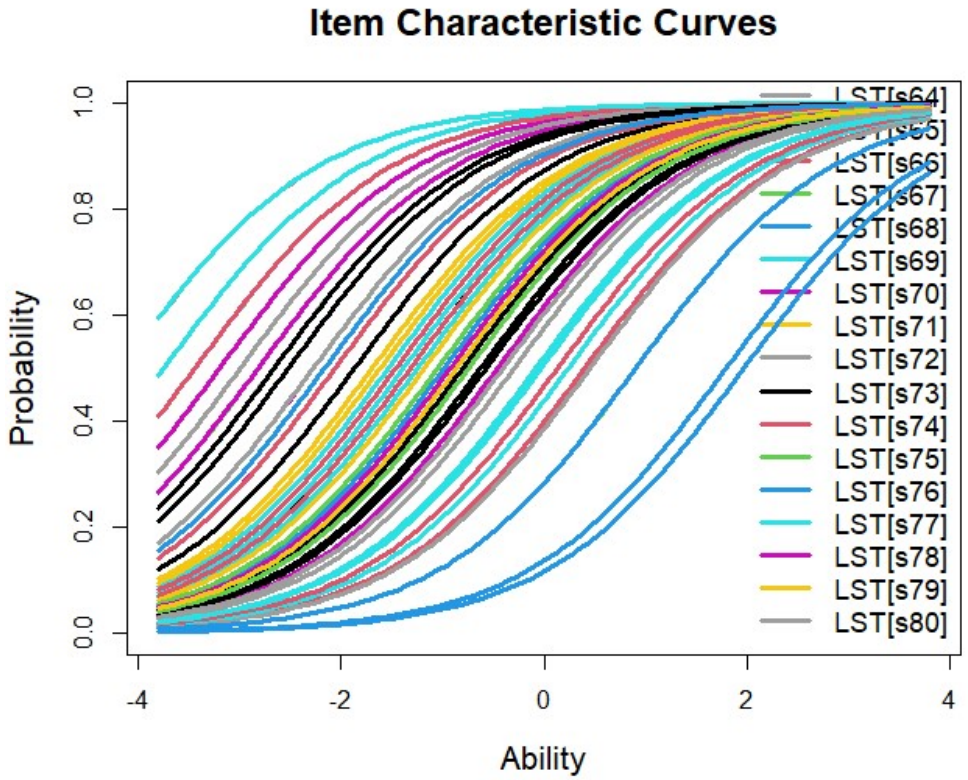
Simulations lexico-semantic experiment Menut et al.

First we analyze the data of the experiment with a Rasch model 2PL (difficulty of items and discrimination are allowed to vary; is the standard Rasch analysis). Datafile is LST_LMM.xlsx.

```
library(readxl)
library(ltm)
LST_LMM <- read_excel("LST_LMM.xlsx")
mydata <- LST_LMM[ -c(1:3) ]
fit1 <- rasch(mydata)
summary(fit1)
plot(fit1, legend = TRUE, cx = "bottomright", lwd = 3, cex.main = 1.5, cex.lab = 1.3, cex = 1.1)
```

This is a plot of how the analysis looks like (ordinate = ability of the participant, probability = chances of getting the item right; different lines = different stimuli – easy and more difficult ones)



Next we generate new stimuli according to a similar Rasch model. We assume that the 90 participants are equally spread between z-values -2 and +2. We also assume that items are equally spread between -2 and +1 (asymmetric, so that items are not too easy).

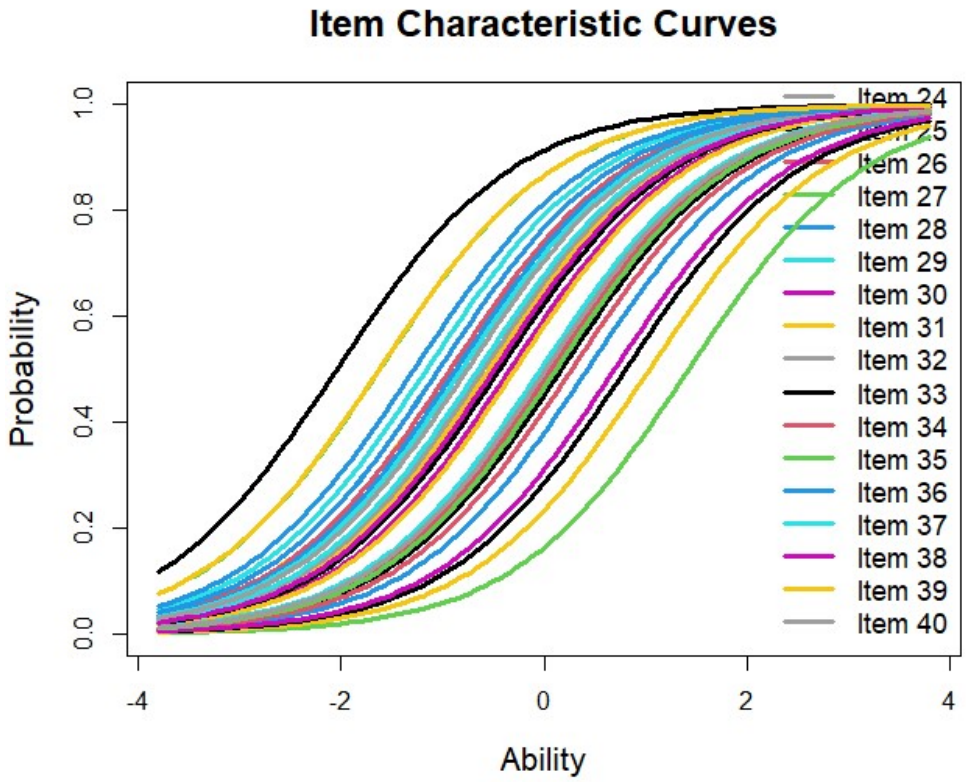
We generate the first 40 stimuli for the non-shared condition.

```
ppar <- runif(90,min = -2, max = 2)
spar <- runif(40,min =-2.0, max = 1)
mydata2 <- sim.2pl(ppar, spar, discrim = 0.5,cutpoint="randomized")
```

Next we generate the second 40 stimuli according to the same equation for the shared condition (i.e., d = .0)

```
d = 0.0
spar <- runif(40,min =-(2.0-d), max = (1-d))
mydata3 <- sim.2pl(ppar, spar, discrim = 0.5,cutpoint="randomized")
```

This is how the Rasch looks like for some stimuli:



We also have to define the proficiency of the participants. To some extent, this is given by the variable ppar (ability of the participants). However, if we use this variable in our MME analysis, there is no variance in participant intercepts left. It is also unlikely that we have a proficiency measure that correlates 1.00 with true language proficiency. A correlation of r = .7 seems reasonable. So, we use a proficiency measure that correlates more or less r = .7 with the ability used to generate Rasch. This is done by adding noise to the ppar variable.

```
prof = ppar  + rnorm(90,0,1.7)
```

Now we have everything to generate a data file to be analyzed with MME.

For this we can use the R lines:

```
fit3 <- glmer(Value ~ Proficiency*Condition + (Condition|Part) +( 1 | Stim), data=output,
family=binomial)
summary(fit3)
```

Which gives the following output.

```
Generalized linear mixed model fit by maximum likelihood (Laplace Approximation) ['glmerMod']
 Family: binomial  ( logit )
Formula: Value ~ Proficiency * Condition + (Condition | Part) + (1 | Stim)
   Data: output

     AIC      BIC   logLik deviance df.resid
  8241.1   8296.1  -4112.5   8225.1     7192

Scaled residuals:
    Min      1Q  Median      3Q     Max
-5.0377 -0.7293  0.3480  0.6732  3.4451

Random effects:
 Groups Name            Variance  Std.Dev.  Corr
 Part   (Intercept)     0.76053   0.8721
        Conditionshared 0.02546   0.1596    -0.38
 Stim   (Intercept)     0.56181   0.7495
Number of obs: 7200, groups:  Part, 90; Stim, 80

Fixed effects:
                            Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)                 0.35822   0.15587     2.298    0.0215 *
Proficiency                 0.34494   0.05501     6.271    3.59e-10 ***
Conditionshared             0.01028   0.17752     0.058    0.9538
Proficiency:Conditionshared -0.02139  0.03249     -0.658   0.5102
---
```

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Correlation of Fixed Effects:
```
            (Intr) Prfcnc Cndtns
Proficiency -0.090
Conditnshrd -0.586  0.010
Prfcncy:Cnd  0.019 -0.373 -0.021
```

For comparison, this is the jamovi output:

Model Info

Info	Value	Comment
Model Type	Logistic	Model for binary y
Call	glm	Value ~ 1 + Proficiency + Condition + Condition:Proficiency + (1 | Stim) + (1 + Condition | Part)
Link function	Logit	Log of the odd of y=1 over y=0
Direction	P(y=1)/P(y=0)	P(Value =) / P(Value =)
Distribution	Binomial	Dichotomous event distribution of y
LogLikel.	-4112.5436	Less is better
R-squared	0.0757	Marginal
R-squared	0.3349	Conditional
AIC	8241.0900	Less is better
BIC	8296.1418	Less is better
Deviance	7659.0813	Conditional
Residual DF	7192.0000	
Converged	yes	
Optimizer	bobyqa	

Fixed Effect Omnibus tests

	X^2	df	p
Proficiency	42.598	1.00	< .001
Condition	5.31e-4	1.00	0.982
Condition ✻ Proficiency	0.433	1.00	0.510

Random Components

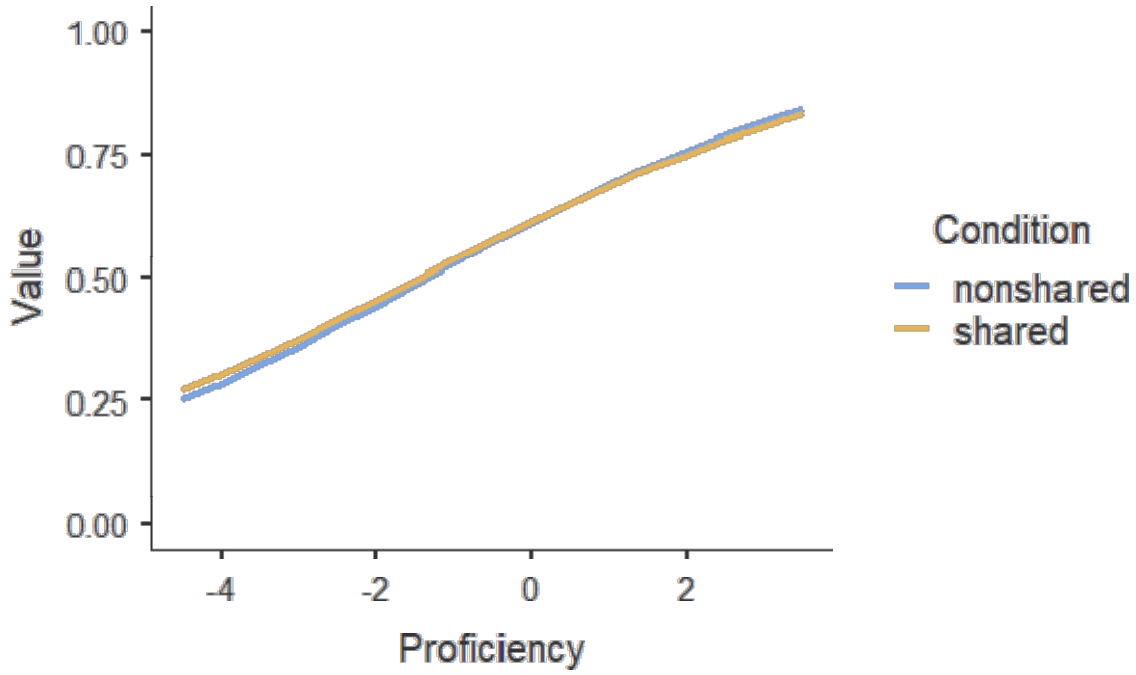
Groups	Name	SD	Variance
Part	(Intercept)	0.845	0.7139
	Condition1	0.160	0.0255
Stim	(Intercept)	0.750	0.5618
Residuals		1.000	1.0000

Note. Number of Obs: 7200 , groups: Part 90, Stim 80

Random Parameters correlations

Groups	Param.1	Param.2	Corr.
Part	(Intercept)	Condition1	-0.298

Effects Plots



Compare the jamovi output to the output of the experiment itself.

Model Info

Info	Value	Comment
Model Type	Logistic	Model for binary y
Call	glm	response ~ 1 + suffix + sPLexT + suffix:sPLexT + (1 + suffix | id) + (1 | items)
Link function	Logit	Log of the odd of y=1 over y=0
Direction	P(y=1)/P(y=0)	P(response =) / P(response =)
Distribution	Binomial	Dichotomous event distribution of y
LogLikel.	-2947.4284	Less is better
R-squared	0.0923	Marginal
R-squared	0.4200	Conditional
AIC	5910.8600	Less is better
BIC	5965.6218	Less is better
Deviance	5401.9640	Conditional
Residual DF	6936.0000	
Converged	yes	
Optimizer	bobyqa	

Fixed Effect Omnibus tests

	X^2	df	p
suffix	0.124	1.00	0.725
sPLexT	116.295	1.00	< .001
suffix ✻ sPLexT	3.887	1.00	0.049

Random Components

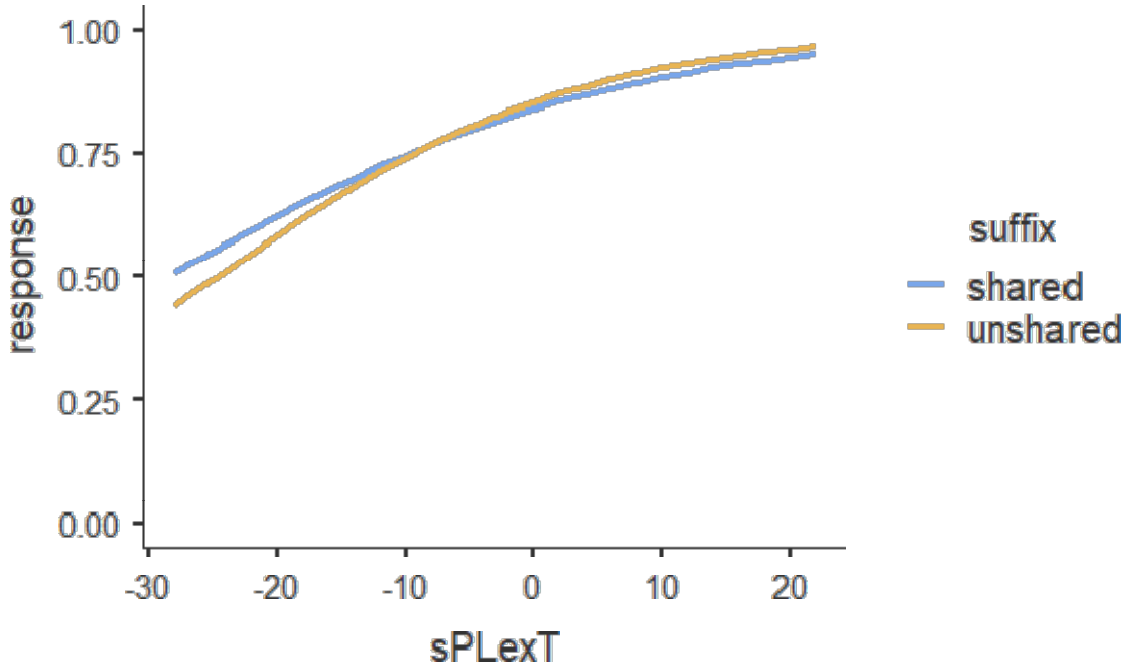
Groups	Name	SD	Variance
id	(Intercept)	0.543	0.2947
	suffix1	0.221	0.0489
items	(Intercept)	1.245	1.5493
Residuals		1.000	1.0000

Note. Number of Obs: 6944 , groups: id 92, items 80

Random Parameters correlations

Groups	Param.1	Param.2	Corr.
id	(Intercept)	suffix1	0.612

Effects Plots

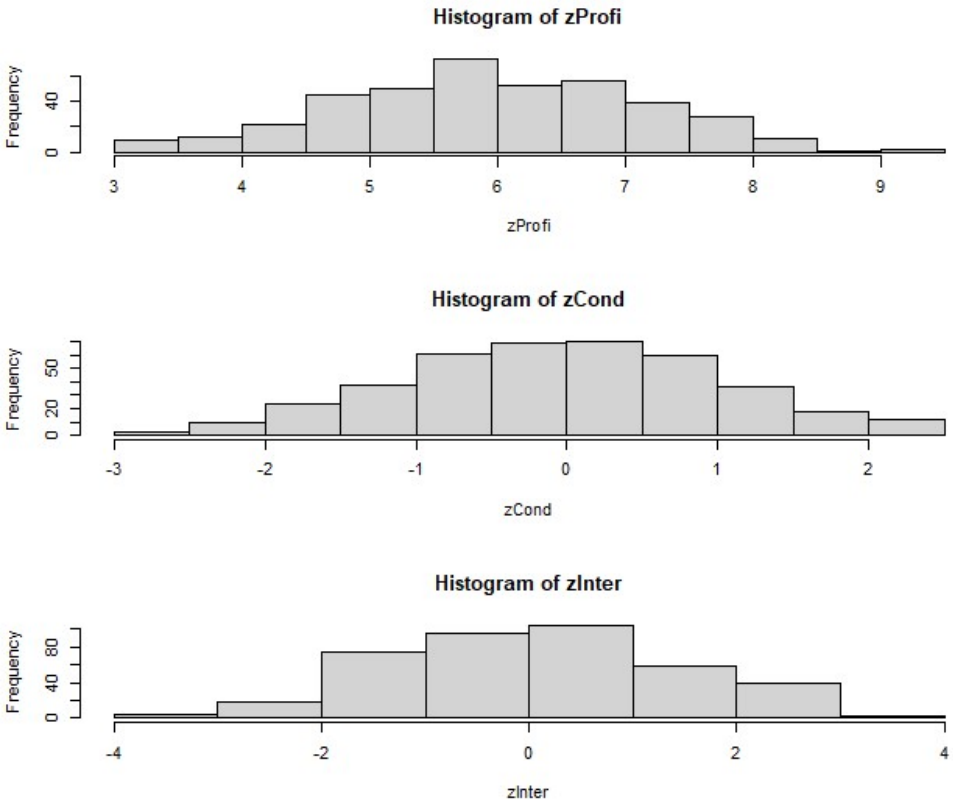


Now that we have the basic model, we can run simulations. In the program below, we have 400 datasets simulated and analyzed.

#Simulate and look at the distribution of z-values

```
nSim = 400
zProfi <- numeric(nSim)
zCond <- numeric(nSim)
zInter <- numeric(nSim)

# create progress bar in case it takes a while
pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)

for(i in 1:nSim){ #for each simulated experiment
  setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  ppar <- runif(90,min = -2, max = 2)
  prof = ppar  + rnorm(90,0,1.7)
  proficiency <- rep(prof, times=40)
  spar <- runif(40,min =-2.0, max = 1)
  mydata2 <- sim.2pl(ppar, spar, discrim = 0.5,cutpoint="randomized")
  output1 <- melt(mydata2, id.vars=rownames(mydata2))
  condition <- rep("nonshared", times = 40*90)
  output1 <- cbind(output1,proficiency,condition)

  d=0.0
  spar <- runif(40,min = (-2.0-d), max = (1-d))
  mydata3 <- sim.2pl(ppar, spar, discrim = 0.5,cutpoint="randomized")
  output2 <- melt(mydata3, id.vars=rownames(mydata3))
  condition <- rep("shared", times = 40*90)
  output2 <- cbind(output2,proficiency,condition)
  output2[,2] <- output2[,2] + 40

  output <- rbind(output1,output2)
  colnames(output) <- c("Part","Stim", "Value", "Proficiency", "Condition")
  fit3 <- glmer(Value ~ Proficiency*Condition + (Condition|Part) +( 1 | Stim), data=output, family=binomial)
  zProfi[i] <- coef(summary(fit3))[2,"z value"]
  zCond[i] <- coef(summary(fit3))[3,"z value"]
  zInter[i] <- coef(summary(fit3))[4,"z value"]
}
close(pb)#close progress bar

par(mfrow=c(3,1))
hist(zProfi)
hist(zCond)
hist(zInter)
```

This gives the following outcond for proficiency, suffix condition, and the interaction (z-values). We see that the effect of proficiency is significant in each simulation ($z > 1.96$). There is no effect of condition beyond the alpha level (power = .05). However, there are too many interactions that are significant (power = .17). Further simulation suggests that this is due to the wide distribution of participants. When the distribution becomes narrower (z-values participants form a standard normal distribution) then you get better values. Bottom line is that one should be careful to interpret significant interactions! Are likely to be a fluke.



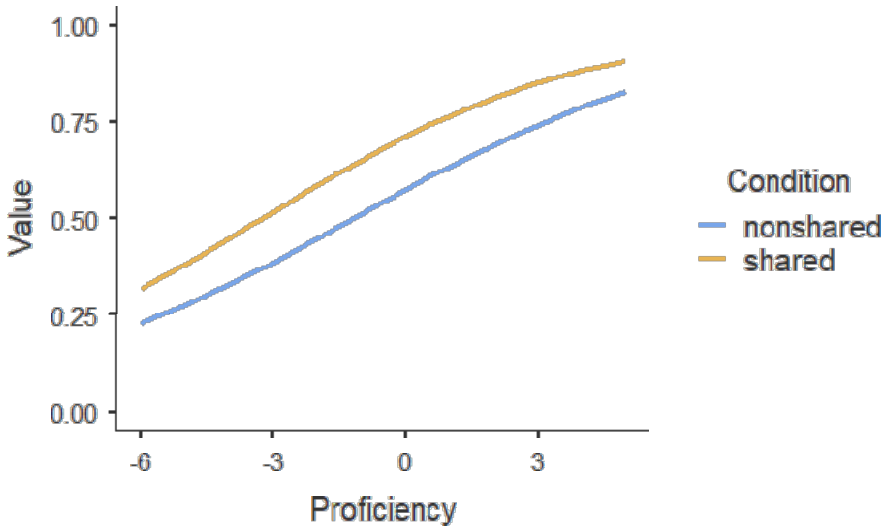
We can now shift the item values in the shared condition by d = .4 or d = .6. Below are the powers we get. Bottom line is that we have 44% chance of finding a shift of d = .4 and 78% to find a shift of d = .6.

Power (p < .05)	**d = .0**	**d = .4**	**d = .6**
Proficiency	1.00	1.00	1.00
Shared	0.05	0.44	0.78
P * S	0.17	0.13	0.16

Below you see how the effects look like if the data of a d = .6 experiment are analyzed.

d = .6

Effects Plots



The programs used are available at https://osf.io/cv8ny/files/ in the folder **Simulations power**.