Statistics-for-Linguists-2024-01-17-Multilevel-models

Adam Tallman

2024-01-16

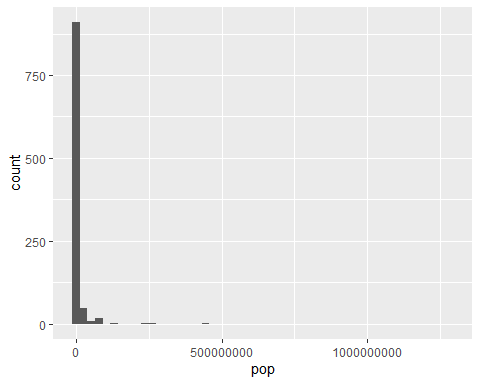
## The data

df <- read.csv("/Users/Adan Tallman/Desktop/StatisticsinLinguistics\_FSU\_2023\_2024/07\_data/phoneme\_inventories.csv", header = TRUE)

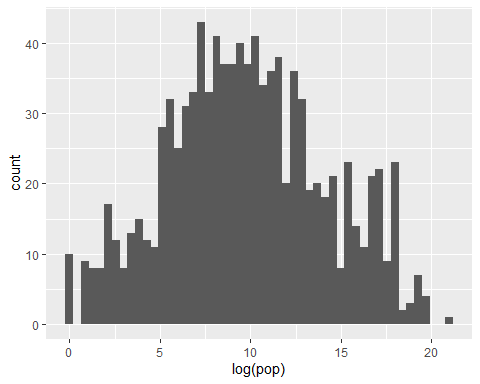
head(df)

## X pop family.name area inv language  
## 1 1 77233270 Koreanic Eurasia 40 Korean  
## 2 2 210 Yeniseian Eurasia 32 Ket  
## 3 3 154800 Nakh-Daghestanian Eurasia 69 Lak  
## 4 4 1628500 Abkhaz-Adyge Eurasia 56 Kabardian  
## 5 5 3669500 Kartvelian Eurasia 35 Georgian  
## 6 6 96800 Eurasia 53 Burushaski

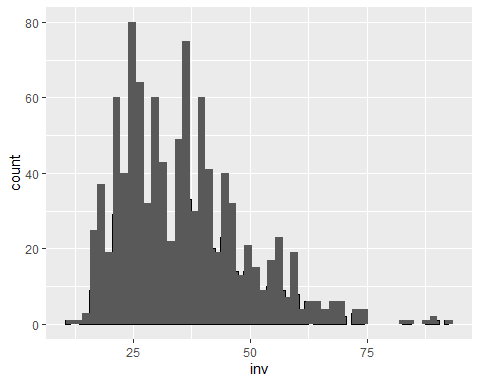
options(scipen = 999)  
ggplot(df, aes(x=pop))+  
 geom\_bar(color="black")+  
 stat\_bin(bins=50)



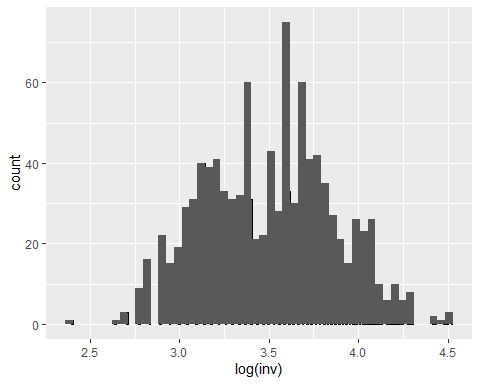
ggplot(df, aes(x=log(pop)))+  
 geom\_bar(color="black")+  
 stat\_bin(bins=50)



ggplot(df, aes(x=inv))+  
 geom\_bar(color="black")+  
 stat\_bin(bins=50)



ggplot(df, aes(x=log(inv)))+  
 geom\_bar(color="black")+  
 stat\_bin(bins=50)

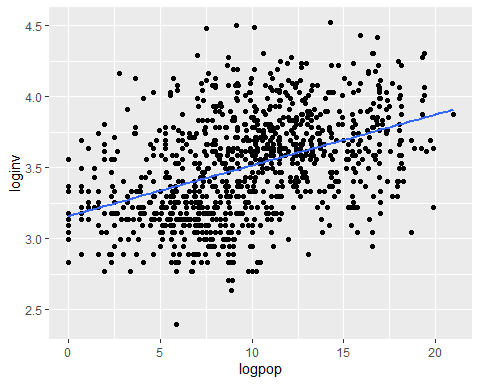


So let’s use variables that are more normally distributed.

df$logpop <- log(df$pop)  
df$loginv <- log(df$inv)

plot.ods <- ggplot(df, aes(x=logpop, y=loginv))+  
 geom\_point()+  
 geom\_smooth(method = "lm", se =FALSE)  
plot.ods

## `geom\_smooth()` using formula = 'y ~ x'



model.ols1 <- lm(loginv~logpop, data=df)   
summary(model.ols1)

##   
## Call:  
## lm(formula = loginv ~ logpop, data = df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.97096 -0.22219 -0.00436 0.22272 1.05104   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.158794 0.025180 125.4 <0.0000000000000002 \*\*\*  
## logpop 0.035688 0.002348 15.2 <0.0000000000000002 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3258 on 1001 degrees of freedom  
## Multiple R-squared: 0.1875, Adjusted R-squared: 0.1867   
## F-statistic: 231 on 1 and 1001 DF, p-value: < 0.00000000000000022

## Levels in the data

But there are huge differences across the families in terms of representativeness

table(df$family.name)

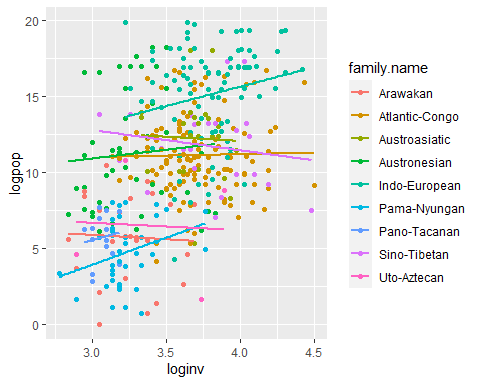
##   
## Abkhaz-Adyge Afro-Asiatic   
## 41 2 41   
## Algic Araucanian Arawakan   
## 2 1 29   
## Arawan Athabaskan-Eyak-Tlingit Atlantic-Congo   
## 2 2 126   
## Austroasiatic Austronesian Aymaran   
## 25 51 1   
## Barbacoan Bookkeeping Boran   
## 2 1 2   
## Border Bororoan Bunaban   
## 2 1 1   
## Caddoan Cahuapanan Cariban   
## 1 1 19   
## Central Sudanic Chapacuran Chibchan   
## 9 1 3   
## Chicham Chocoan Chukotko-Kamchatkan   
## 2 5 4   
## Dravidian East Strickland Eskimo-Aleut   
## 17 1 1   
## Garrwan Goilalan Guahiboan   
## 1 1 4   
## Guaicuruan Gunwinyguan Harakmbut   
## 5 1 1   
## Heibanic Huitotoan Ijoid   
## 2 3 3   
## Indo-European Iroquoian Iwaidjan Proper   
## 93 2 2   
## Japonic Jodi-Saliban Kadugli-Krongo   
## 2 2 3   
## Kartvelian Khoe-Kwadi Koman   
## 4 1 1   
## Konda-Yahadian Koreanic Kuliak   
## 1 2 1   
## Maban Mande Mangarrayi-Maran   
## 1 8 2   
## Maningrida Mayan Mixed Language   
## 2 2 1   
## Mongolic Muskogean Nadahup   
## 2 4 3   
## Nakh-Daghestanian Nambiquaran Narrow Talodi   
## 8 2 1   
## Nilotic North Halmahera Nuclear-Macro-Je   
## 11 1 14   
## Nuclear Torricelli Nuclear Trans New Guinea Nyimang   
## 4 13 1   
## Nyulnyulan Pama-Nyungan Pano-Tacanan   
## 2 39 16   
## Peba-Yagua Quechuan Sahaptian   
## 1 2 1   
## Salishan Sentanic Sepik   
## 2 1 3   
## Sino-Tibetan Siouan Sko   
## 23 1 1   
## Songhay South Bird's Head Family South Omotic   
## 1 1 2   
## Surmic Ta-Ne-Omotic Tai-Kadai   
## 1 5 6   
## Tangkic Ticuna-Yuri Timor-Alor-Pantar   
## 2 2 1   
## Tucanoan Tungusic Tupian   
## 13 2 36   
## Turkic Uralic Uru-Chipaya   
## 6 5 2   
## Uto-Aztecan Western Daly Worrorran   
## 4 1 4   
## Yangmanic Yanomamic Yeniseian   
## 1 4 1   
## Zamucoan Zaparoan   
## 2 2

Let’s take the largest numbers

df.lgsubset <- subset(df, family.name == "Indo-European"|  
 family.name =="Uto-Aztecan"|  
 family.name == "Atlantic-Congo"|  
 family.name == "Sino-Tibetan"|  
 family.name == "Otomanguean"|  
 family.name == "Austronesian"|  
 family.name == "Pama-Nyungan"|  
 family.name == "Austroasiatic"|  
 family.name =="Arawakan"|  
 family.name == "Pano-Tacanan")

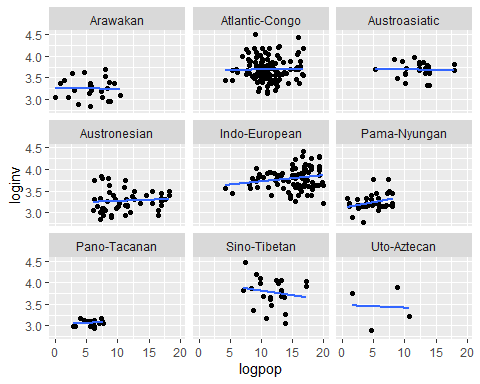
ggplot(df.lgsubset, aes(x=loginv, y=logpop, color=family.name))+  
 geom\_point()+  
 geom\_smooth(method="lm", se = FALSE)

## `geom\_smooth()` using formula = 'y ~ x'



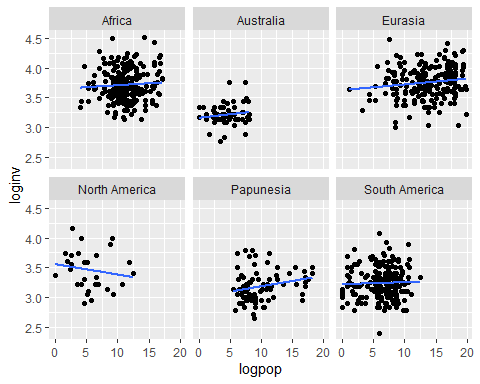
plot1 <- ggplot(df.lgsubset, aes(x=logpop, y=loginv, group=family.name))+  
 geom\_point()+  
 geom\_smooth(method="glm",se = FALSE)  
plot1 + facet\_wrap(~ family.name, ncol=3)

## `geom\_smooth()` using formula = 'y ~ x'



df.areasubset <- subset(df, area !="")  
  
plot2 <- ggplot(df.areasubset, aes(x=logpop, y=loginv, group=area))+  
 geom\_point()+  
 geom\_smooth(method="glm",se = FALSE)  
plot2 + facet\_wrap(~ area, ncol=3)

## `geom\_smooth()` using formula = 'y ~ x'



## Multilevel model

One disadvantage of multilevel models is that their complexity makes them hard to interpret.

Its typical to AICs in order to assess multilevel models. These are multilevel models without the predictor variable.

mod.lmer.null1 <- lmer(loginv~(1|family.name)+ (1|area), data=df)  
mod.lmer.null2 <- lmer(loginv~(1|family.name), data=df)  
mod.lmer.null3 <- lmer(loginv~(1|area), data=df)

anova(mod.lmer.null1,  
 mod.lmer.null2,  
 mod.lmer.null3)

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

## Data: df  
## Models:  
## mod.lmer.null2: loginv ~ (1 | family.name)  
## mod.lmer.null3: loginv ~ (1 | area)  
## mod.lmer.null1: loginv ~ (1 | family.name) + (1 | area)  
## npar AIC BIC logLik deviance Chisq Df Pr(>Chisq)   
## mod.lmer.null2 3 204.31 218.39 -99.157 198.31   
## mod.lmer.null3 3 151.88 165.96 -72.938 145.88 52.438 0   
## mod.lmer.null1 4 117.22 135.99 -54.609 109.22 36.658 1 0.000000001408 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The first null model is the best because it has the lowest AIC and lowest BIC.

mod.lmer.a <- lmer(loginv~logpop+(1|family.name)+(1|area), data=df) #This is the code for a varying intercept model  
  
  
mod.lmer.ab <- lmer(loginv~logpop+(1+logpop|family.name) +(1+logpop|area),data=df) #This is the code for a varying intercept and slope model

## boundary (singular) fit: see help('isSingular')

summary(mod.lmer.a)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [  
## lmerModLmerTest]  
## Formula: loginv ~ logpop + (1 | family.name) + (1 | area)  
## Data: df  
##   
## REML criterion at convergence: 116.3  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -3.8071 -0.5921 -0.0520 0.5813 3.4080   
##   
## Random effects:  
## Groups Name Variance Std.Dev.  
## family.name (Intercept) 0.01955 0.1398   
## area (Intercept) 0.05705 0.2389   
## Residual 0.05848 0.2418   
## Number of obs: 807, groups: family.name, 104; area, 7  
##   
## Fixed effects:  
## Estimate Std. Error df t value Pr(>|t|)   
## (Intercept) 3.381498 0.099630 6.737558 33.941 0.00000000875 \*\*\*  
## logpop 0.007141 0.003044 804.900951 2.346 0.0192 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Correlation of Fixed Effects:  
## (Intr)  
## logpop -0.263

summary(mod.lmer.ab)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [  
## lmerModLmerTest]  
## Formula: loginv ~ logpop + (1 + logpop | family.name) + (1 + logpop |   
## area)  
## Data: df  
##   
## REML criterion at convergence: 116.3  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -3.8076 -0.5875 -0.0518 0.5812 3.4039   
##   
## Random effects:  
## Groups Name Variance Std.Dev. Corr   
## family.name (Intercept) 0.0188750126 0.1373864   
## logpop 0.0000001337 0.0003656 1.00   
## area (Intercept) 0.0604632223 0.2458927   
## logpop 0.0000005919 0.0007694 -1.00  
## Residual 0.0584361795 0.2417358   
## Number of obs: 807, groups: family.name, 104; area, 7  
##   
## Fixed effects:  
## Estimate Std. Error df t value Pr(>|t|)   
## (Intercept) 3.381285 0.102232 5.778341 33.075 0.0000000827 \*\*\*  
## logpop 0.007291 0.003055 214.409012 2.386 0.0179 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Correlation of Fixed Effects:  
## (Intr)  
## logpop -0.346  
## optimizer (nloptwrap) convergence code: 0 (OK)  
## boundary (singular) fit: see help('isSingular')

So the correlation is still weakly statistically significant.

anova(mod.lmer.null1, mod.lmer.a, mod.lmer.ab)

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

## Data: df  
## Models:  
## mod.lmer.null1: loginv ~ (1 | family.name) + (1 | area)  
## mod.lmer.a: loginv ~ logpop + (1 | family.name) + (1 | area)  
## mod.lmer.ab: loginv ~ logpop + (1 + logpop | family.name) + (1 + logpop | area)  
## npar AIC BIC logLik deviance Chisq Df Pr(>Chisq)   
## mod.lmer.null1 4 117.22 135.99 -54.609 109.22   
## mod.lmer.a 5 113.66 137.12 -51.828 103.66 5.5612 1 0.01836 \*  
## mod.lmer.ab 9 121.59 163.83 -51.797 103.59 0.0624 4 0.99952   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

According to the BIC the null model accounts for the variation better than the models with the predictor variables.

## Coefficients from a multilevel model

There isn’t just a single coefficient for a multilevel model

beta1 <- coef(mod.lmer.ab)$family.name  
colnames(beta1) <- c("Intercept", "Slope")  
  
beta2 <- coef(mod.lmer.ab)$area  
colnames(beta2) <- c("Intercept", "Slope")

If we plot the coefficients we can see that they follow a normal distribution.

library(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':  
##   
## combine

p1 <- ggplot(beta1, aes(Slope))+  
 geom\_density(fill="slategray2", color="slategray2", alpha=0.8)+  
 ggtitle("Random slopes by linguistic family")  
  
  
p2 <- ggplot(beta1, aes(Intercept))+  
 geom\_density(fill="slategray2", color="slategray2", alpha=0.8)+  
 ggtitle("Random intercepts by linguistic family")  
  
p3 <- ggplot(beta2, aes(Slope))+  
 geom\_density(fill="slategray2", color="slategray2", alpha=0.8)+  
 ggtitle("Random slopes by area")  
  
p4 <- ggplot(beta2, aes(Intercept))+  
 geom\_density(fill="slategray2", color="slategray2", alpha=0.8)+  
 ggtitle("Random intercepts by area")  
  
grid.arrange(p1, p2, p3, p4, ncol=2)

