

Sources of convergence in indigenous languages: Lexical variation in Yucatec Maya

Supplementary Material 3: Data processing

Barbara Blaha Pfeiler, Stavros Skopeteas

March 18, 2022

Contents

1	Global	2
1.1	Libraries	2
1.2	Global settings	3
2	Map functions	3
2.1	General	3
2.2	Map of Yucatán	3
2.3	Categorical features	4
2.4	Gradient features	5
2.5	MDS values to RGB colours	5
2.6	Features coded on a single RGB color dimension	6
2.7	GAM coefficients of space	6
3	Data processing functions	7
3.1	Matrix of dummy coded variants	7
3.2	Reducing dimensions of a distance matrix with MDS	7
3.3	Logistic regression	8
4	Data	9
4.1	Demographic data (general)	9
4.2	Speakers and Locations (in the sample)	11
4.3	Linguistic data	15
5	Predictions	17

6 Spatial diffusion	18
6.1 Descriptives	18
6.2 Plot features	19
6.3 Indigenous Variants	19
6.4 Spanish Variants	30
7 Sources	33
7.1 References	33
7.2 R and package versions	33

1 Global

1.1 Libraries

io

```
1 library(readxl)
2 library(writexl)
```

data processing

```
1 library(reshape2)
2 library(varhandle)
3 library(dplyr)
```

graphs

```
1 library(ggplot2)
2 library(ggmap)
3 library(pdp)
```

geography

```
1 library(gmt)
2 library(sp)
3 library(geosphere)
```

statistics

```
1 library(MASS)
2 library(cluster)
3 library(psych)
4 library(scales)
5 library(regclass)
6 library(lme4)
7 library(mgcv)
8 library(itsadug)
```

1.2 Global settings

Setting active directory

```
1 setwd(dirname(rstudioapi::getActiveDocumentContext())$path))
```

Global graph settings

```
1 theme_set(theme_bw())
```

Setting the seed to a fixed value in order to control jitter values

```
1 myseed = 2000
```

2 Map functions

2.1 General

2.1.1 Default settings

```
1 jitter <- position_jitter(width = 0.05, height = 0.05)
2 colors = c("white","black","grey60","white","black","grey60","white","black","grey60")
3 shapes = c(21,21,21,24,24,24,22,22,22)
4 gglegend = theme(
5   legend.position = c(0.002, 0.002),
6   legend.justification = c("left", "bottom"),
7   legend.box.just = "right",
8   legend.box.background = element_rect(color="grey", size=.5)
9 )
```

2.1.2 Geographical data

The .rds files are downloaded at [1] and saved in a directory “data” under the root directory.

```
1 gadm.MEX <- readRDS("data/gadm36_MEX_1_sf.rds")
2 gadm.BLZ <- readRDS("data/gadm36_BLZ_1_sf.rds")
3 gadm <- rbind(gadm.MEX,gadm.BLZ)
```

2.2 Map of Yucatán

basic map, geographical data

```
1 yuc.map.bare <- ggplot(data = gadm) +
2   geom_sf(fill = "grey99",color = "grey20",size = 0.3) +
3   coord_sf(xlim = c(-91, -86.7), ylim = c(18.1, 21.8), expand = T)+
4   scale_x_continuous(labels = abs,
5                       breaks = seq(-180,180,by=1),
```

```

6         minor_breaks = seq(-180,0,by=.1))+
7     scale_y_continuous(labels = abs,
8         breaks = seq(0, 100, by=1),
9         minor_breaks = seq(0,100,by=0.2))+
10     labs(x="Longitude (°W)",y="Latitude (°N)")

```

Labeled maps:

- labels for states
- labels for reported dialectal areas
- labels for city names

```

1 coord <- read_excel("S2_data.xlsx", sheet = "maplabels")
2 states <- droplevels(subset(coord,class=="states"))
3 states2 <- droplevels(subset(coord,class=="states2"))
4 areas <- droplevels(subset(coord,class=="areas"))
5 cities <- droplevels(subset(coord,class=="cities" & type == "text"))
6 cities2 <- droplevels(subset(coord,class=="cities2" & type == "text"))
7 cities.sgm <- droplevels(subset(coord,class=="cities" & type == "segment"))
8 cities.sgm2 <- droplevels(subset(coord,class=="cities2" & type == "segment"))
9 yuc.map.states <- yuc.map.bare+
10     annotate("text",x=states$x,y=states$y, label = states$label,size=3)
11 yuc.map.areas <- yuc.map.states+
12     annotate("text",x=areas$x,y=areas$y, label = areas$label,size=3)
13 yuc.map.cities <- yuc.map.states+
14     annotate("text",x=cities$x,y=cities$y, label = cities$label,size=3)+
15     annotate("segment",x=cities.sgm$x,y=cities.sgm$y,
16         xend=cities.sgm$xend, yend=cities.sgm$yend,size=.05)

```

2.3 Categorical features

The “plot.map” function visualizes categorical features (e.g., variants) on the map of Yucatán.

```

1 plot.map <- function(data.subset,feature)
2 {
3     data.subset <- droplevels(subset(data.subset,Concept_ID == feature))
4     data.subset <- droplevels(subset(data.subset,Variant != "N.V."))
5     Variant.freq <- data.frame(xtabs(~Variant,data.subset))
6     Variant.freq <- Variant.freq[order(-Variant.freq$Freq),]
7     Variant.freq$labels <- paste0(Variant.freq$Variant," (n=", Variant.freq$Freq,")")
8     Variant.sum <- sum(Variant.freq$Freq)
9     colnames(Variant.freq)[2] <- "Variant.freq"
10    data.subset <- merge(data.subset,Variant.freq,by="Variant",all.y = T)
11    data.subset$Variant <- factor(data.subset$Variant,
12        levels=factor(Variant.freq$Variant))
13    title <- concepts[concepts$Concept_ID == feature,]$Label
14    set.seed(myseed)
15    geo.plot <- yuc.map.areas+
16        geom_point(data = data.subset,
17            aes(x = Longitude, y = Latitude, fill=Variant,shape=Variant),
18            size=3,

```



```

19         alpha=.7,
20         position=jitter)+
21     scale_fill_manual(values=colors, labels=Variant.freq$labels)+
22     scale_shape_manual(values=shapes, labels=Variant.freq$labels)+
23     labs(fill = element_blank(),shape = element_blank())+
24     gglegend+
25     ggtitle(label=title)+
26     theme(plot.title = element_text(size = 11))
27 png(paste0("figs/map_",feature,".png"),width =3000, height=3000, res=600)
28 print(ggplot)
29 dev.off()
30 }

```

2.4 Gradient features

The “plot.map.gradient” function visualizes gradient features (e.g., frequencies of variants) on the map of Yucatán.

```

1 plot.map.gradient <- function(data,legend,midpoint)
2 {
3     caption <- substitute(data)
4     plot.gradient <- yuc.map.areas +
5         geom_point(data = data,
6                     aes(x = Longitude, y = Latitude,fill=Variant.p),
7                     size=3,
8                     shape=21,
9                     alpha=.9,
10                    colour="black",
11                    position=jitter)+
12     scale_fill_gradient2(low="white",
13                         midpoint=midpoint,
14                         high="black",
15                         mid="grey99",
16                         guide = guide_colourbar(direction = "horizontal",
17                                                  frame.colour="black",
18                                                  barwidth = 4,
19                                                  barheight = 1),
20                         breaks=c(0,midpoint,1),
21                         name=legend)+
22     gglegend
23 png(paste0("figs/mapgr_",caption,".png", sep=""),width =3000, height=3000, res=600)
24 print(plot.gradient)
25 dev.off()
26 }

```

2.5 MDS values to RGB colours

The “plot.mds” function visualizes mds dimensions in RGB colors on the map of Yucatán.

```

1 plot.mds <- function(mds,type,map) {
2     name <- substitute(mds)

```

```

3  set.seed(myseed)
4  if(type == "location"){
5    mds.ggplot <- map+
6      geom_point(data = mds, aes(x = as.numeric(Longitude),
7                                y = as.numeric(Latitude),
8                                size=log2(as.numeric(Size)),
9                                fill=I(rgb)),shape=21,alpha=.8,position=jitter)+
10     theme(axis.title=element_text(size=12))+
11     guides(size = "none",fill="none")+
12     scale_size(range = c(1,7))
13   }
14   else{
15     mds.ggplot <- map+
16       geom_point(data = mds, aes(x = as.numeric(Longitude),
17                                 y = as.numeric(Latitude),
18                                 fill=I(rgb)),size=3,shape=21,alpha=.8,position=jitter)+
19     theme(axis.title=element_text(size=12))+
20     guides(size = 'none',fill='none')
21   }
22   png(paste0("figs/mds_",name,".png"),width =3000, height=3000, res=600)
23   print(mds.ggplot)
24   dev.off()
25 }

```

2.6 Features coded on a single RGB color dimension

The “plot.mds.dim” function visualizes data coded in one of the RGB color dimensions on the map of Yucatán.

```

1  plot.mds.dim <- function(fit,variable,color,label,caption){
2    set.seed(myseed)
3    plot.dim <- yuc.map.bare +
4      geom_point(data = fit, aes(x = Longitude, y = Latitude, fill=I(variable)),
5                        size=3,shape=21,alpha=.9,position=jitter)+
6      ggtitle(label=label)+
7      theme(legend.position = "none")
8    png(paste0("figs/mds_",caption,".png"),width =2000, height=2000, res=600)
9    print(plot.dim)
10   dev.off()
11 }

```

2.7 GAM coefficients of space

The “plot.gam” function plots the GAM coefficients to the map of Yucatán.

```

1  plot.gam <- function(model) {
2    caption = substitute(model)
3    png(paste0("figs/gam_",caption,".png", sep=""),width =3000, height=3000, res=500)
4    vis.gam(model,plot.type="contour", color="terrain",too.far=0.15,
5            view=c("Longitude","Latitude"),
6            main="",cex.axis=0.9, cex.lab=1.0)

```

```

7     text(-89.6,21.2, label = "Metropolitan",cex=1)
8     text(-90.2,19.95, labels = "Camino\nReal", xpd=TRUE,cex=1)
9     text(-88,21.2, labels = "Northeast", xpd=TRUE,cex=1)
10    text(-88.7,20.8, labels = "Agave fields", xpd=TRUE,cex=1)
11    text(-89.8,19.2, labels = "Los Chenes", xpd=TRUE,cex=1)
12    text(-89.1,20, labels = "Center", xpd=TRUE,cex=1)
13    text(-88.3,18.5, labels = "Belize", xpd=TRUE,cex=1)
14    dev.off()
15 }

```

3 Data processing functions

3.1 Matrix of dummy coded variants

The “feature2dummy” function converts a variable with a set of values to a dummy-coded matrix.

```

1 feature2dummy <- function(data.set,feature.set) {
2   feature.dta <- data.frame(Speaker_ID = levels(factor(data.set$Speaker_ID)))
3   rownames(feature.dta) <- feature.dta$Speaker_ID
4   for(i in 1:length(feature.set)) {
5     feature.subset <- data.set[data.set$Concept_ID == feature.set[i],]
6     rownames(feature.subset) <- feature.subset$Speaker_ID
7     if (length(levels(factor(feature.subset$Variant))) > 1) {
8       feature.dummy <- to.dummy(feature.subset$Variant, feature.set[i])
9       feature.dummy.2 <- cbind(rownames(feature.subset),feature.dummy)
10      feature.dta <- cbind(feature.dta,
11                           feature.dummy.2[match(feature.dta$Speaker_ID,
12                                                  rownames(feature.subset)),
13                                           colnames(feature.dummy)])
14    }
15    feature.dta[is.na(feature.dta)] <- 0
16  }
17  feature.dta <- subset(feature.dta,select=-c(Speaker_ID))
18  dummy.matrix <- feature.dta
19 }

```

3.2 Reducing dimensions of a distance matrix with MDS

The “dist2mds” function:

- applies MDS (3-dimensional) to a distance matrix,
- rescales the mds dimensions to a [0,1] scale,
- plots the three mds dimensions to the RGB color dimensions,
- reports the r-square by comparing the input distances with the output (mds) distances; see [2].

```

1 dist2mds <- function(dm,coordinates){
2   caption <- substitute(dm)
3   mds.all <- cmdscale(as.dist(dm),eig=TRUE, k=3,x.ret = T)
4   mds <- data.frame(mds.all$points)
5   dist.output <- dist(mds.all$points, diag=TRUE, upper=TRUE)

```

```

6      r <- cor(c(as.dist(dm)), c(dist.output))
7      rsquared <- r * r
8      mds$d1 <- scales::rescale(mds$X1,to=c(0,1))
9      dist.output.1 <- dist(mds$X1, diag=TRUE, upper=TRUE)
10     r1 <- cor(c(as.dist(dm)), c(dist.output.1))
11     rsquared1 <- r1 * r1
12     mds$d2 <- scales::rescale(mds$X2,to=c(0,1))
13     dist.output.2 <- dist(mds$X2, diag=TRUE, upper=TRUE)
14     r2 <- cor(c(as.dist(dm)), c(dist.output.2))
15     rsquared2 <- r2 * r2
16     mds$d3 <- scales::rescale(mds$X3,to=c(0,1))
17     dist.output.3 <- dist(mds$X3, diag=TRUE, upper=TRUE)
18     r3 <- cor(c(as.dist(dm)), c(dist.output.3))
19     rsquared3 <- r3 * r3
20     mds$rgb <- rgb(r = mds$d1,g = mds$d2,b = 1-mds$d3)
21     mds$Longitude <- coordinates[match(rownames(mds),
22                                     rownames(coordinates)), "Longitude"]
23     mds$Latitude <- coordinates[match(rownames(mds),
24                                     rownames(coordinates)), "Latitude"]
25     mds$Location <- coordinates[match(rownames(mds),
26                                     rownames(coordinates)), "Location"]
27     mds$Size <- coordinates[match(rownames(mds),
28                                     rownames(coordinates)), "Pop.size"]
29     mds <- mds
30 }

```

3.3 Logistic regression

Examining the relation between Variants in the data (dependent variable) and mds-dimensions (independent variable) in logistic regressions, output in an excel table

```

1 mds2regression <- function(fit,data){
2   title <- substitute(fit)
3   mds.corr.table <- data.frame(variable = character(),
4                               d1.b = numeric(),d1.z = numeric(),d1.p = numeric(),
5                               d2.b = numeric(),d2.z = numeric(),d2.p = numeric(),
6                               d3.b = numeric(),d3.z = numeric(),d3.p = numeric())
7   data.fit <- cbind(data,fit)
8   for (i in 1:(ncol(data)))
9   {
10     variable <- colnames(data.fit)[i]
11     fit.frame <- data.frame(xtabs(~data.fit[,i],data.fit))
12     min.obs = 1
13     if((fit.frame[1,]$Freq > min.obs)&(fit.frame[2,]$Freq > min.obs))
14     {
15       variable.glm <- summary(glm(as.factor(data.fit[,i])~
16                               d1+d2+d3,family="binomial",
17                               data.fit))
18       d1.b <- round(variable.glm$coef[2],digits=3)
19       d2.b <- round(variable.glm$coef[3],digits=3)
20       d3.b <- round(variable.glm$coef[4],digits=3)
21       d1.z <- round(variable.glm$coef[10],digits=3)

```

```

22     d2.z <- round(variable.glm$coef[11],digits=3)
23     d3.z <- round(variable.glm$coef[12],digits=3)
24     d1.p <- round(variable.glm$coef[14],digits=3)
25     d2.p <- round(variable.glm$coef[15],digits=3)
26     d3.p <- round(variable.glm$coef[16],digits=3)
27     mds.corr.table.row <- data.frame(variable, d1.b,d1.z,d1.p,
28                                     d2.b,d2.z,d2.p,
29                                     d3.b,d3.z,d3.p)
30     mds.corr.table <- rbind(mds.corr.table,mds.corr.table.row)
31   }
32 }
33 write_xlsx(mds.corr.table, paste0("mds_logregr_",title,".xlsx"))
34 }

```

4 Data

The data used in this study are available in the excel file “S2.data.xlsx”. Information about the contents of sheets and columns is found in the excel sheet “notes”.

```

1   info <- read_excel("S2_data.xlsx", sheet = "notes")[1:3]

```

4.1 Demographic data (general)

In order to create the demographic maps, you will need to download the excel sheets with population statistics from the Mexican States of Yucatán (Campeche, CAM, Yucatán, YUC, Quintana Roo, QR) [3].

The analyses use the following information, see [4]:

- POBTOT: total population (of the location at issue)
- P3YM_HLI: persons (3-130 years old) speaking an indigenous language

Procedure:

- bind the data from all states in a single dataframe
- drop lines without longitude or without information about the indigenous speakers
- convert to GPS data
- calculate proportion of population speaking indigenous languages out of population total per location.

```

1   YUC.ddta.2 <- rbind(data.frame(read_excel("data/ITER_04XLS10 CAM.xls")),
2                       data.frame(read_excel("data/ITER_31XLS10 YUC.xls")),
3                       data.frame(read_excel("data/ITER_23XLS10 QR.xls")))
4   YUC.ddta <- subset(YUC.ddta.2,select=c("NOM_MUN", "NOM_LOC",
5                                         "LONGITUD", "LATITUD", "POBTOT", "P3YM_HLI"))
6   YUC.ddta <- droplevels(subset(YUC.ddta, (LONGITUD != "") & (P3YM_HLI != "*")))
7   YUC.ddta$LONGITUD.GPS <- as.numeric(char2dms(paste0(substr(YUC.ddta$LONGITUD, 1, 3),
8                                                         "d", substr(YUC.ddta$LONGITUD, 4, 5), "\'",
9                                                         chs = substr(YUC.ddta$LONGITUD, 6, 7), "\"W\"")))
10  YUC.ddta$LATITUD.GPS <- as.numeric(char2dms(paste0(substr(YUC.ddta$LATITUD, 1, 2),
11                                                         "d", substr(YUC.ddta$LATITUD, 3, 4), "\'",
12                                                         chs = substr(YUC.ddta$LATITUD, 5, 6), "\"N\"")))

```

```

13 YUC.ddta$POBTOT.NUM <- as.numeric(YUC.ddta$POBTOT)
14 YUC.ddta$P3YM_HLI.NUM <- as.numeric(YUC.ddta$P3YM_HLI)
15 YUC.ddta$PROP.MAYA <- YUC.ddta$P3YM_HLI.NUM/YUC.ddta$POBTOT.NUM
16 YUC.ddta <- droplevels(subset(YUC.ddta, is.na(POBTOT.NUM) == F))

1 demog.ggplot <- yuc.map.bare+
2   theme(panel.background = element_blank(),
3         panel.border = element_rect(colour = "grey60", fill=NA, size=0.5))+
4   geom_point(data = YUC.ddta,
5             aes(x = LONGITUD.GPS,
6                 y = LATITUD.GPS,
7                 fill=100*PROP.MAYA,
8                 size=log(POBTOT.NUM)),
9             shape=21,
10            color="grey30",
11            alpha=1)+
12   scale_fill_gradient2('% speakers of indig. lgs',
13                       low = "white",
14                       high = "blue",
15                       mid="lightblue",
16                       na.value = "grey50",
17                       midpoint=50,
18                       guide = guide_legend(
19                         direction = "horizontal",
20                         title.position = "top",
21                         label.position = "bottom",
22                         label.hjust = 0.5,
23                         label.vjust = 0))+
24   scale_size(range = c(.1,3))+
25   guides(size = 'none')+
26   theme(axis.title=element_text(size=12))+
27   theme(plot.title=element_text(size=20))+
28   annotate("text",x=states2$x,y=states2$y, label = states2$label,size=5)+
29   annotate("text",x=cities2$x,y=cities2$y, label = cities2$label,size=3.5)+
30   annotate("segment",x=cities.sgm2$x,y=cities.sgm2$y,
31          xend=cities.sgm2$xend, yend=cities.sgm2$yend,size=.6)+
32   theme(
33     legend.position = c(0.002, 0.002),
34     legend.justification = c("left", "bottom"),
35     legend.box.just = "right",
36     legend.box.background = element_rect(color="grey", size=.5)
37   )
38
39 png(paste0("figs/yuc.population.png"),width =3000, height=3000, res=600)
40 demog.ggplot
41 dev.off()

```

Population statistics

The data analysed below are extracted from the table “05_01B_ESTATAL.xls”, downloaded at [5].

- aggregated per age groups, starting with 10 years old, age groups of 5 years

- calculating percentages of bilingual and monolingual indigenous speakers out of the total speakers with language specification (excluding speakers that are unspecified for this feature)
- creating a plot with percentages of bilingual and monolingual indigenous speakers.

```

1 yuc.pop.states <- data.frame(read_excel("data/yuc.age.xlsx"))
2 yuc.pop <- aggregate(. ~ age, yuc.pop.states[2:8], FUN=sum)
3 yuc.pop$birth <- 2010 - as.numeric(as.character(substr(yuc.pop$age,1,2)))
4 yuc.pop$total <- yuc.pop$Indlang + yuc.pop$Spanish
5 yuc.pop$p.bilingual <- 100*yuc.pop$Indlang.Spanish / yuc.pop$total
6 yuc.pop$p.monolingual <- 100*yuc.pop$Indlang.NoSpanish / yuc.pop$total
7 yuc.age <- yuc.pop[,c("birth","p.bilingual","p.monolingual")]
8 yuc.age.m <- melt(yuc.age,id=c("birth"))
9
10 age.plot <- ggplot(yuc.age.m,
11                   aes(x=birth, y=value,group=variable,colour=variable)) +
12   ylim(0,100)+
13   xlim(1925,2000)+
14   stat_smooth(span = 0.2,se=F,aes(y=value),size=1) +
15   scale_colour_manual(name="indigenous speakers",
16                       values=c("black","grey"),labels=c("bilingual","monolingual"))+
17   labs(x="year of birth",y="% out of population total")+
18   theme(axis.text.x=element_text(angle=90, hjust=1, vjust=.5))+
19   theme(axis.title.x=element_text(size=13))+
20   theme_bw()+
21   theme(legend.position="top")
22 png(paste0("figs/yuc.age.png"),width =3000, height=3000, res=700)
23 age.plot
24 dev.off()

```

```

1 grid.arrange(demog.ggplot,age.plot, ncol = 2)

```

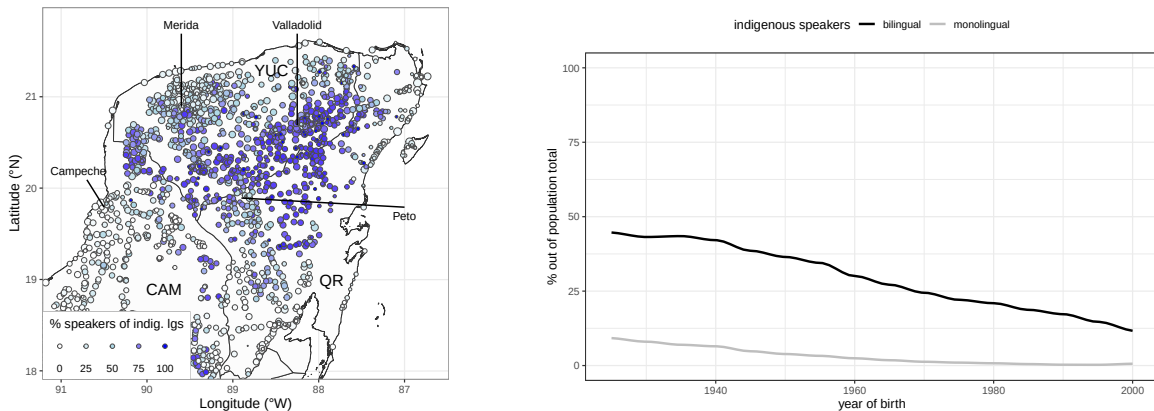


Figure 1: Proportions of indigenous speakers out of population total in the Yucatecan Peninsula (left panel), monolingual/bilingual speakers and year of birth (right panel)

4.2 Speakers and Locations (in the sample)

Dataframe speakers containing information about the speaker sample, with the following variables:

```
1 droplevels(subset(info,Sheet=="speakers"))
```

Sheet	Column	Content
speakers	Speaker_ID	unique identifier of speaker
speakers	Location	name of the speaker's location
speakers	Birthyear	year of birth of the speaker
speakers	Gender	male (M) or female (F)

Importing “speakers”, aggregating by location

```
1 speakers <- read_excel("S2_data.xlsx", sheet = "speakers",
2                       col_types = c("text","text","numeric","text"))
3 head(speakers)
```

Speaker_ID	Location	Birthyear	Gender
S001	San Narciso	1961	M
S002	San Pedro	1937	F
S003	Becal	1931	M
S004	Becal	1958	M
S005	Becal	1962	M
S006	Becal	1946	M

```
1 speaker.location <- data.frame(xtabs(~Location,speakers))
2 speaker.location <- speaker.location[order(speaker.location$Freq),]
```

The dataframe “locations” contains a list of locations, with geographic and demographic information.

```
1 droplevels(subset(info,Sheet=="locations"))
```

Sheet	Column	Content
locations	Location_ID	unique identifier of the sample location
locations	Location	name of location
locations	Latitude	Latitude of the location (based on googlemaps)
locations	Longitude	Longitude of the location (based on googlemaps)
locations	Pop.size	Population size of the location, based on the available census data
locations	Ind.size	Size of the indigenous population of the location

Importing locations, calculating the proportion of indigenous population out of population total.

```
1 locations <- read_excel("S2_data.xlsx", sheet = "locations",
2                       col_types = c("text","text","numeric","numeric","numeric","numeric"))
3 locations$Ind.prop <- 100*locations$Ind.size/locations$Pop.size
4 locations$Pop.size.log <- log2(as.numeric(locations$Pop.size/1000))
5 head(locations)
```


Location_ID	Location	Latitude	Longitude	Pop.size	Ind.size	Ind.prop	Pop.size.log
L001	San Pedro	18.33950	-88.49521	11746	587	4.997446	3.5540976
L002	San Narciso	18.30497	-88.52152	2423	484	19.975237	1.2767944
L003	Pachacan	18.40316	-88.47325	1373	274	19.956300	0.4573316
L004	Chan Ch'en	18.44052	-88.42751	715	143	20.000000	-0.4839849
L005	Pach Huitz	19.14901	-89.24913	266	230	86.466165	-1.9105018
L006	Xmaben	19.23579	-89.31440	1228	1025	83.469055	0.2963106

Testing distributions

```

1 size.density <- ggplot(locations, aes(x = Pop.size)) +
2   xlab("population size") +
3   geom_density()
4 size.log.density <- ggplot(locations, aes(x = log2(Pop.size))) +
5   xlab("logarithmized population size") +
6   geom_density()
7 grid.arrange(size.density,size.log.density, ncol = 2)

```

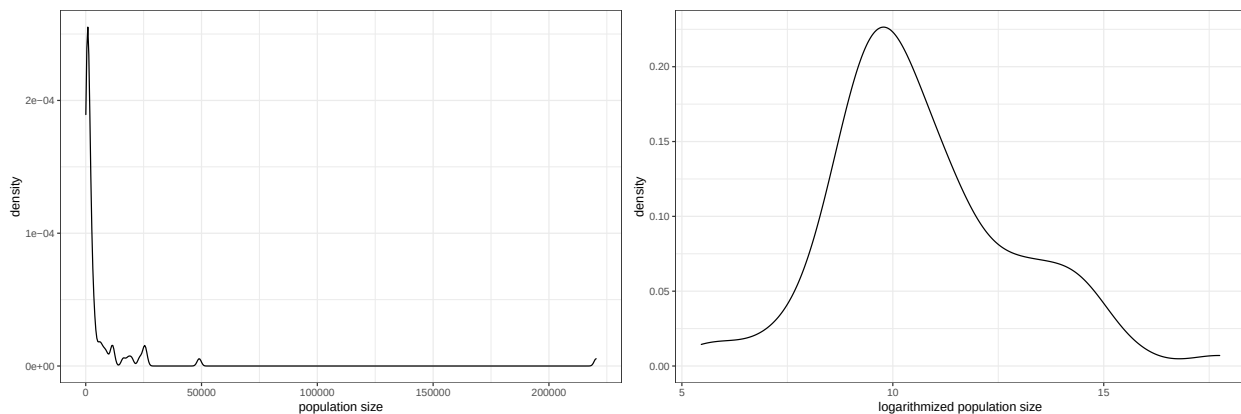


Figure 2: Density of population sizes (left panel) and of logarithmized population sizes (right panel) of the sample locations

```

1 png(paste0("figs/size.density.png"),width =1600, height=800, res=300)
2   print(size.density)
3   dev.off()
4 png(paste0("figs/size.log.density.png"),width =1600, height=800, res=300)
5   print(size.log.density)
6   dev.off()

```

Population size and indigenous proportions

```

1 print(corr.test(locations$Pop.size.log,locations$Ind.prop),short=FALSE)

## Call:corr.test(x = locations$Pop.size.log, y = locations$Ind.prop)
## Correlation matrix
## [1] -0.52
## Sample Size

```

```
## [1] 80
## These are the unadjusted probability values.
## The probability values adjusted for multiple tests are in the p.adj object.
## [1] 0
##
## Confidence intervals based upon normal theory. To get bootstrapped values, try cor.ci
##      raw.lower raw.r raw.upper raw.p lower.adj upper.adj
## NA-NA      -0.67 -0.52      -0.34      0      -0.67      -0.34
```

Speakers and Locations

```
1  locations <- merge(locations,speaker.location,by="Location",all.y = T)
2  rownames(locations) <- locations$Location
3  speaker.dta <- merge(speakers,locations, by ="Location",all.x=T)
4  rownames(speaker.dta) <- speaker.dta$Speaker_ID
5
6  speakers.summaries <- c("n speakers","in n locations",
7      "birthyear range","birthyear average","birthyear median",
8      "n women","n men")
9  result <- c(nrow(speakers),length(unique(speakers$Location)),
10      paste(min(speakers$Birthyear),"-",max(speakers$Birthyear)),
11      round(mean(speakers$Birthyear),1),median(speakers$Birthyear),
12      nrow(speakers[speakers$Gender == "F",]),
13      nrow(speakers[speakers$Gender == "M",]))
14  data.frame(speakers.summaries, result)
```

speakers.summaries	result
n speakers	157
in n locations	80
birthyear range	1906 - 1989
birthyear average	1953.7
birthyear median	1953
n women	53
n men	104

Population size and indigenous proportions

```
1  print(corr.test(speaker.dta$Birthyear,speaker.dta$Ind.prop),short=FALSE)

## Call:corr.test(x = speaker.dta$Birthyear, y = speaker.dta$Ind.prop)
## Correlation matrix
## [1] 0.05
## Sample Size
## [1] 157
## These are the unadjusted probability values.
## The probability values adjusted for multiple tests are in the p.adj object.
## [1] 0.5
##
## Confidence intervals based upon normal theory. To get bootstrapped values, try cor.ci
##      raw.lower raw.r raw.upper raw.p lower.adj upper.adj
## NA-NA      -0.1  0.05      0.21  0.5      -0.1      0.21
```

```

1 print(corr.test(speaker.dta$Birthyear,speaker.dta$Pop.size.log),short=FALSE)

## Call:corr.test(x = speaker.dta$Birthyear, y = speaker.dta$Pop.size.log)
## Correlation matrix
## [1] 0.14
## Sample Size
## [1] 157
## These are the unadjusted probability values.
## The probability values adjusted for multiple tests are in the p.adj object.
## [1] 0.07
##
## Confidence intervals based upon normal theory. To get bootstrapped values, try cor.ci
## raw.lower raw.r raw.upper raw.p lower.adj upper.adj
## NA-NA -0.01 0.14 0.29 0.07 -0.01 0.29

```

4.3 Linguistic data

Variants: list of elicited variants per concept, classified for language of origin

```

1 droplevels(subset(info,Sheet=="variants"))

```

Sheet	Column	Content
variants	Concept_ID	unique identifier of the sample concepts
variants	Variant	variant occurring as lexicalization of the concept at issue
variants	Language	origin of the variant: Maya or Spanish

```

1 variants <- data.frame(read_excel("S2_data.xlsx", sheet = "variants",
2                               col_types = c("text","text","text")))
3 head(variants)

```

Concept_ID	Variant	Language
C001	chúuj	Maya
C001	k'úum	Maya
C002	jool	Maya
C002	joom	Maya
C003	ki bok buuts'	Maya
C003	poom	Maya

Importing “forms”, containing the elicited tokens per speaker/concept

```

1 droplevels(subset(info,Sheet=="forms"))

```

Sheet	Column	Content
forms	Speaker_ID	speaker identifier
forms	Concept_ID	concept identifier

Sheet	Column	Content
forms	Variant	variant produced for the concept at issue by the speaker at issue

Creating a unique key for tokens occurring as responses to a specific Concept.

```

1 forms <- data.frame(read_excel("S2_data.xlsx", sheet = "forms",
2                             col_types = c("text","text","text")))
3 forms$Variant_ID <- paste0(forms$Concept_ID,forms$Variant)
4 variants$Variant_ID <- paste0(variants$Concept_ID,variants$Variant)
5 head(forms)

```

Speaker_ID	Concept_ID	Variant	Variant_ID
S001	C001	chúuj	C001chúuj
S001	C002	jool	C002jool
S001	C003	N.V.	C003N.V.
S001	C004	pool	C004pool
S001	C005	xéet'	C005xéet'
S001	C006	mixba'al	C006mixba'al

Adding basic biographical/demographic data to the “forms” dataframe.

```

1 forms$Language <- variants[match(forms$Variant_ID, variants$Variant_ID),"Language"]
2 speaker.dta2 <- subset(speaker.dta,select = c("Speaker_ID","Birthyear",
3                                             "Location","Latitude","Longitude",
4                                             "Pop.size.log","Ind.prop"))
5 forms <- merge(forms,speaker.dta2,by="Speaker_ID")

```

Concepts: list of concepts and the corresponding prompts in the questionnaire

```

1 droplevels(subset(info,Sheet=="concepts"))

```

Sheet	Column	Content
concepts	Concept_ID	unique identifier of the sample concepts
concepts	Concept	target concept
concepts	Prompt	Spanish prompt used for eliciting the concept
concepts	English	English translation of the Spanish prompt

Importing the data frame “concepts” from the excel table and creating labels (to be used for plots of features)

```

1 concepts <- data.frame(read_excel("S2_data.xlsx", sheet = "concepts"))
2 head(concepts)

```

Concept_ID	Concept	Prompt	English
C001	squash	calabazo	squash
C002	hole	agujero	hole

Concept_ID	Concept	Prompt	English
C003	incense	incienso	incense
C004	head	su cabeza	your head
C005	piece	pedazo de piedra	piece of stone
C006	nothing	nada	nothing

```

1  concepts$Label <- paste0("Concept: ",
2                             paste(concepts$Concept,"Prompt: ",sep="\n"),
3                             concepts$Prompt," ",
4                             concepts$English,"")

```

Data summary

```

1  data.summaries <- c("n variants","n forms","n concepts")
2  result <- c(nrow(variants),nrow(forms),nrow(concepts))
3  data.frame(data.summaries, result)

```

data.summaries	result
n variants	149
n forms	8164
n concepts	52

5 Predictions

Calculating distances between places: Dij is the distance between center i and center j - creating a matrix with the logarithmized distances between locations - removing the upper triangle and the diagonal of the matrix (to exclude duplicates and self distances) - reducing the distances with multidimensional scaling

```

1  Dij <- log2(distm(locations[,c("Longitude","Latitude")]))
2  rownames(Dij) <- locations$Location
3  colnames(Dij) <- locations$Location
4  Dij[upper.tri(Dij,diag=T)] <- NA
5  Dij.dist <- as.dist(scales::rescale(Dij,to=c(.001,.999)))
6  dist2mds(Dij.dist,locations); pred.by.distance <- mds
7  print(paste0("R-squared, all dimensions: ",round(rsquared,digits=2)))

```

```
## [1] "R-squared, all dimensions: 0.74"
```

Population sizes

- creating a vector with logarithmized population sizes (in thousands)
- calculating population size products
- rescaling population size products to a (0,1) interval
- centering population size products around 1
- since the input to mds is a “distance” matrix, the factors of the gravity model must be reversed: ratio of distances to population sizes
- reducing dimensions of variation of the matrix with multidimensional scaling

```

1 Pop.vector <- locations$Pop.size.log
2 PiPj <- Pop.vector %o% Pop.vector
3 rownames(PiPj) <- locations$Location
4 colnames(PiPj) <- locations$Location
5
6 PiPj.scaled <- rescale(PiPj,to=c(.001,.999))
7 PiPj.centered <- PiPj.scaled-mean(PiPj.scaled)+1
8 PiPj.dist <- as.dist(PiPj.centered)
9 gravity.dm <- Dij.dist/PiPj.dist
10
11 dist2mds(gravity.dm,locations); pred.by.gravity <- mds

1 print(paste0("R-squared, all dimensions: ",round(rsquared,digits=2)))

## [1] "R-squared, all dimensions: 0.65"

1 plot.mds(pred.by.distance,"location",yuc.map.cities); mds.distance <- mds.ggplot
2 plot.mds(pred.by.gravity,"location",yuc.map.cities); mds.gravity <- mds.ggplot

1 grid.arrange(mds.distance,mds.gravity, ncol = 2)

```

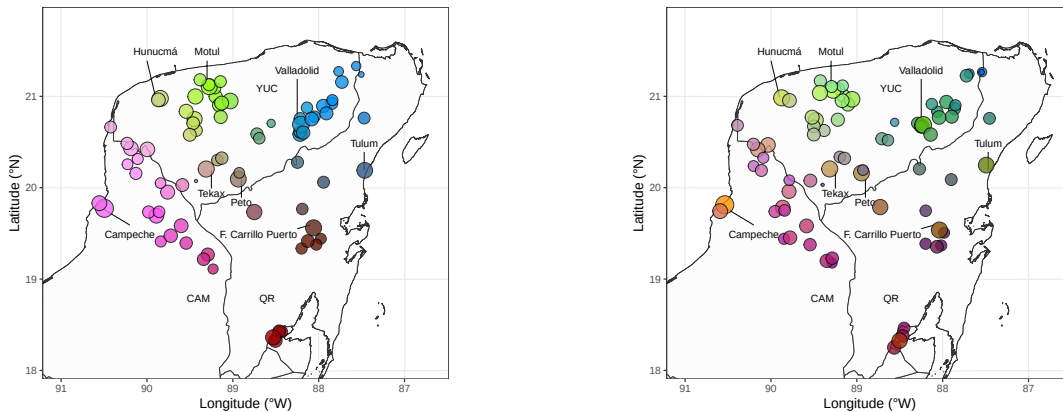


Figure 3: Predictions, wave model (left panel), gravity model (right panel)

6 Spatial diffusion

6.1 Descriptives

- creating a unique list of features occurring in the “form” data frame.
- removing non-valid values
- creating subsets with the variants of Mayan and Spanish origin
- calculating a speaker/cells matrix

```

1 features <- unique(forms$Concept_ID)
2 valid.dta <- droplevels(subset(forms,Variant != "N.V."))

```

```

3 maya.dta <- droplevels(subset(valid.dta,Language == "Maya"))
4 spanish.dta <- droplevels(subset(valid.dta,Language == "Spanish"))
5 speakers.xtabs <- data.frame(xtabs(~Speaker_ID,forms))

```

Data summary

```

1 indig.summaries <- c("n features",
2                     "n speakers",
3                     "n tokens",
4                     "n valid",
5                     "n non-valid",
6                     "% valid out of n tokens",
7                     "n Mayan",
8                     "% Mayan out of n valid",
9                     "n Spanish",
10                    "% Spanish out of n valid")
11 indig.results <- c(length(features),
12                  nrow(speakers.xtabs),
13                  nrow(forms),
14                  nrow(valid.dta),
15                  nrow(forms)-nrow(valid.dta),
16                  round((nrow(valid.dta)*100/nrow(forms)),digits=0),
17                  nrow(maya.dta),
18                  round((nrow(maya.dta)*100/nrow(valid.dta)),digits=0),
19                  nrow(spanish.dta),
20                  round((nrow(spanish.dta)*100/nrow(valid.dta)),digits=0))
21 data.frame(summaries = indig.summaries, results = as.character(indig.results))

```

summaries	results
n features	52
n speakers	157
n tokens	8164
n valid	6475
n non-valid	1689
% valid out of n tokens	79
n Mayan	6304
% Mayan out of n valid	97
n Spanish	171
% Spanish out of n valid	3

6.2 Plot features

```

1 for(i in 1:length(features)){
2   plot.map(forms,as.character(unique(forms$Concept_ID)[i]))}

```

6.3 Indigenous Variants

6.3.1 Lexical distances between speakers

Dummy coding of variants

```

1 feature2dummy(maya.dta,features)
2 data <- dummy.matrix
3 lex.dist <- daisy(data, metric = "gower",
4                   type = list(asymm = c(1:(length(data)))))

```

Reliability (Cronbach's alpha)

```

1 data.num <- data %>% mutate_all(funs(type.convert(as.factor(.))))
2 summary(psych::alpha(data.num,check.keys=TRUE))

```

```

##
## Reliability analysis
## raw_alpha std.alpha G6(smc) average_r S/N ase mean sd median_r
##      0.91      0.9      0.99      0.072 9.3 0.01 0.54 0.11 0.065

```

```

1 dist2mds(lex.dist,speaker.dta); lex.mds <- mds
2 print(paste0("R-squared, all dimensions: ",round(rsquared,digits=2),"; ",
3           "dimension 1: ",round(rsquared1,digits=2),"; ",
4           "dimension 2: ",round(rsquared2,digits=2),"; ",
5           "dimension 3: ",round(rsquared3,digits=2)))

```

```
## [1] "R-squared, all dimensions: 0.61; dimension 1: 0.33; dimension 2: 0.37; dimension 3: 0.03"
```

```

1 plot.mds(lex.mds,"speaker",yuc.map.areas); plot.lex.mds.speaker <- mds.ggplot

```

aggregated per location

```

1 lex.mds.2 <- subset(lex.mds,select=c("Location",
2                                     "d1","d2","d3",
3                                     "Longitude","Latitude",
4                                     "Size"))
5 lex.mds.aggr <- aggregate(x = lex.mds.2[,2:7],
6                           by = list(lex.mds.2$Location), FUN = mean)
7 rownames(lex.mds.aggr) <- lex.mds.aggr$Group.1
8 lex.mds.aggr$rgb <- rgb(r = lex.mds.aggr$d1,
9                         g = lex.mds.aggr$d2,
10                        b = 1-lex.mds.aggr$d3)
11 plot.mds(lex.mds.aggr,"location",yuc.map.cities); plot.lex.mds.location <- mds.ggplot

```

```

1 grid.arrange(plot.lex.mds.speaker,plot.lex.mds.location,ncol = 2)

```

```

1 plot.mds.dim(lex.mds,rgb(r = mds$d1,g = 0,b = 0),
2              "red","red by dim 1 (mds)","mds.lex.d1")
3 plot.mds.d1 <- plot.dim
4 plot.mds.dim(lex.mds,rgb(r = 0,g = mds$d2,b = 0),
5              "green3","green by dim 2 (mds)","mds.lex.d2")
6 plot.mds.d2 <- plot.dim
7 plot.mds.dim(lex.mds,rgb(r = 0,g = 0,b = 1-mds$d3),
8              "blue3","blue by inverse dim 3 (mds)","mds.lex.d3")
9 plot.mds.d3 <- plot.dim
10 mds2regression(lex.mds,data)

```

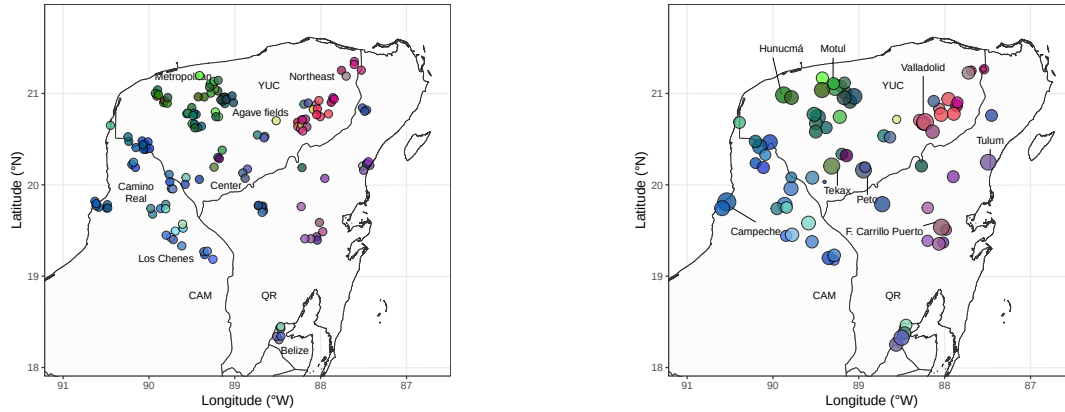



Figure 4: Estimates of multidimensional scaling mapped onto RGB colours, per speaker (left panel) and per location (right panel)

```
1 grid.arrange(plot.mds.d1,plot.mds.d2,plot.mds.d3,ncol = 3)
```

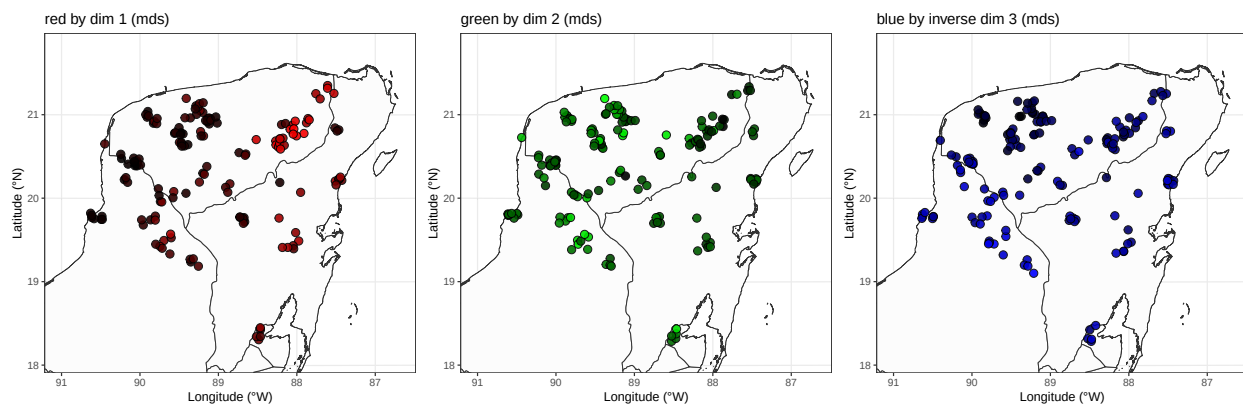


Figure 5: MDS dimensions

6.3.2 Linear model on similarity

Distance in dialect and distance in space

- create a matrix of linguistic distances
- remove self-distances
- remove duplicates

```
1 lex.dist.0 <- melt(as.matrix(lex.dist), varnames = c("sp1", "sp2"))
2 lex.dist.1 <- droplevels(subset(lex.dist.0, sp1 != sp2))
3 p <- t(apply(lex.dist.1[,c(1,2)], 1, FUN=sort))
4 p <- paste(p[,1], p[,2], sep="|")
5 rmv <- which(duplicated(p))
6 sim.dta <- lex.dist.1[-c(rmv),]
7 sim.dta$similarity <- 1-sim.dta$value
```

Geographical distance

- calculating kilometric distances between speakers out of coordinates
- setting distance of speakers in the same location to 1km
- logarithmizing distances
- rescaling to (0,1)

```
1 sim.dta$sp1.Longitude <- speaker.dta[match(sim.dta$sp1,
2                                     speaker.dta$Speaker_ID), "Longitude"]
3 sim.dta$sp1.Latitude <- speaker.dta[match(sim.dta$sp1,
4                                     speaker.dta$Speaker_ID), "Latitude"]
5 sim.dta$sp2.Longitude <- speaker.dta[match(sim.dta$sp2,
6                                     speaker.dta$Speaker_ID), "Longitude"]
7 sim.dta$sp2.Latitude <- speaker.dta[match(sim.dta$sp2,
8                                     speaker.dta$Speaker_ID), "Latitude"]
9 sim.dta$Distance.km <- geodist(sim.dta$sp2.Latitude,
10                               sim.dta$sp2.Longitude,
11                               sim.dta$sp1.Latitude,
12                               sim.dta$sp1.Longitude, units="km")
13 sim.dta[sim.dta$Distance.km < 1, ]$Distance.km <- 1
```

```
1 sim.dist.lm <- lm(sim.dta$value ~ Distance.km, data=sim.dta)
2 sim.logdist.lm <- lm(sim.dta$value ~ log2(Distance.km), data=sim.dta)
3 summary(sim.dist.lm)
```

```
##
## Call:
## lm(formula = sim.dta$value ~ Distance.km, data = sim.dta)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44664 -0.09508  0.00256  0.09495  0.42373
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.758e-01  2.501e-03   230.3  <2e-16 ***
## Distance.km  4.267e-04  1.519e-05    28.1  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1298 on 12244 degrees of freedom
## Multiple R-squared:  0.06058,    Adjusted R-squared:  0.0605
## F-statistic: 789.6 on 1 and 12244 DF,  p-value: < 2.2e-16
```

```
1 summary(sim.logdist.lm)
```

```
##
## Call:
## lm(formula = sim.dta$value ~ log2(Distance.km), data = sim.dta)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.41642 -0.09439 0.00253 0.09440 0.56449
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.4355082  0.0062259   69.95  <2e-16 ***
## log2(Distance.km) 0.0295894  0.0008943   33.09  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1283 on 12244 degrees of freedom
## Multiple R-squared:  0.08207,    Adjusted R-squared:  0.082
## F-statistic: 1095 on 1 and 12244 DF,  p-value: < 2.2e-16
```

```
1 AIC(sim.dist.lm)
```

```
## [1] -15256.96
```

```
1 AIC(sim.logdist.lm)
```

```
## [1] -15540.36
```

```
1 sim.dta$Distance <- scales::rescale(log2(sim.dta$Distance.km),to=c(0,1))
2
3 #test distribution
4 jitter3 <- position_jitter(width = 0.01, height = 0.01)
5
6 distance.plot <- ggplot(sim.dta, aes(x = Distance.km, y = value)) +
7   geom_point(size=0.01,colour="grey50",position=jitter3)+
8   xlab("Distance (km)") + ylab("Dissimilarity") +
9   stat_smooth(method = lm, formula = y ~ log2(x), size = 1, colour="black")
10 distance.plot
```

```
1 png(paste0("figs/distance.plot.png"),width =1600, height=800, res=300)
2   print(distance.plot)
3 dev.off()
```

Population size

- logarithmized products of the population size of speakers' pair locations
- rescaling to (0,1)

```
1 sim.dta$sp1.Size <- speaker.dta[match(sim.dta$sp1,
2   speaker.dta$Speaker_ID), "Pop.size.log"]
3 sim.dta$sp2.Size <- speaker.dta[match(sim.dta$sp2,
4   speaker.dta$Speaker_ID), "Pop.size.log"]
5 sim.dta$Size.products <- sim.dta$sp1.Size * sim.dta$sp2.Size
6 sim.dta$Size <- scales::rescale(sim.dta$Size.products,to=c(0,1))
```

%Indigenous population

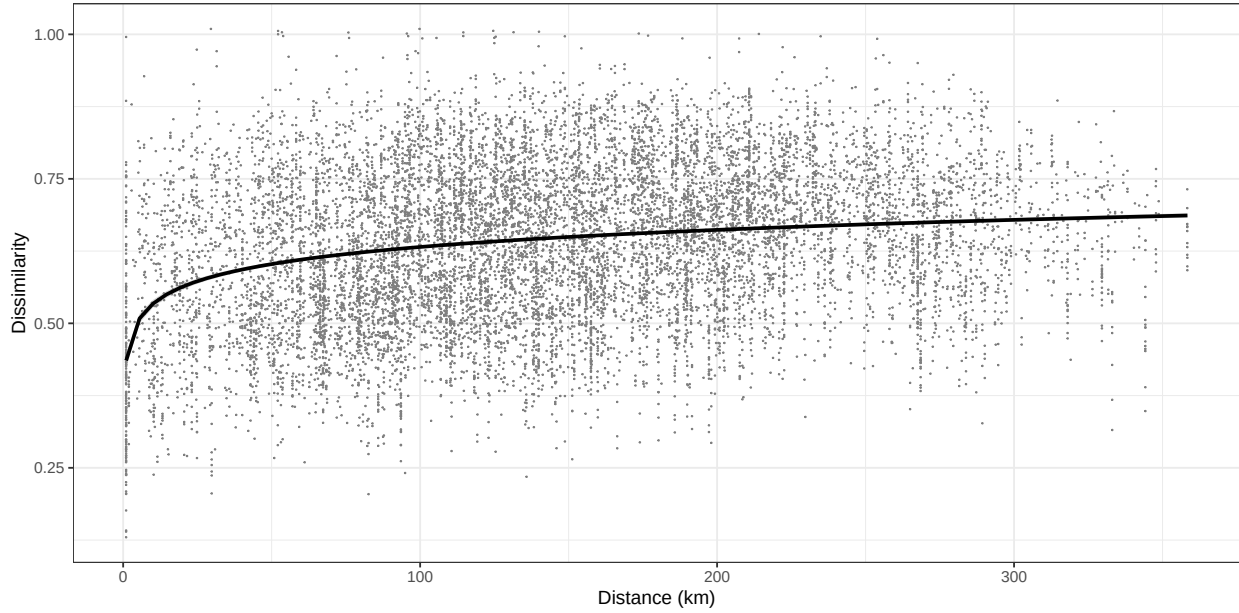


Figure 6: Dissimilarity and geographical distance (logarithmic curve)

```

1 sim.dta$sp1.Indig <- speaker.dta[match(sim.dta$sp1,
2                                     speaker.dta$Speaker_ID), "Ind.prop"]
3 sim.dta$sp2.Indig <- speaker.dta[match(sim.dta$sp2,
4                                     speaker.dta$Speaker_ID), "Ind.prop"]
5 sim.dta$Indig <- scales::rescale(sim.dta$sp1.Indig * sim.dta$sp2.Indig, to=c(0,1))

```

Time

```

1 sim.dta$sp1.Time <- speaker.dta[match(sim.dta$sp1,
2                                     speaker.dta$Speaker_ID), "Birthyear"]
3 sim.dta$sp2.Time <- speaker.dta[match(sim.dta$sp2,
4                                     speaker.dta$Speaker_ID), "Birthyear"]
5 sim.dta$Time <- scales::rescale(sim.dta$sp1.Time *
6                                sim.dta$sp2.Time, to=c(0,1))

```

Correlations between fixed effects

```

1 data.frame(cor_matrix(sim.dta[,c("Distance", "Indig", "Size", "Time")]))

```

	Distance	Indig	Size	Time
Distance	1.000	-0.124	-0.071	0.052
Indig	-0.124	1.000	-0.296	0.050
Size	-0.071	-0.296	1.000	0.095
Time	0.052	0.050	0.095	1.000

Linear model on dissimilarity

- Converting dissimilarity values to similarities

- Testing the linear model

```
1 sim.lm.0 <- lm(sim.dta$value ~ 1,data=sim.dta)
2 data.frame(summary(sim.lm.0)$coef)
```

	Estimate	Std..Error	t.value	Pr...t..
(Intercept)	0.6379017	0.0012098	527.2837	0

Model expansion with:

- forwards update
- model comparison with analysis of variance
- model selection based on difference between AIC values

```
1 sim.lm.D <- update(sim.lm.0, . ~ . +Distance)
2 anova(sim.lm.0,sim.lm.D)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
12245	219.4683	NA	NA	NA	NA
12244	201.4563	1	18.01202	1094.725	0

```
1 round(AIC(sim.lm.0)-AIC(sim.lm.D),digits=1)
```

```
## [1] 1046.7
```

```
1 sim.lm.DT <- update(sim.lm.D, . ~ . +Time)
2 anova(sim.lm.D,sim.lm.DT)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
12244	201.4563	NA	NA	NA	NA
12243	200.4817	1	0.9745812	59.51563	0

```
1 round(AIC(sim.lm.D)-AIC(sim.lm.DT),digits=1)
```

```
## [1] 57.4
```

```
1 sim.lm.DTP <- update(sim.lm.DT, . ~ . +Size)
2 anova(sim.lm.DT,sim.lm.DTP)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
12243	200.4817	NA	NA	NA	NA
12242	196.8840	1	3.59769	223.6998	0

```
1 round(AIC(sim.lm.DT)-AIC(sim.lm.DTP),digits=1)
```

```
## [1] 219.8
```

```
1 sim.lm.DTPI <- update(sim.lm.DTP, . ~ . +Indig)
```

```
2 anova(sim.lm.DTP,sim.lm.DTPI)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
12242	196.8840	NA	NA	NA	NA
12241	191.2128	1	5.671292	363.0631	0

```
1 round(AIC(sim.lm.DTP)-AIC(sim.lm.DTPI),digits=1)
```

```
## [1] 355.9
```

Coefficients of the winner model

```
1 nrow(sim.dta)
```

```
## [1] 12246
```

```
1 data.frame(round(summary(sim.lm.DTPI)$coefficients,digits=3))
```

	Estimate	Std..Error	t.value	Pr...t..
(Intercept)	0.425	0.010	40.483	0
Distance	0.262	0.008	34.844	0
Time	0.056	0.007	7.552	0
Size	-0.147	0.018	-8.384	0
Indig	0.086	0.005	19.054	0

6.3.3 Variants with wide geographical distribution

```
1 maya.dta$Variant <- as.factor(maya.dta$Variant)
2 maya.dta$Concept_ID <- as.factor(maya.dta$Concept_ID)
3 variant.sums <- data.frame(maya.dta %>%
4   group_by(Concept_ID)%>%
5   dplyr::count(Variant))
6 concept.sums <- data.frame(valid.dta %>%
7   dplyr::count(Concept_ID))
8 variant.sums$concept.sum <- concept.sums[match(variant.sums$Concept_ID,
9   concept.sums$Concept_ID),"n"]
10 variant.sums$Variant.p <- variant.sums$n/variant.sums$concept.sum
11 DI.density <- ggplot(variant.sums, aes(x = Variant.p)) +
12   xlab("Distribution index") +
13   geom_density()
```

```
1 mean(variant.sums$Variant.p)
```

```
## [1] 0.4165293
```

```
1 sd(variant.sums$Variant.p)
```

```
## [1] 0.3155451
```

```
1 print(DI.density)
```

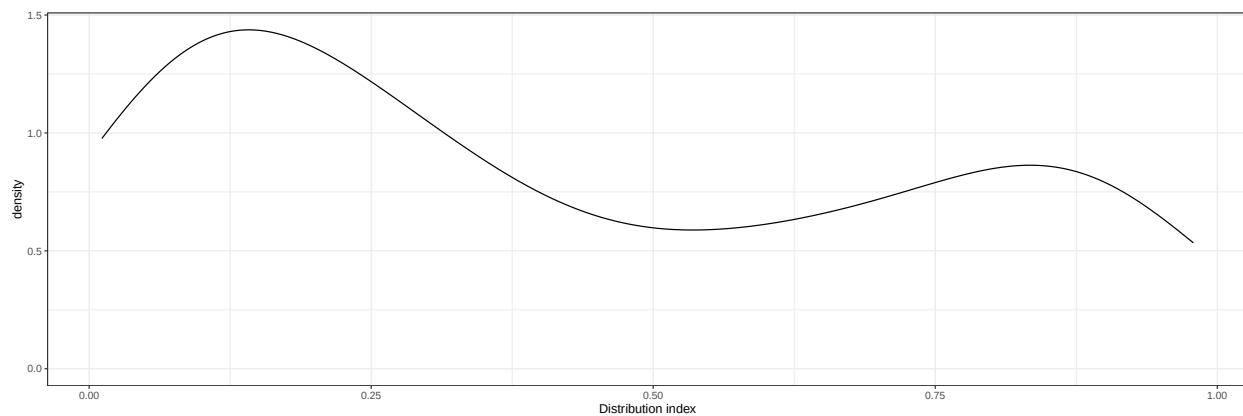


Figure 7: Density of the distribution indices of indigenous variants

```
1 png(paste0("figs/DI.density.png"),width =1600, height=800, res=300)
2 print(DI.density)
3 dev.off()
```

```
1 maya.dta$Variant.p <-
2     variant.sums[match(paste0(maya.dta$Concept_ID,maya.dta$Variant),
3     paste0(variant.sums$Concept_ID,variant.sums$Variant)),"Variant.p"]
4
5 diffusion.aggr <- aggregate(x = maya.dta$Variant.p,
6     by = list(maya.dta$Speaker_ID,
7     maya.dta$Longitude,
8     maya.dta$Latitude,
9     maya.dta$Location,
10    maya.dta$Birthyear,
11    maya.dta$Pop.size.log),
12    FUN = mean)
13 colnames(diffusion.aggr) <- c("Speaker","Longitude","Latitude","Location",
14    "Birthyear","Pop.size.log","Variant.p")
15 plot.map.gradient(diffusion.aggr,"distr. index",.6)
16 plot.gradient.indig <- plot.gradient
```

```
1 plot.gradient.indig
```

Rescaling: beta regression requires values in (0,1)

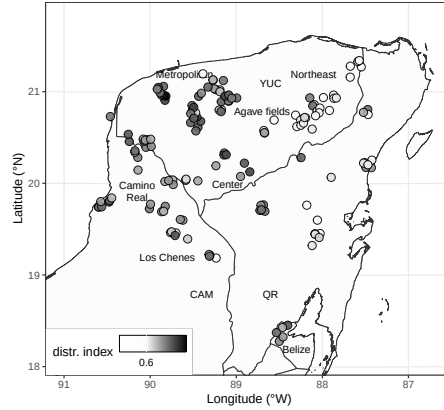


Figure 8: Occurrence of Spanish words

```

1 maya.dta$Variant.p <- scales::rescale(maya.dta$Variant.p,to=c(.001,0.999))
2 maya.dta$Pop.size <- scales::rescale(maya.dta$Pop.size.log,to=c(0.001,0.999))
3 maya.dta$Time <- scales::rescale(maya.dta$Birthyear,to=c(0.001,0.999))
4 maya.dta$Concept <- as.factor(maya.dta$Concept_ID)
5 maya.dta$Ind.prop <- scales::rescale(maya.dta$Ind.prop,to=c(0.001,0.999))
6 nrow(maya.dta)

```

```
## [1] 6304
```

Correlation matrix

```
1 data.frame(cor_matrix(maya.dta[,c("Pop.size","Ind.prop","Time")]))
```

	Pop.size	Ind.prop	Time
Pop.size	1.000	-0.674	0.135
Ind.prop	-0.674	1.000	0.063
Time	0.135	0.063	1.000

Generalized Additive Mixed-Effects Model

- calculating a gam with the smooth terms (Concept and Space)
- stepwise forwards update with model comparison

```

1 gam.indigenous.0 = gam(Variant.p ~ s(Concept, bs="re")+
2   s(Longitude, Latitude, bs="tp",k=15),
3   data=maya.dta,method="ML", family="betar")
4 gam.indigenous.T <- update(gam.indigenous.0, . ~ . +Time)
5 cm.ind.0.T <- compareML(gam.indigenous.0,gam.indigenous.T,suggest.report = T)$table
6 gam.indigenous.TP <- update(gam.indigenous.T, . ~ . +Pop.size)
7 cm.ind.T.TP <- compareML(gam.indigenous.T,gam.indigenous.TP,suggest.report = T)$table
8 gam.indigenous.TI <- update(gam.indigenous.T, . ~ . +Ind.prop)
9 cm.ind.T.TI <- compareML(gam.indigenous.T,gam.indigenous.TI,suggest.report = T)$table

```



```
1 cm.ind.0.T
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.indigenous.0	-3054.636	5				
gam.indigenous.T	-3057.027	6	2.391	1.000	0.029	*

```
1 cm.ind.T.TP
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.indigenous.T	-3057.027	6				
gam.indigenous.TP	-3057.793	7	0.765	1.000	0.216	

```
1 cm.ind.T.TI
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.indigenous.T	-3057.027	6				
gam.indigenous.TI	-3058.252	7	1.225	1.000	0.117	

Coefficients of the winner model

```
1 gam.indigenous <- gam.indigenous.T
2 summary(gam.indigenous)
```

```
##
## Family: Beta regression(4.275)
## Link function: logit
##
## Formula:
## Variant.p ~ s(Concept, bs = "re") + s(Longitude, Latitude, bs = "tp",
##      k = 15) + Time
##
## Parametric coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.74009    0.12105   6.114 9.73e-10 ***
## Time        -0.13742    0.06281  -2.188  0.0287 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df Chi.sq p-value
## s(Concept)      50.49  51.0 5045.7 <2e-16 ***
## s(Longitude,Latitude) 10.35  12.4  119.4 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.434   Deviance explained = 55.9%
## -ML = -3057   Scale est. = 1           n = 6304
```

Creating plot of the smooth term of Space

```
1 plot.gam(gam.indigenous)
```

6.4 Spanish Variants

- coding languages as 0 (Maya) and 1 (Spanish)

```
1 valid.dta$Variant.p <- as.numeric(as.factor(valid.dta$Language))-1
```

- reliability (Cronbach's alpha)

```
1 valid.lang.2 <- dcast(subset(valid.dta,select = c(Speaker_ID, Concept_ID, Language)),
2                       Speaker_ID ~ Concept_ID)
3 valid.lang <- valid.lang.2[,c(2:ncol(valid.lang.2))]
4 valid.lang <- valid.lang %>%
5     mutate_all(funs(type.convert(as.integer(as.factor(.))-1)),
6                na.rm = T, as.is = FALSE)
7 valid.lang <- valid.lang %>%
8     mutate_all(funs(type.convert(as.logical(as.integer(as.factor(.))-1))),
9                na.rm = T, as.is = FALSE)
10 #summary(psych::alpha(valid.lang, na.rm=T))
11 #Likely variables with missing values are C034 C020 C021 C011 C008 C029
12 valid.subset <- subset(valid.lang,select=-c(C008, C011, C020, C021, C029, C034))
13 summary(psych::alpha(valid.subset, na.rm=T))
```

```
##
```

```
## Reliability analysis
```

```
## raw_alpha std.alpha G6(smc) average_r S/N ase mean sd median_r
## 0.76 0.74 0.85 0.19 2.8 0.024 0.034 0.051 0.15
```

- calculating the proportions of Spanish per speaker by aggregation
- rescaling fixed factors to a (0,1) interval

```
1 valid.dta$Pop.size <- scales::rescale(valid.dta$Pop.size.log,to=c(.001,.999))
2 valid.dta$Time <- scales::rescale(valid.dta$Birthyear,to=c(.001,.999))
3 valid.dta$Ind.prop <- scales::rescale(valid.dta$Ind.prop,to=c(.001,.999))
4 valid.dta$Concept <- as.factor(valid.dta$Concept_ID)
5 valid.dta$Language <- as.factor(valid.dta$Language)
6 valid.dta.aggr <- aggregate(x = valid.dta[,c("Speaker_ID","Latitude",
7                                              "Longitude","Variant.p")],
8                             by = list(valid.dta$Speaker_ID,
9                                         valid.dta$Longitude,
10                                        valid.dta$Latitude), FUN = mean)
11 plot.map.gradient(valid.dta.aggr,"%Spanish",0)
12 plot.gradient.spanish <- plot.gradient
```

```
1 plot.gradient.spanish
```

Correlation matrix

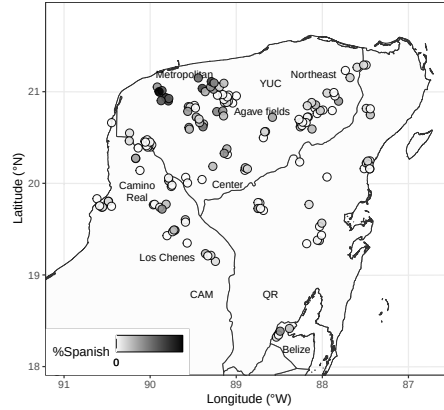


Figure 9: Occurrence of Spanish words

```
1 data.frame(cor_matrix(valid.dta[,c("Pop.size", "Ind.prop", "Time")]))
```

	Pop.size	Ind.prop	Time
Pop.size	1.000	-0.674	0.137
Ind.prop	-0.674	1.000	0.061
Time	0.137	0.061	1.000

Generalized Additive Mixed-Effects Model

- calculating a gam with the smooth terms (Concept and Space)
- stepwise forwards update with model comparison

```
1 gam.loans.0 = mgcv::gam((Language=="Spanish") ~
2                       s(Longitude, Latitude, bs="tp", k=15)+
3                       s(Concept, bs="re"),
4                       data=valid.dta,
5                       method="ML",
6                       family=binomial)
7 gam.loans.T <- update(gam.loans.0, . ~ . +Time)
8 cm.loans.0.T <- compareML(gam.loans.0, gam.loans.T, suggest.report = T)$table
9 gam.loans.TP <- update(gam.loans.T, . ~ . +Pop.size)
10 cm.loans.T.TP <- compareML(gam.loans.T, gam.loans.TP, suggest.report = T)$table
11 gam.loans.TI <- update(gam.loans.T, . ~ . +Ind.prop)
12 cm.loans.T.TI <- compareML(gam.loans.T, gam.loans.TI, suggest.report = T)$table
```

```
1 cm.loans.0.T
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.loans.0	464.4198	5				
gam.loans.T	461.4400	6	2.980	1.000	0.015	*

```
1 cm.loans.T.TP
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.loans.T	461.4400	6				
gam.loans.TP	461.4393	7	0.001	1.000	0.971	

```
1 cm.loans.T.TI
```

Model	Score	Edf	Difference	Df	p.value	Sig.
gam.loans.T	461.4400	6				
gam.loans.TI	461.4325	7	0.007	1.000	0.903	

Coefficients of the winner model

```
1 gam.loans <- gam.loans.T
2 print(summary(gam.loans))

##
## Family: binomial
## Link function: logit
##
## Formula:
## (Language == "Spanish") ~ s(Longitude, Latitude, bs = "tp", k = 15) +
##   s(Concept, bs = "re") + Time
##
## Parametric coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -6.9182    0.6449  -10.728  <2e-16 ***
## Time          1.3480    0.5589   2.412   0.0159 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##              edf Ref.df Chi.sq p-value
## s(Longitude, Latitude) 12.06  13.48  132.5  <2e-16 ***
## s(Concept)             36.44  51.00  332.1  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.384   Deviance explained = 51.7%
## -ML = 461.44   Scale est. = 1           n = 6475
```

Creating plot of the smooth term of Space

```
1 plot.gam(gam.loans)
```

7 Sources

7.1 References

- [1] “Database of Global Administrative Areas” at https://gadm.org/download_country.html. (accessed 2021-12-02)
- [2] Dawson, Michael R.W. 2015. “Computing The Fit Of An MDS Solution Using R”, in *Cognition and Reality*, blogspot: <http://cognitionandreality.blogspot.com/2015/04/computing-fit-of-mds-solution-using-r.html> (accessed 2021-12-02)
- [3] INEGI, Censo de Población y Vivienda 2010; at <https://www.inegi.org.mx/programas/ccpv/2010/>.
- [4] INEGI, Censo de Población y Vivienda 2010,Infraestructura y Características Socioeconómicas de las Localidades con menos de 5 mil habitantes, Descripción de la estructura de datos; at https://www.inegi.org.mx/contenidos/programas/ccpv/2010/doc/fd_resloc_2010.pdf (accessed 2021-11-29)
- [5] INEGI, Censo de Población y Vivienda 2010, Tabulados, at <https://www.inegi.org.mx/programas/ccpv/2010/#Tabulados> (accessed 2021-11-29).

7.2 R and package versions

```
1 print(sessionInfo(), RNG = F, locale = F,loadedOnly=T)

## R version 4.1.1 (2021-08-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19041)
##
## Matrix products: default
##
## attached base packages:
## [1] splines      stats4      stats      graphics  grDevices  utils      datasets
## [8] methods     base
##
## other attached packages:
## [1] itsadug_2.4      plotfunctions_1.4  mgcv_1.8-36
## [4] nlme_3.1-152     lme4_1.1-27.1      Matrix_1.3-4
## [7] regclass_1.6     randomForest_4.6-14 rpart_4.1-15
## [10] VGAM_1.1-5       bestglm_0.37.3     leaps_3.1
## [13] scales_1.1.1     psych_2.1.9        cluster_2.1.2
## [16] MASS_7.3-54      geosphere_1.5-10   sp_1.4-5
## [19] gmt_2.0.2        pdp_0.7.0          ggmap_3.0.0
## [22] ggplot2_3.3.5    dplyr_1.0.7        varhandle_2.0.5
## [25] reshape2_1.4.4   writexl_1.4.0      readxl_1.3.1
##
## loaded via a namespace (and not attached):
## [1] bitops_1.0-7      sf_1.0-2           httr_1.4.2
## [4] grpreg_3.4.0      tools_4.1.1        utf8_1.2.2
## [7] R6_2.5.0          KernSmooth_2.23-20 DBI_1.1.1
## [10] colorspace_2.0-2  withr_2.4.2        tidymodels_1.1.1
## [13] gridExtra_2.3     mnormt_2.0.2       compiler_4.1.1
## [16] cli_3.0.1         glmnet_4.1-2       labeling_0.4.2
## [19] classInt_0.4-3    proxy_0.4-26       stringr_1.4.0
```

## [22] digest_0.6.27	minqa_1.2.4	rmarkdown_2.10
## [25] jpeg_0.1-9	pkgconfig_2.0.3	htmltools_0.5.1.1
## [28] highr_0.9	rlang_0.4.11	rstudioapi_0.13
## [31] farver_2.1.0	shape_1.4.6	generics_0.1.0
## [34] magrittr_2.0.1	s2_1.0.6	Rcpp_1.0.7
## [37] munsell_0.5.0	fansi_0.5.0	lifecycle_1.0.0
## [40] stringi_1.7.3	yaml_2.2.1	plyr_1.8.6
## [43] grid_4.1.1	parallel_4.1.1	pls_2.7-3
## [46] crayon_1.4.1	lattice_0.20-44	tmvnsim_1.0-2
## [49] knitr_1.33	pillar_1.6.2	boot_1.3-28
## [52] rjson_0.2.20	codetools_0.2-18	wk_0.5.0
## [55] glue_1.4.2	evaluate_0.14	rpart.plot_3.1.0
## [58] png_0.1-7	vctrs_0.3.8	nloptr_1.2.2.2
## [61] foreach_1.5.1	RgoogleMaps_1.4.5.3	cellranger_1.1.0
## [64] gtable_0.3.0	purrr_0.3.4	tidyr_1.1.3
## [67] assertthat_0.2.1	xfun_0.25	e1071_1.7-8
## [70] class_7.3-19	survival_3.2-11	tibble_3.1.3
## [73] iterators_1.0.13	units_0.7-2	ellipsis_0.3.2