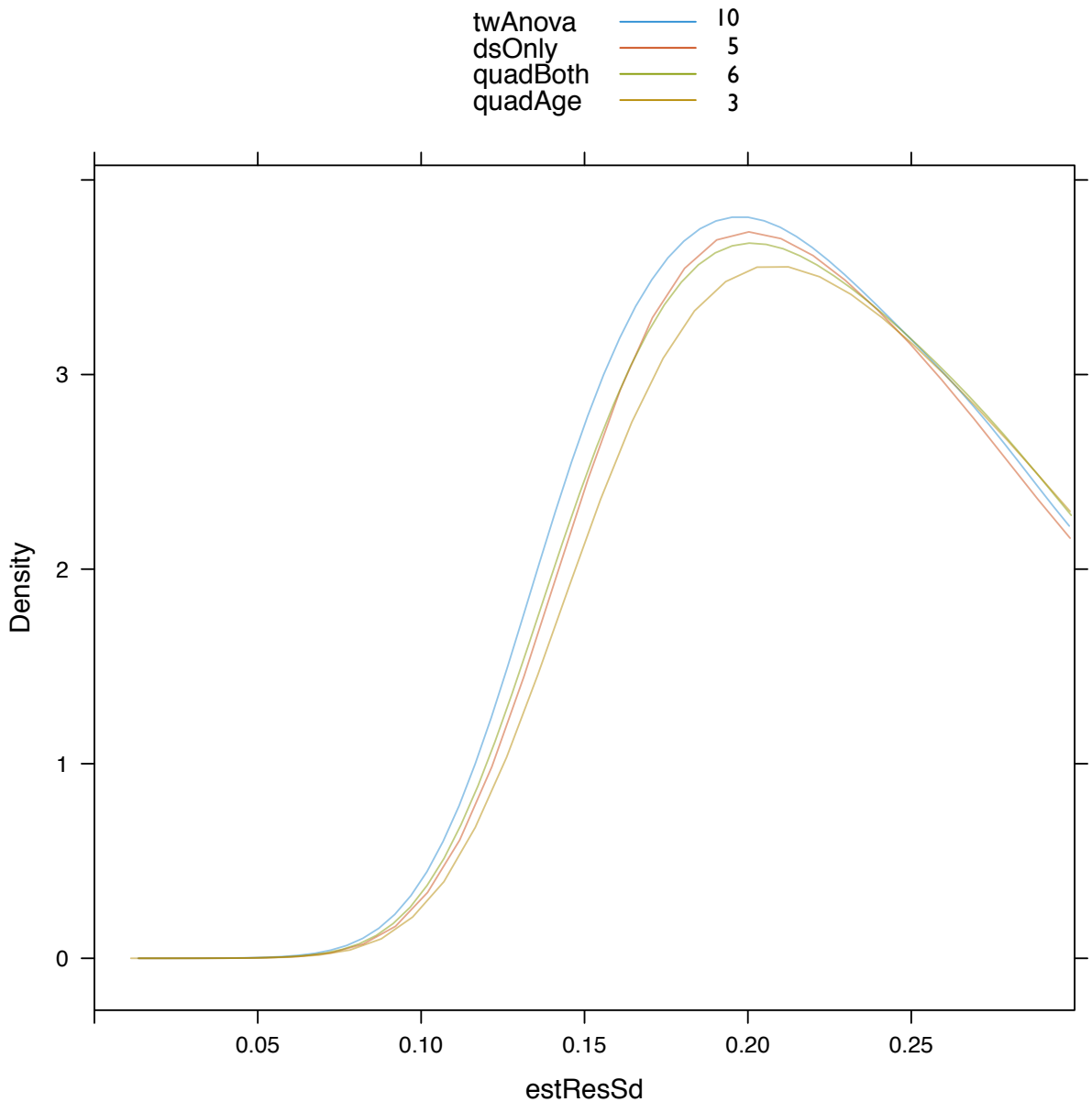
$$y_i = f(x_i;\alpha) + \varepsilon_i, \mathrm{var}(\varepsilon) = \sigma^2$$

Let's look at estimates of the error standard deviation.



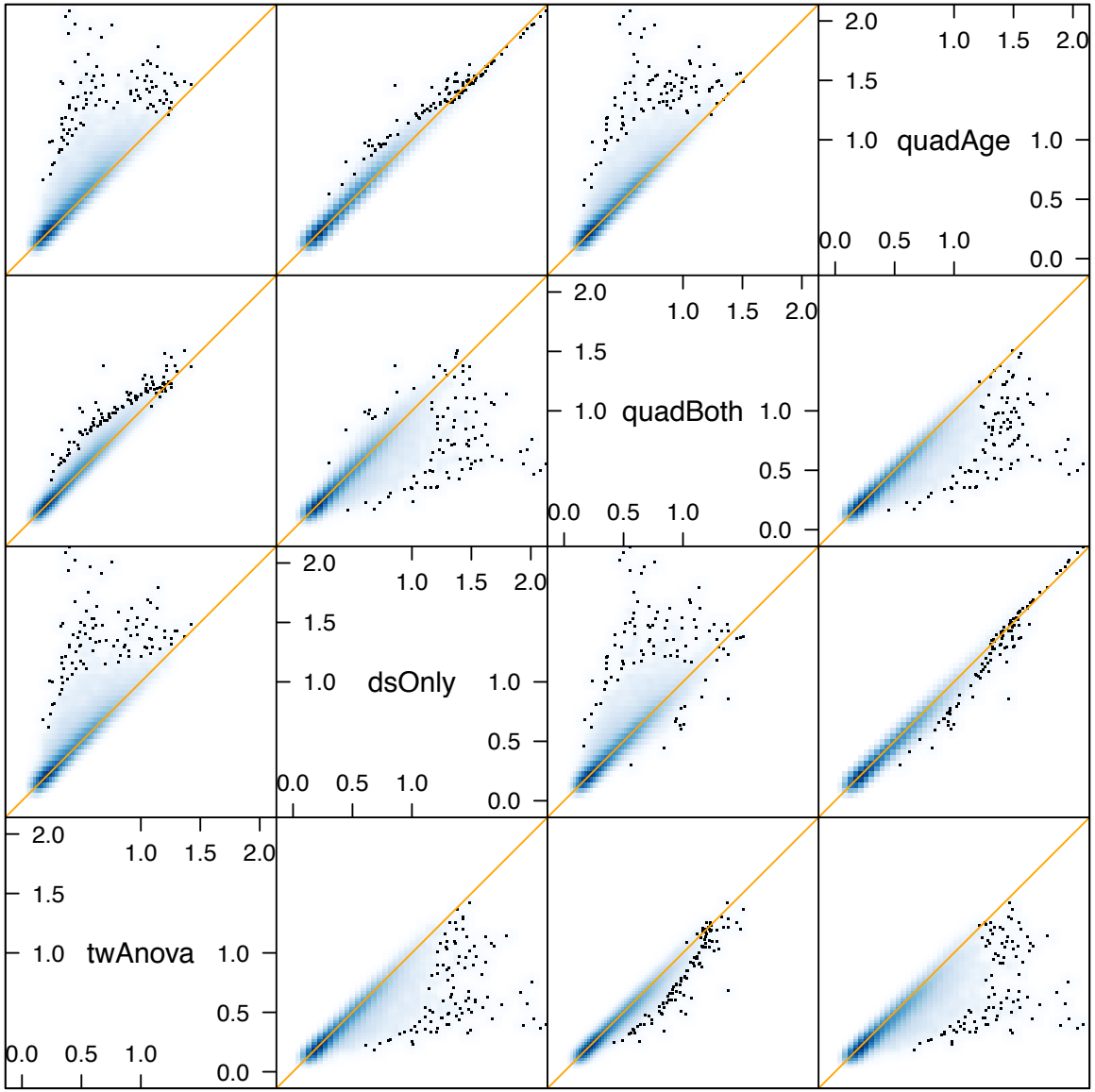
$$y_i = f(x_i;\alpha) + \varepsilon_i, \mathrm{var}(\varepsilon) = \sigma^2$$

Let's look at estimates of the error standard deviation.



$$y_i = f(x_i;\alpha) + \varepsilon_i, \mathrm{var}(\varepsilon) = \sigma^2$$

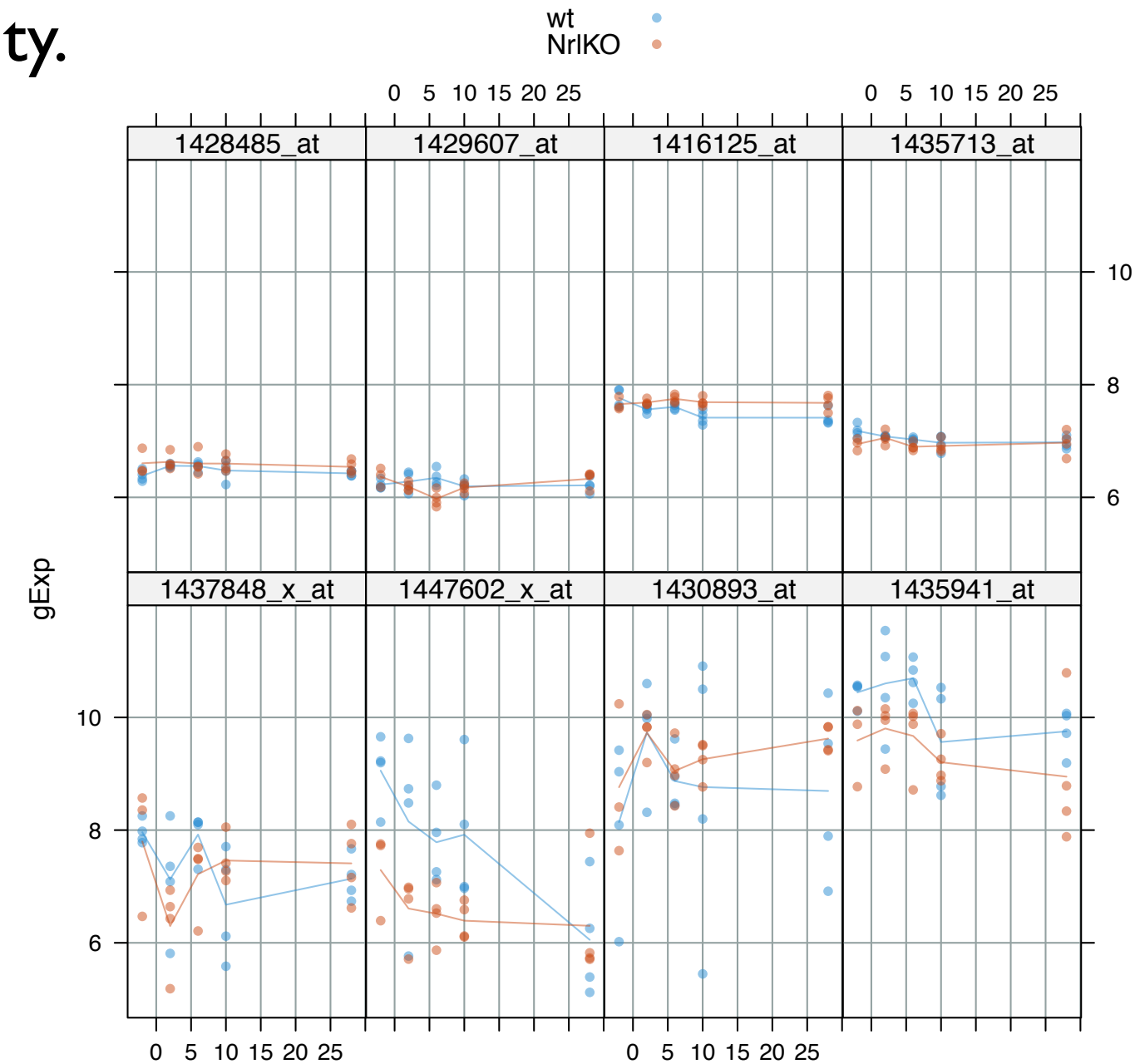
Let's look at estimates of the error standard deviation.



Scatter Plot Matrix

$$y_i = f(x_i;\alpha) + \varepsilon_i, \mathrm{var}(\varepsilon) = \sigma^2$$

Let's look genes exhibiting extremely low or high variability.

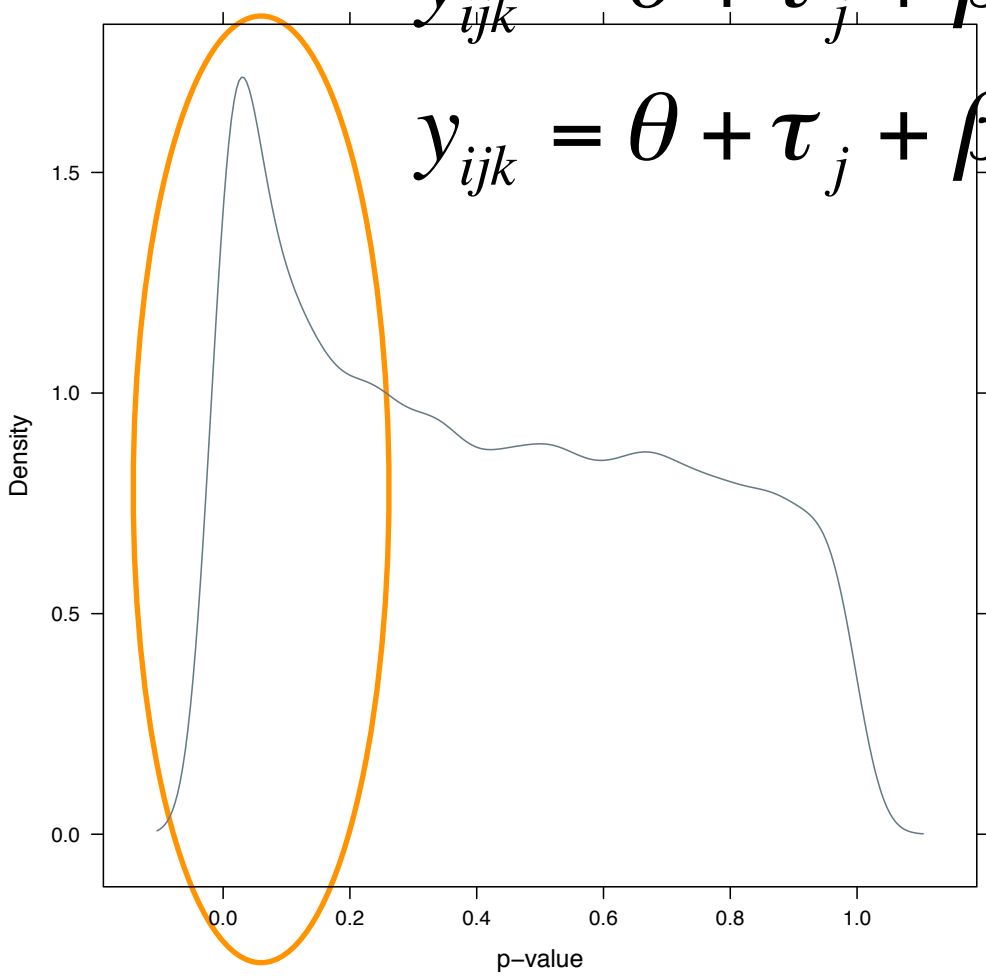


In the two-way ANOVA model, is there evidence for gType * devStage interaction? YES.

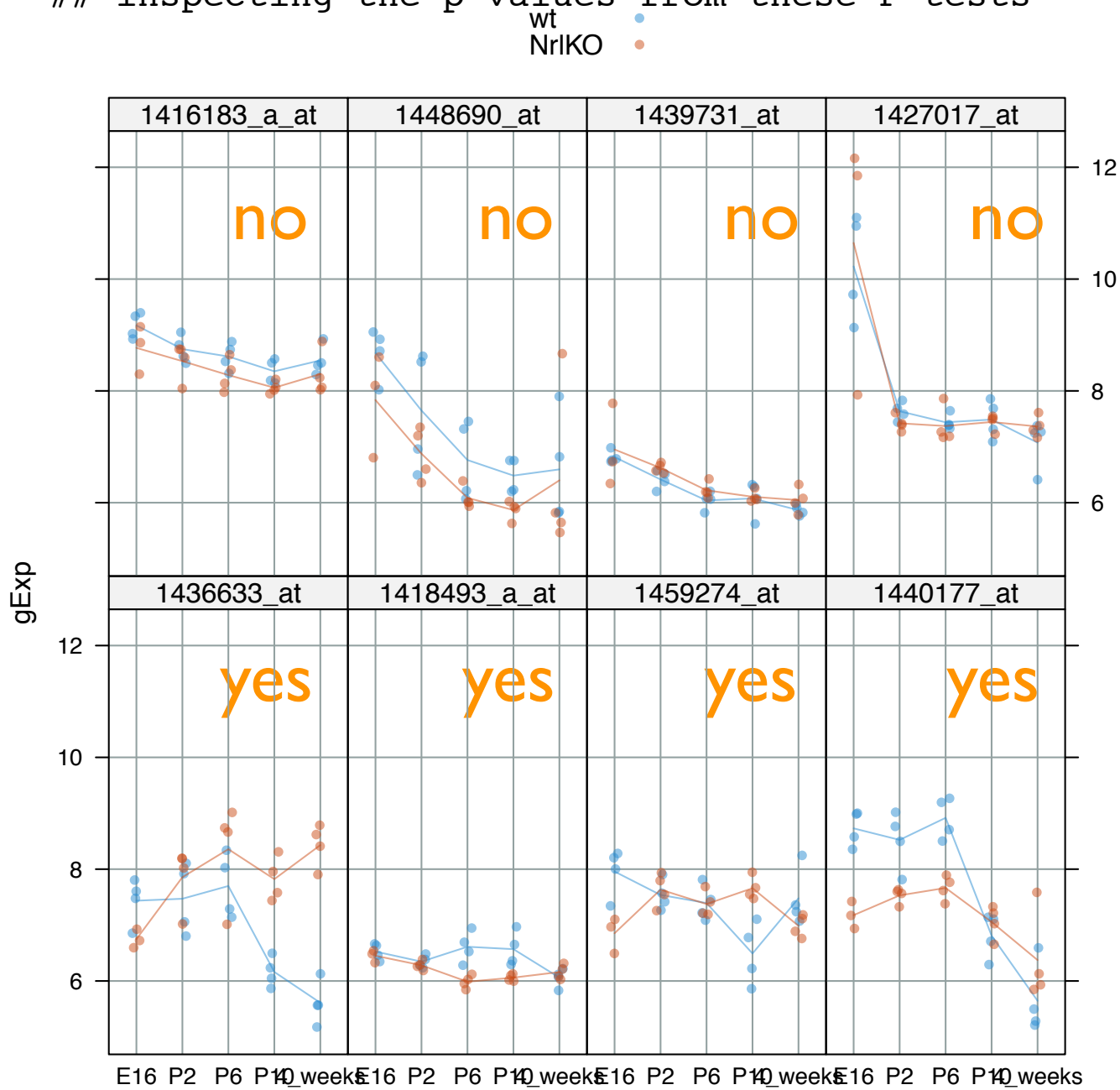
```
## this code is fictional but conveys the point
anova(lm(gExp ~ gType * devStage), lm(gExp ~ gType + devStage))
## inspecting the p-values from these F tests
```

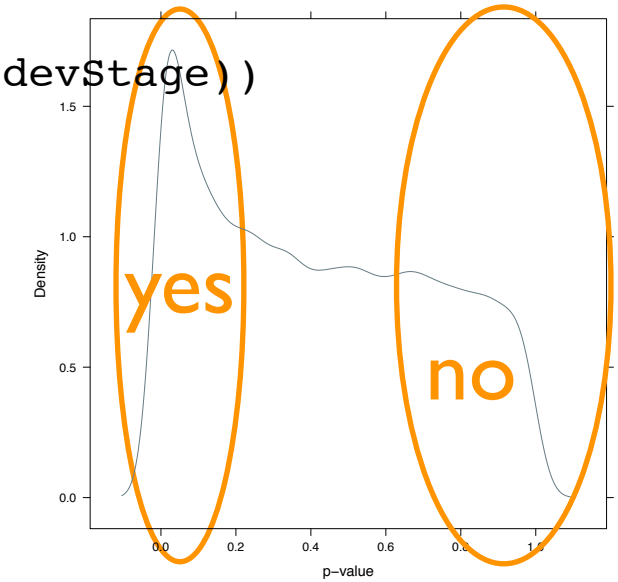
$$y_{ijk} = \theta + \tau_j + \beta_k + (\tau\beta)_{jk} + \varepsilon_{ijk} \text{ “big”}$$

$$y_{ijk} = \theta + \tau_j + \beta_k + (\tau\beta)_{jk} + \varepsilon_{ijk} \text{ “small”}$$



```
## this code is fictional but conveys the point
anova(lm(gExp ~ gType * devStage), lm(gExp ~ gType + devStage))
## inspecting the p-values from these F tests
```

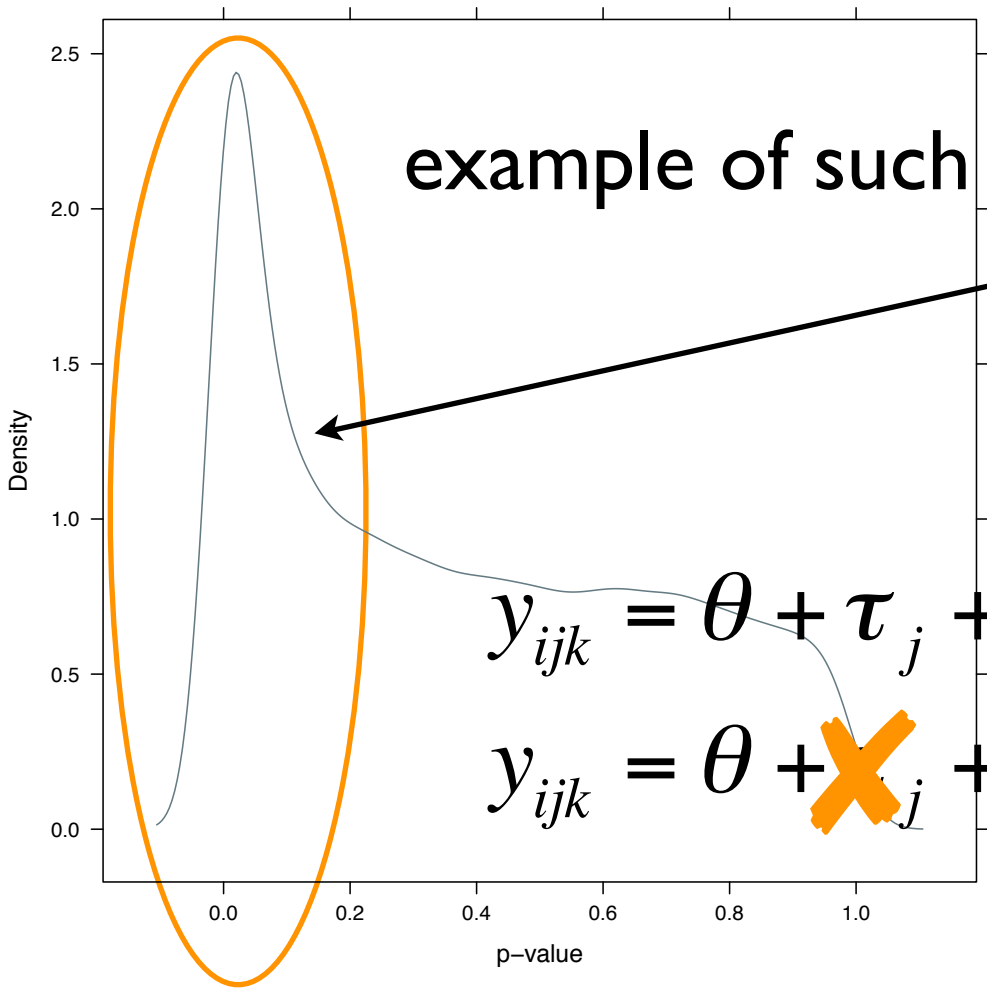




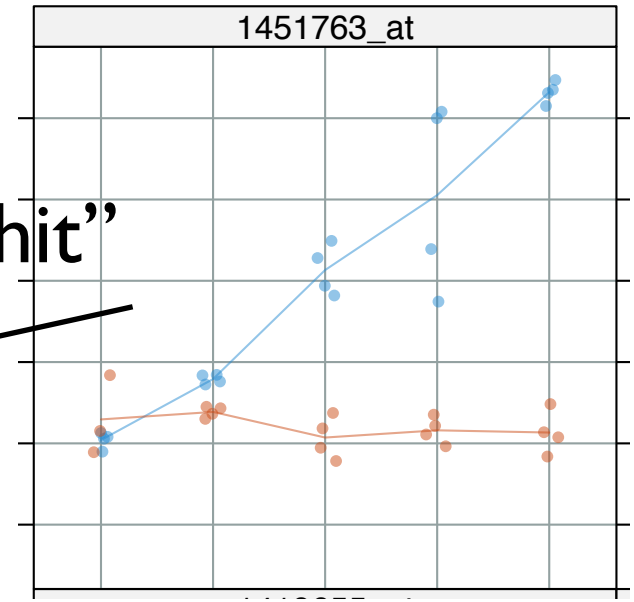
interaction?

In the two-way ANOVA model, is there evidence that genotype matters? YES.

```
## this code is fictional but conveys the point
anova(lm(gExp ~ gType * devStage), lm(gExp ~ devStage))
## inspecting the p-values from these F tests
```



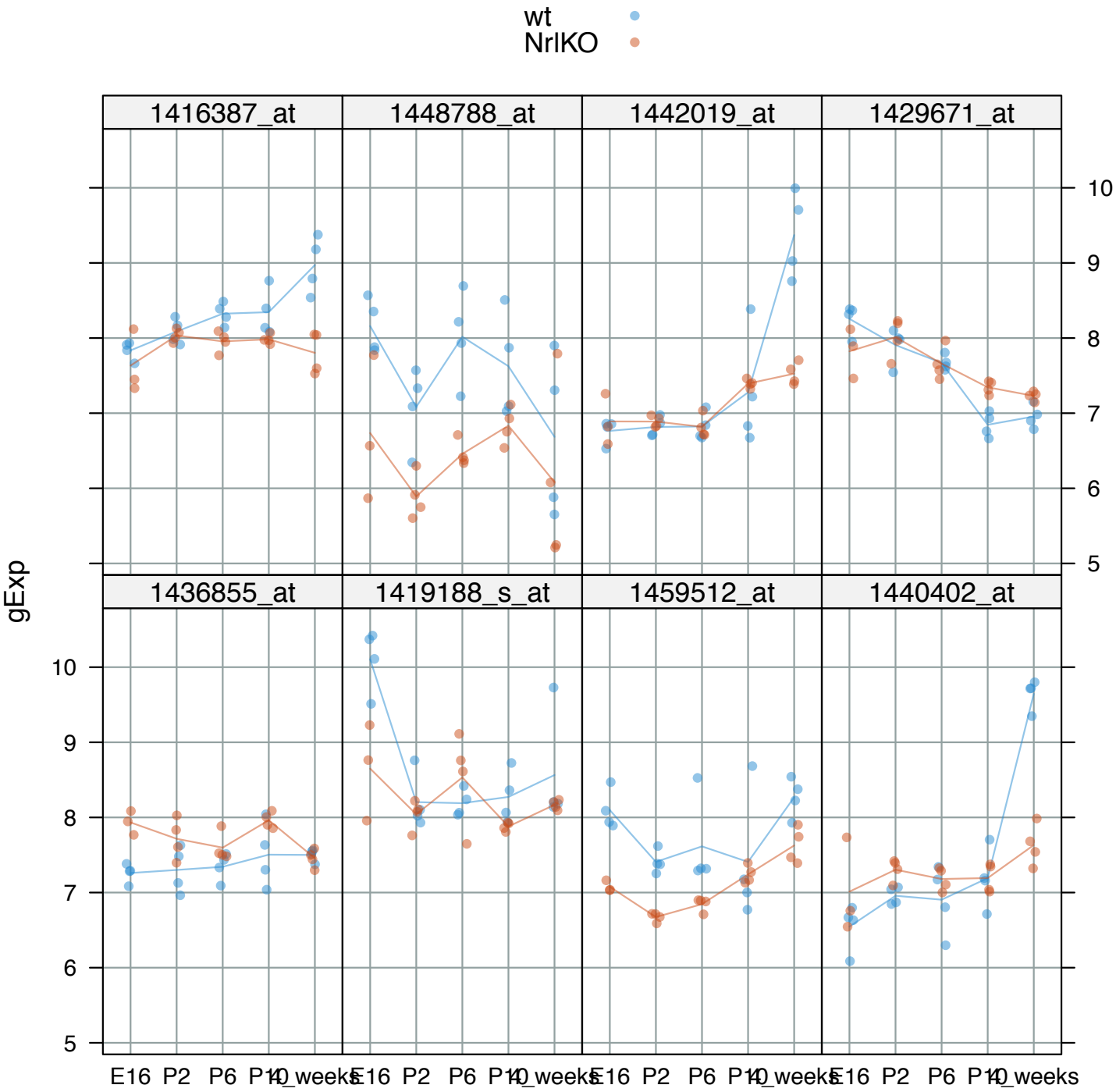
example of such a "hit"



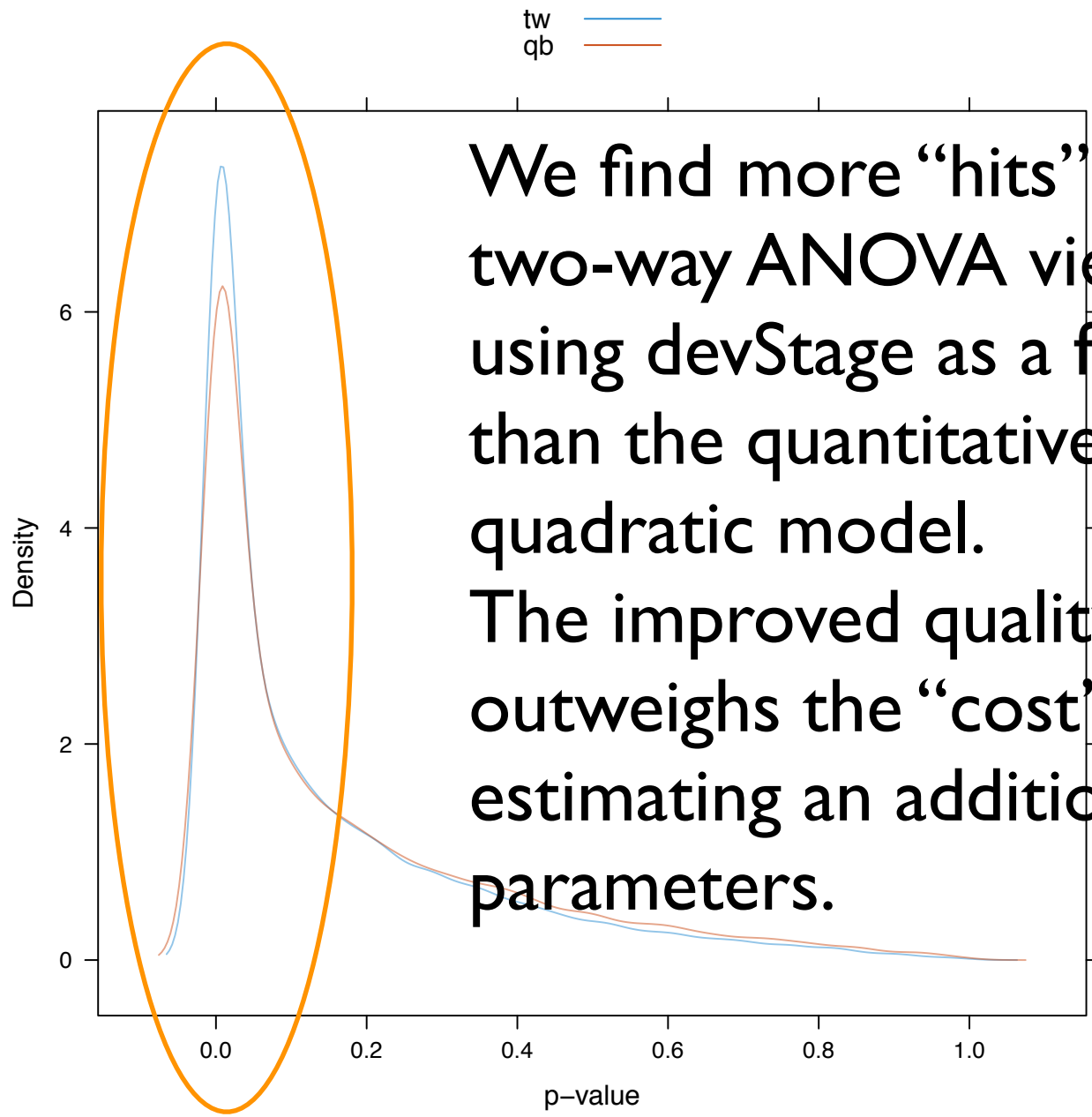
$$y_{ijk} = \theta + \tau_j + \beta_k + (\tau\beta)_{jk} + \varepsilon_{ijk} \quad \text{"big"}$$

$$y_{ijk} = \theta + \cancel{\tau_j} + \beta_k + \cancel{(\tau\beta)}_{jk} + \varepsilon_{ijk} \quad \text{"small"}$$

more “gType” hits within the ANOVA models



Looking at evidence of any differential expression at all (overall F test) in the two-way ANOVA model vs. the quadratic.



We find more "hits" with the two-way ANOVA viewpoint, i.e. using devStage as a factor rather than the quantitative age and a quadratic model.

The improved quality of fit outweighs the "cost" of estimating an additional 4 parameters.

where to next? ...Wednesday

in many studies, the # replicates is small relative to # params being estimated

can lead to crazy small estimates of error variance
which leads to crazy large test statistics
which leads to crazy small p-values
which leads to “hits” where the observed phenomenon is rather subtle

which leads to people saying the platform and/or analysis method and/or analyst is bad

moderating the variance estimates can be very helpful --> limma!

where to next? ... following Wednesday

multiple testing, large scale inference

analysis of high-throughput data results in thousands of "gene-wise" hypothesis tests

often, "gene-wise" analysis is relatively simple

BUT a recurring and thorny issue is how to handle thousands of p-values, each for a separate hypothesis test

how to guard against crazy # false positives?

which error rate is more relevant ... rate at which null genes are 'discovered' or rate at which 'discoveries' are null?