Test divergence between chains

Contents

Introduction

Winter (2014) suggests that non-linear spaces lead to *output stability* (similarity in acoustic signals), and *input variability* (variation in the motor input).

For our experiment, the prediction is that the variation in acoustic signals should be similar across curvature values, but the variation in motor signals should decrease as curvature increases.

The measure of variability used here is the following: Take two participants from different chains, but using the same curvature value and in the same generation number. The distance between a signal produced by participant A and a signal produced by participant B is calculated using dynamic time warping (not normalised for length). Each participant has 3 signals, but we can't be sure that the meaning mappings are being preserved throught the chain. Therefore, we match signals to each other based on similarity (using a greedy algorithm: the two most similar signals go together, then the next two most similar).

This leads to a set of distances between each participants in the same chain and generation. So, if generations are laid out horizontally, this is like taking a vertical slice across chains.

The same process is applied to the acoustic output (the pitch produced, in Bark scale) and the motor input (where the participant's finger is).

We predict that the distance between motor signals will be predicted by:

- generation (positive, chains diverge from each other)
- curvature x generation (distance will increase faster over generations for higher curvatures)

Libraries

```
library(blme)
library(party)
library(sjPlot)
library(gplots)
library(dplyr)
```

Load data

Note that, because this is a measure of similarity across chains, the SONA data can't be used on its own, because there is only one chain in each curvature condition.

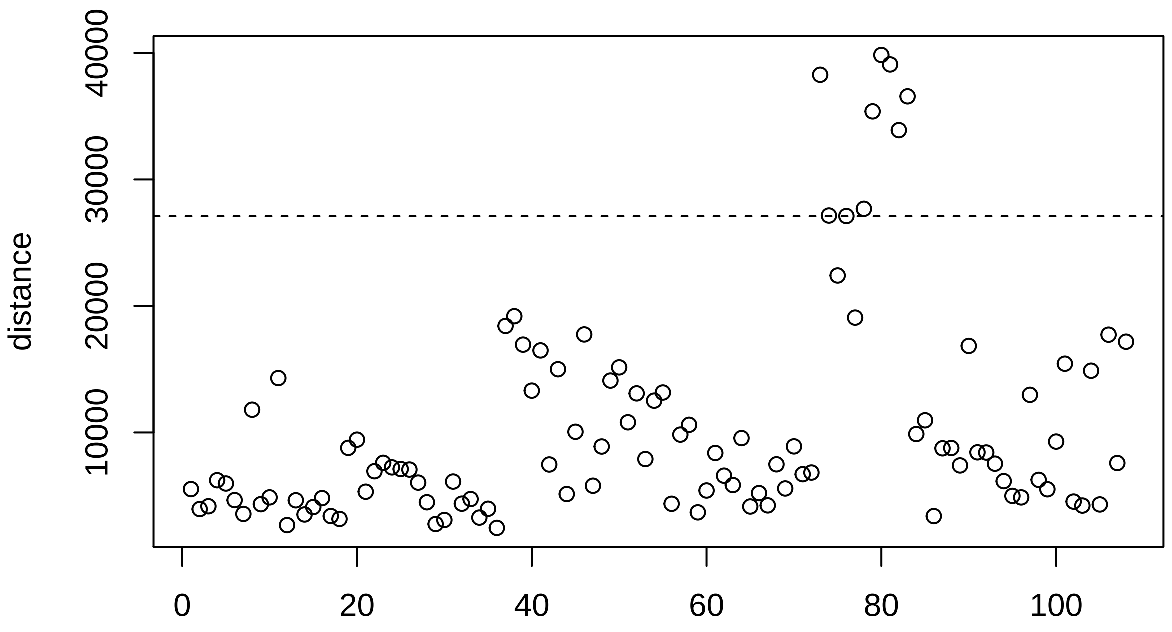
```
# The similarity between physical signals across chains
load("stabilityAcrossChains.Rdat")
dx = stability.accross.chains[complete.cases(stability.accross.chains),]
```

The full data includes chains with only one complete generation. There is a possibility of removing these, but do not for now.

```
# remove chains with only one generaion
#numOfGensPerChainA = tapply(dx$gen,dx$chainA,function(X){length(unique(X))})
#numOfGensPerChainB = tapply(dx$gen,dx$chainB,function(X){length(unique(X))})
#toExclude = c(names(numOfGensPerChainA[numOfGensPerChainA<=1]),
#               names(numOfGensPerChainB[numOfGensPerChainB<=1]))
#dx = dx[!dx$chainA %in% toExclude,]
#dx = dx[!dx$chainB %in% toExclude,]
```

Look for outliers: The following plot looks at the distances in the first generation.

```
plot(dx$distance[dx$gen==1],xlab='',ylab='distance')
abline(h=mean(dx$distance[dx$gen==1]) + (sd(dx$distance[dx$gen==1])*1.96),lty=2)
```



All the outliers are in chain 2 for curvature = 0.4. The first generation produced very strange signals, which then had an impact on the rest of the chain:

```
dx = dx[dx$chainA!="chain 2 log2.csv 0.4",]
dx = dx[dx$chainB!="chain 2 log2.csv 0.4",]
```

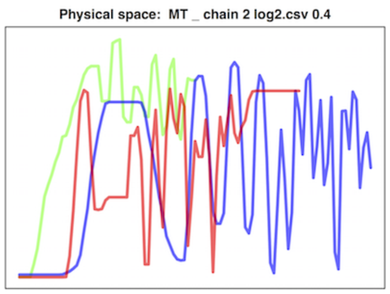


Figure 1: Strange initial signal for chain 2, curvature = 0.4

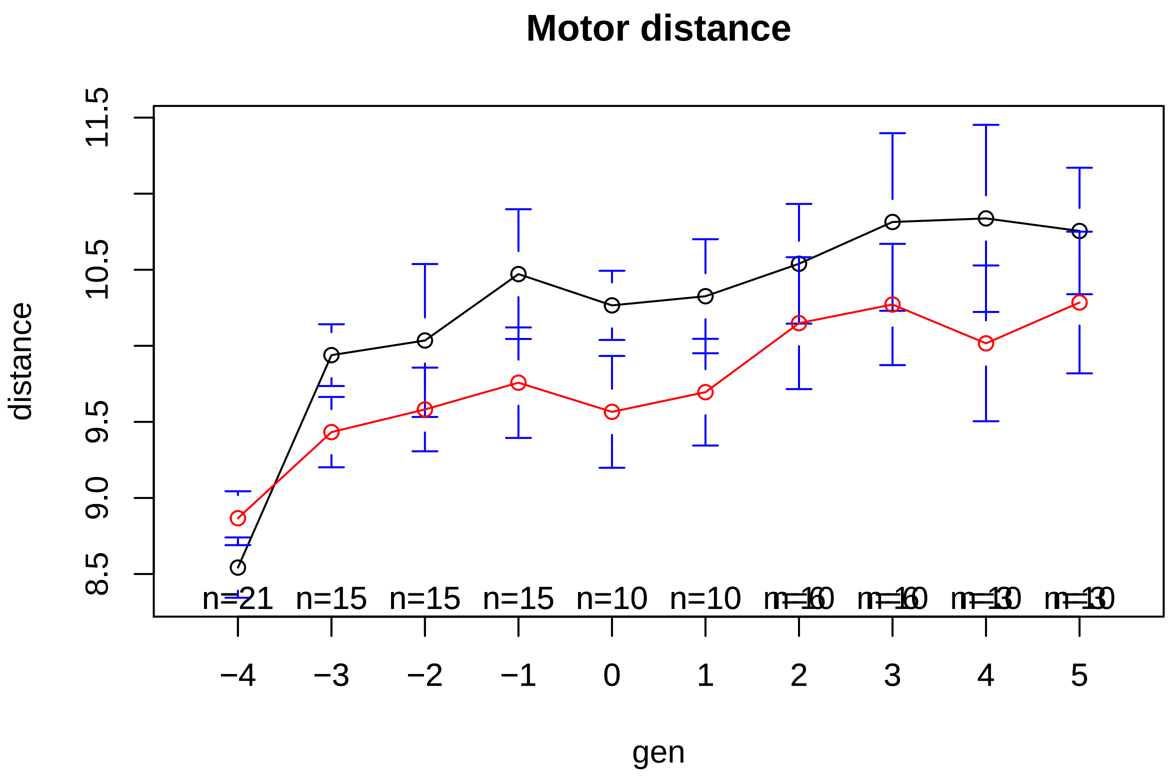
Normalise variables

```
dx$distance = log(dx$distance)
dx$acousticDistance = log(dx$acousticDistance)

dx$distance.norm = dx$distance - mean(dx$distance)
dx$acousticDistance.norm = dx$acousticDistance - mean(dx$acousticDistance)
dx$curvature.norm = dx$curvature - 0.2
dx$curvature.norm.q = dx$curvature.norm^2
dx$curvature.norm.c = dx$curvature.norm^3
dx$gen = dx$gen - 5

#dx = dx[dx$curvature!=0.4,]
```

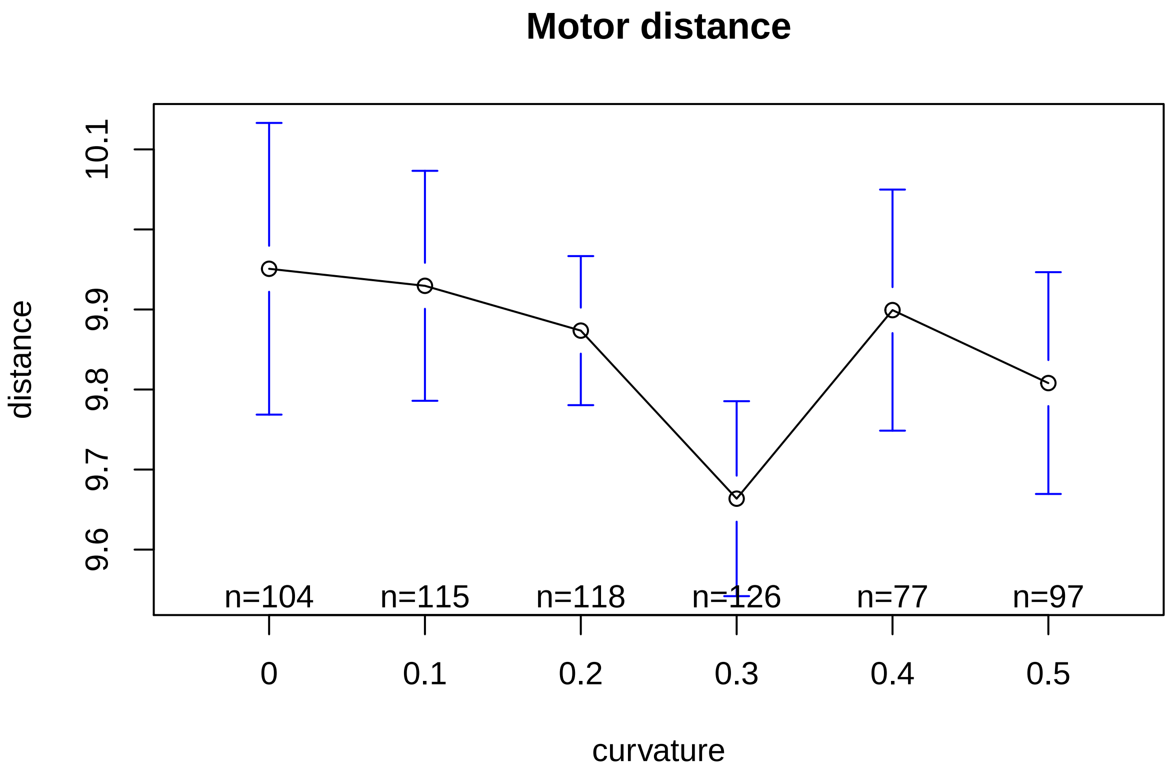
Plots

There appears to be something strange happening with curvature = 0.4.

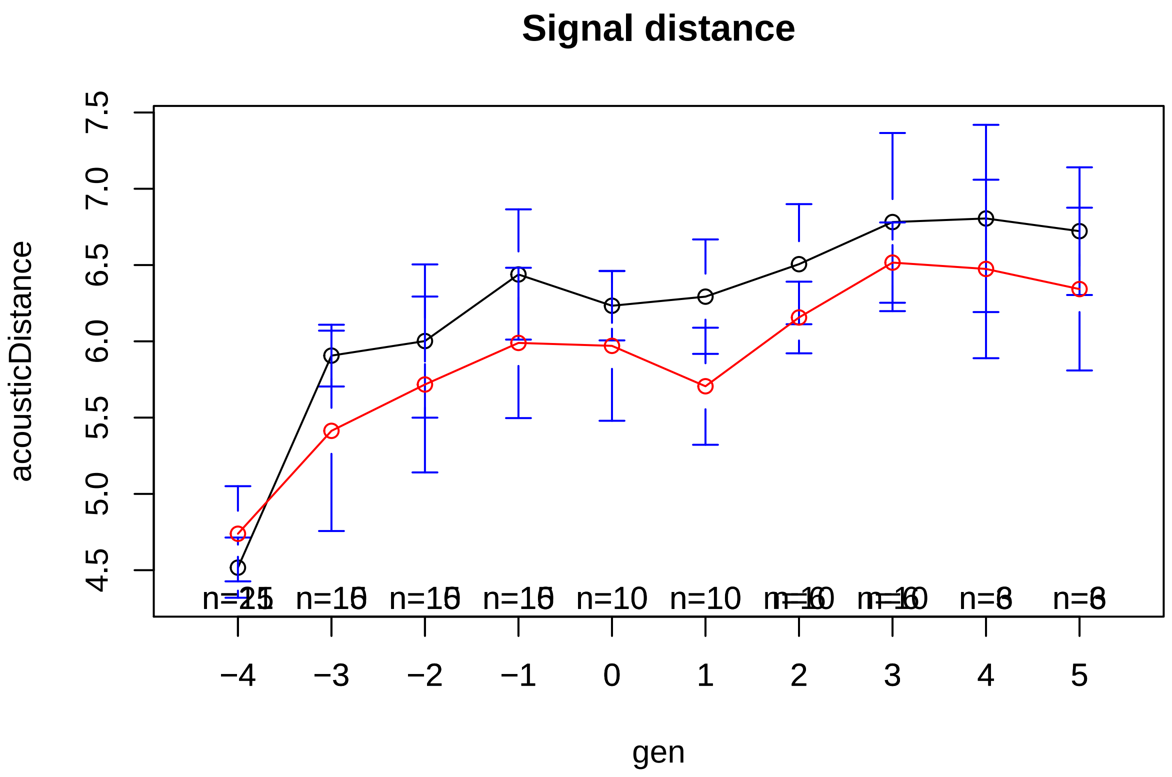
```
plotmeans(distance~gen,dx[dx$curvature==0,], main='Motor distance')
plotmeans(distance~gen,dx[dx$curvature==0.3,], add=T, col=2, main='Motor distance')
legend(1,80000, legend=c("Curvature = 0", "Curvature = 0.3"), lty=1, pch=1, col=1:2)
```



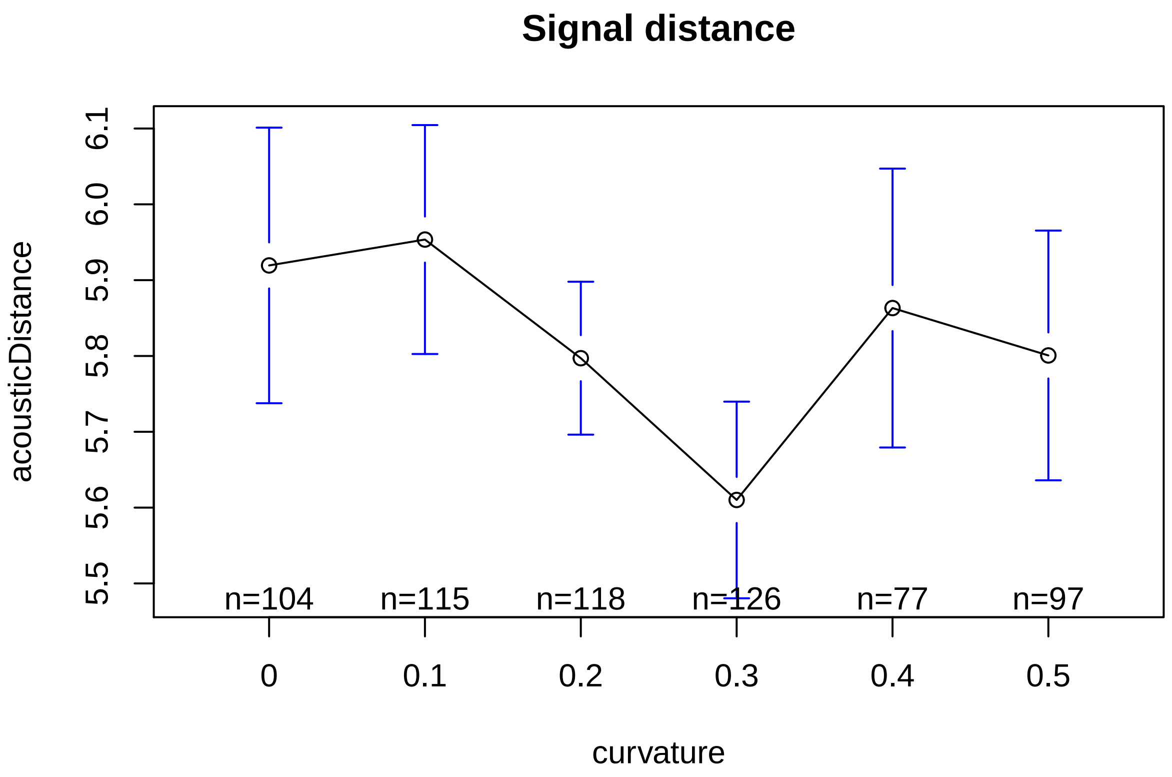
```
plotmeans(distance~curvature,dx, main='Motor distance')
```



```
plotmeans(acousticDistance~gen,dx[dx$curvature==0,], main='Signal distance')
plotmeans(acousticDistance~gen,dx[dx$curvature==0.5,], add=T, col=2, main='Signal distance')
legend(1,4.5e5, legend=c("Curvature = 0", "Curvature = 0.3"), lty=1, pch=1, col=1:2)
```



```r
plotmeans(acousticDistance~curvature,dx, main='Signal distance')
```



Analysis of motor signal

Build a series of mixed effects models, predicitng signal distance with random effects for the two chains being compared. There is a random slope for generation, but not curvature, since curvature does not vary by chain.

```
m0 = lmer(distance.norm~ 1 +
            (1+ gen|chainA) +
            (1+ gen|chainB),data=dx)
m1 = update(m0,~.+gen)
m2 = update(m1,~.+curvature.norm)
m3 = update(m2,~.+curvature.norm:gen)
m4 = update(m3,~.+curvature.norm.q)
m5 = update(m4,~.+curvature.norm.q:gen)
m6 = update(m5,~.+curvature.norm.c)
m7 = update(m6,~.+curvature.norm.c:gen)
```

Test the contribution of each variable.

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

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

```
x[1:nrow(x),1:ncol(x)]
```

```
##    Df    AIC    BIC  logLik deviance   Chisq Chi Df Pr(>Chisq)
## m0  8 1132.0 1167.7 -558.02   1116.0
## m1  9 1103.8 1143.9 -542.91   1085.8 30.2350      1  3.827e-08 ***
## m2 10 1102.4 1147.0 -541.21   1082.4  3.3947      1   0.065406 .
## m3 11 1099.8 1148.8 -538.88   1077.8  4.6623      1   0.030831 *
## m4 12 1100.9 1154.4 -538.44   1076.9  0.8677      1   0.351597
## m5 13 1096.0 1154.0 -535.03   1070.0  6.8352      1   0.008938 **
## m6 14 1098.0 1160.3 -534.98   1070.0  0.0963      1   0.756356
## m7 15 1098.9 1165.8 -534.47   1068.9  1.0195      1   0.312649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

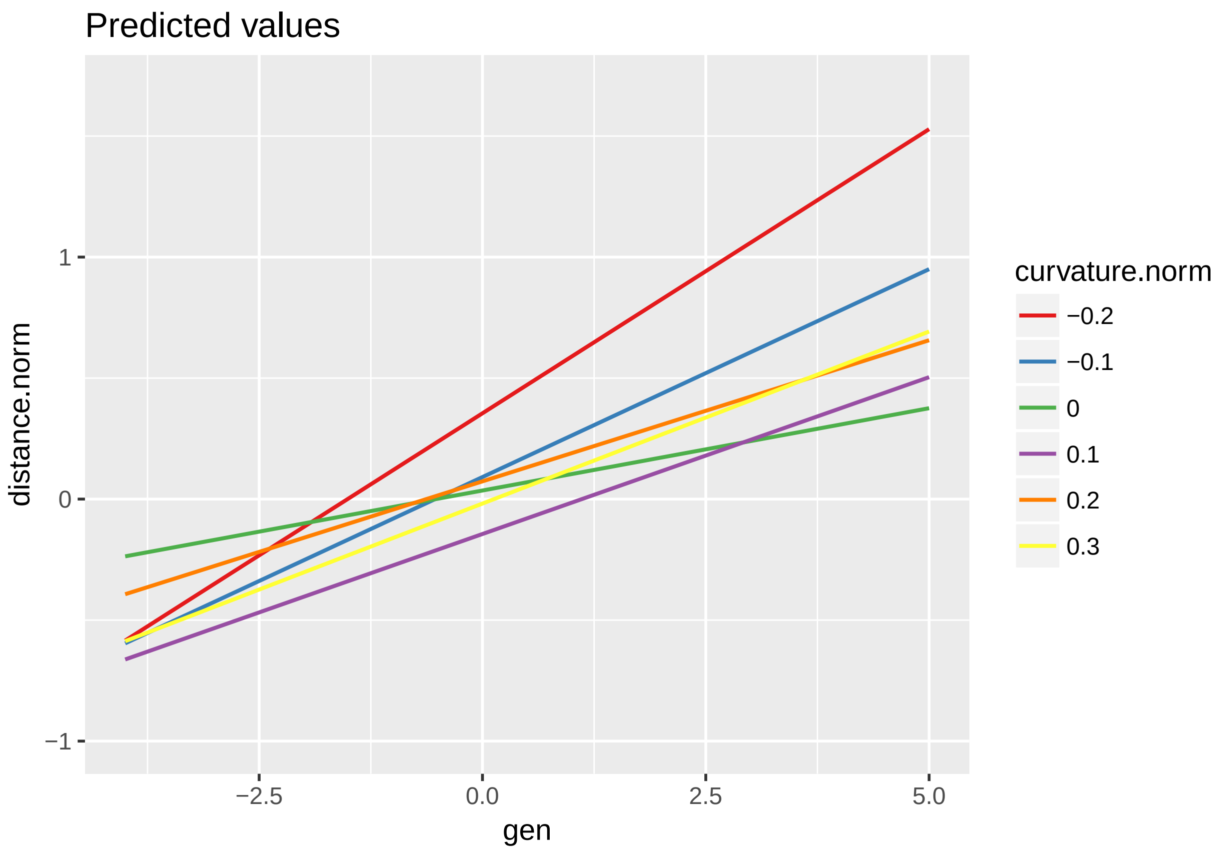
Curvature and the interaction between curvature and generation significantly improve the fit of the model.

```
summary(m5)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: distance.norm ~ (1 + gen | chainA) + (1 + gen | chainB) + gen +
##     curvature.norm + curvature.norm.q + gen:curvature.norm +
##     gen:curvature.norm.q
##    Data: dx
##
## REML criterion at convergence: 1079.7
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.97576 -0.58756  0.03947  0.61396  2.79252
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  chainA   (Intercept) 0.026775 0.16363
##           gen         0.000803 0.02834  -0.08
##  chainB   (Intercept) 0.039303 0.19825
##           gen          0.005872 0.07663  0.52
##  Residual              0.267653 0.51735
## Number of obs: 637, groups:  chainA, 32; chainB, 32
##
## Fixed effects:
##                        Estimate Std. Error t value
## (Intercept)             0.01854    0.08212   0.226
## gen                     0.12490    0.02589   4.824
## curvature.norm         -1.38001    0.40799  -3.382
## curvature.norm.q        4.20727    2.25363   1.867
## gen:curvature.norm     -0.42929    0.12850  -3.341
## gen:curvature.norm.q  1.80864    0.70924   2.550
##
## Correlation of Fixed Effects:
##              (Intr) gen    crvtr. crvt.. gn:cr.
## gen           0.358
## curvatr.nrm  0.239  0.096
## crvtr.nrm.q -0.700 -0.248 -0.551
## gn:crvtr.nr  0.096  0.230  0.442 -0.256
## gn:crvtr.n. -0.249 -0.699 -0.257  0.388 -0.563
```

Plot the model predictions (note that the curvature variable is normed to be centered around 0, so curvature.norm = -0.2 represents a curvature of 0).

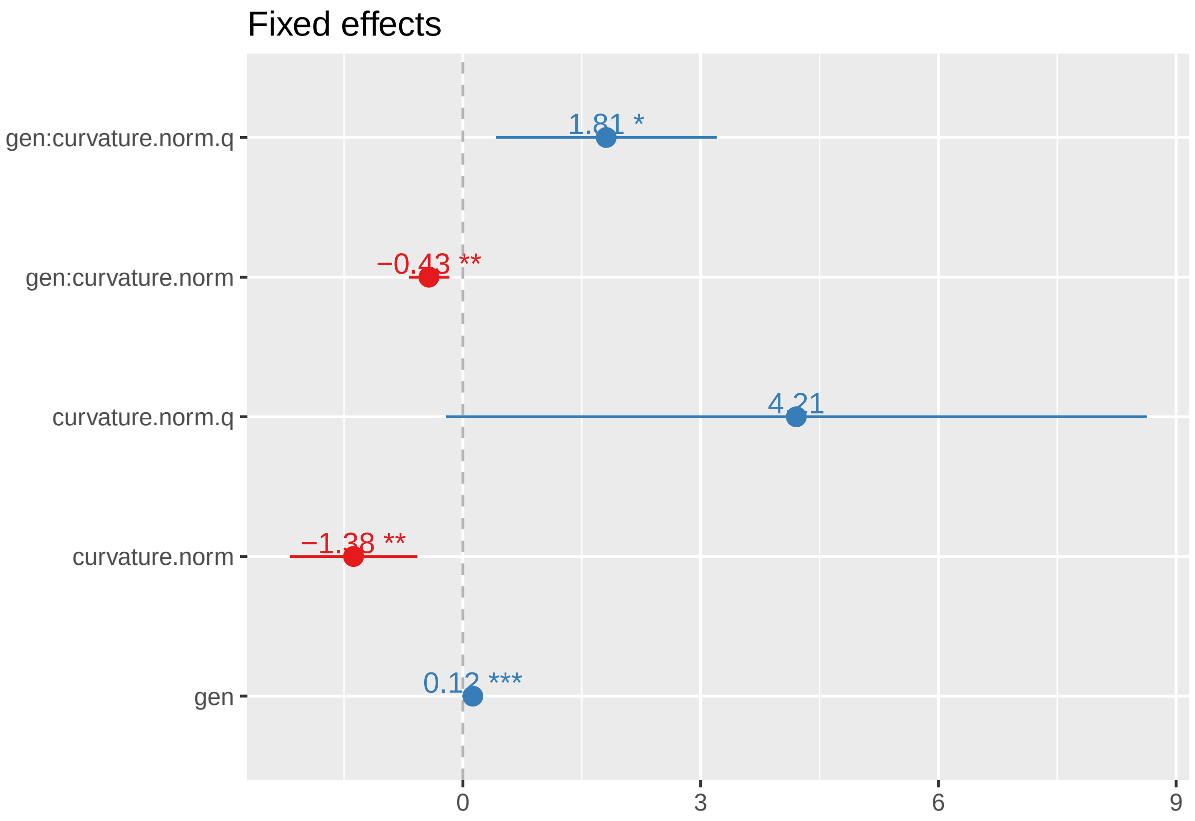
```
sjp.lmer(m5, 'pred', c("gen","curvature.norm"), facet.grid = F)
```



Plot the estimates:

```r
sjp.lmer(m5, 'fe')
```

```
## Warning: replacing previous import 'lme4::sigma' by 'stats::sigma' when
## loading 'pbkrtest'

## 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
```



If we look at the model predictions, we see that:

- signal distance decreases as curvature increases (marginal)
- signal distance increases with each generation (chains diverge)
- signal distance increases faster over generations for **lower** curvatures.

Analysis of acoustic signal

Build a series of mixed effects models, predicitng signal distance with random effects for the two chains being compared.

```
#dx = dx[dx$curvature<0.4,]

m0A = lmer(acousticDistance.norm~ 1 +
            (1+ gen|chainA) +
            (1+ gen|chainB),data=dx)
m1A = update(m0A,~.+gen)
m2A = update(m1A,~.+curvature.norm)
m3A = update(m2A,~.+curvature.norm:gen)
m4A = update(m3A,~.+curvature.norm.q)
m5A = update(m4A,~.+curvature.norm.q:gen)
m6A = update(m5A,~.+curvature.norm.c)
m7A = update(m6A,~.+curvature.norm.c:gen)
```

Test the contribution of each variable.

```
x2 = anova(m0A,m1A,m2A,m3A,m4A,m5A,m6A,m7A)
```

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

```
x2[1:nrow(x2),1:ncol(x2)]
```

```
##     Df    AIC    BIC  logLik deviance   Chisq Chi Df Pr(>Chisq)
## m0A  8 1213.5 1249.1 -598.74   1197.5
## m1A  9 1179.2 1219.4 -580.63   1161.2 36.2292      1  1.754e-09 ***
## m2A 10 1176.5 1221.1 -578.26   1156.5  4.7318      1   0.029610 *
## m3A 11 1176.2 1225.3 -577.12   1154.2  2.2727      1   0.131671
## m4A 12 1176.7 1230.2 -576.36   1152.7  1.5168      1   0.218098
## m5A 13 1170.0 1227.9 -571.97   1144.0  8.7840      1   0.003039 **
## m6A 14 1171.9 1234.3 -571.95   1143.9  0.0367      1   0.848043
## m7A 15 1173.5 1240.4 -571.77   1143.5  0.3623      1   0.547257
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Curvature and the interaction between curvature and generation significantly improve the fit of the model. The estimates are similar to the results for motor distance:

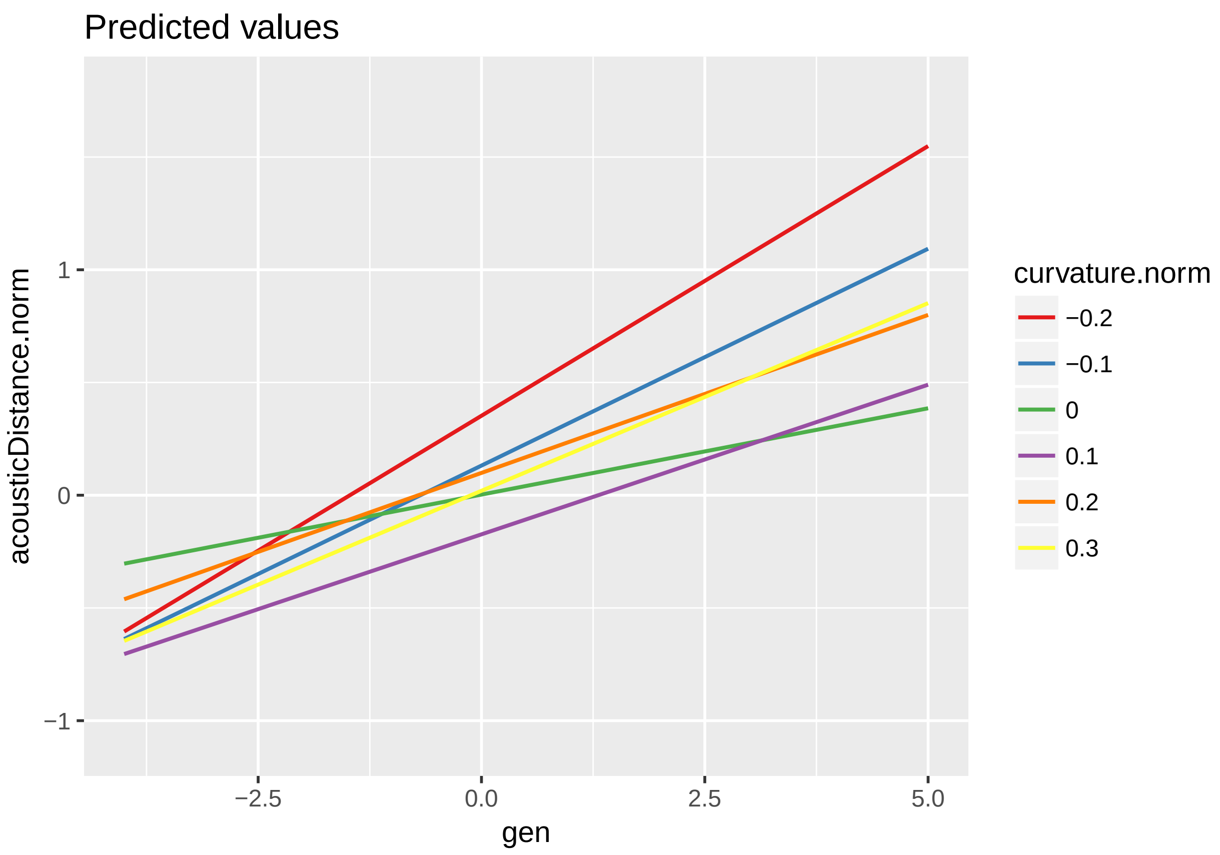
```
summary(m5A)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula:
## acousticDistance.norm ~ (1 + gen | chainA) + (1 + gen | chainB) +
##     gen + curvature.norm + curvature.norm.q + gen:curvature.norm +
##     gen:curvature.norm.q
##    Data: dx
##
## REML criterion at convergence: 1153
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.97673 -0.60094  0.01509  0.59207  2.71655
##
## Random effects:
```

```
##  Groups   Name        Variance Std.Dev. Corr
##  chainA   (Intercept) 0.044195 0.21023
##           gen         0.000387 0.01967  -0.03
##  chainB   (Intercept) 0.033720 0.18363
##           gen         0.005220 0.07225  0.15
##  Residual             0.301340 0.54894
## Number of obs: 637, groups:  chainA, 32; chainB, 32
##
## Fixed effects:
##                       Estimate Std. Error t value
## (Intercept)          -0.009953   0.088286  -0.113
## gen                   0.118199   0.024272   4.870
## curvature.norm       -1.380122   0.432735  -3.189
## curvature.norm.q      4.358040   2.410008   1.808
## gen:curvature.norm   -0.365072   0.120529  -3.029
## gen:curvature.norm.q  2.007962   0.665749   3.016
##
## Correlation of Fixed Effects:
##             (Intr) gen    crvtr. crvt.. gn:cr.
## gen          0.152
## curvatr.nrm  0.239  0.050
## crvtr.nrm.q -0.704 -0.105 -0.546
## gn:crvtr.nr  0.050  0.229  0.253 -0.153
## gn:crvtr.n. -0.105 -0.699 -0.154  0.190 -0.564
```

Plot the model predictions (note that the curvature variable is normed to be centered around 0, so curvature.norm = -0.2 represents a curvature of 0).

```
sjp.lmer(m5A, 'pred', c("gen","curvature.norm"), facet.grid = F)
```

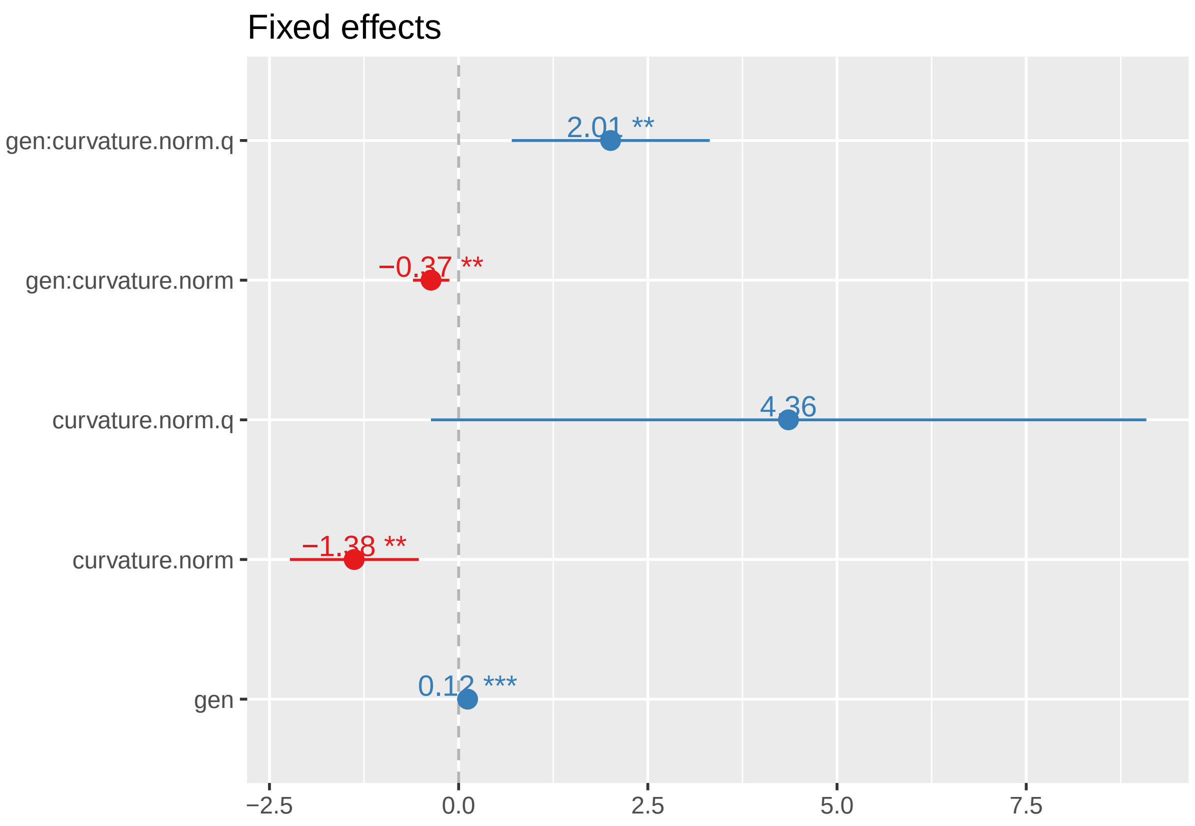


Plot the estimates:

```
sjp.lmer(m5A, 'fe')
```

```
## 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
```



Summary

- Motor signal distance increases with each generation (chains diverge)
- Motor signal distance increases faster over generations for **lower** curvatures.

For acoustic signals, the same patterns are there, but the interaction is weaker. There's also some kind of non-linear effect, but it's unclear why this would be.

The numbers jump around a bit, depending on which data is removed. But the general pattern is the same: motor signal distance increases faster with lower curvatures.