Advanced Statistics

Bernhard Angele

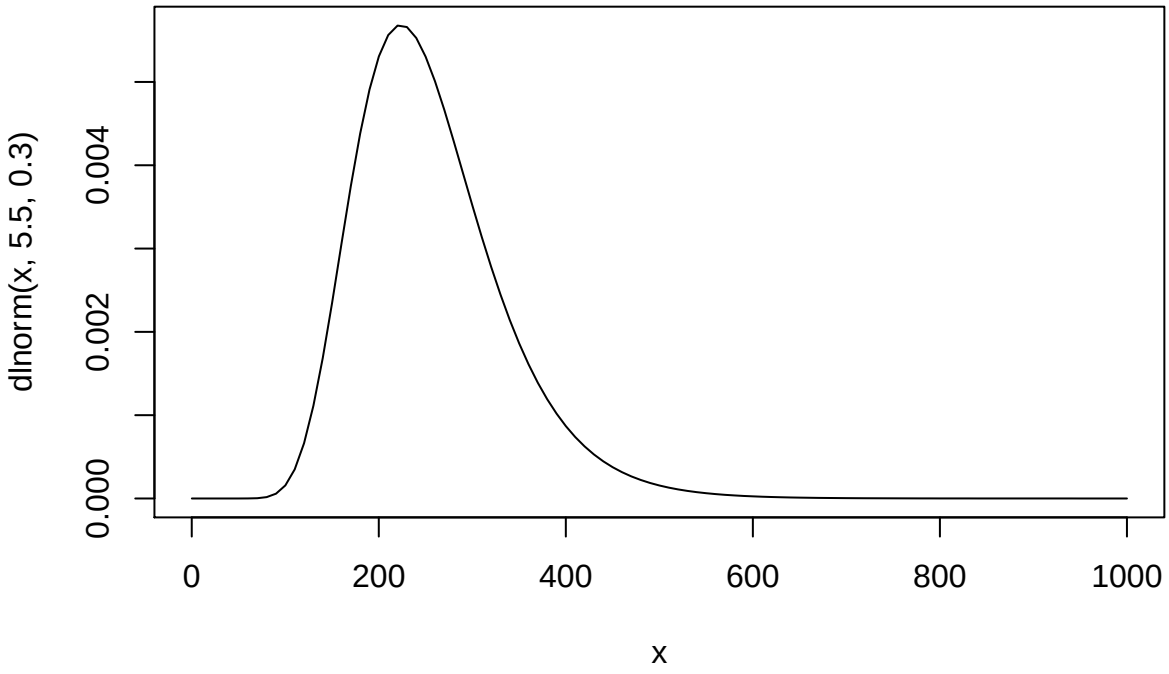
Lecture 5

Transformations

- In some cases, our dependent variable will not be normally distributed
- Example: reaction times – you get a long right tail of slow responses
  - Fixation times in eye movements are very similar

Example

- For example, the probability density function for fixation durations might look like this:



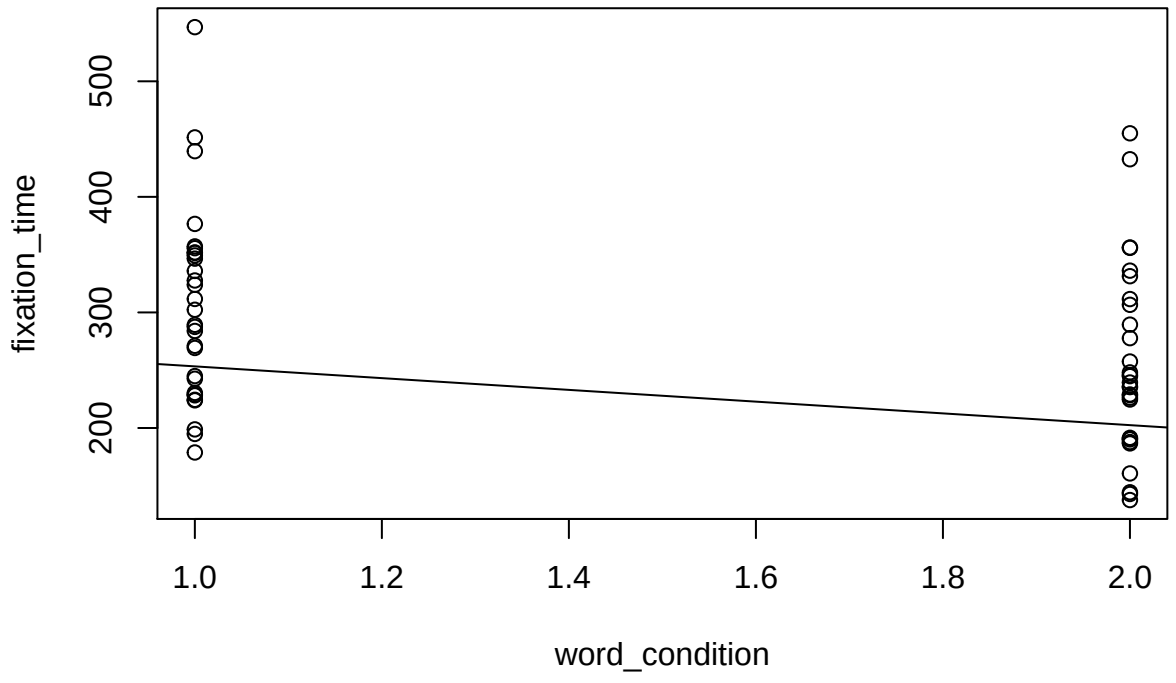
Example data

- Example experiment: how long do people look at swear words vs. non-swear words?
  - Let's assume that the true means are 250 ms for non swear words and 300 ms for swear words
- Let's generate data based on this assumption

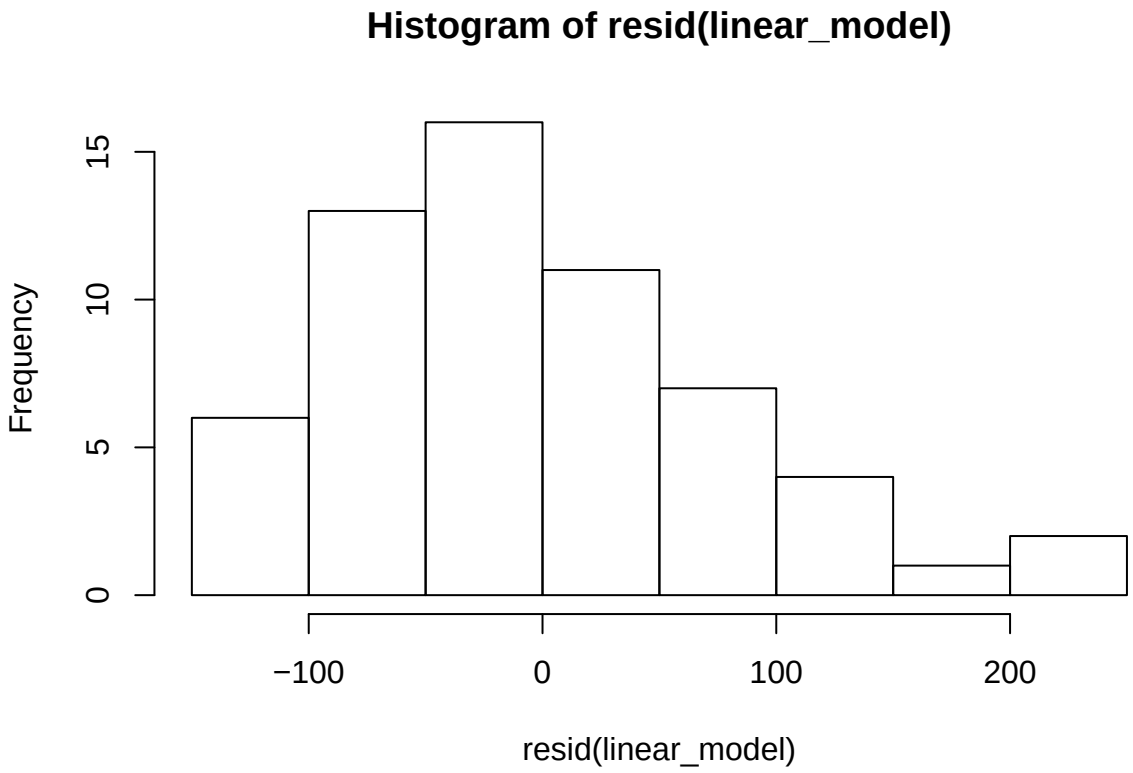
Running a linear model

```
##
## Call:
## lm(formula = fixation_time ~ word_condition, data = swear_exp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -125.34  -62.91  -14.41   47.39  242.69
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 304.16      14.86  20.469   <2e-16 ***
## word_conditionswear word    -50.84      21.01  -2.419   0.0187 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 81.39 on 58 degrees of freedom
## Multiple R-squared:  0.09166,    Adjusted R-squared:  0.076
## F-statistic: 5.853 on 1 and 58 DF,  p-value: 0.01871
```

Plotting the fitted line



Histogram of the residuals



- Clearly not normal - it's right-skewed! - But notice how robust the analysis is. We still find the effect!

Logarithmic transformations

- Better to run a proper model where the values as the dependent variable
- This is our first step into the world of generalised linear models (GLM)
- The most common transformation uses the logarithm
- Remember the logarithm from school?
  - I hope you remember exponentiation: $x \cdot x \cdot x \cdot x = x^4$
  - Every logarithm has a base
  - The logarithm of a number is the exponent to which the base needs to be raised to produce that number
  - For example, the base x logarithm of x^4 is 4, since x needs to be raised to the 4th power to produce x^4: $log_x(x^4) = 4$

The natural logarithm

- It doesn't really matter which base you use for your logarithm
- In mathematics, the algorithm to the base $e = 2.718$ called the *natural logarithm*. It is used frequently, since it has some very convenient properties. The logarithm to the base e is called log_e or ln.

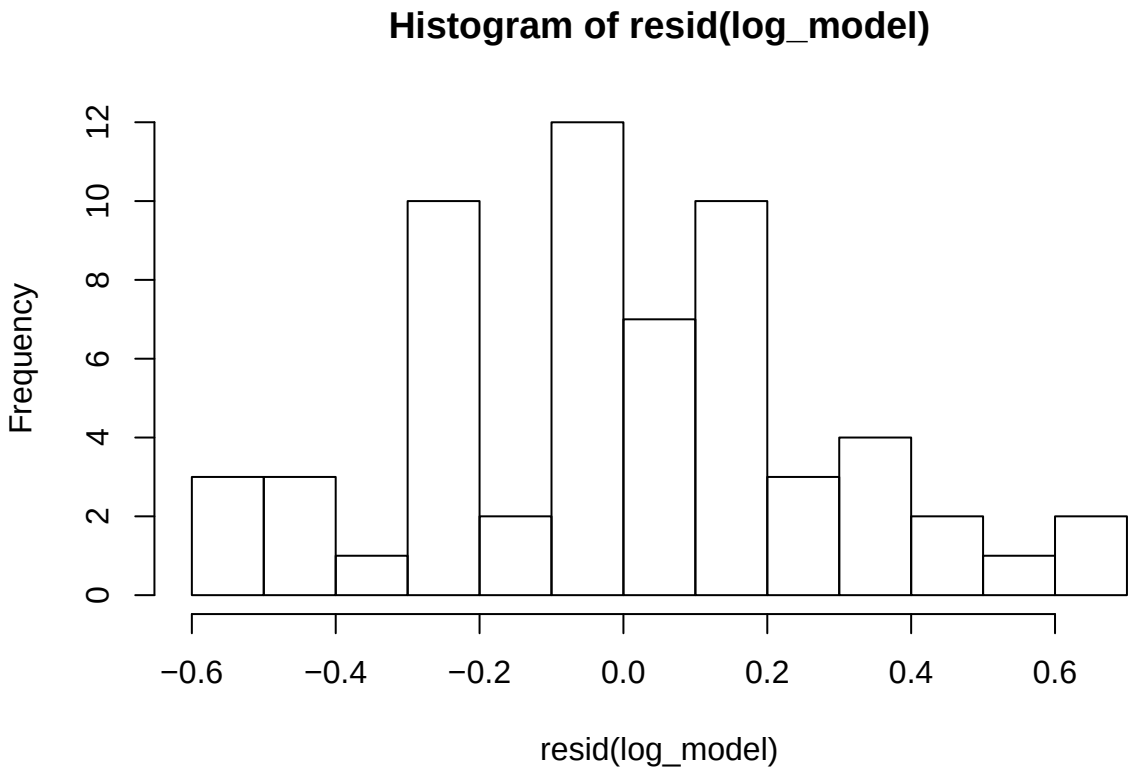
- Just remember that $ln(e^x) = x$ and $e^{ln(x)} = x$.
- Also, remember that $x^{a+b} = x^a \cdot x^b$
  - Because of this, $e^{ln(x)+ln(y)} = x \cdot y$
  - And, the other way around, $ln(e^x \cdot e^y) = x + y$

Running a log model

- Use `Compute` in SPSS to transform the predicted variable

```
##
## Call:
## lm(formula = log(fixation_time) ~ word_condition, data = swear_exp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.56356 -0.23919 -0.01322  0.17969  0.63194
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 5.68347    0.05245 108.368   <2e-16 ***
## word_conditionswear word -0.19513    0.07417  -2.631   0.0109 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2873 on 58 degrees of freedom
## Multiple R-squared:  0.1066, Adjusted R-squared:  0.0912
## F-statistic: 6.921 on 1 and 58 DF,  p-value: 0.01089
```

Histogram of the residuals of the transformed model



- Much better! No longer skewed.

How to interpret a log model

- We are transforming the predicted values using the natural logarithm (ln) here. This is a logarithm to the base $e = 2.7182818$. You may remember from school that $e^{ln(x)} = x$ and $ln(e^x) = x$.
- Important: You can only log-transform positive and non-zero values. If you have zeroes or negative values in your dependent variable, you have to transform it.
- For example, to eliminate zeroes, you might add a tiny amount to all values.
- Log models are *multiplicative* rather than *additive*:
- $e^{x+y} = e^x \cdot e^y$, and $e^{ln(x)+ln(y)} = x \cdot y$
- Our model formula: $ln(y_i) = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i$
- Let's rewrite that: $y_i = e^{\alpha+\beta_1 x_{i1}+\beta_2 x_{i2}+\epsilon_i} = e^{\alpha} \cdot e^{\beta_1 x_{i1}} \cdot e^{\beta_2 x_{i2}}$

How to interpret a log model (2)

- Example: Our swear word fixation time study
  - Fitted model: $ln(y_i) = 5.683 - .195 \cdot x_i$
  - Remember: We're using treatment contrasts. x_i is 0 for non swear words and 1 for swear words

  - Predicted value for non swear words: $e^{5.683} \cdot e^{-.195 \cdot 0} = e^{5.683} = 293.82$
  - Predicted value for swear words: $e^{5.683} \cdot e^{-.195 \cdot 1} = 293.82 \cdot e^{-.195} = 293.82 * .823 = 241.81$
- Conclusion: Fixation times on swear words were 17.7% lower than fixation times on non swear words (b = -.195, SE = .074, t = -2.63, p = .011).

Summary

- If our data aren't normally distributed, we can sometimes transform them to get them closer to normality.
- A very common transformation uses the logarithm (usually the natural logarithm with base e, but others can be used to)
- You can't log-transform negative or zero values, so if your data contain any of those, you have to remove them or transform them (e.g. by adding a constant value) before doing the log-transformation
- The regression equation of a log-model is additive in its log format, but multiplicative when transformed back into raw values - e.g. $log(y) = \alpha + \beta X_i \leftrightarrow y = e^{\alpha+\beta X_i} = e^{\alpha} * e^{\beta X_i}$

Logistic regression

- What if we have a dichotomous dependent variable?
  - Yes vs. no, error vs. no error, alive vs. dead, pregnant vs. not pregnant
- Our example (from A. Johnson): Factors that make (or don't make) you fail your driving test
- 90 candidates
- Dependent variable: `Driving.Test`: Yes or No
- Predictor variables:
  - `Practice`: in hours
  - `Emergency.Stop`: Whether the candidate performed the emergency stop (yes or no)
  - `Examiner`: How difficult the examiner is (on a scale from 0 = easy to 100 = extremely difficult)
  - `Cold.Remedy`: How many doses of cold remedy the candidate had before the test

Examining the data

- These are our data (first 6 rows):

Driving.Test	Practice	Emergency.Stop	Examiner	Cold.Remedy
Yes	45	1	23	0
No	18	0	88	0
Yes	25	0	15	0
Yes	30	1	4	0
No	20	1	82	5
No	25	0	75	0

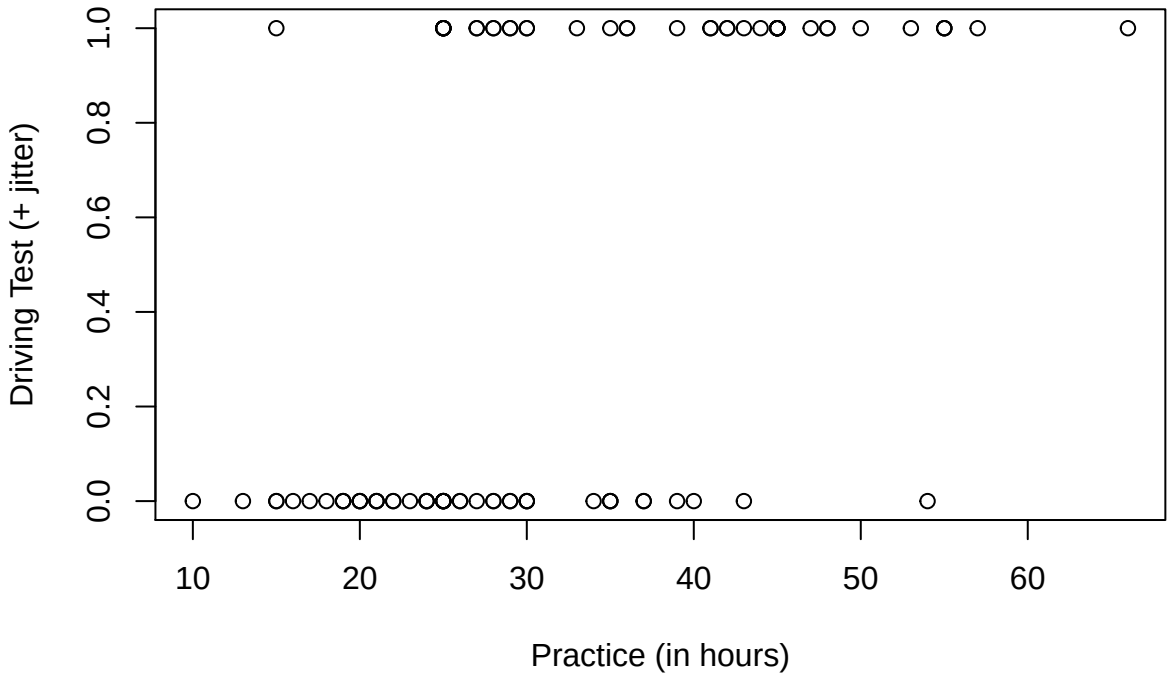
Where to start?

- First, let's recode our dependent variable, replacing "No" with 0 and "Yes" with 1

Driving.Test	Practice	Emergency.Stop	Examiner	Cold.Remedy
1	45	1	23	0
0	18	0	88	0
1	25	0	15	0
1	30	1	4	0
0	20	1	82	5
0	25	0	75	0

Let's plot it

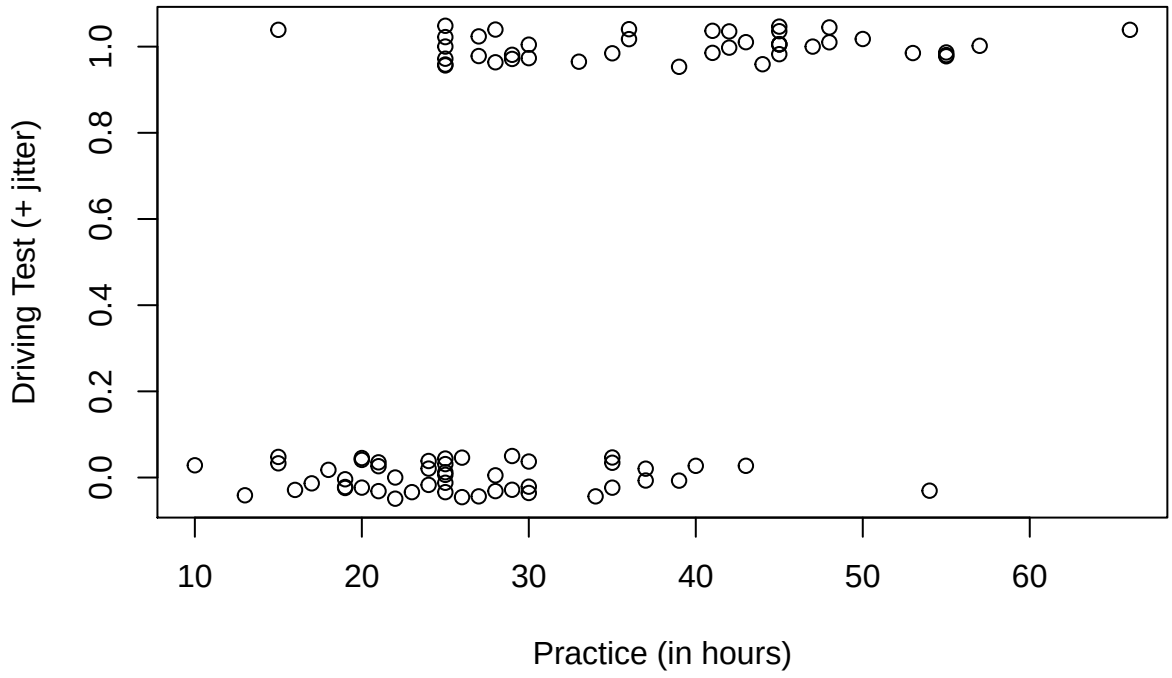
- Now we can make a plot of the situation
- Let's just consider the number of practice hours right now and ignore the other predictors



Adding some jitter

- Since the y-value is either 0 or 1, a lot of the points are plotted on top of each other (overplotting)

- To avoid this, we add or subtract a small amount to the y value ("jitter")
- Now we can see the pattern a little better.

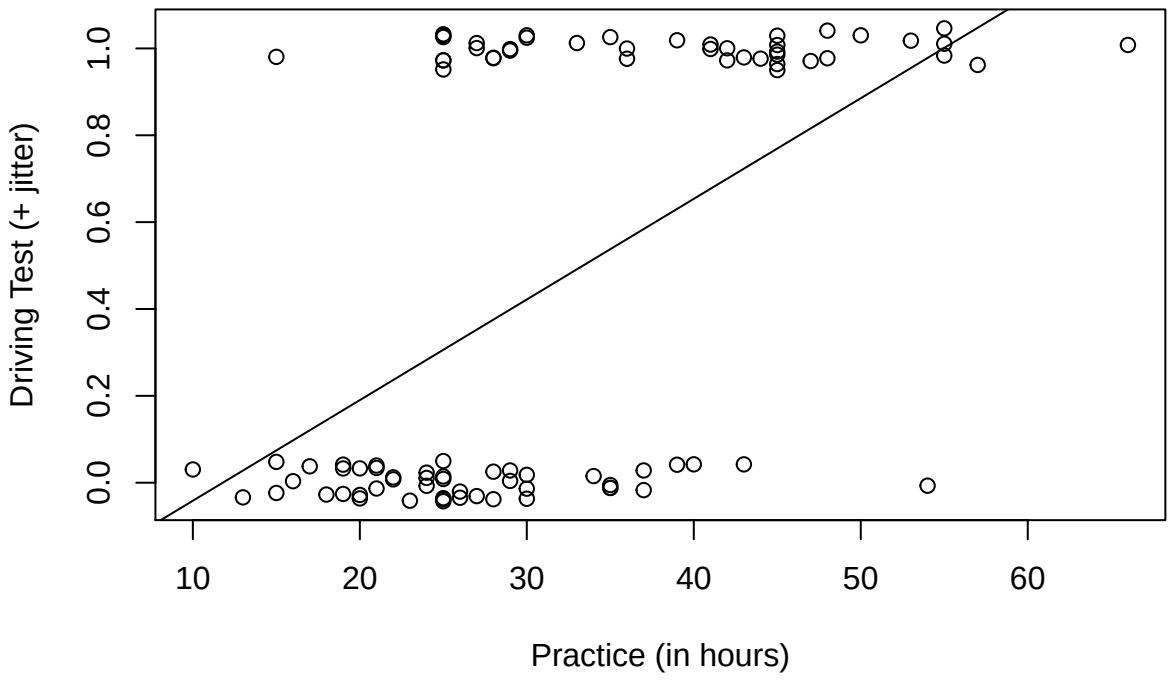


What do we want to estimate?

- The regression line gives us the expected $\hat{y}$-value (the mean y-value) for each value x_i
- In this case, if we code our dependent variable as 0 and 1, the mean (expected value) at each x value will give us the conditional probability $E(Y|x_i) = \pi_i$ of our event.
- Note that we're talking about estimating the theoretical probability in the population, hence the Greek letter. This is not the mathematical constant $\pi = 3.1415927$
- Then our model can give us predictions like this: What's the probability of passing the test given that I've had (at most) 20 hours of practice, did my emergency stop, had at most an average examiner (50) and had only one cup of cold remedy?

Predicting probabilities

- We can try a standard linear function
- This would be our model then: $P(Y_i = 1) = \alpha + \beta X_i + \epsilon_i$



Problems with a linear prediction

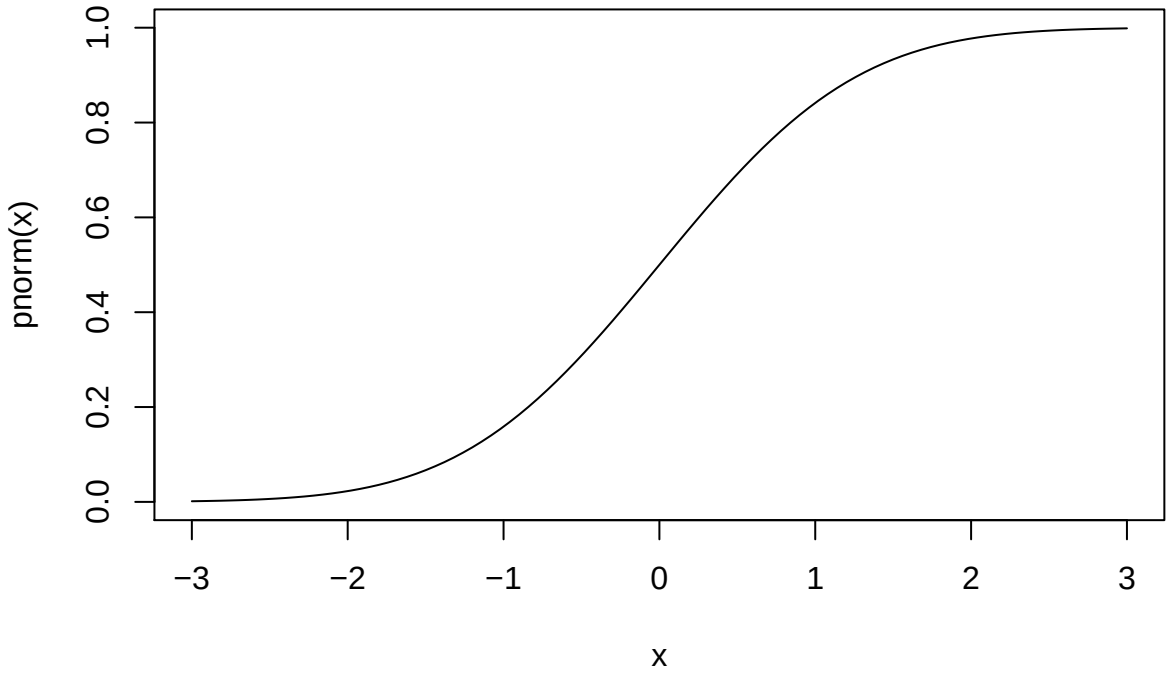
- But it's not very well suited!
- It predicts probabilities $P(Y_i = 1) < 0$ and $P(Y_i = 1) > 1$ for some x-values!
- Also, the error variance is clearly not equal for all x-values
- And we really can't pretend that our data are normally distributed.

Using a different regression function

- We would like a function that makes sure our linear predictor $\alpha + \beta x_i$ stays between 0 and 1
- This **link function** P would apply to our linear predictor and transform it into a probability: $\pi_i = P(\alpha + \beta x_i)$
- If the function is monotone, meaning that it consistently increases as $\alpha + \beta x_i$ gets larger, that would be advantageous, since the inverse of the function P^{-1} then links each value predicted by the linear model with the probability $P^{-1}(\pi_i) = \alpha + \beta x_i$
- There are two link functions that are most commonly used: the cumulative probability function of the normal distribution and the logistic function

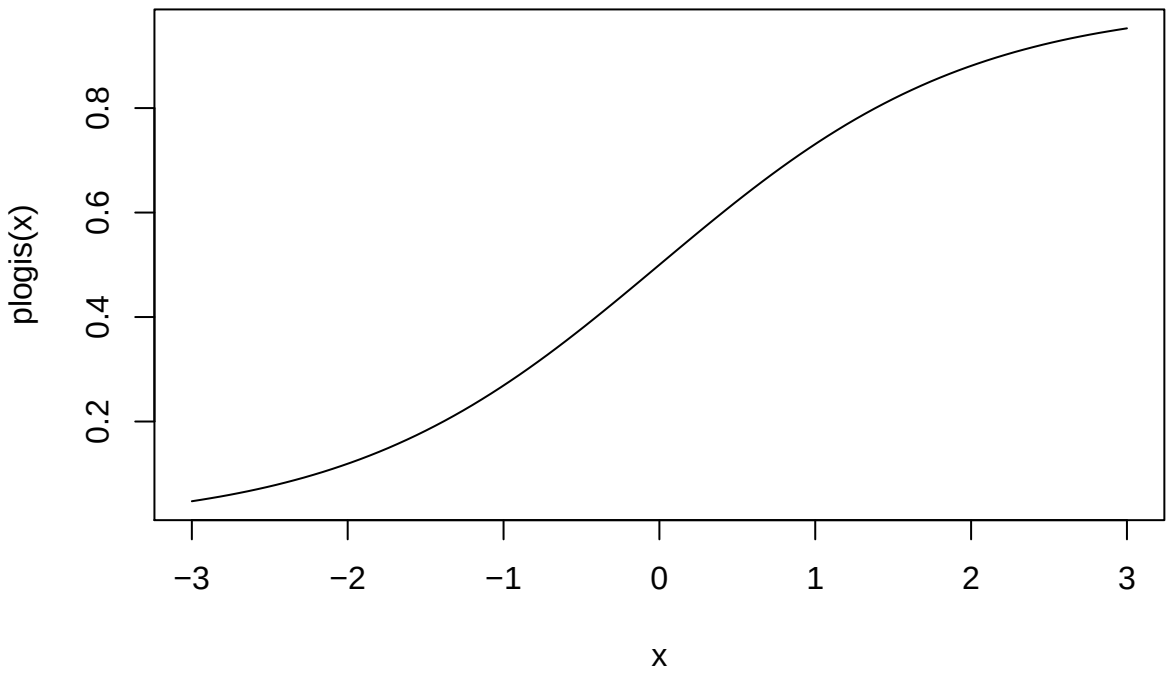
The cumulative probability function of the normal distribution

- Look up the function definition on Wikipedia, if you really care.



The logistic distribution function

- Here, the definition is simpler: $\Lambda(z) = \frac{1}{1+e^{-z}}$
- Where z is a z-value, and e is the constant $e = 2.7182818$. Λ is the capital letter Lambda.



Applying the link functions

- Applying the cumulative normal distribution function gives you the **linear probit model**.
- Applying the logistic distribution gives you the **linear logit model**, also called **linear logistic regression**:

$$\pi_i = \Lambda(\alpha + \beta x_i) = \frac{1}{1 + e^{-(\alpha+\beta x_i)}}$$

- Both functions work equally well, but the logistic distribution function has the advantage that the transformed values are directly interpretable:
- Rewrite the above equation and you get **odds**:

$$\frac{\pi_i}{1 - \pi_i} = e^{\alpha+\beta x_i}$$

Probability and odds

- Very popular in betting, since they make it easy to estimate the payout
- $Odds = \frac{P}{1-P}$
- For example:
  - $P = .5$ gives you even odds $\frac{.5}{.5} = 1/1$
  - $P = .25$ gives you $\frac{.25}{.75} = 1/3$
  - $P = .75$ gives you $\frac{.75}{.25} = 3/1$

- Odds are nice because they aren't bounded, but for high probabilities they get very large very quickly ($P = .99 \Leftrightarrow Odds = 99/1$) and for small probabilities, they get very small very quickly ($P = .01 \Leftrightarrow Odds = 1/99$)

Log odds (logits)

- Just transform our odds (just like we did with our continuous fixation time variable earlier) by taking the natural logarithm: $logit = ln(Odds) = ln(\frac{P}{1-P})$
- Now we have a dependent measure that is suitable for linear relationships
- Log odds is just what we happen to get if we apply the natural logarithm on both sides of our logistic regression equation from earlier:

$$\frac{\pi_i}{1-\pi_i} = e^{\alpha+\beta x_i}$$
$$\leftrightarrow ln(\frac{\pi_i}{1-\pi_i}) = ln(e^{\alpha+\beta x_i})$$
$$\leftrightarrow ln(\frac{\pi_i}{1-\pi_i}) = \alpha + \beta x_i$$

The logistic model

- So, our full model (including errors) is

$$logit_i = ln(\frac{\pi_i}{1-\pi}) = \alpha + \beta x_i + \epsilon_i$$

- If we want to get back from logits to probabilities, we can do that by reversing the transformation:

$$\pi_i = \frac{1}{1 + e^{-(logit_i)}}$$

How do you fit a logistic regression line?

- Sadly, you can't fit the logistic regression model using the least squares approach (the errors still aren't nromally distributed or homoscedastic)
- We can evaluate the **likelihood** of the parameters instead
- In statistics, probability and likelihood aren't the same:
- Probability: observations given parameters
- Likelihood: parameters given observations

Calculating likelihood

- Very simple example: Let's assume we have the following data from flipping a coin: $Y = (HHHTHTHH)$
- Likelihood is the product of all the probabilities given a certain parameter value. In this case, we are trying different parameter values for the probability: -We're interested in the population parameter, so again, our population probability of observing "heads" (ignoring order) will be called π:
- $\pi = .5$: $L(Y|\pi = .5) = .5 \cdot .5 \cdot .5 \cdot .5 \cdot .5 \cdot .5 \cdot .5 \cdot .5 = .5^8 = .00039$
- $\pi = .25$: $L(Y|\pi = .25) = .25 \cdot .25 \cdot .25 \cdot .75 \cdot .25 \cdot .75 \cdot .25 \cdot .25 = .25^6 \cdot .75^2 = .00014$
  - $\pi = .25$ has a lower likelihood than $\pi = .5$.
- Let's try $\pi = .75$: $L(Y|\pi = .75) = .75 \cdot .75 \cdot .75 \cdot .25 \cdot .75 \cdot .25 \cdot .75 \cdot .75 = .75^6 \cdot .25^2 = .00111$
  - This is the highest one yet.

The likelihood function for logistic regression

- Do you see a pattern here? For each element Y_i in Y, the likelihood of π is either
  - $L(Y_i|\pi) = \pi$ if $Y_i = H$, (e.g. .75 for $\pi = .75$), or
  - $L(Y_i|\pi) = 1 - \pi$ if $Y_i = T$, (e.g. .25 for $\pi = .75$)
- Then you get the likelihood for the full data set Y by multiplying all the individual likelihoods
  - $L(Y|\pi) = \prod_{i=1}^{N} L(Y_i|\pi)$
- You can simplify this a bit if you replace H with 1 and T with 0:
  - $L(Y_i|\pi) = \pi^{Y_i} \cdot (1-\pi)^{(1-Y_i)}$
  - And combining the two equations above:
    - $L(Y|\pi) = \prod_{i=1}^{N} \pi^{Y_i} \cdot (1-\pi)^{(1-Y_i)}$

Maximum likelihood

- Likelihood gets a little unwieldy – lots of very small numbers
  - Solution: take the log (who would have thought?)
    - Added bonus: Now our multiplication becomes a sum (remember that from calculus?) $log\ likelihood = \sum_{i=1}^{N} Y_i \cdot ln(\pi) + (1 - Y_i) \cdot ln(1 - \pi)$
- Now we can simply try different values of π until we find the one with the maximum likelihood
  - Remember that in logistic regression, π is defined by our regression equation: $\pi = \frac{1}{1+e^{-(\alpha+\beta x_{i1}+\epsilon_i)}}$
  - Instead of simply trying different values of π, we have to try different values for α, β_1, etc. and compute π. This gets to be quite a lot of work.

Fitting the model through an iterative process

- Trying different values might not seem particularly elegant, but this is essentially what R or SPSS do – no simple solution like with least-squares regression or ANOVA exists
- This is (relatively) processing-intensive, which is one reason why psychologists didn't use logistic regression in the statistics-by-hand era.

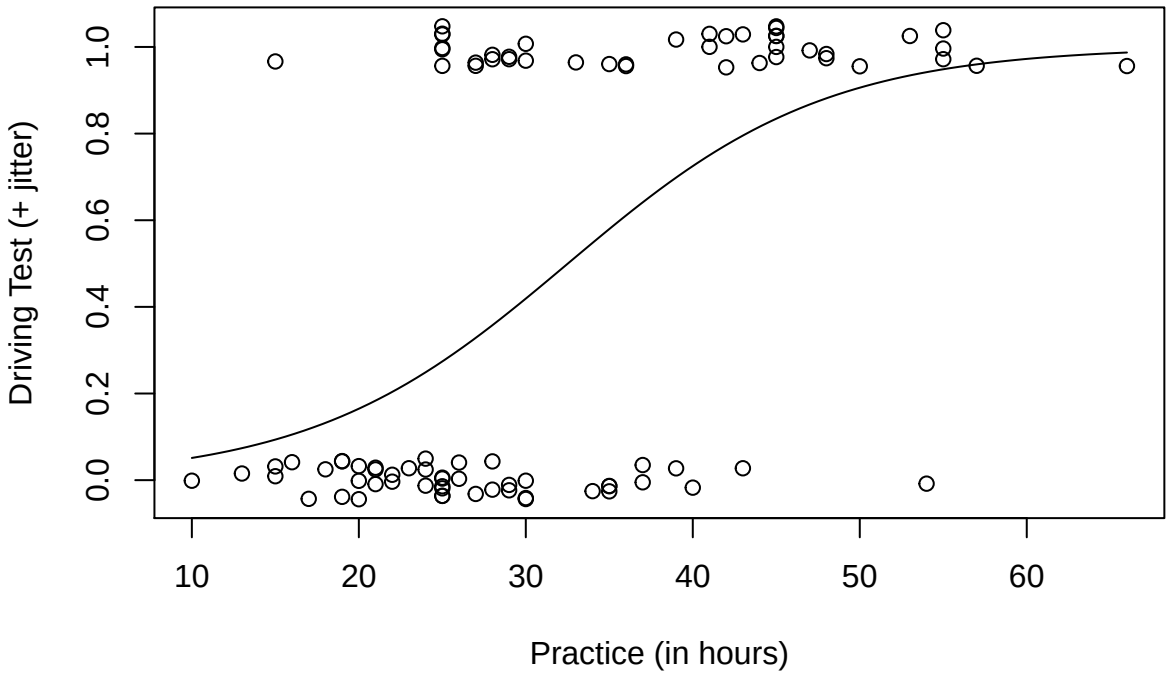
Log likelihood as an indicator of model fit

- The log likelihood (LL) of the final model is an indicator of how well the model fit the data, just like R^2.
  - In fact, there are several ways to estimate R^2 from the log likelihood
- Log likelihood also enables us to make model comparisons
  - The test statistic in that case is χ^2 – more about that later
- Another measure is *deviance*, which is simply $-2 \cdot LL$
  - Conceptually, deviance is like the residual variance.
    - In the case of deviance, lower is better, of course.
- Closely related to this is Akaike's Information Criterion (AIC), which is $-2 \cdot LL + 2 \cdot$number of parameters (lower is better, so adding parameters makes the AIC worse)

How to fit the model in SPSS

- You can't do this in Excel anymore.
- To see how it works, watch Andy Johnson's video on myBU.

Plotting the model fit



Model summary

- The coefficients

	E	stimate Std	. Error	z value Pr	(>|z|)
(Intercept)		-2.3562402	1.0301979	-2.2871724	0.0221858
Practice		0.1295943	0.0302820	4.2795819	0.0000187
Emergency.StopNo		-0.0494204	0.5912031	-0.0835929	0.9333802
Examiner		-0.0348487	0.0130063	-2.6793837	0.0073758
Cold.Remedy		-0.0764577	0.1685272	-0.4536815	0.6500581

Significance tests for coefficients

- Analogous to linear regression: b value, SE, significance test
- Using the **Wald** statistic instead of t tests
  - $z = \frac{b}{SE_b}$
  - We can use z-values because our dependent variable comes from the binomial distribution. In the binomial distribution, the variance is always a function of the mean, so we don't have to estimate it from the sample (and since we're not using the sample variance to estimate the population variance, we don't need to use a t-test).

Interpreting the b values

- If you take the exponential of the coefficients , you get an **odds ratio**
- Odds ratio = Odds after a unit change in the predictor divided by the original odds
- Example: According to our model, each hour of practice increases the odds of passing the test by a factor of $e^{0.12959} = 1.1383616$
  - e.g. if the odds were even (1/1) for X hours of practice, they would be slightly better than even (1.138/1) for X+1 hours of practice
- On the other hand, each "unit" of examiner difficulty decreases the odds of passing the test by a factor of $e^{-0.03485} = 0.9657503$
  - e.g. if the odds were even (1/1) for an instructor with a difficulty of X, they would be slightly worse than even (0.9658/1) for an instructor with a difficulty of X+1

Model comparisons (LRT)

- Deviance has some neat properties
  - We can compare likelihoods just like we compared mean squares in the F-test: by dividing them
  - That is, we compute a likelihood ratio: $LR = \frac{L_{baseline}}{L_{new}}$, where *baseline* is the simpler model and *new* the more complex model.
  - Now we can convert this likelihood ratio into a deviance: $deviance_{LR} = -2\frac{ln(L_{baseline})}{ln(L_{new})} = deviance_{baseline} - deviance_{new}$
  - And now the most fun part: If the H_0 that the two models explain the data equally well is true, this likelihood-ratio deviance is distributed according to a χ^2 distribution.
  - The χ^2 distribution has one parameter, degrees of freedom
  - $df = k_{new}$ - $k_{baseline}$, where k is the number of parameters (including the intercept)

Model comparisons

- Now we can get a p-value! This is called the **Likelihood ratio test (LRT)**
- So, if we want to test if the model is better than a model with just the intercept, we can do an LRT
- $\chi^2 = deviance_{null} - deviance_{model} = 124.366 - 82.572 = 41.794$
- $df = k_{null} - k_{model} = 89 - 85 = 4$
- $p(\chi^2(4) \geq 41.794) < .01$
- This is equivalent to the overall F-test for the model.
- SPSS calls this the "Omnibus test"

Model comparisons (2)

- We can use model comparisons to test how specific predictors contribute to the whole model (analogous to the F-tests in linear regression)

```
## Analysis of Deviance Table (Type II tests)
##
## Response: Driving.Test
##                LR Chisq Df Pr(>Chisq)
## Practice        28.9305  1  7.502e-08 ***
## Emergency.Stop   0.0070  1   0.933367
## Examiner         8.1129  1   0.004395 **
## Cold.Remedy      0.2160  1   0.642120
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model comparisons (3)

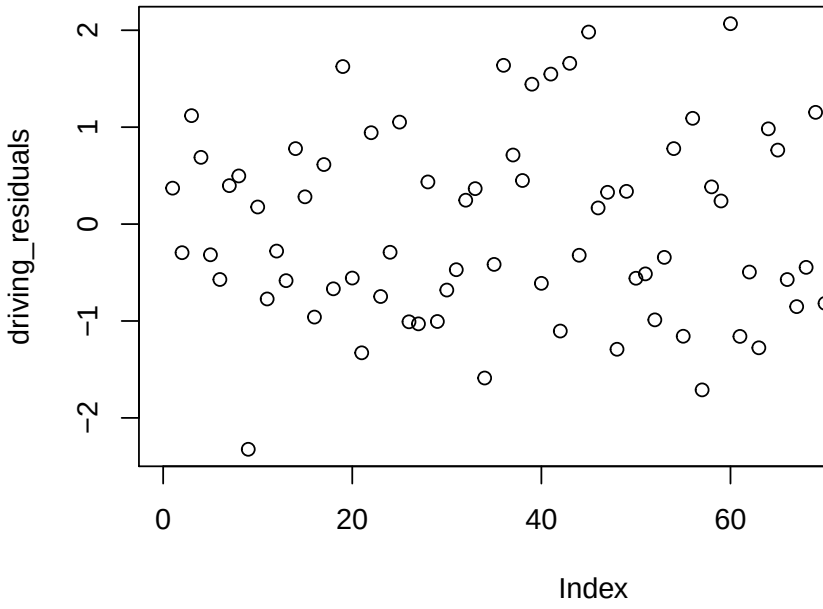
- This analysis of deviance follows the same logic as the ANOVA in the linear regression case
- You can do Type I, Type II, and Type III LRT tests (they are not sums of squares in this case)
- The LRTs are better tests than the Wald tests, since the Wald tests might be prone to overinflating the SE, leading to Type II error.

Diagnostics and assumption tests

- We do not assume normality (so nothing to test for that one)
- All the influence measures from linear regression work in logistic regression as well
- Multicollinearity:
- You can get Variance Inflation Factors (VIFs) and Tolerance etc. in SPSS if you fit the standard linear model first
  - Of course, don't interpret the coefficients that you get!
- You can also take a look at the Hosmer-Lemeshow test (see Andy Johnson's video)

Looking at logistic regression residuals

- Can't test if they are normally distributed (because they are not)
- But look out for very large residuals
- You can get still get standardised residuals.



- Look out for cases that are far away from 0.

Reporting it

A logistic regression was conducted where the dependent variable was passing a driving test and the predictor variables were hours of practice, whether an emergency stop was successfully executed, how much the examiner was difficult, and amount of 'cold remedy drinks' consumed. 90 cases were examined and the model was found to significantly predict whether the test was passed (omnibus chi-square = 41.79, df=4, p<.001). that practice and examiner were the only two variables that reliably predicted if the driving test was passed. Increases in practice was associated with increased rate of passing (odds of passing increased by 1.14 per hour of practice, $b = .130$, SE = .03, $z = 4.28$, $p < .01$). Increases in the examiner difficulty reduced the rate of passing (odds of passing decreased by 0.96 per unit of difficulty rating, b = -.00349, SE = .013, z = -2.679, $p < .01$). None of the other predictors reached significance (all ps $> .05$). There were no issues due to multicollinearity or influential cases.

Summary: Logistic regression

- If you have a dichotomous outcome variable (e.g. Yes vs. No, Heads vs. Tails, etc.), you cannot use standard linear regression
- Instead, you must use a $link function$ to convert your predicted values into probabilities and vice versa. The most commonly used are the cumulative distribution function (CDF) of the normal distribution and the CDF of the logistic distribution
  - if you use the CDF of the normal distribution, you are fitting a **probit** model
  - if you use the CDF of the logistic distribution, you are fitting a **logit** or logistic regression model
- The logit model has the advantage that you can interpret the coefficients directly as changes in log odds = $ln(frac\pi_i 1 - \pi_1)$

- Just like when you are using a log transformation, the coefficients are additive as far as log odds are concerned, but multiplicative when you convert the log odds into odds.
- In both cases, you use the Wald test as a significance test for coefficients (analogous to t-tests in standard linear regression) and the likelihood ratio test (LRT, analogous to F-tests in standard linear regression) in order to test the significance of predictors.
  - Since we don't have to estimate the error variance from the data, the Wald statistic is z-distributed and the LRT statistic is χ^2 distributed.
  - Just like the standard oneway/multiway ANOVA is a special case of linear regression with only categorical predictors, the χ^2-test is a special case of logistic regression with only categorical predictors.

Linear mixed models (LMMs)

- The final step to greatness!
  - Note that we can really only scratch the surface here.
- Main issue:
  - We know how to to regressions for continuous and discrete DVs now
  - We know what the regression equivalent of a between-subjects ANOVA is and we can take the regression analysis much further than an ANOVA or traditional ANCOVA would let us
  - However:
    * What if we have a within-subject or repeated measures design?
    * What if there is some other underlying correlation in the data
    * e.g. data collected from students in different classes in different schools
    * Surely the classes and schools share some variance – how to account for that?

Moving from linear regression to linear mixed models

- In repeated-measures ANOVA, we've dealt with within-subjects effects by removing the variance due to subject differences from the error
  - Essentially, we have added a "subject" factor to the model
  - Linear mixed models enable us to do the same thing for all kinds of regression analyses

Problem: how to add subject as a factor

- We could simply add a "subject" factor to the predictors
  - This would reflect the systematic differences between subjects
    * But that's not quite right: how do we deal with a factor with 40 levels?
    * Also, we want to generalise our model to more than those 40 subjects that are in the analysis
    * How do we do that?
  - Subject is really like a random variable: we get a different set each time we run the experiment
  - Instead of analysing the subject effect in a generalisable way, we really just want to get rid of the subject variance in the most efficient way possible

Problem: how to add subject as a factor (2)

- Fixed effects vs. random effects
  - Fixed effects: repeatable, generalisable (e.g. experiment condition)
  - Random effects: non-repeatable, sampled from a general population
  - Mixed effects models include both fixed and random effects
- Another issue with including subject as a fixed effect:
  - Each subject would take up a degree of freedom
  - That would majorly impact the power of our analysis
  - LMMs solve this issue by a procedure called **shrinkage**

Shrinkage

- Conceptually, LMMs allow subjects to have individual effects (e.g. in an eye-tracking experiment subject 1 might have an intercept of 200 ms, while subject 2 might have an intercept of 210 ms), but they pull each subject's effects towards an underlying distribution of subject effects
- This reflects the idea that if 20 other subjects have intercepts between 180 and 220 ms, the current subject is unlikely to have an intercept of 400 ms, even though it looks like that from the data
- Shrinkage also helps majorly with missing data issues (although it won't fix them for you!)
- The downside of shrinkage is that it isn't clear what the df_{Error} should be
  - This leads to some issues later on.

Example

A PhD student wants to investigate whether our mood affects how we react to visual scenes. In order to do this, she showed 40 subjects a total of 40 scenes. There are two version of each scene: one contains people, the other one doesn't – everything else is identical. The PhD student spent a considerable amount of time taking photos to ensure this (until her supervisor got a bit impatient). Before the experiment, all subjects were asked to rate their current mood on a scale from 0 (very sad) to 100 (very happy). They then looked at each scene and rated how much they liked it on a scale from 0 (hate it) to 20 (love it). The student's hypothesis is that if you are happy, you should want to see scenes with people. If you are unhappy, you should prefer scenes without people. The data are given below. Will the student find what she is looking for? Or will she have to start from scratch and be in even more trouble with her supervisor?

Example Data

- Subject: Subject ID (1-40)
- Item: Item ID (1-40)
- Scene Type: within-item factor ("no people" vs. "people")
- Mood: between-subject factor (scale from 0–100)
- Rating: Dependent variable (scale from 0 to 20)
- For these kinds of data, R has a massive advantage over SPSS in terms of usability.
- But you can still get the same results in SPSS!
- Don't be scared, I'll be walking you through this with R first and then show you how to do it in SPSS.

```r
# Start by loading the data
scene_liking <- read.csv("Class 6 exercise data.csv")
```

Looking at the data

```r
kable(head(scene_liking))
```

subject	item	scene	mood	rating
1	1	people	55	12.2
1	2	no people	55	8.5
1	3	people	55	10.7
1	4	no people	55	12.0
1	5	people	55	12.3
1	6	no people	55	16.0

Looking at the data (2)

```r
str(scene_liking)
```

```
## 'data.frame':    1600 obs. of  5 variables:
##  $ subject: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ item   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ scene  : Factor w/ 2 levels "no people","people": 2 1 2 1 2 1 2 1 2 1 ...
##  $ mood   : int  55 55 55 55 55 55 55 55 55 55 ...
##  $ rating : num  12.2 8.5 10.7 12 12.3 16 8.2 13.2 6.7 6.9 ...
```

```r
# We should set subject and item to be 0 (nominal scale)
scene_liking$subject <- factor(scene_liking$subject)
scene_liking$item <- factor(scene_liking$item)
```

Calculating means

- Let's get condition means for scene type
- In theory, we could report means by subject or means by item
- Either one would be fine, but usually people report subject means.
  - We use `melt` and `cast` from the `reshape` package to calculate the means

```r
library(reshape)
# set rating as the dependent (measure) variable
scene_liking.m <- melt(scene_liking, measure = "rating")
# collapse over item; calculate means
scene_liking.c <- cast(scene_liking.m, subject + mood + scene ~ variable, mean)
```

Calculating means (2)

```
head(scene_liking.c)
```

```
##   subject mood     scene rating
## 1       1   55 no people 10.870
## 2       1   55    people 10.355
## 3       2   56 no people 10.910
## 4       2   56    people 10.785
## 5       3   16 no people 12.330
## 6       3   16    people 11.540
```

Calculating means (3)

- Now we can use this to calculate our means for the scene condition

```
(mean_people <- mean(subset(scene_liking.c, scene == "people")$rating))
```

```
## [1] 11.11975
```

```
(mean_no_people <- mean(subset(scene_liking.c, scene == "no people")$rating))
```

```
## [1] 11.32825
```

Calculating means (3)

- Let's also get sd, N, and SE

```
(sd_people <- sd(subset(scene_liking.c, scene == "people")$rating))
```

```
## [1] 0.6141577
```

```
(sd_no_people <- sd(subset(scene_liking.c, scene == "no people")$rating))
```

```
## [1] 0.6162142
```

```
(N_people <- length(subset(scene_liking.c, scene == "people")$rating))
```

```
## [1] 40
```

```
(N_no_people <- length(subset(scene_liking.c, scene == "no people")$rating))
```

```
## [1] 40
```

```
(SE_people <- sd_people/sqrt(N_people))
```

```
## [1] 0.09710686
```

```
(SE_no_people <- sd_no_people/sqrt(N_no_people))
```

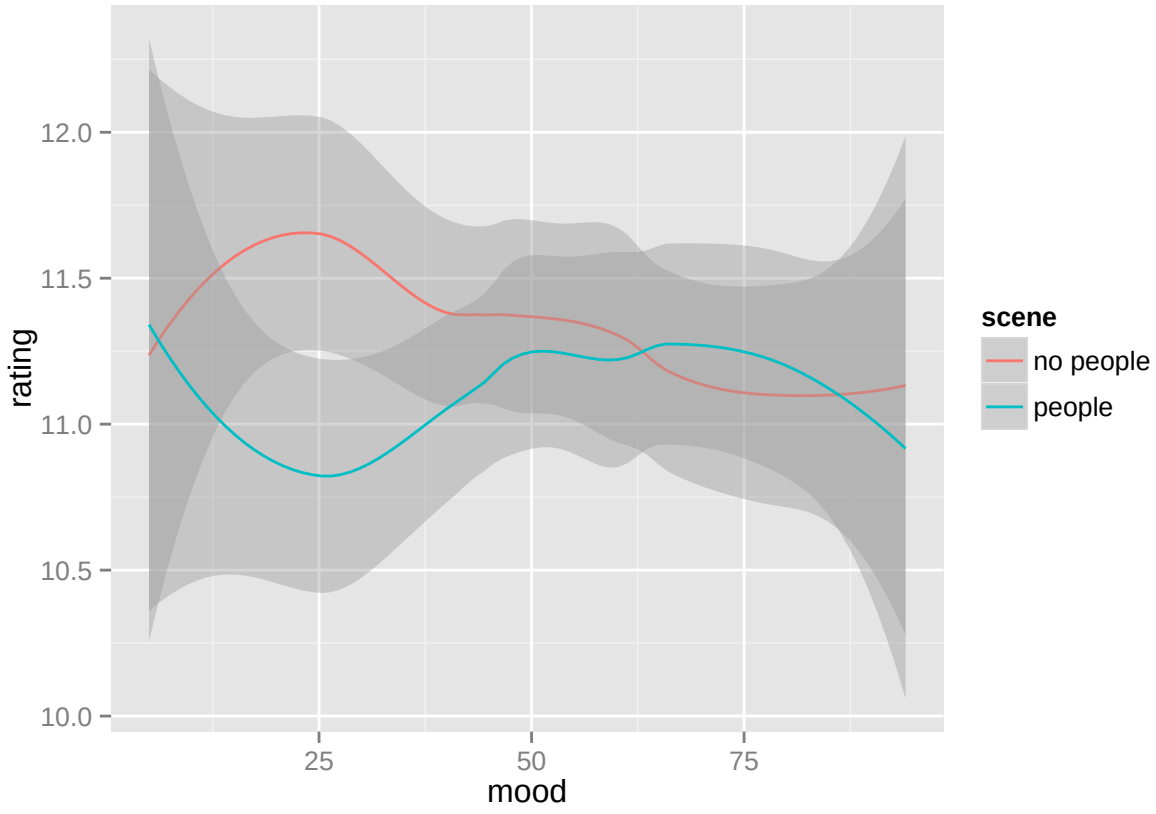
```
## [1] 0.09743202
```

Plotting the interaction

- We're really interested in the interaction between `scene` and `mood`.
  - Unfortunately, mood is a continuous variable
  - How to plot this?
- Use `qplot` from `ggplot2` with `geom = "smooth"`
  - This will give you a plot showing a smoothed conditional mean for each value of mood

Plotting the interaction (2)

```
library(ggplot2)
qplot(data = scene_liking, y = rating, x = mood, colour = scene, geom = "smooth")
```



Plotting the interaction (2)

- Look at how amazingly pretty that is. You just can't do stuff like that with SPSS.
- Looks like the student was right!
- Also looks like the effect is not really completely linear
  - Maybe this is due to the subject and item effects in the data?
  - Let's find out!

Start with linear regression

- Let's check our contrasts for `scene`

```
contrasts(scene_liking$scene)
```

```
##           people
## no people      0
## people         1
```

- Are we happy with this?
  - Sure – we just have to be aware of the coding when we interpret the coefficients

Run the model

```
scene_lm <- lm(data = scene_liking, rating ~ mood * scene)
kable(coef(summary(scene_lm)))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	11.7066203	0.2475307	47.293603	0.0000000
mood	-0.0073399	0.0044471	-1.650467	0.0990441
scenepeople	-0.7583405	0.3500613	-2.166308	0.0304353
mood:scenepeople	0.0106662	0.0062892	1.695945	0.0900914

- Where did the interaction go?
- Let's do some quick diagnostics

Regression diagnostics

- Multicollinearity?

```
vif(scene_lm)
```

```
##       mood      scene mood:scene
##   2.000000   7.029992   8.029992
```

- Aha! Those VIFs are quite a bit larger than 1. That spells trouble.
- What is wrong?

Addressing multicollinearity

- What is wrong?
- We forgot to center the continuous predictor `mood`
- Let's fix this:

```
scene_liking$mood <- scale(scene_liking$mood, scale = FALSE) # See Class 5
```

Run the model again

```
scene_lm <- lm(data = scene_liking, rating ~ mood * scene)
kable(coef(summary(scene_lm)))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	11.3282500	0.0933580	121.341991	0.0000000
mood	-0.0073399	0.0044471	-1.650467	0.0990441
scenepeople	-0.2085000	0.1320282	-1.579208	0.1144865
mood:scenepeople	0.0106662	0.0062892	1.695945	0.0900914

- Still not quite there...
- Let's do more diagnostics

Regression diagnostics – again

- Multicollinearity?

```
vif(scene_lm)
```

```
##       mood      scene mood:scene
##          2          1          2
```

- The VIFs are fine now.

Influential cases

```
kable(head(influence.measures(scene_lm)$infmat))
```

dfb.1_	dfb.mood	dfb.scnp	dfb.md:s	dffit	cov.r	cook.d	hat
0.0000000	0.0000000	0.0101292	0.0016646	0.0145170	1.0033862	0.0000527	0.0012838
-0.0375790	-0.0061758	0.0265724	0.0043670	-0.0380831	1.0009635	0.0003626	0.0012838
0.0000000	0.0000000	-0.0040867	-0.0006716	-0.0058570	1.0037317	0.0000086	0.0012838
0.0093426	0.0015354	-0.0066062	-0.0010857	0.0094679	1.0036233	0.0000224	0.0012838
0.0000000	0.0000000	0.0110770	0.0018204	0.0158754	1.0033054	0.0000630	0.0012838
0.0630143	0.0103559	-0.0445578	-0.0073227	0.0638595	0.9958519	0.0010181	0.0012838

- Any Cook's d greater than 1?

```
sum(cooks.distance(scene_lm) > 1 )
```

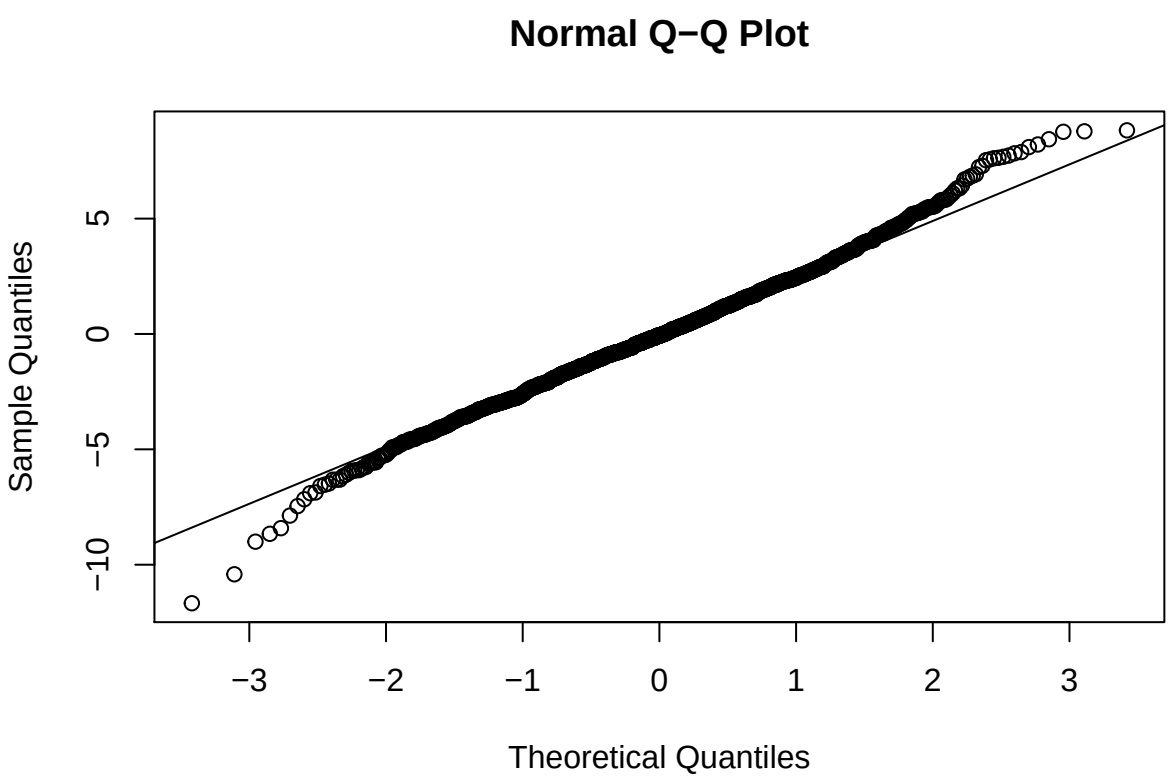
```
## [1] 0
```

- Doesn't look like it, so we should be fine here.

Q-Q Plots

- Here's a visual way to assess normality
- Quantile-Quantile Plot: Split data into a number of quantiles and plot them against the quantiles of a hypothetical normal distribution

```
qqnorm(resid(scene_lm))
# if the distribution is normal, the points should be on this line
qqline(resid(scene_lm))
```



How to fix this?

- As a first step, remember that there are subject and item effects in these data
- `lm` can't account for them, so we need something more powerful
- Linear Mixed Models!
- We use the function `lmer` ("Linear mixed effects regression") from the `lme4` package
- If you don't have `lme4` yet, install it by typing `install.packages("lme4")` in the Console

Adding random subject and item effects

- As a first step, we want our model to allow subjects and items to have different intercepts
  - For example, Subject 1 might just really dislike the whole experiment and rate all scenes lower
  - Or Item 33 might be particularly ugly and be disliked by all subjects
- Formally, our model will look like this: $y_{ij} = \alpha + \beta_1 x_1 + \beta_2 x_2 + \gamma_{0i} + \gamma_{0j} + \epsilon_{ij}$, where y_{ij} is the response of subject i to item j, γ_{0i} is the intercept for subject i and γ_{0j} is the intercept for item j

Running the model

- In `lmer`, we specify the model in a formula just like in `lm`, but we add random effects terms, e.g. `(1|subject)`

- The left side of the pipe stands for the random effect, the right side stands for the group for which we want to define the random effect
- 1 stands for the intercept. It is implicitly added, *except* when there is no other predictor

```
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following object is masked from 'package:reshape':
##
##     expand
```

```
scene_lmm <- lmer(data = scene_liking, rating ~ scene * mood + (1|subject) + (1|item))
```

Examining the model

```
summary(scene_lmm)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: rating ~ scene * mood + (1 | subject) + (1 | item)
##    Data: scene_liking
##
## REML criterion at convergence: 7656.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.3775 -0.6168 -0.0294  0.6378  3.5409
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  subject  (Intercept) 0.08724  0.2954
##  item     (Intercept) 0.13101  0.3620
##  Residual             6.76151  2.6003
## Number of obs: 1600, groups:  subject, 40; item, 40
##
## Fixed effects:
##                   Estimate Std. Error t value
## (Intercept)      11.328250   0.117932   96.06
## scenepeople      -0.208500   0.130014   -1.60
## mood             -0.007549   0.004923   -1.53
## scenepeople:mood  0.011085   0.006227    1.78
##
## Correlation of Fixed Effects:
##             (Intr) scnppl mood
## scenepeople -0.551
## mood         0.000  0.000
## sceneppl:md  0.000  0.000 -0.632
```

Understanding the model output

- Just like in logistic regression, LMMs are fitted in an iterative procedure using Maximum Likelihood (ML)
  - Actually, `lmer` by default uses a slightly modified criterion called Restricted Maximum Likelihood (REML)
- Residuals can be interpreted just like in a regular linear model
- Random effects: Here we get an estimate of the variance (and standard deviation) explained by the random intercepts
  - We also get an estimate of the residual variance
- Check the number of observations to see if there are any missing that shouldn't be missing

Coefficients

```
kable(coef(summary(scene_lmm)))
```

	Estimate	Std. Error	t value
(Intercept)	11.3282500	0.1179322	96.057280
scenepeople	-0.2085000	0.1300145	-1.603667
mood	-0.0075493	0.0049226	-1.533596
scenepeople:mood	0.0110849	0.0062270	1.780145

- First thing you notice: There's no p value, even though the Wald statistic should follow a t-distribution
- That's because, due to the shrinkage procedure, it isn't clear what the df of that t-value should be
- In general, if the number of subjects is > 30, we should be able to interpret the t value like a z value, meaning that any $t > 2$ or < -2 should be significant
- SPSS uses a procedure called the Satterthwaite approximation for coming up with degrees of freedom for the t-values

Correlation of fixed effects

- These are the estimated correlations of the fixed effects
  - If any of these is $> .8$, you're in multicollinearity trouble!

Model comparisons

- Unfortunately, F-tests won't work, because we don't know what the df_{Error} would be
- But we can do the likelihood ratio test (LRT)
- As always, we use `Anova` from `car`. This one gives us p values!

```
library(car)
Anova(scene_lmm)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: rating
##             Chisq Df Pr(>Chisq)
## scene      2.5717  1    0.10879
## mood       0.2770  1    0.59867
## scene:mood 3.1689  1    0.07505 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

More model diagnostics

- Something still seems to be wrong with this model. How about testing the normality assumption again?

```
shapiro.test(resid(scene_lmm))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(scene_lmm)
## W = 0.99388, p-value = 3.583e-06
```

- Still significant? Maybe there still is a random effect that we haven't accounted for.

Random slopes

- We can also allow the regression slopes to vary by subject or item.
- What are possible random slopes that we could consider?
  - Important: in theory, you could add random slopes for all fixed effects, but in practice, your data might not have enough information to fit these
  - In this case, there simply isn't enough data to fit random slopes for the interaction
  - How do you know this?
  - Well, your model will simply fail to converge if there is not enough data for a solution!
  - Even if there *is* enough data, multicollinearity can cause convergence failures, too.
  - In our case, some reasonable random slopes would be `scene` for subjects (do some people react more strongly to scenes with people than others) and `mood` for items (are some items really hated by people in a bad mood?)

Random slopes (2)

- If we include a random slope for subjects for $beta_1$, our model looks like this: $y_{ij} = \alpha + \beta_1 x_1 + \beta_2 x_2 + \gamma_{0i} + \gamma_{1i} x_1 + \gamma_{0j} + \epsilon_{ij}$
- We can tell `lmer` to fit such models like this (note that the intercept is implicit again in `(mood|item)` and `(scene|subject)`.
  - Note that we don't have enough data to include both random effects in one model.

```
scene_lmm_mood <- lmer(data = scene_liking,
                       rating ~ scene * mood + (1|subject) + (mood|item))
scene_lmm_scene <- lmer(data = scene_liking,
                        rating ~ scene * mood + (scene|subject) + (1|item))
```

Testing the effect of random slopes

- We can use the LRT to test whether the slopes actually improve the models.
  - We use the `anova` command (lower case `a`) to compare each random slope model with the random intercept model we fitted earlier

```
anova(scene_lmm, scene_lmm_mood)
```

```
## refitting model(s) with ML (instead of REML)

## Data: scene_liking
## Models:
## scene_lmm: rating ~ scene * mood + (1 | subject) + (1 | item)
## scene_lmm_mood: rating ~ scene * mood + (1 | subject) + (mood | item)
##                Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## scene_lmm       7 7648.0 7685.6 -3817.0   7634.0
## scene_lmm_mood  9 6890.5 6938.9 -3436.3   6872.5 761.46      2  < 2.2e-16
##
## scene_lmm
## scene_lmm_mood ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Seems to improve the model! (Note that R automatically uses ML instead of REML for model comparison)

Testing the effect of random slopes

```
anova(scene_lmm, scene_lmm_scene)
```

```
## refitting model(s) with ML (instead of REML)

## Data: scene_liking
## Models:
## scene_lmm: rating ~ scene * mood + (1 | subject) + (1 | item)
## scene_lmm_scene: rating ~ scene * mood + (scene | subject) + (1 | item)
##                 Df  AIC    BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## scene_lmm        7 7648 7685.6  -3817     7634
## scene_lmm_scene  9 7652 7700.4  -3817     7634 0.0019      2     0.9991
```

- No improvement here.

Diagnostics – yet again!

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(scene_lmm_mood)
## W = 0.99817, p-value = 0.07537
```

- Looks like adding a random slope for `mood` also (mostly) fixed our normality problem

Interpreting the coefficients

```
kable(coef(summary(scene_lmm_mood)))
```

	Estimate	Std. Error	t value
(Intercept)	11.3554240	0.1181483	96.1115820
scenepeople	-0.2628480	0.0979448	-2.6836339
mood	-0.0070341	0.0136624	-0.5148476
scenepeople:mood	0.0100545	0.0046618	2.1568100

- Now we have significant effects!
- Remember that scene was coded as 0 = no people, 1 = people
  - Looks like, on average, subjects gave the scene a rating that was -.26 lower when it contained people than when it didn't.
- The interaction is also significant. When the scene contained no people, there was a very weak, non-significant negative effect of mood.
  - When the scene did contain people, there was a significant change in the effect of mood (with each point on the mood scale increasing the picture rating by -.007 + .0101 = .0031). Not a huge effect, but significant.

Writing it up

- See the exercise!

Summary: Linear mixed models

- We can do repeated-measures analyses as well as more complex hierarchical models (e.g. students within classrooms within schools) using linear mixed models (LMMs)
- Such models combine fixed effects (where we test all possible factor levels or predictor values, e.g. condition) and random effects (where we can only test a subset of a population of factor levels, e.g. participant, or school, or county, or sentence)
- Usually, you start with a fixed-effects model and then add random intercepts and slopes.

  - Random effects are always centered around the corresponding fixed effects. For example, we might have a fixed intercept of 50, with a random intercept for Subject 1 of 50-10 = 40 and a random intercept for Subject 2 of 50+10 = 60.

- Model fitting is iterative and can be tricky. Overspecified models often end up failing to converge.
- Which random effects to include? Figuring out which random effects actually explain variance can take awhile.

Thank you!

- I know this was (and still is) a massive effort
- Thank you for staying motivated and engaging with the material.
- As always, come see me if you have questions!