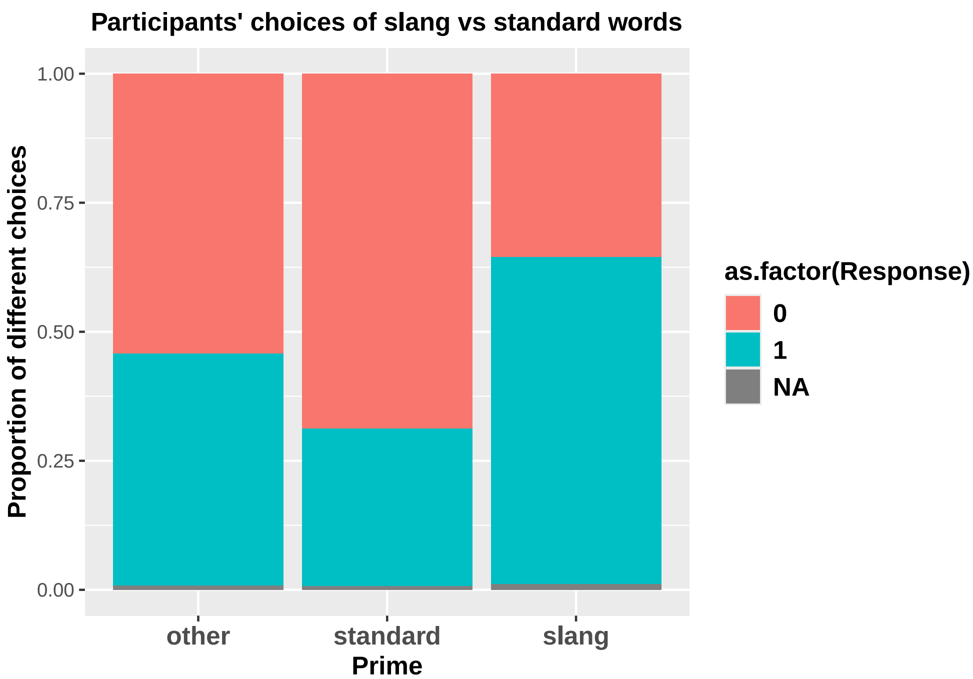
Regression and Mixed-effects Modelling in R : Session 3

Fang Jackson-Yang

2024-05-12

1 Data Wrangling and Visualisation

Language partners' usage of slang (e.g., 'noob') is predicted by whether their interlocutor has just used another slang word (e.g., 'M8') when its alternative standard form is available (e.g., 'M8' vs. 'mate' vs 'friend').



2 Fit Glmer Models

2.1 Maximum Model

```
m_max_dataISD = glmer(Response ~ Prime +
                        (1 + Prime|Participant) +
                        (1 + Prime|Item),
                      data = data_ISD,
                      family='binomial',
```

```
                      na.action=na.exclude)
summary(m_max_dataISD)
```

2.1.1 Think point: Any issue with this model? How do you solve it?

We got a warning: boundary (singular) fit: see ?isSingular.

The warning indicates that the model is 'overfitted' - that is, the random effects structure which we have specified is too complex to be supported by the data.

Perhaps the most intuitive advice would be remove the most complex part of the random effects structure (i.e. random slopes). This leads to a simpler model that is not over-fitted.

Additionally, when variance estimates are low for some specific random effect terms, this indicates that the model is not estimating this parameter to differ much between the levels of your grouping variable. In some experimental designs, it might be perfectly acceptable to remove this.

2.2 Reduced model 1 - drop by-item random slop

```
m2_dataISD = glmer(Response ~ Prime +
                               (1 + Prime|Participant) +
                               (1 | Item), data = data_ISD,
                             family='binomial',
                             na.action=na.exclude)
summary(m2_dataISD)
```

2.2.1 think point: Any issue with this model? How do you solve it?

This model gave convergence warnings.

One way to deal with convergence issue is to adjust / stopping the (convergence) tolerance for the nonlinear optimizer, using the optCtrl argument for [g]lmerControl: - optimizer="bobyqa" for Changing the optimization method - optCtrl=list(maxfun=2e5): for increasing the number of optimization steps.

Next we refit the model adding specifications for convergence tolerance.

2.3 Reduced model 2

```
m2b_dataISD = glmer(Response ~ Prime +
                      (1 + Prime | Participant) +
                      (1 | Item),
                    data = data_ISD,
                    family='binomial',
                    na.action=na.exclude,
                    control=glmerControl(optimizer="bobyqa",
                                         optCtrl=list(maxfun=2e5)))

summary(m2b_dataISD)
```

2.3.1 Think point: Any issue with this model? How do you solve it?

2.4 Simplest Model

```
m3_dataISD = glmer(Response ~ Prime +
                      (1|Participant) +
                      (1|Item),
                    data = data_ISD,
                    family='binomial',
                    na.action=na.exclude,
                    control=glmerControl(optimizer="bobyqa",
                                         optCtrl=list(maxfun=2e5)))

summary(m3_dataISD)
```

2.4.1 Think point:

How do we know our predictor "prime" helps explain the data?

Hint: we can build a null model and then do model comparison.

3 Model Comparision

3.1 Fit a null model

```
m_null_dataISD = glmer(Response ~ 1 +
                      (1|Participant) +
                      (1|Item),
                    data = data_ISD,
                    family='binomial',
                    na.action=na.exclude,
                    control=glmerControl(optimizer="bobyqa",
                                         optCtrl=list(maxfun=2e5)))

summary(m_null_dataISD)
```

3.2 Compare the null model with the one-predictor model

3.3 Think point: which model is better? why?

4 Interpret model results: Log odds to probability

Regression Table of the logistic Model

	Response			
Predictors	*Log-Odds*	*CI*	*Statistic*	*p*
(Intercept)	-0.28	-0.58 – 0.02	-1.83	0.067
Prime [standard]	-0.94	-1.11 – -0.77	-10.92	**<0.001**
Prime [slang]	1.13	0.96 – 1.30	13.15	**<0.001**
Random Effects				
σ^2	3.29			
$\tau_{00\ \text{Participant}}$	2.43			
$\tau_{00\ \text{Item}}$	0.13			
ICC	0.44			
$N_{\text{Participant}}$	160			
N_{Item}	30			
Observations	5386			
Marginal R^2 / Conditional R^2	0.109 / 0.499			

Note that p-values are back when we've fitted a model with glmer(), without installing the lmerTest package. R followed the standard practice to give statistical significant, based on asymptotic Wald tests on the difference in log-likelihood.

4.1 Understand the Coeffecients

What is the intercept? What is the slope?

Hint: In logit models, the model coefficients are in logit units log(log-odds).

To make the estimates/coefficients meaningful, we need to transfer the log-odds to odds.

4.2 Calculate Probabilities

We can convert the coefficients to odds by using $exp()$, and then obtain the probability: $p = \frac{exp(x)}{1+exp(x)}$.

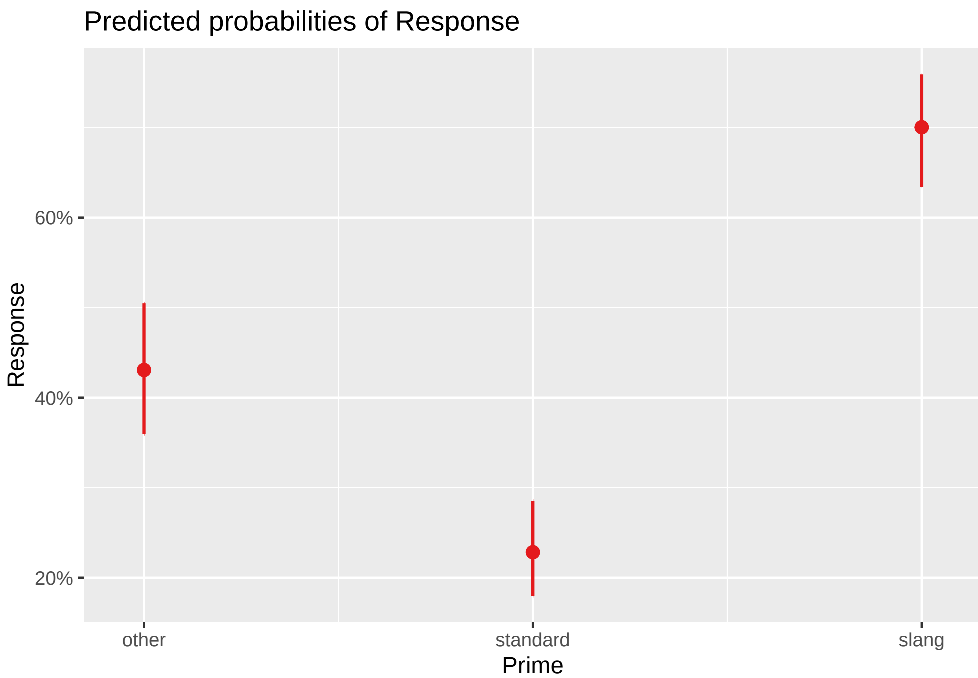
(1) The probability of using slang under the baseline condition "prime=other" is: $\frac{exp(-0.28)}{1+exp(-0.28)} = 0.43$

(2) Participants' tendency to use slang under the condition "prime = standard" is reduced by a factor of -.94 (log odds) compared to the baseline condition. The probability of using slang under the "prime = standard" condition is: $\frac{exp(-0.28+(-0.94))}{1+exp(-0.28+(-0.94))} = 0.23$

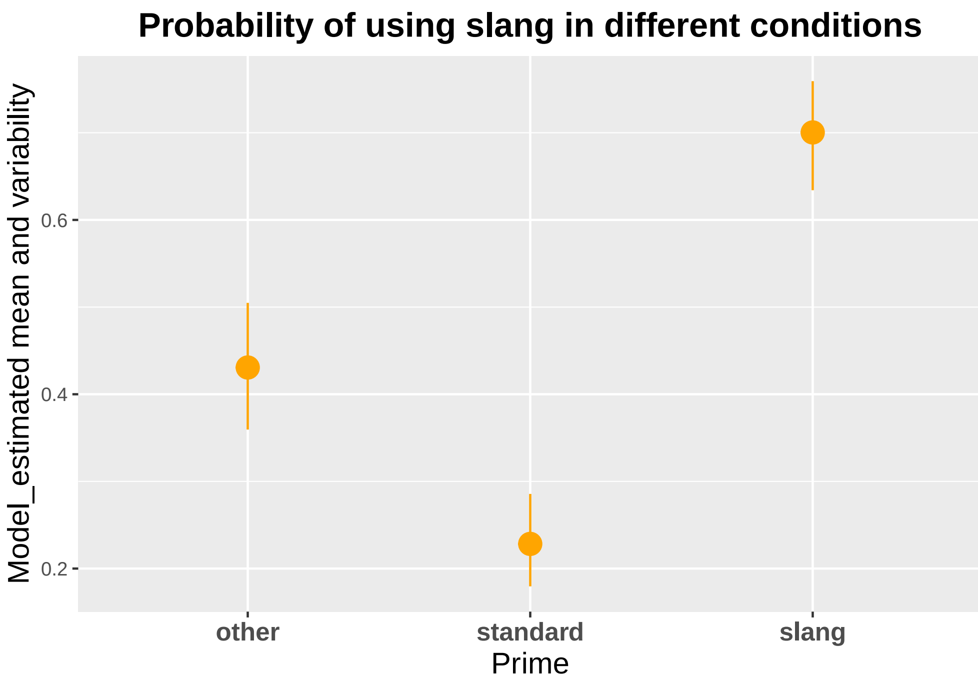
(3) Participants' tendency to use slang under the condition "prime = slang" is increased by a factor of 1.13 (log odds) compared to the baseline condition. The probability of using slang under the "prime = slang" condition is: $\frac{exp(-0.28+1.13)}{1+exp(-0.28+1.13)} = 0.70$

5 Visulise Model Results

5.1 Predicted Probabilities

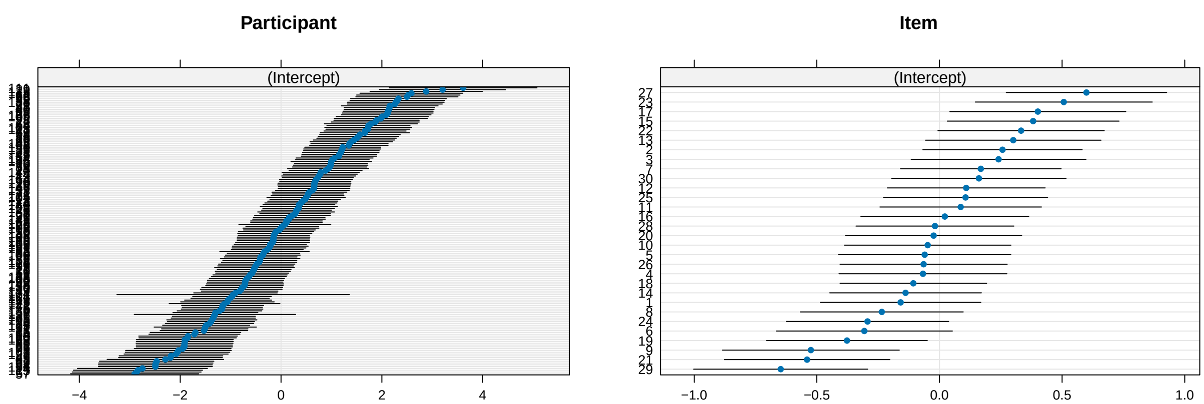


If you have problem installing the sjPlot package, you can do it manually by using the "effects" library. Plot the model-estimated condition means and variability.



5.2 Random Variance

To check the random effect, we extract the deviations for each group from the fixed effect estimates using the ranef() function.

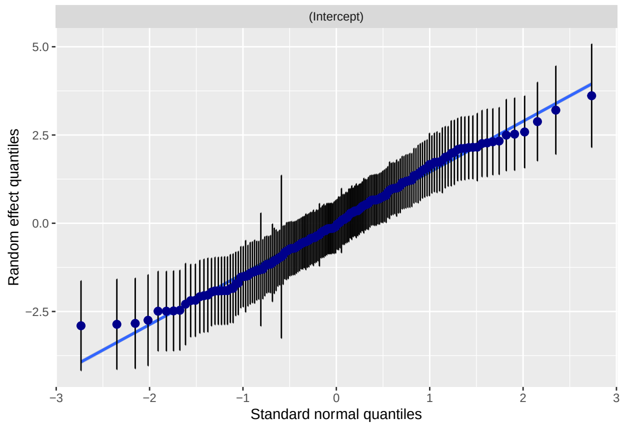


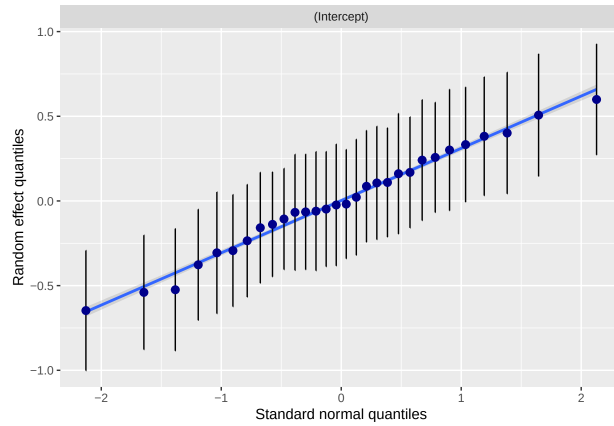
6 Model Assumptions Check

We can use the plot_model() function to visualise whether model assumptions are met.

For linear (mixed) models, this function produces plots for: (1) multicollinearity (i.e., check Variance Inflation Factors); (2) QQ-plots: checks for normal distribution of residuals and homoscedasticity, i.e., constant variance of residuals).

For generalized linear mixed models, this function produces plots for the QQ-plot for random effects.



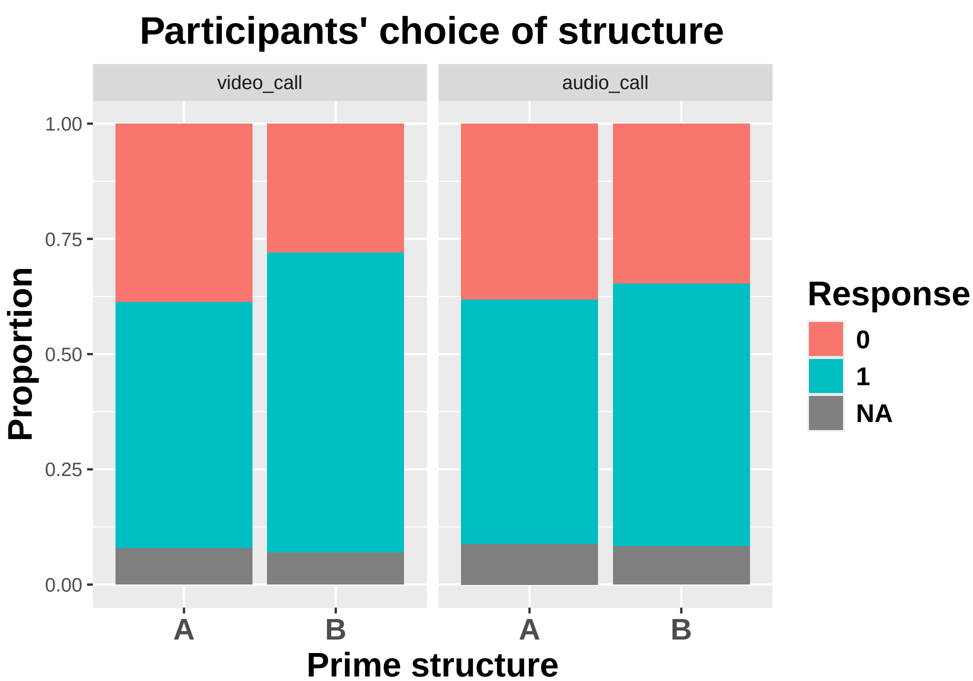


7 Exercise

It's your turn!

Try to explore syntactic priming effect in the cheese data.

7.1 Data loading, cleaning and visualisation



7.2 Fit an additive model

Fit an intercept-only model including the following as fixed effects: (1) main effect of prime and (2) main effect of communication.

7.3 Fit an interactive model

Fit an intercept-only model including the following as fixed effects: (1) main effect of prime, (2) main effect of communication, and (3) the interaction effect between prime and communication.

7.4 Model comparison

7.5 Interpret the results

Hint: remember the coefficients are log odds. you need to transfer them to odds and then probability

7.5.1 Convert log odds to odds

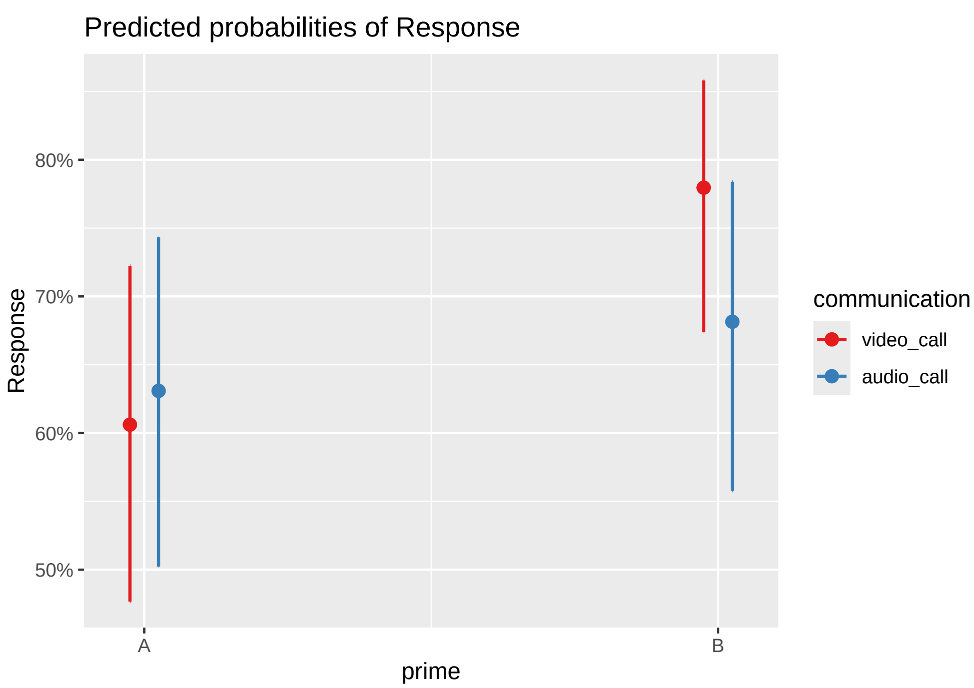
7.5.2 (1) Probability of producing B in video+primeA condition (intercept)

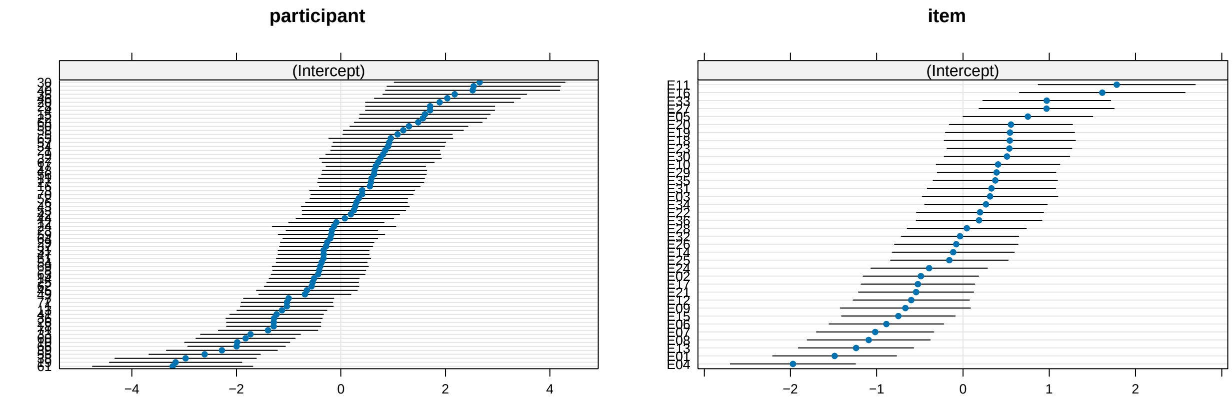
7.5.3 (2) Probability of producing B in video+primeB condition(intercept + slope)

7.5.4 (3) Probability of producing B in audio+primeA condition (intercept + slope)

7.5.5 (4)Probability of producing B in audio+primeB condition (intercept + slope)

7.6 Visulise the Model





7.7 Check and plot random effects

7.8 Check Model Assumptions

For generalized linear mixed models, returns the QQ-plot for random effects.