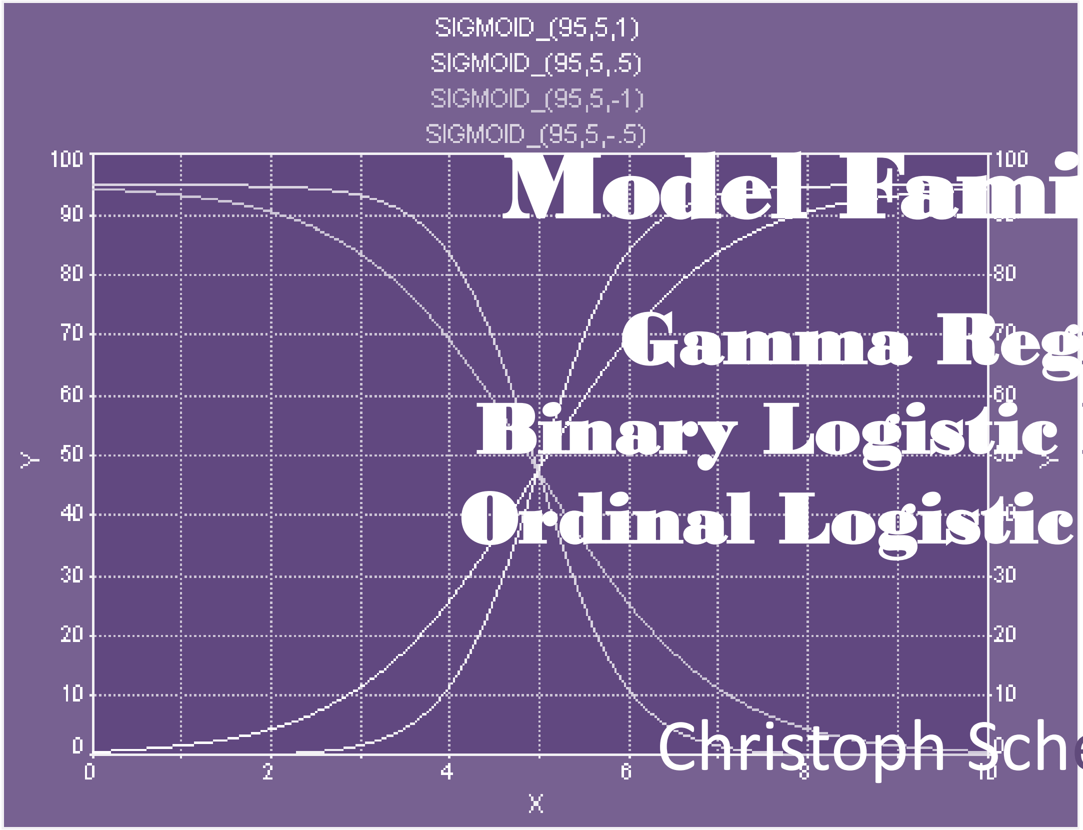
University of Glasgow | Institute of Neuroscience & Psychology

Generalized Linear Models



Model Families, e.g.

Gamma Regression
Binary Logistic Regression
Ordinal Logistic Regression

Christoph Scheepers

christoph.scheepers@glasgow.ac.uk

Generalized Linear Models, `glm()`

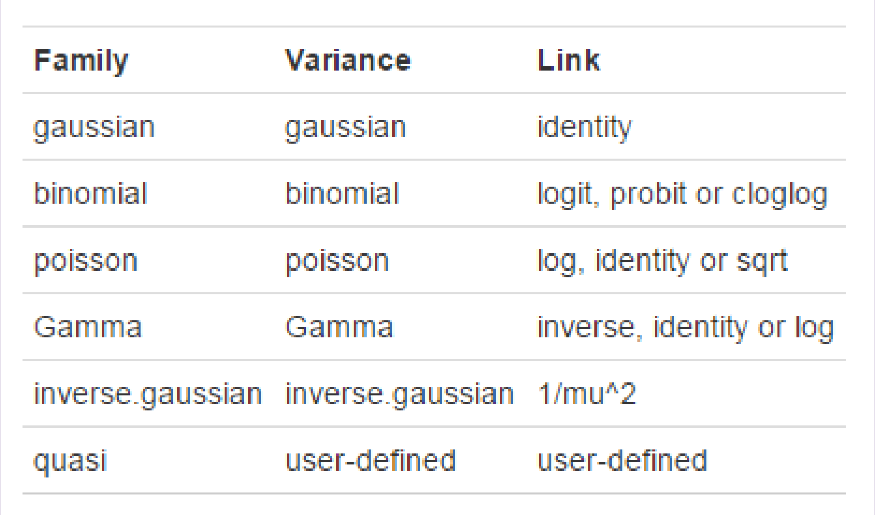
- **Generalized Linear Models** (`glm()`) are an extension of Linear Models (`lm()`) that allow for the specification of ***distribution*** and ***link* functions** (via the **`family`** argument) to **accommodate a variety of different data types:**
  - categorical, count, continuous, etc.
  - Ordinal data (e.g., ratings) require a special package in R (more later)
- This 'generalization' is useful if you want to model data that are not continuous or not normally distributed
- When no family argument is specified, `glm()` assumes a *normal* distribution with *identity* link per default, i.e. `family=gaussian(identity)`
- Estimation of model parameters (a.k.a. optimization) works differently in `glm()` (iterative *maximum likelihood estimation*), and there are also notable differences in the output (e.g., goodness of fit, statistics for model comparison, etc.) compared to `lm()`

`lm()` versus `glm()`

- `lm()` is for ‘standard’ linear models (no transformation of parameters and assuming normality of residuals)
- `glm()` is a generalization of `lm()` that can be applied to a wider range of different data types (incl. binary), via appropriate distribution **(variance) and link functions**
- In fact, **`lm(y~x,…)`** is conceptually equivalent to

```
glm(y~x,
    family=gaussian(identity),
    …)
```

Model families available in `glm()`:



Family	Variance	Link
gaussian	gaussian	identity
binomial	binomial	logit, probit or cloglog
poisson	poisson	log, identity or sqrt
Gamma	Gamma	inverse, identity or log
inverse.gaussian	inverse.gaussian	1/mu^2
quasi	user-defined	user-defined

- ***Variance*** concerns distribution of residuals
- ***Link*** applies a transformation to the model parameters and determines the interpretation of model coefficients (the latter will be given in ‘link’ units)

Previous example: RT as a function of spelling and word frequency

```r
# Simple regression example data
RT.data <- read.csv("http://www.psy.gla.ac.uk/~christop/MScStats/2018/Regress/RTs.csv")

# "deviation coding" (mean-centred dummy coding) of spelling
RT.data$deviat_SP <- scale(ifelse(RT.data$spelling=="lower",0,1), scale=FALSE)
# mean-centring of the continuous logfreq variable
RT.data$cent_LFRQ <- scale(RT.data$logfreq, scale = FALSE)
```

```r
# Perform linear regression using lm()
lm.mod <- lm(RT ~ deviat_SP * cent_LFRQ,
                 data = RT.data)
summary(lm.mod)
```

```
Call:
lm(formula = RT ~ deviat_SP * cent_LFRQ, data = RT.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-128.95  -52.40  -11.84   42.75  325.97 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)          659.763      6.455 102.213  < 2e-16 ***
deviat_SP             17.647     12.910   1.367    0.174    
cent_LFRQ            -25.518      4.975  -5.129 9.51e-07 ***
deviat_SP:cent_LFRQ  -14.315      9.950  -1.439    0.152    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 77.46 on 140 degrees of freedom
Multiple R-squared:  0.1777,	Adjusted R-squared:   0.16 
F-statistic: 10.08 on 3 and 140 DF,  p-value: 4.665e-06
```

```r
# Perform linear regression using glm()
glm.mod <- glm(RT ~ deviat_SP * cent_LFRQ,
                    data = RT.data,
                    family=gaussian(identity))
summary(glm.mod)
```

```
Call:
glm(formula = RT ~ deviat_SP * cent_LFRQ, data = RT.data)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-128.95   -52.40   -11.84    42.75   325.97  

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)          659.763      6.455 102.213  < 2e-16 ***
deviat_SP             17.647     12.910   1.367    0.174    
cent_LFRQ            -25.518      4.975  -5.129 9.51e-07 ***
deviat_SP:cent_LFRQ  -14.315      9.950  -1.439    0.152    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 5999.61)

    Null deviance: 1021421  on 143  degrees of freedom
Residual deviance:  839945  on 140  degrees of freedom
AIC: 1667.3
```

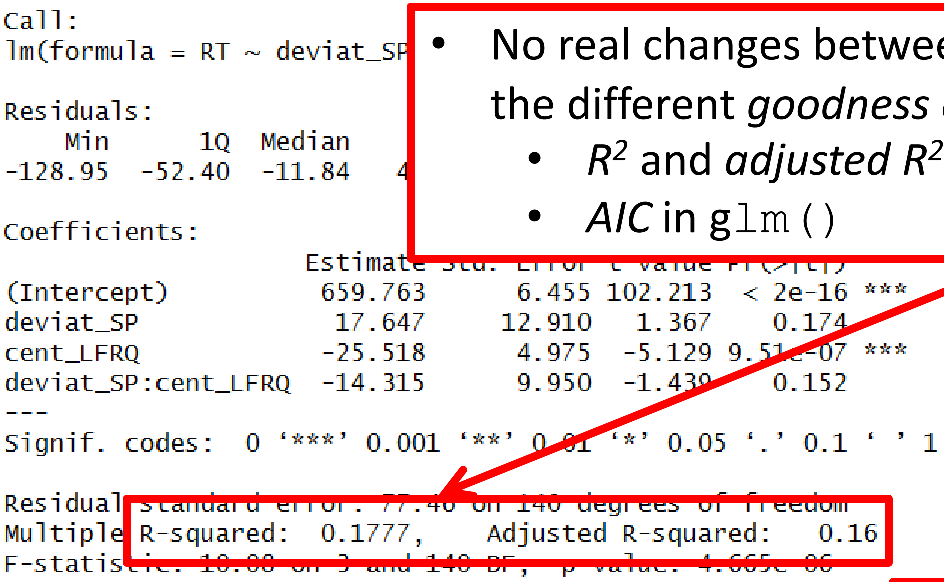
Previous example: RT as a function of spelling and word frequency

```
# Simple regression example data
RT.data <- read.csv("http://www.psy.gla.ac.uk/~christop/MScStats/2018/Regress/RTs.csv")

# "deviation coding" (mean-centred dummy coding) of spelling
RT.data$deviat_SP <- scale(ifelse(RT.data$spelling=="lower",0,1), scale=FALSE)
# mean-centring of the continuous logfreq variable
RT.data$cent_LFRQ <- scale(RT.data$logfreq, scale = FALSE)
```

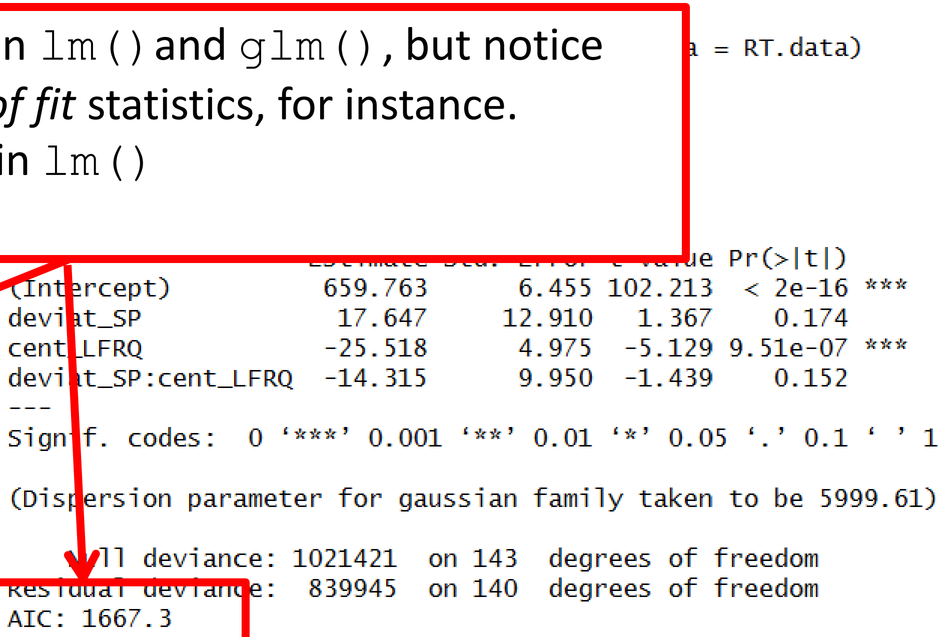
```
# Perform linear regression using lm()
lm.mod <- lm(RT ~ deviat_SP * cent_LFRQ,
             data = RT.data)
summary(lm.mod)
```

```
# Perform linear regression using glm()
glm.mod <- glm(RT ~ deviat_SP * cent_LFRQ,
               data = RT.data,
               family=gaussian(identity))
summary(glm.mod)
```

- No real changes between `lm()` and `glm()`, but notice the different *goodness of fit* statistics, for instance.
  - *R^2* and *adjusted R^2* in `lm()`
  - *AIC* in `glm()`



```
Call:
lm(formula = RT ~ deviat_SP

Residuals:
    Min      1Q  Median
-128.95  -52.40  -11.84    4

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          659.763      6.455 102.213  < 2e-16 ***
deviat_SP             17.647     12.910   1.367    0.174
cent_LFRQ            -25.518      4.975  -5.129 9.51e-07 ***
deviat_SP:cent_LFRQ  -14.315      9.950  -1.439    0.152
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 77.46 on 140 degrees of freedom
Multiple R-squared:  0.1777,    Adjusted R-squared:   0.16
F-statistic: 10.08 on 3 and 140 DF,  p-value: 4.665e-06
```



```
a = RT.data)

                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          659.763      6.455 102.213  < 2e-16 ***
deviat_SP             17.647     12.910   1.367    0.174
cent_LFRQ            -25.518      4.975  -5.129 9.51e-07 ***
deviat_SP:cent_LFRQ  -14.315      9.950  -1.439    0.152
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 5999.61)

    Null deviance: 1021421  on 143  degrees of freedom
Residual deviance:  839945  on 140  degrees of freedom
AIC: 1667.3
```

Goodness of Fit: *AIC*

- As an analogue to $adjusted\ R^2$ in standard linear regression, `glm()` reports ***Akaike Information Criterion*** (*AIC*) as a goodness-of-fit measure (bottom of summary output)

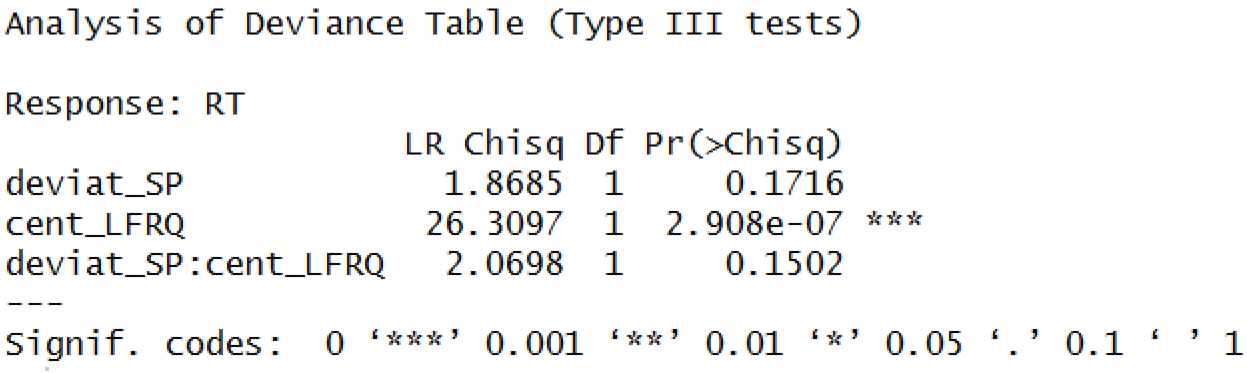
```
# also accessible via AIC() function
AIC(glm.mod)
```

```
[1] 1667.318
```

- See http://en.wikipedia.org/wiki/Akaike_information_criterion
- It takes into account how well the model explains the data, plus a penalty for model complexity
- **Note**: ***Lower* values of AIC mean *better* fit**
- **Cannot be interpreted in an absolute sense**
- But very useful for **model comparison**! (see further down…)

Say goodbye to *F*-values in `glm()`!

```r
# Anova() on glm() object
library(car)
Anova(glm.mod, type="III")
```



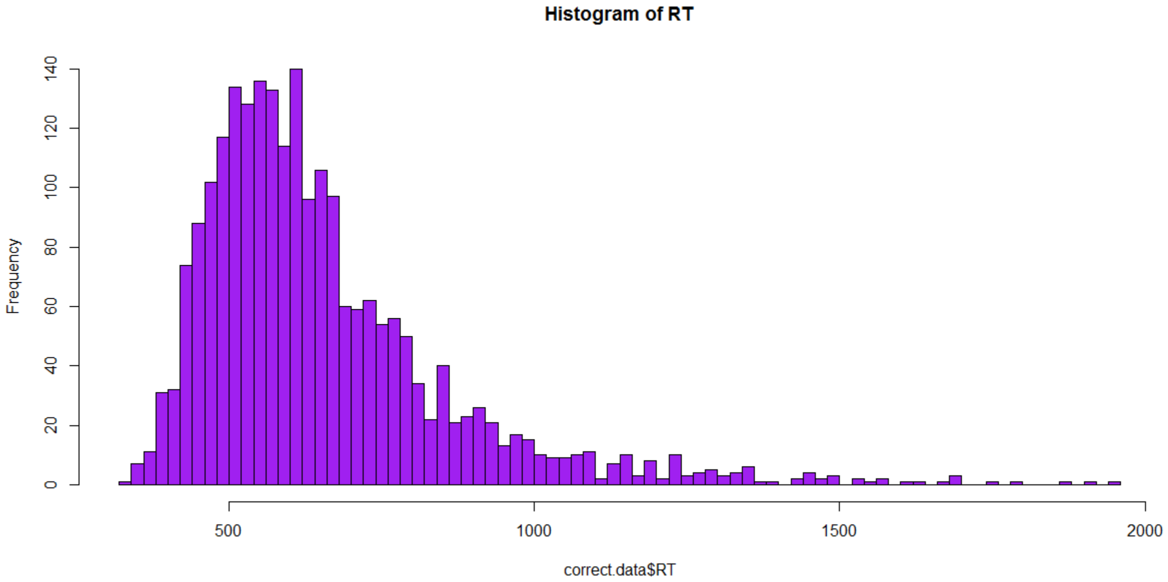
```
Analysis of Deviance Table (Type III tests)

Response: RT
                     LR Chisq Df Pr(>Chisq)
deviat_SP              1.8685  1     0.1716
cent_LFRQ             26.3097  1  2.908e-07 ***
deviat_SP:cent_LFRQ    2.0698  1     0.1502
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- For `glm()` objects (or more generally, model objects based on maximum likelihood estimation), `anova()` and `Anova()` report *Analysis of Deviance Tables*
- *Likelihood Ratio Chi-Square* instead of *F*
- No error degrees of freedom

  (e.g., report $LR\chi^2 = 1.869$, $df = 1$, $p = .172$ for the main effect of spelling)

`glm()` for Response Times

- RTs are hardly ever perfectly normally distributed!
- Characteristic *positive skew* in RT distributions (RTs are theoretically bounded to range from 0 to +∞)



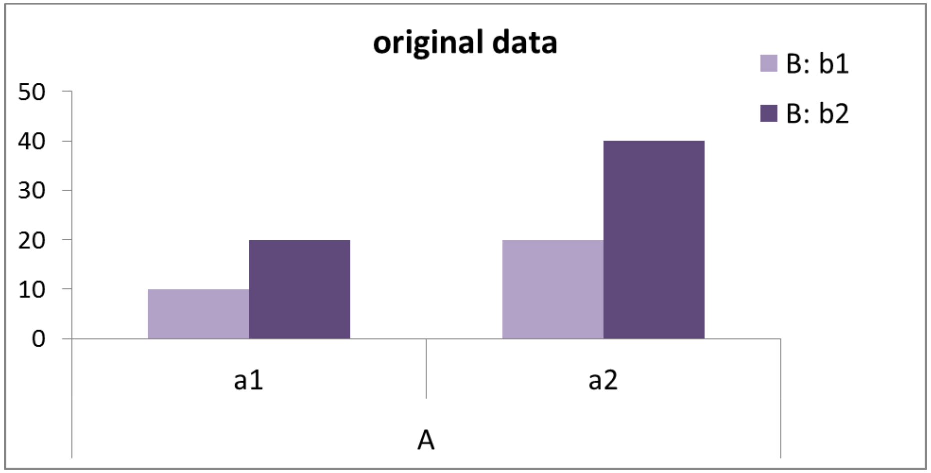
- Some authors therefore recommend *log-transforming* RT data prior to analysis (coerce them into Normal), or model them using a *Gamma* distribution function

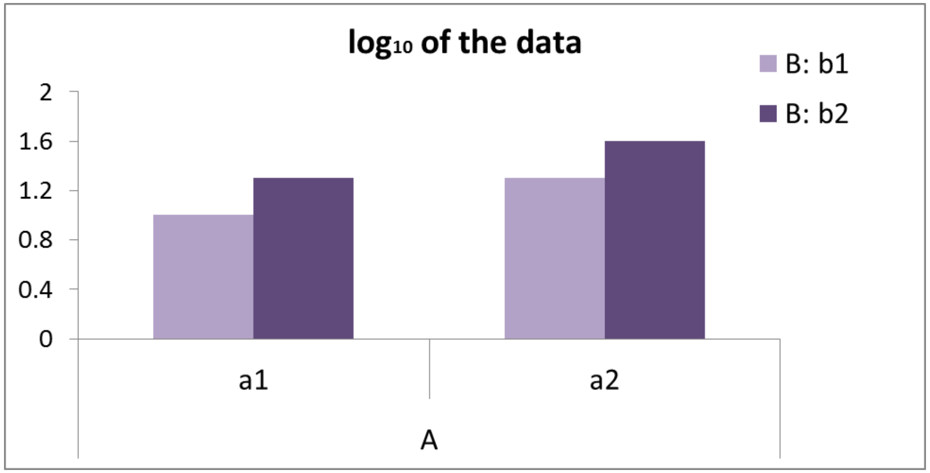
  (see https://www.statlect.com/probability-distributions/gamma-distribution)

`glm()` for Response Times

- Problem with log-transformation of data (same holds true when using a log-link in `glm()`):
  - It sometimes fails (e.g. DV-values ≤ 0)
  - It affects **theoretical interpretation** of model coefficients because it effectively implements a **multiplicative** model of the original data:
    - **Note:** *log*(A) + *log*(B) = *log*(A×B) and *log*(A) - *log*(B) = *log*(A/B)
    - Change in interpretation is not always desirable
- With, say, a `family=Gamma(identity)` approach in `glm()`, we maintain the assumption of additive relationships in the RT data (*identity* link), but account for a positive skew in the residuals (*Gamma* distribution), thereby potentially improving the model fit
- With, say, a `family=Gamma(log)` approach in `glm()`, we specify a multiplicative model of the original RT data (*log* link) and account for a positive skew in the residuals (*Gamma* distribution)

Multiplicative ('log-linear') relationships and (certain kinds of) interactions





Original data: Interaction

Effect of B at a1 = 10 y-units

Effect of B at a2 = 20 y-units

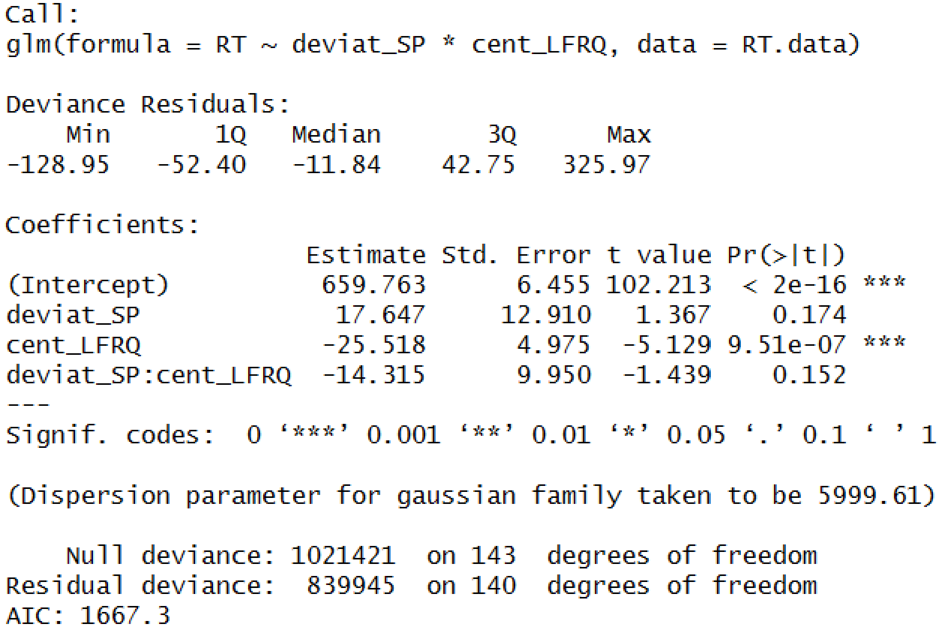
Log data: No Interaction

Effect of B at a1 = 0.3 log y-units

Effect of B at a2 = 0.3 log y-units

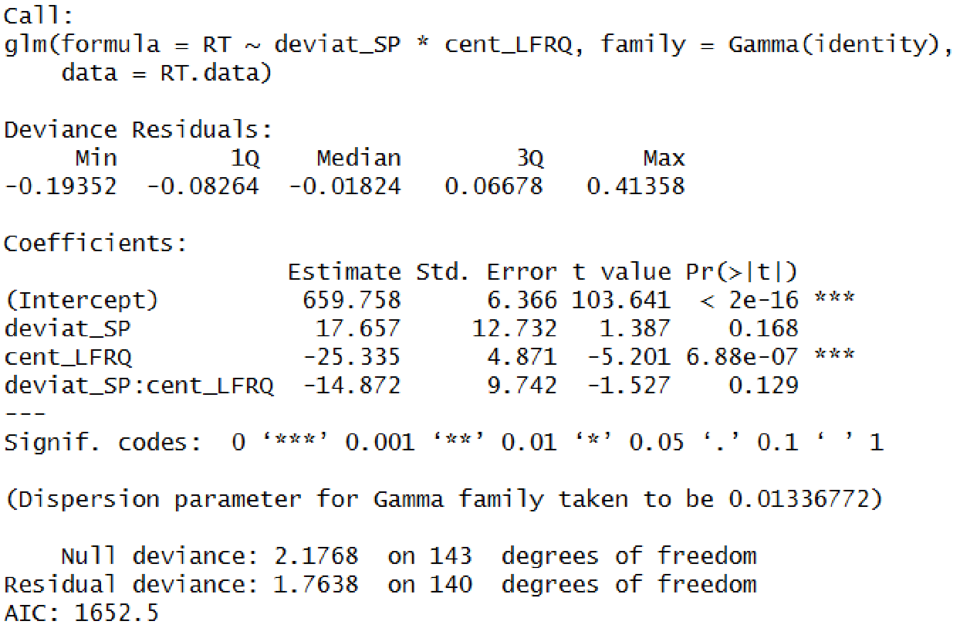
Previous example: RT as a function of spelling and word frequency

```
# Perform linear regression using glm()
glm.mod <- glm(RT ~ deviat_SP * cent_LFRQ,
               data = RT.data,
               family=gaussian(identity))
summary(glm.mod)
```



```
Call:
glm(formula = RT ~ deviat_SP * cent_LFRQ, data = RT.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-128.95   -52.40   -11.84    42.75   325.97

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          659.763      6.455 102.213  < 2e-16 ***
deviat_SP             17.647     12.910   1.367    0.174
cent_LFRQ            -25.518      4.975  -5.129 9.51e-07 ***
deviat_SP:cent_LFRQ  -14.315      9.950  -1.439    0.152
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 5999.61)

    Null deviance: 1021421  on 143  degrees of freedom
Residual deviance:  839945  on 140  degrees of freedom
AIC: 1667.3
```

```
# Perform glm() using Gamma(identity)
glm.mod2 <- glm(RT ~ deviat_SP * cent_LFRQ,
                data = RT.data,
                family=Gamma(identity))
summary(glm.mod2)
```



```
Call:
glm(formula = RT ~ deviat_SP * cent_LFRQ, family = Gamma(identity),
    data = RT.data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.19352  -0.08264  -0.01824   0.06678   0.41358

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          659.758      6.366 103.641  < 2e-16 ***
deviat_SP             17.657     12.732   1.387    0.168
cent_LFRQ            -25.335      4.871  -5.201 6.88e-07 ***
deviat_SP:cent_LFRQ  -14.872      9.742  -1.527    0.129
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for Gamma family taken to be 0.01336772)

    Null deviance: 2.1768  on 143  degrees of freedom
Residual deviance: 1.7638  on 140  degrees of freedom
AIC: 1652.5
```

- **Model coefficients** (intercept and slopes for the various effects) come close between the two approaches, but **are not exactly the same** (because of different model assumptions)
- The **`Gamma(identity)`** model (*AIC* = 1652.5) fits the data slightly better than the standard linear model (*AIC* = 1667.3)
- The difference in *AIC* would be more dramatic with trial-level data!!

What about also using a *log*-link?

```r
# Perform glm() using Gamma(identity)
glm.mod2 <- glm(RT ~ deviat_SP * cent_LFRQ,
                data = RT.data,
                family=Gamma(identity))
summary(glm.mod2)
```

```
Call:
glm(formula = RT ~ deviat_SP * cent_LFRQ, family = Gamma(identity),
    data = RT.data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.19352  -0.08264  -0.01824   0.06678   0.41358

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          659.758      6.366 103.641  < 2e-16 ***
deviat_SP             17.657     12.732   1.387    0.168
cent_LFRQ            -25.335      4.871  -5.201 6.88e-07 ***
deviat_SP:cent_LFRQ  -14.872      9.742  -1.527    0.129
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for Gamma family taken to be 0.01336772)

    Null deviance: 2.1768  on 143  degrees of freedom
Residual deviance: 1.7638  on 140  degrees of freedom
AIC: 1652.5
```

```r
# Perform glm() using Gamma(log)
glm.mod3 <- glm(RT ~ deviat_SP * cent_LFRQ,
                data = RT.data,
                family=Gamma(log))
summary(glm.mod3)
```

```
Call:
glm(formula = RT ~ deviat_SP * cent_LFRQ, family = Gamma(log),
    data = RT.data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.19282  -0.08385  -0.01777   0.06544   0.41512

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          6.490445   0.009636 673.579  < 2e-16 ***
deviat_SP            0.025380   0.019272   1.317    0.190
cent_LFRQ           -0.038598   0.007427  -5.197 7.01e-07 ***
deviat_SP:cent_LFRQ -0.021403   0.014853  -1.441    0.152
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for Gamma family taken to be 0.0133701)

    Null deviance: 2.1768  on 143  degrees of freedom
Residual deviance: 1.7633  on 140  degrees of freedom
AIC: 1652.5
```

- **Model coefficients** (intercept and slopes for the various effects) **are now very different:**
    - In the **`Gamma(identity)`** model, estimates and SEs are in the original **RT-units** (milliseconds)
    - In the **`Gamma(log)`** model, estimates and SEs are in ***log*(RT)-units** (log milliseconds)
- In this instance, goodness of fit stays the same (*AIC* = 1652.5 in each case)

So, what's the best '*family recipe*' for RT data…?

- **Better use your brain / good theories - not recipes!**
- From experience with modelling RT data (or other types of *naturally positively skewed* DVs), I can tell that `Gamma(identity)` yields much better fits than a standard linear approach (`gaussian(identity)`)
- A better fit means better modelling of the generative processes behind the data, and often yields *improved power*
- A `Gamma(log)` (or `Gamma(inverse)`) model may also make sense, but remember that non-identity link functions change the theoretical interpretation of your model coefficients (e.g., turning additive relations into multiplicative ones when considering the original DV)
- **Justify which model family you are using**
- Never ever '*shop around*' for model families *giving the nicest p-values*!

How important is the ‘correct’ family?

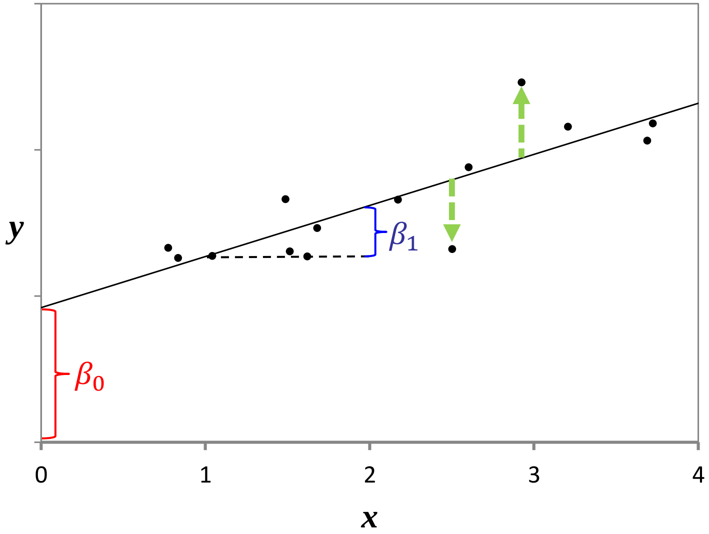
- **For *continuous* data,** the ‘default’ normal distribution / identity link assumption (cf. ANOVA, `lm()`) actually does a fairly good job in most cases
- ANOVA, for example, has been shown to be remarkably robust against violations of normality
  - If anything, such violations are detrimental to power, but not to Type I error rate (e.g., Khan & Rayner, 2003)
- However, other types of data require more careful consideration of the correct model family (for theoretical and statistical reasons)
- A prominent example are ***binary categorical data*** which will be discussed next

Recall simple linear regression

- **Goal**: Predict a continuous DV (*y*) from a continuous IV (x), assuming a *linear relationship* between the two

$$\hat{y}_i = \beta_o + \beta_1 x_i \ ,$$

$$y_i = \hat{y}_i + e_i$$



where

$\hat{y}_i$ = predicted value of y_i

x_i = value of the predictor variable

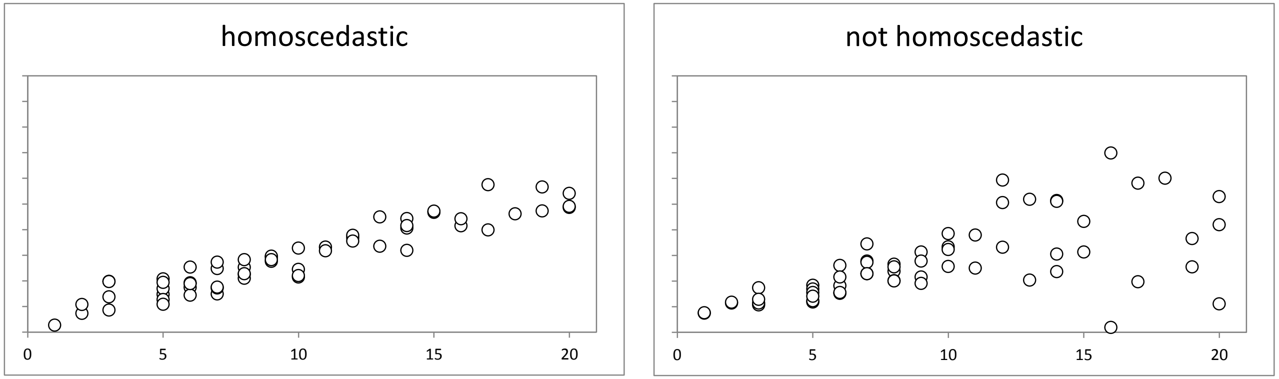
β_0 = the ***intercept*** (or *regression constant*): the value of $\hat{y}_i$ when x = 0

β_1 = the ***slope*** (or *regression coefficient*): the difference in $\hat{y}_i$ associated with a one-unit increase in x

e_i = prediction ***error*** (residuals)

Assumptions

- Both IV and DV are measured on interval scale (continuous data)
  - Can theoretically range from −∞ to +∞
- Linearity / additivity
- Homoscedasticity
  - Constant variance of residuals over the entire x-range, e.g.



- Normality of residuals
  - $e_i \sim N(0, \sigma)$

Binary categorical DVs

- Sometimes the values of the DV of interest come in only two flavours, i.e. 0 or 1
  - Female/Male, pregnant/not pregnant, correct/incorrect, … etc.
- That is, what we want to do is to somehow **predict the probability of belonging to one or the other category** as a function of our IV(s)
- Linear regression would not work in this case
  - Binary data are nominal scale (discrete), and their probabilities are bound between 0 and 1 (with small / large x_i, linear regression will result in $\hat{y}_i$-values <0 or >1)
  - The relationship between x and y will **not** be **linear**
  - **Normality?** The appropriate distribution for numbers of '1s' in a sequence of independent Bernoulli (0,1) trials is actually the **binomial** distribution
  - **Heteroscedasticity** of residuals: the closer predicted probability values are to 0 or 1, the smaller the corresponding residual variances will be (error variance will be greatest when predicted probability is around .5)

Binary categorical DVs

- Sometimes the values of the DV of interest come in only two flavours, i.e. 0 or 1
  - Female/Male, pregnant/not pregnant, correct/incorrect, … etc.
- That is, what we want to do is to somehow **predict the probability of belonging to one or the other category** as a function of our IV
- Linear regression would not work in this case
  - Binary data are nominal scale (discrete), and their probabilities are bound between 0 and 1 (with small / large x_i, linear regression might well result in $\hat{y}_i$-values <0 or >1)
  - The relationship between x and y will **not** be **linear**
  - **Normality?** The appr … a sequence of independent Bernou … bution
  - **Heteroscedasticity** o … values are to 0 or 1, the smaller the … error variance will be greatest wher

Note: For binary variables, the population variance is $P(1) \times P(0)$
Thus,
with P(1) = .5, σ^2 = .5 × .5 = .25
with P(1) = .1, σ^2 = .1 × .9 = .09
etc.

An example

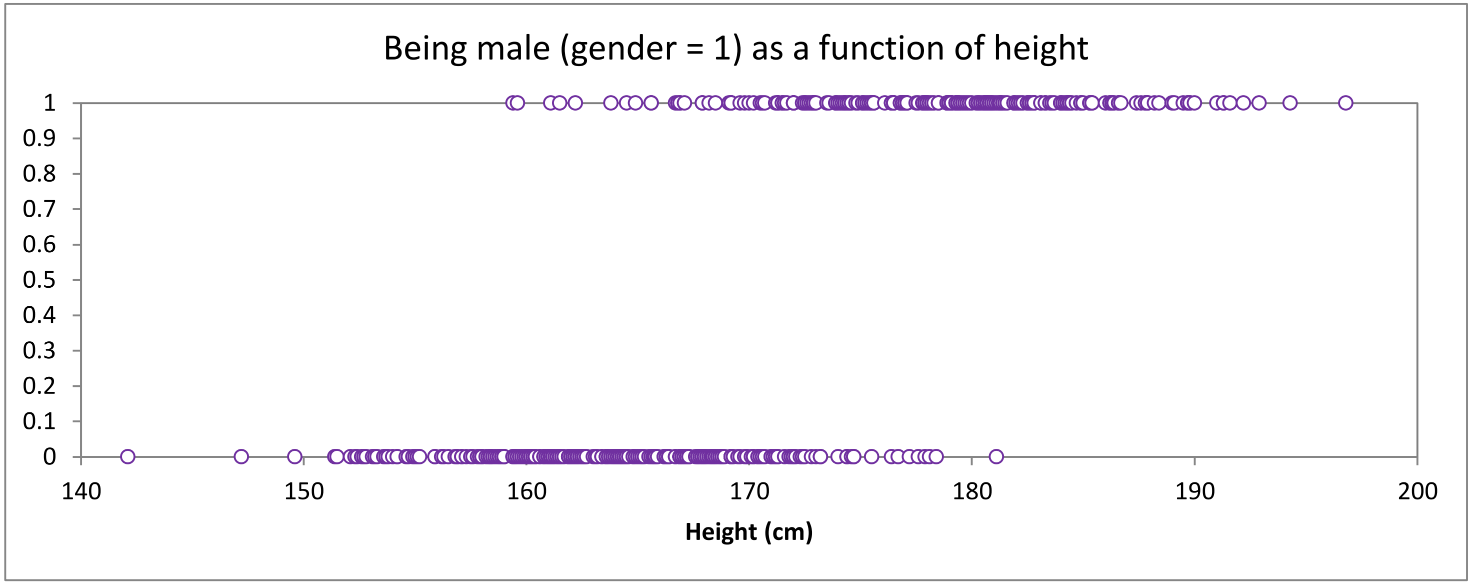
- Let’s suppose we randomly sampled 500 Scottish adults and measured their body height in cm
- I generated such a dataset using the following parameters:
- N = 500
- gender $\sim U(0,1)$ => roughly 50% males
- height | gender = 0 $\sim N(163.5, 6.1)$
- height | gender = 1 $\sim N(178.2, 7.0)$

values taken from Wikipedia

- **Goal**: We want to predict a person’s **gender** from their body height
  - **Classification** problem with
    - Continuous IV (height in cm)
    - Binary DV (female = 0, male = 1)

An example

subj_ID	height	gender
1	156.8	0
2	166.9	0
3	164.5	0
4	191.6	1
5	161.5	0
6	182.8	1
7	180.7	1
8	162.2	0
9	164.4	0
10	166.9	0
11	178.4	0
12	155.9	0
13	161.1	0
14	154.2	0
15	161	0
16	180.6	1
17	162.5	0
18	179.5	1
19	153.3	0
20	165.6	0
21	184.4	1
…	…	…



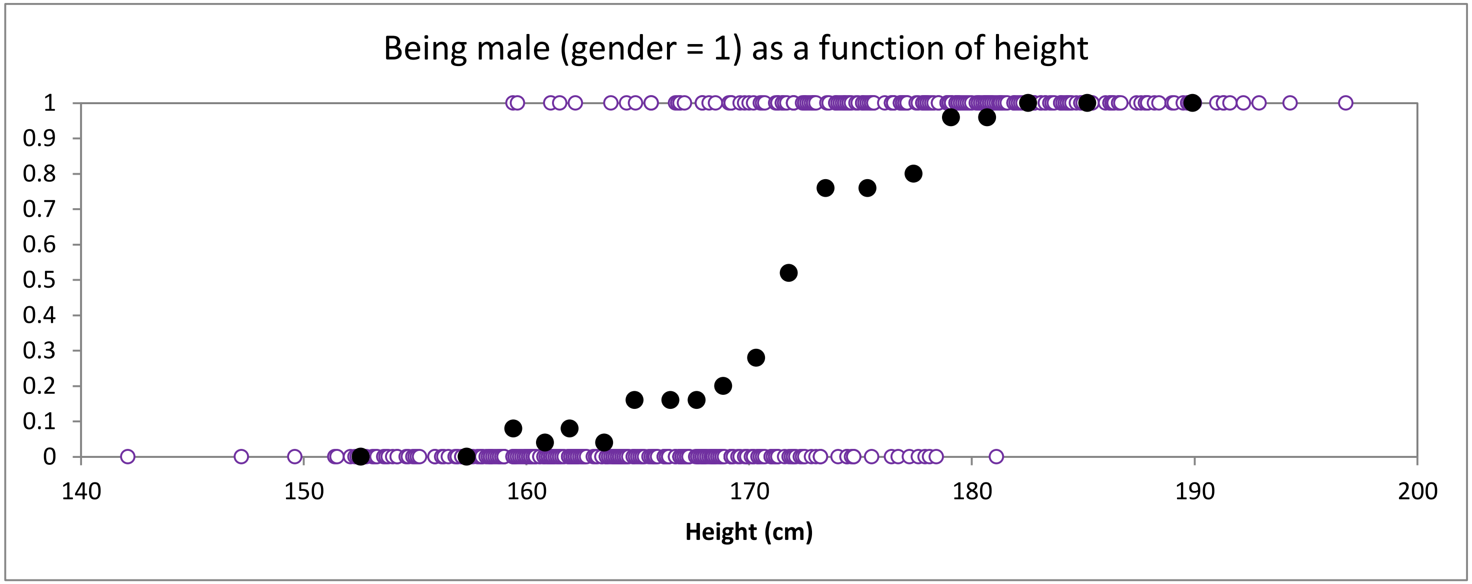
- DV (male) is coded as 0 (for female) or 1 (for male)
- If you plot the data in R using:

```
plot(gender ~ height)
```

It would look like the above

An Example

subj_ID	height	gender
1	156.8	0
2	166.9	0
3	164.5	0
4	191.6	1
5	161.5	0
6	182.8	1
7	180.7	1
8	162.2	0
9	164.4	0
10	166.9	0
11	178.4	0
12	155.9	0
13	161.1	0
14	154.2	0
15	161	0
16	180.6	1
17	162.5	0
18	179.5	1
19	153.3	0
20	165.6	0
21	184.4	1
…	…	…



- When we look at the **probability** of being male (here, for each 5% height-bin), we see that P(gender=1) as a function of height follows a roughly "**S-shaped**" (*sigmoid*) function
- This is a natural consequence of the two **partially overlapping** normal height-distributions (one for males and one for females)

Logistic Function

- The probability of being male as a function of height – or more generally, the **probability of a given binary category $\boldsymbol{y}$ as a function of $\boldsymbol{x}$** (IV) – can be modelled by the following equation:

$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)} \text{ or } \frac{1}{1 + \exp(-(\beta_o + \beta_1 x_i))}$$

Logistic Function

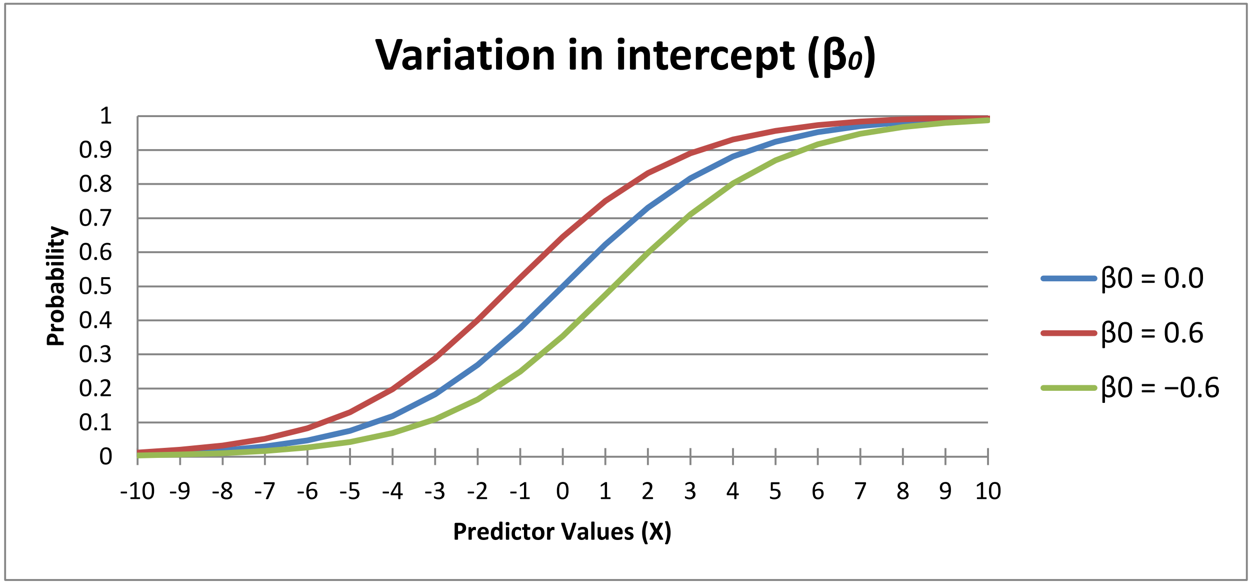
- The probability of being male as a function of height – or more generally, the **probability of a given binary category y as a function of x** (IV) – can be modelled by the following equation:

$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)} \text{ or } \frac{1}{1 + \exp(-(\beta_o + \beta_1 x_i))}$$

Looks suspiciously like our good old *linear regression* model
- More later!

Logistic Function

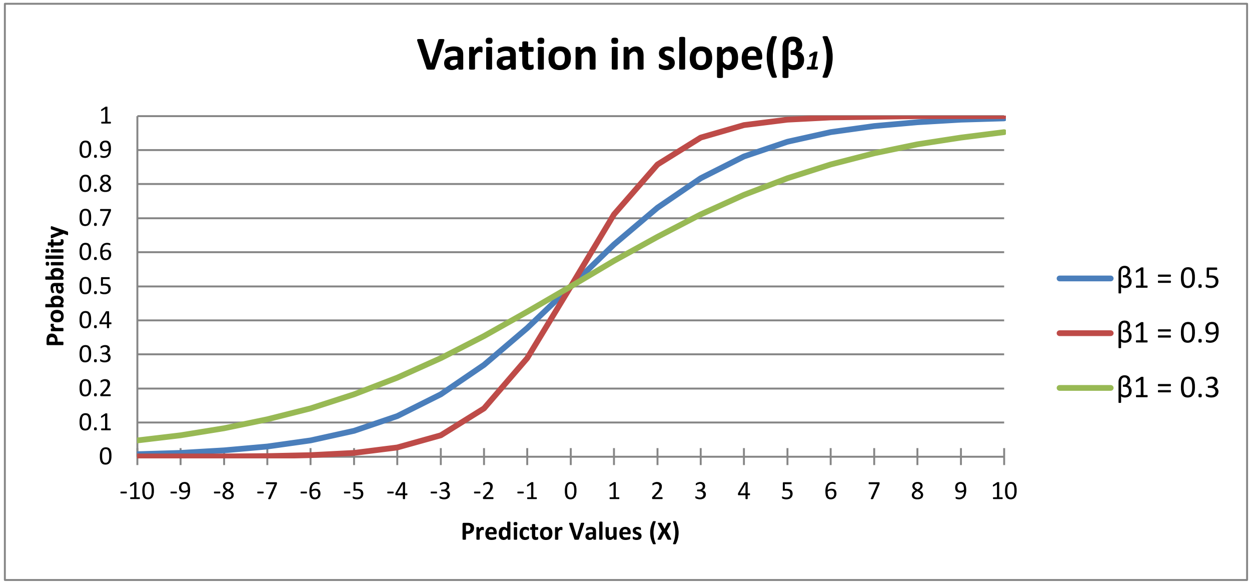
$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)} \text{ or } \frac{1}{1 + \exp(-(\beta_o + \beta_1 x_i))}$$



$(\beta_1 = 0.5)$

Logistic Function

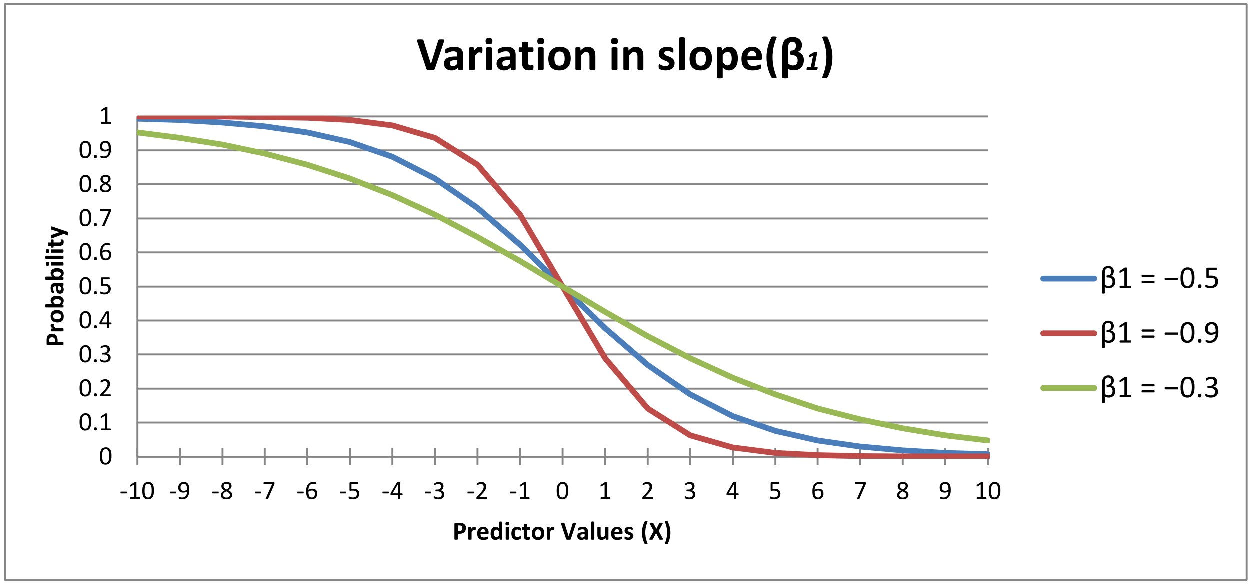
$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)} \text{ or } \frac{1}{1 + \exp(-(\beta_o + \beta_1 x_i))}$$



$(\beta_0 = 0.0)$

Logistic Function

$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)} \text{ or } \frac{1}{1 + \exp(-(\beta_o + \beta_1 x_i))}$$



$(\beta_0 = 0.0)$

Odds

- Instead of probabilities, we could also conceptualize the problem in terms of **odds**
- What are the odds of being male given a certain probability of being male?
- Answer: $odds(male) = \frac{P(male)}{P(female)}$ = $\frac{P(male)}{1 - P(male)}$
- More generally: $odds(y) = \frac{P(y)}{1 - P(y)}$
- Say, if in a given sample the probability of being male is .6, the odds of being male are .6/.4 = 1.5, i.e. in that sample, it’s 1.5 times more likely to find males than females.
- To convert odds back into probabilities, use: $P(y) = \frac{odds(y)}{1 + odds(y)}$
- Probabilities and odds have different properties, e.g.:
  - $P(y)$ ranges from 0 to 1, but $odds(y)$ ranges from 0 to +∞

Log Odds or "Logits"

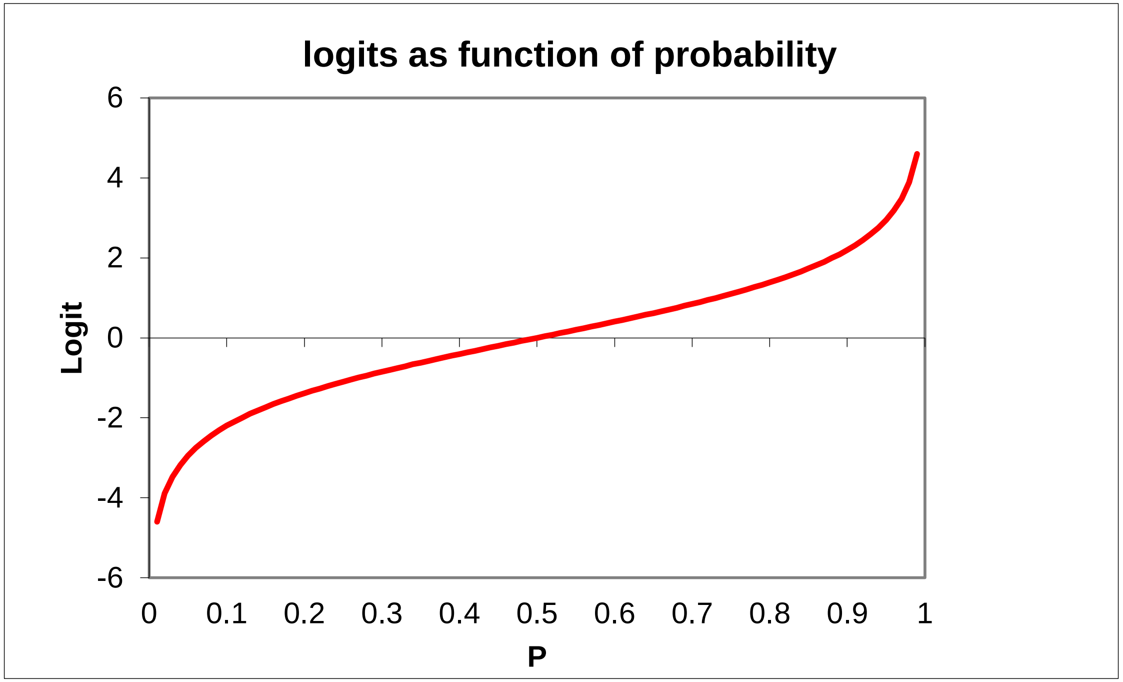
- The natural logarithm of $odds(y)$ is called **log odds** or **logit**

$$logit(y) = \ln\big(odds(y)\big) = \ln(\frac{P(y)}{1-P(y)})$$

- Where $\ln(x)$ refers to the log based on Euler's number (ca. 2.7182818284590452353602874713527…)

- Logits have the following properties:
  - If $odds(y) = 1$; $P(y) = .5$; $logit(y) = 0$
  - If $odds(y) < 1$; $P(y) < .5$; $logit(y) < 0$
  - If $odds(y) > 1$; $P(y) > .5$; $logit(y) > 0$
  - The logit transform fails if $P(y) = 0$ or $P(y) = 1$
  - Logits range between −∞ to +∞

Logit as a function of *P*



- In the “middle” probability range, small changes in *P* imply small changes in *logit*
- When probabilities approach one of the logical boundaries (0 or 1), small changes in *P* imply large changes in *logit*
- Compensates for the heteroscedasticity problem associated with probabilities

To convert logits back into probabilities, use the ***inverse logit*** function:

$$P = \frac{\exp(logit)}{1+\exp(logit)}$$

Interesting, but why care?..

- Since:

$$\widehat{P(y_i)} = \frac{\exp(\beta_o + \beta_1 x_i)}{1 + \exp(\beta_o + \beta_1 x_i)}$$

- It follows that:

$$\frac{\widehat{P(y_i)}}{1 - \widehat{P(y_i)}} = \exp(\beta_o + \beta_1 x_i)$$

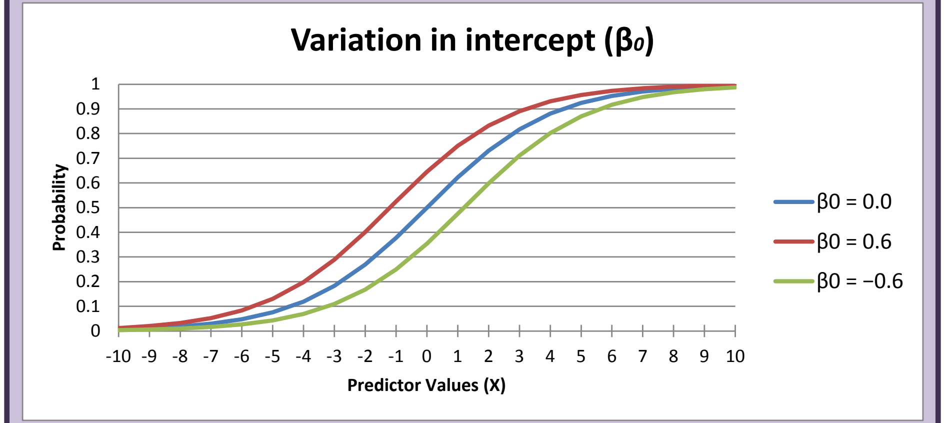
- And therefore:

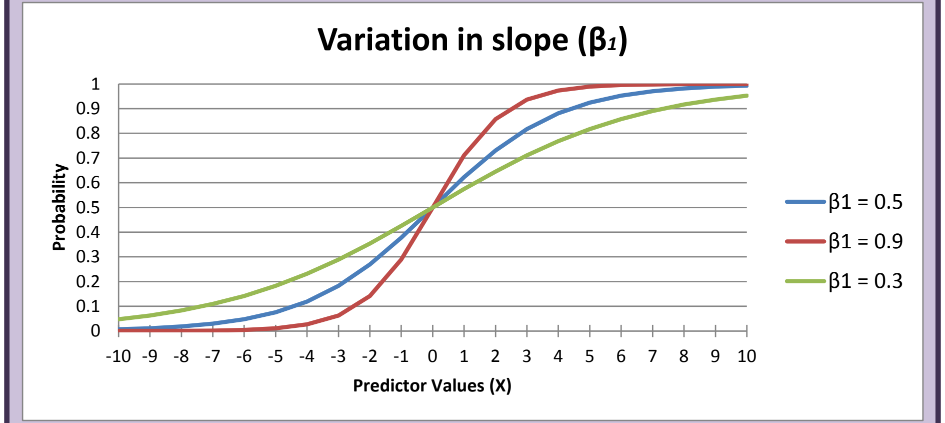
$$\ln\left(\frac{\widehat{P(y_i)}}{1 - \widehat{P(y_i)}}\right) = \beta_o + \beta_1 x_i$$

- In other words: Applying a **logistic function to $\boldsymbol{P(y)}$** is pretty much the same as applying a **linear function to the log odds (or logit) of $\boldsymbol{P(y)}$** - which is essentially what **binary logistic regression** does!

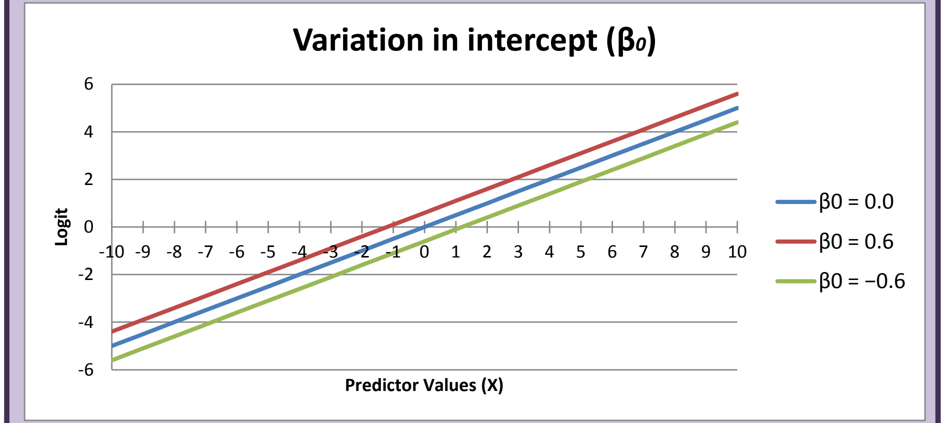
Just to confirm…

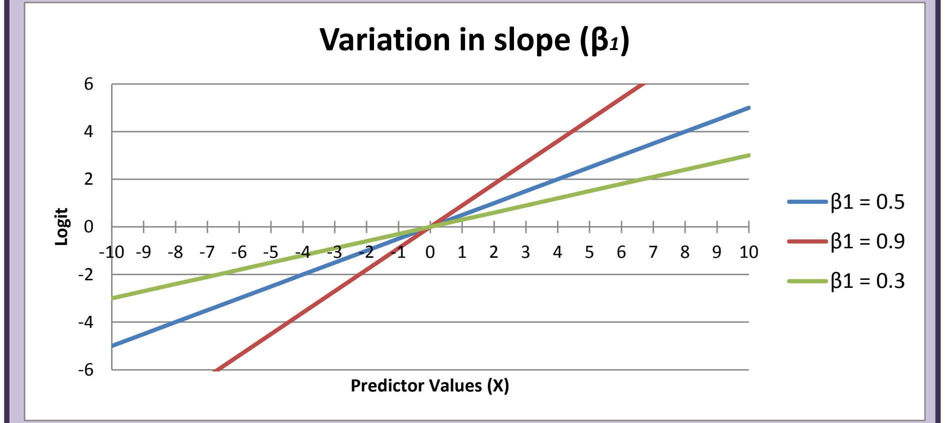
The `parish of probabilities`





The *land of logits*





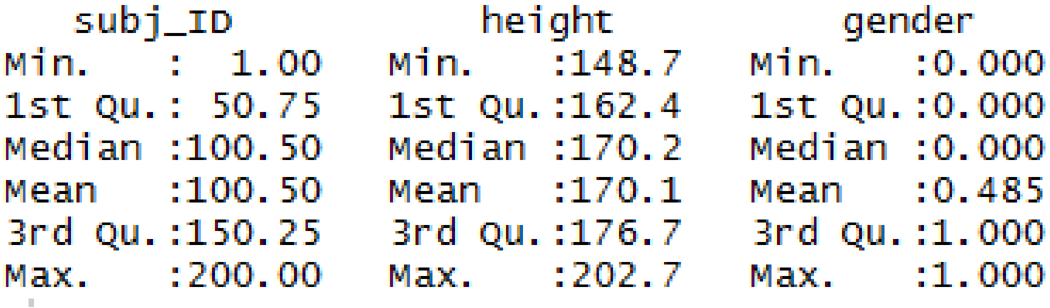
Estimation and error distribution

- In standard linear regression, estimation is based on minimizing sums of squared residuals (SS_{res}) for which simple arithmetic solutions exist
- With non-linear models (e.g., logistic), an *iterative approach* based on **maximum likelihood** estimation must be taken (maximising $P(y|x)$)
- Usually requires specification of convergence criteria such as maximum number of iterations and/or ‘stop’ thresholds (e.g., if likelihood doesn’t improve by more than .00001 then stop iterating).
- Since we are basically dealing with probabilities of binary outcomes, the residuals cannot be normally distributed; instead logistic regression assumes a **binomial distribution** of the errors.

Let’s do this in R…

- Let’s go back to the original example (gender as a function of height)
- New, smaller dataset (N = 200), but generated from the same parameters:
  - gender $\sim U(0,1)$
  - height | gender = 0 $\sim N(163.5, 6.1)$
  - height | gender = 1 $\sim N(178.2, 7.0)$

```
# Example Data
height.data <- read.csv("http://www.psy.gla.ac.uk/~christop/MScStats/2018/Logist/height1.csv")
summary(height.data)
```

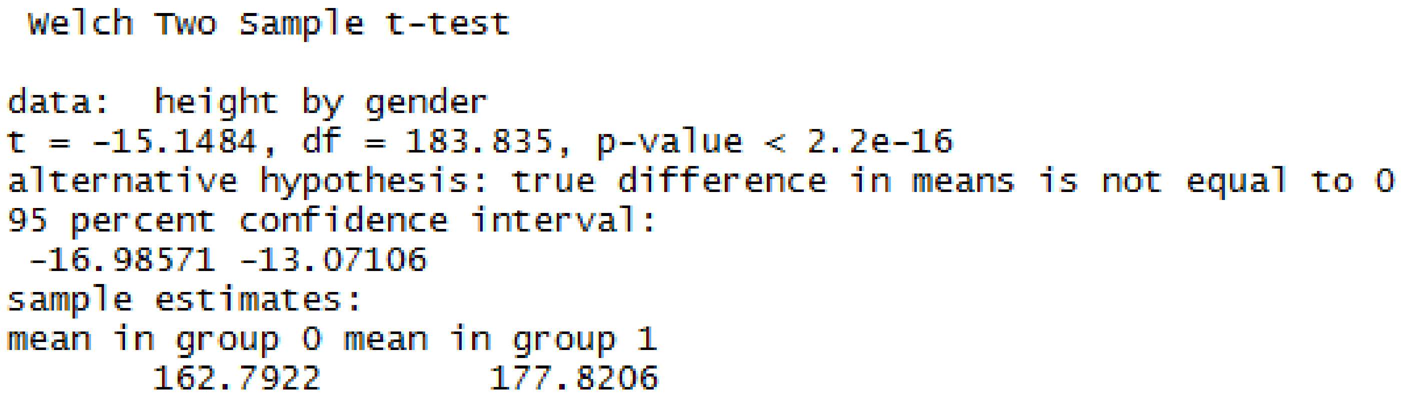


```
    subj_ID           height          gender     
 Min.   :  1.00   Min.   :148.7   Min.   :0.000  
 1st Qu.: 50.75   1st Qu.:162.4   1st Qu.:0.000  
 Median :100.50   Median :170.2   Median :0.000  
 Mean   :100.50   Mean   :170.1   Mean   :0.485  
 3rd Qu.:150.25   3rd Qu.:176.7   3rd Qu.:1.000  
 Max.   :200.00   Max.   :202.7   Max.   :1.000  
```

(height as a function of gender)

- We can clearly see that males and females were drawn from different (normal) height distributions by running a t-test

```r
# t-test with height as a function of gender
t.test(height ~ gender, data=height.data)
```

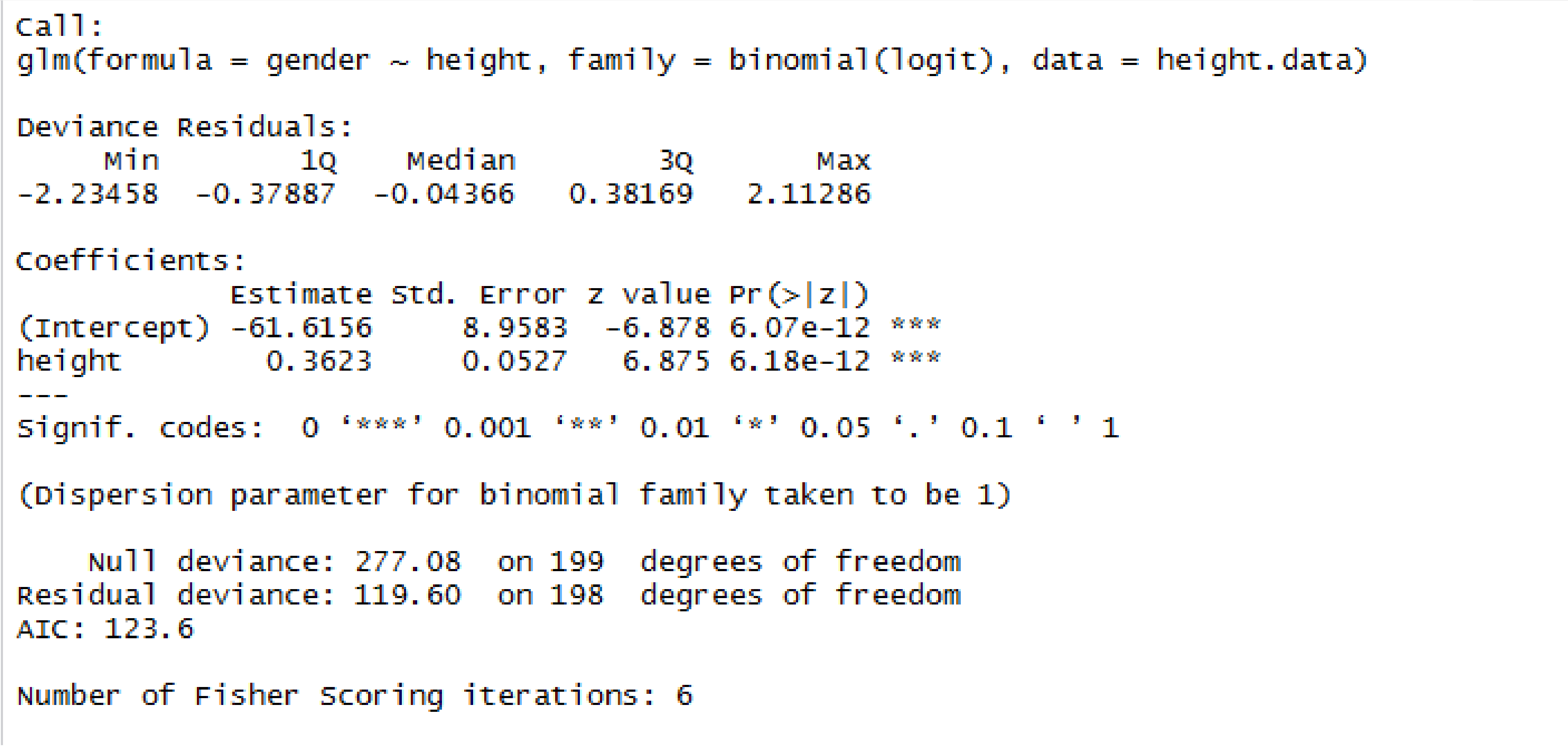


```
 Welch Two Sample t-test

data:  height by gender
t = -15.1484, df = 183.835, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -16.98571 -13.07106
sample estimates:
mean in group 0 mean in group 1 
       162.7922        177.8206 
```

- However, we are actually interested in **gender as a function of height**, for which we need to perform a logistic regression in `glm()`

Ok, here we go…

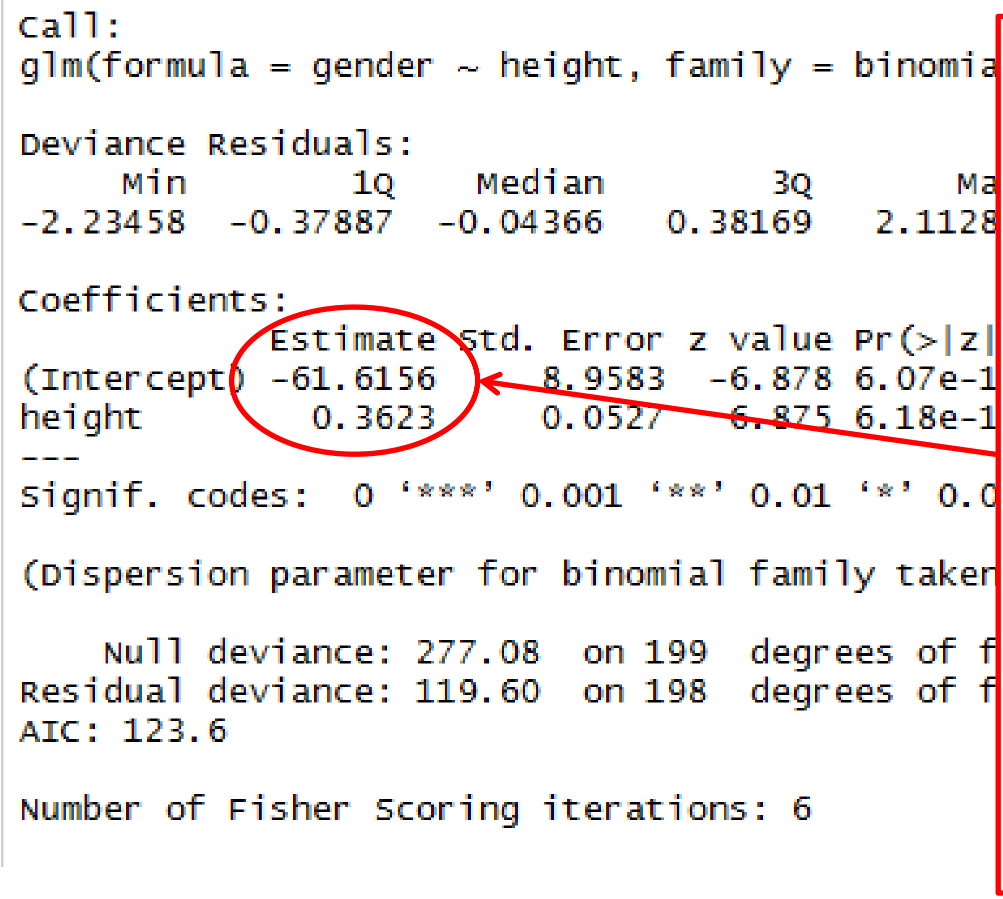
```r
# Performing the binary logistic glm()
glm.out <- glm(gender ~ height, family = binomial(logit),
               data=height.data)
```



```
Call:
glm(formula = gender ~ height, family = binomial(logit), data = height.data)

Deviance Residuals: 
     Min        1Q    Median        3Q       Max  
-2.23458  -0.37887  -0.04366   0.38169   2.11286  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) -61.6156     8.9583  -6.878 6.07e-12 ***
height        0.3623     0.0527   6.875 6.18e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.08  on 199  degrees of freedom
Residual deviance: 119.60  on 198  degrees of freedom
AIC: 123.6

Number of Fisher Scoring iterations: 6
```

Ok, here we go…

```r
# Performing the binary logistic glm()
glm.out <- glm(gender ~ height, family = binomial(logit),
               data=height.data)
```



```
Call:
glm(formula = gender ~ height, family = binomia

Deviance Residuals:
     Min        1Q    Median        3Q       Ma
-2.23458  -0.37887  -0.04366   0.38169   2.1128

Coefficients:
            Estimate Std. Error z value Pr(>|z|
(Intercept) -61.6156     8.9583  -6.878 6.07e-1
height        0.3623     0.0527   6.875 6.18e-1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.0

(Dispersion parameter for binomial family taken

    Null deviance: 277.08  on 199  degrees of f
Residual deviance: 119.60  on 198  degrees of f
AIC: 123.6

Number of Fisher Scoring iterations: 6
```

The all-important coefficient estimates (in **logit** units!)

Thus:

$$\mathit{log\ odds}\ (male) = -61.62 + 0.36 \times height$$

In other words, we can predict that for every 1 cm increase in height, the odds for being male increase by a factor of $exp(0.36) = 1.433$ times

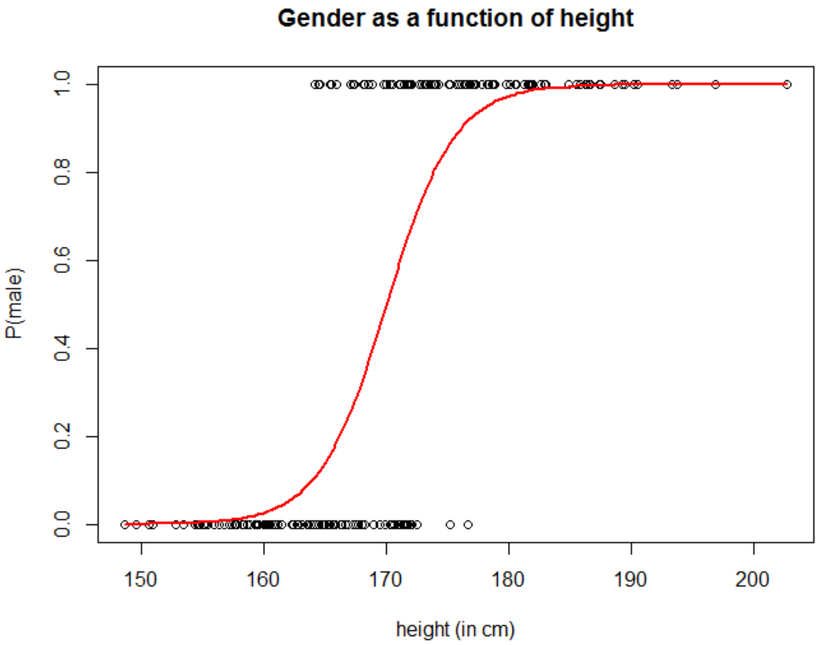
Intercept: At 0 cm body height, the likelihood of being male is

$$\frac{\exp(-61.616)}{1+\exp(-61.616)} = 1.74 \times 10^{-27}$$

(note: body-height is not mean-centred!)

Plot

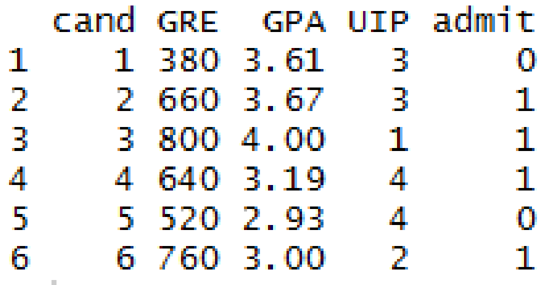
```
# Plot gender against height:
x <- height.data$height
y <- height.data$gender
plot (x, y,
      xlab = "height (in cm)",
      ylab = "P(male)")
title("Gender as a function of height")

# Create new data frame with variables x =
# height and y = fitted probabilities from
# glm output
NF <- data.frame(x=height.data$height,
                 y=glm.out$fitted)

# order the data in NF in increasing height
NF <- NF[order(NF$x),]

# draw a line using the x and y values in NF
lines(NF$x, NF$y, type='l', col='red', lwd=2)
```



A potentially more interesting example

- Fabricated data (shamelessly stolen from the internet)
- **Goal: Predict the likelihood of being admitted to graduate school (1 = admitted; 0 = not admitted) from a combination of 3 predictors:**
  - **GRE** (*Graduate Record Exam* scores, **continuous**)
  - **GPA** (*Grade Point Average*, **continuous**)
  - **UIP** (*Udergrad. Institution Prestige*, **rank scores** 1 “top notch” – 4 “meh”)
- Data from 400 imaginary “candidates”, each measured on all 4

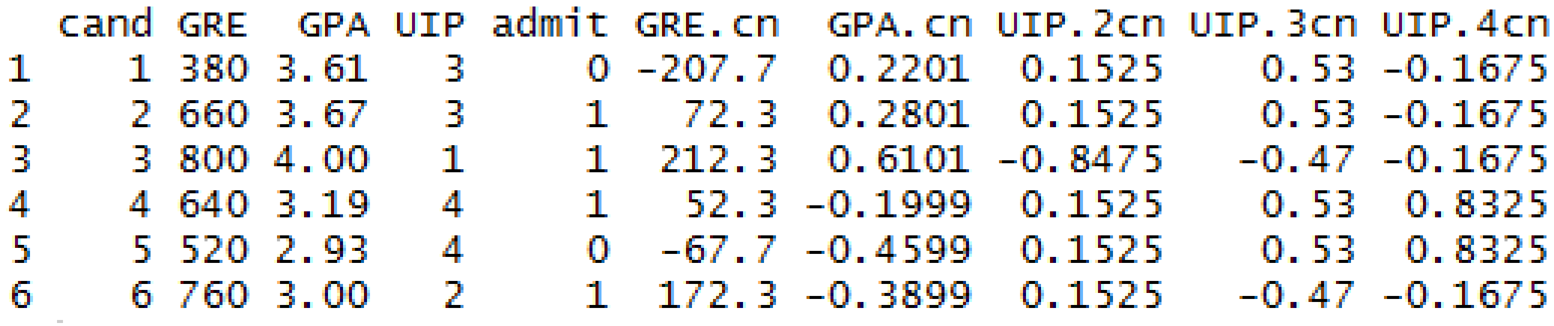
```
# Admission Data
AM.data <- read.csv("http://www.psy.gla.ac.uk/~christop/MScStats/2018/Logist/admission.csv")
head(AM.data)
```



```
  cand GRE  GPA UIP admit
1    1 380 3.61   3     0
2    2 660 3.67   3     1
3    3 800 4.00   1     1
4    4 640 3.19   4     1
5    5 520 2.93   4     0
6    6 760 3.00   2     1
```

Predictor coding

```
# First, mean centre the continuous predictors GRE and GPA
AM.data$GRE.cn <- AM.data$GRE - mean(AM.data$GRE)
AM.data$GPA.cn <- AM.data$GPA - mean(AM.data$GPA)

# Treat UIP as ORDINAL predictor
# mean-centred forward-difference coding (higher means “worse”),
# UIP=1 (“best”) serves as reference
AM.data$UIP.2cn <- scale(ifelse(AM.data$UIP < 2,1,0), scale = FALSE)
AM.data$UIP.3cn <- scale(ifelse(AM.data$UIP < 3,1,0), scale = FALSE)
AM.data$UIP.4cn <- scale(ifelse(AM.data$UIP < 4,1,0), scale = FALSE)
# (note that UIP scores are not evenly distributed in the sample)

head(AM.data)
```



```
  cand GRE  GPA UIP admit GRE.cn  GPA.cn UIP.2cn UIP.3cn UIP.4cn
1    1 380 3.61   3     0 -207.7  0.2201  0.1525    0.53 -0.1675
2    2 660 3.67   3     1   72.3  0.2801  0.1525    0.53 -0.1675
3    3 800 4.00   1     1  212.3  0.6101 -0.8475   -0.47 -0.1675
4    4 640 3.19   4     1   52.3 -0.1999  0.1525    0.53  0.8325
5    5 520 2.93   4     0  -67.7 -0.4599  0.1525    0.53  0.8325
6    6 760 3.00   2     1  172.3 -0.3899  0.1525   -0.47 -0.1675
```

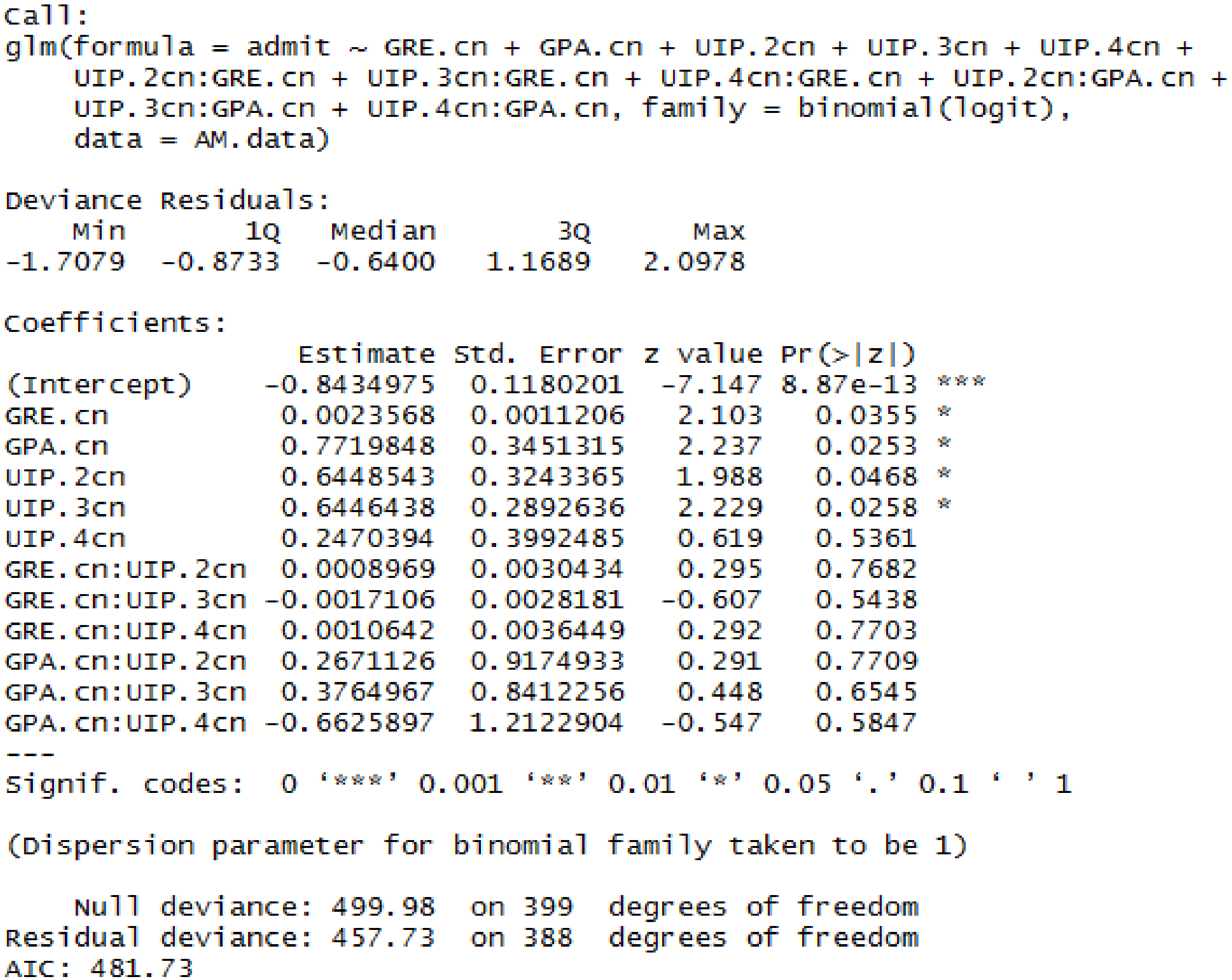
Run binary logistic `glm()`

- Let’s assume we were primarily interested in the ***main effects*** of the three predictors and in ***2-way interactions*** between UIP (ordinal predictor) and each of the two continuous predictors (GRE and GPA)

```
# Run glm()
AM.model <- glm(admit ~
                  # main effect terms:
                  GRE.cn + GPA.cn + UIP.2cn + UIP.3cn + UIP.4cn +
                  # 2-way interaction terms (effect of GRE per level of UIP):
                  UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn +
                  # 2-way interaction terms (effect of GPA per level of UIP):
                  UIP.2cn:GPA.cn + UIP.3cn:GPA.cn + UIP.4cn:GPA.cn,
                data = AM.data,
                family = binomial(logit))
```

Results

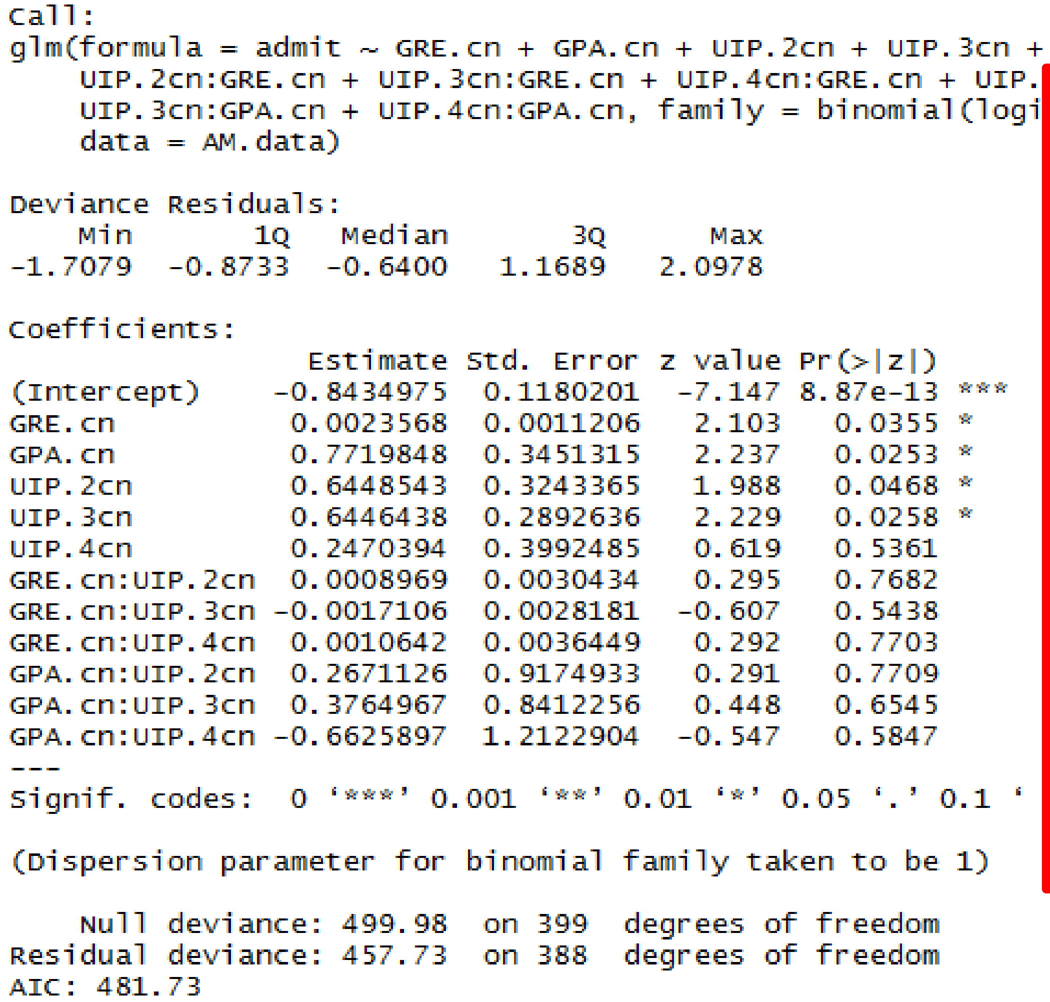
```r
# Output
summary(AM.model)
```



```
Call:
glm(formula = admit ~ GRE.cn + GPA.cn + UIP.2cn + UIP.3cn + UIP.4cn +
    UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn + UIP.2cn:GPA.cn +
    UIP.3cn:GPA.cn + UIP.4cn:GPA.cn, family = binomial(logit),
    data = AM.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7079  -0.8733  -0.6400   1.1689   2.0978

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)    -0.8434975  0.1180201  -7.147 8.87e-13 ***
GRE.cn          0.0023568  0.0011206   2.103   0.0355 *
GPA.cn          0.7719848  0.3451315   2.237   0.0253 *
UIP.2cn         0.6448543  0.3243365   1.988   0.0468 *
UIP.3cn         0.6446438  0.2892636   2.229   0.0258 *
UIP.4cn         0.2470394  0.3992485   0.619   0.5361
GRE.cn:UIP.2cn  0.0008969  0.0030434   0.295   0.7682
GRE.cn:UIP.3cn -0.0017106  0.0028181  -0.607   0.5438
GRE.cn:UIP.4cn  0.0010642  0.0036449   0.292   0.7703
GPA.cn:UIP.2cn  0.2671126  0.9174933   0.291   0.7709
GPA.cn:UIP.3cn  0.3764967  0.8412256   0.448   0.6545
GPA.cn:UIP.4cn -0.6625897  1.2122904  -0.547   0.5847
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 457.73  on 388  degrees of freedom
AIC: 481.73
```

Results

```
# Output
summary(AM.model)
```



```
Call:
glm(formula = admit ~ GRE.cn + GPA.cn + UIP.2cn + UIP.3cn + UIP.4cn +
    UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn + UIP.
    UIP.3cn:GPA.cn + UIP.4cn:GPA.cn, family = binomial(logi
    data = AM.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7079  -0.8733  -0.6400   1.1689   2.0978

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)    -0.8434975  0.1180201  -7.147 8.87e-13 ***
GRE.cn          0.0023568  0.0011206   2.103   0.0355 *
GPA.cn          0.7719848  0.3451315   2.237   0.0253 *
UIP.2cn         0.6448543  0.3243365   1.988   0.0468 *
UIP.3cn         0.6446438  0.2892636   2.229   0.0258 *
UIP.4cn         0.2470394  0.3992485   0.619   0.5361
GRE.cn:UIP.2cn  0.0008969  0.0030434   0.295   0.7682
GRE.cn:UIP.3cn -0.0017106  0.0028181  -0.607   0.5438
GRE.cn:UIP.4cn  0.0010642  0.0036449   0.292   0.7703
GPA.cn:UIP.2cn  0.2671126  0.9174933   0.291   0.7709
GPA.cn:UIP.3cn  0.3764967  0.8412256   0.448   0.6545
GPA.cn:UIP.4cn -0.6625897  1.2122904  -0.547   0.5847
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 457.73  on 388  degrees of freedom
AIC: 481.73
```

Intercept (mean-centred predictors!): The **overall likelihood of admission** is *exp(-0.843) / (1 + exp(-0.843))* ≈ 0.30

Main effect of GRE (higher GRE means increase in admission likelihood)

Main effect of GPA (higher GPA means increase in admission likelihood)

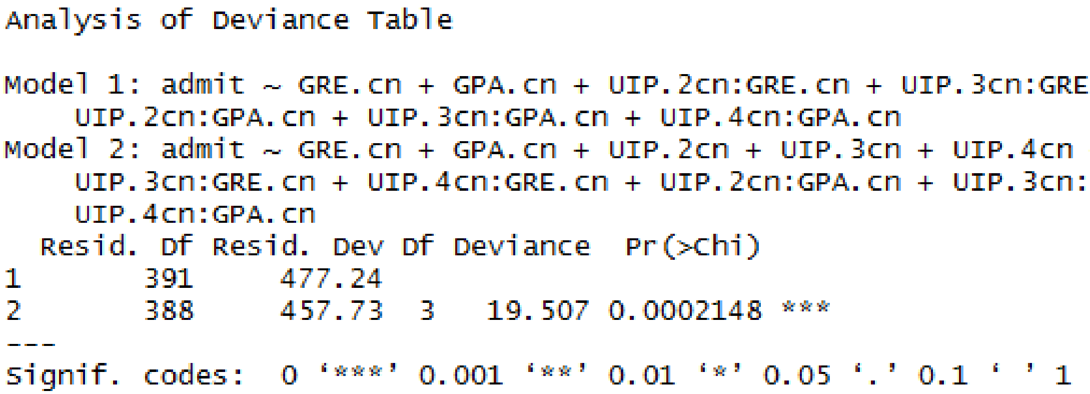
UIP main effect (smaller[="better"] UIP scores mean higher admission likelihood, except UIP=3 vs. UIP=4)

No significant interaction terms

LRχ^2 tests

- Say, we were interested in the **main effect of UIP** (4-level ordinal predictor):

```
# Run a glm() without UIP main effect parameters
AM.noUIP <- glm(admit ~
                  GRE.cn + GPA.cn + # UIP.2cn + UIP.3cn + UIP.4cn +
                  UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn +
                  UIP.2cn:GPA.cn + UIP.3cn:GPA.cn + UIP.4cn:GPA.cn,
                data = AM.data,
                family = binomial(logit))

# Compare with previous model
anova(AM.noUIP, AM.model, test="Chi")
```

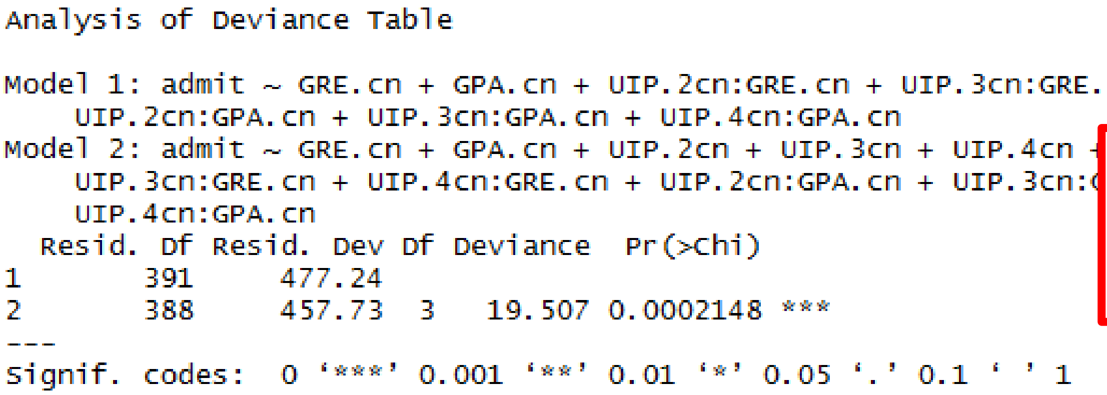


```
Analysis of Deviance Table

Model 1: admit ~ GRE.cn + GPA.cn + UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn +
    UIP.2cn:GPA.cn + UIP.3cn:GPA.cn + UIP.4cn:GPA.cn
Model 2: admit ~ GRE.cn + GPA.cn + UIP.2cn + UIP.3cn + UIP.4cn + UIP.2cn:GRE.cn +
    UIP.3cn:GRE.cn + UIP.4cn:GRE.cn + UIP.2cn:GPA.cn + UIP.3cn:GPA.cn +
    UIP.4cn:GPA.cn
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)    
1       391     477.24                          
2       388     457.73  3   19.507 0.0002148 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

LRχ^2 tests

- Say, we were interested in the **main effect of UIP** (4-level ordinal predictor):

```
# Run a glm() without UIP main effect parameters
AM.noUIP <- glm(admit ~
                  GRE.cn + GPA.cn + # UIP.2cn + UIP.3cn + UIP.4cn +
                  UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn +
                  UIP.2cn:GPA.cn + UIP.3cn:GPA.cn + UIP.4cn:GPA.cn,
                data = AM.data,
                family = binomial(logit))

# Compare with previous model
anova(AM.noUIP, AM.model, test="Chi")
```



```
Analysis of Deviance Table

Model 1: admit ~ GRE.cn + GPA.cn + UIP.2cn:GRE.cn + UIP.3cn:GRE.cn + UIP.4cn:GRE.cn +
    UIP.2cn:GPA.cn + UIP.3cn:GPA.cn + UIP.4cn:GPA.cn
Model 2: admit ~ GRE.cn + GPA.cn + UIP.2cn + UIP.3cn + UIP.4cn +
    UIP.3cn:GRE.cn + UIP.4cn:GRE.cn + UIP.2cn:GPA.cn + UIP.3cn:
    UIP.4cn:GPA.cn
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       391     477.24
2       388     457.73  3   19.507 0.0002148 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Thus:
The main effect of UIP is significant:
$LR\chi^2 = 19.507$, $df = 3$, $p < .001$

Analysing rank data using ordinal logistic regression

Christoph Scheepers

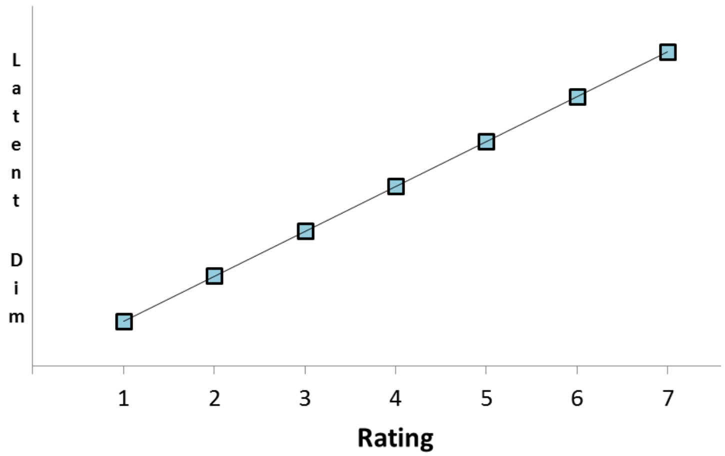
christoph.scheepers@glasgow.ac.uk

Ordinal data

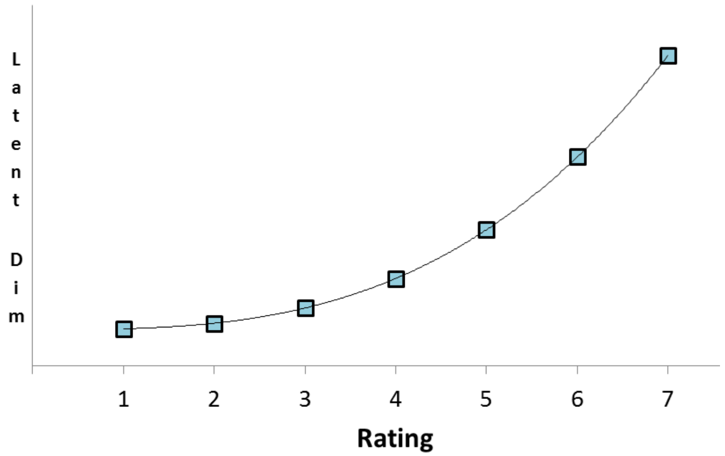
- Are data from which one can reasonably only make assertions about **relative rankings** of observations along some latent dimension (e.g., "trustworthyness", "attractiveness", "degree of agreement", "grammaticality", "plausibility", etc.)
- **Rating data** (e.g., Likert scales) typically fall into this category
- Analysing such data using standard linear models (t-test, ANOVA, linear mixed models) often leads to inaccurate inferences (Type I and Type II errors) due to
  - Violation of linear modelling assumptions (normality, homoskedasticity)
  - Ignoring that ratings are bounded between a minimum and a maximum
  - Ascribing more information to the data than the scale actually supports
    - a mean of *2.734* refers to an observation that cannot possibly be made on a 1-2-3-4-5 scale
    - equal differences on the rating scale do not necessarily mean equal differences in the property being measured

Ordinal data

- Perhaps the biggest theoretical problem with ratings is that we don't know their relation to the latent dimension of interest (can even vary across subjects and trials)



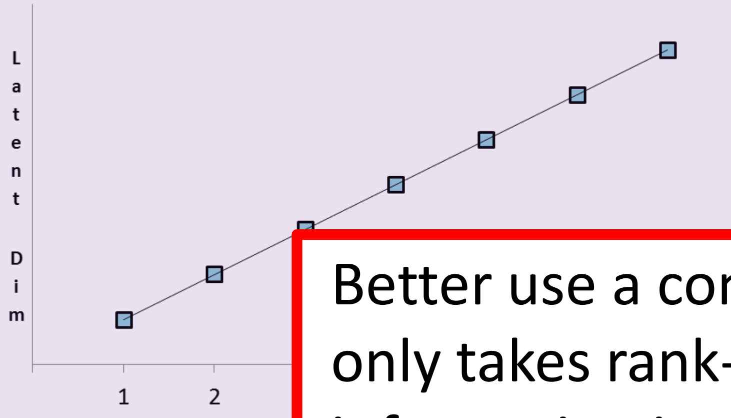
- **Close to linear** = equal distances on the scale mean (roughly) equal distances in the property being measured; differences on the scale can be ranked (=> **ordered metric scale**)



- **Non-linear** = equal distances on the scale do not mean equal distances in the property being measured; differences on the scale can not be ranked

Ordinal data

- Perhaps the biggest theoretical problem with ratings is that we don’t know their relation to the latent dimension of interest (can even vary across subjects and trials)

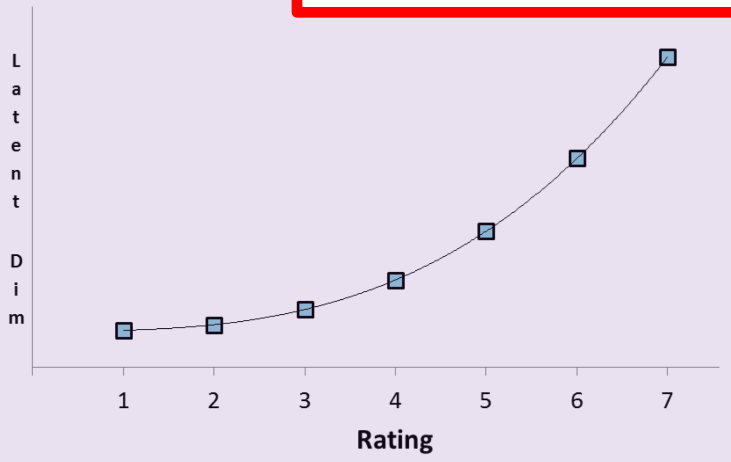


- **Close to linear** = equal distances on the scale mean (roughly) equal distances in the property being

es on the

(=> **ordered**

Better use a conservative approach that only takes rank- and distributional information into account



- **Non-linear** = equal distances on the scale do not mean equal distances in the property being measured; differences on the scale can not be ranked

Ordinal data modelling

- Fortunately, a variety of different R packages (but see also GEE/GLMM in SPSS) have emerged in recent years that allow for appropriate modelling of ordinal data
- There's little excuse for using procedures relying on calculation of means (t-test, ANOVA, etc.) anymore
- Non-parametric tests like Mann-Witney U, Wilcoxon signed-ranks test, etc., are equally bound to become historic side-notes
    - Can't model complex (factorial) designs
    - Not very accurate
    - Not very flexible/powerful in accounting for repeated-measures dependencies
    - Do not allow for simultaneous generalization of findings across subjects and items

`ordinal` Package

- Here, I will primarily focus on **cumulative models** as implemented in the R package `ordinal` (Christensen, 2018) which uses a frequentist approach to inferencing and is comparable to `glm()` (base R) and `lme4` (mixed effects modelling)
- Comes with a superb vignette and tutorial

Ordinal logistic regression

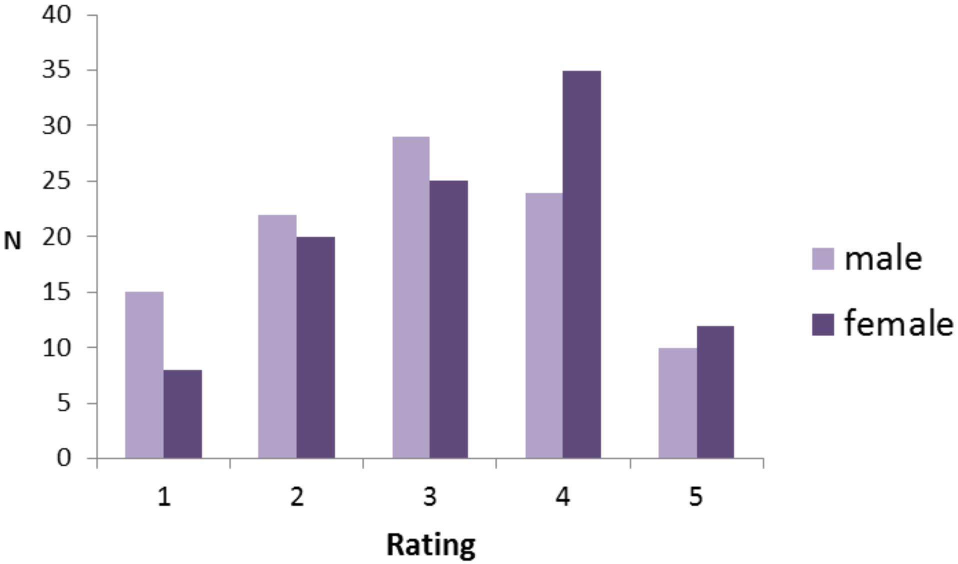
- `ordinal` package: An Implementation of *cumulative link* (mixed) models also known as *ordered regression models*, *proportional odds models*, *proportional hazards models* for grouped survival times and *ordered logit/probit/...* models.
- Mathematically, part of the generalized linear model family
- In essence, modelling ***scale-point occurrences* in terms of a GLM** assuming a *multinomial distribution* and a *cumulative link* (logit, probit, cauchit, loglog, or cloglog)
- I will henceforth call it *ordinal logistic regression,* as I will only use the logit link here
- Two functions
  - `clm()` - comparable to `glm()` in base R (independent measures!)
  - `clmm()` – comparable to `(g)lmer` in `lme4` (mixed models)

An illustrative (fake) example

- Let’s assume we asked 100 male and 100 female participants to rate the attractiveness of a photo of Dr. Dale Barr (right) on a scale from 1-5

not attractive [1]--------[2]-------[3]-------[4]-------[5] *very attractive*

- Every subject (N=200) provides only one rating, and we are interested whether males and females differ in their judgements
- Let’s assume the **distribution of ratings** looks like this **->**

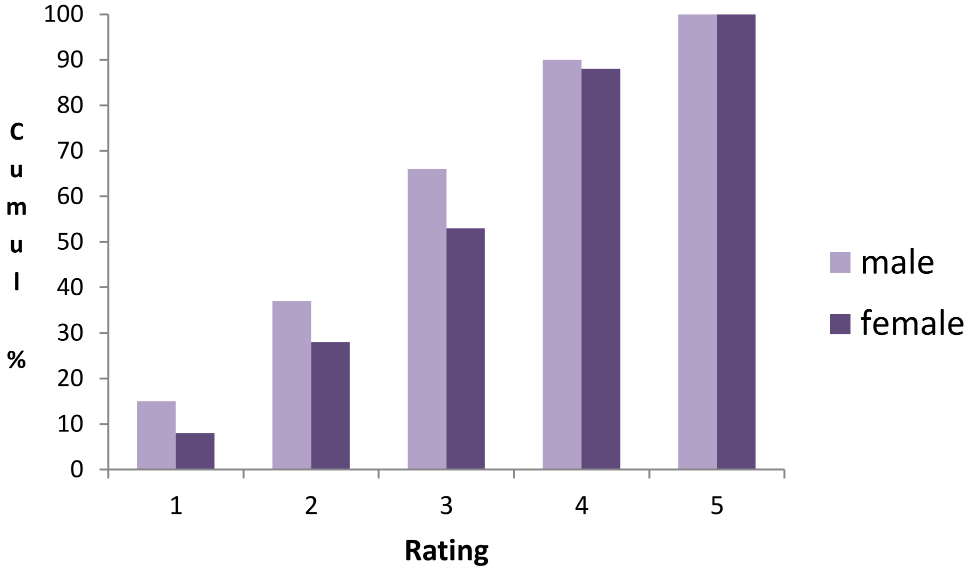


An illustrative (fake) example

- Let’s assume we asked 100 male and 100 female participants to rate the attractiveness of a photo of Dr. Dale Barr (right) on a scale from 1-5

not attractive [1]--------[2]-------[3]-------[4]-------[5] *very attractive*

- The **cumulative percentages** would look like this **->**

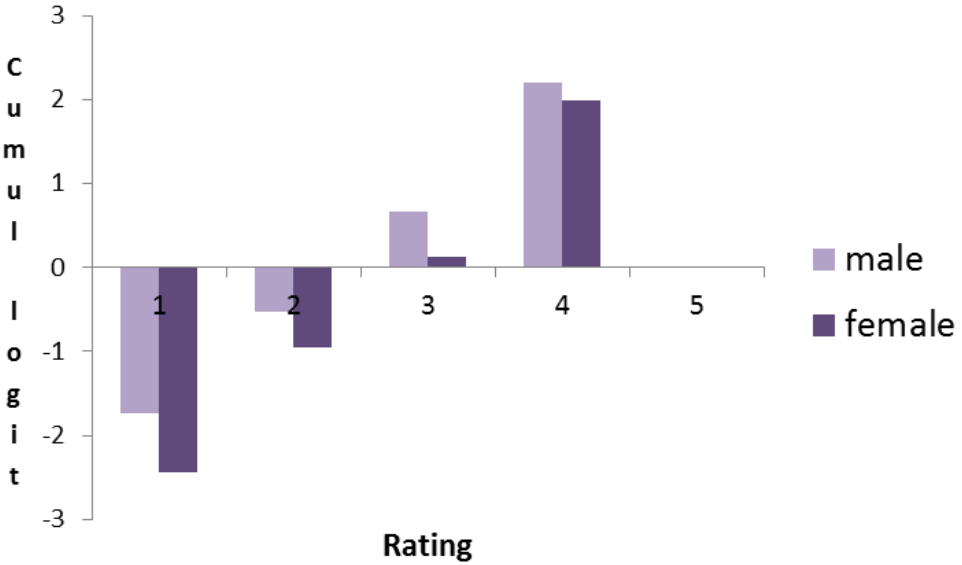


An illustrative (fake) example

- Let’s assume we asked 100 male and 100 female participants to rate the attractiveness of a photo of Dr. Dale Barr (right) on a scale from 1-5

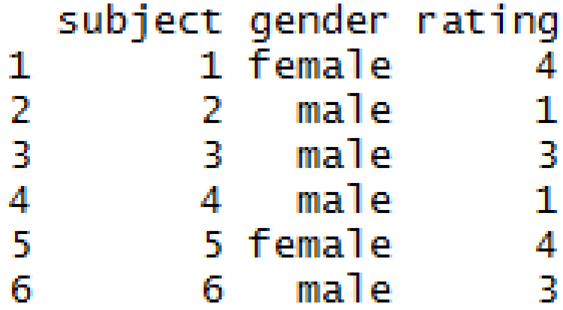
not attractive [1]--------[2]-------[3]-------[4]-------[5] *very attractive*

- And the **log odds** of the cumulative percentages (“**cumulative logits**”) would look like this **->**
- **That’s what is being modelled in ordinal logistic regression**



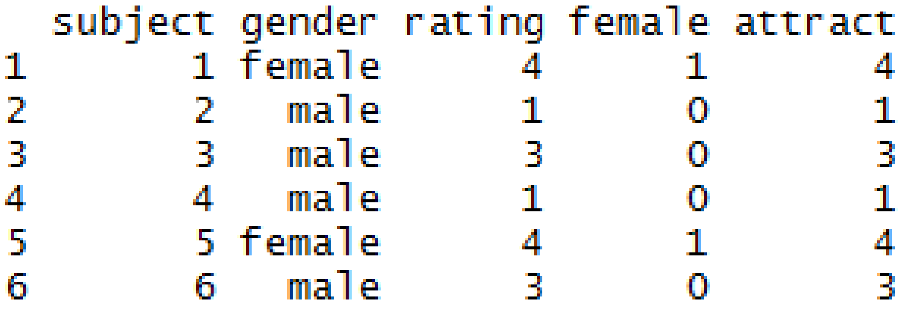
Let’s analyse in R

```
# get the data (included in web-folder)
daleratings <- read.csv("dalerate.csv")

head(daleratings)
```



```
  subject gender rating
1       1 female      4
2       2   male      1
3       3   male      3
4       4   male      1
5       5 female      4
6       6   male      3
```

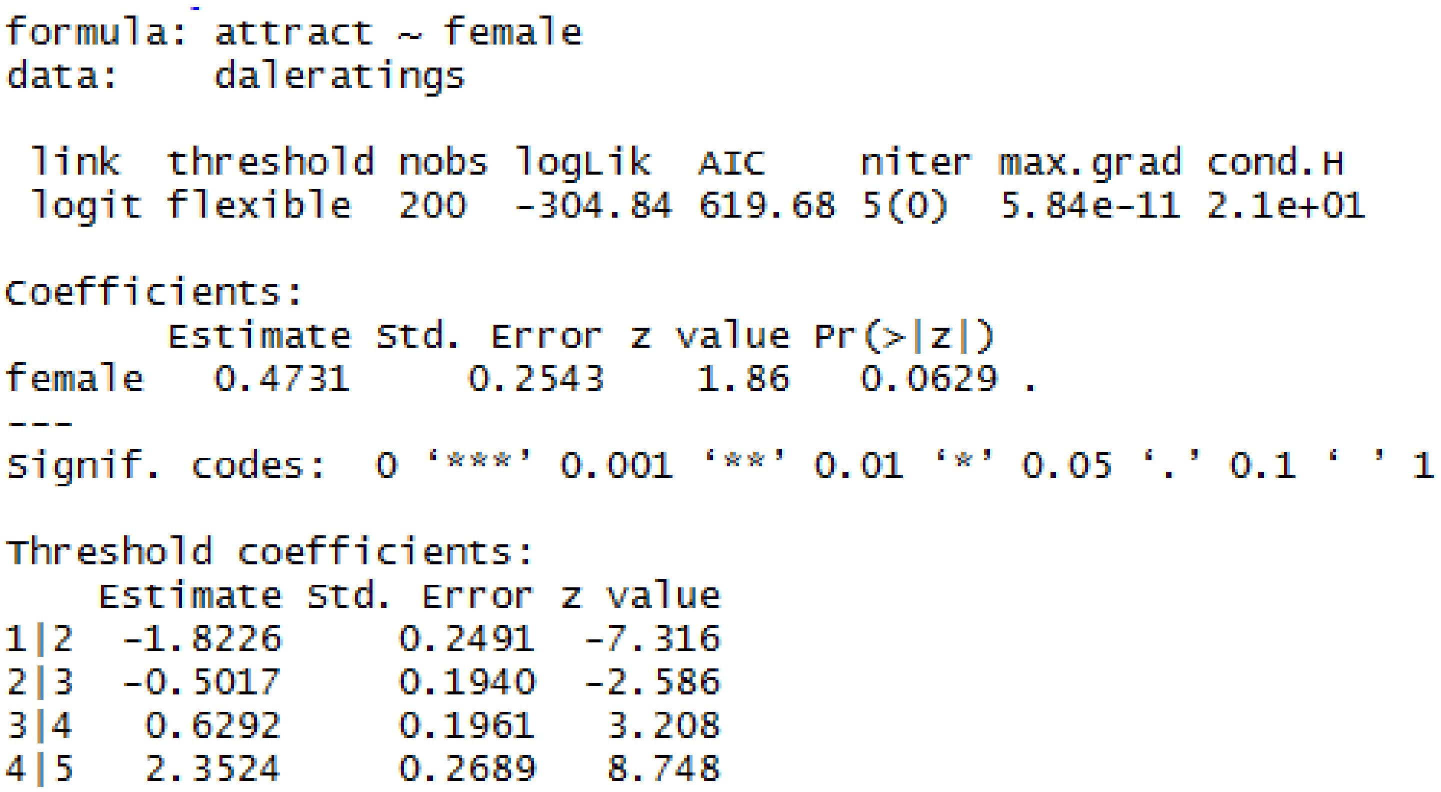
```
# Code predictor (gender); dummy coding should suffice (simple 1-way design)
daleratings$female <- ifelse(daleratings$gender=="female",1,0)

# IMPORTANT: turn DV into a factor!
daleratings$attract <- factor(daleratings$rating)

head(daleratings)
```



```
  subject gender rating female attract
1       1 female      4      1       4
2       2   male      1      0       1
3       3   male      3      0       3
4       4   male      1      0       1
5       5 female      4      1       4
6       6   male      3      0       3
```

Let’s analyse in R

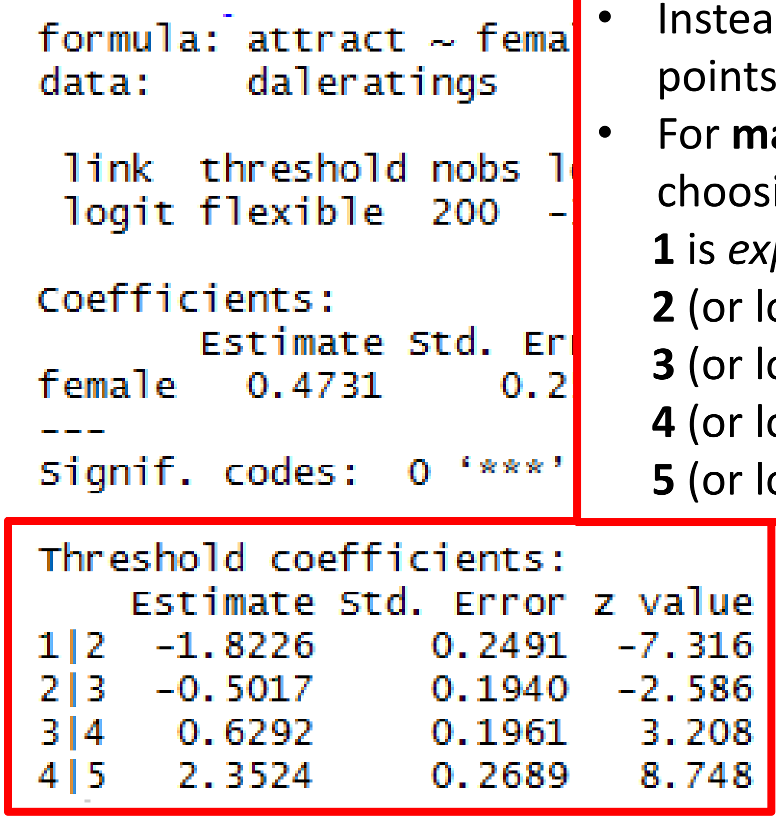
```r
# load ordinal package
library(ordinal)

# run ordinal logistic model
ordinal.mod <- clm(attract ~ female, data = daleratings)
summary(ordinal.mod)
```



```
formula: attract ~ female
data:    daleratings

 link  threshold nobs logLik  AIC    niter max.grad cond.H
 logit flexible  200  -304.84 619.68 5(0)  5.84e-11 2.1e+01

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
female   0.4731     0.2543    1.86   0.0629 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Threshold coefficients:
    Estimate Std. Error z value
1|2  -1.8226     0.2491  -7.316
2|3  -0.5017     0.1940  -2.586
3|4   0.6292     0.1961   3.208
4|5   2.3524     0.2689   8.748
```

Things to note

```r
# load ordinal package
library(ordinal)

# run ordinal logistic model
ordinal.mod <- clm(attract ~ female, data = daleratings)
summary(ordinal.mod)
```



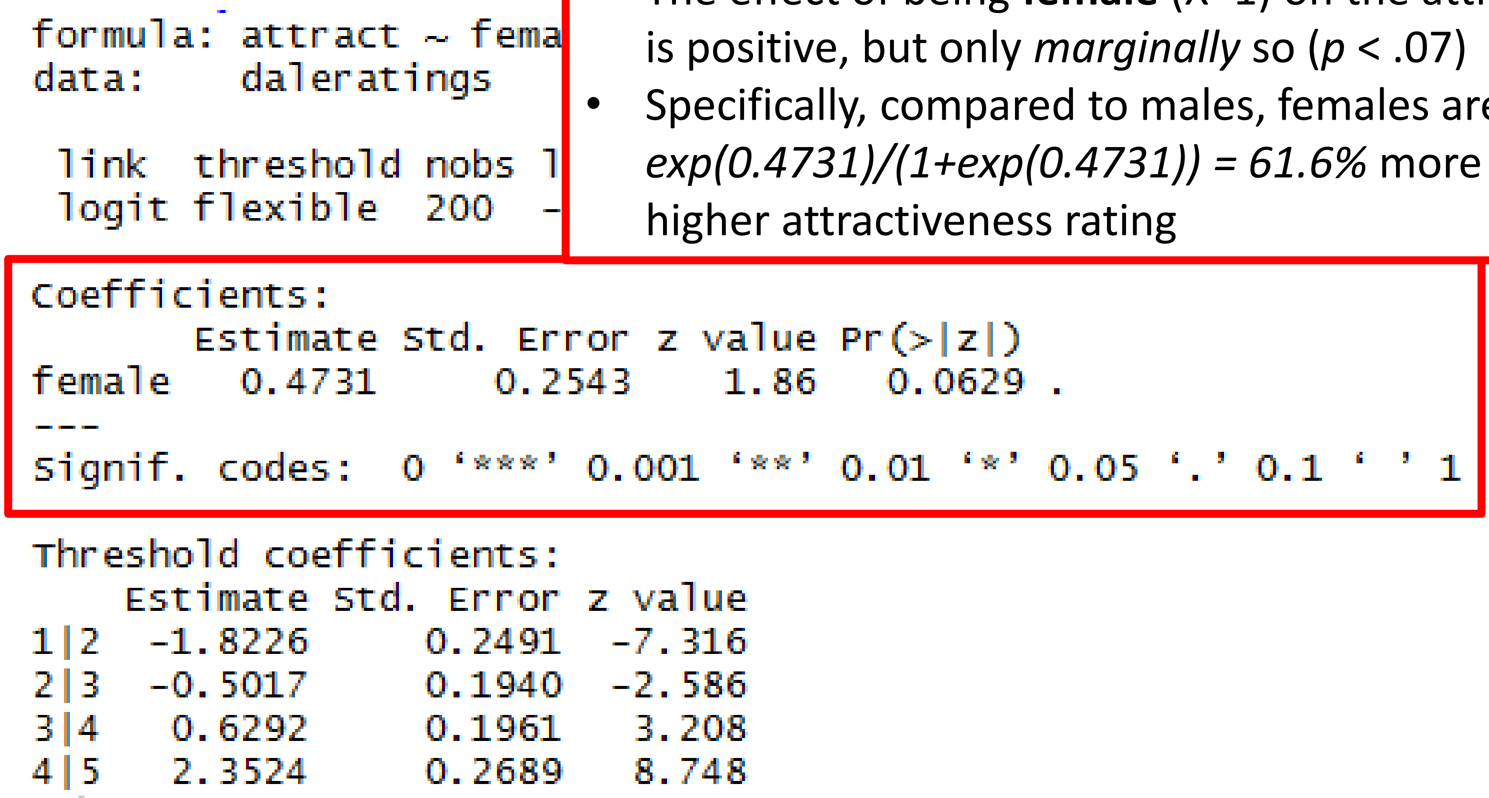
```
formula: attract ~ fema
data:    daleratings

 link  threshold nobs l
 logit flexible  200  -

Coefficients:
       Estimate Std. Er
female   0.4731     0.2
---
Signif. codes:  0 '***'

Threshold coefficients:
    Estimate Std. Error z value
1|2  -1.8226     0.2491  -7.316
2|3  -0.5017     0.1940  -2.586
3|4   0.6292     0.1961   3.208
4|5   2.3524     0.2689   8.748
```

- Instead of a single intercept, one obtains *K* (number of scale points) – 1 cumulative logit **'threshold' coefficients**
- For **males** (X = 0), the estimated cumulative probability of choosing scale point …

  1 is *exp(-1.8226)/(1+exp(-1.8226))* = 0.139 (~14%),
  2 (or lower) is *exp(-0.5017)/(1+exp(-0.5017))* = 0.377 (~38%),
  3 (or lower) is *exp(0.6292)/(1+exp(0.6292))* = 0.652 (~65%),
  4 (or lower) is *exp(2.3524)/(1+exp(2.3524))* = 0.913 (~91%)
  5 (or lower) is 1 (100%)

Things to note

```
# load ordinal package
library(ordinal)

# run ordinal logistic model
ordinal.mod <- clm(attract ~ female, data = daleratings)
summary(ordinal.mod)
```



```
formula: attract ~ fema
data:    daleratings

 link  threshold nobs l
 logit flexible  200  -

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
female   0.4731     0.2543    1.86   0.0629 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Threshold coefficients:
    Estimate Std. Error z value
1|2  -1.8226     0.2491  -7.316
2|3  -0.5017     0.1940  -2.586
3|4   0.6292     0.1961   3.208
4|5   2.3524     0.2689   8.748
```

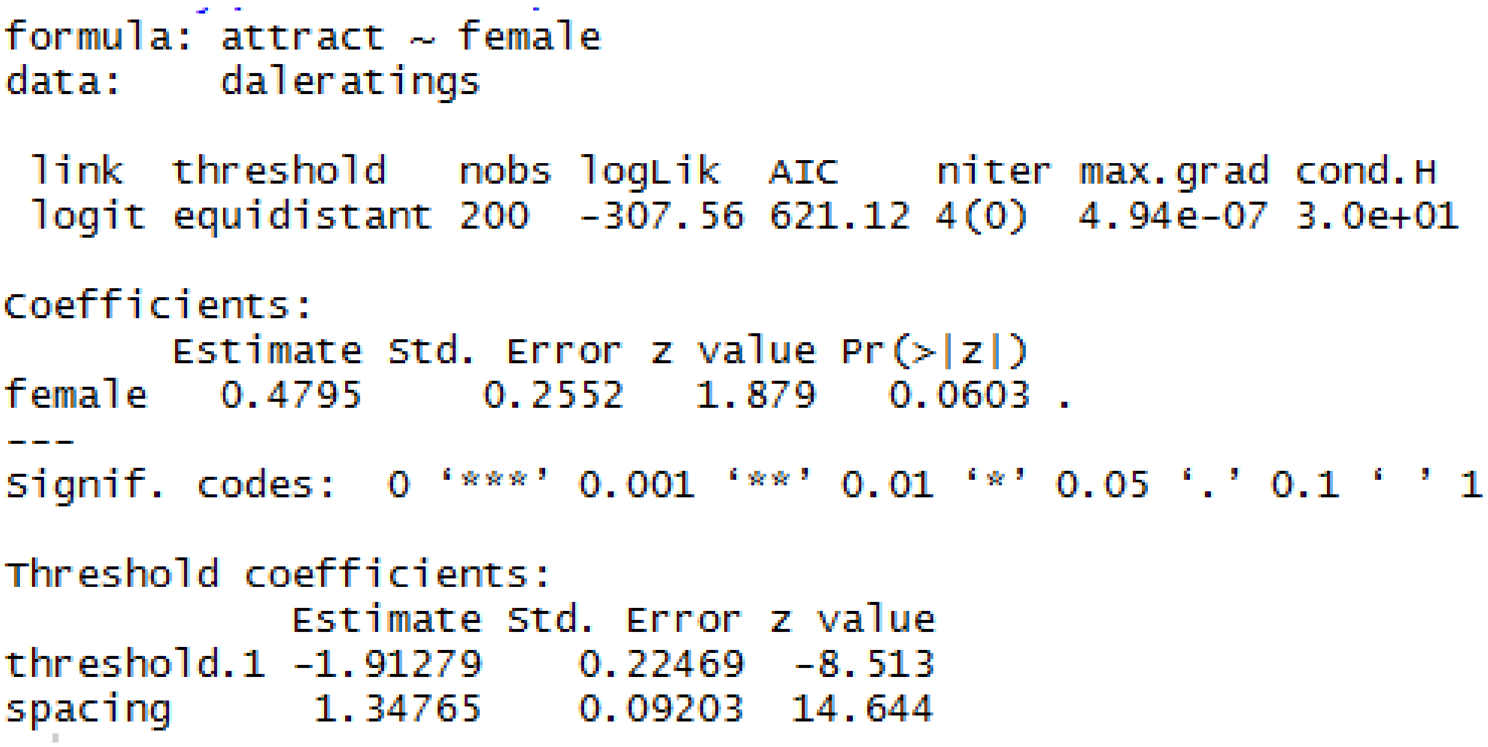
- The effect of being **female** (X=1) on the attractiveness ratings is positive, but only *marginally* so ($p < .07$)
- Specifically, compared to males, females are *exp(0.4731)/(1+exp(0.4731)) = 61.6%* more likely to choose a higher attractiveness rating

Different threshold assumptions

- Via the ***threshold* argument**, it is possible to change assumptions about ‘spacing’ of scale categories:
  - `threshold=“flexible”` (default): each scale category (minus the highest one) gets its own cumulative logit threshold
  - `threshold=“equidistant”`: scale points are assumed to be **evenly spaced**
  - `threshold=“symmetric”`: scale points are assumed to be **evenly spaced below/above scale centre** (apparently a good choice in ‘polarised’ scales like 1 = very <u>un</u>attractive – 7 = very attractive)
- Apart from requiring fewer parameters, the *symmetric* and *equidistant* options make stronger assumptions about the scale
  - Requires *theoretical justification* and/or AIC model comparison (see example at the end of next session)
  - As always, **do not “shop around” for settings that give you the best p-values!**

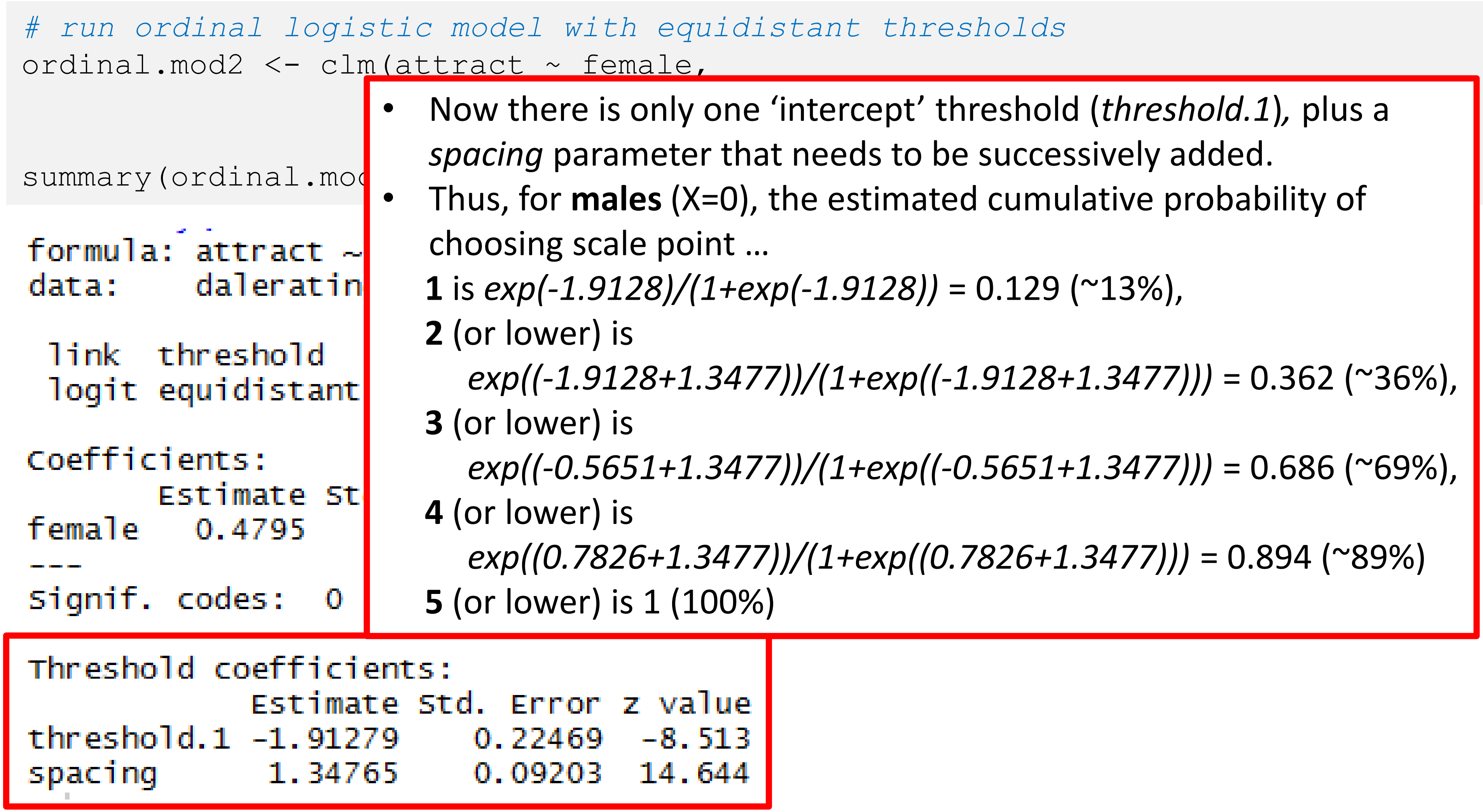
E.g., equidistant thresholds

```
# run ordinal logistic model with equidistant thresholds
ordinal.mod2 <- clm(attract ~ female,
                    data = daleratings,
                    threshold = "equidistant")
summary(ordinal.mod2)
```



```
formula: attract ~ female
data:    daleratings

 link  threshold   nobs logLik  AIC    niter max.grad cond.H
 logit equidistant 200  -307.56 621.12 4(0)  4.94e-07 3.0e+01

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
female   0.4795     0.2552   1.879   0.0603 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Threshold coefficients:
            Estimate Std. Error z value
threshold.1 -1.91279    0.22469  -8.513
spacing      1.34765    0.09203  14.644
```

E.g., equidistant thresholds



```
# run ordinal logistic model with equidistant thresholds
ordinal.mod2 <- clm(attract ~ female,


summary(ordinal.mod
```

```
formula: attract ~
data:    daleratin

 link  threshold
 logit equidistant

Coefficients:
       Estimate St
female   0.4795
---
Signif. codes:  0

Threshold coefficients:
            Estimate Std. Error z value
threshold.1 -1.91279    0.22469  -8.513
spacing      1.34765    0.09203  14.644
```

- Now there is only one ‘intercept’ threshold (*threshold.1*), plus a *spacing* parameter that needs to be successively added.
- Thus, for **males** (X=0), the estimated cumulative probability of choosing scale point …

  1 is $exp(-1.9128)/(1+exp(-1.9128))$ = 0.129 (~13%),

  2 (or lower) is
  $exp((-1.9128+1.3477))/(1+exp((-1.9128+1.3477)))$ = 0.362 (~36%),

  3 (or lower) is
  $exp((-0.5651+1.3477))/(1+exp((-0.5651+1.3477)))$ = 0.686 (~69%),

  4 (or lower) is
  $exp((0.7826+1.3477))/(1+exp((0.7826+1.3477)))$ = 0.894 (~89%)

  5 (or lower) is 1 (100%)

Summary

- **Generalized Linear Models** (`glm()`; see also `clm()` in ordinal package) are an extension/generalization of standard linear models (`lm()`)
  - `lm()` is in fact a special case of `glm()`
- Same principles in terms of predictor coding & model formulae
- **`family`** argument in `glm()` allows for the specification of distribution and link functions appropriate for modelling non-normally distributed DVs and/or non-linear relationships in the data
- Examples: Gamma regression, binary and ordinal logistic regression
- **Important:** `lm()`, `glm()`, and `clm()` assume **independent-measures** data (one observation per variable per sampling unit)
- To model **repeated-measures data** (multiple observations per variable per sampling unit), we need to extend things even further
  - Generalized Estimating Equations (GEE)
  - **Generalized Linear Mixed Models (GLMMs)**