Part II: Generalized Linear Models

Load Packages

Again, we must load the packages that will be used in the second part of this workshop.

```r
library(pastecs, quietly = TRUE)
library(lm.beta,  quietly = TRUE)
library(lmtest,  quietly = TRUE)
library(foreign,  quietly = TRUE)
library(lattice,  quietly = TRUE)
library(lme4,  quietly = TRUE)
library(nlme,  quietly = TRUE)
library(survival,  quietly = TRUE)
library(dplyr,  quietly = TRUE)
library(ggfortify,  quietly = TRUE)
library(survminer,  quietly = TRUE)
library(rms,  quietly = TRUE)
library(MASS, quietly = TRUE)
library(pscl, quietly = TRUE)
```

A generalized linear model (GLM) has three components:

- a random component with mean μ. Generally, the random component is the response variable Y_i.
- a systematic component, η_i, that relates the explanatory variables,
$$\eta_i = \sum_{j=i}^{n} \beta_j x_{ij}$$
- a link function that relates the mean of the random to the systematic component
$$g(\mu) = \eta_i$$

Logistic regression

Logistic regression is a GLM used to model binary (0 or 1) data. The response variable must be binary and is assumed to follow a Bernoulli distribution.

That said, logistic regression has the following components:

- a response binary variable, Y_i, that follows a Bernoulli distribution with mean π_i.
- a systematic component, η_i, that relates the explanatory variables,

$$\eta_i = \sum_{j=1}^{n} \beta_j x_{ij}$$

- a link function that relates the mean of the random to the systematic component

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \sum_{j=i}^{n} \beta_j x_{ij}.$$

$\log\left(\pi_i/(1-\pi_i)\right)$ is known as the log odds.

Logistic regression

Data

Using the iris data, we create binary data. We add the column Sepal.Width_binary to iris. If the Sepal.Width is greater than the median then the associated value in Sepal.Width_binary is 1. Otherwise, Sepal.Width_binary is 0.

```
data <- iris
data$Sepal.Width_binary <- ifelse(data$Sepal.Width
                                  >= median(data$Sepal.Widt
                                  1, 0)
```

Logistic regression

Logistic Regression with only the constant term

Fitting only a constant term, the systematic component is

$$\eta_i = \beta_0.$$

```r
logit <- glm(Sepal.Width_binary ~ 1, data = data, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = Sepal.Width_binary ~ 1, family = "binomial", data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3911  -1.3911   0.9778   0.9778   0.9778
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.4895     0.1682    2.91  0.00361 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 199.22  on 149  degrees of freedom
## Residual deviance: 199.22  on 149  degrees of freedom
## AIC: 201.22
##
```

Logistic regression

Logistic Regression with only the constant term

```r
p_avg <- mean(data$Sepal.Width_binary)
log_odds_avg <- log(p_avg/(1-p_avg))
print(log_odds_avg)
```

```
## [1] 0.4895482
```

Logistic regression

Logistic Regression with Species

Fitting the species term, the systematic component is

$$\eta_i = 1 + \beta_2 X_{1i} + \beta_3 X_{2i}.$$

where

$$X_{1i} = \begin{cases} 1 & \text{if } i\text{th data point is versicolor} \\ 0 & \text{otherwise} \end{cases},$$

$$X_{2i} = \begin{cases} 1 & \text{if } i\text{th data point is virginica} \\ 0 & \text{otherwise} \end{cases}.$$

Logistic regression

Logistic Regression with Species

```
logit <- glm(Sepal.Width_binary ~ as.factor(Species), data = data, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = Sepal.Width_binary ~ as.factor(Species), family = "binomial",
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5373  -0.8782   0.2857   1.0438   1.5096
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    3.1781     0.7215   4.405 1.06e-05 ***
## as.factor(Species)versicolor  -3.9318     0.7826  -5.024 5.06e-07 ***
## as.factor(Species)virginica   -2.8553     0.7763  -3.678 0.000235 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 199.22  on 149  degrees of freedom
## Residual deviance: 147.51  on 147  degrees of freedom
## AIC: 153.51
##
## Number of Fisher Scoring iterations: 5
```

Logistic regression

Logistic Regression with Species

Let's compare the results to the average log odds of each Species group

```
log_odds_avg_fun <- function(data){
  p_avg <- mean(data)
  log_odds_avg <- log(p_avg/(1-p_avg))
  return(log_odds_avg)
}

tapply(data$Sepal.Width_binary,
       data$Species, log_odds_avg_fun)
```

```
##     setosa versicolor  virginica
##  3.1780538 -0.7537718  0.3227734
```

The intercept corresponds to the average log odds of setosa as we would expect. However, the other coefficients do not correspond to the average log odds of the other species. Why?

Logistic regression

Logistic Regression with Species

From the formula, $\eta_i = 1 + \beta_2 X_{2i} + \beta_3 X_{3i}$, the log odds of versicolor actually corresponds to $1 + \beta_2$. The log odds of versicolor actually corresponds to $1 + \beta_3$.

```
coefficients<-unname(coef(logit))
print(c(coefficients[1],coefficients[1]+coefficients[2],
        coefficients[1]+coefficients[3]))
```

```
## [1]  3.1780537 -0.7537718  0.3227734
```

Logistic regression

Logistic Regression with continuous variable, Sepal.Length

Fitting the species term, the systematic component is

$$\eta_i = \beta_3 X_{1i}.$$

where X_{1i} = Sepal.Length of the ith data point.

```r
logit <- glm(Sepal.Width_binary ~ Sepal.Length,
             data = data, family = "binomial")
summary(logit)
```

Logistic regression

Logistic Regression with continuous variable, Sepal.Length

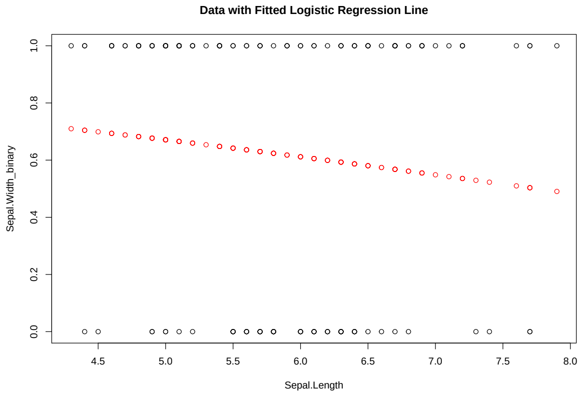
```
logit <- glm(Sepal.Width_binary ~ Sepal.Length,
             data = data, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = Sepal.Width_binary ~ Sepal.Length, family = "binomial",
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5614  -1.3524   0.8883   0.9890   1.1936
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.0088     1.2176   1.650    0.099 .
## Sepal.Length  -0.2591     0.2050  -1.264    0.206
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 199.22  on 149  degrees of freedom
## Residual deviance: 197.61  on 148  degrees of freedom
## AIC: 201.61
##
## Number of Fisher Scoring iterations: 4
```

Logistic regression

Logistic Regression with continuous variable, Sepal.Length

```r
plot(Sepal.Width_binary~Sepal.Length, data=data)
points(data$Sepal.Length[order(data$Sepal.Length)],
       logit$fitted[order(data$Sepal.Length)],  col="red")
title(main="Data with Fitted Logistic Regression Line")
```



Logistic regression

Logistic Regression with Species and Sepal.Length

Fitting the species term, the systematic component is

$$\eta_i = 1 + \beta_2 X_{1i} + \beta_3 X_{2i} + \beta_3 X_{3i}.$$

where

$$X_{1i} = \begin{cases} 1 & \text{if } i\text{th data point is versicolor} \\ 0 & \text{otherwise} \end{cases},$$

$$X_{2i} = \begin{cases} 1 & \text{if } i\text{th data point is virginica} \\ 0 & \text{otherwise} \end{cases}$$

and $X_{3i} = \text{Sepal.Length of the } i\text{th data point.}$

Logistic regression

Logistic Regression with continuous variable, Sepal.Length

Fitting the logistic model accordingly,

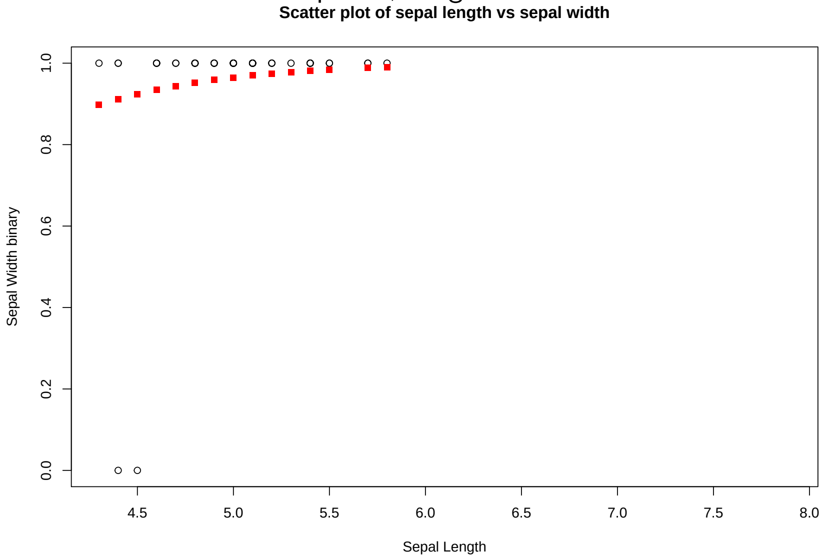
```
logit <- glm(Sepal.Width_binary ~ Species +Sepal.Length,
             data = data, family = "binomial")
summary(logit)
```

```
##
## Call:
## glm(formula = Sepal.Width_binary ~ Species + Sepal.Length, family = "binomial",
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2710  -0.7538   0.2472   0.7020   1.9477
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -4.7988     2.2981  -2.088 0.036784 *
## Speciesversicolor  -5.6936     0.9686  -5.878 4.16e-09 ***
## Speciesvirginica   -5.4812     1.0879  -5.039 4.69e-07 ***
## Sepal.Length        1.6219     0.4510   3.596 0.000323 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 199.22  on 149  degrees of freedom
## Residual deviance: 131.27  on 146  degrees of freedom
## AIC: 139.27
##
```

Logistic regression

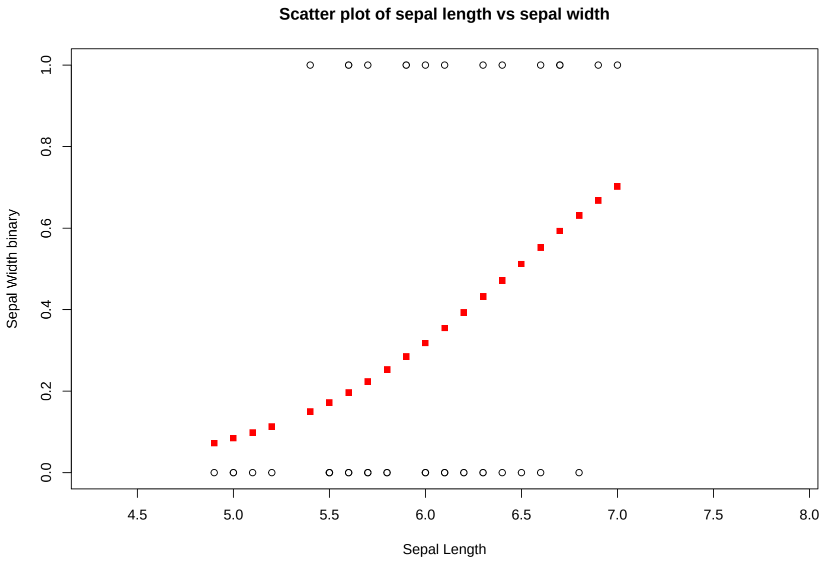
Logistic Regression with continuous variable, Sepal.Length

Plot the results for each species, we get that



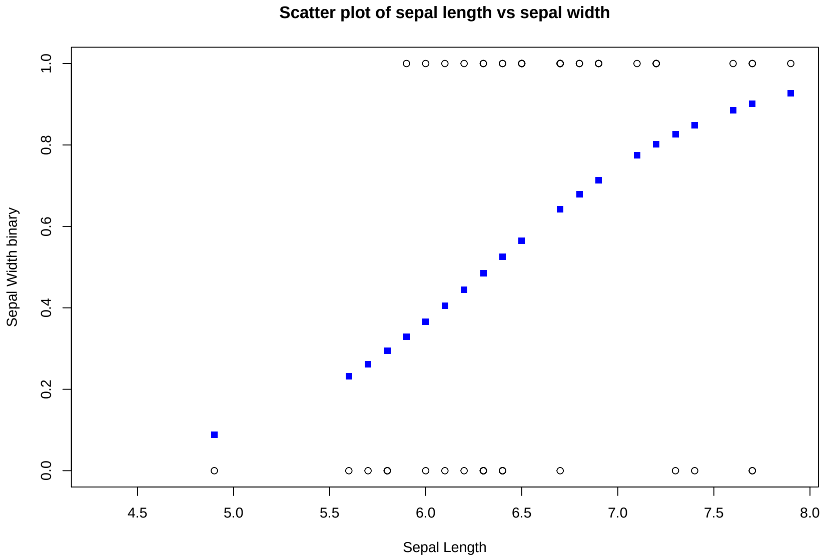
Logistic regression

Logistic Regression with continuous variable, Sepal.Length



Logistic regression

Logistic Regression with continuous variable, Sepal.Length



Logistic regression

Goodness of Fit

Deviance

For general linear models, we use *deviance* to compare model pairs. Deviance is the difference in log likelihood of the models multiples by 2.

For a linear model, *deviance* is equivalent the sum of squares error of the residual.

Logistic regression

Goodness of Fit

Saturated Model

Let's consider the model in which each data point has its own mean and coefficients. This is called the saturated model. It basically replicates the data at hand.

Using deviance, we can compare our fitted model to a saturated model.

If the fitted model behaves similar to the saturated model, then the deviance can be well approximated by a chi-squared distribution with $m - n$ degrees of freedom. m is the number of the data points and n is the number of coefficients in our fitted model.

Logistic regression

Goodness of Fit

Saturated Model

This statistical property of the deviance allows us perform a hypothesis test

$$H_0 : \text{ the fitted model is equivalent to the saturated model}$$

$$H_\alpha : \text{the fitted model is not equivalent to the saturated model}$$

Logistic regression

Goodness of Fit

Saturated Model

`logit$deviance` is the deviance between the saturated model and fitted model.

`logit$df.residual` is equal to the number of observations minus the number of coefficients in the fitted model.

Using this, we can calculate the p value for the hypothesis test above.

```
p_value = pchisq(logit$deviance,
                 logit$df.residual, lower.tail = F)
print(p_value)
```

```
## [1] 0.8032738
```

Since the p value is greater than 0.05, we fail to reject the null hypothesis. (This is a good thing.)

Logistic regression

Goodness of Fit

Null Model

We can also use deviance to determine if our fitted model is better than the null model. The null model is a model with only a linear term.

Like above, we can design a hypothesis test comparing the null model to the fitted model.

Logistic regression

Goodness of Fit

Null Model

$$H_0 = \text{ the fitted model is equivalent to the null model}$$
$$H_\alpha = \text{ the fitted model is not equivalent to the null model}$$

In the limit of large data, it is known that the deviance follows a chi-squared distribution with parameter $n - 1$.

Logistic regression

Goodness of Fit

Null Model

`logit$deviance` is the deviance between the saturated model and fitted model. `logit$df.residual` is equal to the number of observations minus the number of coefficients in the fitted model.

`logit$null.deviance` is the deviance between the saturated model and the null model. `logit$df.null` is the number of observations minus 1.

Logistic regression

Goodness of Fit

Null Model

Using this information, we can calculate the p value for the hypothesis test above.

```
p_value = pchisq(logit$null.deviance-logit$deviance,
                 logit$df.null-logit$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 1.173879e-14
```

Since the p value is less than one, we reject our null hypothesis. (This is a good thing.)

Logistic regression

Goodness of Fit

Anova

`anova` with argument `test="Chisq"` allows us to compare change in deviance after sequentially adding terms our model.

```
anova(logit,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Sepal.Width_binary
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                           149     199.22
## Species       2   51.709       147     147.51 5.910e-12 ***
## Sepal.Length  1   16.239       146     131.27 5.583e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Poisson Generalized Linear Model

A Poisson GLM is used to study *count* data (i.e. discrete numbers, $0, 1, 2, \cdots$). *Count* data describes the number of events that occur within a given time frame.

Poisson Generalized Linear Model

A Poisson GLM is most useful when studying data in which the mean and variance are approximately equal. If they are not not equal, the standard error of the model terms must adjusted to account for the assumption violation.

Poisson Generalized Linear Model

Poisson Regression has the following components:

- response count variables, Y_i, that follows a Poisson distribution with mean μ_i
- a systematic component, η_i, that relates the explanatory variables, $\eta_i = \sum_{j=1}^{n} \beta_j x_{ij}$
- a link function, $log(\mu_i) = \sum_{j=1}^{n} \beta_j x_{ij}$

From Poisson regression, we learn the *mean* of each Y_i given the associated the explanatory variables.

Poisson Generalized Linear Model

Data

We will be consider the `bioChemists` data set in this section.

This data set contains number of articles produced by PhD biochemistry student during the last 3 years of their PhD.

```
attach(bioChemists)
summary(bioChemists)
```

```
##       art            fem           mar            kid5       
##  Min.   : 0.000   Men  :494   Single :309   Min.   :0.0000  
##  1st Qu.: 0.000   Women:421   Married:606   1st Qu.:0.0000  
##  Median : 1.000                             Median :0.0000  
##  Mean   : 1.693                             Mean   :0.4951  
##  3rd Qu.: 2.000                             3rd Qu.:1.0000  
##  Max.   :19.000                             Max.   :3.0000  
##       phd             ment       
##  Min.   :0.755   Min.   : 0.000  
##  1st Qu.:2.260   1st Qu.: 3.000  
##  Median :3.150   Median : 6.000  
##  Mean   :3.103   Mean   : 8.767  
##  3rd Qu.:3.920   3rd Qu.:12.000  
##  Max.   :4.620   Max.   :77.000  
```

Poisson Generalized Linear Model

Data

The data set also contains demographic data associated with each student. The data set has five variables:

- *art* - number of articles produced by the student in the last 3 years of their PhD
- *fem* - gender
- *mar* - martial status
- *kid5* - number of children less than 5
- *phd* - prestige of PhD program
- *ment* - number of articles of the mentor in the last 3 years

Poisson Generalized Linear Model

Data

```
sapply(bioChemists, class)
```

```
##       art       fem       mar      kid5       phd      ment
## "integer"  "factor"  "factor" "numeric" "numeric" "integer"
```

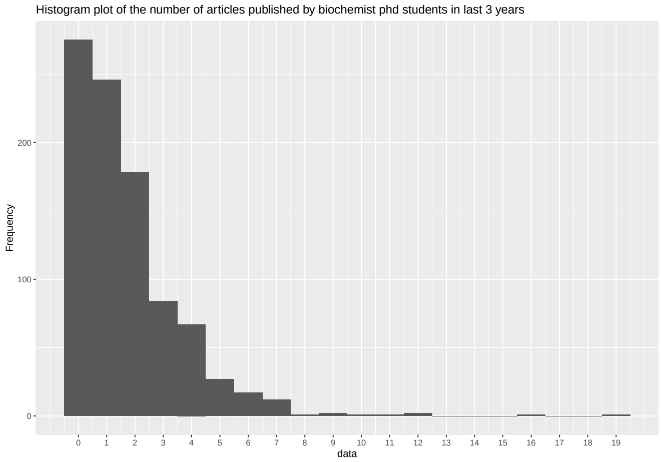
I convert bioChemists$kid5 from numeric to factor. This will be used later.

```
bioChemists$kid5 <- factor(bioChemists$kid5,
                            levels= unique(bioChemists$kid5),
                            labels= unique(bioChemists$kid5))
```

Poisson Generalized Linear Model

Data Visualization

Plotting the bar graph of `bioChemists$art`, we can see that the data looks Poisson-like since there is a large number of observations at 0.



Poisson Generalized Linear Model

Data

We can “quantify” the Poisson-ness by analyzing the mean and variance of the data.

```
mean(bioChemists$art)
```

```
## [1] 1.692896
```

```
var(bioChemists$art)
```

```
## [1] 3.709742
```

Although mean and variance are not equal, we will still fit it to a Poisson model.

Poisson Generalized Linear Model

Possion Regression with constant term

To model only the constant term, I use the formula `art ~ 1`. This formula is equivalent to

$$\log \mu_i = \beta_0.$$

Poisson Generalized Linear Model

Possion Regression with constant term

```
poisson_model = glm(art ~ 1, family=poisson(link=log),data=bioChemists)
summary(poisson_model)
```

```
##
## Call:
## glm(formula = art ~ 1, family = poisson(link = log), data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8401  -1.8401  -0.5770   0.2294   7.5677
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.52644    0.02541   20.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1817.4  on 914  degrees of freedom
## AIC: 3487.1
##
## Number of Fisher Scoring iterations: 5
```

Poisson Generalized Linear Model

Possion Regression with constant term

Note that the constant term is the log mean number of counts.

```r
print(coef(poisson_model))
```

```
## (Intercept)
##   0.5264408
```

```r
print(log(mean(bioChemists$art)))
```

```
## [1] 0.5264408
```

Goodness of fit

Saturated model

We can again compare the current model to the saturated model (best possible fit).

```r
p_value = pchisq(poisson_model$deviance,
                 poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 3.304511e-62
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Goodness of fit

Null model

We can also compare the current model to the null model (worst possible fit).

```
p_value = pchisq(poisson_model$null.deviance
                  -poisson_model$deviance,
                 poisson_model$df.null
                 -poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

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

We fail to reject the null hypothesis. This makes sense since the models are literally the same thing.

Possion Regression with martial status covariate

To model the martial status covariate, I use the formula `art ~ 1+mar`. This formula is equivalent to

$$\log \mu_i = \beta_0 + \beta_1 X_{1i}$$

where

$$X_{1i} = \begin{cases} 1 & \text{if mar = Married} \\ 0 & \text{otherwise} \end{cases} .$$

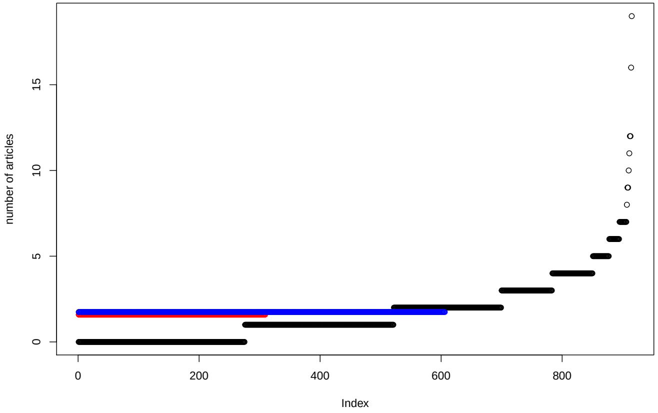
Possion Regression with martial status covariate

```r
poisson_model = glm(art~1+mar , family=poisson(link=log),data=bioChemists)
summary(poisson_model)
```

```
##
## Call:
## glm(formula = art ~ 1 + mar, family = poisson(link = log), data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8677  -1.7845  -0.5042   0.3107   7.4992
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.46514    0.04508  10.317   <2e-16 ***
## marMarried   0.09117    0.05458   1.671   0.0948 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1814.6  on 913  degrees of freedom
## AIC: 3486.3
##
## Number of Fisher Scoring iterations: 5
```

Possion Regression with martial status covariate

```
plot(bioChemists$art,ylab='number of articles',xlab = 'Index')
points(poisson_model$fitted[bioChemists$mar=='Single'],col="red")
points(poisson_model$fitted[bioChemists$mar=='Married'],col="blue")
```



Graphically, we can see than that martial status is not good indicator of number articles published.

Goodness of fit

Saturated model

We can again compare the current model to the saturated model (best possible fit).

```r
p_value = pchisq(poisson_model$deviance,
                 poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 4.731233e-62
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent and our model is a bad fit.

Goodness of fit

Null model

We can also compare the current model to the null model (worst possible fit).

```
p_value = pchisq(poisson_model$null.deviance-logit$deviance,
                 poisson_model$df.null-logit$df.residual, lower.tail = F)
print(p_value)
```

```
## [1] 1.016236e-70
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent. Though our current model does not capture much deviance, the current model captures much more variance than the null model.

Anova

```r
anova(poisson_model,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: art
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                   914     1817.4
## mar   1   2.8211       913     1814.6  0.09304 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Possion Regression with martial status and children covariate

To model the martial status and children as covariates, I use the formula `art ~ 1+mar + kid5`. This formula is equivalent to

$$\log \mu_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{3i} + \beta_4 X_{4i}$$

where

$$X_{1i} = \begin{cases} 1 & \text{if the ith data point is married} \\ 0 & \text{otherwise} \end{cases},$$

$$X_{2i} = \begin{cases} 1 & \text{if the number of children of ith data point is 1} \\ 0 & \text{otherwise} \end{cases},$$

$$X_{3i} = \begin{cases} 1 & \text{if the number of children of ith data point is 2} \\ 0 & \text{otherwise} \end{cases}$$

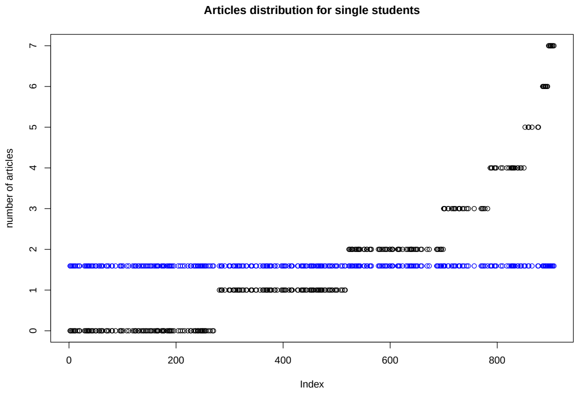
and $X_{4i} = \begin{cases} 1 & \text{if the number of children of ith data point is 3} \\ 0 & \text{otherwise} \end{cases}.$

Possion Regression with martial status and children covariate

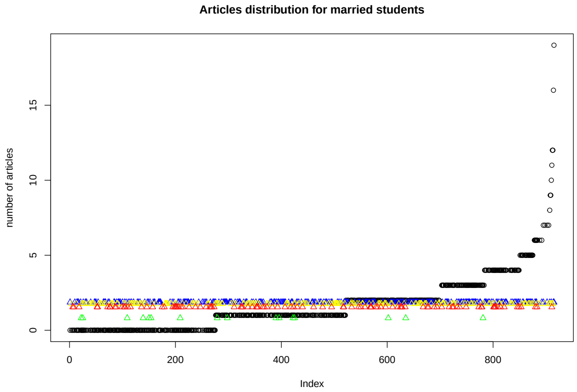
```r
poisson_model = glm(art ~ 1 + kid5 + mar,
                    family=poisson(link=log),data=bioChemists)
summary(poisson_model)
```

```
##
## Call:
## glm(formula = art ~ 1 + kid5 + mar, family = poisson(link = log),
##     data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9280  -1.7845  -0.5042   0.3518   7.3520
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.46514    0.04508  10.317   <2e-16 ***
## kid51       -0.05510    0.06907  -0.798   0.4250
## kid52       -0.18620    0.08960  -2.078   0.0377 *
## kid53       -0.82747    0.28067  -2.948   0.0032 **
## marMarried   0.15470    0.06235   2.481   0.0131 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1799.9  on 910  degrees of freedom
## AIC: 3477.7
##
## Number of Fisher Scoring iterations: 5
```

Possion Regression with martial status and children covariate



Possion Regression with martial status and children covariate



Graphically, we can see than that martial status and number of children is not good indicator of number articles published.

Goodness of Fit

Saturated model

We can again compare the current model to the saturated model (best possible fit).

```r
p_value = pchisq(poisson_model$deviance,
                 poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 6.462874e-61
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Goodness of fit

Null model

We can also compare the current model to the null model (worst possible fit).

```r
p_value = pchisq(poisson_model$null.deviance
                  -poisson_model$deviance,
                 poisson_model$df.null
                 -poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 0.001567133
```

Goodness of fit

Anova

We can also determine the model terms that cause a significance reduction in deviance.

```r
anova(poisson_model,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: art
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                   914     1817.4
## kid5  3  11.3045       911     1806.1  0.01019 *
## mar   1   6.1638       910     1799.9  0.01304 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Possion Regression with continuous variables, mentor articles and martial status

To model the martial status and number of mentor articles as covariates, I use the formula `art ~ 1+mar + ment`. This formula is equivalent to

$$\log \mu_i = \beta_0 + \beta_1 X_{1i} + X_{2i}$$

where

$$X_{1i} = \begin{cases} 1 & \text{if the i data point is Married} \\ 0 & \text{otherwise} \end{cases}$$

and X_{2i} is the number of publications of the ith data point's mentor.

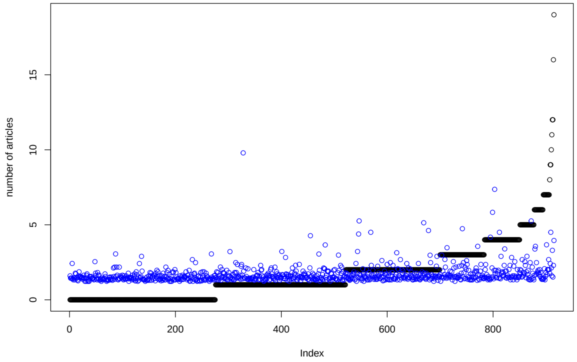
Possion Regression with continuous variables, mentor articles and martial status

```
poisson_model = glm(art ~ 1 + ment +mar,
                    family=poisson(link=log),data=bioChemists)
summary(poisson_model)
```

```
##
## Call:
## glm(formula = art ~ 1 + ment + mar, family = poisson(link = log),
##     data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6086  -1.6317  -0.3608   0.5039   5.8942
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 0.210726   0.049847   4.227 2.36e-05 ***
## ment        0.025917   0.001915  13.530  < 2e-16 ***
## marMarried  0.075332   0.054643   1.379    0.168
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1667.6  on 912  degrees of freedom
## AIC: 3341.4
##
## Number of Fisher Scoring iterations: 5
```

Possion Regression with continuous variables, mentor articles and martial status

```
plot(bioChemists$art,ylab='number of articles',xlab = 'Index')

points(poisson_model$fitted,col="blue",pch=1)
```



Graphically, we can see than that martial status and number of children is not good indicator of number articles published.

Goodness of Fit

Saturated model

We can again compare the current model to the saturated model (best possible fit).

```
p_value = pchisq(poisson_model$deviance,
                 poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 6.132629e-47
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Goodness of fit

Null model

We can also compare the current model to the null model (worst possible fit).

```r
p_value = pchisq(poisson_model$null.deviance
                  -poisson_model$deviance,
                 poisson_model$df.null
                 -poisson_model$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 2.993003e-33
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Goodness of fit

Anova

We can also determine the model terms that cause a significance reduction in deviance.

```
anova(poisson_model,test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: art
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                   914     1817.4
## ment  1  147.860       913     1669.5   <2e-16 ***
## mar   1    1.918       912     1667.6   0.1661
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Quasi-Poisson Regression with continuous variables, mentor articles and martial status

If the process that generated our data does not have mean and variable equal to each other, we can assume that the variance is approximately the mean multiplied by some constant. This constant is called the dispersion parameter.

Quasi-Poisson Regression with continuous variables, mentor articles and martial status

If the process that generated our data does not have mean and variable equal to each other, we can assume that the variance is approximately the mean multiplied by some constant. This constant is called the dispersion parameter.

Fitting the data with this scaled relationship between the mean and the variance does not change the coefficient estimates. It however adjusts the standard errors and the p values.

Quasi-Poisson Regression with continuous variables, mentor articles and martial status

If the process that generated our data does not have mean and variable equal to each other, we can assume that the variance is approximately the mean multiplied by some constant. This constant is called the dispersion parameter.

Fitting the data with this scaled relationship between the mean and the variance does not change the coefficient estimates. It however adjusts the standard errors and the p values.

The Poisson model with the assumption that the variance is equal to the mean multiplied by some constant is called ‘Quasi-Poisson‘.

Quasi-Poisson Regression with continuous variables, mentor articles and martial status

If the process that generated our data does not have mean and variable equal to each other, we can assume that the variance is approximately the mean multiplied by some constant. This constant is called the dispersion parameter.

Fitting the data with this scaled relationship between the mean and the variance does not change the coefficient estimates. It however adjusts the standard errors and the p values.

The Poisson model with the assumption that the variance is equal to the mean multiplied by some constant is called ‘Quasi-Poisson‘.

To fit our data to a ‘Quasi-Poisson‘ model, set ‘family = quasipoisson(link=log)‘

Quasi-Poisson Regression with continuous variables, mentor articles and martial status

```r
poisson_model = glm(art ~ 1 + ment +mar,
                    family=quasipoisson(link=log),data=bioChemists)
summary(poisson_model)
```

```
##
## Call:
## glm(formula = art ~ 1 + ment + mar, family = quasipoisson(link = log),
##     data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6086  -1.6317  -0.3608   0.5039   5.8942
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.210726   0.068367   3.082  0.00212 **
## ment        0.025917   0.002627   9.865  < 2e-16 ***
## marMarried  0.075332   0.074946   1.005  0.31509
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 1.881168)
##
##     Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1667.6  on 912  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Log-Linear Regression

Contingency table displays the number of observations for a given combination of factors.

	low mentor articles	high mentor articles
low articles published	321	200
high articles published	171	223

Log-Linear Regression

Contingency table displays the number of observations for a given combination of factors.

	low mentor articles	high mentor articles
low articles published	321	200
high articles published	171	223

Log-linear models estimate the group mean count of each cell of the contingency table

Log-Linear Regression

Contingency table displays the number of observations for a given combination of factors.

	low mentor articles	high mentor articles
low articles published	321	200
high articles published	171	223

Log-linear models estimate the group mean count of each cell of the contingency table

Log-linear models allow us to model association between two or more variables.

Log-Linear Regression

Contingency table displays the number of observations for a given combination of factors.

	low mentor articles	high mentor articles
low articles published	321	200
high articles published	171	223

Log-linear models estimate the group mean count of each cell of the contingency table

Log-linear models allow us to model association between two or more variables.

There are no well defined explanatory/response variables in the contingency table. This allows us to focus on the interaction between variables.

Log-Linear Regression

Contingency Table

Contingency table displays number of observations for all combinations of levels of given discrete variables.

We would like to learn a contingency table for the `biochemist` data set using number of articles published, number of mentor articles published and number of children.

First, we discretize our continuous variables.

```r
bioChemists$art_binary <- sapply(bioChemists$art,
                                 function(x) ifelse(x > 1, 1, 0))
bioChemists$ment_binary <- sapply(bioChemists$ment,
                                  function(x)
                                    ifelse(x > median(bioChemists$ment),
                                           1, 0))
```

Log-Linear Regression

Contingency Table

One-Way Contingency Table

A one-way contingency table shows the counts according to one covariate.

```
table(art_relative=bioChemists$art_binary)
```

```
## art_relative
##   0   1
## 521 394
```

This one-way contingency table shows that:

- there are 521 biochemists with 1 or less papers
- there are 394 biochemists with greater than 1 papers.

Log-Linear Regression

Contingency Table

Two-Way Contingency Table

A two-way contingency table shows the counts according to two covariates.

```
table(art_relative=bioChemists$art_binary,ment=bioChemists$ment_binary)
```

```
##             ment
## art_relative   0   1
##            0 321 200
##            1 171 223
```

This two-way contingency table shows that:

- there are 321 biochemists with 1 or less papers and with a mentor that produced less than or equal to 6 papers
- there are 200 biochemists with 1 or less papers and with a mentor that produced more than 6 papers

Log-Linear Regression

Contingency Table

Two-Way Contingency Table

A two-way contingency table shows the counts according to two covariates.

```
table(art_relative=bioChemists$art_binary,ment=bioChemists$ment_binary)
```

```
##             ment
## art_relative   0   1
##            0 321 200
##            1 171 223
```

- there are 171 biochemists with more than 1 paper and with a mentor that produced less than or equal to 6 papers
- there are 200 biochemists with more than 1 paper and with a mentor that produced more than 6 papers

Log-Linear Regression

Contingency Table

Three-Way Contingency Table

A three-way contingency table shows the counts according to three covariates.

```r
table(art_relative=bioChemists$art_binary,ment=bioChemists$ment_binary,
      kid5=bioChemists$kid5)
```

```
## , , kid5 = 0
##
##             ment
## art_relative   0   1
##            0 208 128
##            1 116 147
##
## , , kid5 = 1
##
##             ment
## art_relative   0   1
##            0  66  46
##            1  38  45
##
## , , kid5 = 2
##
##             ment
## art_relative   0   1
##            0  39  21
```

Log-Linear Regression

Contingency Table

Three-Way Contingency Table

This three-way contingency table shows that:

- With no children,
  - there are 208 biochemists with 1 or less papers and with a mentor that produced less than or equal to 6 papers
  - there are 128 biochemists with 1 or less papers and with a mentor that produced more than 6 papers
  - there are 116 biochemists with more than 1 paper and with a mentor that produced less than or equal to 6 papers
  - there are 147 biochemists with more than 1 paper and with a mentor that produced more than 6 papers

Independent Model for two-way contigency table

Remember using a log-linear model, our primary goal is to learn the interaction effects between covariates.

Again, we build the same two-way contingency table. We need to convert the contingency table in a form that is acceptable to `glm`.

```
contigency_table = table(art_relative=bioChemists$art_binar
                         ment=bioChemists$ment_binary)
contigency_table.df = as.data.frame(contigency_table)
```

Independent Model for two-way contigency table

```
print(contigency_table.df)
```

```
##   art_relative ment Freq
## 1            0    0  321
## 2            1    0  171
## 3            0    1  200
## 4            1    1  223
```

Independent Model for two-way contigency table

Assuming each number of articles and mentor do not affect each other, we build a model of the cell count that does not take into account interaction effects. Such a model is called the *independent* model.

To do this, we use formula `Freq ~ art_relative + ment`.

```r
log_linear_model_int <- glm(Freq ~ art_relative + ment,
             data = contigency_table.df, family = poisson)
```

Independent Model for two-way contigency table

```
summary(log_linear_model_int)
```

```
##
## Call:
## glm(formula = Freq ~ art_relative + ment, family = poisson, data = contigency_table.df)
##
## Deviance Residuals:
##      1       2       3       4
##  2.385  -2.905  -2.713   2.923
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    5.63530    0.05347 105.392  < 2e-16 ***
## art_relative1 -0.27940    0.06676  -4.185 2.85e-05 ***
## ment1         -0.15111    0.06631  -2.279   0.0227 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 52.927  on 3  degrees of freedom
## Residual deviance: 30.035  on 1  degrees of freedom
## AIC: 65.008
##
## Number of Fisher Scoring iterations: 4
```

Independent Model for two-way contigency table

```
log_linear_model_int$coefficients
```

```
##   (Intercept) art_relative1          ment1
##     5.6353047    -0.2793991    -0.1511065
```

	mentor article : 0	mentor article : 1
student article : 0	$\exp(5.64)$	$\exp(5.64 - 0.15)$
mentor article : 1	$\exp(5.64 - 0.26 - 0.15)$	$\exp(5.63 - 0.27 - 0.15)$

Independent Model for two-way contigency table

Goodness of fit

We compare the current model to the saturated model (best possible fit).

```r
p_value = pchisq(log_linear_model_int$deviance,
                 log_linear_model_int$df.residual,
                 lower.tail = F)
print(p_value)
```

```
## [1] 4.243721e-08
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Independent Model for two-way contigency table

Saturated Model for the two-way contingency table

Assuming each number of articles and mentor affect each other, we build a model of the cell count that takes into account all interaction effects. Such a model is called the *saturated* model. To do this, we use formula `Freq ~ art_relative*ment`.

```
log_linear_model_sat <- glm(Freq ~ art_relative*ment,
            data = contigency_table.df,
            family = poisson)
```

Independent Model for two-way contigency table

Saturated Model for the two-way contingency table

```
summary(log_linear_model_sat)
```

```
##
## Call:
## glm(formula = Freq ~ art_relative * ment, family = poisson, data = contigency_table.df)
##
## Deviance Residuals:
## [1]  0  0  0  0
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          5.77144    0.05581 103.404  < 2e-16 ***
## art_relative1       -0.62978    0.09467  -6.652 2.89e-11 ***
## ment1               -0.47312    0.09008  -5.252 1.50e-07 ***
## art_relative1:ment1  0.73863    0.13582   5.438 5.38e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 5.2927e+01  on 3  degrees of freedom
## Residual deviance: 1.8874e-14  on 0  degrees of freedom
## AIC: 36.973
##
## Number of Fisher Scoring iterations: 2
```

Independent Model for two-way contigency table

Goodness of fit

We compare the current model to the saturated model (best possible fit).

```
p_value = pchisq(0,
                 log_linear_model_sat$df.residual,
                 lower.tail = F)
print(p_value)
```

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

We fail to reject the null hypothesis. This makes sense since the models are literally the same thing.

Independent Model for two-way contigency table

Model Comparison

We use `anova` with `test='Chisq'` to compare the independent and saturated model.

```
anova(log_linear_model_int,log_linear_model_sat,
      test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: Freq ~ art_relative + ment
## Model 2: Freq ~ art_relative * ment
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1         1     30.035
## 2         0      0.000  1   30.035 4.244e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From `anova`, we can see that the saturated model provides a statistically significant result.

Independent Model for two-way contigency table

Anova

We use also `anova` to determine what caused the significant decrease in the deviance.

```
anova(log_linear_model_sat,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Freq
##
## Terms added sequentially (first to last)
##
##
##                   Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                                  3     52.927
## art_relative       1  17.6844         2     35.243 2.608e-05 ***
## ment               1   5.2082         1     30.035   0.02248 *
## art_relative:ment  1  30.0348         0      0.000 4.244e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Adding `art_relative:ment` to the independent model caused significant decrease in deviance.

Independent Model for the three-way contingency table

Again, we build the same three-way contingency table. We need to convert the contingency table in a form that is acceptable to `glm`.

To create the *independent* model for the three-way contingency table, we use formula `Freq ~  art_relative + ment + kid5`.

```
log_linear_model_int <- glm(Freq ~ art_relative + ment + kid5,
            data = contigency_table.df, family = poisson)
```

Independent Model for the three-way contingency table

```
summary(log_linear_model_int)
```

```
##
## Call:
## glm(formula = Freq ~ art_relative + ment + kid5, family = poisson,
##     data = contigency_table.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4439  -1.4070  -0.1702   1.1974   2.4521
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    5.21164    0.05861  88.915  < 2e-16 ***
## art_relative1 -0.27940    0.06676  -4.185 2.85e-05 ***
## ment1         -0.15111    0.06631  -2.279   0.0227 *
## kid51         -1.12226    0.08245 -13.612  < 2e-16 ***
## kid52         -1.74130    0.10580 -16.459  < 2e-16 ***
## kid53         -3.62267    0.25331 -14.301  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 901.879  on 15  degrees of freedom
## Residual deviance:  36.651  on 10  degrees of freedom
## AIC: 131.03
##
## Number of Fisher Scoring iterations: 4
```

Goodness of fit

We compare the current model to the saturated model (best possible fit).

```r
p_value = pchisq(log_linear_model_int$deviance,
                 log_linear_model_int$df.residual, lower.tail = F)
print(p_value)
```

```
## [1] 6.50121e-05
```

Since our p value is less than 0.05, we reject the null hypothesis. The models are not equivalent.

Saturated Model

To create the *saturated* model for the three-way contingency table, we use formula `Freq ~  art_relative*ment*kid5`.

```
log_linear_model_sat <- glm(Freq ~ art_relative*ment*kid5,
            data = contigency_table.df, family = poisson)
```

Saturated Model

```r
summary(log_linear_model_sat)
```

```
##
## Call:
## glm(formula = Freq ~ art_relative * ment * kid5, family = poisson,
##     data = contigency_table.df)
##
## Deviance Residuals:
##  [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)               5.33754    0.06934  76.979  < 2e-16 ***
## art_relative1            -0.58395    0.11588  -5.039 4.67e-07 ***
## ment1                    -0.48551    0.11234  -4.322 1.55e-05 ***
## kid51                    -1.14788    0.14128  -8.125 4.47e-16 ***
## kid52                    -1.67398    0.17450  -9.593  < 2e-16 ***
## kid53                    -3.25810    0.36029  -9.043  < 2e-16 ***
## art_relative1:ment1       0.72235    0.16746   4.314 1.61e-05 ***
## art_relative1:kid51       0.03188    0.23430   0.136    0.892
## art_relative1:kid52      -0.30703    0.31870  -0.963    0.335
## art_relative1:kid53      -1.49549    1.06697  -1.402    0.161
## ment1:kid51               0.12449    0.22251   0.559    0.576
## ment1:kid52              -0.13353    0.29305  -0.456    0.649
## ment1:kid53               0.01550    0.58105   0.027    0.979
## art_relative1:ment1:kid51 -0.19226    0.33686  -0.571    0.568
## art_relative1:ment1:kid52  0.49140    0.44529   1.104    0.270
## art_relative1:ment1:kid53  0.44080    1.36126   0.324    0.746
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
```

Goodness of fit

We compare the current model to the saturated model (best possible fit).

```
p_value = pchisq(log_linear_model_sat$deviance,
                 log_linear_model_sat$df.residual, lower.tail = F)
print(p_value)
```

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

We fail to reject the null hypothesis. This makes sense since the models are literally the same thing.

Model Comparison

We use `anova` with `test='Chisq'` to compare the independent and saturated model.

```r
anova(log_linear_model_int,log_linear_model_sat,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: Freq ~ art_relative + ment + kid5
## Model 2: Freq ~ art_relative * ment * kid5
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1        10     36.651
## 2         0      0.000 10   36.651 6.501e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From `anova`, we can see that the saturated model provides a statistically significant result.

Model Comparison

We use also `anova` to determine what caused the significant decrease in the deviance.

```r
anova(log_linear_model_sat,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Freq
##
## Terms added sequentially (first to last)
##
##
##                        Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                                      15     901.88
## art_relative            1    17.68        14     884.20 2.608e-05 ***
## ment                    1     5.21        13     878.99   0.02248 *
## kid5                    3   842.34        10      36.65 < 2.2e-16 ***
## art_relative:ment       1    30.03         9       6.62 4.244e-08 ***
## art_relative:kid5       3     4.45         6       2.17   0.21665
## ment:kid5               3     0.19         3       1.97   0.97873
## art_relative:ment:kid5  3     1.97         0       0.00   0.57819
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Adding `art_relative:ment` to the independent model caused significant decrease in deviance.

Hierarchical modeling

Data Wrangling

Often you'll have to merge the different levels of data and center the predictors.

```r
student_data <- read.csv("hsb1.csv")
school_data  <- read.csv("hsb2.csv")
student_data$ses_grandmean <- student_data$ses -
  mean(student_data$ses) # Grand-mean centered student SES
school_data$sm_ses_grandmean <- school_data$meanses -
  mean(school_data$meanses) # Grand-mean centered school SE
```

```r
data <- merge(student_data, school_data, by = "id")
```

Hierarchical modeling

Data Wrangling

Above we grand-mean centered student SES and school SES, here we're group-mean centering student SES. Which one you use affects the interpretation of the intercept.

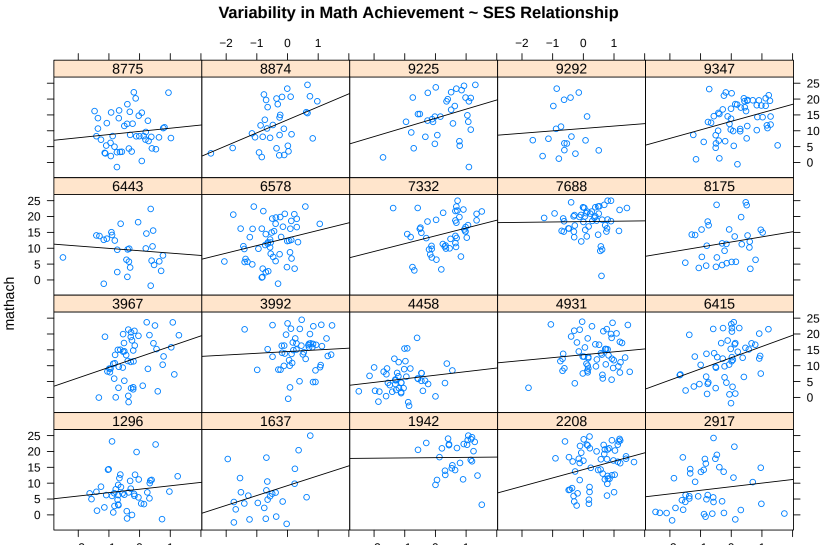
```r
# Group-mean centered student SES
ses_group_mean <- aggregate(data$ses, list(data$id),
                            FUN = mean, data = data)

names(ses_group_mean)<- c('id','groupmeanSES')
data <- merge(data, ses_group_mean, by = "id")
```

```r
groups <- unique(data$id)[sample(1:160,20)]
subset <- data[data$id%in%groups, ]
```

Hierarchical modeling

Hierarchical models are in order!

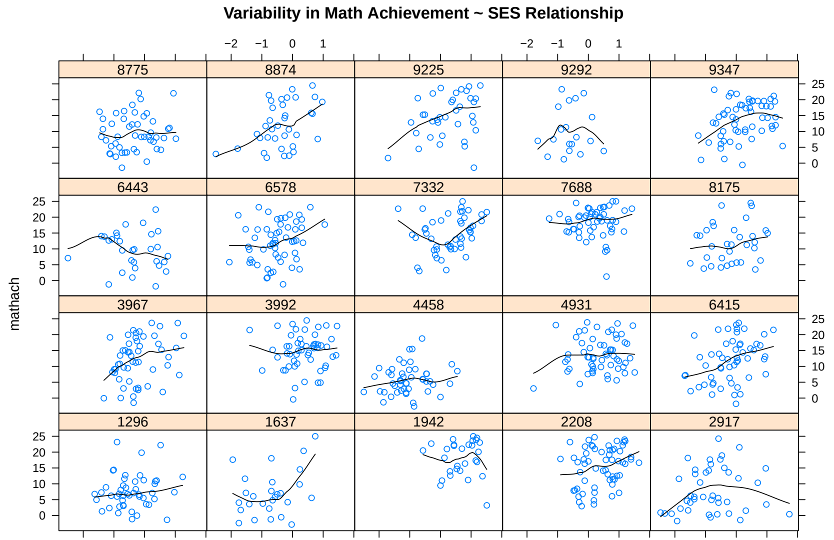
```
xyplot(mathach ~ ses | as.factor(id), subset, col.line = 'black', type = c("p", "r"),
       main = 'Variability in Math Achievement ~ SES Relationship')
```



Hierarchical modeling

Smoothing instead of regression lines.

```
xyplot(mathach ~ ses |as.factor(id), subset, col.line = 'black',type = c("p", "smooth"),
                           main = 'Variability in Math Achievement ~ SES Relationship')
```



Hierarchical modeling

It's good practice to start with an unconditional model to have estimates of the variance.

```
unconditional <- lmer(mathach ~ 1 + (1|id), data = data)
summary(unconditional) # on p-values in lme4: https://stat.ethz.ch/pipermail/r-help/2006-May/094765.html
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + (1 | id)
##    Data: data
##
## REML criterion at convergence: 47116.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.0631 -0.7539  0.0267  0.7606  2.7426
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  id       (Intercept)  8.614   2.935
##  Residual             39.148   6.257
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  12.6370     0.2444   51.71
```

Hierarchical modeling

lme4 doesn't calculate p-values (see link above), but you can calculate approximate confidence intervals.

```
confint(unconditional) # you can also just calculate an app
```

```
## Computing profile confidence intervals ...
```

```
##                 2.5 %    97.5 %
## .sig01       2.594729  3.315880
## .sigma       6.154803  6.361786
## (Intercept) 12.156289 13.117121
```

Hierarchical modeling

We'll estimate the same models with `lmer()` (lme4) and `lme()`nlme.

```r
unconditional_2 <- lme(mathach ~ 1, random = ~ 1 | id, data = data)
summary(unconditional_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC   logLik
##   47122.79 47143.43 -23558.4
##
## Random effects:
##  Formula: ~1 | id
##         (Intercept) Residual
## StdDev:    2.934966 6.256862
##
## Fixed effects: mathach ~ 1
##                Value Std.Error   DF  t-value p-value
## (Intercept) 12.63697 0.2443936 7025 51.70747       0
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.06312473 -0.75387398  0.02670132  0.76062171  2.74262579
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

In `lmer()`, the random effects go in the parentheses. 1 stands, as in the usual formula syntax, for the intercept. The grouping variable goes after |in the parentheses as well.

```
random_intercept_fixed_slope <- lmer(mathach ~ 1 + groupmeanSES + (1|id), data = data)
summary(random_intercept_fixed_slope)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + groupmeanSES + (1 | id)
##    Data: data
##
## REML criterion at convergence: 46961.3
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.13493 -0.75254  0.02413  0.76766  2.78515
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  id       (Intercept)  2.639   1.624
##  Residual             39.157   6.258
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)   12.6846     0.1493   84.97
## groupmeanSES   5.8635     0.3615   16.22
##
## Correlation of Fixed Effects:
##             (Intr)
## groupmenSES 0.010
```

Hierarchical modeling

```r
confint(random_intercept_fixed_slope)
```

```
## Computing profile confidence intervals ...
```

```
##                  2.5 %    97.5 %
## .sig01        1.385193  1.871127
## .sigma        6.155502  6.362511
## (Intercept)  12.391774 12.976903
## groupmeanSES  5.155743  6.572440
```

Hierarchical modeling

In `lme()`, the random effects go after `ramdom =`. Again, the `1` stands for the intercept and `|` goes before the grouping variable.

```
random_intercept_fixed_slope_2 <- lme(mathach ~ 1 + groupmeanSES,
                                      random = ~ 1 | id, data = data)
```

Hierarchical modeling

```r
summary(random_intercept_fixed_slope_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC    logLik
##   46969.29 46996.81 -23480.65
##
## Random effects:
##  Formula: ~1 | id
##         (Intercept) Residual
## StdDev:    1.624462 6.257562
##
## Fixed effects: mathach ~ 1 + groupmeanSES
##                  Value Std.Error   DF  t-value p-value
## (Intercept)  12.684609 0.1492900 7025 84.96624       0
## groupmeanSES  5.863539 0.3614712  158 16.22132       0
##  Correlation:
##              (Intr)
## groupmeanSES 0.01
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.13493066 -0.75254260  0.02413095  0.76766113  2.78515398
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here we added a random slope for `groupmeanSES`

```
random_intercept_random_slope <- lmer(mathach ~ 1 + groupmeanSES + (1 + groupmeanSES|id), data = data)
```

```
## boundary (singular) fit: see ?isSingular
```

```
summary(random_intercept_random_slope)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + groupmeanSES + (1 + groupmeanSES | id)
##    Data: data
##
## REML criterion at convergence: 46960.9
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.13245 -0.75164  0.02212  0.76876  2.79449
##
## Random effects:
##  Groups   Name         Variance Std.Dev. Corr
##  id       (Intercept)   2.62707 1.6208
##           groupmeanSES  0.05417 0.2327   -1.00
##  Residual              39.15798 6.2576
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)   12.6832     0.1491   85.04
## groupmeanSES   5.8379     0.3644   16.02
##
## Correlation of Fixed Effects:
##             (Intr)
```

Hierarchical modeling

Here we added a random slope for `groupmeanSES`

```r
random_intercept_random_slope_2 <- lme(mathach ~ 1 + groupmeanSES,
                                       random = ~ 1 + groupmeanSES | id,
                                       data = data)
```

Hierarchical modeling

```r
summary(random_intercept_random_slope_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC    logLik
##   46973.29 47014.57 -23480.65
##
## Random effects:
##  Formula: ~1 + groupmeanSES | id
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev      Corr
## (Intercept)  1.624460932 (Intr)
## groupmeanSES 0.008272356 -0.003
## Residual     6.257561467
##
## Fixed effects: mathach ~ 1 + groupmeanSES
##                  Value Std.Error   DF  t-value p-value
## (Intercept)  12.684610 0.1492901 7025 84.96616       0
## groupmeanSES  5.863533 0.3614729  158 16.22122       0
##  Correlation:
##              (Intr)
## groupmeanSES 0.01
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.13493049 -0.75254293  0.02413128  0.76766157  2.78515572
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here the 0 stands for no intercept, so we're leaving a fixed intercept.

```r
fixed_intercept_random_slope <- lmer(mathach ~ 1 +
                                       groupmeanSES
                                     + (0 + groupmeanSES|id),
                                     data = data)
```

Hierarchical modeling

```r
summary(fixed_intercept_random_slope)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + groupmeanSES + (0 + groupmeanSES | id)
##    Data: data
##
## REML criterion at convergence: 47065
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1150 -0.7431  0.0317  0.7651  2.8202
##
## Random effects:
##  Groups   Name         Variance Std.Dev.
##  id       groupmeanSES 27.05    5.201
##  Residual              39.75    6.304
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)   12.7640     0.1226  104.07
## groupmeanSES   5.4202     0.5271   10.28
##
## Correlation of Fixed Effects:
##             (Intr)
## groupmenSES -0.045
```

Hierarchical modeling

Again, the 0 stands for no intercept, so we're leaving a fixed intercept.

```
fixed_intercept_random_slope_2 <- lme(mathach ~ 1 + groupmeanSES,
                                      random = ~ 0 + groupmeanSES | id, data = data)
```

Hierarchical modeling

```r
summary(fixed_intercept_random_slope_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC   logLik
##   47072.99 47100.51 -23532.5
##
## Random effects:
##  Formula: ~0 + groupmeanSES | id
##         groupmeanSES Residual
## StdDev:     5.201045 6.304462
##
## Fixed effects: mathach ~ 1 + groupmeanSES
##                  Value Std.Error   DF   t-value p-value
## (Intercept)  12.764014 0.1226493 7025 104.06918       0
## groupmeanSES  5.420157 0.5270957  158  10.28306       0
##  Correlation:
##              (Intr)
## groupmeanSES -0.045
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.11504273 -0.74308714  0.03169931  0.76511017  2.82021818
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here we added another level-2 variable, `sm_ses_grandmean`, which stands for the grand-mean centered school SES. Since it's not in the parentheses, it's added as a fixed effect.

```r
fixed_slope_level_two_variable <- lmer(mathach ~ 1 + groupmeanSES
                                       + sm_ses_grandmean + (1|id),
                                       data = data)
```

Hierarchical modeling

```r
summary(fixed_slope_level_two_variable)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + groupmeanSES + sm_ses_grandmean + (1 | id)
##    Data: data
##
## REML criterion at convergence: 46946.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.13127 -0.75215  0.02439  0.76700  2.78177
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  id       (Intercept)  2.659   1.631
##  Residual             39.157   6.258
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##                   Estimate Std. Error t value
## (Intercept)         11.675      3.299   3.539
## groupmeanSES      -157.361    532.675  -0.295
## sm_ses_grandmean  163.223    532.668   0.306
##
## Correlation of Fixed Effects:
##             (Intr) grpSES
## groupmenSES  0.999
## sm_ss_grndm -0.999 -1.000
```

Hierarchical modeling

Again, we added another level-2 variable, sm_ses_grandmean, which stands for the grand-mean centered school SES. Since it's not after random =, it's added as a fixed effect.

```
fixed_slope_level_two_variable_2 <- lme(mathach ~ 1 + groupmeanSES
                                        + sm_ses_grandmean,
                                        random = ~ 1 | id, data = data)
```

Hierarchical modeling

```r
summary(fixed_slope_level_two_variable_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC     BIC   logLik
##   46956.81 46991.2 -23473.4
##
## Random effects:
##  Formula: ~1 | id
##         (Intercept) Residual
## StdDev:    1.630771 6.257562
##
## Fixed effects: mathach ~ 1 + groupmeanSES + sm_ses_grandmean
##                       Value Std.Error   DF   t-value p-value
## (Intercept)        11.67469    3.2988 7025  3.539111  0.0004
## groupmeanSES     -157.36077  532.6748  157 -0.295416  0.7681
## sm_ses_grandmean  163.22262  532.6683  157  0.306425  0.7597
##  Correlation:
##                  (Intr) grpSES
## groupmeanSES      0.999
## sm_ses_grandmean -0.999 -1.000
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.13126623 -0.75215319  0.02439264  0.76699775  2.78176653
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here we model groupmeanSES, the group-mean centered student SES, as a random effect by putting it inside the parentheses.

```
random_slope_level_two_variable <- lmer(mathach ~ 1 + groupmeanSES
                                        + sm_ses_grandmean
                                        + (1 + groupmeanSES|id),
                                        data = data)
```

```
## boundary (singular) fit: see ?isSingular
```

Hierarchical modeling

```r
summary(random_slope_level_two_variable)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula:
## mathach ~ 1 + groupmeanSES + sm_ses_grandmean + (1 + groupmeanSES |
##     id)
##    Data: data
##
## REML criterion at convergence: 46946.3
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.12727 -0.74930  0.02286  0.76841  2.79122
##
## Random effects:
##  Groups   Name         Variance Std.Dev. Corr
##  id       (Intercept)   2.64688 1.6269
##           groupmeanSES  0.05901 0.2429   -1.00
##  Residual              39.15801 6.2576
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##                  Estimate Std. Error t value
## (Intercept)        11.493      3.292   3.491
## groupmeanSES     -186.500    531.608  -0.351
## sm_ses_grandmean  192.333    531.597   0.362
##
## Correlation of Fixed Effects:
##             (Intr) grpSES
## groupmenSES  0.999
## sm_ss_grndm -0.999 -1.000
## convergence code: 0
## boundary (singular) fit: see ?isSingular
```

Hierarchical modeling

Here we model `groupmeanSES`, the group-mean centered student SES, as a random effect by putting it after the `random =`.

```r
random_slope_level_two_variable_2 <- lme(mathach ~ 1 + groupmeanSES
                                         + sm_ses_grandmean,
                                         random = ~ 1 + groupmeanSES | id,
                                         data = data)
```

Hierarchical modeling

```r
summary(random_slope_level_two_variable_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC   logLik
##   46960.81 47008.96 -23473.4
##
## Random effects:
##  Formula: ~1 + groupmeanSES | id
##  Structure: General positive-definite, Log-Cholesky parametrization
##              StdDev    Corr
## (Intercept)  1.6307657 (Intr)
## groupmeanSES 0.0130297 -0.005
## Residual     6.2575620
##
## Fixed effects: mathach ~ 1 + groupmeanSES + sm_ses_grandmean
##                      Value Std.Error   DF   t-value p-value
## (Intercept)       11.67466    3.2988 7025  3.539092  0.0004
## groupmeanSES     -157.36607  532.6762  157 -0.295425  0.7681
## sm_ses_grandmean  163.22790  532.6697  157  0.306434  0.7597
##  Correlation:
##                  (Intr) grpSES
## groupmeanSES      0.999
## sm_ses_grandmean -0.999 -1.000
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.13126549 -0.75215462  0.02439169  0.76699807  2.78177069
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here we add a cross-level interaction as a fixed effect.

```
fixed_slope_cl_interaction <- lmer(mathach ~ 1 + groupmeanSES*sm_ses_grandmean
                                   + (1|id), data = data)
```

Hierarchical modeling

```r
summary(fixed_slope_cl_interaction)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: mathach ~ 1 + groupmeanSES * sm_ses_grandmean + (1 | id)
##    Data: data
##
## REML criterion at convergence: 46945
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.11930 -0.75112  0.02448  0.76597  2.78831
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  id       (Intercept)  2.664   1.632
##  Residual             39.158   6.258
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##                                Estimate Std. Error t value
## (Intercept)                     11.4252     3.3155   3.446
## groupmeanSES                  -213.6963   537.6427  -0.397
## sm_ses_grandmean               219.4884   537.6248   0.408
## groupmeanSES:sm_ses_grandmean   -0.5799     0.7253  -0.800
##
## Correlation of Fixed Effects:
##             (Intr) grpSES sm_ss_
## groupmenSES  0.998
## sm_ss_grndm -0.998 -1.000
## grpmnSES:__  0.094  0.131 -0.131
```

Hierarchical modeling

Again, we add a cross-level interaction as a fixed effect.

```
fixed_slope_cl_interaction_2 <- lme(mathach ~ 1 + groupmeanSES*sm_ses_grandmean,
                                    random = ~ 1 | id, data = data)
```

Hierarchical modeling

Again, we add a cross-level interaction as a fixed effect.

```r
summary(fixed_slope_cl_interaction_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##        AIC      BIC    logLik
##   46956.97 46998.25 -23472.49
##
## Random effects:
##  Formula: ~1 | id
##         (Intercept) Residual
## StdDev:    1.632105 6.257638
##
## Fixed effects: mathach ~ 1 + groupmeanSES * sm_ses_grandmean
##                                 Value Std.Error   DF   t-value p-value
## (Intercept)                  11.42519    3.3155 7025  3.445968  0.0006
## groupmeanSES               -213.69625  537.6427  156 -0.397469  0.6916
## sm_ses_grandmean            219.48842  537.6248  156  0.408256  0.6836
## groupmeanSES:sm_ses_grandmean -0.57991   0.7253  156 -0.799543  0.4252
##  Correlation:
##                              (Intr) grpSES sm_ss_
## groupmeanSES                  0.998
## sm_ses_grandmean             -0.998 -1.000
## groupmeanSES:sm_ses_grandmean  0.094  0.131 -0.131
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.11929841 -0.75112002  0.02448373  0.76596673  2.78831371
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

Here we add the cross-level interaction as a random effect.

```r
random_slope_cl_interaction <- lmer(mathach ~ 1 + groupmeanSES*sm_ses_grandmean
                                    + (1 + groupmeanSES|id), data = data)
```

```
## boundary (singular) fit: see ?isSingular
```

Hierarchical modeling

```r
summary(random_slope_cl_interaction)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula:
## mathach ~ 1 + groupmeanSES * sm_ses_grandmean + (1 + groupmeanSES |
##     id)
##    Data: data
##
## REML criterion at convergence: 46944.6
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.11654 -0.75065  0.02247  0.76812  2.79659
##
## Random effects:
##  Groups   Name         Variance Std.Dev. Corr
##  id       (Intercept)   2.65355 1.6290
##           groupmeanSES  0.04692 0.2166   -1.00
##  Residual              39.15898 6.2577
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##                                Estimate Std. Error t value
## (Intercept)                     11.3252     3.3070   3.425
## groupmeanSES                  -228.1145   536.0175  -0.426
## sm_ses_grandmean               233.9132   536.0005   0.436
## groupmeanSES:sm_ses_grandmean   -0.5251     0.7363  -0.713
##
## Correlation of Fixed Effects:
##             (Intr) grpSES sm_ss_
## groupmenSES  0.998
## sm_ss_grndm -0.998 -1.000
## grpmnSES:__  0.080  0.118 -0.118
## convergence code: 0
```

Hierarchical modeling

Again, we add the cross-level interaction as a random effect.

```
random_slope_cl_interaction_2 <- lme(mathach ~ 1 + groupmeanSES*sm_ses_grandmean,
                                     random = ~ 1 + groupmeanSES | id,
                                     data = data)
```

Hierarchical modeling

```r
summary(random_slope_cl_interaction_2)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##         AIC      BIC    logLik
##   46960.97 47016.01 -23472.49
##
## Random effects:
##  Formula: ~1 + groupmeanSES | id
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev      Corr
## (Intercept)  1.632105137 (Intr)
## groupmeanSES 0.005745282 -0.002
## Residual     6.257637586
##
## Fixed effects: mathach ~ 1 + groupmeanSES * sm_ses_grandmean
##                                 Value Std.Error   DF   t-value p-value
## (Intercept)                  11.42519    3.3155 7025  3.445964  0.0006
## groupmeanSES               -213.69606  537.6432  156 -0.397468  0.6916
## sm_ses_grandmean            219.48823  537.6253  156  0.408255  0.6836
## groupmeanSES:sm_ses_grandmean   -0.57990    0.7253  156 -0.799534  0.4252
##  Correlation:
##                              (Intr) grpSES sm_ss_
## groupmeanSES                  0.998
## sm_ses_grandmean             -0.998 -1.000
## groupmeanSES:sm_ses_grandmean  0.094  0.131 -0.131
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -3.11929848 -0.75111989  0.02448355  0.76596678  2.78831459
##
## Number of Observations: 7185
## Number of Groups: 160
```

Hierarchical modeling

`glmer` is the function in lme4 to estimate generalized hierarchical linear models. In this case, we're estimating a logit model.

```
logit_random_intercept_and_slope <- glmer(minority ~ groupmeanSES +
                                            (1 + groupmeanSES | id),
                                          data = data, family = binomial(link="logit"))
```

Hierarchical modeling

```r
summary(logit_random_intercept_and_slope)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: minority ~ groupmeanSES + (1 + groupmeanSES | id)
##    Data: data
##
##      AIC      BIC   logLik deviance df.resid
##   5453.9   5488.3  -2721.9   5443.9     7180
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.2886 -0.3942 -0.2073  0.1590  6.1544
##
## Random effects:
##  Groups Name         Variance Std.Dev. Corr
##  id     (Intercept)   2.529   1.590
##         groupmeanSES 11.445   3.383    -0.32
## Number of obs: 7185, groups:  id, 160
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.7382     0.1678 -10.359  < 2e-16 ***
## groupmeanSES  -2.0523     0.5370  -3.822 0.000132 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## groupmenSES -0.230
```

Hierarchical modeling

The argument `correlation =` allows you to specify the correlation structure in nlme.

```
specified_variance_covariance_matrix_for_random_effects <- lme(mathach ~ 1
                                                               +groupmeanSES*sm_ses_grandmean,
                                                               random = ~ 1 + groupmeanSES | id,
                                                               correlation = corAR1(),
                                                               data = data)

# just an example, not needed in this case (useful for growth curve models)!
```

Hierarchical modeling

```r
summary(specified_variance_covariance_matrix_for_random_effects)
```

```
## Linear mixed-effects model fit by REML
##  Data: data
##       AIC      BIC   logLik
##   46962.8 47024.71 -23472.4
##
## Random effects:
##  Formula: ~1 + groupmeanSES | id
##  Structure: General positive-definite, Log-Cholesky parametrization
##              StdDev     Corr
## (Intercept)  1.62876592 (Intr)
## groupmeanSES 0.07080922 -0.039
## Residual     6.25836130
##
## Correlation Structure: AR(1)
##  Formula: ~1 | id
##  Parameter estimate(s):
##         Phi
## 0.005104377
## Fixed effects: mathach ~ 1 + groupmeanSES * sm_ses_grandmean
##                                  Value Std.Error   DF   t-value p-value
## (Intercept)                   11.42446    3.3152 7025  3.446083  0.0006
## groupmeanSES                -213.84493  537.5895  156 -0.397785  0.6913
## sm_ses_grandmean             219.63651  537.5716  156  0.408572  0.6834
## groupmeanSES:sm_ses_grandmean   -0.57970    0.7256  156 -0.798894  0.4256
##  Correlation:
##                              (Intr) grpSES sm_ss_
## groupmeanSES                  0.998
## sm_ses_grandmean             -0.998 -1.000
## groupmeanSES:sm_ses_grandmean  0.094  0.131 -0.131
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
```