Lecture 1: The Generalized Linear Model

LSA 2013, LI539

Mixed Effect Models

T. Florian Jaeger

Brain and Cognitive Sciences
University of Rochester

July 11, 2013

Class goals

- This course provides an introduction to Generalized Linear Model (GLM) and Generalized Linear Mixed Model (GLMM)
  - Mathematical backgroung
  - Intuition and conceptualization
  - Geometical interpretation
  - Common issues and solutions for GLM and GLMM analyses
  - Relation to ANOVA
- We will learn
  - how to conduct, interpret and report GLM and GLMM analyses in R
  - how to visualize data in R
  - how to prepare data for visualization and analysis (transformation)

  - The course will be part lecture, part learning by doing.

Course overview

Lecture 1:

- (re-)introducing **Generalized Linear Models (GLM)**
- relation between GLM and **ANOVA**
- example (linear) models and geometric interpretation

Course overview

Lecture 1:

- (re-)introducing **Generalized Linear Models (GLM)**
- relation between GLM and **ANOVA**
- example (linear) models and geometric interpretation

Lecture 2:

- **Generalized Linear Mixed Models (GLMM)**
- relation between GLMM and ANOVA
- random effects
- example models

Course overview

Lecture 3: Beyond linear models

- Binomial models (logistic regression and mixed logit models)
- Empirical logit weighted linear regression
- Poisson models

Course overview

Lecture 3: Beyond linear models

- Binomial models (logistic regression and mixed logit models)
- Empirical logit weighted linear regression
- Poisson models

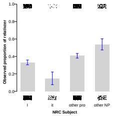
Lecture 4: Tools for data analysis, exploration, and transformation

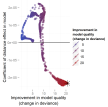
- Being able to summarize and understand your data is crucial (perhaps more important than knowing fancy models)
  - library `plyr`
  - library `reshape2`

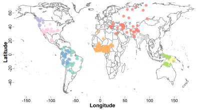
Course overview

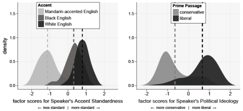
Lecture 5: Visualizing and summarizing your data

- library `ggplot2`





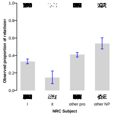


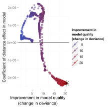


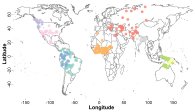
Course overview

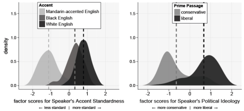
Lecture 5: Visualizing and summarizing your data

- library `ggplot2`









- library `stargazer`
- library `knitr`

Table: Example Stargazer table generated from R

	Dependent variable:		
	(logged) RT		Correct response?
	OLS		*logistic*
	(1)	(2)	(3)
Intercept	6.497^{***}	6.466^{***}	1.664^{**}
	(0.030)	(0.028)	(0.666)
Word frequency (logged)	-0.031^{***}	-0.031^{***}	0.412^{***}
	(0.006)	(0.006)	(0.154)
Native language	0.285^{***}	0.286^{***}	-1.642^{*}
	(0.042)	(0.042)	(0.886)
Trial position	-0.0003^{**}		
	(0.0001)		
Word frequency (logged):Native language	-0.027^{***}	-0.027^{***}	0.261
	(0.009)	(0.009)	(0.212)
Observations	1,659	1,659	1,659
R^2	0.161	0.158	
Adjusted R^2	0.159	0.157	
Akaike Inf. Crit.			520.100
Note:			*p<0.1; **p<0.05; ***p<0.01

Course overview

Lecture 6 and 7: Common issues and solutions in GLMs and GLMMs

- collinearity
- model evaluation
- over-fitting
- non-linearities

Course overview

Lecture 6 and 7: Common issues and solutions in GLMs and GLMMs

- collinearity
- model evaluation
- over-fitting
- non-linearities

Lecture 8: Remaining issues and continued discussion

- Reporting GLMMs in your article

Oh, we are all so different ...

Folks in this class represent varied linguistic interest and varied degrees of expertise in statistics and R.

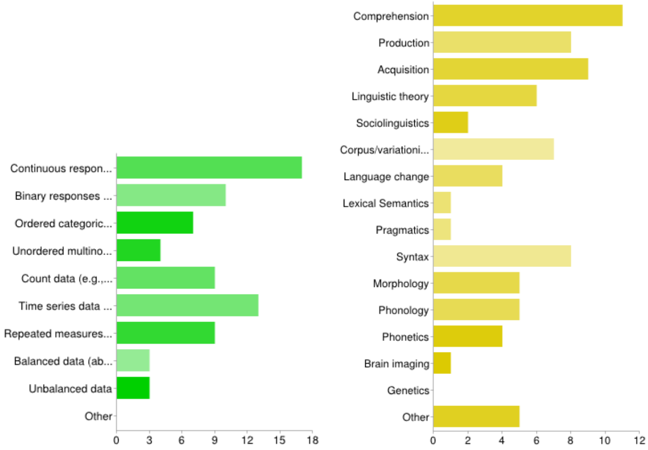


Figure: Background 1(i) and areas of interest (j)

Oh, we are all so different ...

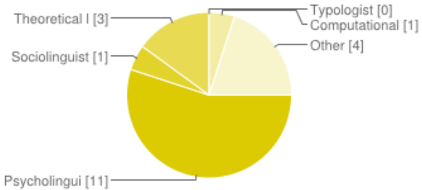


Figure: Background (a)

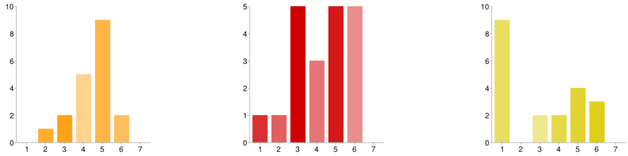


Figure: Expertise in regression (a), R (b), and lme4 (c)

→ Please be patient and help each other out. (Change seating arrangement?)

Acknowledgments

- The slides for this class include (usually modified) materials prepared by:
  - Judith Degen (Rochester)
  - Maureen Gillespie (New Hampshire)
  - Dave Kleinschmidt (Rochester)
  - Victor Kuperman (Stanford)
  - Roger Levy (UCSD)

  ... with their permission
- I am also grateful for feedback from:
  - Austin Frank (Rochester)
  - Previous audiences to similar workshops at CUNY, Haskins, Rochester, Buffalo, UCSD, MIT, Iowa, and Groningen.

Our work environment

- Throughout this class, I'll be using **R** to illustrate statistical concepts.
- It's highly recommended that you use the code on these slides to follow along, but don't loose track of the bigger picture (i.e., continue to listen). If necessary, tell me to slow down.
- Occasionally, you will get stuck on something. Be willing to let go. Otherwise you miss most of the class.

Getting Help

- Subscribe to ling-R-lang: `https://mailman.ucsd.edu/mailman/listinfo/ling-r-lang-l`
- In R: try `?foo` or `help(foo)` first
- Great FAQs for GLMMs: `http://glmm.wikidot.com/faq`
- For more HLP Lab materials, check out:
  - http://www.hlp.rochester.edu/
  - http://wiki.bcs.rochester.edu:2525/HlpLab/StatsCourses
  - http://hlplab.wordpress.com/ (e.g. multinomial mixed models code)
  - Subscribe to our paper feed: http://rochester.academia.edu/tiflo/Papers

Introductions and Tutorials

- R:
  - (Zuur, Ieno, & Meesters, 2009): general purpose introduction to R
  - (Gries, 2009): introduction directed at linguists
  - (Baayen, 2008): introduction to R directed at linguists and basic visualization and data summary toolkit; also provides an introduction to GLM and GLMM, though it's more intended as a (very useful) collection of tools, rather than a conceptual introduction (for a review and summary, see Frank & Jaeger, 2010)

NB: I have only read and worked with Baayen (2008)

Preliminaries


```
version

##                _
## platform       x86_64-w64-mingw32
## arch           x86_64
## os             mingw32
## system         x86_64, mingw32
## status
## major          3
## minor          0.0
## year           2013
## month          04
## day            03
## svn rev        62481
## language       R
## version.string R version 3.0.0 (2013-04-03)
## nickname       Masked Marvel

ls()

## [1] "g1a"    "g1b"    "g2"     "lexdec"
```

Our work environment

- The LSA and UMich organizational staff has kindly set up R and RStudio for you. **RStudio** is a work environment that combines working in R with several conveniences.

Task

Let's take a couple of minutes to familiarize ourselves with RStudio (start RStudio).

RStudio

- The windows:
  - R script window
  - R console
  - Workspace (objects, functions, etc. you've loaded) and history of commands
  - Help, plots, files (browse directories), packages/libraries (1-click load, install, update)

RStudio

- The windows:
  - R script window
  - R console
  - Workspace (objects, functions, etc. you've loaded) and history of commands
  - Help, plots, files (browse directories), packages/libraries (1-click load, install, update)
- Loading, saving, creating, etc. of scripts, sessions, projects (collections of files, working paths, defaults, etc.)

RStudio

- The windows:
  - R script window
  - R console
  - Workspace (objects, functions, etc. you've loaded) and history of commands
  - Help, plots, files (browse directories), packages/libraries (1-click load, install, update)
- Loading, saving, creating, etc. of scripts, sessions, projects (collections of files, working paths, defaults, etc.)
- Loading and displaying of data files

RStudio

- The windows:
  - R script window
  - R console
  - Workspace (objects, functions, etc. you've loaded) and history of commands
  - Help, plots, files (browse directories), packages/libraries (1-click load, install, update)
- Loading, saving, creating, etc. of scripts, sessions, projects (collections of files, working paths, defaults, etc.)
- Loading and displaying of data files
- Understands and compiles R, Latex, Sweave, and Knitr (e.g., from Rnw file to PDF)

RStudio

- The windows:
  - R script window
  - R console
  - Workspace (objects, functions, etc. you've loaded) and history of commands
  - Help, plots, files (browse directories), packages/libraries (1-click load, install, update)
- Loading, saving, creating, etc. of scripts, sessions, projects (collections of files, working paths, defaults, etc.)
- Loading and displaying of data files
- Understands and compiles R, Latex, Sweave, and Knitr (e.g., from Rnw file to PDF)
- Code highlighting, auto-completion of commands, options, etc.

RStudio and Knitr

- If Latex (a typesetting language) is installed on your computer and the `knitr` package (a sweaving language) is installed in R (the latter can be done through RStudio), RStudio understands Latex documents with R code.
- That is you go from Latex to PDF and knitr interprets the R code in the document, automatically creating and inserting tables, figures, models, etc. into your PDF.

RStudio and Knitr

- If Latex (a typesetting language) is installed on your computer and the `knitr` package (a sweaving language) is installed in R (the latter can be done through RStudio), RStudio understands Latex documents with R code.
- That is you go from Latex to PDF and knitr interprets the R code in the document, automatically creating and inserting tables, figures, models, etc. into your PDF.

Demonstration

Time permitting, we'll learn more about that in a later lecture. But here's a quick demonstration.

Check your work environment

Task

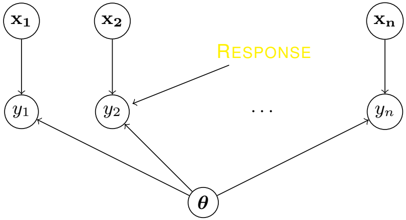
- Check that your R version is 3.0 or higher
- Make sure all libraries we will need are installed (not loaded):
  - `lme4`
  - `languageR`
  - `rms`
  - `gam`
  - `gregmisc`
  - `ggplot2`
  - `stargazer`
  - `knitr`
  - `formatR`
  - `plyr`
  - `reshape2`
  - `labeling`

Generalized Linear Models

Goal: model the effects of PREDICTORS (independent variables) $\mathbf{x}$ on a RESPONSE (dependent variable) $\mathbf{y}$.

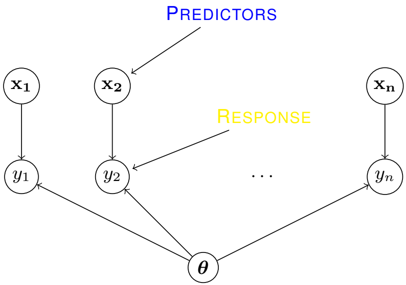
Generalized Linear Models

Goal: model the effects of PREDICTORS (independent variables) $\mathbf{x}$ on a RESPONSE (dependent variable) $\mathbf{y}$.



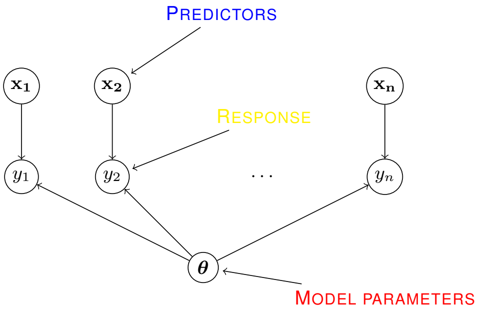
Generalized Linear Models

Goal: model the effects of PREDICTORS (independent variables) $\mathbf{x}$ on a RESPONSE (dependent variable) $\mathbf{y}$.



Generalized Linear Models

Goal: model the effects of PREDICTORS (independent variables) $\mathbf{x}$ on a RESPONSE (dependent variable) $\mathbf{y}$.



Introductions and Tutorials

- (Baayen, 2008): lots of useful tools, data sets, and examples directed at *linguists*; comes with its own library, `languageR`
- (Vasishth & Broe, 2011): a simulation-based approach to statistics that builds up to GLM, though it's not focused on it; great in providing deep intuitions about statistical methods (e.g., what *is* the central limit theorem? etc.); comes with R code for simulations.
- (Harrell, 2001): an amazing concepts and recipe book, which –among many other things– provides guidance and principles for model building and comparison, the assessment of non-linear relations, etc.; not directed at beginners, but also not purely technical; Harrell is a very influential regression statistician; he is also the developer of `Design` (now `rms`), an `R` library for running, validating, and evaluating GLMs

Reviewing GLMs

Assumptions of the generalized linear model (GLM):

- Predictors $\{\mathbf{x_i}\}$ influence $\mathbf{y}$ through the mediation of a LINEAR PREDICTOR η;

Reviewing GLMs

Assumptions of the generalized linear model (GLM):

- Predictors $\{\mathbf{x_i}\}$ influence $\mathbf{y}$ through the mediation of a LINEAR PREDICTOR η;
- η is a linear combination of the $\{\mathbf{x_i}\}$:

Reviewing GLMs

Assumptions of the generalized linear model (GLM):

- Predictors $\{\mathbf{x_i}\}$ influence $\mathbf{y}$ through the mediation of a LINEAR PREDICTOR η;
- η is a linear combination of the $\{\mathbf{x_i}\}$:

$$\eta = \alpha + \beta_1\mathbf{x_1} + \cdots + \beta_n\mathbf{x_n} \qquad \text{(linear predictor)}$$

Reviewing GLMs

Assumptions of the generalized linear model (GLM):

- Predictors $\{\mathbf{x_i}\}$ influence $\mathbf{y}$ through the mediation of a LINEAR PREDICTOR η;
- η is a linear combination of the $\{\mathbf{x_i}\}$:

$$\eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n} \qquad \text{(linear predictor)}$$

- η determines the predicted mean μ of $\mathbf{y}$

$$\eta = g(\mu) \qquad \text{(link function)}$$

Reviewing GLMs

Assumptions of the generalized linear model (GLM):

- Predictors $\{\mathbf{x_i}\}$ influence $\mathbf{y}$ through the mediation of a LINEAR PREDICTOR η;
- η is a linear combination of the $\{\mathbf{x_i}\}$:

$$\eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n} \qquad \text{(linear predictor)}$$

- η determines the predicted mean μ of $\mathbf{y}$

$$\eta = g(\mu) \qquad \text{(link function)}$$

- There is some NOISE DISTRIBUTION of $\mathbf{y}$ around the predicted mean μ of Y:

$$P(\mathbf{y} = y; \mu)$$

Reviewing Linear Regression

LINEAR REGRESSION, which underlies ANOVA, is a kind of generalized linear model.

Reviewing Linear Regression

LINEAR REGRESSION, which underlies ANOVA, is a kind of generalized linear model.

- The predicted mean is just the linear predictor:

$$\eta = I(\mu) = \mu$$

Reviewing Linear Regression

LINEAR REGRESSION, which underlies ANOVA, is a kind of generalized linear model.

- The predicted mean is just the linear predictor:

$$\eta = I(\mu) = \mu$$

- Noise is normally (=Gaussian) distributed around 0 with standard deviation σ_ϵ:

$$\epsilon \sim N(0, \sigma_\epsilon)$$

Reviewing Linear Regression

LINEAR REGRESSION, which underlies ANOVA, is a kind of generalized linear model.

- The predicted mean is just the linear predictor:

$$\eta = I(\mu) = \mu$$

- Noise is normally (=Gaussian) distributed around 0 with standard deviation σ_ϵ:

$$\epsilon \sim N(0, \sigma_\epsilon)$$

- This gives us the traditional linear regression equation:

$$\mathbf{y} = \overbrace{\alpha + \beta_1 \mathbf{x}_1 + \cdots + \beta_n \mathbf{x}_n}^{\text{Predicted Mean } \mu = \eta} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

Other common notations of the linear model

- To refer to individual data points (or simply to highlight the fact that, e.g., $\mathbf{y}$ is a vector), we sometimes write:

$$\mathbf{y_i} = \alpha + \beta_1 \mathbf{x}_{1,i} + \cdots + \beta_n \mathbf{x}_{n,i} + \epsilon_i, \epsilon_i \sim N(0, \sigma_\epsilon)$$

Other common notations of the linear model

- To refer to individual data points (or simply to highlight the fact that, e.g., $\mathbf{y}$ is a vector), we sometimes write:

$$\mathbf{y_i} = \alpha + \beta_1 \mathbf{x}_{1,i} + \cdots + \beta_n \mathbf{x}_{n,i} + \epsilon_i, \epsilon_i \sim N(0, \sigma_\epsilon)$$

NB: $\mathbf{y}, \mathbf{x_1}, \ldots, \mathbf{x_n}$ are vectors of equal length that together constitute the data.

Other common notations of the linear model

- To refer to individual data points (or simply to highlight the fact that, e.g., $\mathbf{y}$ is a vector), we sometimes write:

$$\mathbf{y_i} = \alpha + \beta_1 \mathbf{x}_{1,i} + \cdots + \beta_n \mathbf{x}_{n,i} + \epsilon_i, \epsilon_i \sim N(0, \sigma_\epsilon)$$

NB: $\mathbf{y}, \mathbf{x_1}, \ldots, \mathbf{x_n}$ are vectors of equal length that together constitute the data.

- Instead of α, we sometimes write β_0 or even $\beta_0 X_0$, where X_0 is assumed to be a vector of 1s:

$$\mathbf{y} = \beta_0 \mathbf{x}_0 + \beta_1 \mathbf{x}_1 + \cdots + \beta_n \mathbf{x}_n + \epsilon, \epsilon \sim N(0, \sigma_\epsilon)$$

...or in matrix notation:

$$\mathbf{y} = \mathbf{X}\beta + \epsilon, \epsilon \sim N(0, \sigma_\epsilon)$$

...where the columns of $\mathbf{X}$ are the vectors $\mathbf{x}_0, \ldots, \mathbf{x}_n$, and β is a vector consisting of $\beta_0, \ldots, \beta_n$.

Other common notations of the linear model

- Sometimes this is further simplified and we directly relate the *expectation*, or expected value, of $\mathbf{y}$ to the predictors:

$$E[\mathbf{y}] = \mathbf{X}\beta$$

Understanding the idea of a linear combination

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

```
##                          LangId SId perWordInfo OOVCount DocId
## 4968                  Norwegian   3       8.039        2    10
## 6786                    Swedish   6       8.462        4     8
## 5303                 Portuguese   8       6.367        3    17
## 5915                    Russian   5       6.906        4    39
## 2864 Spanish (Latin-American)  14       7.532        0    13
## 1623                    English   3       7.333        0    61
## 7262                    Swedish   2       7.813        5    40
## 6076                    Russian   1       7.893        1    50
```

(taken from Qian & Jaeger, 2012)

Understanding the idea of a linear combination

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

```
##                          LangId SId perWordInfo OOVCount DocId
## 4968                  Norwegian   3       8.039        2    10
## 6786                    Swedish   6       8.462        4     8
## 5303                 Portuguese   8       6.367        3    17
## 5915                    Russian   5       6.906        4    39
## 2864 Spanish (Latin-American)  14       7.532        0    13
## 1623                    English   3       7.333        0    61
## 7262                    Swedish   2       7.813        5    40
## 6076                    Russian   1       7.893        1    50
```

(taken from Qian & Jaeger, 2012)

Task

- Calculate the predictions of the linear combination $\beta_0 + \beta_1 \mathbf{x_1}$, where the two **coefficients/parameters** are $\beta_0 = 5$ and $\beta_1 = .1$ and the **predictor** $\mathbf{x_1}$ is `SId` (the position of a sentence in a discourse). (btw, this is what `predict(model)` does in R)

Understanding the idea of a linear combination

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

```
##                              LangId SId perWordInfo OOVCount DocId
## 4968                      Norwegian   3       8.039        2    10
## 6786                        Swedish   6       8.462        4     8
## 5303                     Portuguese   8       6.367        3    17
## 5915                        Russian   5       6.906        4    39
## 2864 Spanish (Latin-American)  14       7.532        0    13
## 1623                        English   3       7.333        0    61
## 7262                        Swedish   2       7.813        5    40
## 6076                        Russian   1       7.893        1    50
```

(taken from Qian & Jaeger, 2012)

Task

- Calculate the predictions of the linear combination $\beta_0 + \beta_1 \mathbf{x_1}$, where the two **coefficients/parameters** are $\beta_0 = 5$ and $\beta_1 = .1$ and the **predictor** $\mathbf{x_1}$ is `SId` (the position of a sentence in a discourse). (btw, this is what `predict(model)` does in R)
- Intuitively, how would you tell whether this linear combination is a good predictor of the **outcome** `perWordInfo` (the number of bits per word in that sentence)? Try to quantify that intuition.

Understanding the idea of a linear combination

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

```
##                          LangId SId perWordInfo OOVCount DocId
## 4968                  Norwegian   3       8.039        2    10
## 6786                    Swedish   6       8.462        4     8
## 5303                 Portuguese   8       6.367        3    17
## 5915                    Russian   5       6.906        4    39
## 2864 Spanish (Latin-American)  14       7.532        0    13
## 1623                    English   3       7.333        0    61
## 7262                    Swedish   2       7.813        5    40
## 6076                    Russian   1       7.893        1    50
```

Questions

- Can you think of a way how we can build a better model *without changing the predictor?*

Understanding the idea of a linear combination

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0,\sigma_\epsilon)}$$

```
##                          LangId SId perWordInfo OOVCount DocId
## 4968                  Norwegian   3       8.039        2    10
## 6786                    Swedish   6       8.462        4     8
## 5303                 Portuguese   8       6.367        3    17
## 5915                    Russian   5       6.906        4    39
## 2864 Spanish (Latin-American)  14       7.532        0    13
## 1623                    English   3       7.333        0    61
## 7262                    Swedish   2       7.813        5    40
## 6076                    Russian   1       7.893        1    50
```

Questions

- Can you think of a way how we can build a better model *without changing the predictor?*
- *Now, imagine we want to predict the outcome* `perWordInfo` *from the predictor* `LangId`*. How would that work? What would we have to do?*

Reviewing Logistic Regression

LOGISTIC REGRESSION, too, is a kind of generalized linear model.

Reviewing Logistic Regression

LOGISTIC REGRESSION, too, is a kind of generalized linear model.

- The linear predictor:

$$\eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n}$$

Reviewing Logistic Regression

LOGISTIC REGRESSION, too, is a kind of generalized linear model.

- The linear predictor:

$$\eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n}$$

- The link function g is the logit transform:

$$\mathrm{E}(\mathbf{y}) = p = \mathrm{g}^{-1}(\eta) \Leftrightarrow \mathrm{g}(p) = \ln \frac{p}{1-p} = \eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n} \quad (1)$$

Reviewing Logistic Regression

LOGISTIC REGRESSION, too, is a kind of generalized linear model.

- The linear predictor:

$$\eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n}$$

- The link function g is the logit transform:

$$\mathrm{E}(\mathbf{y}) = p = \mathrm{g}^{-1}(\eta) \Leftrightarrow$$
$$\mathrm{g}(p) = \ln \frac{p}{1-p} = \eta = \alpha + \beta_1 \mathbf{x_1} + \cdots + \beta_n \mathbf{x_n} \quad (1)$$

- The distribution around the mean is taken to be binomial.

Reviewing GLM

- Logistic regression
- Poisson regression
- Beta-binomial model (for low count data, for example)
- Ordered and unordered multinomial regression.
- ...

The Linear Model

- To illustrate the Linear Model (a GLM with a Gaussian link function), we will be studying reaction times (RTs) in a visual lexical-decision task.
- In such tasks, participants are presented string of letters and have to decide as fast as possible (typically by button press) whether the string is a word in the target language (here English) or not:

The Linear Model

- To illustrate the Linear Model (a GLM with a Gaussian link function), we will be studying reaction times (RTs) in a visual lexical-decision task.
- In such tasks, participants are presented string of letters and have to decide as fast as possible (typically by button press) whether the string is a word in the target language (here English) or not:

  `tpozt` *Word or non-word?*

The Linear Model

- To illustrate the Linear Model (a GLM with a Gaussian link function), we will be studying reaction times (RTs) in a visual lexical-decision task.
- In such tasks, participants are presented string of letters and have to decide as fast as possible (typically by button press) whether the string is a word in the target language (here English) or not:

`tpozt`	*Word or non-word?*
`house`	*Word or non-word?*

Example data: Lexical decision RTs

- Data set `lexdec` based on Baayen, Feldman, and Schreuder (2006) (available through `languageR` library in `R`)

ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT

Journal of Memory and Language 55 (2006) 290–313

Journal of Memory and Language

www.elsevier.com/locate/jml

Morphological influences on the recognition of monosyllabic monomorphemic words

R.H. Baayen [a,*], L.B. Feldman [b], R. Schreuder [c]

[a] *Radboud University Nijmegen and Max Planck Institute for Psycholinguistics, P.O. Box 310, 6500 AH Nijmegen, The Netherlands*
[b] *State University of New York at Albany, Department of Psychology, SS112 Albany, NY 12222, USA*
[c] *Radboud University Nijmegen, P.O. Box 310, 6500 AH Nijmegen, The Netherlands*

Received 15 July 2005; revision received 28 March 2006

Example data: Lexical decision RTs

- **Outcome:** (log-transformed) lexical decision latency `RT`
- **Inputs:**
  - factors `Subject` (21 levels) and `Word` (79 levels),
  - factor `NativeLanguage` (*English* and *Other*)
  - continuous predictors `Frequency` (log word frequency), `Trial` (rank in the experimental list) . . . and many more

NB: only responses to word stimuli are included in `lexdec`

```
##      Subject    RT Trial Sex NativeLanguage     Word Frequency
## 1515        I 5.974    50   F          Other    grape     5.193
## 616        T1 6.184   145   F        English    moose     2.708
## 1149       R2 6.585   113   M        English   peanut     4.595
## 1000        C 6.146   131   F        English    bunny     3.332
## 1227       T2 6.901   102   F          Other eggplant     1.792
## 916        W2 6.290   129   M        English      dog     7.668
```

Get to know the data.frame `lexdec`

Task (5mins)

- Load the data set and start exploring it.

  (e.g., how many variables are in there? what do they mean? what distributions do these variables have?)

```
library(languageR)
data(lexdec)

# Try some of the following:
help(lexdec)       # learn about this R object
?lexdec            # the same
class(lexdec)      # what type of R object is lexdec?
nrow(lexdec)       # number of rows (cases)
str(lexdec)        # works on almost all R objects
summary(lexdec)    # summary of each variable in the data.frame
summary(lexdec$RT)
mean(lexdec$RT)
var(lexdec$RT)
with(lexdec, mean(RT))

# create a new data.frame that is a subset of lexdec
new = subset(lexdec, RT > 7)
nrow(new)
summary(new$RT)
```

Example: Investigating frequency effects

- Here we are interested in the effect of (log-transformed) word frequency on (log-transformed) lexical decision RTs: Specifically, does word frequency affect how fast we recognize a string as word? If so, this would argue that retrieval of lexical representations is frequency sensitive.

```
summary(lexdec[,c('RT','Frequency')])

##        RT          Frequency
##  Min.   :5.83   Min.   :1.79
##  1st Qu.:6.21   1st Qu.:3.95
##  Median :6.35   Median :4.75
##  Mean   :6.38   Mean   :4.75
##  3rd Qu.:6.50   3rd Qu.:5.65
##  Max.   :7.59   Max.   :7.77
```

A linear model to investigate frequency effects

- To that end, . . .
  - we define a linear model to the data to predict (log-transformed) RTs from (log-transformed) word frequency
  - we find the parameters to this model that maximize the fit against the outcome data (i.e., minimize our prediction error against RTs). Conveniently, statistics program do these for us.
  - interpret the model and derive conclusions based on the distribution of these parameters about the significance of word frequency as a predictor of lexical decision RTs

Defining the linear model

- As a reminder here is the general formula for a linear model

$$\mathbf{y} = \overbrace{\beta_0\mathbf{x_0} + \ldots + \beta_n\mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

Defining the linear model

- As a reminder here is the general formula for a linear model

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

- For our current case with one predictor, we can thus write:

$$\mathbf{y} = \overbrace{\beta_0 + \beta_1 \mathbf{x_1}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

... where β_0 is the intercept, x_1 is the predictor `Frequency` and y is the outcome `RTs` in the `lexdec` data (we discuss later why we include an intercept).

Defining the linear model

- As a reminder here is the general formula for a linear model

$$\mathbf{y} = \overbrace{\beta_0 \mathbf{x_0} + \ldots + \beta_n \mathbf{x_n}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- For our current case with one predictor, we can thus write:

$$\mathbf{y} = \overbrace{\beta_0 + \beta_1 \mathbf{x_1}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

... where β_0 is the intercept, x_1 is the predictor `Frequency` and y is the outcome `RTs` in the `lexdec` data (we discuss later why we include an intercept).

I.e.,

$$\texttt{RT} = \overbrace{\beta_0 + \beta_1 \texttt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

Fitting the linear model

$$\texttt{RT} = \overbrace{\beta_0 + \beta_1 \texttt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

- Luckily, we don't need to be Gauss to fit a linear model. We just let R (or another statistics program) fit the model to the data. For example, in R:

```
# glm() is R's call to a GLM
# 1, in an R-formula, is a specific symbol for the intercept
# Frequency and RT are variables in the data.frame lexdec
# data tells glm which data.frame to use
# family tells glm which distributions the outcome is assumed to have
m = glm(RT ~ 1 + Frequency, data = lexdec, family = gaussian)
```

Fitting the linear model

$$\texttt{RT} = \overbrace{\beta_0 + \beta_1 \texttt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- Luckily, we don't need to be Gauss to fit a linear model. We just let R (or another statistics program) fit the model to the data. For example, in R:

```
# glm() is R's call to a GLM
# 1, in an R-formula, is a specific symbol for the intercept
# Frequency and RT are variables in the data.frame lexdec
# data tells glm which data.frame to use
# family tells glm which distributions the outcome is assumed to have
m = glm(RT ~ 1 + Frequency, data = lexdec, family = gaussian)
```

- This fits the coefficients/parameters, i.e.
  - the intercept β_0,
  - the slope of the Frequency effect β_1, and
  - the standard deviation of the residuals σ_ϵ

  ...to the data, using MAXIMUM LIKELIHOOD (ML) ESTIMATION.

Some shortcuts in R

- By default R models include an intercept, so the `1` below in the call on the previous page (repeated below) is redundant.
- The default family for a `glm` (and `lmer`) is `gaussian`.
- As long as we provide arguments in the default order (see `?glm`), we can omit the variable identifier, so that the following statements are equivalent:

```
library(languageR)
data(lexdec)
glm(RT ~ 1 + Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec)
```

Some shortcuts in R

- By default R models include an intercept, so the `1` below in the call on the previous page (repeated below) is redundant.
- The default family for a `glm` (and `lmer`) is `gaussian`.
- As long as we provide arguments in the default order (see `?glm`), we can omit the variable identifier, so that the following statements are equivalent:

```
library(languageR)
data(lexdec)
glm(RT ~ 1 + Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec)
```

- To remove the intercept from a model, incude `- 1` in the model's formula:

```
glm(RT ~ - 1 + Frequency, data = lexdec)
```

Some shortcuts in R

- By default R models include an intercept, so the `1` below in the call on the previous page (repeated below) is redundant.
- The default family for a `glm` (and `lmer`) is `gaussian`.
- As long as we provide arguments in the default order (see `?glm`), we can omit the variable identifier, so that the following statements are equivalent:

```
library(languageR)
data(lexdec)
glm(RT ~ 1 + Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec, family = gaussian)
glm(RT ~ Frequency, data = lexdec)
```

- To remove the intercept from a model, incude `- 1` in the model's formula:

```
glm(RT ~ - 1 + Frequency, data = lexdec)
```

Task (1min)

Try it out for yourself.

Fitted model

$$\mathtt{RT} = \overbrace{\beta_0 + \beta_1 \mathtt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- The ML-fited model provides coefficient *estimates* (hence, the hat above the βs and σ), and estimates of their standard errors.

  (for linear models, the analytic optimal solution is known, so these estimates are guaranteed to be optimal in that their minimize the prediction error against the known data $\mathbf{y}$)

```
summary(m)$coefficients[,1:2]

##              Estimate Std. Error
## (Intercept)  6.58878   0.022296
## Frequency   -0.04287   0.004533

sqrt(summary(m)$dispersion)

## [1] 0.2353
```

Fitted model

$$\mathtt{RT} = \overbrace{\beta_0 + \beta_1 \mathtt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

- The ML-fited model provides coefficient *estimates* (hence, the hat above the βs and σ), and estimates of their standard errors.

  (for linear models, the analytic optimal solution is known, so these estimates are guaranteed to be optimal in that their minimize the prediction error against the known data $\mathbf{y}$)

```
summary(m)$coefficients[,1:2]

##               Estimate Std. Error
## (Intercept)   6.58878   0.022296
## Frequency    -0.04287   0.004533

sqrt(summary(m)$dispersion)

## [1] 0.2353
```

$\widehat{\beta_0}$

Fitted model

$$\text{RT} = \overbrace{\beta_0 + \beta_1 \texttt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

- The ML-fited model provides coefficient *estimates* (hence, the hat above the βs and σ), and estimates of their standard errors.

  (for linear models, the analytic optimal solution is known, so these estimates are guaranteed to be optimal in that their minimize the prediction error against the known data $\mathbf{y}$)

```
summary(m)$coefficients[,1:2]

##              Estimate Std. Error
## (Intercept)   6.58878   0.022296
## Frequency    -0.04287   0.004533

sqrt(summary(m)$dispersion)

## [1] 0.2353
```

$\widehat{\beta_0}$

$\widehat{\beta_1}$

Fitted model

$$\texttt{RT} = \overbrace{\beta_0 + \beta_1 \texttt{Frequency}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise} \sim N(0, \sigma_\epsilon)}$$

- The ML-fited model provides coefficient *estimates* (hence, the hat above the βs and σ), and estimates of their standard errors.

  (for linear models, the analytic optimal solution is known, so these estimates are guaranteed to be optimal in that their minimize the prediction error against the known data $\mathbf{y}$)

```
summary(m)$coefficients[,1:2]

##              Estimate Std. Error
## (Intercept)   6.58878   0.022296
## Frequency    -0.04287   0.004533

sqrt(summary(m)$dispersion)

## [1] 0.2353
```

$\widehat{\beta_0}$ → 6.58878

$\widehat{\beta_1}$ → -0.04287

$\widehat{\sigma_\epsilon}$ → 0.2353

Geometrical intuitions

- Before we discuss how to draw inferences based in a (ML-fitted) linear model, let's get a bit more of a geometrical intuition for what those coefficients mean.

Geometrical intuitions

- Before we discuss how to draw inferences based in a (ML-fitted) linear model, let's get a bit more of a geometrical intuition for what those coefficients mean.
- For that, it's helpful to look at an even simpler model: a linear model with only the intercept, i.e.

$$\mathtt{RT} = \overbrace{\beta_0}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

. . . where we are predicting (log-transformed) lexical decision RTs based on only a constant (the intercept).

Geometrical intuitions

- Before we discuss how to draw inferences based in a (ML-fitted) linear model, let's get a bit more of a geometrical intuition for what those coefficients mean.
- For that, it's helpful to look at an even simpler model: a linear model with only the intercept, i.e.

$$\mathtt{RT} = \overbrace{\beta_0}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

. . . where we are predicting (log-transformed) lexical decision RTs based on only a constant (the intercept).

```
m0 = glm(RT ~ 1, data = lexdec, family = gaussian)
```

Linear Model with just an intercept


```
summary(m0)$coefficients[,1:2]

##    Estimate Std. Error
##    6.385090   0.005929
```

- Note that the intercept estimate for `m0` differs from that of `m` (6.5888)

Question

What does the intercept (i.e., $\widehat{\beta_0}$) encode here?

Linear Model with just an intercept

```
summary(m0)$coefficients[,1:2]

##   Estimate Std. Error
##   6.385090   0.005929
```

- Note that the intercept estimate for `m0` differs from that of `m` (6.5888)

Question

What does the intercept (i.e., $\widehat{\beta_0}$) encode here?

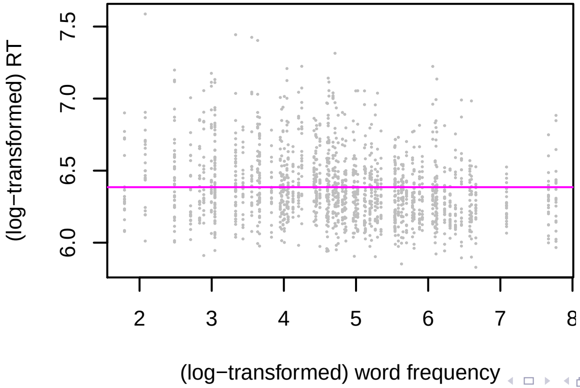
```
options(digits=6)
mean(lexdec$RT)

## [1] 6.38509
```

Proof.

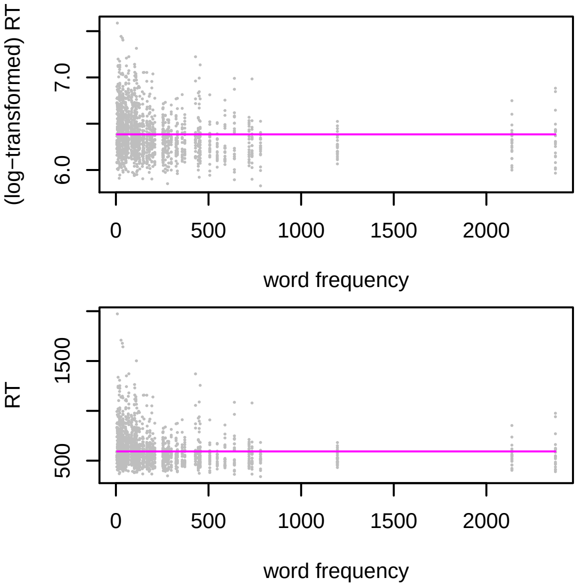
The mean is the maximum likelihood estimate of **y**. □

Visualization of Intercept Model


```
par(cex=.7, mar=c(4,4,0.1,0.1))
plot(x = lexdec$Frequency,
     y = lexdec$RT,
     ylab = "(log-transformed) RT",
     xlab = "(log-transformed) word frequency",
     type = "n"
)
points(x = lexdec$Frequency, y = lexdec$RT, pch=1, cex=.1, col = "grey")
abline(m0, col = "magenta")
```



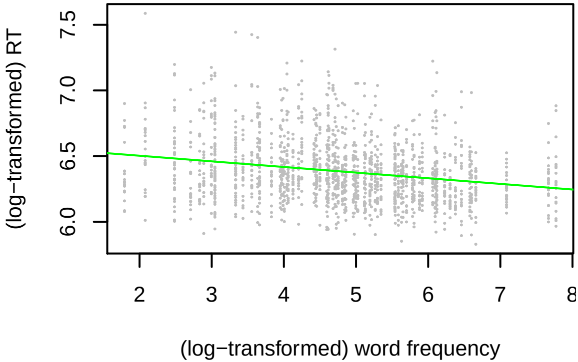
Changing scales



Going back to the frequency model

$$\mathbf{y} = \overbrace{\beta_0 + \beta_1 \mathbf{x_1}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- The coefficient estimate $\widehat{\beta_0}$ describes the intercept (where the line cross the y-axis)
- The coefficient estimate $\widehat{\beta_1}$ is our ML estimate for the *slope* of `Frequency`.



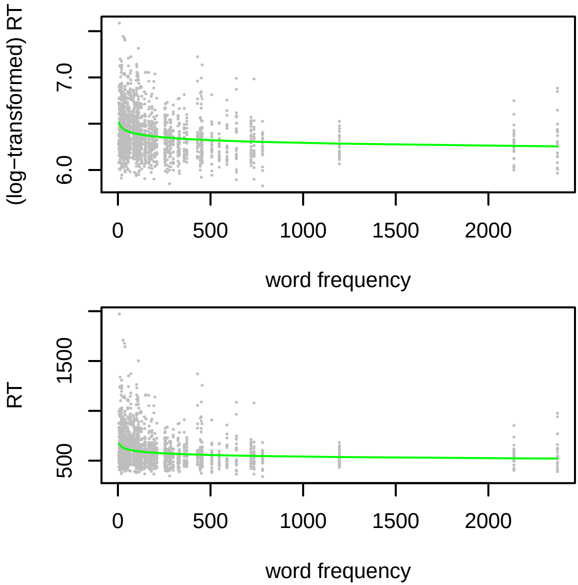
Interpretation: Ordinary least squares (OLS) regression

- The slope for `Frequency` (i.e., $\widehat{\beta_1}$) minimizes the sum of the squared *vertical* distances between the line and all points.

NB: The directionality in this statement is important - we are minimizing the (squared) error in predicting the *outcome* (not the distance from the line).

NB: Maximum likelihood (ML) fitting is identical to least-squared error for Gaussian errors, but ML fitting is the more general approach then this geometrical interpretation, as it extends to other types of GLMs.

Changing scales



Drawing inferences from the linear model

$$\mathbf{y} = \overbrace{\beta_0 + \beta_1 \mathbf{x_1}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- Based on the ML-fitted linear model, we can draw inferences about the statistical significance of word frequency as a predictor of RTs
- That is, we can draw test whether β is statistically different from 0 (the null hypothesis)
- If it is, we will reject the null hypothesis that `Frequency` has no effect on `RT`s (i.e, we conclude that word frequency affect lexical decision RTs).

Drawing inferences from the linear model

$$\mathbf{y} = \overbrace{\beta_0 + \beta_1 \mathbf{x_1}}^{\text{Predicted Mean}} + \overbrace{\epsilon}^{\text{Noise}\sim N(0,\sigma_\epsilon)}$$

- Based on the ML-fitted linear model, we can draw inferences about the statistical significance of word frequency as a predictor of RTs
- That is, we can draw test whether β is statistically different from 0 (the null hypothesis)
- If it is, we will reject the null hypothesis that `Frequency` has no effect on `RT`s (i.e, we conclude that word frequency affect lexical decision RTs).

- As a matter of fact, all the information we need is already there:

```
##              Estimate Std. Error   t value    Pr(>|t|)
## (Intercept)  6.5887784 0.02229593 295.51482 0.00000e+00
## Frequency   -0.0428718 0.00453251  -9.45874 1.02656e-20
```

Question

- What is the *t*-statistic based on?
- What would be an intuitive interpretation of what the *t*-statistic tells us?

Drawing inferences from the linear model

- The standard summary of a `glm` object already contains all this information and more:

```
summary(m)

##
## Call:
## glm(formula = RT ~ 1 + Frequency, family = gaussian, data = lexdec)
##
## Deviance Residuals:
##     Min        1Q    Median        3Q       Max
## -0.5541   -0.1615   -0.0349   0.1170   1.0877
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.58878    0.02230  295.51   <2e-16 ***
## Frequency    -0.04287    0.00453   -9.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.0553721)
##
##     Null deviance: 96.706  on 1658  degrees of freedom
## Residual deviance: 91.752  on 1657  degrees of freedom
## AIC: -88.58
##
## Number of Fisher Scoring iterations: 2
```

So, is this the absolute truth?

- The conclusions drawn from a model are *conditional on the model we assumed*!

  (that is *always* true in statistics, although we tend to forget that)

- That is, our conclusions depend on
  - the general *type* of model we used, in this case an linear model:
    - The effect of `Frequency` in `RT`s, i.e. that log-transformed word frequency has a linear effect (if any) on log-transformed lexical decision RTs
    - Trial-level noise is *normally distributed*, i.e. log-transformed RTs are normally distributed
  - the specific predictors in the model, in this case meaning that we assume that no factors other than `Frequency` affect `RT`s or that they do so in ways independent of `Frequency`

Relation to ANOVA

- Now that we have a good intuition about GLM, let's relate it to **Analysis of Variance (ANOVA)**
- The two methods share the goal of assessing statistical significance of one set of variables (predictors or independent vaiables) in the explanation of the distribution of other variables (outcomes or dependent variables).
- The two methods also share some, though not all, assumptions.
- But first a quick refresher for ANOVA (figures on next three slides are taken from Wikipedia, `https://en.wikipedia.org/wiki/Analysis_of_variance`).

ANOVA

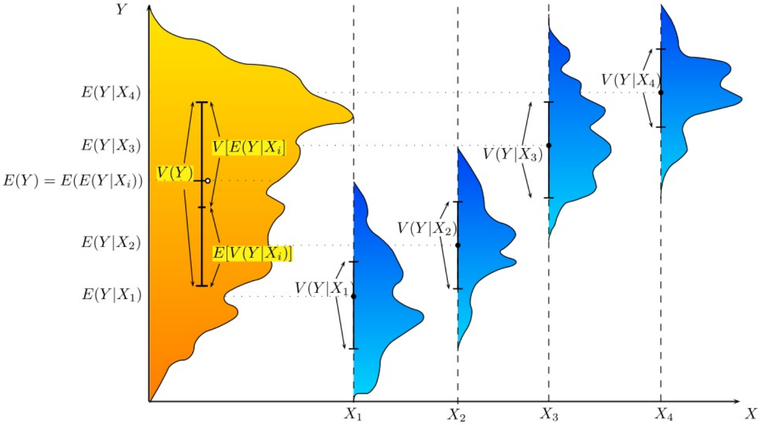


Figure 1: ANOVA : Fair fit

ANOVA: no effect

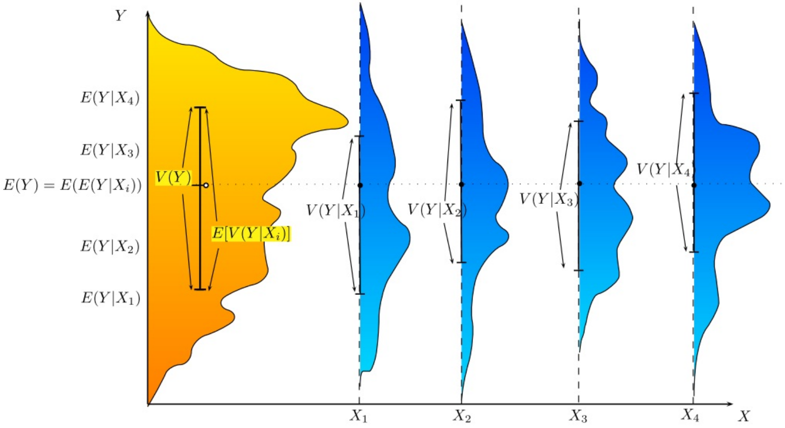


Figure 2: ANOVA : No fit

ANOVA: clear effect

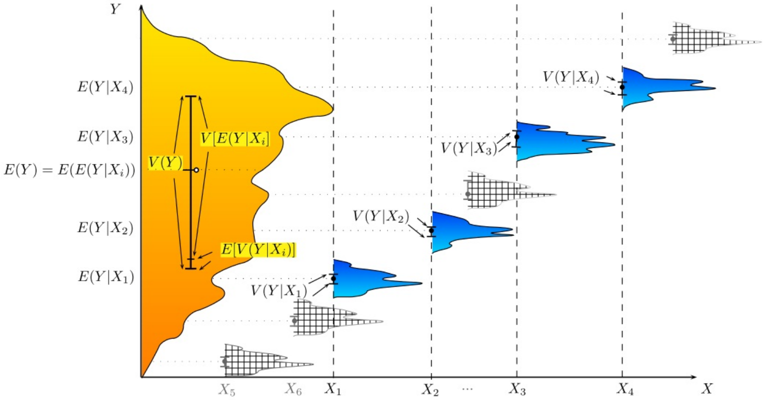


Figure 3: ANOVA : very good fit

Linear Model vs. ANOVA

- Shared with ANOVA:
  - Linearity assumption (though many types of non-linearity can be investigated)
  - Assumption of normality, but part of a more general framework that extends to other distribution in a conceptually straightforward way.
  - Assumption of independence

  NB: **ANOVA** is linear model with (only) categorical predictors. An **ANCOVA** contains categorical *and* continuous predictors.
- Differences:
  - **GLM** readily extends to other instances from the **exponential family** (e.g., Binomials, Poisson).
  - **GLM** encourages a priori explicit coding of hypothesis.

Linearity Assumption

NB: Like AN(C)OVA, the linear model assumes that the outcome is linear *in the coefficients* (**linearity assumption**).

- This does not mean that the outcome and the **input variable** have to be linearly related (cf. previous page).
- To illustrate this, consider that we can back-transform the log-transformed `Frequency` (as shown above)

Hypothesis testing in psycholinguistic research

- Typically, we make predictions not just about the existence, but also the *direction* of effects.
- Sometimes, we're also interested in effect *shapes* (non-linearities, etc.)
- Unlike in ANOVA, regression analyses reliably test hypotheses about **effect direction**, **effect shape**, and **effect size** without requiring post-hoc analyses if (a) *the predictors in the model are coded appropriately* (cf. lecture on Coding Categorical Predictors) and (b) *the model can be trusted* (cf. lecture on Common Issues and Solutions in Regression).

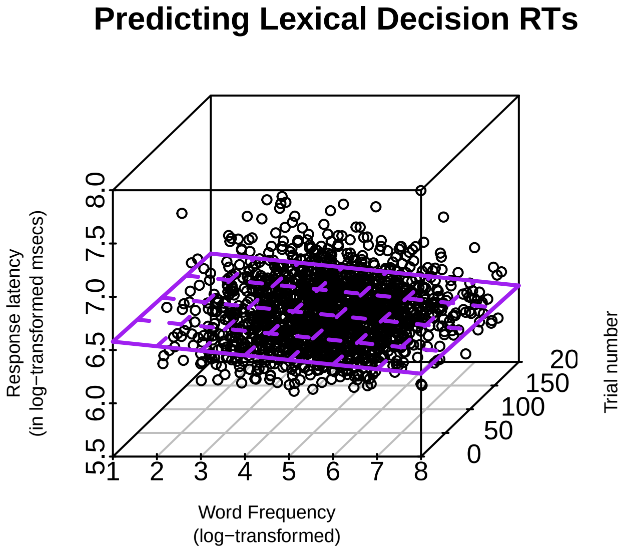
A linear model with two continuous predictors

- `Trial` is the position of the word trial within the lists

```
summary(glm(RT ~ Frequency + Trial, data = lexdec))$coefficients

##                  Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)   6.621395319 0.025731603 257.32541 0.00000e+00
## Frequency    -0.042904434 0.004525177  -9.48127 8.37213e-21
## Trial        -0.000309301 0.000122405  -2.52687 1.16009e-02
```

Questions

- What is the interpretation of the intercept?
- What is the interpretation of the other coefficients?


Predicting Lexical Decision RTs
Response latency
(in log−transformed msecs)
5.5
6.0
6.5
7.0
7.5
8.0
1
2
3
4
5
6
7
8
0
50
100
150
20
Trial number
Word Frequency
(log−transformed)

Continuous and categorical predictors

- `NativeLanguage` codes whether the subject was a native speaker of English (`NativeLanguage` = "Native") or not (`NativeLanguage` = "Other")

```
s = summary(glm(RT ~ Frequency + Trial + NativeLanguage, data = lexdec))
s$coefficients

##                        Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)         6.552507668 0.024802819 264.18399 0.00000e+00
## Frequency          -0.042902172 0.004276534 -10.03200 4.94192e-23
## Trial              -0.000287856 0.000115689  -2.48819 1.29374e-02
## NativeLanguageOther 0.155461166 0.011015873  14.11247 8.66926e-43
```

Questions

- What is the interpretation of the intercept?
- What is the interpretation of the other coefficients?

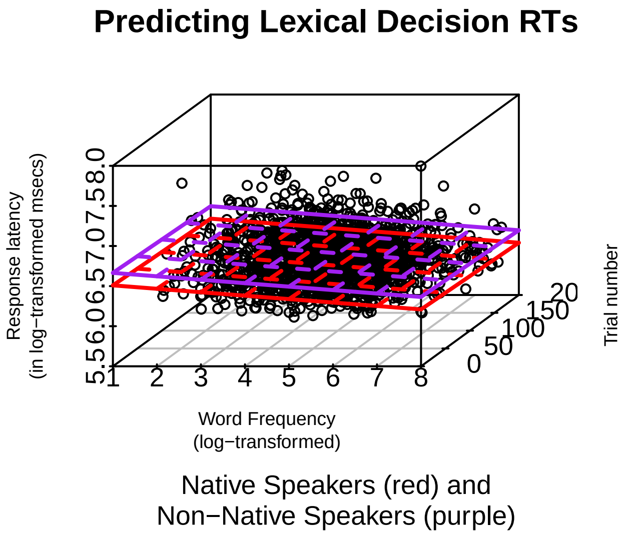
Continuous and categorical predictors

- Recall that we're describing the **output** as a linear combination of the **predictors**.

→ Categorical predictors need to be (re)coded numerically. (cf. **coding lecture**)

NB: The default is 'dummy'/'treatment' coding for regression, where the treatment contrasts are based on the alphabetical order of the levels of the categorical variable.

- This can be quite confusing, as in the current case, where `NativeLanguage` recoded to 1 for "Other" vs. 0 for "Native".

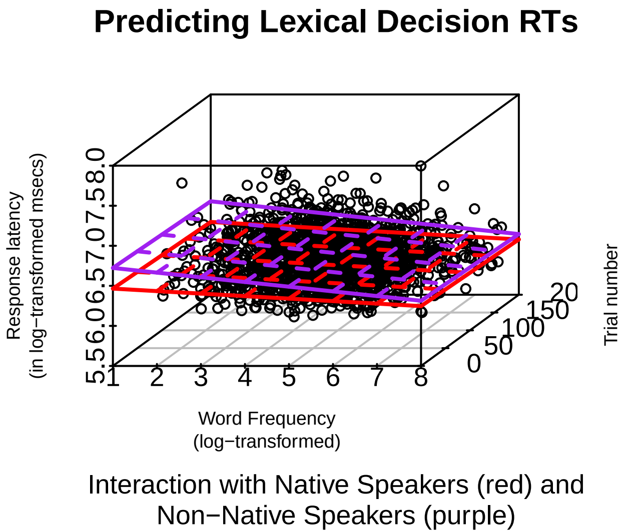

Predicting Lexical Decision RTs
Response latency
(in log−transformed msecs)
5.5 6.0 6.5 7.0 7.5 8.0
1 2 3 4 5 6 7 8
0 50 100 150 20
Trial number
Word Frequency
(log−transformed)
Native Speakers (red) and
Non−Native Speakers (purple)

Interactions

- Let's test whether `NativeLanguage` and `Frequency` interact: Perhaps non-native speakers are only slower on average because they don't know some of the words? In that case, we should see that natives and non-natives have similar RTs for high frequency words.

```
m4 = glm(RT ~
           Frequency +
           Trial +
           NativeLanguage +
           Frequency:NativeLanguage,
         data = lexdec
)
summary(m4)$coefficients

##                                 Estimate  Std. Error   t value
## (Intercept)                  6.496609331 0.030389907 213.77523
## Frequency                   -0.031207253 0.005642111  -5.53113
## Trial                       -0.000284685 0.000115379  -2.46739
## NativeLanguageOther          0.285110406 0.042396907   6.72479
## Frequency:NativeLanguageOther -0.027287364 0.008618527  -3.16613
##                                Pr(>|t|)
## (Intercept)                  0.00000e+00
## Frequency                    3.69219e-08
## Trial                        1.37110e-02
## NativeLanguageOther          2.41421e-11
## Frequency:NativeLanguageOther 1.57299e-03
```

Visualization



Interactions

```
##                                  Estimate  Std. Error   t value
## (Intercept)                   6.496609331 0.030389907 213.77523
## Frequency                    -0.031207253 0.005642111  -5.53113
## Trial                        -0.000284685 0.000115379  -2.46739
## NativeLanguageOther           0.285110406 0.042396907   6.72479
## Frequency:NativeLanguageOther -0.027287364 0.008618527  -3.16613
##                                  Pr(>|t|)
## (Intercept)                   0.00000e+00
## Frequency                     3.69219e-08
## Trial                         1.37110e-02
## NativeLanguageOther           2.41421e-11
## Frequency:NativeLanguageOther 1.57299e-03
```

Questions

- What is the interpretation of the intercept?
- What is the interpretation of the other coefficients?

Baayen, R. H. (2008). *Analyzing linguistic data* (Vol. 505). Cambridge University Press Cambridge, UK.

Baayen, R. H., Feldman, L., & Schreuder, R. (2006). Morphological influences on the recognition of monosyllabic monomorphemic words. *Journal of Memory and Language*, *55*(2), 290–313.

Frank, A. F., & Jaeger, T. F. (2010). Analyzing linguistic data: a practical introduction to statistics using r. *Functions of Language*, *17*(1), 134–143.

Gries, S. T. (2009). *Statistics for linguistics with r: a practical introduction* (Vol. 208). Walter de Gruyter.

Harrell, F. E. J. (2001). *Regression modeling strategies*. Oxford: Springer-Verlag.

Qian, T., & Jaeger, T. (2012). Cue effectiveness in communicatively efficient discourse production. *Cognitive Science*, *36*(7), 1312-1336.

Vasishth, S., & Broe, M. B. (2011). *The foundations of statistics: A simulation-based approach*. Springer.

Zuur, A. F., Ieno, E. N., & Meesters, E. H. (2009). *A beginner's guide to r*. Springer.