Supplementary information

1. Measuring iconicity

Constructing the initial languages
The set of 12 signals for the initial languages was constructed as follows. The set included 18 characters, all with approximately the same frequency. In other iterated language learning experiments, participants tended to remove or reshuffle the characters present in the initial languages, but did not add new characters. If a variety of characters are present in the initial language, mappings between “spiky” and “rounded” characters and spiky and rounded objects is allowed, though not forced. The set of 12 signals in initial languages included the five vowels *a, i, o, u* with frequency 7, and *e* with frequency 8; and the 13 consonants with monophonemic, unambiguous pronunciation in Spanish (the participants were all Spanish speakers): *b, d, l, k, l, m, p, r, s, t* with frequency 3 and *f, n, z,* with frequency 2. In order to construct signals that would not be rated as particularly spiky or round, we generated 30 sets of 12 signals with the character tokens, and then selected one set as follows. We first randomly created consonant-vowel syllables with the characters, then joined the syllables again randomly to form four 2-syllable words, four 3-syllable words and four 4-syllable words. We selected six such sets of 12 words that did not include any recognizeable words, acronyms or close neighbours. The words from these six sets were rated for spikiness-roundedness: 24 participants (75% female, mean age = 21.38, range 19 - 39, SD = 3.99) were asked to rate each of the words in a 10-point Likert scale, with the spiky and the rounded shapes positioned on either end of the scale (reversed in half of the cases) (Fig. 1). Finally, we selected the 12-word set that had the lowest variance in spiky-roundedness scores. The selected word set comprised (in order of increasing spikiness) the words *neroro, romiba, faludelu, sitobuna, doje, fukopo, tezadu, pilu, timajeba, zisa, kemusije, kipe.*

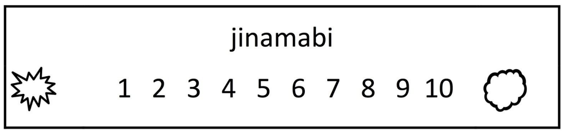


Figure 1. A snapshot of the norming task.

Transmitted languages

Sixteen native Spanish-speaking judges (24% males; mean age = 22.9, range 18-30, SD=4.71), different from the ones used to norm the words for the input languages, were recruited to rate, in random order, (randomly ordered) all the word types in the initial languages, and in the languages that were transmitted to the next generation (one language for each of generations 1 to 6), as well as our extra words --bouba, kiki, maluma and takete-- to check that their ratings corresponded to those expected by previous experiments --Kohler, (1929), Ramachandran and Hubbard (2001)--, and finally all the isolated letters in the alphabet. The ratings were collected through an online form that presented each word above a 7-point Likert scale, which also had and a spiky and round shape (the same shapes as in the game, but with a thin black border and white fill) on either side, in a counterbalanced design. See table 1 for average ratings.

Estimating spikiness ratings for all produced words

The data includes spikiness ratings for words and individual letters from several participants, but only for the transmitted words. For ratings of all produced words, we needed to estimate the spikiness ratings. We used a mixed effects model to estimate the mean spikiness rating of each transmitted word (there were no significant effects of participant sex, age or the direction that the Likert scale was presented). We then tested 2 models which tried to predict these ratings. The first used the mean ratings of each letter in a word. This model's predictions correlated with the true ratings with r = 0.73. The second model used a random forests model (using the R package *party*) with unigram and bigram letters as predictors. This model produced predictions which correlated with the true ratings with r = 0.87 (trained on 75%, tested on the held-out 25%). A random forests model was then trained on all transmitted words, and used to predict the spikiness ratings of all produced words in the experiment. See **Section 5** for the R code.

Table 1. Average rating for each signal presented to the judges.

string	rating	string	rating	string	rating
a	1.929	jutisu	4.929	remuuba	2.333
b	1.467	jutumu	4.75	remuva	3.533
c	3.571	kamicazu	6.133	remuvaa	3.375
d	1.933	kamicazuneroro	6.067	remuve	3.067
e	3.933	kamicazuu	6.2	remuvec	4.867
f	3.733	kamidefa	5.5	remuvee	3.267
g	3.467	kamijace	6.2	riba	3.688
h	3.25	kamijazu	6	rimba	2.5
i	5.733	kamikazu	6.333	rmba	3.467
j	5.933	kemusije	6.143	rombebe	2.667
k	6.667	keroro	4.8	rombeber	2.867
l	4.188	kiasu	6.133	rombebo	2.667
m	2	kimaceba	5.067	rombembe	2.267
n	2.5	kimapeba	5.5	rombembo	2
o	1.25	kimaziba	5.933	rombila	3.067
p	3.6	kipe	5.688	romembe	2.533
q	6	kiper	6.133	romiaz	3.867
r	4.214	kite	6.133	romiba	3.2
s	2.571	komibu	5.625	romibaa	2.8
t	4.733	kopo	4.313	romibe	2.875
u	3.067	kusijo	6.267	romibu	2.813
v	4.733	lea	2.333	romiva	4.067
w	5.25	nabuleddu	2.467	romive	3.813
z	5.938	nabuledu	2.4	romkbebe	4.688
kiki	5.563	nagudelu	2.813	romkbeber	5.467
buba	1.688	nagudelur	3.6	romkbebo	5.133
maluma	1.733	nagudelurr	3.667	ronbebe	2.4
takete	5.563	naguleddu	2.733	ronbebo	2.667
dejuzme	5.067	naguledurr	3.733	rremuva	3.467
dejuzmee	4.813	neroro	3.333	samurama	2.8
doje	5.063	nerorooo	2.733	samurana	3.333
dojo	4.6	neru	3.857	sitobuna	3.333
doze	3.688	neruru	2.8	siturama	3.8
duje	5.133	neruseje	4.375	sonorama	2.467
faculito	3.188	neruzeje	5.2	tajeba	4.875
facultito	3.813	neruzejee	5.8	tajibe	5.214
fadulele	3.188	neuror	3.333	tajuba	4.267
fagudelu	3.6	neuroro	3.125	tamiziba	4
falabea	2.625	nezuve	4.313	tezadu	4.875
faladea	2.733	paladea	2.667	tezuja	5.533
falepa	3.667	paludelu	3.4	tibaceba	2.8
falodea	3.333	pilo	3.533	timajeba	4.438
falodelu	3	piloluuuu	3.267	timaziba	4.563
falucit	3.933	pilu	3.938	tukebona	5.4
falucito	3	pilubl	2.813	tuquebona	3.8
falucitu	3.857	piluuu	3.214	vremuba	3.938
faludefa	3.733	pitu	3.667	vremuvaa	3.267
faludefu	3.267	pitubl	4	zimba	3.667
faludela	3.188	piu	4.667	zisa	5.4
faludelu	2.875	quemusije	5.133	zisu	5
farulele	3.267	quesimuje	5.4	zueje	5.688
fukopo	5.733	ralucito	3.938	zureje	5.733
juke	6.375	remuba	2.467		
jukee	5.867	remuru	3.333		

2. Systematicity, error, expressivity and task success

Systematicity

Systematicity in a language was measured using the Mantel test (Mantel, 1967, following e.g. Tamariz (2008), Kirby et al. (2008, 2015). We first calculated the correlation between the word-form pairwise differences (normalized Levenshtein distance between signals -- character strings) and object pairwise differences (Hamming distance between properties of objects -- shape, colour and border) for all possible pairs of items in the language. Then we ran a Monte Carlo randomization (N=10,000) by shuffling all the word-object mappings and obtained a z-score indicating the significance of the correlation.

The initial languages had non-significant systematicity levels (Mantel test: -0.70 < z-score < 0.70; 2-tailed p > 0.24).

To analyse the data from the transmission chains, a linear mixed effect model analysis of the effects on the systematicity of languages with condition (Communication or Reproduction) and Generation (0-6) as fixed effects and Chain (0-7) as random effect returned a significant effect of Generation (log likelihood difference = 28.15, df = 1, χ^2= 56.22, p = 10^{-13}), but no significant effect of Condition or Generation x Condition interaction. Systematicity increased cumulatively over generations in both conditions. This result was expected as transmission, the driver of systematicity (Kirby et al. 2008), was present in both the communication and the reproduction conditions. See section 5 for model code and full results.

Transmission error

This measure of how difficult to reproduce a language is was calculated as the average normalized Levenshtein distance between the word for each object at the generation of interest and the previous generation (Kirby et al., 2008, 2015).

Transmission error values were submitted to a linear mixed effect model analysis with Condition (communication or reproduction) and Generation (1-6) as fixed effects and Chain (0-7) as random effect. We found only a significant effect of Generation (log likelihood difference = 12.37, df = 1, χ^2= 26.3, p = $3x10^{-7}$), but no significant effect of Condition or Generation x Condition interaction. The transmission error decreased cumulatively in both conditions, indicating the languages became more compressible and easier to learn and reproduce over the generations. This was expected, as transmission, the driver of compressibility (Kirby et al. 2008) was present in both conditions. See section 5 for model code and full results.

Expressivity

Expressivity is measured as the number of word types per language, which indicates how many distinct meanings the language can express.

We found a significant effect of Generation on the number of word types per language (log likelihood difference = 10.47 , df = 1, χ^2= 20.93, p = 10^{-5}) and a significant interaction between Generation and Condition (log likelihood difference = 2.72, df = 1, χ^2 = 5.65, p =

0.017), indicating that participants in the reproduction condition cumulatively introduced more homonyms than in the communication condition.   See section 5 for model code and full results.

Task success

Despite the difference in expressivity, the scores obtained by participants in both conditions was similar across conditions. The mean proportion of correct item guesses was 47.2% and there were no significant differences between conditions (communication condition = 45.14%, reproduction condition = 51.39%; log likelihood difference = 0.84 , df = 1 , χ^2 = 1.67 , p = 0.2). This confirms the finding that expressivity is an adaptation to communication (Kirby et al. 2015), and shows that the feedback in our reproduction condition did not pose a pressure for expressivity as strong as the interlocutor feedback in the communication condition. See section 5 for model code and full results.

Unique words

In the communication condition, participants produced 188 unique words in total across all generations and all chains, while in the reproduction condition they produced 84 unique words.

3. Task instructions given to participants

3. 2. Instructions for the communication condition (translated from the Spanish original)

You will play a game together with a partner; you will need to use an artificial language to ask objects to each other. It is a cooperative game, so you do not play against each other, but score points together.
First, you will be trained on the artificial language. You will see each object on the screen together with its name. You will have a chance to see all objects-name pairs several times.
After training, you will play the game. You will see screens similar to those in Figs. 1 and 2. You are always **the character on the left** and your partner is the smaller character on the right. During the game you will take turns to give and request objects.

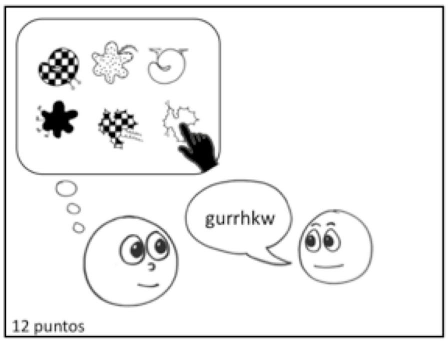


Figure 1. You (on the left) are selecting the object that your partner has requested from you.

When it is your turn to **give** (Fig. 1), your partner will request an object from you, and your task is to select, from the array of six objects, the one you think he is requesting. Select the object by clicking on it. Shortly after you make your selection, you will see the object your partner meant; if it is the same you have given him or her, both of you score a point.
When it is your turn to **request** (Fig. 2), you will see an object and you have to write its name in the artificial language to let your partner know which object you want. He or she will receive your request and give you an object. If it is the one you requested, you both score a point. Otherwise, the score stays the same.

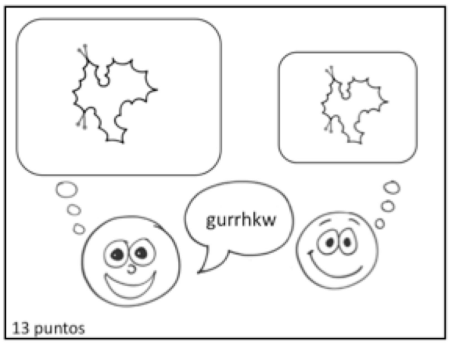


Figure 2. You (on the left) have requested an object from your partner and he has given you the correct object.

· *Please, do not use recognizeable words.*
· *Do not speak out loud during the game.*
· *Please keep the game flowing by not spending too much time thinking or deciding.*

3. 2. Instructions for the reproduction condition (translated from the Spanish original)

You are going to be trained on an artificial language and then you will be tested to check how well you have learned it. During training, you will see each object together with its name in the artificial language. you will have a chance to see each object and name several times. After training, you will be tested in two ways.

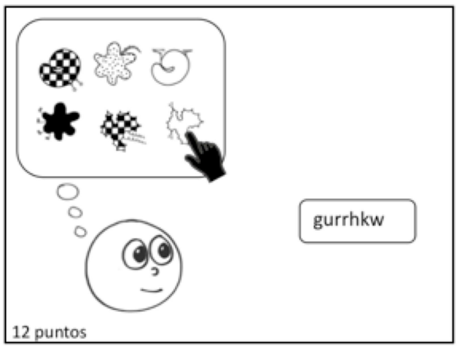


Figure 1. You are choosing the object that corresponds to the name in the box on the right..

In the **association** trials, you will see a word in the artificial language in the box on the right, and your task will be to select, from the six options, the object the word refers to. (In order to select, click on the object, as seen in Fig. 1). Shortly afterwards, you will see the correct object. If your selection is the same as the correct object, you score a point.

In the **production** trials, you will see an object and your task will be to write its name in the artificial language. Shortly afterwards, you will see the object that the name you typed name corresponds to in the artificial language. Again, if both objects are the same, you score a point (Fig. 2).

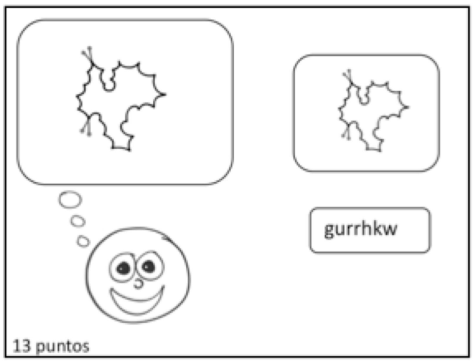


Figure 2. You have typed the correct name.

The score reflects your level of proficiency in the artificial language.
NOTES: *Do not take too long to think about your answers.*

4. Extra tests of iconicity: ANOVA, Binary Regression Tree analysis and Monte Carlo analysis

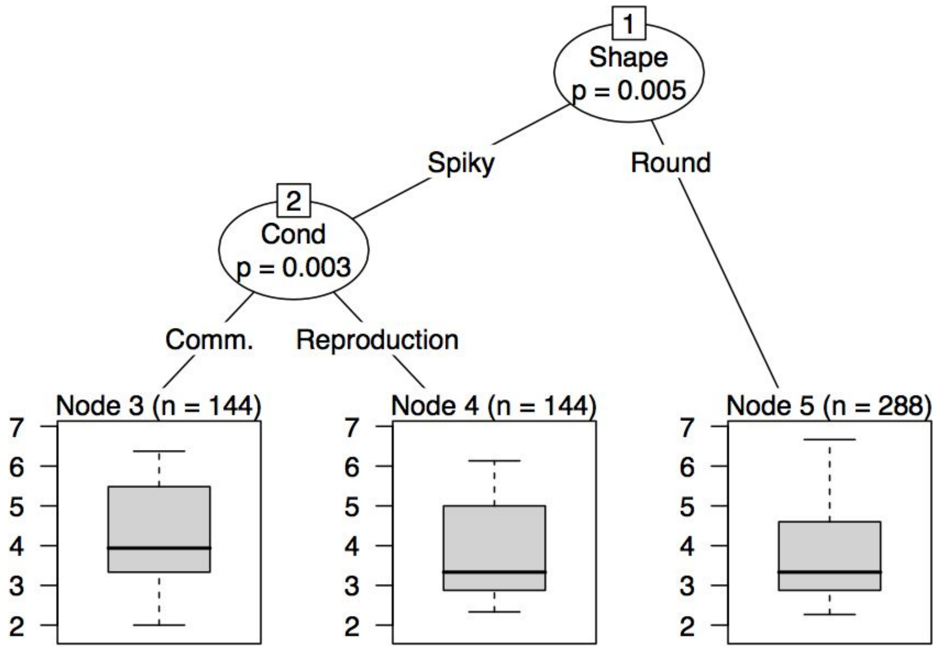
ANOVA.
Word spikiness values were submitted to a univariate ANOVA with Communication condition (Communication or Reproduction), Object Shape (Spiky or Round) and Generation (0 to 6) as factors. (A repeated-measures design was impracticable because we had, for each language, a large number of non-alignable word spikiness values.) We found main effects of Object Shape (F(1,644)=10.240, p=0.001) and Generation (F(6,644)=4.297, p<0.001), but not of Condition (F(1,644)=3.225, p=0.073), but we found a significant Object Shape x Condition interaction (*F*(1,644)=8.788, *p*=0.003). The effect of Generation was due to a general decrease in spikiness values over time. The effect of Object Shape resulted from spiky objects being denoted by words with higher spikiness values than round objects (spiky: *M*=4.16, *SD*=1.23; round: *M*=3.87, *SD*=1.11).

The interaction between Condition and Shape was investigated by submitting the spikiness values in each condition to a further ANOVA with Object shape as the dependent factor. The results showed a significant effect of Object shape in the Communication (*F*(1,334)=18.447; *p*<0.001), but not in the reproduction condition (*F*(1,334)= 0.27; *p*=0.869). This suggests that the emergence of iconicity in the languages was mediated by aspects of the interactive communicative task between two interlocutors.

Binary Regression Tree analysis

We use a binary decision tree to predict spikiness ratings by condition, generation, item shape, item colour and item border type. The tree was produced using the R package *party*[1] . The results agree with those above, namely that the main effects are for shape, but spiky meanings are rated as more spiky in the communication condition. See **Section 5** for the model code.

[1] Hothorn, T., Hornik, K., Strobl, C., and Zeileis, A. (2012). party: A laboratory for recursive partytioning (version 1.0-2.)



Monte Carlo analysis
Iconicity was further tested using a permutation test carried out for the language produced each chain at each generation. We calculated the difference in spikiness values between the words for spiky objects and the words for round objects in two ways: as 2-tailed t-tests and as the difference of the means. We obtained the z-score of the veridical difference against a sample of 10,000 randomizations of the pairings between words and their spikiness values.

Below is the R code for implementing this test.

The interactive origin of iconiciy: Permutation tests

Load libraries

```
library(gplots)
library(lattice)
library(lme4)
```

Load data

```
finalLangs = read.csv("../data/finalLanguages/FinalLanguages.csv", stringsAsFactors = F)
# make a variable which stores condition, chain and generation
finalLangs$cond2 = paste(finalLangs$Cond,finalLangs$Chain, finalLangs$Gen)
```

Run permutation test

For each output language, take the difference in means in spikiness ratings between spiky and non-spiky meanings. Compare this to 1000 permutations of the numbers.

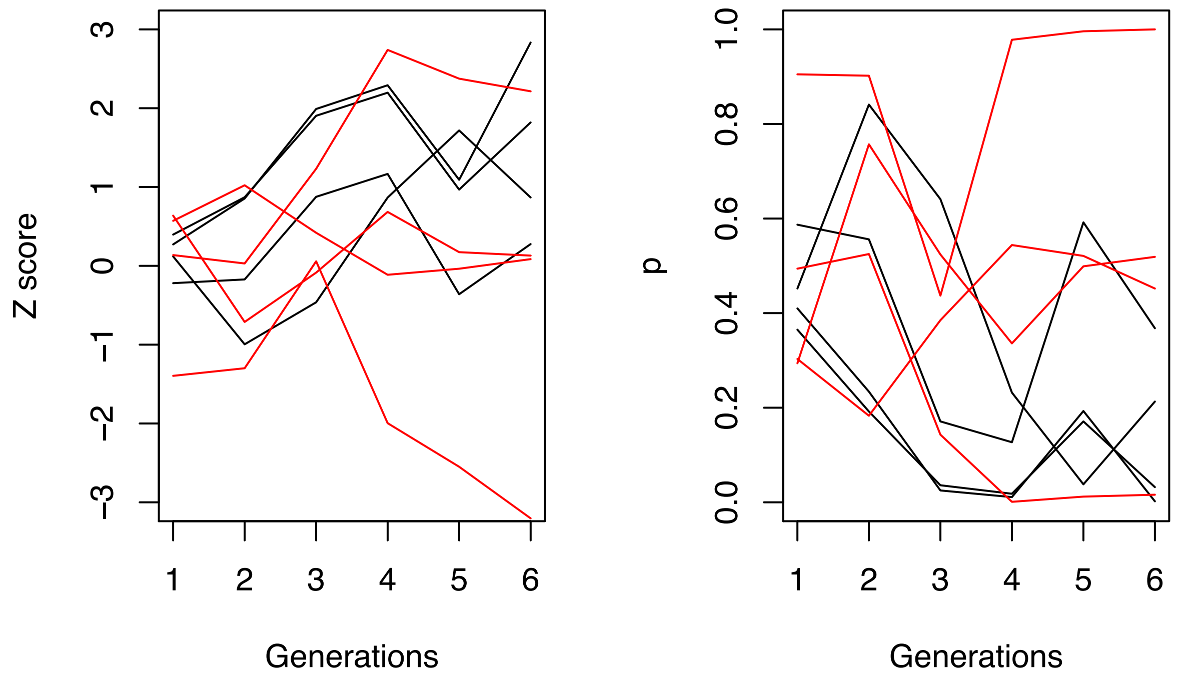
```
# Set the random seed for reproducibility
set.seed(1278)

# what factor should the data be split by?
split = finalLangs[finalLangs$cond2==unique(finalLangs$cond2)[1],]$Shape

# for each language (a single generation's output)
res = tapply(finalLangs$RatedSpikiness, finalLangs$cond2, function(X){
  # calculate the true difference
  trueDiff = -diff(tapply(X, split,mean))
  # permute the numbers and re-calculate difference
  permDiff = replicate(1000,
        {-diff(tapply(sample(X), split,mean)) })

  # work out p and z-socres
  p = 1- sum(trueDiff > permDiff ) / length(permDiff)
  z.score = (trueDiff - mean(permDiff)) / sd(permDiff)
  return(c(p,z.score))
})
```

Recast results into data frame:

```
res2 = data.frame(
p = sapply(res,head,n=1),
z = sapply(res,tail,n=1),
t(sapply(names(res),function(X){
  strsplit(X," ")[[1]]
}))
, stringsAsFactors = F)
names(res2) = c("p",'z','condition','chain','gen')
```

Plot the results. Each line represents an independent chain. The red lines show the results for the Learning condition. The results suggest that there is no strong difference between the conditions. One of the learning chains decreases in iconicity, due to that chain focussing on specifying the colour rather than the shape.

```
par(mfrow=c(1,2))
plot(c(1,6), c(-3,3), type='n',ylab='Z score', xlab='Generations')
for(i in unique(res2$chain)){
  dx =res2[res2$chain==i,]
  lines(dx$gen,dx$z, col=c("black","red")[(dx$condition=="Learn")+1])
}

plot(c(1,6), c(0,1), type='n',ylab='p', xlab='Generations')
for(i in unique(res2$chain)){
  dx =res2[res2$chain==i,]
  lines(dx$gen,dx$p, col=c("black","red")[(dx$condition=="Learn")+1])
}
```



```
rownames(res2) = NULL
res2
```

```
##        p          z     condition chain gen
## 1  0.365  0.39651582 Communication     0   1
## 2  0.192  0.87023750 Communication     0   2
## 3  0.036  1.90398924 Communication     0   3
## 4  0.018  2.19686551 Communication     0   4
## 5  0.171  0.96732544 Communication     0   5
## 6  0.032  1.81968720 Communication     0   6
## 7  0.410  0.27163156 Communication     1   1
## 8  0.234  0.85438818 Communication     1   2
## 9  0.025  1.99017220 Communication     1   3
## 10 0.011  2.29125403 Communication     1   4
## 11 0.193  1.09194544 Communication     1   5
## 12 0.002  2.83378086 Communication     1   6
## 13 0.452  0.11931504 Communication     2   1
## 14 0.841 -0.99573110 Communication     2   2
## 15 0.641 -0.46244255 Communication     2   3
## 16 0.232  0.86643742 Communication     2   4
## 17 0.038  1.71762100 Communication     2   5
## 18 0.213  0.86711511 Communication     2   6
## 19 0.587 -0.21997100 Communication     3   1
## 20 0.556 -0.17252929 Communication     3   2
## 21 0.171  0.87557683 Communication     3   3
## 22 0.127  1.16617467 Communication     3   4
## 23 0.592 -0.36031071 Communication     3   5
## 24 0.368  0.27612849 Communication     3   6
## 25 0.303  0.57103499         Learn     4   1
## 26 0.183  1.02227196         Learn     4   2
## 27 0.385  0.41949120         Learn     4   3
## 28 0.544 -0.11348068         Learn     4   4
## 29 0.521 -0.03655111         Learn     4   5
## 30 0.452  0.08509545         Learn     4   6
## 31 0.494  0.13627918         Learn     5   1
## 32 0.525  0.02997840         Learn     5   2
## 33 0.143  1.23116378         Learn     5   3
## 34 0.001  2.73954890         Learn     5   4
## 35 0.012  2.37478332         Learn     5   5
## 36 0.016  2.21463208         Learn     5   6
## 37 0.905 -1.39552168         Learn     6   1
## 38 0.902 -1.29934990         Learn     6   2
## 39 0.437  0.05754949         Learn     6   3
## 40 0.978 -1.99616121         Learn     6   4
## 41 0.996 -2.54919070         Learn     6   5
## 42 1.000 -3.20395726         Learn     6   6
## 43 0.294  0.63720477         Learn     7   1
## 44 0.757 -0.71198396         Learn     7   2
## 45 0.524 -0.08605136         Learn     7   3
## 46 0.336  0.68413893         Learn     7   4
## 47 0.499  0.17370403         Learn     7   5
## 48 0.519  0.12991654         Learn     7   6
```

Looking only at red-coloured meanings

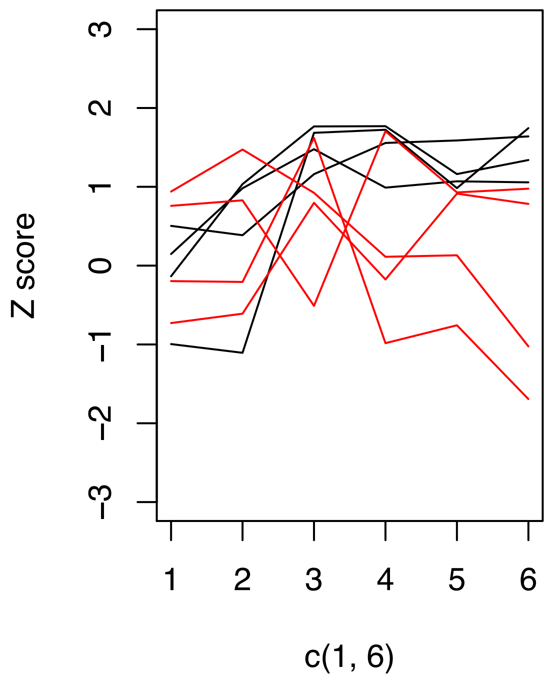
The mixed effects results suggested that there is a difference between conditions. The absence of an effect of condition on iconicity here is probably due to the fact that condition may interact with the other meaning dimension colour and border (e.g. iconicity may be stronger in words for red objects than green or blue objects).

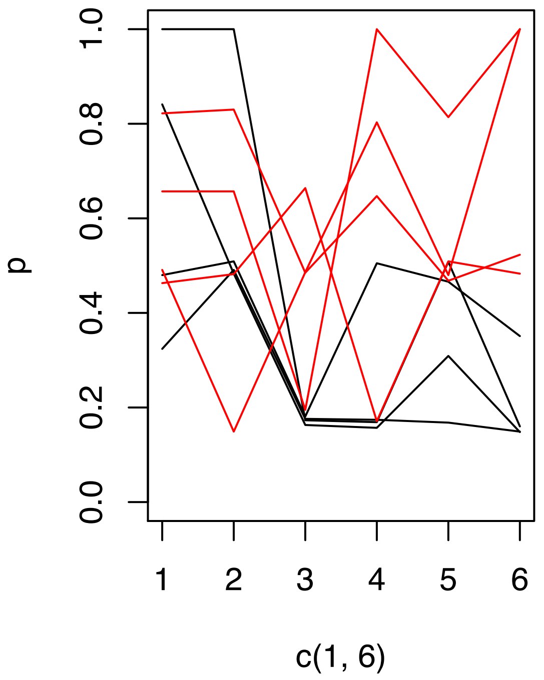
Below we run the same analysis, but just for the colour 'Red'.

```
finalLangs2 = finalLangs[finalLangs$Colour=="Rojo",]


# Set the random seed for reproducibility
set.seed(1278)

# what factor should the data be split by?
split = finalLangs2[finalLangs2$cond2==unique(finalLangs2$cond2)[1],]$Shape

# for each language (a single generation's output)
resRed = tapply(finalLangs2$RatedSpikiness, finalLangs2$cond2, function(X){
  # calculate the true difference
  trueDiff = -diff(tapply(X, split,mean))
  # permute the numbers and re-calculate difference
  permDiff = replicate(1000,
        {-diff(tapply(sample(X), split,mean)) })

  # work out p and z-socres
  p = 1- sum(trueDiff > permDiff ) / length(permDiff)
  z.score = (trueDiff - mean(permDiff)) / sd(permDiff)
  return(c(p,z.score))
})
res2Red = data.frame(
p = sapply(resRed,head,n=1),
z = sapply(resRed,tail,n=1),
t(sapply(names(resRed),function(X){
  strsplit(X," ")[[1]]
}))
, stringsAsFactors = F)
names(res2Red) = c("p",'z','condition','chain','gen')
```

Plot the results. In this case, we do see a division between the two conditions by the last generation.

```
par(mfrow=c(1,2))
plot(c(1,6), c(-3,3), type='n',ylab='Z score')
for(i in unique(res2Red$chain)){
  dx =res2Red[res2Red$chain==i,]
  lines(dx$gen,dx$z, col=c("black","red")[(dx$condition=="Learn")+1])
}

plot(c(1,6), c(0,1), type='n',ylab='p')
for(i in unique(res2Red$chain)){
  dx =res2Red[res2Red$chain==i,]
  lines(dx$gen,dx$p, col=c("black","red")[(dx$condition=="Learn")+1])
}
```


Z score
3
2
1
0
−1
−2
−3
1
2
3
4
5
6
c(1, 6)


p
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
c(1, 6)

5. R code for statistical tests

Mixed effects modelling was carried out in R using the lme4 package. All code and output is available as an R markdown document included below (also available here alongside the raw data: https://github.com/seannyD/ILMIconicity).

The interactive origin of iconiciy: Estimating spikiness ratings

Load libraries

```r
library(ngram)
library(gplots)
library(lme4)
library(party)
```

Helper functions

Estimate the spikiness ratngs of words from a finite sample of participant judgements, controlling for a random effect for each participant.

```r
predictSpikinessWithLMER = function(ratings.words){
  m.words = lmer(RatingSpikiness~ Item + (1|Part), data=ratings.words)
  #plot(ratings.words$RatingSpikiness,resid(m.words))
  words = sort(unique(ratings.words$Item))
  words.predictions = predict(m.words,newdata=data.frame(Item=words, Part=1), re.form=NULL)
  names(words.predictions) = words
  #cor(words.predictions, tapply(ratings.words$RatingSpikiness, ratings.words$Item, mean))
  return(words.predictions)
}
```

Take a set of words and generate a feature matrix of ngrams.

```r
makeFeatureFrame = function(dx,ngrams){
  r = matrix(nrow=nrow(dx), ncol=2+length(ngrams))
  r[,1] = dx$Item
  r[,2] = dx$RatingSpikiness
  colnames(r) = c("Item","RatingSpikiness",ngrams)
  for(i in 3:ncol(r)){
    r[,i] = grepl(colnames(r)[i],r[,1])
  }
  r = as.data.frame(r)
  for(i in 3:ncol(r)){
    r[,i] = as.logical(r[,i])
  }
  return(r)
}
```

Load data

The data includes spikiness ratings for words and individual letters from several participants.

```r
ratings = read.delim("../data/ratings/SpikinessRatings", sep='\t', stringsAsFactors = F)
```

Check whether there are effects by participant sex, age or the direction that the Likert scale was presented.

```
m0 = lmer(RatingSpikiness ~ Sex + Age + Likert + (1|Item)  + (1|Part), data=ratings)
summary(m0)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: RatingSpikiness ~ Sex + Age + Likert + (1 | Item) + (1 | Part)
##    Data: ratings
##
## REML criterion at convergence: 9124.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3831 -0.6667  0.0133  0.6978  3.0814
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Item     (Intercept) 1.4523   1.2051
##  Part     (Intercept) 0.0917   0.3028
##  Residual             2.1603   1.4698
## Number of obs: 2411, groups:  Item, 163; Part, 16
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)  3.942780   0.448760   8.786
## Sexmale      0.091512   0.192050   0.477
## Age          0.002476   0.018451   0.134
## Likertspiky -0.040992   0.173811  -0.236
##
## Correlation of Fixed Effects:
##             (Intr) Sexmal Age
## Sexmale      0.025
## Age         -0.929 -0.215
## Likertspiky -0.288  0.283  0.084
```

There are no significant effects.

Split the data into ratings for letters and ratings for whole words. Then get an esimation of the mean rating for each item.

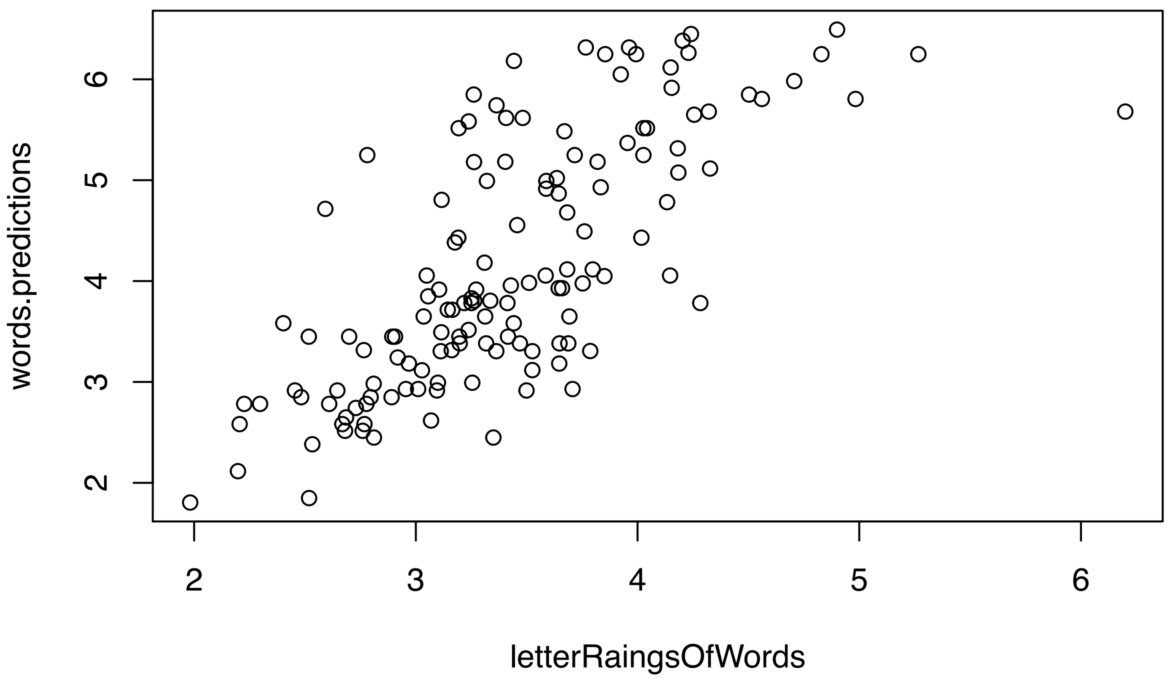
```
ratings.letters = ratings[nchar(ratings$Item)==1,]
ratings.words = ratings[nchar(ratings$Item)>1,]


letter.predictions = predictSpikinessWithLMER(ratings.letters)
words.predictions = predictSpikinessWithLMER(ratings.words)
```

Model based on letter scores

Estimate the spikiness rating of a word by taking the mean spikiness score for each of the letters in the word. This can be used as baseline to see if it's worth building a more complicated model.

```
letterRaingsOfWords = sapply(names(words.predictions), function(X){
  mean(letter.predictions[strsplit(X,'')[[1]]])
})
```

```r
plot(letterRaingsOfWords, words.predictions)
```



```r
# Baseline for just using letters:
cor(letterRaingsOfWords, words.predictions)
```

```
## [1] 0.733007
```

The model predictions correlate with the real values with r = 0.73 (on seen data).

Random forests model based on unigrams and bigrams

Build a model of spikiness ratings based on a training set, then predict the spikiness ratings of an unseen test set.

Set parameters:

```r
proportionOfDataInTrainingSet = 0.75
numberOfFolds = 20
maxNGram = 2
```

Run the trainig and test cycles:

```r
# set random seed
set.seed(2189)

# variable for storing correlation between predictions and real ratings for each run
res = c()

for(run in 1:numberOfFolds){
  # get list of items
  items = unique(ratings$Item)
  # select training items:
  #  all single characters plus a random selection of words
  trainItems = c(items[nchar(items)==1],
                 sample(items[nchar(items)>1],
                 sum(nchar(items)>1)*proportionOfDataInTrainingSet))
  # test items - unseen items
  testItems = items[!items %in% trainItems]

  # get data for training and test items
  trainSet = ratings[ratings$Item %in% trainItems,]
  testSet = ratings[ratings$Item %in% testItems,]

  # make list of ngrams in training set
  ngrams = unique(unlist(sapply(trainSet$Item, function(X){
    if(nchar(X)==1){
      return(X)
    }
    unique(ngram_asweka(X,min=1,max=maxNGram,sep=''))
  })))

  # make feature frame of ngrams
  rTrain = makeFeatureFrame(trainSet,ngrams)
  rTrain$RatingSpikiness = as.numeric(rTrain$RatingSpikiness)

  # predict mean spikiness with lmer for test set
  rTest.predictions = predictSpikinessWithLMER(testSet)
  # biuld feature frame of ngrams for test set
  rTest = makeFeatureFrame(testSet[!duplicated(testSet$Item),],ngrams)
  rTest$RatingSpikiness = rTest.predictions[rTest$Item]

  colselect = 2:ncol(rTrain)

  # Build the random forest
  cf = cforest(RatingSpikiness ~ . ,
               data= rTrain[,colselect],
               controls = cforest_control(mtry = 10))
  #importance=  varimp(cf)
  #dotplot(sort(importance))

  predictedRatings = predict(cf,newdata=rTest[,colselect])

  res = c(res,cor(predictedRatings,rTest$RatingSpikiness))
}
```

The mean correlation between predictions and real data was r = 0.886. This is an acceptable level and a marked improvement on the baseline model (also consiering the random forests predictions were on unseen data).

Informal testing found that performance did not increase significantly when including trigrams.

Make model with whole data

```
ngrams.all = unique(unlist(sapply(ratings$Item, function(X){
  if(nchar(X)==1){
    return(X)
  }
  unique(ngram_asweka(X,min=1,max=maxNGram,sep=''))
})))

rAll = makeFeatureFrame(ratings,ngrams.all)
rAll$RatingSpikiness = as.numeric(rAll$RatingSpikiness)

rAll.predictions = predictSpikinessWithLMER(ratings)
rAll = makeFeatureFrame(ratings[!duplicated(ratings$Item),],ngrams.all)
rAll$RatingSpikiness = rAll.predictions[rAll$Item]

cf.all = cforest(RatingSpikiness ~ . ,
            data= rAll[,2:ncol(rAll)],
            controls = cforest_control(mtry = 10))

tx =ctree(RatingSpikiness ~ ., data=rAll[,2:ncol(rAll)])
```

Build a function to predict iconicity results.

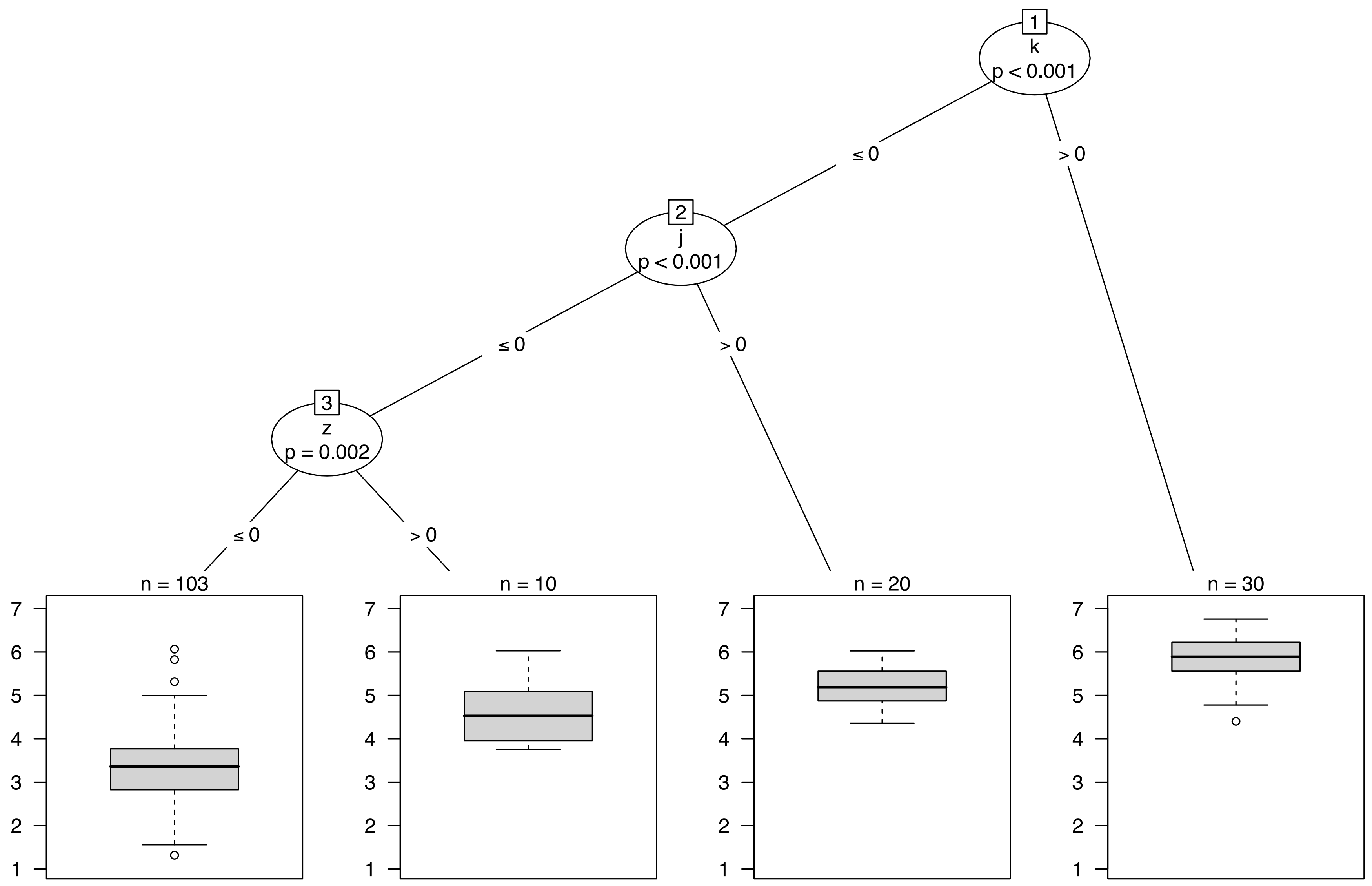
```
getIconicityFromRForest = function(words){

  xdat = t(sapply(
    words,
    function(word){
      sapply(
        ngrams.all,
        function(X){
          grepl(X,word)
    })}))
  xdat = as.data.frame(cbind(rep(NA,nrow(xdat)),rep(NA,nrow(xdat)),xdat))

  predictedRatings = predict(cf.all,newdata=xdat)
  return(as.vector(predictedRatings))
}
```

Save the function and variables to be used in other scripts.

```
save(getIconicityFromRForest, ngrams.all, cf.all, file='PredictSpikinessModel.RDat')
```

Here's a sample tree from the forest:

```r
plot(tx, terminal_panel=node_boxplot(tx,id=F))
```



The interactive origin of iconiciy: Mixed effects models

Contents

Introduction **2**

Spikiness ratings **2**

Load libraries 2
Load data 2
Mixed effects model 3
Mixed effects model with binarised spikiness ratings 9
Binary tree analysis 14

Iconicity of innovations **15**

Load data 15
Distribution of iconicity in innovations 16
Increase in iconicity by survival 23

Accuracy **27**

Mixed effects model 27
Iconicity and accuracy for innovations 29
Iconicity and accuracy for whole data 31

Iconicity and systematicity **33**

Mixed effects model 34
Results 35

Introduction

This file contains an analysis of the spikiness ratings of the final output languages and the accuracy of guessing during the experiments. The spikiness ratings are not bimodally disributed, so the analysis of spikiness ratings is done using both the continuous spikiness rating values and a binarised version of the ratings.

Note that in the main text, we refer to the two conditions as "communication" and "reproduction", while the data is coded as "communication" and "learning".

Spikiness ratings

Load libraries

```
library(gplots)
library(lattice)
library(ggplot2)
library(lme4)
library(party)
library(sjPlot)
library(lawstat)
```

Load data

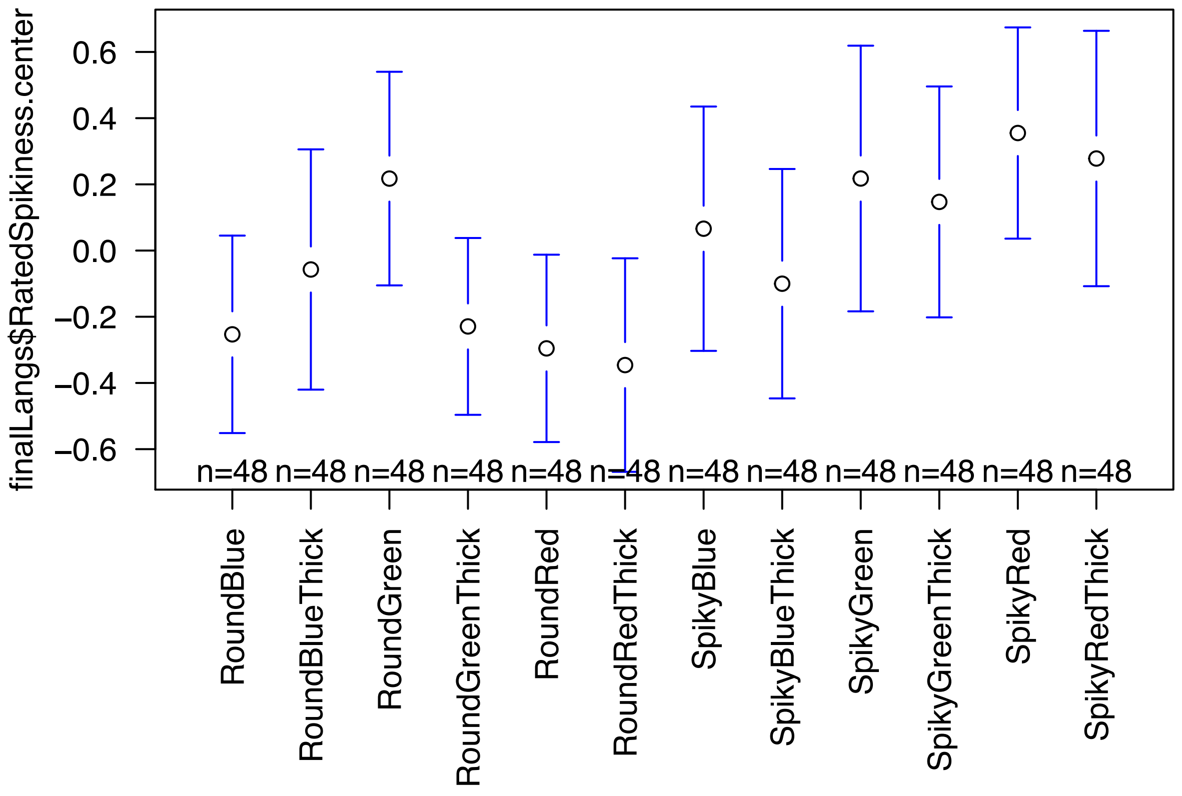
```
finalLangs = read.csv("../data/finalLanguages/FinalLanguages.csv", stringsAsFactors = F)
# convert labels to English
finalLangs$Shape[finalLangs$Shape=="Picudo"] = "Spiky"
finalLangs$Shape[finalLangs$Shape=="Redondo"] = "Round"

# load all trial data
alldatx = read.csv("../results/AllTrialData.csv",stringsAsFactors = F)
```

Center spikiness ratings and re-level factors.

```
finalLangs$RatedSpikiness.center =
  finalLangs$RatedSpikiness- mean(finalLangs$RatedSpikiness)

finalLangs$Cond = factor(finalLangs$Cond, levels=c("Learn","Communication"))
finalLangs$Shape = factor(finalLangs$Shape, levels=c("Round","Spiky"))
```

Plot the data by item (all conditions, all generations)

```
par(mar=c(8,4,2,2))
plotmeans(finalLangs$RatedSpikiness.center~finalLangs$Item, las=2, xlab="", connect=F)
```



There are differences between items

Mixed effects model

Build a series of models with random effects for Chain and Item.

```
# null model
m0 = lmer(RatedSpikiness.center ~ 1 + (1 |Chain)  + (1|Item), data=finalLangs)
# + condition
m1 = lmer(RatedSpikiness.center ~ Cond + (1 |Chain)  + (1|Item), data=finalLangs)
# + generation
m2 = lmer(RatedSpikiness.center ~ Cond + Gen + (1 |Chain) + (1|Item), data=finalLangs)
# + shape
m3 = lmer(RatedSpikiness.center ~ Cond + Gen + Shape + (1 |Chain)
          + (1|Item), data=finalLangs)
# + interaction between shape and generation
m4 = lmer(RatedSpikiness.center ~ Cond + (Gen * Shape) + (1 |Chain)
          + (1|Item), data=finalLangs)
# + interaction between condition and generation
m5 = lmer(RatedSpikiness.center ~ (Cond*Gen) + (Gen * Shape) + (1 |Chain)
          + (1|Item), data=finalLangs)
# + interaction between shape and condition
m6 = lmer(RatedSpikiness.center ~ (Cond*Gen) + (Gen * Shape) + (Shape:Cond)
          + (1 |Chain)  + (1|Item), data=finalLangs)
# + 3-way interaction
m7 = lmer(RatedSpikiness.center ~ Cond * Gen * Shape + (1 |Chain)
          + (1|Item), data=finalLangs)
```

Results

Look inside main model

```r
summary(m7)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: RatedSpikiness.center ~ Cond * Gen * Shape + (1 | Chain) + (1 |
##     Item)
##    Data: finalLangs
##
## REML criterion at convergence: 1767.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.8411 -0.8370 -0.1665  0.7906  2.3066
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Item     (Intercept) 0.01058  0.1029
##  Chain    (Intercept) 0.18043  0.4248
##  Residual             1.17881  1.0857
## Number of obs: 576, groups:  Item, 12; Chain, 8
##
## Fixed effects:
##                                Estimate Std. Error t value
## (Intercept)                    0.022530   0.299064   0.075
## CondCommunication              0.096751   0.418750   0.231
## Gen                           -0.033860   0.052978  -0.639
## ShapeSpiky                    -0.003530   0.297764  -0.012
## CondCommunication:Gen         -0.064573   0.074923  -0.862
## CondCommunication:ShapeSpiky  -0.032181   0.412642  -0.078
## Gen:ShapeSpiky                 0.002764   0.074923   0.037
## CondCommunication:Gen:ShapeSpiky  0.189234   0.105957   1.786
##
## Correlation of Fixed Effects:
##              (Intr) CndCmm Gen    ShpSpk CndC:G CnC:SS Gn:ShS
## CondCmmnctn -0.700
## Gen         -0.620  0.443
## ShapeSpiky  -0.498  0.341  0.623
## CndCmmnct:G  0.438 -0.626 -0.707 -0.440
## CndCmmnc:SS  0.345 -0.493 -0.449 -0.693  0.635
## Gen:ShpSpky  0.438 -0.313 -0.707 -0.881  0.500  0.635
## CndCmm:G:SS -0.310  0.443  0.500  0.623 -0.707 -0.899 -0.707
```

Test the differences between model fits.

```r
anova(m0,m1,m2,m3,m4,m5,m6,m7)
```

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

```
## Data: finalLangs
## Models:
## m0: RatedSpikiness.center ~ 1 + (1 | Chain) + (1 | Item)
## m1: RatedSpikiness.center ~ Cond + (1 | Chain) + (1 | Item)
## m2: RatedSpikiness.center ~ Cond + Gen + (1 | Chain) + (1 | Item)
## m3: RatedSpikiness.center ~ Cond + Gen + Shape + (1 | Chain) + (1 |
## m3:     Item)
## m4: RatedSpikiness.center ~ Cond + (Gen * Shape) + (1 | Chain) +
## m4:     (1 | Item)
## m5: RatedSpikiness.center ~ (Cond * Gen) + (Gen * Shape) + (1 | Chain) +
## m5:     (1 | Item)
## m6: RatedSpikiness.center ~ (Cond * Gen) + (Gen * Shape) + (Shape:Cond) +
## m6:     (1 | Chain) + (1 | Item)
## m7: RatedSpikiness.center ~ Cond * Gen * Shape + (1 | Chain) + (1 |
## m7:     Item)
##    Df    AIC    BIC  logLik deviance   Chisq Chi Df Pr(>Chisq)
## m0  4 1779.7 1797.1 -885.83   1771.7
## m1  5 1781.2 1803.0 -885.61   1771.2  0.4475      1  0.5035471
## m2  6 1782.8 1808.9 -885.40   1770.8  0.4234      1  0.5152634
## m3  7 1777.7 1808.2 -881.87   1763.7  7.0627      1  0.0078704 **
## m4  8 1776.4 1811.3 -880.21   1760.4  3.3049      1  0.0690737 .
## m5  9 1778.1 1817.3 -880.05   1760.1  0.3156      1  0.5742584
## m6 10 1768.1 1811.6 -874.04   1748.1 12.0326      1  0.0005228 ***
## m7 11 1766.9 1814.8 -872.43   1744.9  3.2087      1  0.0732495 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There was a significant main effect of shape (beta = -0.0035 , std.err = 0.3 , Wald t = -0.012 ; log likelihood difference = 3.5 , df = 1 , Chi Squared = 7.06 , p = 0.0079).

There was a significant interaction between shape and condition (beta = -0.032 , std.err = 0.41 , Wald t = -0.078 ; log likelihood difference = 6 , df = 1 , Chi Squared = 12.03 , p = 0.00052).

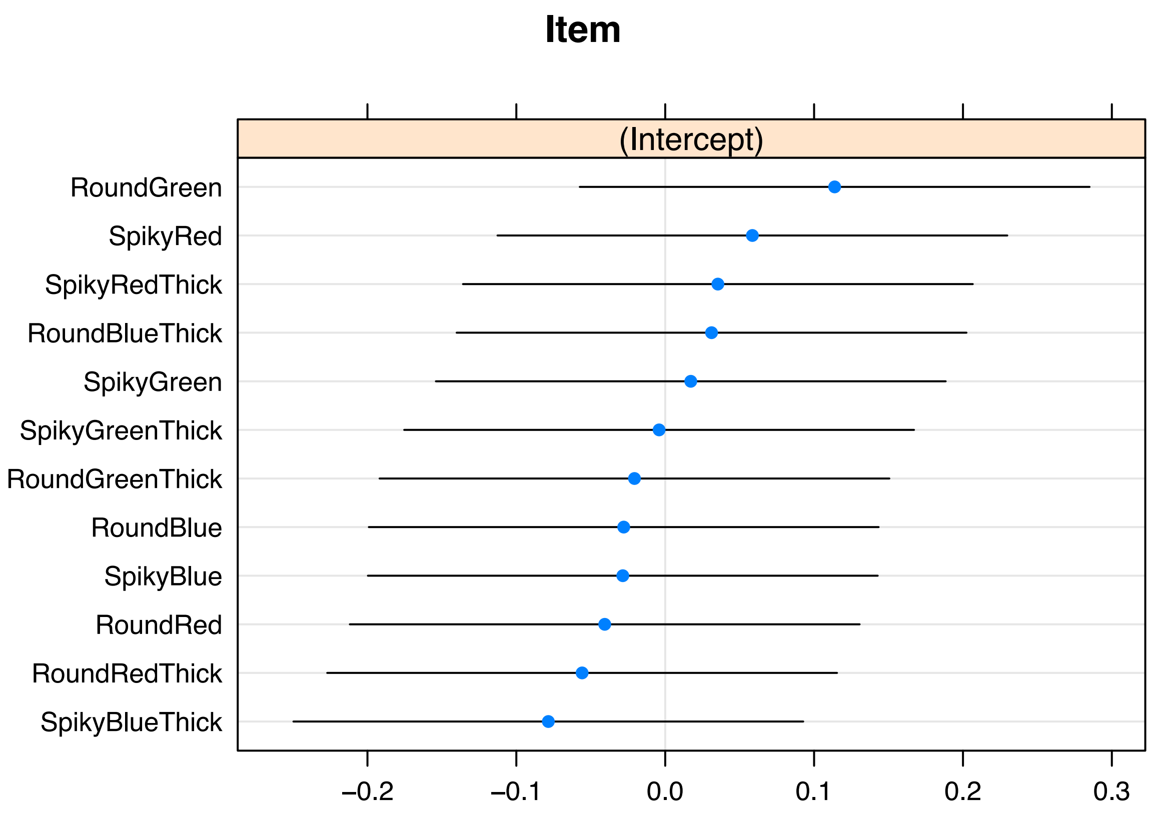
There was a marginal interaction between shape and generation (beta = 0.0028 , std.err = 0.075 , Wald t = 0.037 ; log likelihood difference = 1.7 , df = 1 , Chi Squared = 3.3 , p = 0.069).

There was a marginal three-way interaction between shape, condition and generation (beta = 0.19 , std.err = 0.11 , Wald t = 1.8 ; log likelihood difference = 1.6 , df = 1 , Chi Squared = 3.21 , p = 0.073).

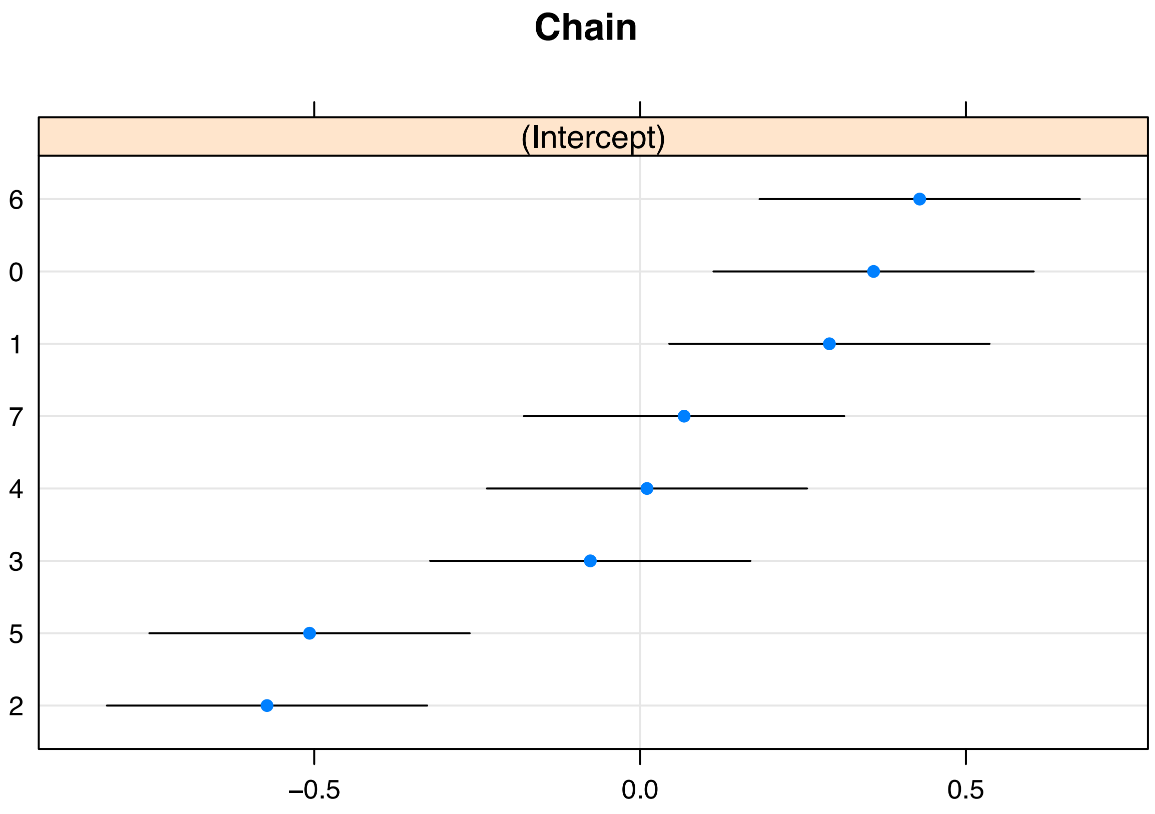
Plot the random effects.

```
dotplot(ranef(m7, condVar=T))
```

```
## $Item
```

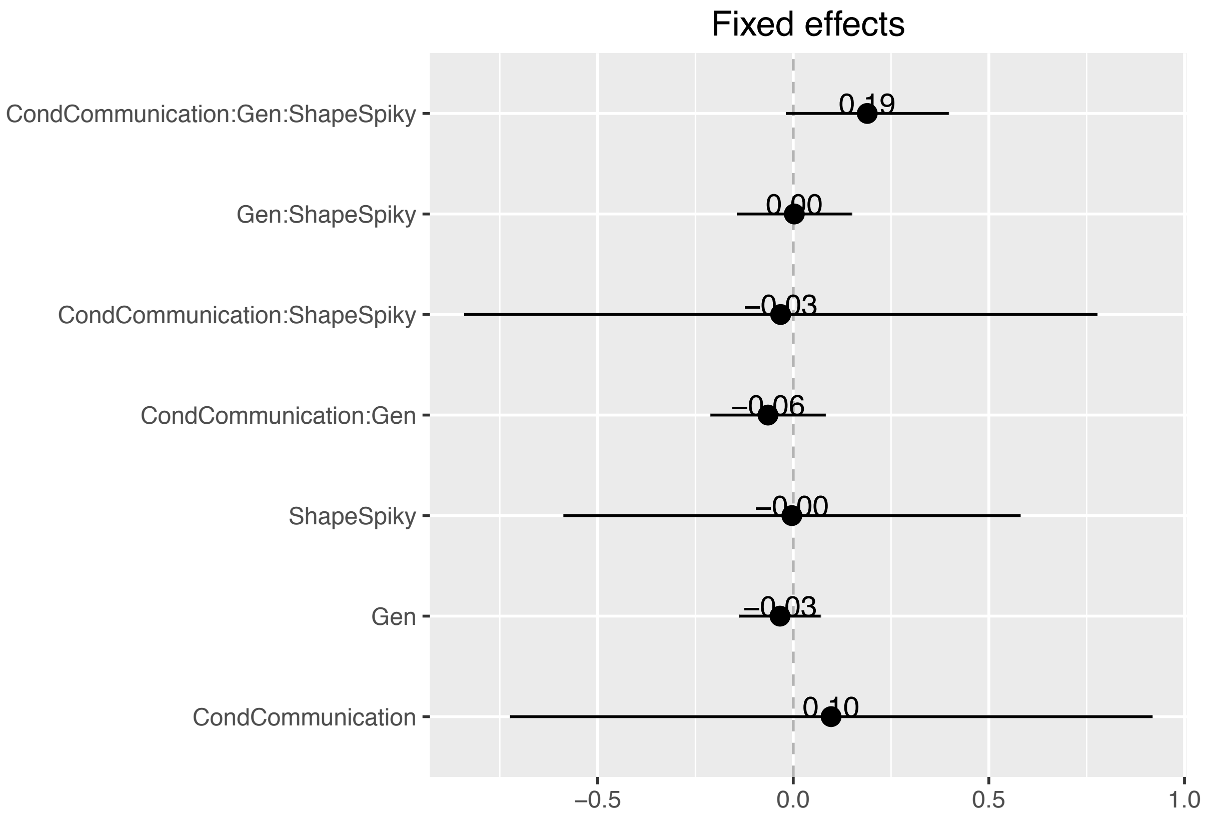


```
##
## $Chain
```



Plot the fixed effects with error estiamtes from the final model. The 3-way interaction between condition, generation and shape is marginaly significant:

```r
sjp.lmer(m7, type='fe', geom.colors=c(1,1))
```

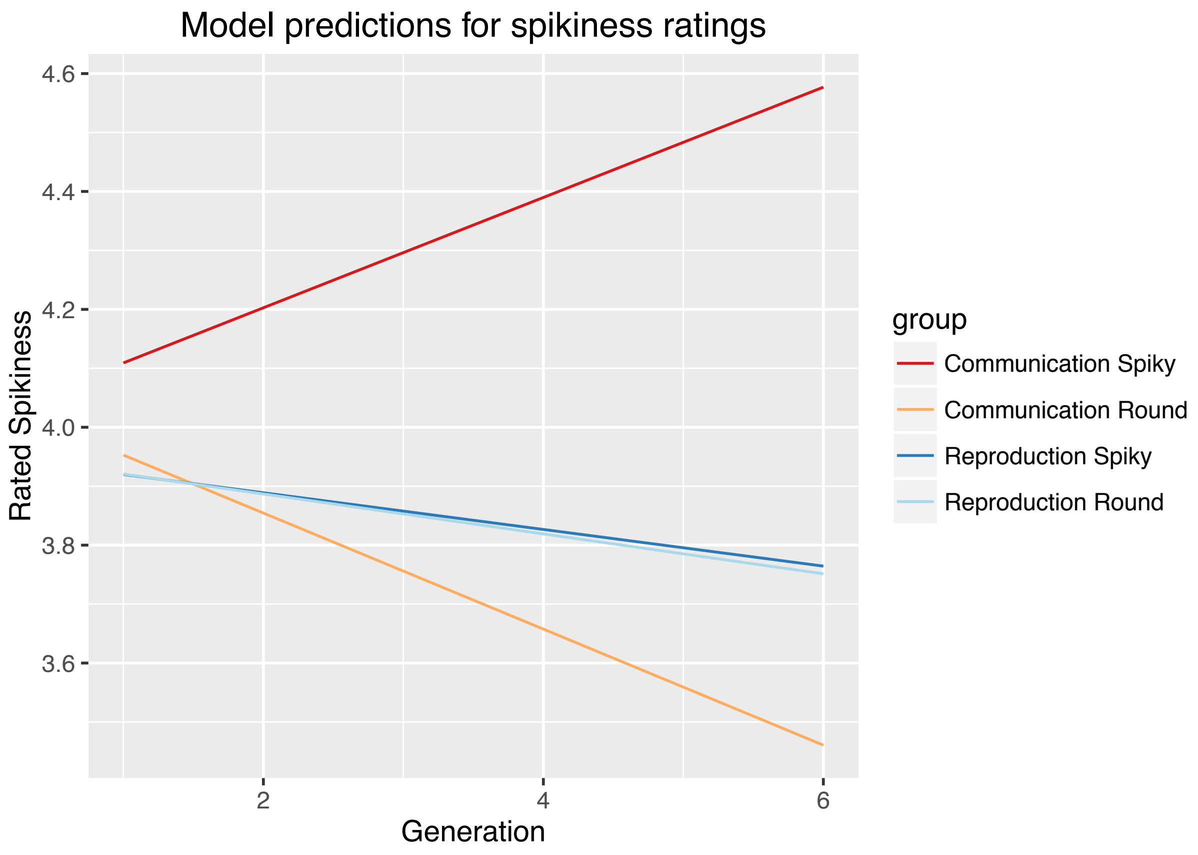


Visualise the three-way interaction. The first part of this script creates some new data and uses the model to predict values.

The interaction is driven by Spiky and Round shapes diverging over generations in the communication condition, but not in the reproduction condition.

```r
predictedData = data.frame(
  RatedSpikiness.center = NA,
  Gen=rep(1:6,4),
  Shape= rep(rep(c("Spiky","Round"),each=6),2),
  Cond = rep(c("Communication","Learn"),each=12)
)
predictedData$RatedSpikiness.center = predict(m7,
                                newdata=predictedData, re.form=NA)
#Re-scale
predictedData$RatedSpikiness =
  predictedData$RatedSpikiness.center + mean(finalLangs$RatedSpikiness)

# Make groups and set order for plotting
predictedData$group = paste(predictedData$Cond, predictedData$Shape)
predictedData$group = gsub("Learn","Reproduction", predictedData$group)
predictedData$group = factor(
  predictedData$group,
  levels = c("Communication Spiky","Communication Round",
             "Reproduction Spiky", "Reproduction Round"))
```

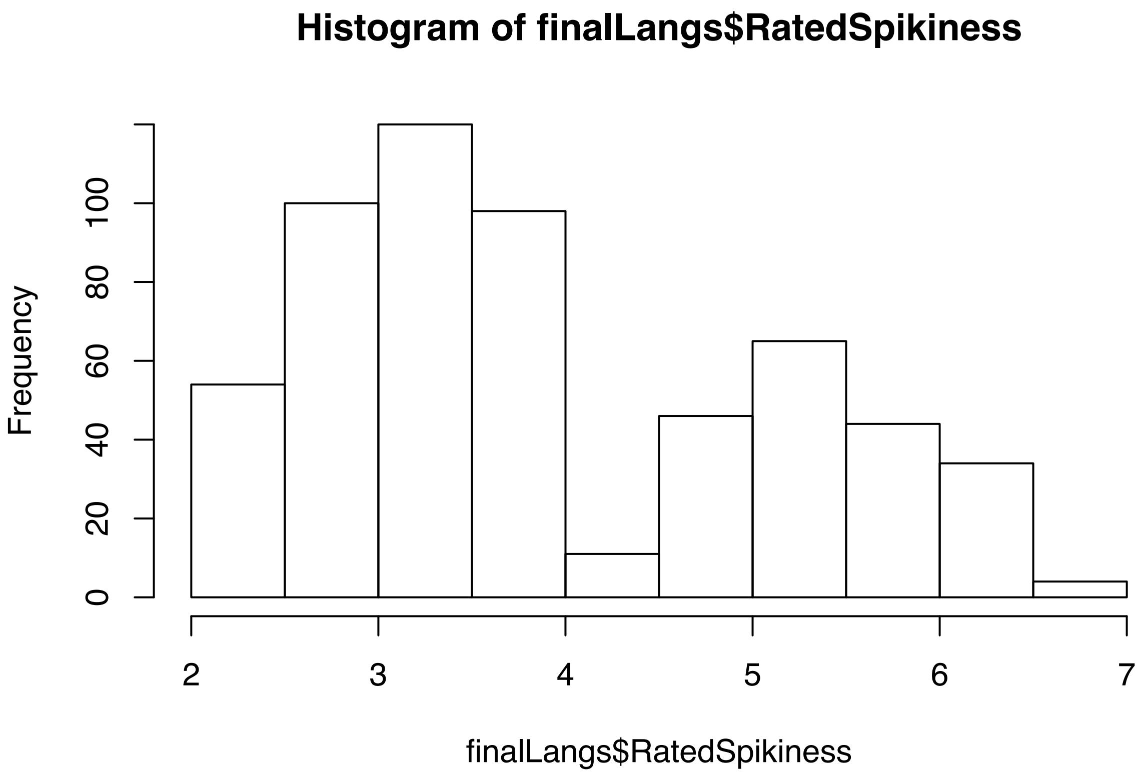
```r
# Plot the predictions
qplot(Gen, RatedSpikiness,
      data=predictedData,
      color=group,
      geom=c("line")) +
  scale_color_manual(values=c("#d7191c", "#fdae61", "#2c7bb6", "#abd9e9")) +
  xlab("Generation") +
  ylab("Rated Spikiness") +
  ggtitle("Model predictions for spikiness ratings")
```



Mixed effects model with binarised spikiness ratings

The spikiness ratings are not normally distributed:

```
hist(finalLangs$RatedSpikiness)
```



So we binarise the variable into spiky/not spiky:

```
finalLangs$RatedSpikiness.bin = finalLangs$RatedSpikiness >4
```

Run a series of models. Note that intermediate models 5 and 6 do not converge, but the final model 7 does.

```
mcontrol = glmerControl(optCtrl = list(maxfun = 500000))

mb0 = glmer(RatedSpikiness.bin ~ 1 + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
mb1 = glmer(RatedSpikiness.bin ~ Cond + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
mb2 = glmer(RatedSpikiness.bin ~ Cond + Gen + (1 |Chain) + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
mb3 = glmer(RatedSpikiness.bin ~ Cond + Gen + Shape + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
mb4 = glmer(RatedSpikiness.bin ~ Cond + (Gen * Shape) + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
mb5 = glmer(RatedSpikiness.bin ~ (Cond*Gen) + (Gen * Shape) + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control
## $checkConv, : Model failed to converge with max|grad| = 0.00112016 (tol =
## 0.001, component 1)
```

```
mb6 = glmer(RatedSpikiness.bin ~ (Cond*Gen) + (Gen * Shape) + (Shape:Cond) + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control
## $checkConv, : Model failed to converge with max|grad| = 0.00254262 (tol =
## 0.001, component 1)
```

```
mb7 = glmer(RatedSpikiness.bin ~ Cond * Gen * Shape + (1 |Chain)  + (1|Item),
            data=finalLangs, family=binomial, control = mcontrol)
```

Results

Look inside main model

```
summary(mb7)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: RatedSpikiness.bin ~ Cond * Gen * Shape + (1 | Chain) + (1 |
##     Item)
##    Data: finalLangs
## Control: mcontrol
##
##      AIC      BIC   logLik deviance df.resid
##    722.9    766.4   -351.4    702.9      566
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.4001 -0.7152 -0.4951  0.9714  2.5752
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  Item   (Intercept) 0.06298  0.2510
##  Chain  (Intercept) 0.30153  0.5491
## Number of obs: 576, groups:  Item, 12; Chain, 8
##
## Fixed effects:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -0.80967    0.50900  -1.591    0.112
## CondCommunication                0.11711    0.72308   0.162    0.871
## Gen                              0.06152    0.10567   0.582    0.560
## ShapeSpiky                       0.52479    0.60063   0.874    0.382
## CondCommunication:Gen           -0.25227    0.16195  -1.558    0.119
## CondCommunication:ShapeSpiky    -0.06135    0.83301  -0.074    0.941
## Gen:ShapeSpiky                  -0.16967    0.15042  -1.128    0.259
## CondCommunication:Gen:ShapeSpiky  0.39112    0.21894   1.786    0.074 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndCmm Gen    ShpSpk CndC:G CnC:SS Gn:ShS
## CondCmmnctn -0.675
## Gen         -0.736  0.518
## ShapeSpiky  -0.600  0.398  0.623
## CndCmmnct:G  0.480 -0.751 -0.653 -0.407
## CndCmmnc:SS  0.407 -0.617 -0.449 -0.679  0.652
## Gen:ShpSpky  0.518 -0.364 -0.703 -0.871  0.459  0.628
## CndCmm:G:SS -0.356  0.556  0.483  0.599 -0.740 -0.893 -0.687
```

Test model comparison:

```
anova(mb0,mb1,mb2,mb3,mb4,mb5,mb6,mb7)
```

```
## Data: finalLangs
## Models:
## mb0: RatedSpikiness.bin ~ 1 + (1 | Chain) + (1 | Item)
## mb1: RatedSpikiness.bin ~ Cond + (1 | Chain) + (1 | Item)
## mb2: RatedSpikiness.bin ~ Cond + Gen + (1 | Chain) + (1 | Item)
## mb3: RatedSpikiness.bin ~ Cond + Gen + Shape + (1 | Chain) + (1 |
## mb3:     Item)
## mb4: RatedSpikiness.bin ~ Cond + (Gen * Shape) + (1 | Chain) + (1 |
## mb4:     Item)
## mb5: RatedSpikiness.bin ~ (Cond * Gen) + (Gen * Shape) + (1 | Chain) +
## mb5:     (1 | Item)
## mb6: RatedSpikiness.bin ~ (Cond * Gen) + (Gen * Shape) + (Shape:Cond) +
## mb6:     (1 | Chain) + (1 | Item)
## mb7: RatedSpikiness.bin ~ Cond * Gen * Shape + (1 | Chain) + (1 |
## mb7:     Item)
##     Df    AIC    BIC  logLik deviance   Chisq Chi Df Pr(>Chisq)
## mb0  3 729.66 742.72 -361.83   723.66
## mb1  4 731.64 749.07 -361.82   723.64  0.0130      1  0.9092167
## mb2  5 733.09 754.87 -361.54   723.09  0.5560      1  0.4558874
## mb3  6 730.23 756.37 -359.12   718.23  4.8538      1  0.0275855 *
## mb4  7 732.22 762.71 -359.11   718.22  0.0115      1  0.9147795
## mb5  8 734.12 768.97 -359.06   718.12  0.1001      1  0.7517608
## mb6  9 724.09 763.29 -353.04   706.09 12.0352      1  0.0005221 ***
## mb7 10 722.88 766.44 -351.44   702.88  3.2044      1  0.0734423 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There was a significant main effect of shape (beta = 0.52 , std.err = 0.6 , Wald t = 0.87 , Wald p = 0.38 ; log likelihood difference = 2.4 , df = 1 , Chi Squared = 4.85 , p = 0.028).

There was a significant interaction between shape and condition (beta = -0.061 , std.err = 0.83 , Wald t = -0.074 , Wald p = 0.94 ; log likelihood difference = 6 , df = 1 , Chi Squared = 12.04 , p = 0.00052).

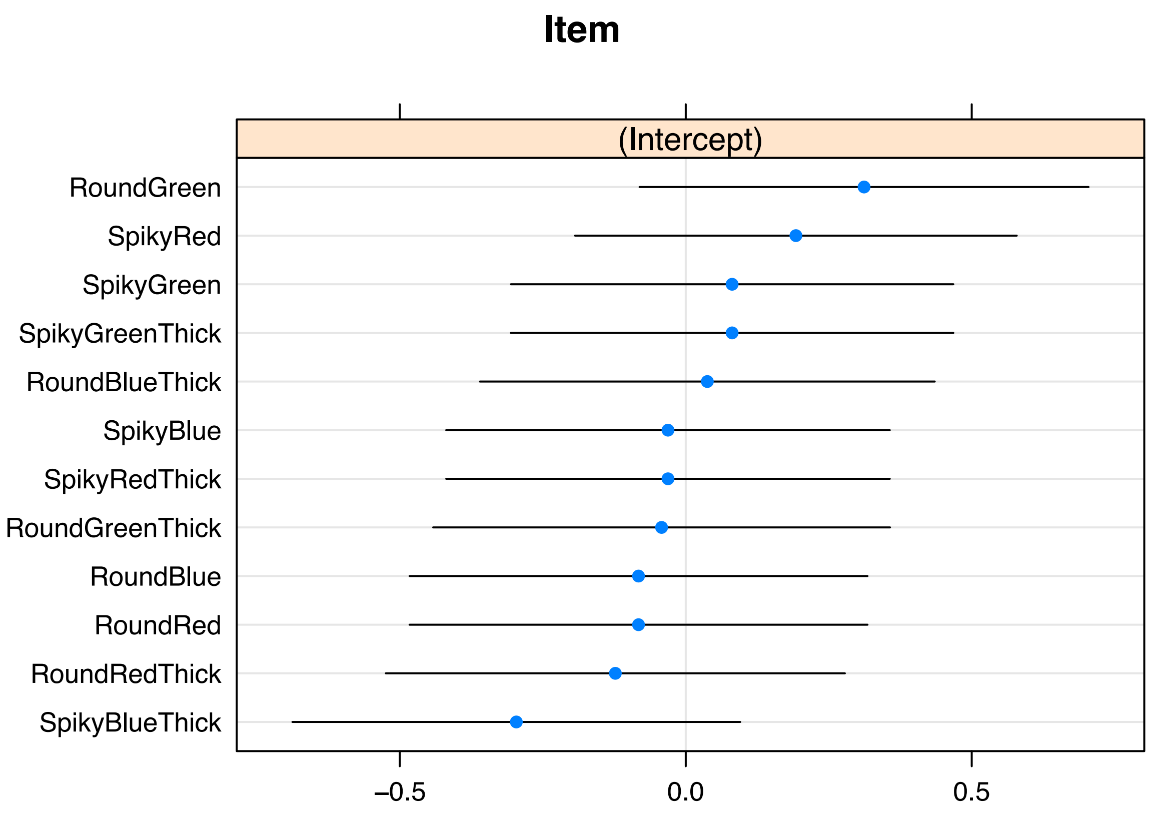
There was no significant interaction between shape and generation (beta = -0.17 , std.err = 0.15 , Wald t = -1.1 , Wald p = 0.26 ; log likelihood difference = 0.0057 , df = 1 , Chi Squared = 0.01 , p = 0.91).

There was a marginal three-way interaction between shape, condition and generation (beta = 0.39 , std.err = 0.22 , Wald t = 1.8 , Wald p = 0.074 ; log likelihood difference = 1.6 , df = 1 , Chi Squared = 3.2 , p = 0.073).

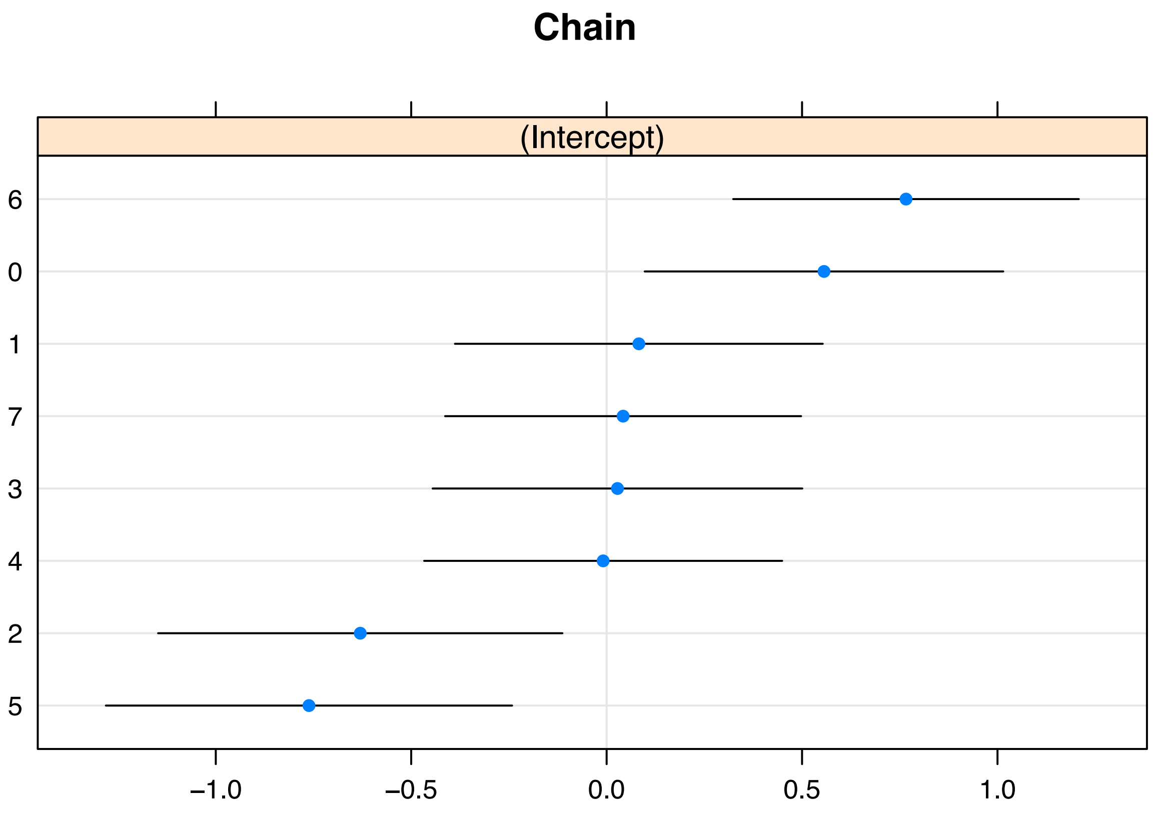
Plot random effects of final model

```
dotplot(ranef(mb7,  condVar=T))
```

```
## $Item
```

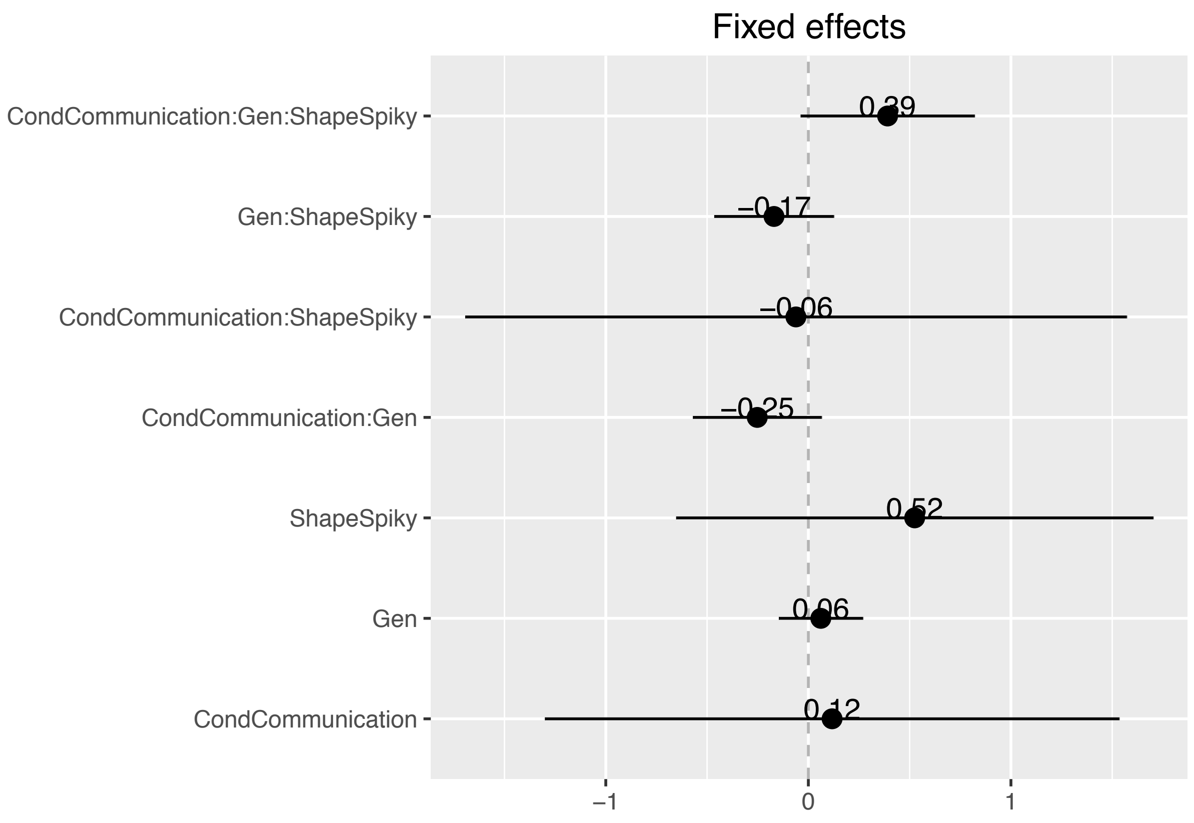


```
##
## $Chain
```



Plot fixed effects with standard error from final model.

```r
sjp.lmer(mb7, type='fe', geom.colors=c(1,1))
```

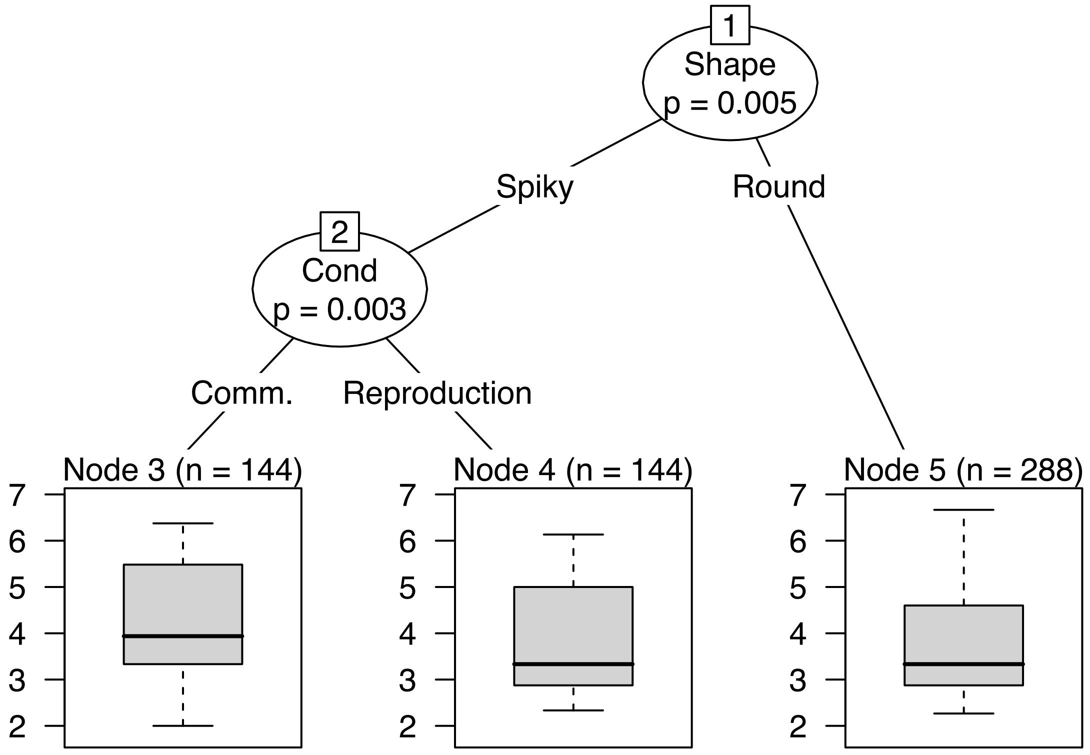


Binary tree analysis

We use a binary decision tree to predict spikiness ratings by condition, generation, item shape, item colour and item border type.

The results agree with those above, namely that the main effects are for shape, but spiky meanings are rated as more spiky in the communication condition

```r
finalLangs2 = finalLangs
finalLangs2$Shape = factor(finalLangs2$Shape)
finalLangs2$Colour = factor(finalLangs2$Colour)
finalLangs2$Border = factor(finalLangs2$Border)
finalLangs2$Cond = factor(finalLangs2$Cond, labels = c("Reproduction",'Comm.'))

cx = ctree(RatedSpikiness~Cond+Gen+Shape+Colour+Border, data=finalLangs2)
plot(cx)
```



Iconicity of innovations

Load data

Note that the column *Human* in the data indicates whether the signal was sent by a human. This is always the case in the communication condition, but only true for half of the trials in the reproduction condition. In the reproduction condition, when *Human* is `FALSE`, the human participant is guessing meaning from the signal sent by the program.

```
datax = read.csv("../results/IncreaseInIconicity.csv", stringsAsFactors = F)
alldatx = read.csv("../results/AllTrialData.csv", stringsAsFactors = F)
```

Number of innovations in each conditon, by whether the innovation was an unseen word (versus a change in mapping):

```
tx = table(datax[datax$Human,]$innovation.mutation,
           datax[datax$Human,]$condition)
tx
```

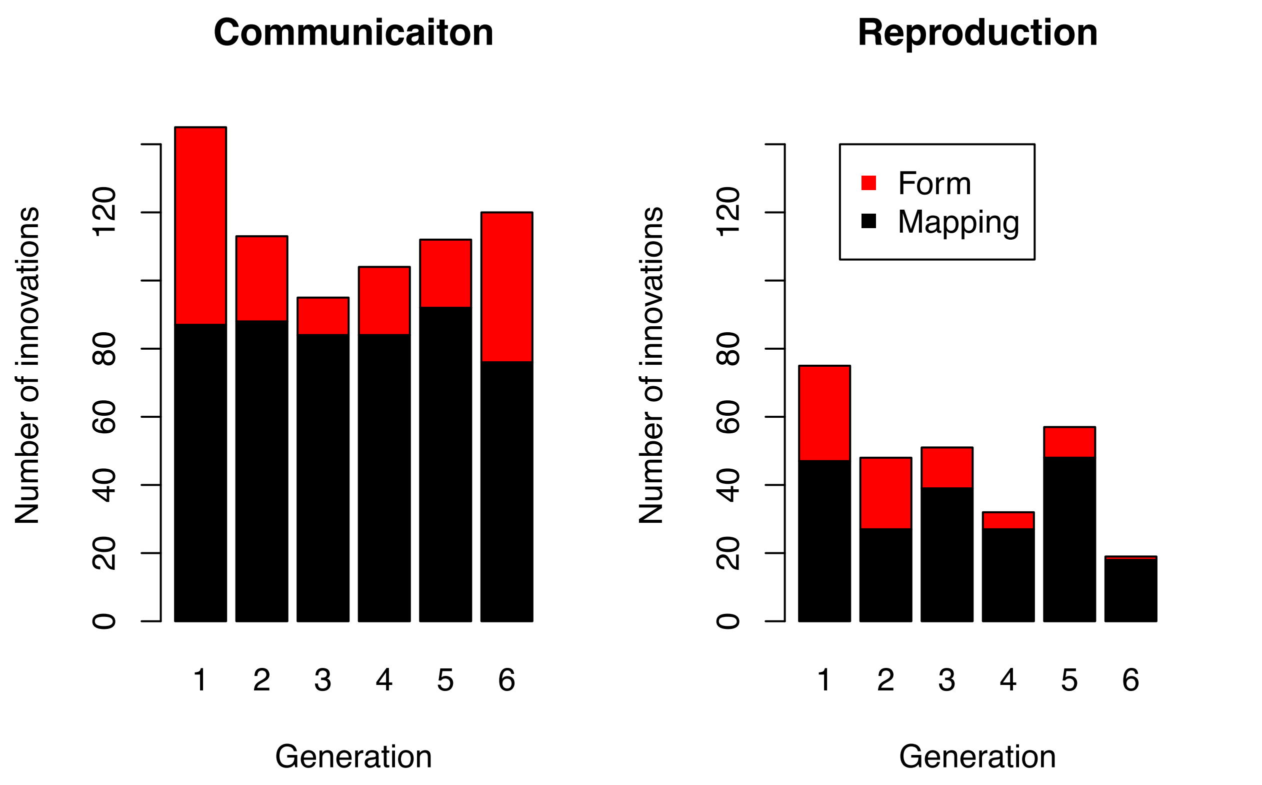
```
##
##         Comm Learn
##   FALSE  511   206
##   TRUE   178    76
```

```
tx/rep(colSums(tx),each=2)
```

```
##
##              Comm     Learn
##   FALSE 0.7416546 0.7304965
##   TRUE  0.2583454 0.2695035
```

The number of innovations in each condition by generation. The number of innovations declines in the learning condition:

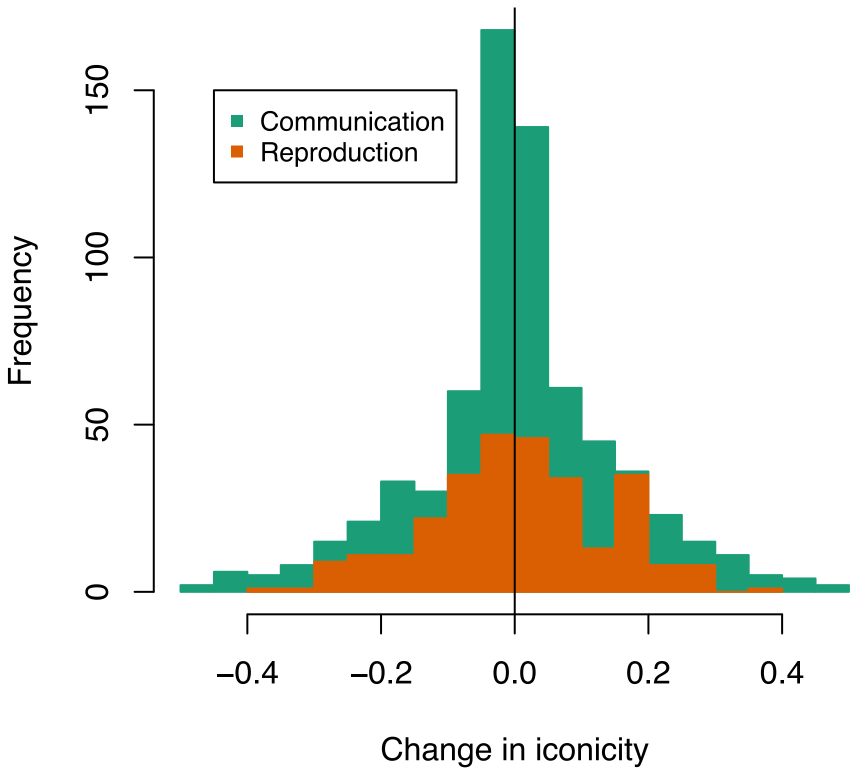
```
par(mfrow=c(1,2))
barplot((table(
  datax[datax$condition=="Comm",]$innovation.mutation,
  datax[datax$condition=="Comm",]$gen)), beside = F,
  ylab="Number of innovations"
  ,ylim=c(0,150),
  col=1:2,
  main="Communicaiton",
  xlab="Generation")

lx = datax$condition=="Learn" & datax$Human

barplot((table(
  datax[lx,]$innovation.mutation,
  datax[lx,]$gen)), beside = F,
  ylab="Number of innovations"
  ,ylim=c(0,150),
  col=1:2,
  main="Reproduction",
  xlab="Generation")

legend(1,140,legend=c("Form","Mapping"),
  col=2:1, pch=15)
```



Distribution of iconicity in innovations

Below is the distribution of how the innovation increases the iconicity of the mapping compared to the word that it replaced. The increases are small compared to the full Likert scale (1-7), and centered around zero.

```
# innovations produced by participants in the communication condition
comm.innovation.iconicity.dist =
  datax[datax$condition=='Comm' & datax$Human,]$increaseIconicity
# innovations produced by the human in the reproduction condition
learn.innovation.iconicity.dist =
  datax[datax$condition=='Learn' & datax$Human,]$increaseIconicity

cols = c('#1b9e77','#d95f02')

hist(comm.innovation.iconicity.dist, col=cols[1],
     breaks=14,
     border = cols[1],
     main='',
     xlab="Change in iconicity")
hist(learn.innovation.iconicity.dist,
     add=T,
     col=cols[2],
     breaks=14,
     border = cols[2])
abline(v=0)
legend(-0.45,150, legend = c("Communication","Reproduction"), col=cols, pch=15, cex=0.8)
```



Are the distributions biased? The curve for the learning condition looks like it has a bump on the right. Test the symmetry with a Wilcox signed rank test and the MGG test (Miao, Gel & Gastwirth, 2006, see `symmetry.test` function in `lawstat` package).

```
wilcox.test(comm.innovation.iconicity.dist)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  comm.innovation.iconicity.dist
## V = 117360, p-value = 0.3562
## alternative hypothesis: true location is not equal to 0
```

```
wilcox.test(learn.innovation.iconicity.dist)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  learn.innovation.iconicity.dist
## V = 20314, p-value = 0.6351
## alternative hypothesis: true location is not equal to 0
```

```
symmetry.test(comm.innovation.iconicity.dist)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth
##  (2006)
##
## data:  comm.innovation.iconicity.dist
## Test statistic = 1.1233, p-value = 0.328
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                  118
```

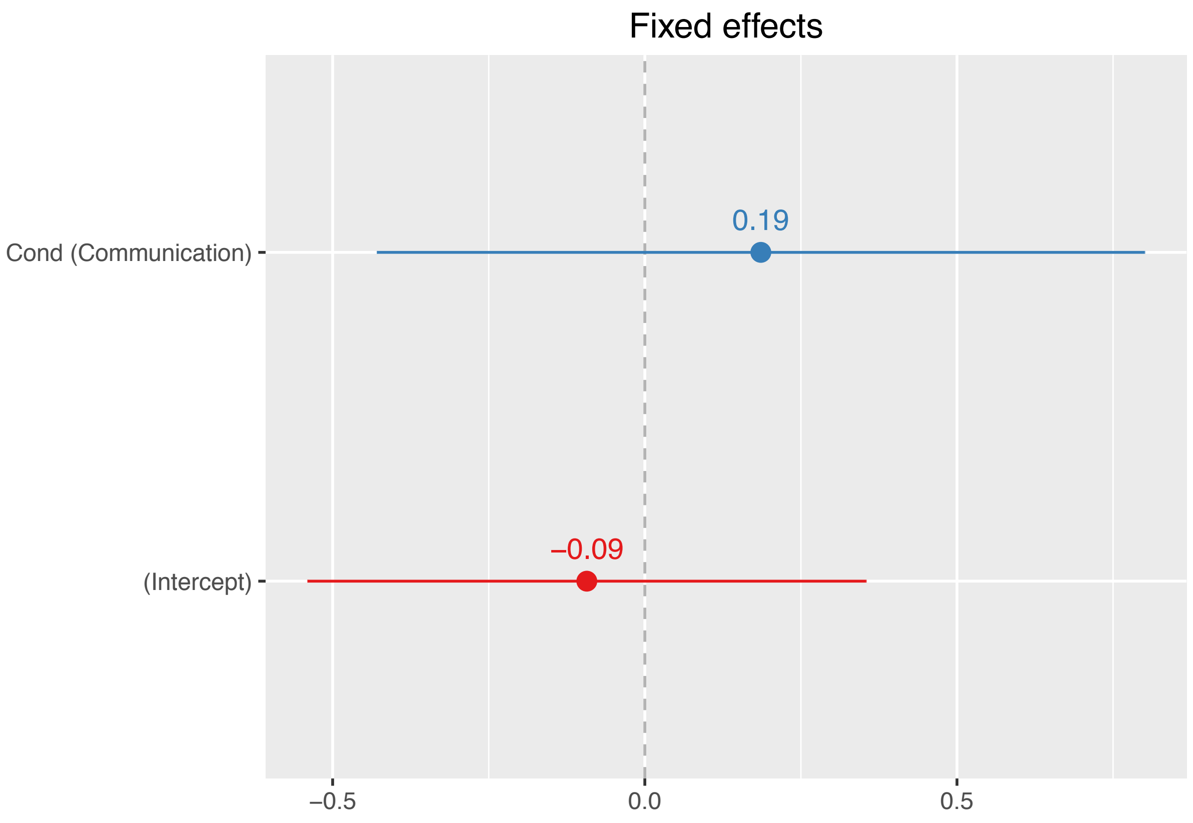
```
symmetry.test(learn.innovation.iconicity.dist)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth
##  (2006)
##
## data:  learn.innovation.iconicity.dist
## Test statistic = 0.31437, p-value = 0.698
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                   54
```

The tests show that the distributions are not different from zero and are not asymmetric. In other words, innovations are randomly distributed. A mixed effects model also shows that the intercept is not significantly different from zero:

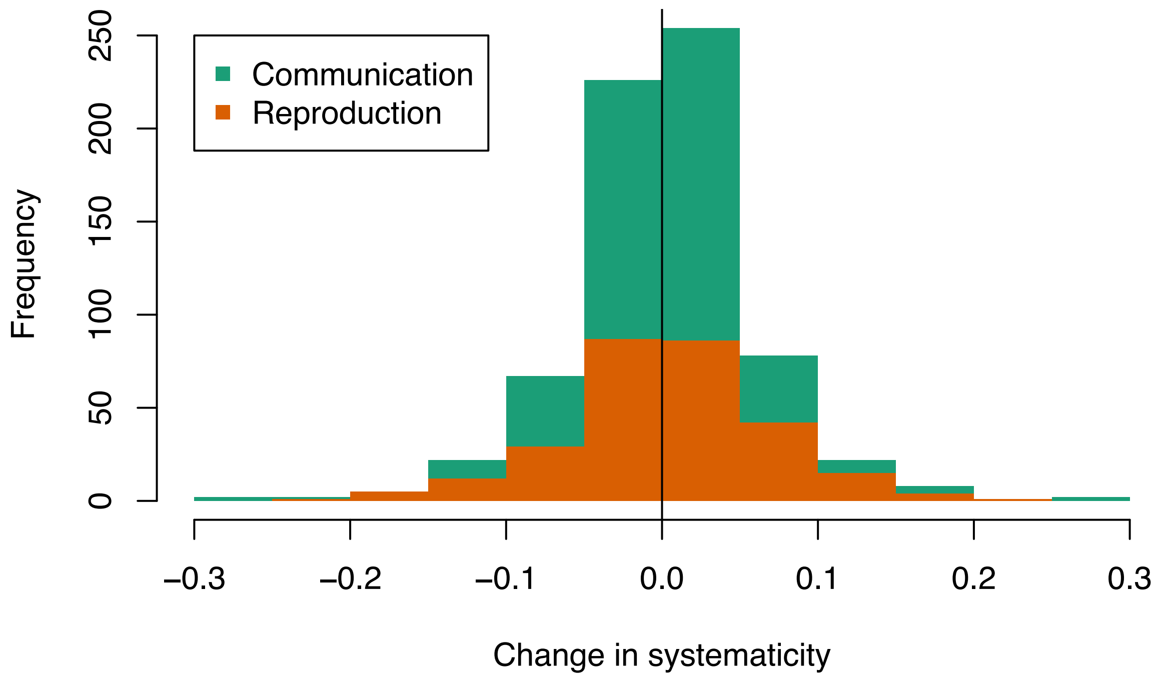
```
m0 = lmer(increaseIconicity ~ condition*inFinalLang + (1|chain) + (1|gen) + (1|meaning),
          data=datax[datax$Human,])
sjp.lmer(m1,'fe',show.intercept=T)
```

```
## Computing p-values via Kenward-Roger approximation. Use `p.kr = FALSE` if computation takes too long

## Warning in deviance.merMod(object, ...): deviance() is deprecated for REML
## fits; use REMLcrit for the REML criterion or deviance(.,REML=FALSE) for
## deviance calculated at the REML fit
```



Interestingly, the results look similar for the change in systematicity.

```
# innovations produced by participants in the communication condition
comm.innovation.sys.dist =
  datax[datax$condition=='Comm' & datax$Human,]$systematicity.increase
# innovations produced by the human in the reproduction condition
learn.innovation.sys.dist =
  datax[datax$condition=='Learn' & datax$Human,]$systematicity.increase

cols = c('#1b9e77','#d95f02')

hist(comm.innovation.sys.dist, col=cols[1],
     breaks=14,
     border = NA,
     main='',
     xlab="Change in systematicity")
hist(learn.innovation.sys.dist,
     add=T,
     col=cols[2],
     breaks=14,
     border = NA)
abline(v=0)
legend(-0.3,250, legend = c("Communication","Reproduction"), col=cols, pch=15)
```



```
wilcox.test(comm.innovation.sys.dist)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  comm.innovation.sys.dist
## V = 125380, p-value = 0.1655
## alternative hypothesis: true location is not equal to 0
```

```
wilcox.test(learn.innovation.sys.dist)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  learn.innovation.sys.dist
## V = 21640, p-value = 0.1178
## alternative hypothesis: true location is not equal to 0
```

```
symmetry.test(comm.innovation.sys.dist)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth
##  (2006)
##
## data:  comm.innovation.sys.dist
## Test statistic = 0.22877, p-value = 0.864
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                 430
```

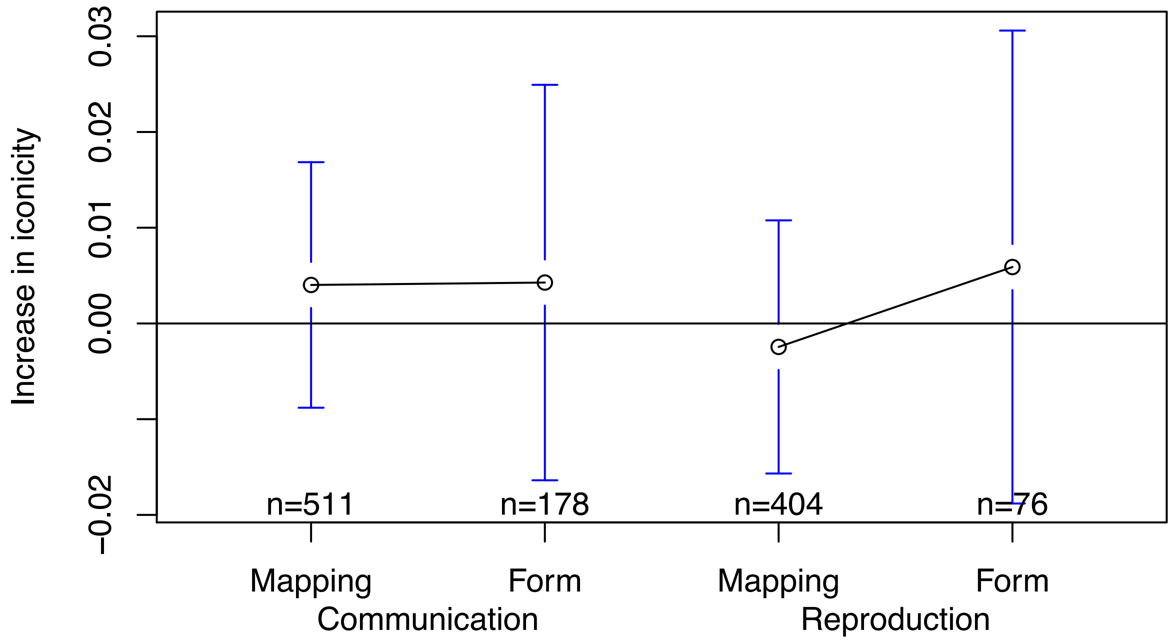
```
symmetry.test(learn.innovation.sys.dist)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth
##  (2006)
##
## data:  learn.innovation.sys.dist
## Test statistic = 0.99137, p-value = 0.324
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                 176
```

Innovations in form versus innovations in mapping

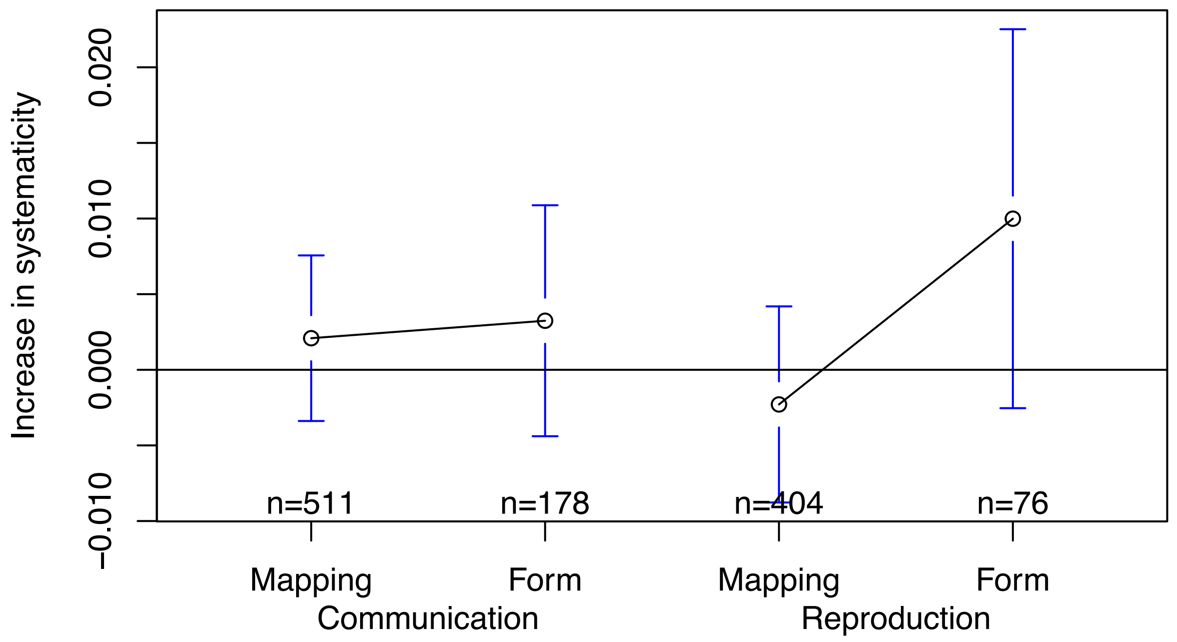
There appear to be no differences with regards to iconiciy according to the category of innovation:

```
plotmeans(increaseIconicity ~ paste(condition,innovation.mutation),
          data=datax,
          legends = c("Mapping","Form","Mapping",'Form'),
          xlab='',
          ylab='Increase in iconicity',
          connect = list(1:2,3:4)
          )
axis(1,at=c(1.5,3.5),c("Communication","Reproduction"),line=1, tick=F)
abline(h=0)
```



However, there seems like a weak bias for form innovations in the learning condition to lead to an increase in systematicity. Future work could explore these implications.

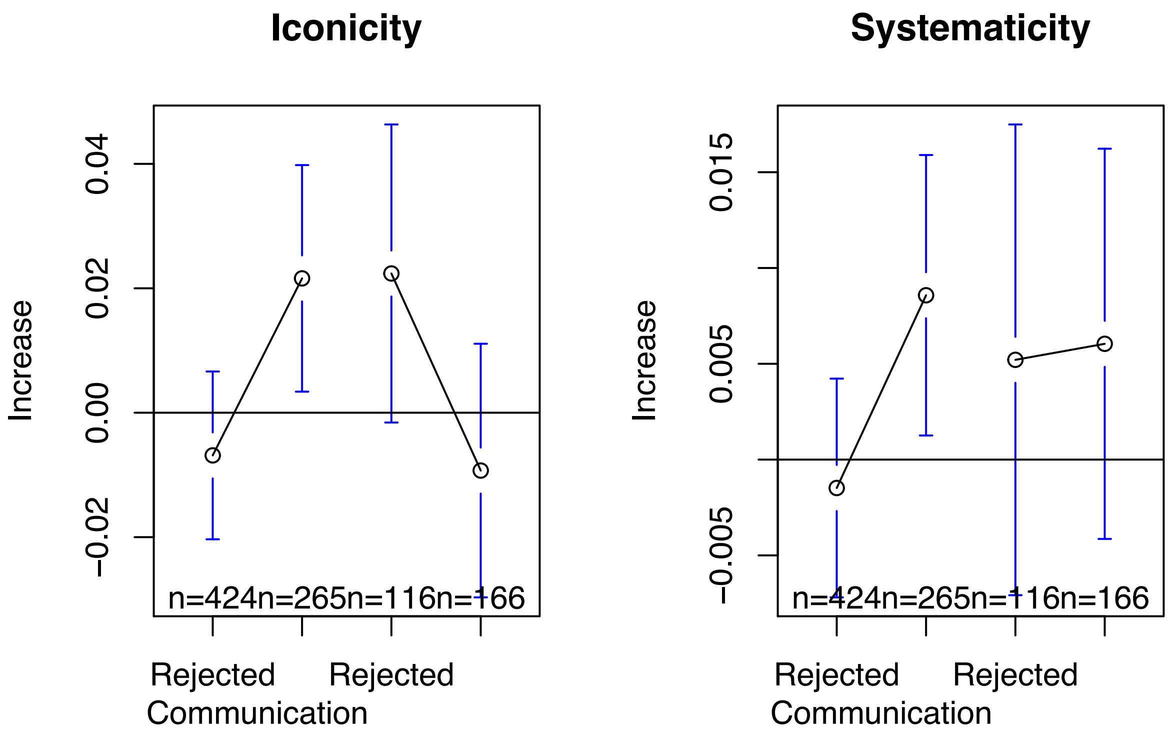
```
plotmeans(systematicity.increase ~ paste(condition,innovation.mutation),
          data=datax,
          legends = c("Mapping","Form","Mapping",'Form'),
          xlab='',
          ylab='Increase in systematicity',
          connect = list(1:2,3:4)
          )
axis(1,at=c(1.5,3.5),c("Communication","Reproduction"),line=1, tick=F)
abline(h=0)
```


Increase in systematicity
0.020
0.010
0.000
−0.010
n=511
n=178
n=404
n=76
Mapping
Form
Mapping
Form
Communication
Reproduction

Increase in iconicity by survival

In the communication condition, innovations that survive tend to increase both iconicity and systematicity. However, in the learning condition, the innovations only contribute to systematicity but not to iconicity.

```
par(mfrow=c(1,2))
plotmeans(increaseIconicity ~ paste(condition,inFinalLang), data=datax[datax$Human,], connect = list(1:
title(main="Iconicity")
axis(1,at=c(1.5,3.5),c("Communication","Reproduction"),line=1, tick=F)
abline(h=0)
plotmeans(systematicity.increase ~ paste(condition, inFinalLang), data = datax[datax$Human,], connect =
title(main="Systematicity")
axis(1,at=c(1.5,3.5),c("Communication","Reproduction"),line=1, tick=F)
abline(h=0)
```



Build a mixed effects model predicting the increase in iconicity with random effects for chain, generation and item:

```
m0= lmer(increaseIconicity ~ 1 +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m1= lmer(increaseIconicity ~ condition +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m2= lmer(increaseIconicity ~ condition+inFinalLang +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m3= lmer(increaseIconicity ~ condition*inFinalLang +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
```

Model comparison test:

```
anova(m0,m1,m2,m3)
```

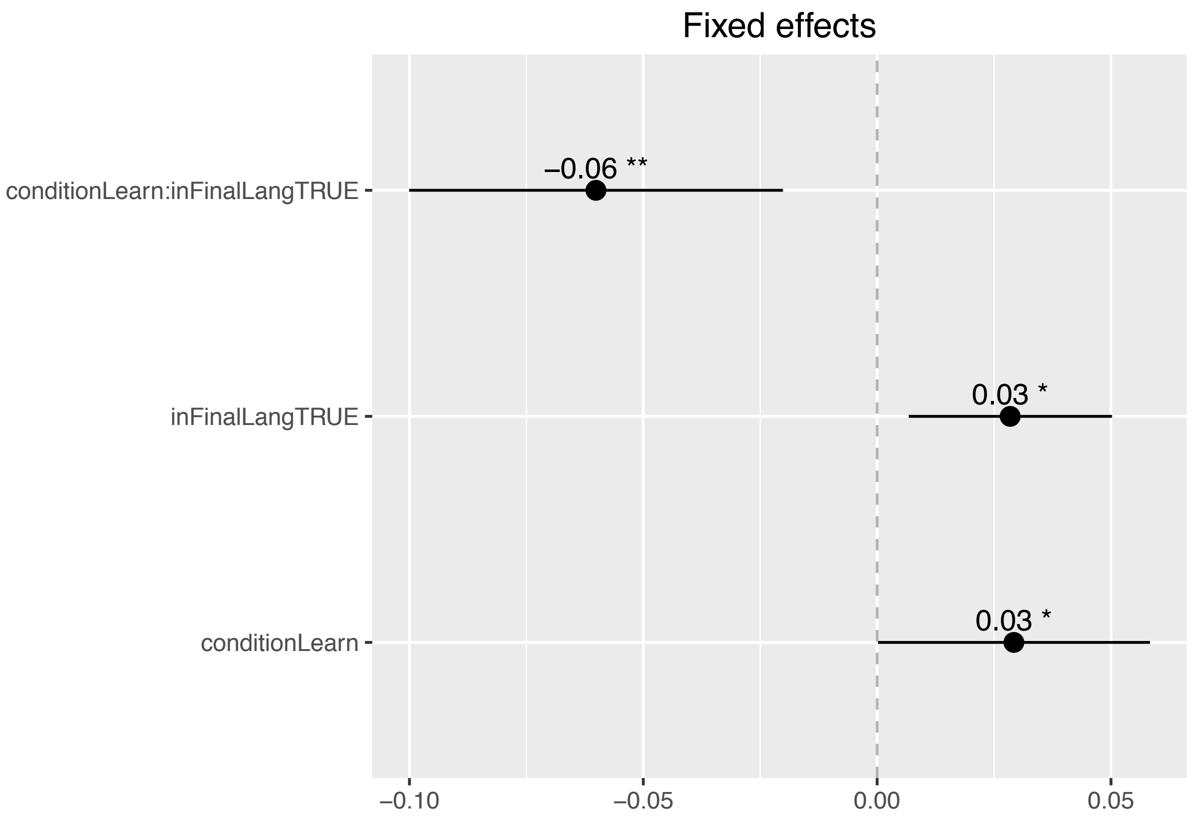
```
## refitting model(s) with ML (instead of REML)

## Data: datax[datax$Human, ]
## Models:
## m0: increaseIconicity ~ 1 + (1 | chain) + (1 | meaning) + (1 | gen)
## m1: increaseIconicity ~ condition + (1 | chain) + (1 | meaning) +
## m1:     (1 | gen)
## m2: increaseIconicity ~ condition + inFinalLang + (1 | chain) + (1 |
## m2:     meaning) + (1 | gen)
## m3: increaseIconicity ~ condition * inFinalLang + (1 | chain) + (1 |
## m3:     meaning) + (1 | gen)
##    Df     AIC      BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0  5 -1029.3 -1004.91 519.65  -1039.3
## m1  6 -1027.3  -998.03 519.65  -1039.3 0.0012      1   0.971891
## m2  7 -1026.6  -992.47 520.31  -1040.6 1.3218      1   0.250265
## m3  8 -1033.3  -994.31 524.67  -1049.3 8.7170      1   0.003153 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There was a significant interaction between condition and survival to transmission (beta = -0.06 , std.err = 0.02 , Wald t = -3 ; log likelihood difference = 4.4 , df = 1 , Chi Squared = 8.72 , p = 0.0032).

Plot the fixed effects from the final model:

```
sjp.lmer(m3,'fe', geom.colors = c(1,1), p.kr=F)
```

```
## Computing p-values via Wald-statistics approximation (treating t as Wald z).
```



Increase in systematicity by survival

Build a mixed effects model predicting the increase in systematicity with random effects for chain, generation and item:

```
m0= lmer(systematicity.increase ~ 1 +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m1= lmer(systematicity.increase ~ condition +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m2= lmer(systematicity.increase ~ condition+inFinalLang +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
m3= lmer(systematicity.increase ~ condition*inFinalLang +(1|chain) + (1|meaning) + (1|gen),
         data=datax[datax$Human,])
```

Model comparison test:

```
anova(m0,m1,m2,m3)
```

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

## Warning in optwrap(optimizer, devfun, x@theta, lower = x@lower, calc.derivs
## = TRUE, : convergence code 3 from bobyqa: bobyqa -- a trust region step
## failed to reduce q

## Data: datax[datax$Human, ]
## Models:
## m0: systematicity.increase ~ 1 + (1 | chain) + (1 | meaning) + (1 |
## m0:     gen)
## m1: systematicity.increase ~ condition + (1 | chain) + (1 | meaning) +
## m1:     (1 | gen)
## m2: systematicity.increase ~ condition + inFinalLang + (1 | chain) +
## m2:     (1 | meaning) + (1 | gen)
## m3: systematicity.increase ~ condition * inFinalLang + (1 | chain) +
## m3:     (1 | meaning) + (1 | gen)
##    Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0  5 -2632.7 -2608.3 1321.3  -2642.7
## m1  6 -2631.2 -2602.0 1321.6  -2643.2 0.5701      1     0.4502
## m2  7 -2632.5 -2598.3 1323.2  -2646.5 3.2414      1     0.0718 .
## m3  8 -2631.5 -2592.5 1323.8  -2647.5 1.0682      1     0.3014
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

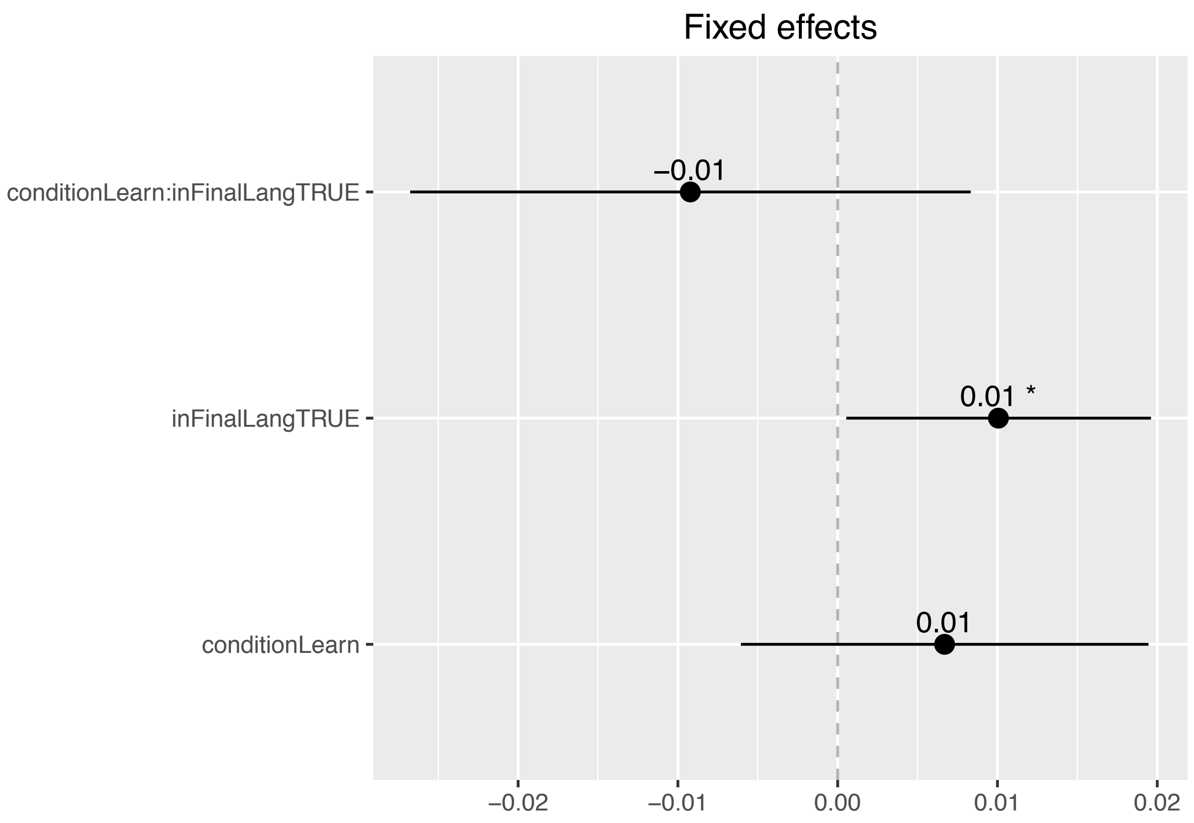
There was no significant main effect of condition (beta = -0.0092 , std.err = 0.0089 , Wald t = -1 ; log likelihood difference = 0.53 , df = 1 , Chi Squared = 1.07 , p = 0.3).

There was no significant interaction between condition and survival to transmission (beta = -0.0092 , std.err = 0.0089 , Wald t = -1 ; log likelihood difference = 0.53 , df = 1 , Chi Squared = 1.07 , p = 0.3).

Plot the fixed effects from the final model:

```
sjp.lmer(m3,'fe', geom.colors = c(1,1), p.kr = F)
```

```
## Computing p-values via Wald-statistics approximation (treating t as Wald z).
```

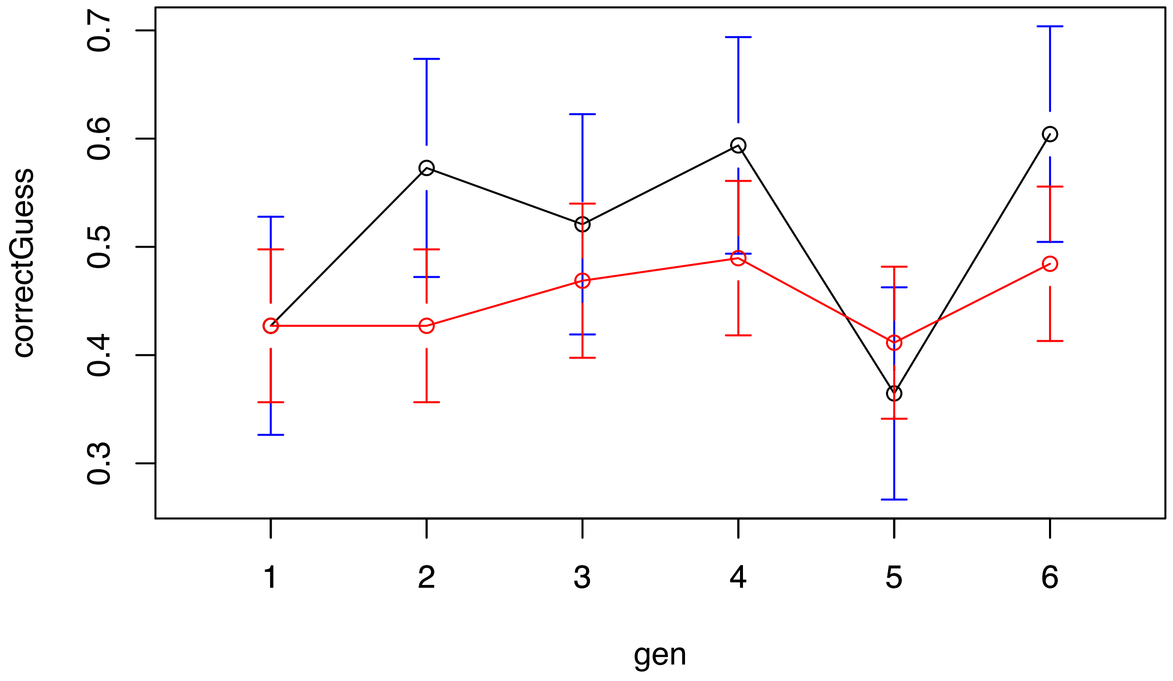

Fixed effects
conditionLearn:inFinalLangTRUE
inFinalLangTRUE
conditionLearn
−0.01
0.01 *
0.01
−0.02
−0.01
0.00
0.01
0.02

Accuracy

The mean proportion of correct guesses in the communication condition was 45.14%. The mean proportion of correct guesses by the human participant in the learning condition was 51.39%.

Plot the correct guesses by generation (means and 95% confidence intervals):

```
plotmeans(correctGuess~gen,alldatx[alldatx$condition=='Learn' & !alldatx$Human,], n.label = F)
plotmeans(correctGuess~gen,alldatx[alldatx$condition=='Comm',],add=T,col=2,barcol=2, n.label = F)
```



Mixed effects model

Binomial mixed effects model, with random effects for chain, target item. Test whether there are differences between conditions.

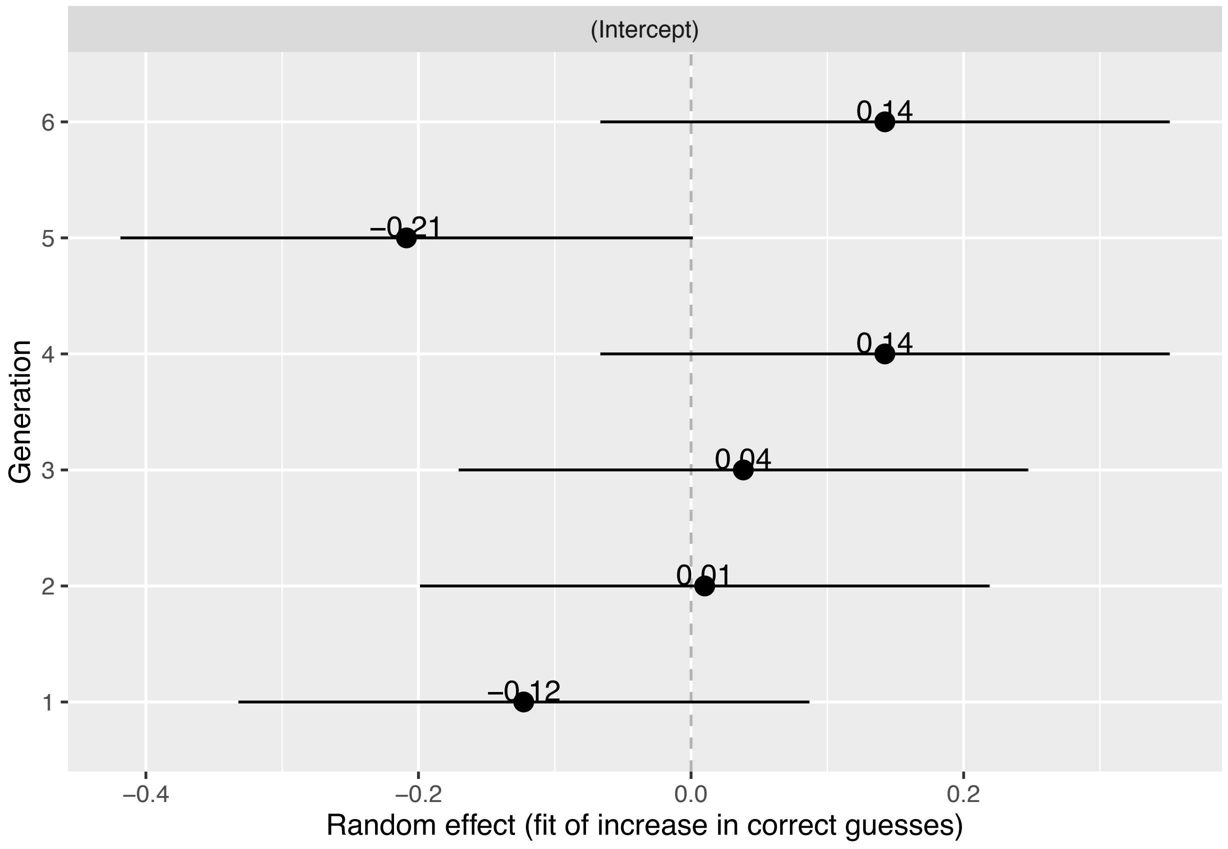
```
ctrl = glmerControl(optCtrl = list(maxfun=50000))
# we want to exclude trials where the computer is guessing meanings
# from the participant's signals in the reproduction condition
m0 = glmer(correctGuess ~ 1 + (1|chain) + (1|target.meaning) ,
    data=alldatx[alldatx$condition=='Comm' | (!alldatx$Human),],
    family = binomial, control= ctrl)

m1 = glmer(correctGuess ~ 1 + (1|chain) + (1|target.meaning) + (1|gen),
    data=alldatx[alldatx$condition=='Comm' | (!alldatx$Human),],
    family = binomial, control= ctrl)
m2 = glmer(correctGuess ~ condition + (1|chain) + (1|target.meaning)+ (1|gen),
    data=alldatx[alldatx$condition=='Comm' | (!alldatx$Human),],
    family = binomial, control= ctrl)
anova(m0,m1,m2)
```

```
## Data: alldatx[alldatx$condition == "Comm" | (!alldatx$Human), ]
## Models:
## m0: correctGuess ~ 1 + (1 | chain) + (1 | target.meaning)
## m1: correctGuess ~ 1 + (1 | chain) + (1 | target.meaning) + (1 |
## m1:     gen)
## m2: correctGuess ~ condition + (1 | chain) + (1 | target.meaning) +
## m2:     (1 | gen)
##    Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0  3 2375.7 2392.0 -1184.8   2369.7
## m1  4 2372.8 2394.6 -1182.4   2364.8 4.9062      1    0.02676 *
## m2  5 2373.1 2400.4 -1181.5   2363.1 1.6748      1    0.19562
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There was no significant main effect of condition (beta = 0.26 , std.err = 0.19 , Wald t = 1.4 , Wald p = 0.17 ; log likelihood difference = 0.84 , df = 1 , Chi Squared = 1.67 , p = 0.2).

There was a significant difference between generations (log likelihood difference = 2.5 , df = 1 , Chi Squared = 4.91 , p = 0.027). There is a weak trend for the proportion of correct guesses to increase by generation, as shown by the estimates for the random effects for generation:

```r
x = sjp.lmer(m2, sort.est='sort.all', prnt.plot=F,
             geom.colors=c(1,1),
             facet.grid = T)
x$plot.list[[3]] +
  xlab("Generation") +
  ylab("Random effect (fit of increase in correct guesses)")
```

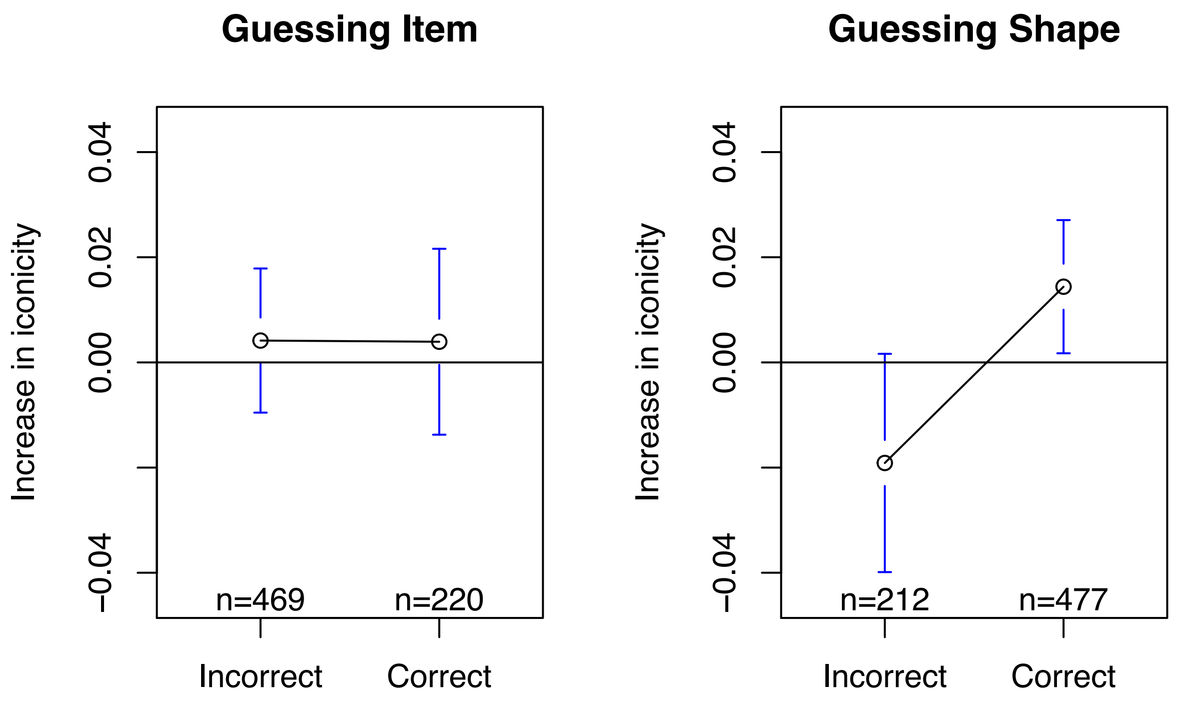


Iconicity and accuracy for innovations

Innovations are either more or less iconic than the words they replace. There is no difference in how accurate the guesses are in terms of choosing the right item (see below, left), but the innovation tends to be more iconic when the shape of a meaning is guessed correctly (spiky or round). That is, the iconicity is helping participants guess the shape of a target meaning correctly.

Note that this analysis only makes sense for the communication condition.

```r
par(mfrow=c(1,2))
ylimx = c(-0.045,0.045)
plotmeans(increaseIconicity ~ paste(condition, correctGuess),
          data = datax[datax$Human & datax$condition=="Comm",],
          ylim=ylimx, legends = c("Incorrect","Correct"),
          xlab='',
          ylab="Increase in iconicity")
title("Guessing Item")
abline(h=0)
plotmeans(increaseIconicity ~ paste(condition, correctSpikiness),
          data = datax[datax$Human& datax$condition=="Comm",],
          ylim=ylimx,legends = c("Incorrect","Correct"),
          xlab='',
          ylab="Increase in iconicity")
title("Guessing Shape")
abline(h=0)
```



Mixed effects model for accuracy and iconicity

A mixed effects model predicting the increase in iconicity by whether the reciever selected the correct target item, and by whether the reciever selected an item which matched the target in the shape dimension, with random effects for chain, generation and item. Note that it would make more intuitive sense to predict accuracy by increase in iconicity, but this way we can compare the effects of item accuracy versus shape accuracy.

```
m0 = lmer(increaseIconicity ~ 1 + (1|chain) + (1|gen) + (1|meaning),
          data=datax[datax$condition=="Comm",])

m1 = lmer(increaseIconicity ~ correctGuess + (1|chain) + (1|gen)+ (1|meaning),
          data=datax[datax$condition=="Comm",])

m2 = lmer(increaseIconicity ~ correctGuess + correctSpikiness + (1|chain) + (1|gen)+ (1|meaning),
          data=datax[datax$condition=="Comm",])

anova(m0,m1,m2)
```

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

## Data: datax[datax$condition == "Comm", ]
## Models:
## m0: increaseIconicity ~ 1 + (1 | chain) + (1 | gen) + (1 | meaning)
## m1: increaseIconicity ~ correctGuess + (1 | chain) + (1 | gen) +
## m1:     (1 | meaning)
## m2: increaseIconicity ~ correctGuess + correctSpikiness + (1 | chain) +
## m2:     (1 | gen) + (1 | meaning)
##    Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0  5 -692.34 -669.66 351.17  -702.34
## m1  6 -690.34 -663.13 351.17  -702.34 0.0004      1   0.984852
## m2  7 -698.34 -666.59 356.17  -712.34 9.9963      1   0.001569 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(m2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula:
## increaseIconicity ~ correctGuess + correctSpikiness + (1 | chain) +
##     (1 | gen) + (1 | meaning)
##    Data: datax[datax$condition == "Comm", ]
##
## REML criterion at convergence: -689.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3006 -0.3909 -0.0257  0.4642  3.2271
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  meaning  (Intercept) 0.00000  0.0000
##  gen      (Intercept) 0.00000  0.0000
##  chain    (Intercept) 0.00000  0.0000
##  Residual             0.02091  0.1446
## Number of obs: 689, groups:  meaning, 12; gen, 6; chain, 4
```

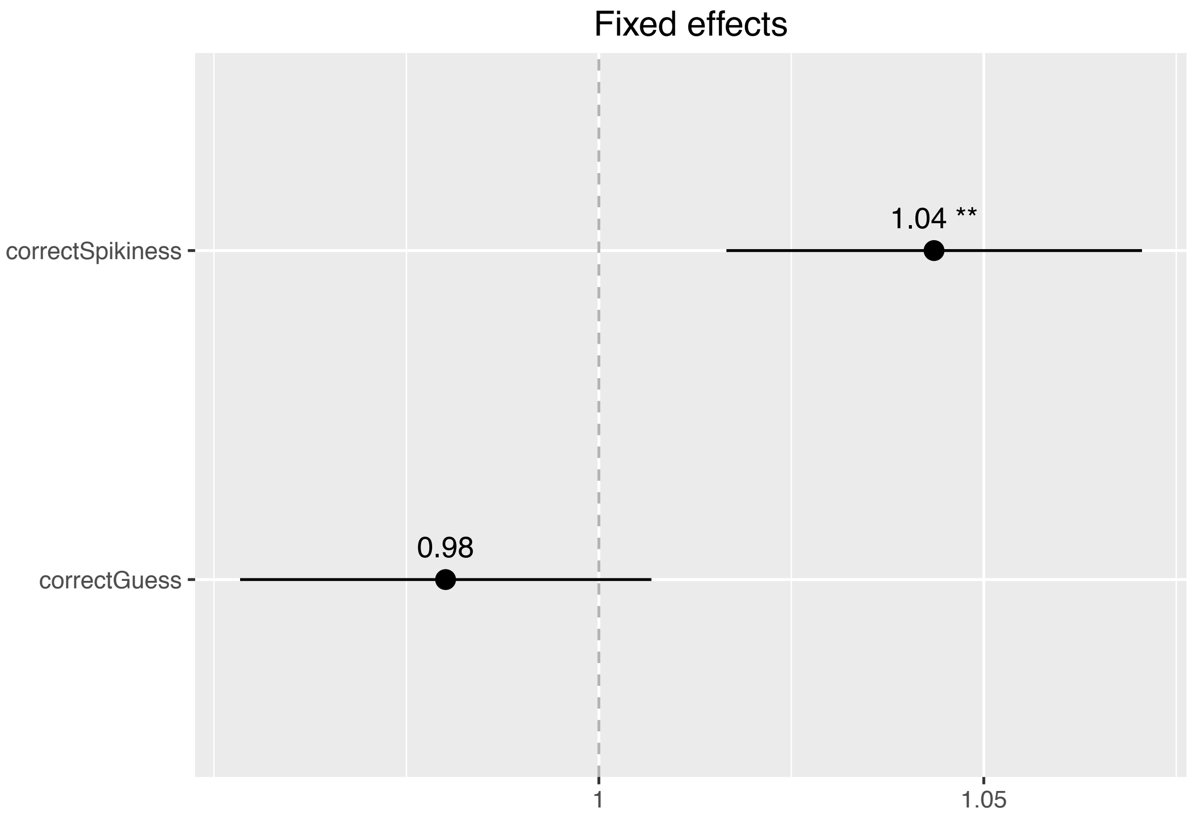
```
##
## Fixed effects:
##                      Estimate Std. Error t value
## (Intercept)         -0.019121   0.009932  -1.925
## correctGuessTRUE    -0.019429   0.013283  -1.463
## correctSpikinessTRUE  0.042482   0.013417   3.166
##
## Correlation of Fixed Effects:
##              (Intr) cGTRUE
## crrctGsTRUE  0.000
## crrctSpTRUE -0.740 -0.457
```

There was no significant main effect of guessing the item correctly (beta = -0.019 , std.err = 0.013 , Wald t = -1.5 ; log likelihood difference = 0.00018 , df = 1 , Chi Squared = 0 , p = 0.98).

There was a significant main effect of guessing the shape correctly (beta = 0.042 , std.err = 0.013 , Wald t = 3.2 ; log likelihood difference = 5 , df = 1 , Chi Squared = 10 , p = 0.0016).

Plot the fixed effects:

```
sjp.glmer(m2, type='fe', geom.colors=c(1,1) )
```



Note that the model is probably overfitted, since the random effects are singulative. But the effect is clear from the plot of the raw data.

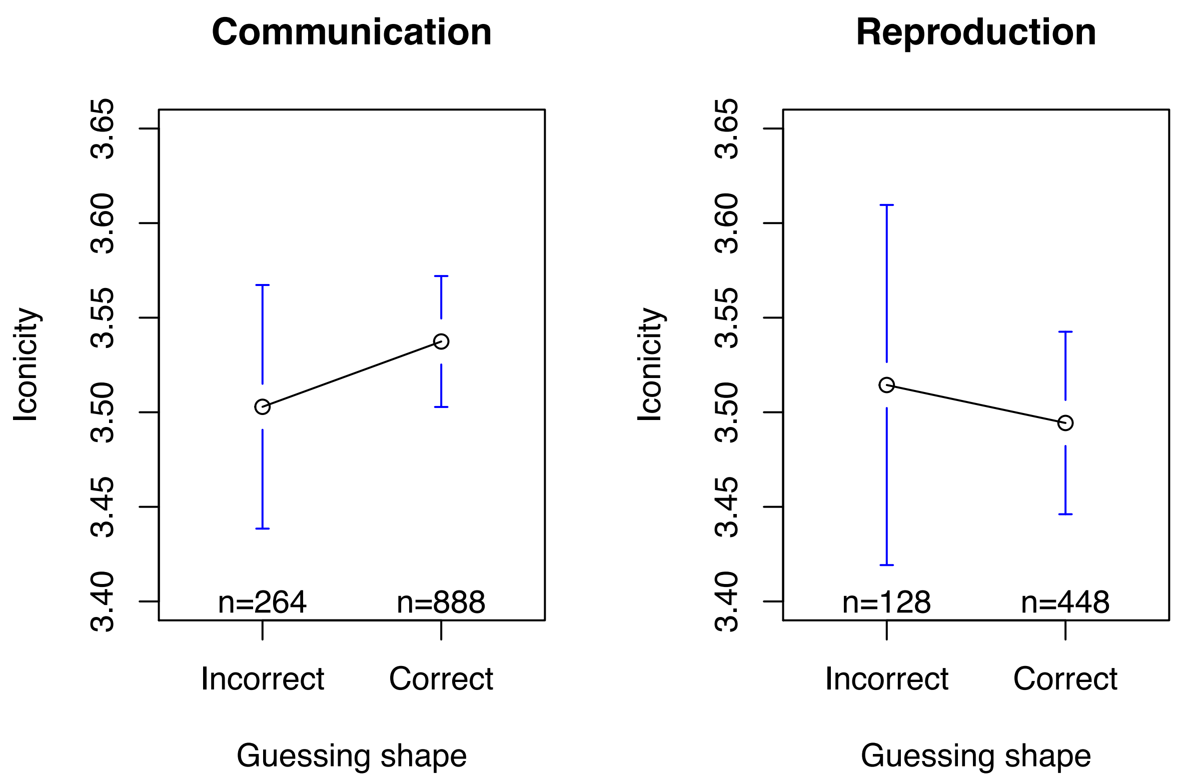
Iconicity and accuracy for whole data

Make a variable that indicates the iconicity of a word according to its shape. i.e. high if it aligns with shape, low if it does not. In other words, reverse the scale for round meanings.

```
alldatx$estimatedIconicity = alldatx$estimatedSpikinessRating

alldatx$estimatedIconicity[alldatx$target.meaning>5] =
  7 - alldatx$estimatedIconicity[alldatx$target.meaning>5]
```

Plot the raw data

```
par(mfrow=c(1,2))
ylimx= c(3.4,3.65)
plotmeans(estimatedIconicity~correctSpikiness,
          data=alldatx[alldatx$condition=='Comm',],
          ylim = ylimx,
          xlab="Guessing shape",
          ylab="Iconicity",
          legends = c("Incorrect","Correct"),
          main="Communication")

plotmeans(estimatedIconicity~correctSpikiness,
          data=alldatx[alldatx$condition=='Learn' &
                          (!alldatx$Human),],
          ylim = ylimx,
          xlab="Guessing shape",
          ylab="Iconicity",
          legends = c("Incorrect","Correct"),
          main="Reproduction")
```



Iconicity and systematicity

We would like to test whether there is a link between the increase in iconicity and the increase in systematicity for innovations.

Overall correlation between systematicity increase and iconicity increase:

```
cor.test(datax[datax$Human,]$increaseIconicity,datax[datax$Human,]$systematicity.increase)
```

```
##
##  Pearson's product-moment correlation
##
## data:  datax[datax$Human, ]$increaseIconicity and datax[datax$Human, ]$systematicity.increase
## t = 1.6028, df = 969, p-value = 0.1093
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.01152866  0.11396482
## sample estimates:
##        cor
## 0.05142107
```

For the communication condition:

```
cor.test(
  datax[datax$condition=="Comm",]$increaseIconicity,
  datax[datax$condition=="Comm",]$systematicity.increase)
```

```
##
##  Pearson's product-moment correlation
##
## data:  datax[datax$condition == "Comm", ]$increaseIconicity and datax[datax$condition == "Comm", ]$sy
## t = 2.5225, df = 687, p-value = 0.01188
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.02125801 0.16927945
## sample estimates:
##        cor
## 0.09579831
```

For the reproduction condition:

```
cor.test(
  datax[datax$condition=="Learn" & datax$Human,]$increaseIconicity,
  datax[datax$condition=="Learn" & datax$Human,]$systematicity.increase)
```

```
##
##  Pearson's product-moment correlation
##
## data:  datax[datax$condition == "Learn" & datax$Human, ]$increaseIconicity and datax[datax$condition
## t = -0.9347, df = 280, p-value = 0.3508
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.17145923  0.06143279
## sample estimates:
##        cor
## -0.0557718
```

Simple linear model:

```r
summary(lm(increaseIconicity~systematicity.increase*condition, data=datax[datax$Human,]))
```

```
##
## Call:
## lm(formula = increaseIconicity ~ systematicity.increase * condition,
##     data = datax[datax$Human, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51356 -0.06968 -0.00254  0.07290  0.46513
##
## Coefficients:
##                                         Estimate Std. Error t value
## (Intercept)                             0.003534   0.005393   0.655
## systematicity.increase                  0.231376   0.089539   2.584
## conditionLearn                          0.000833   0.010027   0.083
## systematicity.increase:conditionLearn -0.342597   0.155334  -2.206
##                                        Pr(>|t|)
## (Intercept)                             0.51240
## systematicity.increase                  0.00991 **
## conditionLearn                          0.93381
## systematicity.increase:conditionLearn  0.02765 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1414 on 967 degrees of freedom
## Multiple R-squared:  0.007642,   Adjusted R-squared:  0.004563
## F-statistic: 2.482 on 3 and 967 DF,  p-value: 0.0596
```

Significant main effect for systematicity increase and interaction.

Mixed effects model

Build a series of mixed effects models predicting the increase in iconicity with random effects for chain and item.

```r
# select data - either trails from the communication condition
#  or from the reproduction condition when the human was the speaker
datax.sel = datax[datax$Human,]

# Null model, predicting increase in iconicity by condition and generation
m0 = lmer(increaseIconicity~condition*gen + (1|chain) + (1|meaning),
          data = datax.sel)

# Add main effect of systematicity increase
m1 = lmer(increaseIconicity~systematicity.increase+(condition*gen) + (1|chain) + (1|meaning),
          data = datax.sel)

# Add interaction between Sys. and condition
m2 = lmer(increaseIconicity~systematicity.increase + (systematicity.increase:condition) +
            (condition*gen) + (1|chain) + (1|meaning),
          data = datax.sel)

# Add 3-way interaction
```

```
m3 = lmer(increaseIconicity~systematicity.increase*condition*gen +
            (1|chain) + (1|meaning),
          data = datax.sel)
```

Results

Test the model fit. Note that the model converges on signualtive random effect estimates.

```
anova(m0,m1,m2,m3)
```

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

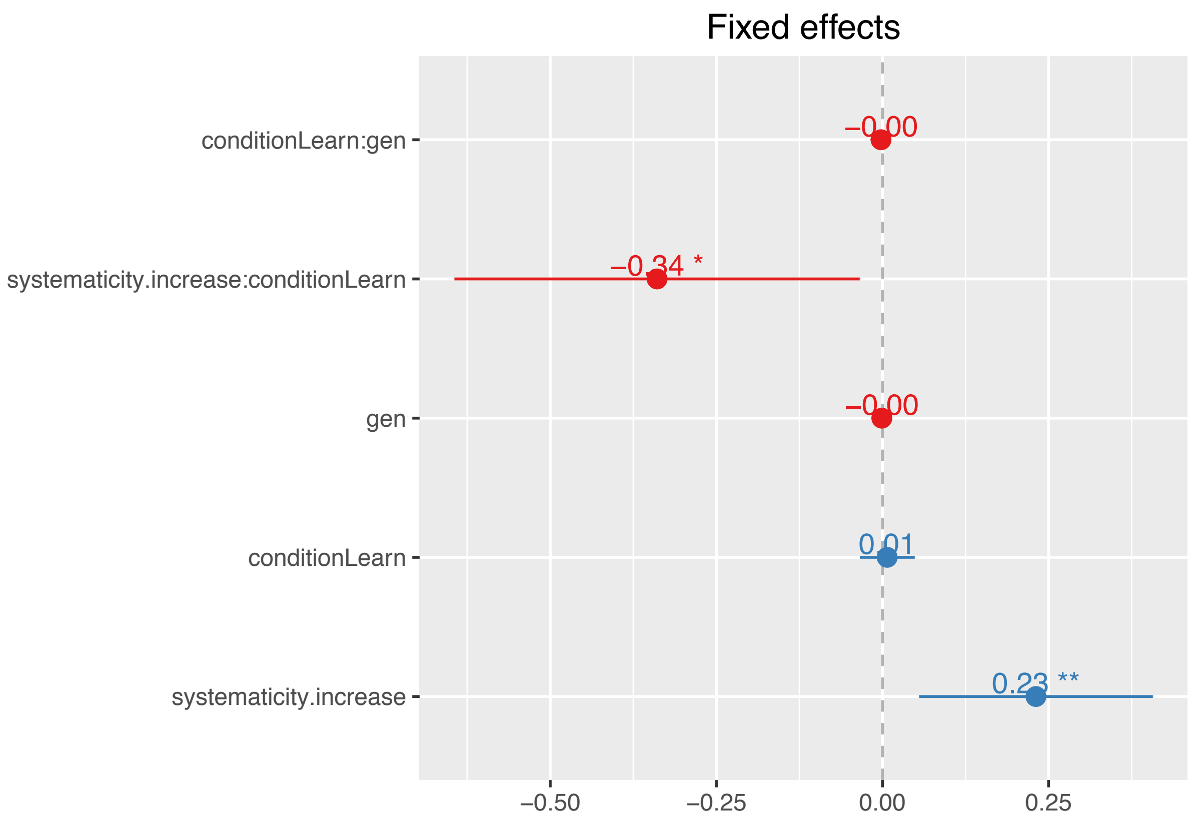
```
## Data: datax.sel
## Models:
## m0: increaseIconicity ~ condition * gen + (1 | chain) + (1 | meaning)
## m1: increaseIconicity ~ systematicity.increase + (condition * gen) +
## m1:     (1 | chain) + (1 | meaning)
## m2: increaseIconicity ~ systematicity.increase + (systematicity.increase:condition) +
## m2:     (condition * gen) + (1 | chain) + (1 | meaning)
## m3: increaseIconicity ~ systematicity.increase * condition * gen +
## m3:     (1 | chain) + (1 | meaning)
##    Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0  7 -1025.8 -991.71 519.93  -1039.8
## m1  8 -1026.5 -987.44 521.23  -1042.5 2.6088      1     0.1063
## m2  9 -1029.2 -985.33 523.62  -1047.2 4.7734      1     0.0289 *
## m3 11 -1027.6 -973.93 524.80  -1049.6 2.3605      2     0.3072
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There was a significant interaction between systematicity increase and condition (log likelihood difference = 2.4 , df = 1 , Chi Squared = 4.77 , p = 0.029).

Plot the fixed effects of model 2. The relationship between systematicity and iconicity is more negative in the reproduction condition:

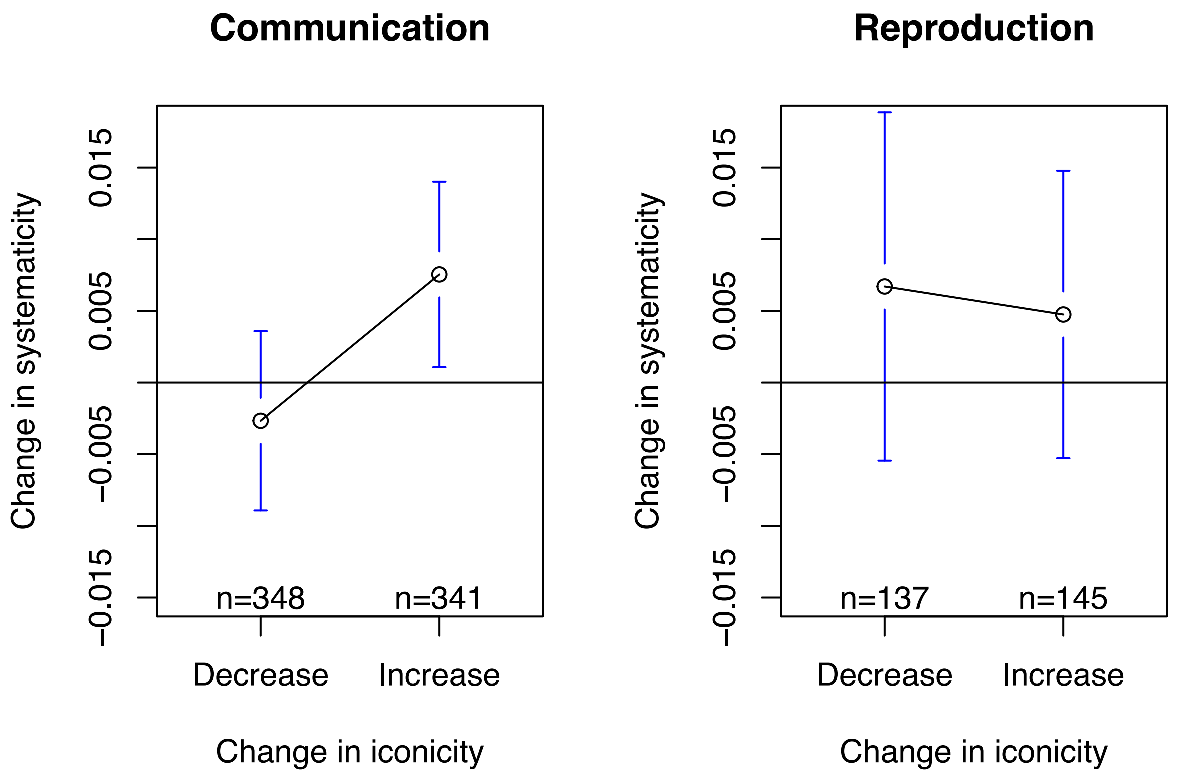
```
sjp.lmer(m2,type='fe', p.kr = F)
```

```
## Computing p-values via Wald-statistics approximation (treating t as Wald z).
```



Plot differences in raw data:

```r
ylimx= c(-0.015,0.018)
par(mfrow=c(1,2))
plotmeans(systematicity.increase ~ increaseIconicity>0,
          data=datax[datax$condition=="Comm",],
          legends = c("Decrease","Increase"),
          xlab="Change in iconicity",
          ylab="Change in systematicity",
          main = "Communication",
          ylim = ylimx)
abline(h=0)
plotmeans(systematicity.increase ~ increaseIconicity>0,
          data=datax[datax$Human & datax$condition=="Learn",],
          legends = c("Decrease","Increase"),
          xlab="Change in iconicity",
          ylab="Change in systematicity",
          main = "Reproduction",
          ylim = ylimx)

abline(h=0)
```


Communication
Reproduction
Change in systematicity
0.015
0.005
−0.005
−0.015
n=348
n=341
n=137
n=145
Decrease
Increase
Change in iconicity