Slide whistle stats

Contents

Load libraries

```
library(lme4)
library(blme)
library(lattice)
library(gplots)
library(party)
library(influence.ME)
library(Rmisc)
```

Introduction

In this document we test whether signallers avoid steep regions when the curvature is higher. We predict speakers indeed will avoid unstable regions of the articulator for stronger biases, and instead use the stable (quantal) ones. Moreover, we expect that the effect will become more pronounced as generations go by, that is, that nonlinear biases indeed get amplified over time.

Load data

```
signals_MT = read.csv("../Data/Signals_MT_AllProductions.csv",stringsAsFactors=F)
signals_SONA = read.csv("../Data/Signals_SONA_AllProductions.csv",stringsAsFactors=F)

signals_MT$run = "MT"
signals_SONA$run = "SONA"
```

Variables

A description of the variables included in the datasets:

- averageSlope/averageSlope.hz: average absolute difference in physical space/hz space between samples. High value = lots of movement. mean(abs(diff(sig)))

- propZero/propZero.hz: proportion of times in which the difference between sample values was negligible. High value = lots of time standing still. sum(diff(sig)<5) / length(sig
- switches/switches.hz: Number of changes in direction. sum(abs((diff(diff(sig)>=0))))
- maxslope/maxslope.hz: Maximum change in singal. max(abs(diff(sig)))
- slope.move: average slope change excluding zero. mean(abs(diff(sig)[abs(diff(sig))>=5]))
- jerkyness: standard deviation between derivative of signal. Low value = smoothly changing signal. sd(abs(diff(sig)))
- prop.time.plateau: Proportion of time spent in the plateau region (physical space only). Steep regions are defined as: 0.1-0.3 and 0.5-0.7, all other values are plateau. (the prop.time.plateau.hz variable should be ignored)
- switches.between.sections: If the signal space is divided into three plateau regions and two steep regions, the number of times the signal switches from one region to another.
- steepness.sig.mean - the signal's average steepenss of curve. For each change in a unit of physical space, what is the proportional change in the bark scale signal (perceptual difference)? The steepness values are taken from the curve used to produce the signal. So, any signal produced with zero curvature should have the same steepness.sig.mean. The same signal drawn on different curvatures will have different values of steepness.sig.mean, in a way that's not easy to predict (high curvatures have more steep regions, but also more flat regions).
- steepness.sig.mean.max - same as steepness.sig.mean, but where the steepenss values are taken from the maximum curvature steepness, to normalise over curvature values.
- logfile: The name of the log file used in the experiment
- chain: chain number (according to the online server)
- curvature: value of the curvature c of the articulator mapping
- run: source of the data (MT = mechanical turk, SONA = SONA)
- physSignal: the raw values of the physical signal recorded by the program, as a string of values separated by underscores. Values are proportion of the way along the slider x 1000, where 0 is low and 1000 is high
- hzSignal: the raw values of the signal in hz played to the participant. Same format as above, but in hz.
- numLearningRounds: the number of learning trials that the participant took to learn all 3 meanings (multiples of 3). This is the total value for the participant, not per meaning. For a given participant, this value is duplicated across the entries for each meaning. (so to get the number of learning rounds for a given participant, you should just look at values for e.g. meaning==1)
- abortedReproductionAttempts: The total number of times a player retries to produce a signal. As above, the number is reproduced over the 3 meanings.

- physSigDistinctiveness: How distinctive are the three signals that a participant produced? Split the time/signal space into a 10 x 10 grid. Count the number of appearances the signal makes in each cell to get a matrix of numbers that represents the journey of the signal. Do that for each signal. Then for each signal pair, calculate the absolute difference between the two matrices. Two very different signals will produce a high value. Two identical signals will produce a value of zero. Note that, because the time dimension is considered, two signals with the same trajectory but different speeds will look different. See the function "compareSignals" in GetData.R.
- hzSigDistinctiveness: Same as above, but in hz space.
- dataFileHzSignal: The file where the raw data is stored.

- dataFileHzSignal: The file where the raw data is stored.
- chain2: A unique identifier for the diffusion chain (chain number, log file, curvature). This should be used instead of 'chain', because 'chain' can have identical chain numbers for different curvature values.
- rawDataSource: Source of the raw data.

Results

Select dataset:

```
dx = rbind(signals_SONA,signals_MT)
# Add run number to distinguish chains from different experiment platforms
dx$chain2 = paste(dx$run,dx$chain2)

# Optionally, choose only first attempt
#dx= dx[dx$attempt==1 & dx$curvature!=0.5,]
dx =dx[dx$gen!=10,]
```

Center variables

```
dx$prop.time.plateau.norm = dx$prop.time.plateau - mean(dx$prop.time.plateau)
dx$steepness.sig.mean.max.norm = dx$steepness.sig.mean.max - mean(dx$steepness.sig.mean.max)

dx$gen = dx$gen - 5
dx$curvature.center = dx$curvature - 0.2

dx$participant = paste(dx$chain2, dx$gen)
```

Some stats:

```
# Datapoints:
nrow(dx)
```

```
## [1] 1612
```

```
# Number of chains:
length(unique(dx$chain2))
```

```
## [1] 45
```

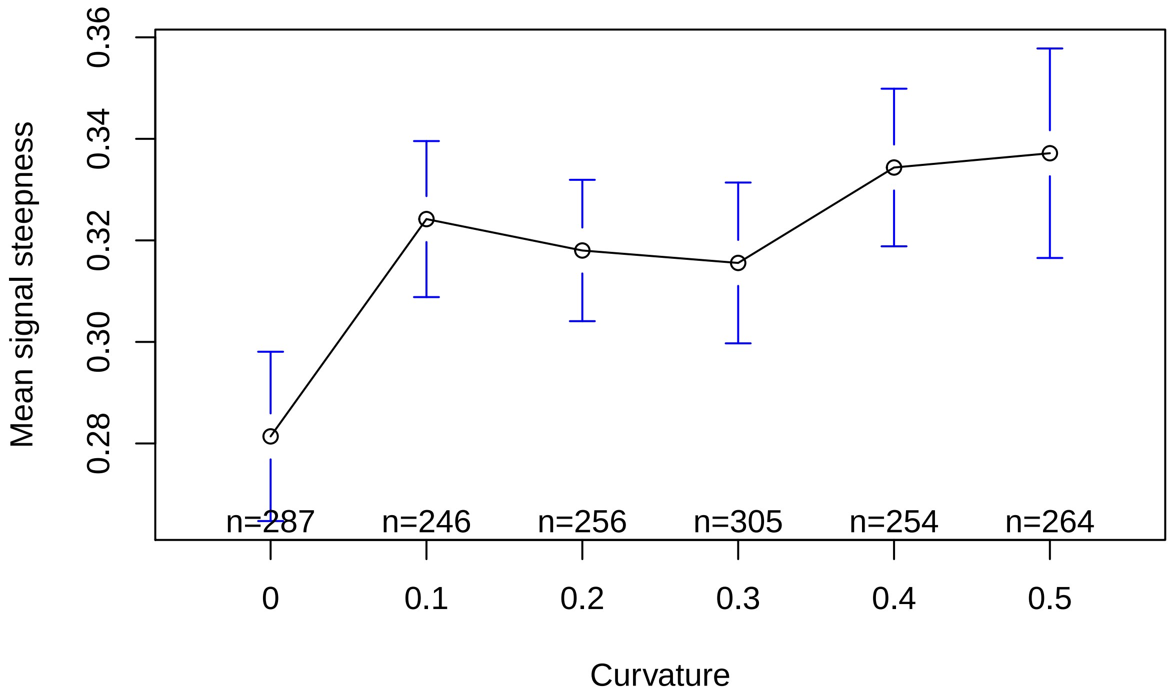
```
# Number of participants:
length((unique(paste(dx$chain2, dx$participant))))
```

```
## [1] 341
```

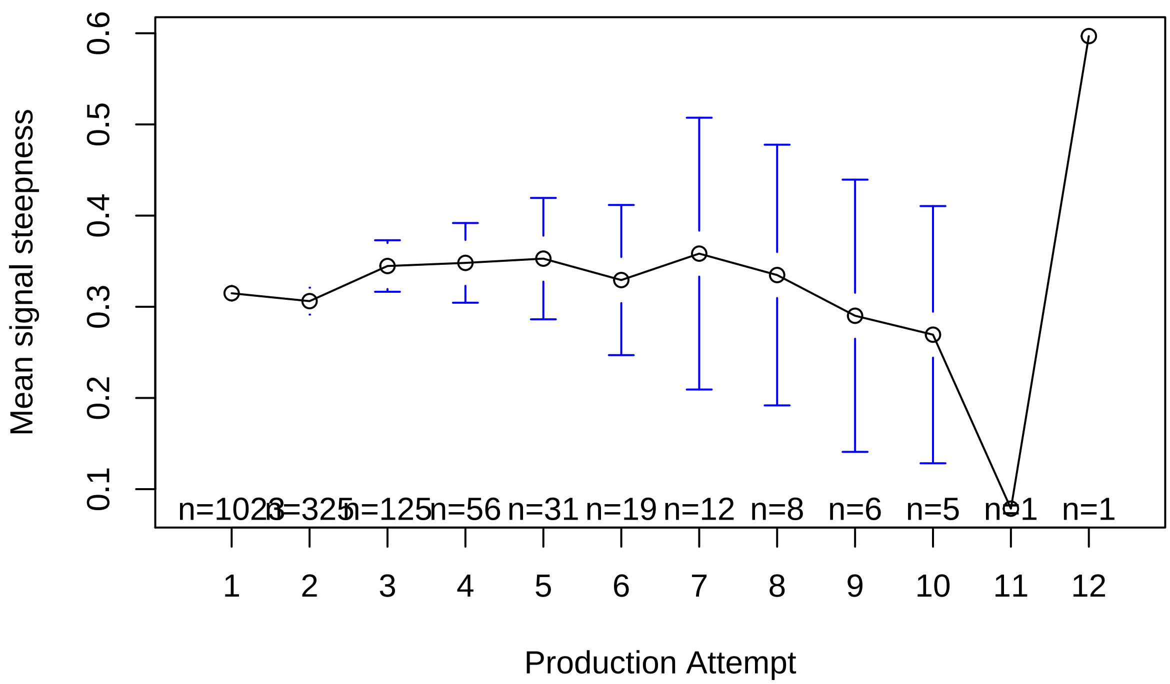
Steepness measure

Plot mean steepenss by curvature.

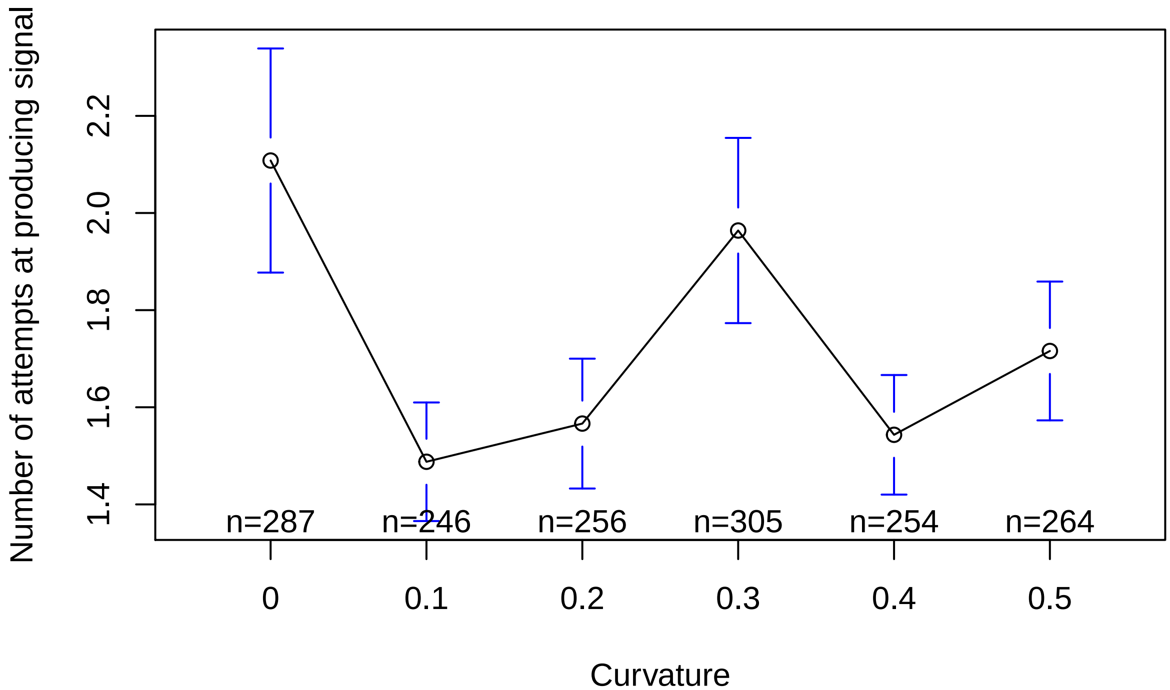
```
plotmeans(steepness.sig.mean.max~curvature, data=dx,
          ylab="Mean signal steepness",
          xlab='Curvature')
```




```r
plotmeans(steepness.sig.mean.max~attempt, data=dx,
          ylab="Mean signal steepness",
          xlab='Production Attempt')
```




```r
plotmeans(attempt~curvature, data=dx,
          ylab="Number of attempts at producing signal",
          xlab='Curvature')
```



Random effects

Use model comparison to test whether particular random effects are needed.

```
bcontrol = lmerControl(optimizer = 'Nelder_Mead')

m0x = blmer(steepness.sig.mean.max~
        1 + attempt +
        (1 | chain2/run),
        data = dx,
        control = bcontrol)
m1x = update(m0x,~.+(1 |participant))
anova(m0x,m1x)
```

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

## Data: dx
## Models:
## m0x: steepness.sig.mean.max ~ 1 + attempt + (1 | chain2/run)
## m1x: steepness.sig.mean.max ~ attempt + (1 | chain2/run) + (1 | participant)
##     Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0x  5 -1851.7 -1824.8 930.87  -1861.7
## m1x  6 -1939.9 -1907.5 975.93  -1951.9 90.132      1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m2x = update(m1x,~.+(1 |meaning))
anova(m1x,m2x)
```

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

## Data: dx
## Models:
```

```
## m1x: steepness.sig.mean.max ~ attempt + (1 | chain2/run) + (1 | participant)
## m2x: steepness.sig.mean.max ~ attempt + (1 | chain2/run) + (1 | participant) +
## m2x:     (1 | meaning)
##     Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m1x  6 -1939.9 -1907.5 975.93  -1951.9
## m2x  7 -1938.4 -1900.7 976.22  -1952.4 0.5713      1     0.4498
```

```
m3x = update(m1x,~.+(1|chain2:run))
anova(m1x,m3x)
```

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

## Data: dx
## Models:
## m1x: steepness.sig.mean.max ~ attempt + (1 | chain2/run) + (1 | participant)
## m3x: steepness.sig.mean.max ~ attempt + (1 | chain2/run) + (1 | participant) +
## m3x:     (1 | chain2:run)
##     Df     AIC     BIC logLik deviance Chisq Chi Df Pr(>Chisq)
## m1x  6 -1939.9 -1907.5 975.93  -1951.9
## m3x  7 -1937.9 -1900.2 975.93  -1951.9     0      1          1
```

Test fixed effects

```
# Null model
m0= blmer(steepness.sig.mean.max.norm~
        1 +
        (1 | chain2/run) +
        (1 | participant) +
        (1 | meaning),
        data = dx,
        control = bcontrol)
# add attempt number
m0.5= update(m0, ~.+attempt)
# add curavture
m1= update(m0.5, ~.+curvature.center)
# add generation
m2= update(m1,  ~.+gen)
# add interaction between curvature and generation
m3= update(m2,  ~.+curvature.center : gen)
# Quadratic term of curvature
m4=  update(m3, ~.+ I(curvature^2))
# Interaction between quadratic term of curvature and generation
m5= update(m4, ~.+I(curvature^2):gen)
```

Use model comparison to test the effect of each variable.

```
anova(m0,m0.5,m1,m2,m3,m4,m5)
```

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

## Data: dx
## Models:
## m0: steepness.sig.mean.max.norm ~ 1 + (1 | chain2/run) + (1 | participant) +
## m0:     (1 | meaning)
## m0.5: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m0.5:     (1 | meaning) + attempt
```

```
## m1: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m1:     (1 | meaning) + attempt + curvature.center
## m2: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m2:     (1 | meaning) + attempt + curvature.center + gen
## m3: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m3:     (1 | meaning) + attempt + curvature.center + gen + curvature.center:gen
## m4: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m4:     (1 | meaning) + attempt + curvature.center + gen + I(curvature^2) +
## m4:     curvature.center:gen
## m5: steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
## m5:     (1 | meaning) + attempt + curvature.center + gen + I(curvature^2) +
## m5:     curvature.center:gen + gen:I(curvature^2)
##      Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## m0    6 -1935.6 -1903.3 973.82  -1947.6
## m0.5  7 -1938.4 -1900.7 976.22  -1952.4 4.7968      1    0.02851 *
## m1    8 -1941.5 -1898.4 978.74  -1957.5 5.0477      1    0.02466 *
## m2    9 -1943.2 -1894.7 980.59  -1961.2 3.6962      1    0.05454 .
## m3   10 -1946.8 -1892.9 983.39  -1966.8 5.5908      1    0.01805 *
## m4   11 -1945.0 -1885.8 983.49  -1967.0 0.2138      1    0.64380
## m5   12 -1943.3 -1878.7 983.66  -1967.3 0.3276      1    0.56707
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

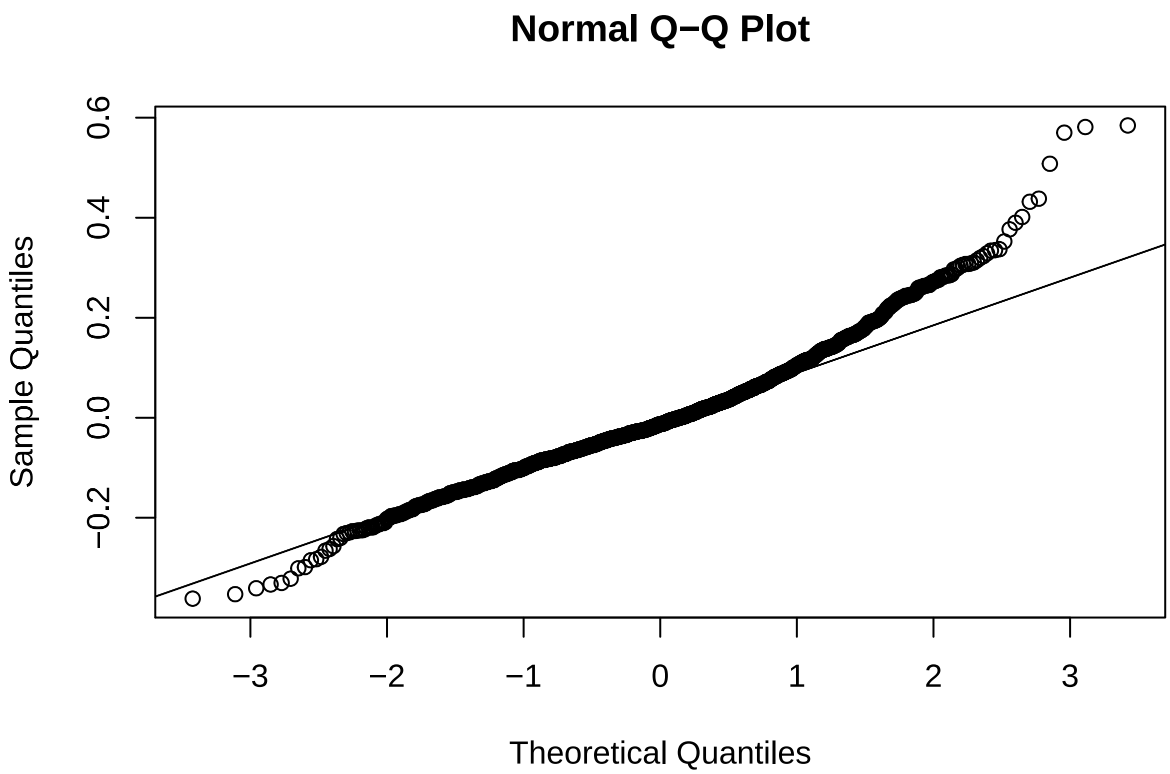
```
summary(m3)
```

```
## Cov prior  : participant ~ wishart(df = 3.5, scale = Inf, posterior.scale = cov, common.scale = TRUE)
##            : chain2 ~ wishart(df = 3.5, scale = Inf, posterior.scale = cov, common.scale = TRUE)
##            : run:chain2 ~ wishart(df = 3.5, scale = Inf, posterior.scale = cov, common.scale = TRUE)
##            : meaning ~ wishart(df = 3.5, scale = Inf, posterior.scale = cov, common.scale = TRUE)
## Prior dev  : 18.6887
##
## Linear mixed model fit by REML ['blmerMod']
## Formula:
## steepness.sig.mean.max.norm ~ (1 | chain2/run) + (1 | participant) +
##     (1 | meaning) + attempt + curvature.center + gen + curvature.center:gen
##    Data: dx
## Control: bcontrol
##
## REML criterion at convergence: -1923
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.9887 -0.5782 -0.1085  0.4826  4.8251
##
## Random effects:
##  Groups      Name        Variance  Std.Dev.
##  participant (Intercept) 0.0033159 0.05758
##  chain2      (Intercept) 0.0003950 0.01988
##  run:chain2  (Intercept) 0.0003951 0.01988
##  meaning     (Intercept) 0.0003472 0.01863
##  Residual                0.0146678 0.12111
## Number of obs: 1612, groups:
## participant, 341; chain2, 45; run:chain2, 45; meaning, 3
##
## Fixed effects:
```

```
##                        Estimate Std. Error t value
## (Intercept)           -0.013798   0.013182  -1.047
## attempt                0.005508   0.002589   2.127
## curvature.center       0.071049   0.037116   1.914
## gen                   -0.002191   0.001843  -1.189
## curvature.center:gen -0.024005   0.010421  -2.304
##
## Correlation of Fixed Effects:
##              (Intr) attmpt crvtr. gen
## attempt      -0.294
## curvtr.cntr -0.140  0.001
## gen           0.046  0.044 -0.055
## crvtr.cntr: -0.023 -0.017  0.110 -0.299
```

How good is the fit?

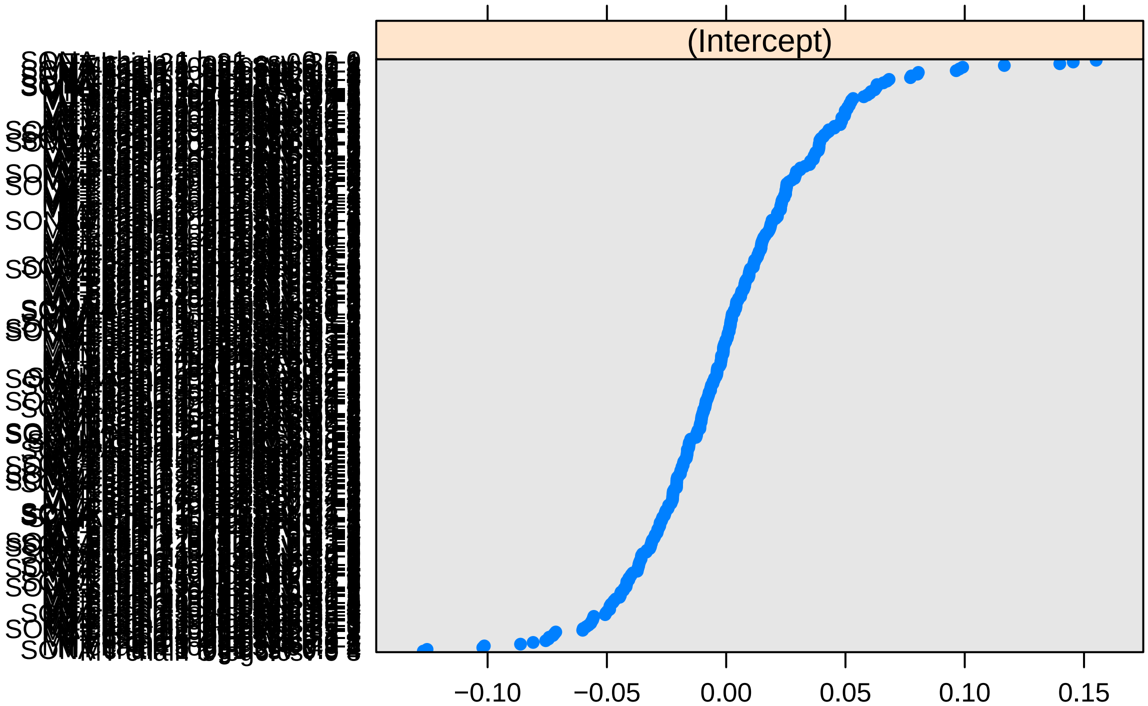
```
qqnorm(resid(m3))
qqline(resid(m3))
```




```
dotplot(ranef(m3))
```

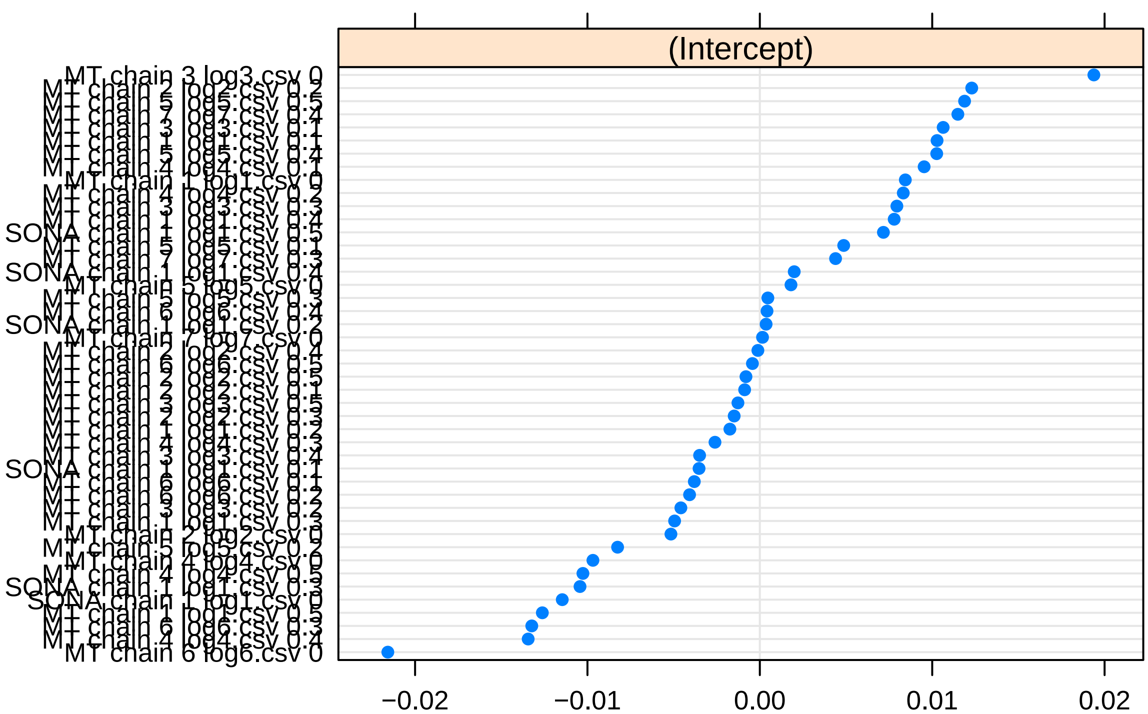
```
## $participant
```

participant



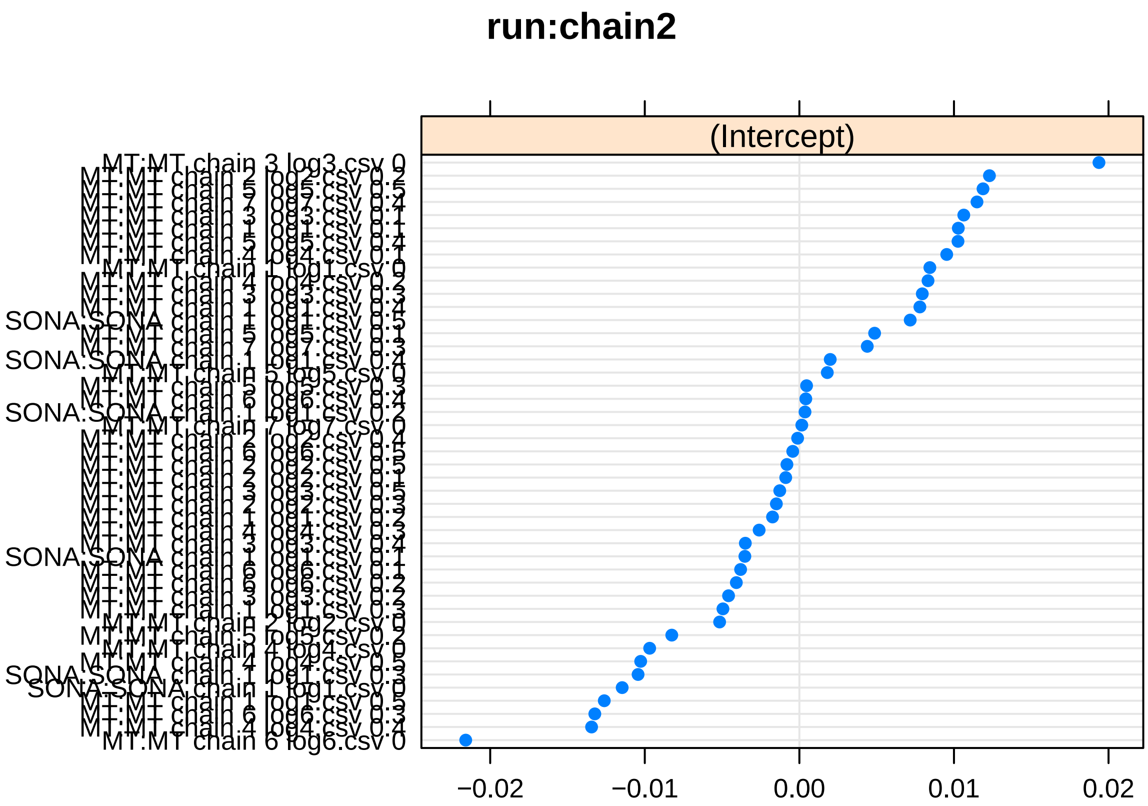

```
##
## $chain2
```

chain2

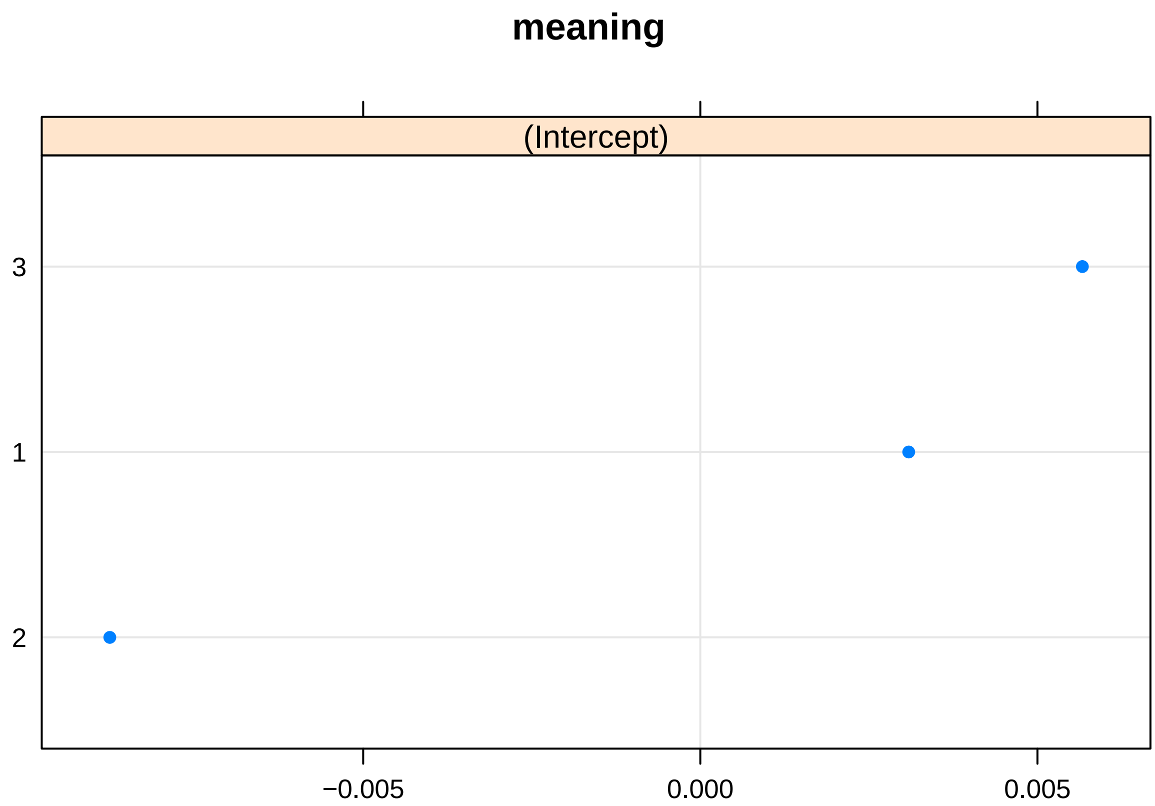



```
##
```

```
## $`run:chain2`
```




```
##
## $meaning
```


meaning
(Intercept)
3
1
2
−0.005
0.000
0.005