Homework Week 5: Slides

Elizabeth Brannon, Kesicia Dickinson, and Shane Wery

2/21/2018

Using polity data

```
load("/Users/elizabethbrannon/Dropbox/PLS 900/polity_dataframe.rda")
```

Slide 47

```
#create a function that can return the values we are looking for
sum.fun = function(x){
  numberNA = sum(is.na(x))
  x=x[!is.na(x)]
  n=length(x)
  return(c(n=n,
           numberNA=numberNA,
           mean=mean(x),
           median=median(x),
           max=max(x),
           min=min(x),
           stdev=sd(x)))
}



#Create functions for each test

lapply.test <- function(data){
  lapplydata <- do.call(cbind, lapply(data, sum.fun))
  row.names(lapplydata) <- (c('n','NA', 'mean', 'median', 'max', 'min', 'sd'))
  return(lapplydata)
}

sapply.test <- function(data){
  sapplydata <- sapply(data, sum.fun)
  row.names(sapplydata) <- (c('n','NA', 'mean', 'median', 'max', 'min', 'sd'))
  return(sapplydata)
}

apply.test <- function(data){
  applydata <- apply(data, 2, sum.fun)
  row.names(applydata) <- (c('n','NA', 'mean', 'median', 'max', 'min', 'sd'))
  return(applydata)
}

loop.test <- function(variable){
```

```
  loopdata <- data.frame('democ'= rep(NA, 7), 'autoc' = rep(NA), 'polity2' = rep(NA), 'xconst' =
                           rep(NA))
  for (i in variable){
    loopdata[,i] <- sum.fun(polity[,i])
  }

  row.names(loopdata) <- c("n","NA", "mean","median","max","min","sd")

  return(loopdata)
}
#create a vector for the 'i' values in the loop
variables <- c('democ','autoc','polity2','xconst')


#run benchmark
library(rbenchmark)

benchmark(replications =10,
          lapply.test(polity[,c('democ','autoc','polity2','xconst')]),
          sapply.test(polity[,c('democ','autoc','polity2','xconst')]),
          apply.test(polity[,c('democ','autoc','polity2','xconst')]),
          loop.test(variables)
)
```

```
##                                                              test
## 3  apply.test(polity[, c("democ", "autoc", "polity2", "xconst")])
## 1 lapply.test(polity[, c("democ", "autoc", "polity2", "xconst")])
## 4                                          loop.test(variables)
## 2 sapply.test(polity[, c("democ", "autoc", "polity2", "xconst")])
##   replications elapsed relative user.self sys.self user.child sys.child
## 3           10   0.062    3.263     0.061    0.001          0         0
## 1           10   0.021    1.105     0.019    0.002          0         0
## 4           10   0.024    1.263     0.024    0.000          0         0
## 2           10   0.019    1.000     0.019    0.000          0         0
```

Slide 50

```
#create an object for the new dataframe by using tapply to apply each function to the desired variables
tapply.test <- cbind(tapply(polity$polity2, polity$year, mean, na.rm=T),
               tapply(polity$xconst, polity$year, mean, na.rm=T),
               tapply(polity$polity2, polity$year, median, na.rm=T),
               tapply(polity$xconst, polity$year, median, na.rm=T),
               tapply(polity$polity2, polity$year, sd, na.rm=T),
               tapply(polity$xconst, polity$year, sd, na.rm=T)
)
#add column names
colnames(tapply.test) <- c('democ_mean','xconst_mean',
                           'democ_median','xconst_median',
                           'democ_sd', 'xconst_sd')
tapply.test
```

```
##      democ_mean xconst_mean democ_median xconst_median democ_sd xconst_sd
```

```
## 1960 -0.6454545    3.803738        -3.5          3.0 7.628515  2.372958
## 1961 -0.7719298    3.781818        -3.5          3.0 7.516395  2.367137
## 1962 -0.7627119    3.842105        -3.5          3.0 7.493220  2.348334
## 1963 -1.0756303    3.684211        -3.0          3.0 7.396930  2.320802
## 1964 -1.1157025    3.710526        -4.0          3.0 7.339607  2.314723
## 1965 -1.1639344    3.661017        -5.0          3.0 7.503565  2.339626
## 1966 -1.2640000    3.565574        -6.0          3.0 7.501768  2.349565
## 1967 -1.5634921    3.451613        -6.0          3.0 7.446337  2.342103
## 1968 -1.4108527    3.452381        -6.0          3.0 7.456763  2.371859
## 1969 -1.7054264    3.336000        -7.0          3.0 7.379663  2.324290
## 1970 -1.6946565    3.377953        -6.0          3.0 7.344169  2.322865
## 1971 -2.1194030    3.276923        -7.0          3.0 7.308502  2.322840
## 1972 -2.4740741    3.196970        -7.0          3.0 7.387797  2.371714
## 1973 -2.4117647    3.261194        -7.0          3.0 7.448063  2.393378
## 1974 -2.2919708    3.270677        -7.0          3.0 7.392525  2.374511
## 1975 -2.2447552    3.316547        -7.0          3.0 7.371640  2.334480
## 1976 -2.4895105    3.255319        -7.0          3.0 7.405619  2.349481
## 1977 -2.5174825    3.230216        -7.0          3.0 7.389423  2.350814
## 1978 -2.0694444    3.391304        -7.0          3.0 7.416343  2.328606
## 1979 -1.6382979    3.514925        -6.0          3.0 7.555770  2.341952
## 1980 -1.8591549    3.419118        -6.5          3.0 7.509419  2.352477
## 1981 -1.8943662    3.416058        -6.5          3.0 7.416397  2.328349
## 1982 -1.8169014    3.460432        -7.0          3.0 7.517940  2.356816
## 1983 -1.6126761    3.532374        -6.0          3.0 7.562364  2.363068
## 1984 -1.6830986    3.485714        -6.0          3.0 7.535608  2.339789
## 1985 -1.4366197    3.536232        -6.0          3.0 7.582609  2.365073
## 1986 -1.3169014    3.589928        -6.0          3.0 7.645858  2.395082
## 1987 -1.2605634    3.633094        -6.0          3.0 7.612680  2.371766
## 1988 -0.9860140    3.695035        -6.0          3.0 7.609285  2.393274
## 1989 -0.5244755    3.791367        -4.0          3.0 7.545013  2.381921
## 1990  0.5833333    4.151079         0.5          4.0 7.517346  2.358806
## 1991  1.3375000    4.408163         2.0          5.0 7.088735  2.271790
## 1992  2.0062500    4.554054         5.0          5.0 6.998200  2.247550
## 1993  2.2085890    4.588235         5.0          5.0 6.873149  2.196077
## 1994  2.4876543    4.662338         5.0          5.0 6.796270  2.157888
## 1995  2.4596273    4.623377         5.0          5.0 6.774948  2.160483
## 1996  2.3478261    4.605096         5.0          5.0 6.900417  2.183146
## 1997  2.3291925    4.563291         5.0          5.0 6.895448  2.198533
## 1998  2.4844720    4.619355         5.0          5.0 6.747319  2.151022
## 1999  2.7018634    4.647059         5.0          5.0 6.616121  2.150491
## 2000  2.9440994    4.714286         5.0          5.0 6.566247  2.094856
## 2001  3.1500000    4.779221         6.0          5.0 6.549473  2.102743
## 2002  3.2608696    4.839744         6.0          5.0 6.568411  2.105283
## 2003  3.2893082    4.844156         6.0          5.0 6.539409  2.102379
## 2004  3.4187500    4.896104         6.0          5.5 6.589941  2.136422
## 2005  3.6335404    4.961538         6.0          6.0 6.471755  2.091101
## 2006  3.7177914    5.006211         6.0          6.0 6.468832  2.038679
## 2007  3.6851852    4.981250         6.0          6.0 6.441782  2.045003
## 2008  3.7975460    5.012422         7.0          6.0 6.405206  2.046300
## 2009  3.7852761    5.024845         6.0          6.0 6.338928  2.024692
## 2010  3.8414634    5.037500         6.0          6.0 6.256225  2.012188
## 2011  4.0121212    5.086420         6.0          6.0 6.172310  1.976241
## 2012  4.0000000    5.125000         6.0          6.0 6.203854  1.977070
## 2013  4.1636364    5.181250         7.0          6.0 6.157774  1.948623
```

```
## 2014  4.0963855    5.180124          6.0           6.0 6.146917  1.945791
## 2015  4.1927711    5.243750          7.0           6.0 6.127846  1.941752
## 2016  4.2000000    5.253086          7.0           6.0 6.141820  1.941094
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#use dplyr summarise command to do the work of tapply. dplyr allows us to group by year prior to the su
polity %>%
  group_by(year)%>%
  summarise(democ_mean = mean(polity2, na.rm=T),
            xconst_mean = mean(xconst, na.rm=T),
            democ_median = median(polity2, na.rm=T),
            xconst_median = median(xconst, na.rm=T),
            democ_sd = sd(polity2, na.rm=T),
            xconst_sd = sd(xconst, na.rm=T)
  )
```

```
## # A tibble: 57 x 7
##     year democ_mean xconst_mean democ_median xconst_median democ_sd
##    <int>      <dbl>       <dbl>        <dbl>         <dbl>    <dbl>
##  1  1960 -0.6454545    3.803738         -3.5             3 7.628515
##  2  1961 -0.7719298    3.781818         -3.5             3 7.516395
##  3  1962 -0.7627119    3.842105         -3.5             3 7.493220
##  4  1963 -1.0756303    3.684211         -3.0             3 7.396930
##  5  1964 -1.1157025    3.710526         -4.0             3 7.339607
##  6  1965 -1.1639344    3.661017         -5.0             3 7.503565
##  7  1966 -1.2640000    3.565574         -6.0             3 7.501768
##  8  1967 -1.5634921    3.451613         -6.0             3 7.446337
##  9  1968 -1.4108527    3.452381         -6.0             3 7.456763
## 10  1969 -1.7054264    3.336000         -7.0             3 7.379663
## # ... with 47 more rows, and 1 more variables: xconst_sd <dbl>
```