MLE Midterm

Jeremy Spater

March 10, 2015

The purpose of this exercise is to create a function that mimics R's built-in "lm" command by calculating the OLS estimator and standard errors. The code is included in an appendix below. In addition, we used our code to compare two methods of dealing with missing data, i.e. listwise deletion and AMELIA imputation, with the results we obtained for no missing data. The original data contained 46 data points; the set with missing data had NA's in six rows, leaving 40 data points after listwise deletion. The AMELIA process imputed the missing data in these 6 rows, leaving a total of 46 data points. The results are shown below in the coefficient plots. Because the coefficients for "polity2" are difficult to see in the scale of the first plot, we include a separate coefficient plot for just this variable.

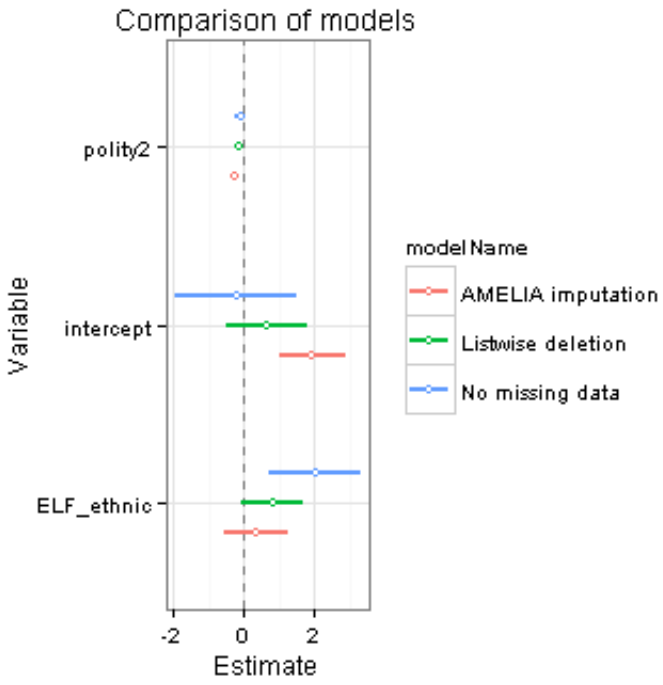

Figure 1: Comparison of regressions with full data, listwise deletion, and AMELIA imputation.

From the figures, we note that the confidence bands overlap for all three models. However, the polity score is significantly negative for the listwise deletion and AMELIA models, but not significant for the model with full data. In addition, ethnolinguistic fragmentation is significantly positive for the model with full data, but not for the listwise deletion and AMELIA models. The interpretation of our regressions, then, differs substantively when data is missing, regardless of whether we use imputation or listwise deletion.

Strangely, listwise deletion gives results that are closer to the full-data results than does AMELIA imputation. This is contrary to theory, which implies that AMELIA imputation should give less biased results than simply deleting cases with missing data. This result deserves further study. It may be that the average of coefficients estimated from repeated imputations would converge on the values estimated for the full non-missing dataset. Checking whether this is the case would be an interesting future project.

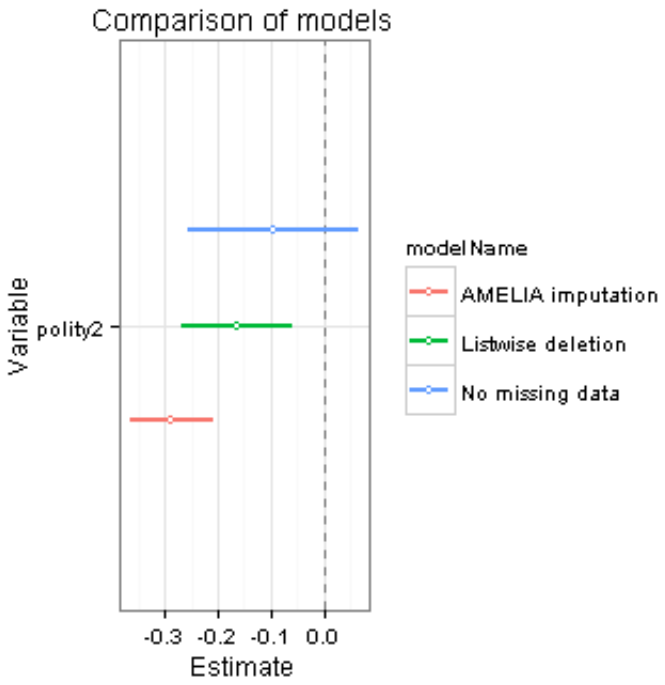

Figure 2: Comparison of estimates for polity2.

Appendix: R code for regressions

```
# Set up workspace
rm(list=ls())
setwd("C:/Users/Jeremy/Desktop/mle lab/week7")

# Function to load packages
loadPkg=function(toLoad){
  for(lib in toLoad){
    if(! lib %in% installed.packages()[,1])
    { install.packages(lib, repos='http://cran.rstudio.com/') }
    suppressMessages( library(lib, character.only=TRUE) ) }
}

# Load libraries
packs=c('foreign', 'lmtest', 'sandwich', 'Amelia', 'sbgcop', 'ggplot2')
loadPkg(packs)

#Bring in data . . .
load("midTermData.rda")

#model = ols(formula=form, data=data)
#modelListDel = ols(formula = form, data=dataMiss)
#modelAmelia = ols(formula = form, data=dataMiss, impute=TRUE)

#set seed
set.seed(6886)

#set regression formula
form = formula(gini_net_std ~ polity2 + ELF_ethnic)
ols <- function(formula, data, impute=FALSE){
  # Retrieve vars from formula input

    if(impute==TRUE){
    print("Imputing missing data with Amelia")
    AmeliaOutput=amelia(x=data,m=1)
    data=AmeliaOutput$imp$imp1
  }

  #listwise deletion of missing data (won't be any if impute==true)
  data=na.omit(data)

  formula=form
  dv = all.vars(formula)[1]
  ivs = all.vars(formula)[ 2:length(all.vars(formula)) ]

  # Create matrix with column for intercept and
  ## data from independent variables
  y = data[,dv]
  x = data.matrix(cbind(1, data[,ivs]))#include 1
```

```
# General parameters
n = nrow(x) # Number of observations
p = length(ivs) # Number of parameters
df = n-p-1 # degrees of freedom

# Derive OLS estimator of beta
xTx = t(x) %*% x
xTxinv=solve(xTx)
xTy = t(x) %*% y
beta = solve(xTx) %*% xTy
yhat=x %*% beta

#calculate residuals
e=y-yhat
ssr = t(e) %*% e
ssr2=sum(e^2)

#calculate standard error
stder=ssr/df

#calculate estimated covariance matrix of betahat
Sigma2=as.numeric(stder)*xTxinv #square root of these diagonals give std errors as repor
sigma=sqrt(diag(Sigma2))

#t values
t=beta/sigma

#p values
pvals=2*pt(abs(t),df,lower.tail=FALSE)#2 accounts for prob of getting more-extreme value

#confidence intervals
numstd=qt(0.975,43,lower.tail=TRUE)
conf=matrix(data=NA,nrow=p+1,ncol=2)
conf=cbind(beta-numstd*sigma,beta+numstd*sigma)

#make matrix
coefficients=cbind(beta,sigma,t,pvals,conf)
colnames(coefficients)=c("Estimate", "Std. Error", "T-Statistic", "P-Value", "Lower 95% C

#rename varcov matrix
varcov=Sigma2

#calculate r2
ymean=sum(y)/n
tss=sum((y-ymean)^2)
ess=sum((yhat-ymean)^2)
rss=ssr
Rsq=1-rss/tss

#F statistic
```

```
  MSreg = sum(yhat^2)/p
  MSres = sum(e^2)/(n-p-1)
  fstat=MSreg/MSres
  pvalf=1-pf(fstat,p,n-p-1)

  Fstat =paste("F-statistic: ", as.character(round(fstat,3)), " on ", as.character(p), " an

  output=list(coefficients, varcov, Rsq, Fstat)
  names(output)=c("coefficients", "varcov", "Rsq", "Fstat")

  return(output)

}
```

Appendix: R code for plots

```
#run this AFTER running midtermfunction.R and
#model = ols(formula=form, data=data)
#modelListDel = ols(formula = form, data=dataMiss)
#modelAmelia = ols(formula = form, data=dataMiss, impute=TRUE)

#Make a data frame
model1Frame=data.frame(model$coefficients)
model1Frame$modelName="No missing data"
row.names(model1Frame)[1]="intercept"
model1Frame$Variable=row.names(model1Frame)


model2Frame=data.frame(modelListDel$coefficients)
model2Frame$modelName="Listwise deletion"
row.names(model2Frame)[1]="intercept"
model2Frame$Variable=row.names(model1Frame)


model3Frame=data.frame(modelAmelia$coefficients)
model3Frame$modelName="AMELIA imputation"
row.names(model3Frame)[1]="intercept"
model3Frame$Variable=row.names(model1Frame)


allModelFrame <- data.frame(rbind(model3Frame, model2Frame, model1Frame))

#make plot . . .
zp1 <- ggplot(allModelFrame, aes(colour = modelName))
zp1 <- zp1 + geom_hline(yintercept = 0, colour = gray(1/2), lty = 2)
zp1 <- zp1 + geom_linerange(aes(x = Variable, ymin = Lower.95..CI,
                                ymax = Upper.95..CI),
                            lwd = 1, position = position_dodge(width = 1/2))
zp1 <- zp1 + geom_pointrange(aes(x = Variable, y = Estimate, ymin = Lower.95..CI,
                                 ymax = Upper.95..CI),
                             lwd = 1/2, position = position_dodge(width = 1/2),
                             shape = 21, fill = "WHITE")
zp1 <- zp1 + coord_flip() + theme_bw()
zp1 <- zp1 + ggtitle("Comparison of models")
print(zp1)  # The trick to these is position_dodge().


#make another plot with just polity, since it doesn't show up well
littleframe=allModelFrame[allModelFrame$Variable=="polity2",]


zp2 <- ggplot(littleframe, aes(colour = modelName))
zp2 <- zp2 + geom_hline(yintercept = 0, colour = gray(1/2), lty = 2)
zp2 <- zp2 + geom_linerange(aes(x = Variable, ymin = Lower.95..CI,
                                ymax = Upper.95..CI),
```

```
                                    lwd = 1, position = position_dodge(width = 1/2))
zp2 <- zp2 + geom_pointrange(aes(x = Variable, y = Estimate, ymin = Lower.95..CI,
                                     ymax = Upper.95..CI),
                                   lwd = 1/2, position = position_dodge(width = 1/2),
                                   shape = 21, fill = "WHITE")
zp2 <- zp2 + coord_flip() + theme_bw()
zp2 <- zp2 + ggtitle("Comparison of models")
print(zp2)  # The trick to these is position_dodge().
```