Poli 5D Social Science Data Analytics

R: Loops

Shane Xinyang Xuan
ShaneXuan.com

March 3, 2017

Contact Information

Shane Xinyang Xuan

xxuan@ucsd.edu

The teaching staff is a team!

Professor Roberts	M	1600-1800 (SSB 299)
Jason Bigenho	Th	1000-1200 (Econ 116)
Shane Xuan	M	1100-1150 (SSB 332)
	Th	1200-1250 (SSB 332)

Supplemental Materials

UCLA STATA starter kit

`http://www.ats.ucla.edu/stat/stata/sk/`

Princeton data analysis

`http://dss.princeton.edu/training/`

Road map

Some quick notes before we start today's section:

- Make sure that you pass around the attendance sheet

Road map

Some quick notes before we start today's section:

- Make sure that you pass around the attendance sheet
- We will talk about loops in R

For Loops in R

- Syntax: for (i in X) will create a loop in R

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter
  - X is the vector of values that the counter takes on

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter
  - X is the vector of values that the counter takes on
- Example: for (i in 1:n)

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter
  - X is the vector of values that the counter takes on
- Example: for (i in 1:n)
  - In each iteration, the counter i takes an integer value

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter
  - X is the vector of values that the counter takes on
- Example: for (i in 1:n)
  - In each iteration, the counter i takes an integer value
  - i starts with 1, and ends with n, with an increment of 1

For Loops in R

- Syntax: for (i in X) will create a loop in R
  - i is the counter
  - X is the vector of values that the counter takes on
- Example: for (i in 1:n)
  - In each iteration, the counter i takes an integer value
  - i starts with 1, and ends with n, with an increment of 1
- Syntax: We use the curly brackets {} to denote a block of code in a function

Example: For Loops in R

Example 1

```
for (year in 2010:2017){
  print(paste("The year is", year))
}
```

Example: For Loops in R

Example 1

```
for (year in 2010:2017){
  print(paste("The year is", year))
}
```

Example 2

```
n = 10
x = rep(0,n)
for (j in 1:n){
  x[j] = j^2
}
```

Example: For Loops in R

Example 1

```
for (year in 2010:2017){
  print(paste("The year is", year))
}
```

Example 2

```
n = 10
x = rep(0,n)
for (j in 1:n){
  x[j] = j^2
}
```

Result

```
> x
[1]   1   4   9  16  25  36  49  64  81 100
```

If Statement in R

Example 3

```
x <- 2
if(x > 0){
  print("Non-negative number")
} else {
  print("Negative number")
}
```

- Syntax: if (logical){...} runs the block of code in the curly brackets when the logical statement in the parenthesis is TRUE

If Statement in R

Example 3

```
x <- 2
if(x > 0){
  print("Non-negative number")
} else {
  print("Negative number")
}
```

- Syntax: if (logical){...} runs the block of code in the curly brackets when the logical statement in the parenthesis is TRUE
- Syntax: else{...} runs the block of code in the following curly brackets if the logical statement in the previous parenthesis is FALSE

If Statement in R

Example 3

```
x <- 2
if(x > 0){
  print("Non-negative number")
} else {
  print("Negative number")
}
```

- Syntax: if (logical){...} runs the block of code in the curly brackets when the logical statement in the parenthesis is TRUE
- Syntax: else{...} runs the block of code in the following curly brackets if the logical statement in the previous parenthesis is FALSE
- Recall the ifelse() function:
  `ifelse(x>0, "Non-negative number", "Negative number")`

Data: Loops in R

Task 1: We will first look at the poll data that have been discussed in lecture on Wednesday

Data: Loops in R

Task 1: We will first look at the poll data that have been discussed in lecture on Wednesday

Task 2: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

Data: Loops in R

Task 1: We will first look at the poll data that have been discussed in lecture on Wednesday

Task 2: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- Load the data:

```
library(foreign)
data <- read.dta("ajr.dta")
```

Data: Loops in R

Task 1: We will first look at the poll data that have been discussed in lecture on Wednesday

Task 2: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- Load the data:

```
library(foreign)
data <- read.dta("ajr.dta")
```

- Set up empty columns:

```
set.seed(1994)
n <- 1000
alpha <- rep(NA, n)
beta <- rep(NA, n)
```

Data: Loops in R

Task 1: We will first look at the poll data that have been discussed in lecture on Wednesday

Task 2: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- Load the data:

```
library(foreign)
data <- read.dta("ajr.dta")
```

- Set up empty columns:

```
set.seed(1994)
n <- 1000
alpha <- rep(NA, n)
beta <- rep(NA, n)
```

- Set up plot:

```
plot(data$logem4, data$logpgp95, xlim=c(1,8), ylim=c(6,12),
  xlab="Log Settler Mortality", ylab="Log GDP pc growth",
  pch=19, cex=0.5, col="skyblue1")
```

Data: Loops in R (2)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

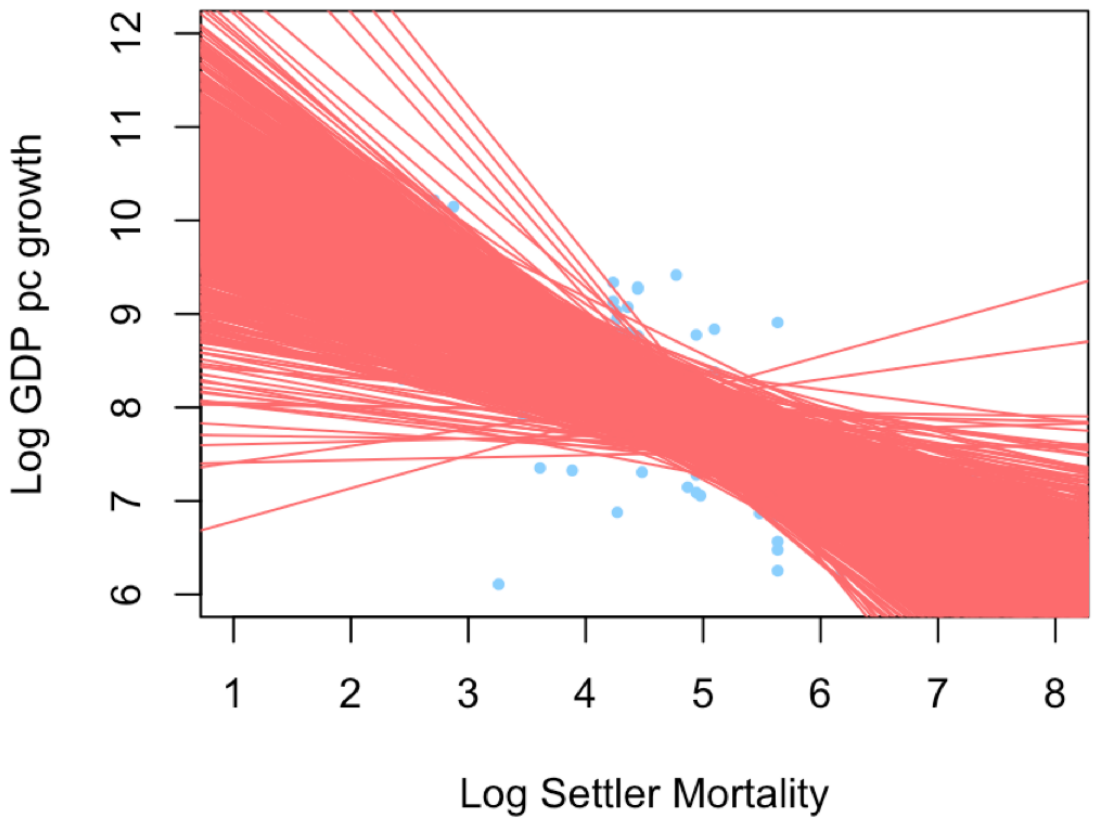
- Loops:

```
for(i in 1:n){
  # sample 30 observations with replacement from
  # the data.frame observations are in rows
  sample <- data[sample(1:nrow(data), size=30, replace=T),]
  # regress GDP growth on mortality using lm(y~x)
  reg <- lm(sample$logpgp95~sample$logem4)
  # we can plot a regression line for each sample
  abline(reg, col="indianred1", lwd=1)
  # save objects in the empty columns we created
  # check 'the magic of i' part in our lecture slides
  alpha[i] <- coef(reg)[1]
  beta[i] <- coef(reg)[2]
}
```

Data: Loops in R (3)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

Figure: Looping example in R



Data: Loops in R (4)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- What happened to the columns alpha and beta?

Data: Loops in R (4)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- What happened to the columns alpha and beta?

```
> length(alpha)
[1] 1000
```

Data: Loops in R (4)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- What happened to the columns alpha and beta?

```
> length(alpha)
[1] 1000
> head(alpha, n=20)
[1] 11.081918 10.378140 10.385019 10.748861 11.153170 11.787
[11]  8.717526 11.570535 10.282285 10.099646 12.625225 10.79
```

Data: Loops in R (4)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- What happened to the columns alpha and beta?

```
> length(alpha)
[1] 1000
> head(alpha, n=20)
[1] 11.081918 10.378140 10.385019 10.748861 11.153170 11.787
[11]  8.717526 11.570535 10.282285 10.099646 12.625225 10.79
> length(beta)
[1] 1000
```

Data: Loops in R (4)

Task: We are going to sample some data, fit a regression on the sampled data, and plot the fitted line. We then repeat this process for 1,000 times!

- What happened to the columns alpha and beta?

```
> length(alpha)
[1] 1000
> head(alpha, n=20)
[1] 11.081918 10.378140 10.385019 10.748861 11.153170 11.787
[11]  8.717526 11.570535 10.282285 10.099646 12.625225 10.79
> length(beta)
[1] 1000
> head(beta, n=20)
 [1] -0.7092444 -0.5168424 -0.5167437 -0.4563491 -0.6308942
 [10] -0.5552498 -0.1817991 -0.7385015 -0.4492643 -0.4645350
 [19] -0.5573325 -0.7314316
```

What did we learn from the previous exercise?

- Load a package

What did we learn from the previous exercise?

- Load a package
- for loop

What did we learn from the previous exercise?

- Load a package
- for loop
- Save objects during loops

What did we learn from the previous exercise?

- Load a package
- for loop
- Save objects during loops
- Random sampling

What did we learn from the previous exercise?

- Load a package
- for loop
- Save objects during loops
- Random sampling
- Add a regression line in R using abline()

What did we learn from the previous exercise?

- Load a package
- for loop
- Save objects during loops
- Random sampling
- Add a regression line in R using abline()
- The magic of i

What did we learn from the previous exercise?

- Load a package
- for loop
- Save objects during loops
- Random sampling
- Add a regression line in R using abline()
- The magic of i

Announcement:
Problem set 3 due at midnight – I will see you next week!