

# Predicting semantic alignment by cultural similarity: Subtitles data

Bill Thompson, Seán Roberts & Gary Lupyan

## Contents

<b>Introduction</b>	<b>97</b>
<b>Load libraries</b>	<b>97</b>
<b>All domains</b>	<b>98</b>
Load data . . . . .	98
LMER models . . . . .	101
MRM . . . . .	105
<b>Mantel tests</b>	<b>107</b>
Data prep . . . . .	107
Tests . . . . .	111
MRM . . . . .	112

## Introduction

This file replicates the tests for the main Wikipedia data on the subtitles data.

## Load libraries

```
library(ape)
library(ecodist)
library(lme4)
library(sjPlot)
library(ggplot2)
library(igraph)
library(lattice)
library(xtable)
```

Parameters (using data from NorthEuraLex and common crawl, k=100, unfiltered):

```
datasetName = "subs"
datasetLabel = "Subtitles"
lingDistancesFile = "../data/FAIR/nel-k100-subs-alignments-by-language-pair.csv"
lingDistancesByDomainFile = "../results/EA_distances/nel-k100-subs_with_ling.csv"
# (generated by ../processing/combineCultAndLingDistances.R)
```

## All domains

### Load data

Read the cultural distances:

```
cult = read.csv("../results/EA_distances/CulturalDistances_Long.csv", stringsAsFactors = F)
names(cult) = c("l1", "l2", "cult.dist")
```

Add language family:

```
l = read.csv("../data/FAIR_langauges_glottol_xdid.csv", stringsAsFactors = F)
g = read.csv("../data/glottolog-languoid.csv/languoid.csv", stringsAsFactors = F)
l$family = g[match(l$glotto, g$id),]$family_pk
l$family = g[match(l$family, g$pk),]$name
```

Read the semantic distances

```
ling = read.csv(lingDistancesFile, stringsAsFactors = F)
```

There are very few possible comparisons for Slovenian and Northern Sami, so we'll remove these:

```
ling = ling[!(ling$l1=="se" | ling$l2 == "se"),]
ling = ling[!(ling$l1=="sl" | ling$l2 == "sl"),]
```

Combine the linguistic and cultural distances. Note that we flip the cultural measure from a distance measure to a similarity measure.

```
cult$l1.iso2 = l[match(cult$l1, l$Language2),]$iso2
cult$l2.iso2 = l[match(cult$l2, l$Language2),]$iso2

fairisos = unique(c(ling$l1, ling$l2))
cultisos = unique(c(cult$l1.iso2, cult$l2.iso2))

cult = cult[(cult$l1.iso2 %in% fairisos) & (cult$l2.iso2 %in% fairisos),]
ling = ling[(ling$l1 %in% cultisos) & (ling$l2 %in% cultisos),]

matches = sapply(1:nrow(ling), function(i){
  which(cult$l1.iso2==ling$l1[i] & cult$l2.iso2==ling$l2[i])
})

ling$cult.dist = cult[matches,]$cult.dist
# Flip
ling$cult.dist = 1 - ling$cult.dist
# Scale
ling$cult.dist.center = scale(ling$cult.dist)
cdc.s = attr(ling$cult.dist.center, "scaled:scale")
cdc.c = attr(ling$cult.dist.center, "scaled:center")
ling$cult.dist.center = as.numeric(ling$cult.dist.center)
ling$comparison_count.center =
  scale(ling$comparison_count)

ling$family1 = l[match(ling$l1, l$iso2),]$family
ling$family2 = l[match(ling$l2, l$iso2),]$family
l[l$Language=="Arabic",]$autotyp.area= "Greater Mesopotamia"
l[l$Language=="Persian",]$autotyp.area= "Greater Mesopotamia"
ling$area1 = l[match(ling$l1, l$iso2),]$autotyp.area
ling$area2 = l[match(ling$l2, l$iso2),]$autotyp.area
```

```

fgroup = cbind(ling$family1,ling$family2)
fgroup = apply(fgroup,1,sort)
ling$family.group = apply(fgroup,2,paste,collapse=":")
agroup = cbind(ling$area1,ling$area2)
agroup = apply(agroup,1,sort)
ling$area.group = apply(agroup,2,paste,collapse=":")

ling$rho.center = scale(ling$local_alignment)

```

Each observation is now associated with a language family pair:

```
head(ling[,c("l1","l2","local_alignment","family.group")])
```

```

##      l1 l2 local_alignment      family.group
## 18 hy sq      0.01014219 Indo-European:Indo-European
## 23 hy ko      0.01862729      Indo-European:Koreanic
## 32 hy is      0.03000878 Indo-European:Indo-European
## 48 hy ja      0.05326118      Indo-European:Japonic
## 49 et hy      0.05535150      Indo-European:Uralic
## 52 hy nl      0.05835286 Indo-European:Indo-European

```

And the same is true for area:

```
tail(ling[,c("l1","l2","local_alignment","area.group")])
```

```

##      l1 l2 local_alignment      area.group
## 651 cs es      0.3339303      Europe:Europe
## 653 bg cs      0.3353707      Europe:Europe
## 656 el bg      0.3378855      Europe:Europe
## 659 el ru      0.3446584 Europe:Inner Asia
## 661 el cs      0.3625137      Europe:Europe
## 664 cs ru      0.3790251 Europe:Inner Asia

```

Number of observations:

```

# Number of datapoints:
nrow(ling)

```

```
## [1] 190
```

```

# Number of unique languages:
length(unique(unlist(ling[,c("l1","l2")]))))

```

```
## [1] 20
```

```

# Number of unique language families:
uniqueFamilies = unique(unlist(ling[,c("family1","family2")]))
length(uniqueFamilies)

```

```
## [1] 6
```

```

# Number of unique areas:
uniqueAreas = unique(unlist(ling[,c("area1","area2")]))
length(uniqueAreas)

```

```
## [1] 4
```

Cross-over between language families and areas:

```
tx = data.frame(lang= c(ling$l1,ling$l2),
  fam = c(ling$family1,ling$family2),
  area= c(ling$area1,ling$area2))
tx = tx[!duplicated(tx),]
table(tx$fam,tx$area)
```

```
##
##           Europe Greater Mesopotamia Inner Asia N Coast Asia
## Afro-Asiatic      0                1          0          0
## Indo-European     9                1          4          0
## Japonic           0                0          0          1
## Koreanic          0                0          0          1
## Turkic            0                1          0          0
## Uralic            1                0          1          0
```

## LMER models

Mixed effects model, predicting Linguistic similarities from cultural similarity, with random intercept for family and area and random slope for cultural similarity for family and area.

We start with a null model with random intercepts for family and area, and random slopes for cultural similarity by both. We add a fixed effect of the number of comparisons made for each data point (number of concepts that were available to compare). Then we add a fixed effect of cultural similarity

```
m0 = lmer(
  rho.center ~ 1 +
    (1 + cult.dist.center | family.group) +
    (1 + cult.dist.center | area.group),
  data = ling
)

## boundary (singular) fit: see ?isSingular

m0.5 = lmer(
  rho.center ~ 1 +
    comparison_count.center +
    (1 + cult.dist.center | family.group) +
    (1 + cult.dist.center | area.group),
  data = ling
)

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.0073357 (tol = 0.002, component 1)

m1 = lmer(
  rho.center ~ 1 +
    comparison_count.center +
    cult.dist.center +
    (1 + cult.dist.center | family.group) +
    (1 + cult.dist.center | area.group),
  data = ling
)

## boundary (singular) fit: see ?isSingular

an1 = anova(m0,m0.5,m1)

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

an1

## Data: ling
## Models:
## m0: rho.center ~ 1 + (1 + cult.dist.center | family.group) + (1 +
## m0:      cult.dist.center | area.group)
## m0.5: rho.center ~ 1 + comparison_count.center + (1 + cult.dist.center |
## m0.5:      family.group) + (1 + cult.dist.center | area.group)
## m1: rho.center ~ 1 + comparison_count.center + cult.dist.center +
## m1:      (1 + cult.dist.center | family.group) + (1 + cult.dist.center |
## m1:      area.group)
##      Df    AIC    BIC logLik deviance   Chisq Chi Df Pr(>Chisq)
## m0      8 409.01 434.99 -196.51   393.01
## m0.5    9 288.62 317.84 -135.31   270.62 122.3948      1    <2e-16 ***
## m1     10 289.88 322.35 -134.94   269.88   0.7417      1    0.3891
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Cultural similarity is significantly correlated with Linguistic similarity. Here are the model estimates:

```
summary(m1)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: rho.center ~ 1 + comparison_count.center + cult.dist.center +
##      (1 + cult.dist.center | family.group) + (1 + cult.dist.center |
##      area.group)
## Data: ling
##
## REML criterion at convergence: 282.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.8968 -0.7471 -0.0734  0.6181  3.6760
##
## Random effects:
## Groups      Name                Variance Std.Dev. Corr
## family.group (Intercept)         0.0002465 0.01570
##              cult.dist.center  0.0056335 0.07506 -1.00
## area.group   (Intercept)         0.0015278 0.03909
##              cult.dist.center  0.0211421 0.14540 -1.00
## Residual                        0.2322780 0.48195
## Number of obs: 190, groups:  family.group, 17; area.group, 10
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)    -0.01139   0.03975  -0.286
## comparison_count.center  0.81265   0.03807  21.346
## cult.dist.center    0.06060   0.07272   0.833
##
## Correlation of Fixed Effects:
##              (Intr) cmpr_
## cmprsn_cnt.   0.013
## clt.dst.cnt -0.231 -0.136
## convergence code: 0
## boundary (singular) fit: see ?isSingular
```

Plot the estimates, rescaling the variables back to the original units:

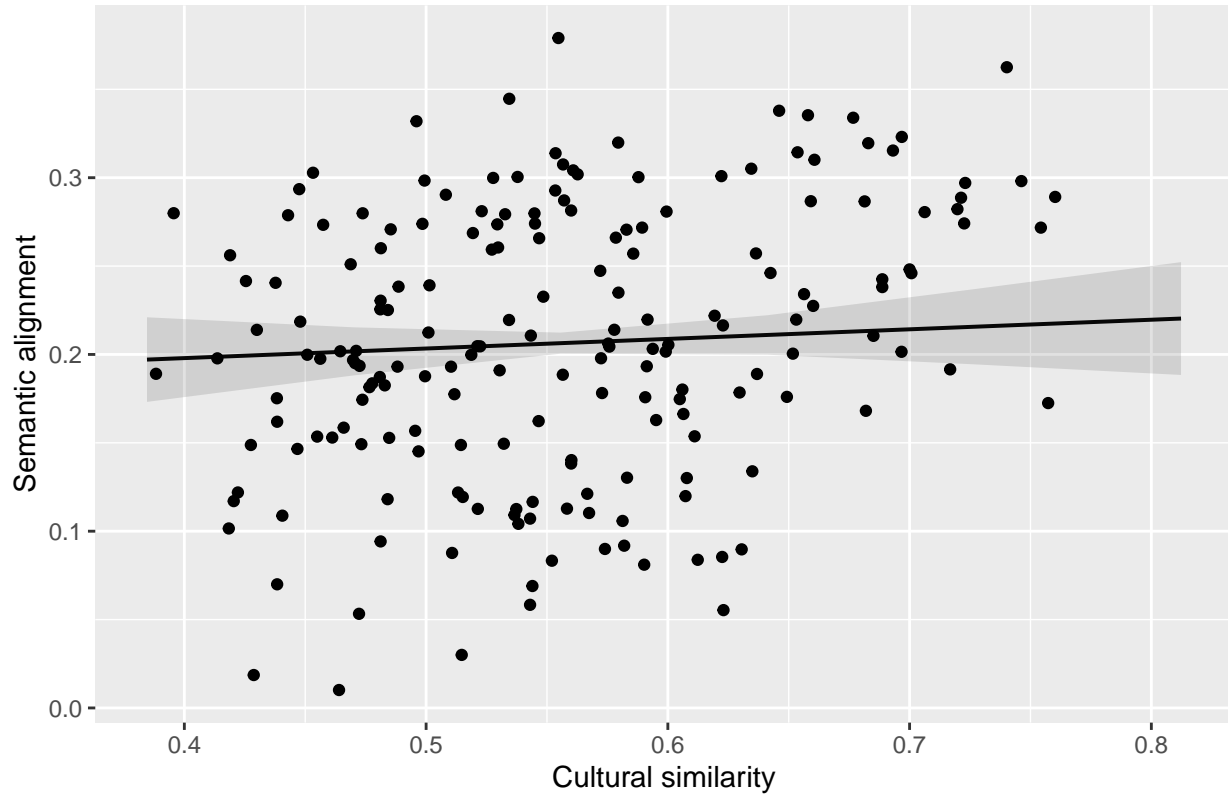
```
trans = function(X){
  X * attr(ling$rho.center,"scaled:scale") +
  attr(ling$rho.center,"scaled:center")
}

gx = plot_model(m1, 'pred', terms='cult.dist.center')
gx$data$predicted = trans(gx$data$predicted)
gx$data$conf.low = trans(gx$data$conf.low)
gx$data$conf.high = trans(gx$data$conf.high)
gx$data$x = gx$data$x *
  cdc.s + cdc.c
gx = gx + #coord_cartesian(ylim=c(0,0.5),
  # xlim=c(0.15,0.85)) +
```

```

xlab("Cultural similarity") +
ylab("Semantic alignment") +
ggtitle("") +
geom_point(data=ling,aes(x=cult.dist,y=local_alignment))
gx

```



```

pdf(paste0("../results/stats/",datasetName,"/CulturalDistance_Rho_Graph.pdf"),
    height=2.5, width=2.5)
gx
dev.off()

```

```

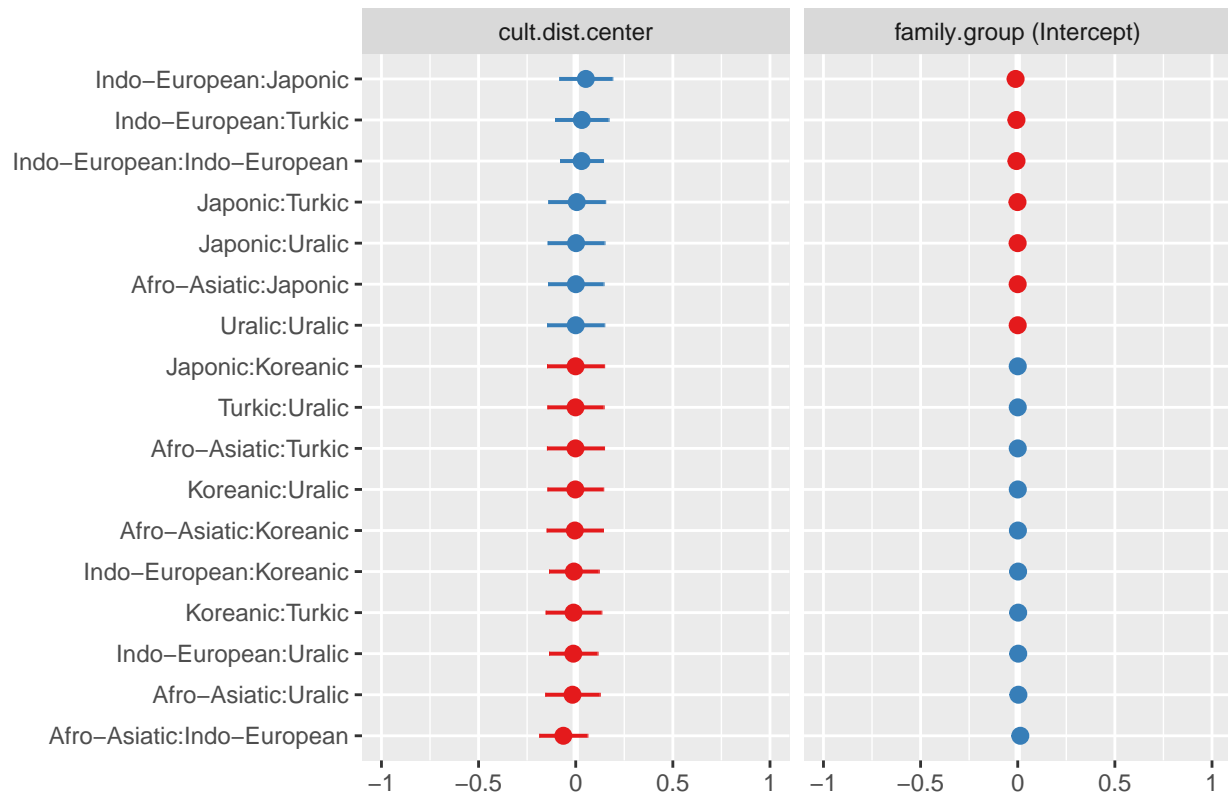
## pdf
## 2

Plot the random effects:
plot_model(m1,'re', sort.est = "cult.dist.center")

## [[1]]

```

## Random effects

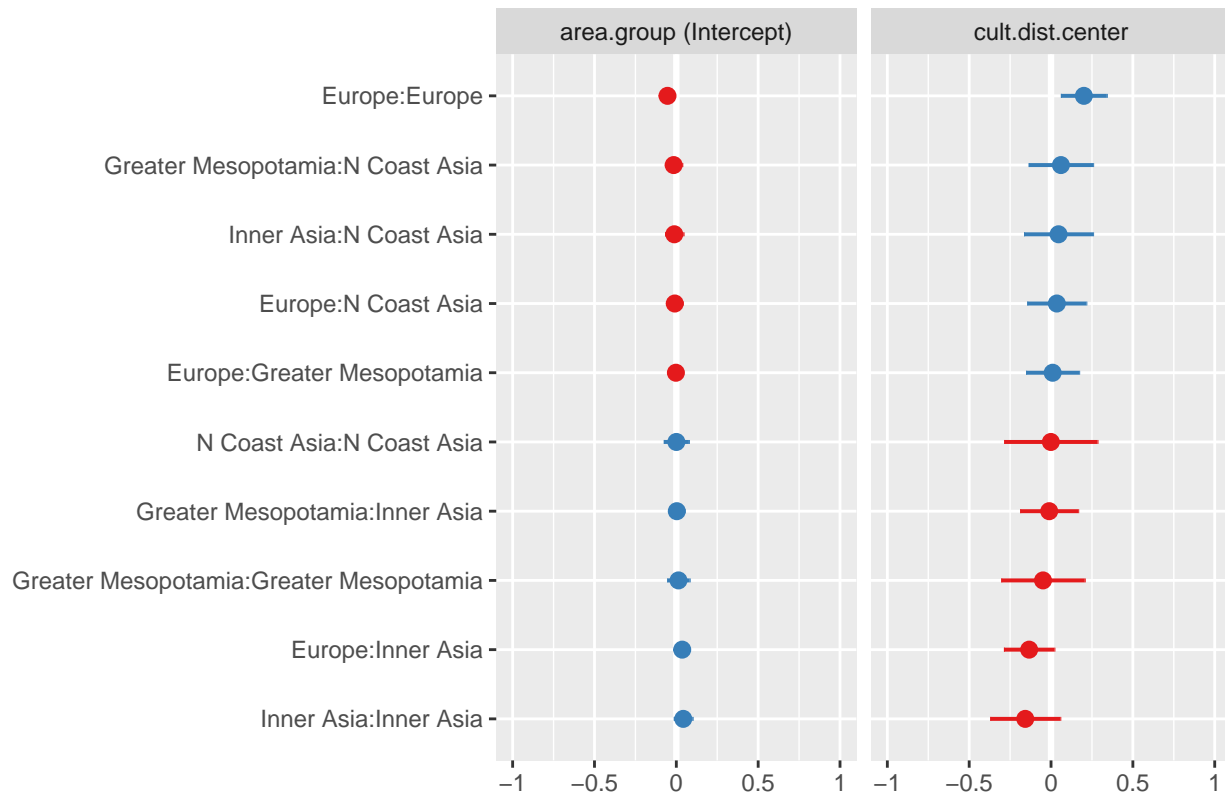


##

## [[2]]



## Random effects



## MRM

Use multiple regression on distance matrices to do the same test as above. The code below loads the data into a matrix format:

```
# Use graph method to make distance matrix
grph <- graph.data.frame(ling[,c("l1", "l2", 'local_alignment')], directed=FALSE)
# add value as a weight attribute
ling.m = get.adjacency(grph, attr="local_alignment", sparse=FALSE)
rownames(ling.m) = 1[match(rownames(ling.m), l$iso2), l$Language2]
colnames(ling.m) = 1[match(colnames(ling.m), l$iso2), l$Language2]
# Same for comparison_count.center
grph <- graph.data.frame(ling[,c("l1", "l2", 'comparison_count')], directed=FALSE)
# add value as a weight attribute
cc.m = get.adjacency(grph, attr="comparison_count", sparse=FALSE)
rownames(cc.m) = 1[match(rownames(cc.m), l$iso2), l$Language2]
colnames(cc.m) = 1[match(colnames(cc.m), l$iso2), l$Language2]

cult.m = read.csv("../results/EA_distances/CulturalDistances.csv", stringsAsFactors = F)
rownames(cult.m) = cult.m[,1]
cult.m = cult.m[,2:ncol(cult.m)]
cult.m = as.matrix(cult.m)
# Flip cultural value to distance
cult.m = 1-cult.m
mx = match(rownames(ling.m), rownames(cult.m))
cult.m = cult.m[mx, mx]
colnames(cult.m) = rownames(cult.m)
```

```

# Same/different matrix for language family
family.matrix = 1[match(rownames(ling.m),l$Language),]$family
family.matrix = outer(family.matrix,family.matrix,"!=") *1

# Load ASJP distances for second test
asjp = readRDS("../data/ASJP/asjp17-dists_FAIR.RData")
ling.m.glotto = 1[match(rownames(cult.m),l$Language2),]$glotto
ling.m.glotto = ling.m.glotto[ling.m.glotto %in% rownames(asjp)]
asjp.m = asjp[ling.m.glotto,ling.m.glotto]
asjp.lang.names = 1[match(rownames(asjp.m),l$glotto),]$Language2
# Matrices for second analysis with asjp
ling.m2 = ling.m[asjp.lang.names,asjp.lang.names]
cult.m2 = cult.m[asjp.lang.names,asjp.lang.names]
cc.m2 = cc.m[asjp.lang.names,asjp.lang.names]

# Load the geographic distances:
geoDist = read.csv("../data/GeographicDistances.csv",stringsAsFactors = F)
geoDist.m = as.matrix(geoDist)
geoDist.m = geoDist.m[!is.na(geoDist.m[,1]),!is.na(geoDist.m[1,])]
# Convert to log distance in thousand km
geoDist.m = log10(geoDist.m/1000)
geoDist.m[is.infinite(geoDist.m)] = 0
colnames(geoDist.m) = gsub("\\.", " ",colnames(geoDist.m))
rownames(geoDist.m) = colnames(geoDist.m)
geoDist.m1 = geoDist.m[rownames(ling.m),rownames(ling.m)]
geoDist.m2 = geoDist.m[rownames(ling.m2),rownames(ling.m2)]

# For missing comparisons, impute the mean:
ling.m[ling.m==0] = mean(ling$local_alignment)
diag(ling.m) = 0
ling.m2[ling.m2==0] = mean(ling.m2[ling.m2!=0])
diag(ling.m2) = 0

# center and scale values
ling.m = matrix(scale(as.vector(ling.m)),nrow=nrow(ling.m))
cc.m = matrix(scale(as.vector(cc.m)),nrow=nrow(cc.m))
cult.m = matrix(scale(as.vector(cult.m)),nrow=nrow(cult.m))
geoDist.m1 = matrix(scale(as.vector(geoDist.m1)),nrow=nrow(geoDist.m1))

asjp.m = matrix(scale(as.vector(asjp.m)),nrow=nrow(asjp.m))
ling.m2 = matrix(scale(as.vector(ling.m2)),nrow=nrow(ling.m2))
cc.m2 = matrix(scale(as.vector(cc.m2)),nrow=nrow(cc.m2))
cult.m2 = matrix(scale(as.vector(cult.m2)),nrow=nrow(cult.m2))
geoDist.m2 = matrix(scale(as.vector(geoDist.m2)),nrow=nrow(geoDist.m2))

```

Run the MRM model, predicting semantic alignment by cultural distance, controlling for family distance, geographic distance, and the comparison count (number of observations). Here, the family distance between two languages is just whether they are part of the same family. Note that this does not take into account particular values for particular families, nor the random slopes within families.

```
set.seed(289)
MRM.fam = ecodist::MRM(as.dist(ling.m) ~
  as.dist(cult.m) +
  as.dist(family.matrix) +
  as.dist(geoDist.m1) +
  as.dist(cc.m), nperm = 10000)

MRM.asjp = ecodist::MRM(as.dist(ling.m2) ~
  as.dist(cult.m2) +
  as.dist(asjp.m) +
  as.dist(geoDist.m2) +
  as.dist(cc.m2), nperm = 10000)

rownames(MRM.fam$coef) = c("Intercept", "Cultural distance", "Language family",
  "Geographic distance", "Comparison count")
colnames(MRM.fam$coef) = c("Estimate", "p-value")
statMRM.fam = xtable(MRM.fam$coef, digits = 3, display=c("s", "f", "fg"),
  caption = paste0(
    "MRM analysis predicting semantic alignment (",
    datasetLabel, "), with family control.  $R^2$=",
    signif(MRM.fam$r.squared[1], 3)))
print(statMRM.fam, "latex",
  file="../results/stats/tex/MRM_family_SUBS.tex")

rownames(MRM.asjp$coef) = c("Intercept", "Cultural distance", "ASJP",
  "Geographic distance", "Comparison count")
colnames(MRM.asjp$coef) = c("Estimate", "p-value")
statMRM.asjp = xtable(MRM.asjp$coef, digits = 3, display=c("s", "f", "fg"),
  caption = paste0(
    "MRM analysis predicting semantic alignment (",
    datasetLabel, "), with ASJP control.  $R^2$=",
    signif(MRM.asjp$r.squared[1], 3)))
print(statMRM.asjp, "latex",
  file="../results/stats/tex/MRM_ASJP_SUBS.tex")$$ 
```

## Mantel tests

Read the historical distances for Indo-European, based on the phylogenetic distances.

### Data prep

The geographic distances are loaded above (from “../data/GeographicDistances.csv”).

Load historical distances:

```
hist = read.csv("../data/trees/IndoEuropean_historical_distances.csv", stringsAsFactors = F)
hist = hist[!duplicated(hist[,1]), !duplicated(hist[,1])]
rownames(hist) = hist[,1]
hist = hist[,2:ncol(hist)]
hist.m = as.matrix(hist)
```

```
colnames(hist.m) = rownames(hist.m)
hist.m = hist.m/max(hist.m)
```

Read the cultural distance as a matrix:

```
cult.m = read.csv("../results/EA_distances/CulturalDistances.csv", stringsAsFactors = F)
rownames(cult.m) = cult.m[,1]
cult.m = cult.m[,2:ncol(cult.m)]
```

Flip the cultural distance into a cultural similarity measure:

```
cult.m = 1-cult.m
```

Convert the linguistic similarities to a matrix. This uses `igraph` to make an undirected graph from the long format with `local_alignment` as the edge weights, then output a matrix of adjacencies.

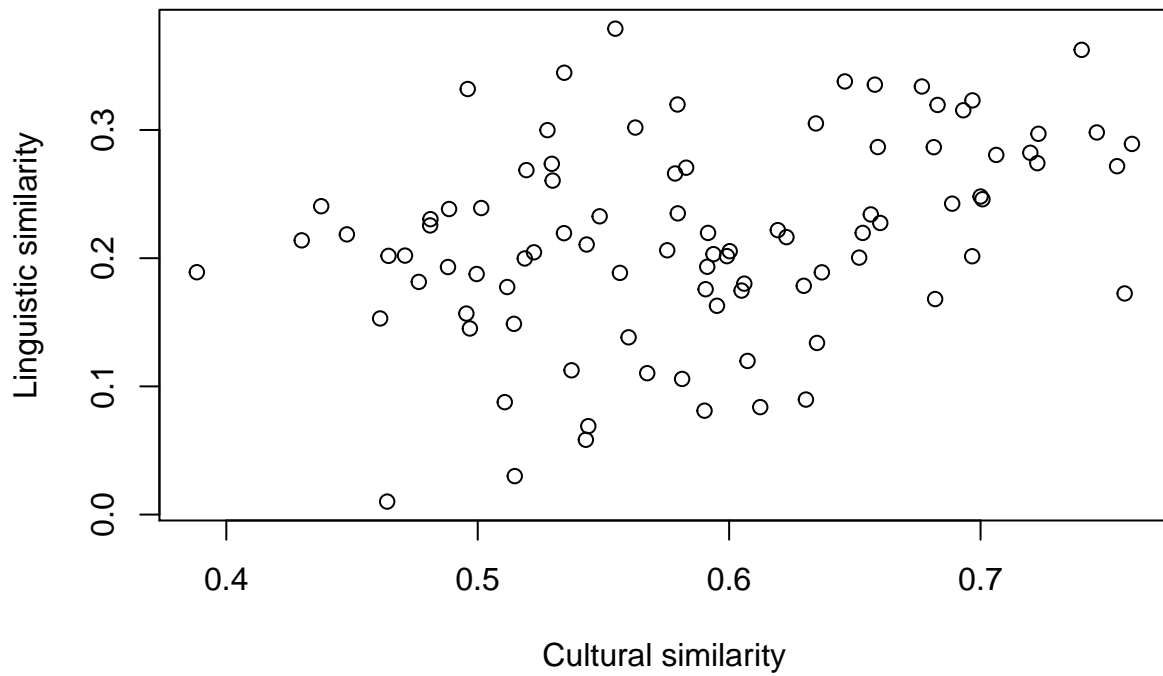
```
grph <- graph.data.frame(ling[,c("l1", 'l2', 'local_alignment')], directed=FALSE)
# add value as a weight attribute
ling.m = get.adjacency(grph, attr="local_alignment", sparse=FALSE)
rownames(ling.m) = 1[match(rownames(ling.m), l$iso2), l$Language2]
colnames(ling.m) = 1[match(colnames(ling.m), l$iso2), l$Language2]
# For missing comparisons, impute the mean:
ling.m[ling.m==0] = mean(ling$local_alignment)
diag(ling.m) = 0
```

Match the distance matrices

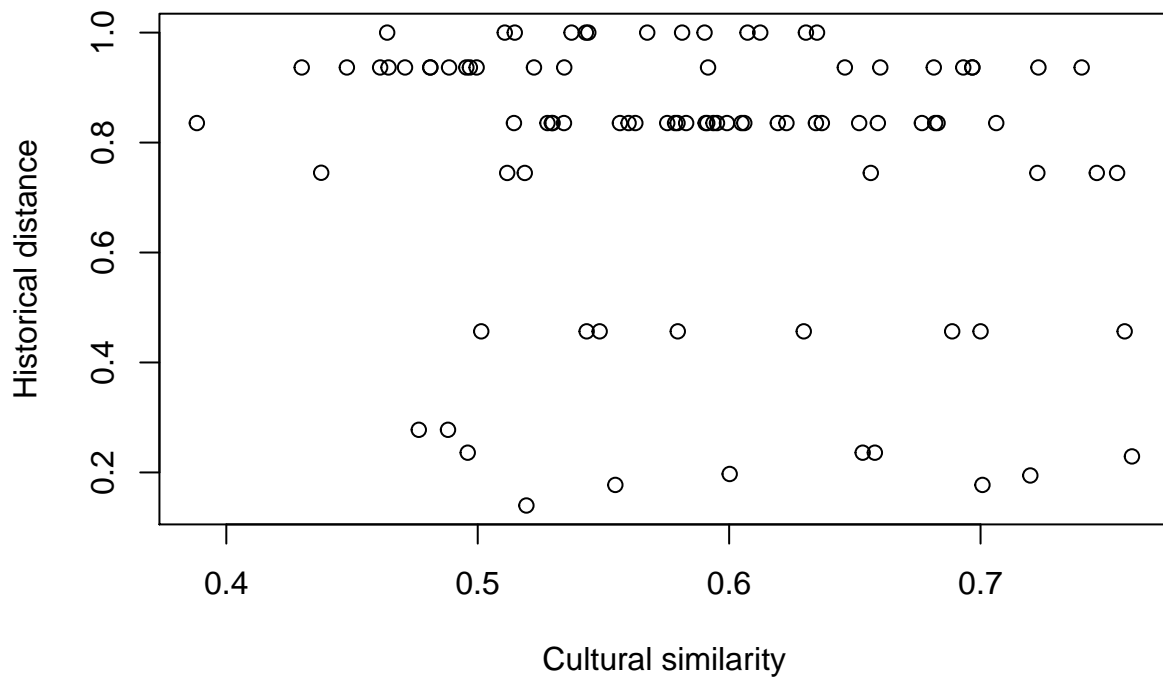
```
in.analysis = intersect(rownames(ling.m), rownames(cult.m))
in.analysis = intersect(in.analysis, rownames(hist.m))
cult.m2 = cult.m[in.analysis, in.analysis]
ling.m2 = ling.m[in.analysis, in.analysis]
hist.m2 = hist.m[in.analysis, in.analysis]
geo.m2 = geoDist.m[in.analysis, in.analysis]
```

Note that there are only 14 languages with data on linguistic, cultural and historical distance. This is because the historical distances are derived from a tree of Indo-European languages (there are currently no reliable phylogenetic trees constructed from cognates that span different language families). The languages in this test include: Albanian, Armenian, Bulgarian, Czech, Dutch, English, French, Greek, Icelandic, Latvian, Lithuanian, Russian, Spanish, Ukrainian.

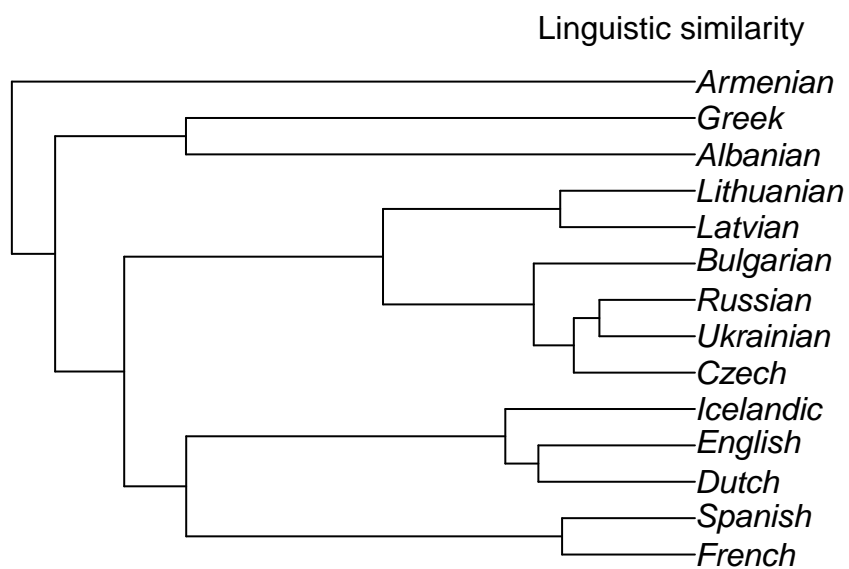
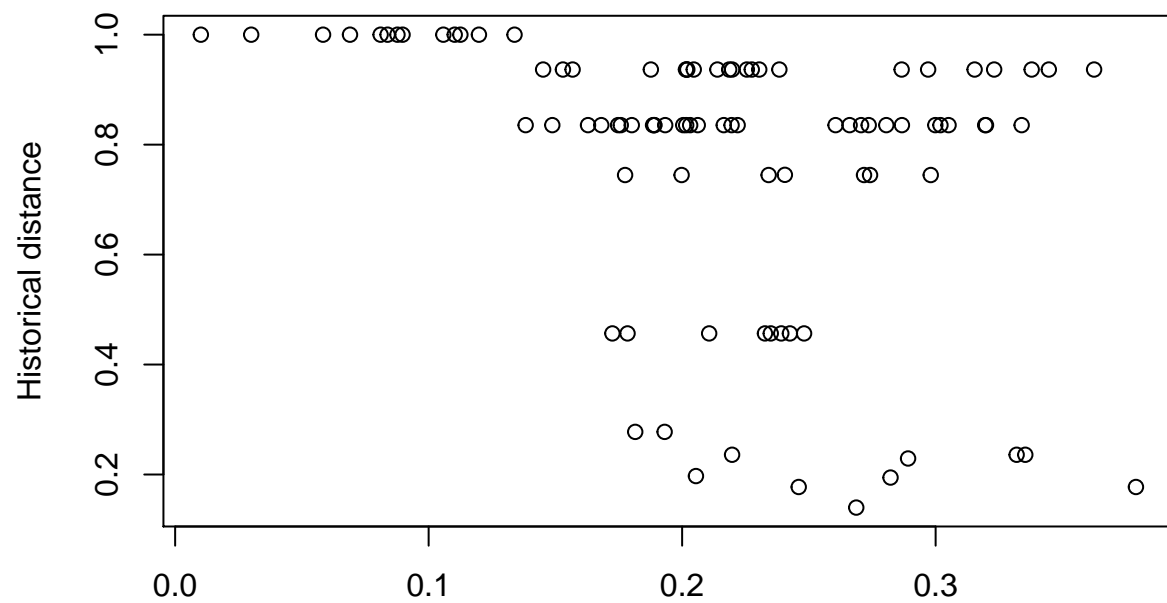
```
plot(as.dist(cult.m2), as.dist(ling.m2),
     xlab="Cultural similarity",
     ylab="Linguistic similarity")
```



```
plot(as.dist(cult.m2),as.dist(hist.m2),
     xlab="Cultural similarity",
     ylab="Historical distance")
```



```
plot(as.dist(ling.m2),as.dist(hist.m2),
     xlab="Linguistic similarity",
     ylab="Historical distance")
```



## Tests

The results of the test list the following measures:

- mantelr: Mantel correlation coefficient.
- pval1: one-tailed p-value (null hypothesis:  $r \leq 0$ ).
- pval2: one-tailed p-value (null hypothesis:  $r \geq 0$ ).
- pval3: two-tailed p-value (null hypothesis:  $r = 0$ ).
- llim: lower confidence limit for  $r$ .
- ulim: upper confidence limit for  $r$ .

```
set.seed(1498)
```

Run tests between each pair of measures.

```
distms = list("Cultural"= cult.m2,
              "Linguistic" = ling.m2,
              "Historical" = hist.m2,
              "Geographic" = geo.m2)

mantelRes1 = data.frame(
  Var1 = NA, Var2 = NA, r = NA,
  llim = NA, ulim = NA, p = NA,
  stringsAsFactors = F)

for(i in 1:3){
  for(j in (i+1):4){
    var1 = names(distms)[i]
    var2 = names(distms)[j]
    print(paste("Correlation between",
               var1,"and",var2))
    stat = ecodist::mantel(as.dist(distms[[i]]),
                          as.dist(distms[[j]]),
                          nperm = 100000)

    print(stat)
    mantelRes1 = rbind(mantelRes1,
                       c(var1,var2,stat[1],stat[5],stat[6],
                         min(c(stat[2],stat[3]))))
    stat = round(stat,2)
    stat2 = sprintf("$r$ = %s[%s,%s], one-tailed $p$ = %s",
                    stat[1],
                    stat[5],
                    stat[6],
                    min(c(stat[2],stat[3])))
    # TODO: output stats
    #cat(stat2,file=
    #    paste0("../results/stats/tex/Mantel",var1,"Vs",var2,"Distance_SUBS.tex"))
  }
}

## [1] "Correlation between Cultural and Linguistic"
##   mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## 0.3509606 0.0883600 0.9116500 0.1382200 0.2413895 0.5377689
## [1] "Correlation between Cultural and Historical"
##   mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## -0.16966425 0.84537000 0.15464000 0.30858000 -0.28569297 -0.01526378
## [1] "Correlation between Cultural and Geographic"
```

```
##      mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## -0.3397939  0.9723900  0.0276200  0.0308800 -0.5704463 -0.1718367
## [1] "Correlation between Linguistic and Historical"
##      mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## -0.35203243  0.96988000  0.03013000  0.03453000 -0.50321827 -0.08218648
## [1] "Correlation between Linguistic and Geographic"
##      mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## -0.2734487  0.9193900  0.0806200  0.1277600 -0.4617151 -0.0818242
## [1] "Correlation between Historical and Geographic"
##      mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
##  0.3457543  0.0100300  0.9899800  0.0100300  0.1811727  0.5217647

mantelRes1= mantelRes1[2:nrow(mantelRes1),]
mantelRes1[,3:6] = apply(mantelRes1[,3:6],2,function(X){
  signif(as.numeric(X),3)
})
```

Run a mantel test comparing the Linguistic alignment to the cultural similarity, controlling for the historical distance between languages:

```
ecodist::mantel(as.dist(ling.m2)~
  as.dist(cult.m2) +
  as.dist(hist.m2),
  nperm = 100000)
```

```
##      mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
##  0.3157282  0.1119100  0.8881000  0.1894500  0.1495754  0.5004242
```

*Main Test:* Run a mantel test comparing the Linguistic alignment to the cultural similarity, controlling for the historical distance and geographic distance between languages:

```
mainMantel = ecodist::mantel(as.dist(ling.m2)~
  as.dist(cult.m2) +
  as.dist(hist.m2) +
  as.dist(geo.m2),
  nperm = 100000)
mainMantel = signif(mainMantel,3)

mantelRes1 = rbind(mantelRes1,
  c("Linguistic", "Cultural **",
    mainMantel[1], mainMantel[5], mainMantel[6],
    min(mainMantel[2:3])))

mantelRes1Text = xtable(mantelRes1,
  caption = paste0(
    "Mantel tests (",
    datasetLabel,
    "). ** = partial Mantel test, controlling for historical and geographical distance."))
print(mantelRes1Text,
  file="../results/stats/tex/Mantel_SUBS.tex")
```

## MRM

Perform the main test, but using multiple regression on distance matrices (MRM).

```
mainMRM = ecodist::MRM(as.dist(ling.m2)~
  as.dist(cult.m2) +
```



```

as.dist(hist.m2) +
as.dist(geo.m2), nperm=10000)
mainMRM

## $coef
##           as.dist(ling.m2)    pval
## Int           0.14290281 0.6955
## as.dist(cult.m2)      0.24395908 0.2398
## as.dist(hist.m2)     -0.08531160 0.1408
## as.dist(geo.m2)      -0.02225994 0.6608
##
## $r.squared
##      R2      pval
## 0.2169477 0.1326000
##
## $F.test
##      F      F.pval
## 8.034563 0.132600

mainMRM2 = sprintf("$\\beta= %s, $p=%s",
                    round(mainMRM$coef[2,1],2),
                    round(mainMRM$coef[2,2],2))
cat(mainMRM2,
    file="../results/stats/tex/MRMCulturalVsLinguisticDistance_Partial_SUBS.tex")

```