Math Camp II

R Session

Yiqing Xu

MIT

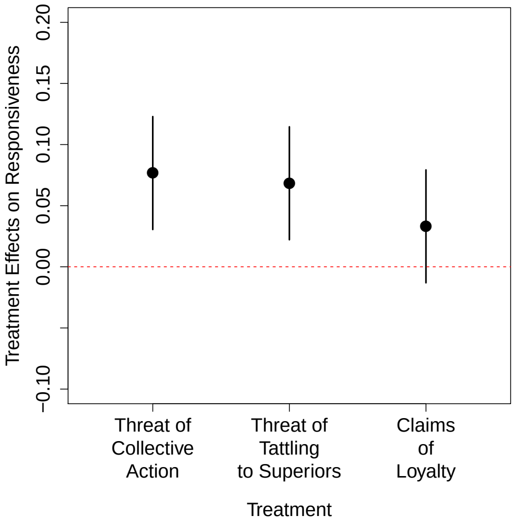
August 29, 2014

1 Simple Plots

2 Parallel Computing

3 Accessing Remote Servers

Plot 1 Treatment Effects



Plot 1 Treatment Effects

R Code

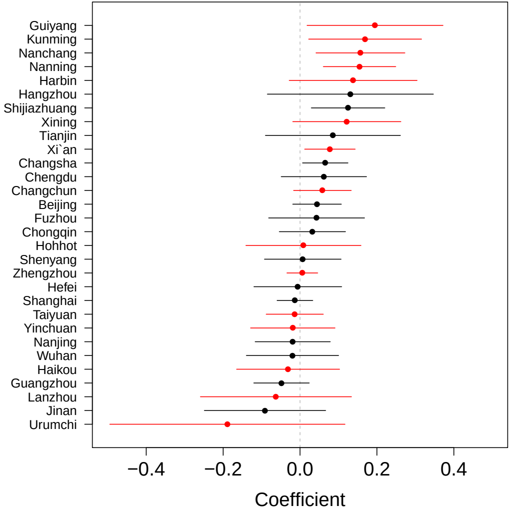
```
par(mar=c(8, 4, 1, 2) + 0.1)

plot(1:3,res.t[,1],pch=16,cex=2,ylim=c(-0.1,0.2),xlim=c(0.5, 3.5),
     main = "",xlab="",ylab="",axes=FALSE)
for(i in 1:3)  segments(i,res.t[i,2],i,res.t[i,3],lwd=2)
abline(h=0,lty=2,col=2)

box()
axis(1,1:3,labels=c("...","...","..."),cex.axis=1.5,mgp=c(3,4,0))
axis(2,cex.axis=1.5)

mtext("Treatment",side=1,line=6.5,cex=1.5)
mtext("Treatment Effects on Responsiveness",side=2,line=3,cex=1.5)
```

Plot 2 Heterogeneous Treatment Effects



Plot 2 Heterogeneous Treatment Effects

R Code

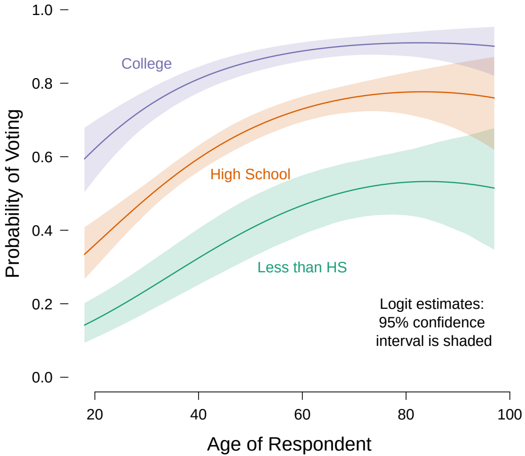
```
d <- d[order(d$est),]

par(mar=c(4,6,2,2),las=1)   # label turn 90 degree
plot(1,axes=F,xlab="",ylab="",xlim=c(-0.5,0.5),ylim=c(0.5,30.5))

abline(v=0,lty=2,col="gray")
points(d$est[d$low_mkt==0],c(1:30)[d$low_mkt==0],col=1,pch=16)
points(d$est[d$low_mkt==1],c(1:30)[d$low_mkt==1],col=2,pch=16)
for (i in which(d$low_mkt==0)) lines(c(d$ci1[i],d$ci2[i]),c(i,i))
for (i in which(d$low_mkt==1)) lines(c(d$ci1[i],d$ci2[i]),c(i,i),col=2)

axis(side=1,cex.axis=1.5)
axis(side=2,at=1:30,labels=d$lab,cex.axis=1)
mtext("Coefficient",1,line=3,cex=1.5)
box()
```

Plot 3 Expected Values



This example is borrowed from Chris Adolph's short course on data visualization

Plot 3 Expected Values

R Code

```
library(RColorBrewer) # gives nice colors
col<-brewer.pal(3,"Dark2")

par(las=1)
plot(1,xlim=range(age),ylim=c(0,1),axes=F,cex.lab=1.5,
     xlab="Age of Respondents",ylab="Probability of Voting")

polygon(c(rev(age), age), c(rev(coef.nohs[,3]), coef.nohs[,2]),
        col = paste(col[1],"30",sep=""), border = NA)
lines(age,coef.nohs[,1],col=col[1],lwd=1.5)
...

axis(1,cex.axis=1.2)
axis(2,cex.axis=1.2,lwd=0,lwd.ticks = 1)
text(60,0.30,"Less than HS",cex=1.2,col=col[1])
...
```

1. Simple Plots
2. Parallel Computing
3. Accessing Remote Servers

Why Going Parallel

- It is necessary
  - Before too long, you'll need to perform some tasks repeatedly
  - In statistics, some (re)sampling techniques become more and more popular, e.g., bootstrap, Monte Carlo, MCMC, etc.
- It is feasible
  - Now almost everyone have computers with multiple cores
  - Cloud computing clusters have many nodes

Parallel Computing in R

- There are multiple ways of parallel computing using `R`
- Many of them are really simple
- Packages like `foreach`, `snowfall` are readily available
- We will be using the `foreach` package in this short introduction
- If you can do loop (`for`), you can do parallel computing (`foreach`)

- Check out Using The `foreach` Package
- In our example, `foreach` is the frontend, `doParallel` is the backend

A Brief Intro

Most parallel computing procedures involve the following three steps:

1. Split the problem into pieces
2. Execute the pieces in parallel
3. Combine the results back together

`foreach` does the above all together

In our exercise, we need other two steps

0. At the beginning, register clusters
4. In the end, summarize the results

R code

R Code

```
library(doParallel)
library(foreach)

# register clusters
cl<-makeCluster(4)
registerDoParallel(cl)

# loop (split, excite, and combine)
result<-foreach (i=1:sims,.combine=c,.inorder=FALSE) \%dopar\% {
  ...
  return(out)
}
stopCluster(cl)

# summarize
mean(result)
sd(result)
```

R code

R Code

```
# loop (split, excite, and combine)
result<-foreach (i=1:sims,.combine=c,.inorder=FALSE) %dopar% {
  ...
  return(out)
}
```

In the above example

1. `i=1:sims` splits the tasks
2. `%dopar%` executes each task
3. `.combine=c` combines the results

The .combine Option

`foreach` can:

1. combine numbers to a vector: `.combine=c`
2. combine vectors to a matrix: `.combine=rbind`; `.combine=cbind`
3. combine matrix to an array (of three dimensions): need to define a new function, e.g. `.combine=f()`

```
library(abind)
f <- function(){
  function(...) abind(...,along=3)
}
```

- We will have examples for each of the scenarios
- Time the algorithm using `Sys.time()` and see the difference
- That's it!

1 Simple Plots

2 Parallel Computing

3 Accessing Remote Servers

Reasons for Using Remote Servers

- It is useful
  - Your computer not powerful enough (it just cannot do it)
  - Your computer not fast enough
  - You need your computer for some other purposes (e.g., R, Word, Skype, Netflix...)

- It is feasible
  - More and more available cheap/free cloud computing services e.g., HMDC, Amazon (EC2), Microsoft
  - MIT Athena is free, but it is not designed for cloud computation
  - We use it as an example; it is at least as powerful as your laptop

Connecting to Athena

First, we need to remotely display a unix windows

- If you are a windows user, you'll need to download `SecureCRT`
- If you are a Unix/Mac OS user, you can work directly in the console

Unix/Mac OS User

- UNIX/Linux users and users of the Mac OS X "Terminal" application can also use the command line programs "ssh", "scp"
- Dial-up

  ```
  ssh username@athena.dialup.mit.edu
  ```
- Upload:

  ```
  scp file1 file2 username@ftp.dialup.mit.edu:/mit/username
  ```
- Download

  ```
  scp username@ftp.dialup.mit.edu:/mit/username/filename local_file
  ```

Getting SecureCRT/FX to Work

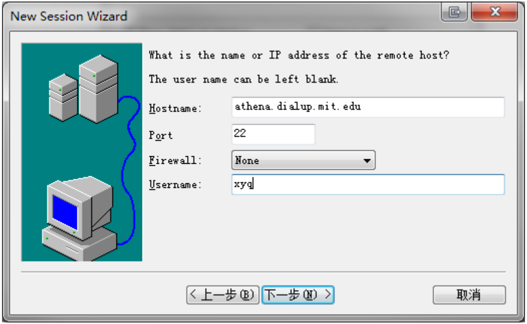
Download SecureCRT from MIT IS&T: http://ist.mit.edu/securecrt-fx and install it

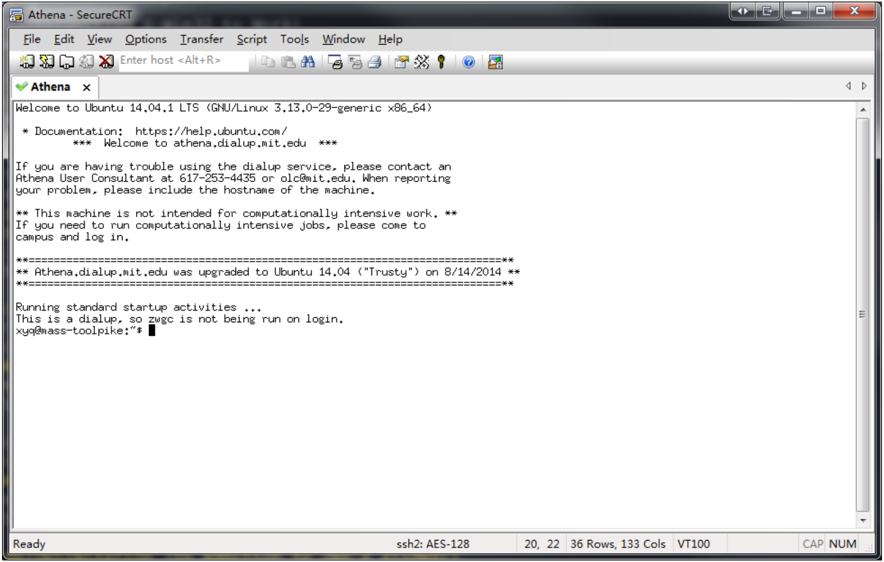
Configurations:

- New connection
- Hostname: `athena.dialup.mit.edu`
- User name: your Kerberos username
- Type of connection: `SSH2`
- File Transfer: `SFTP`
- Session name: whatever

You'll arrive at a unix console

Getting SecureCRT/FX to Work

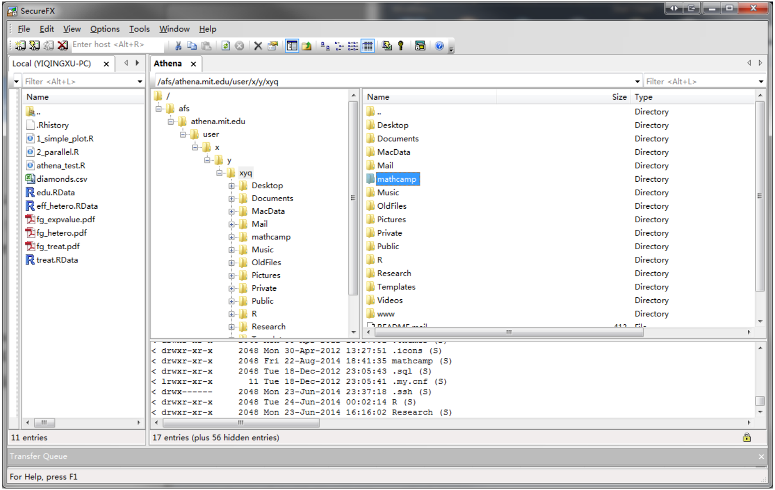




Make a new directory: `$ mkdir mathcamp`

Open SecureFX for Data Transfer

Open Secure FX and Connect to Athena. You'll able to see the dir you created.



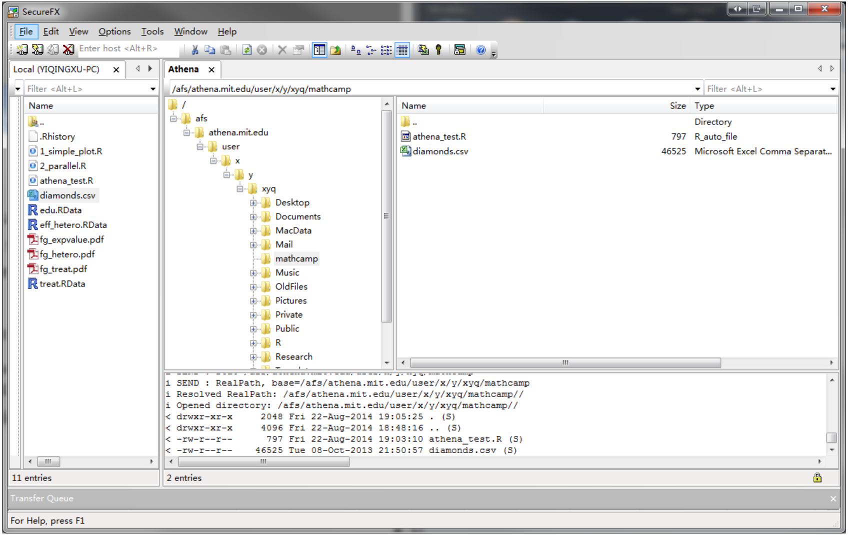
Run R Codes at Athena

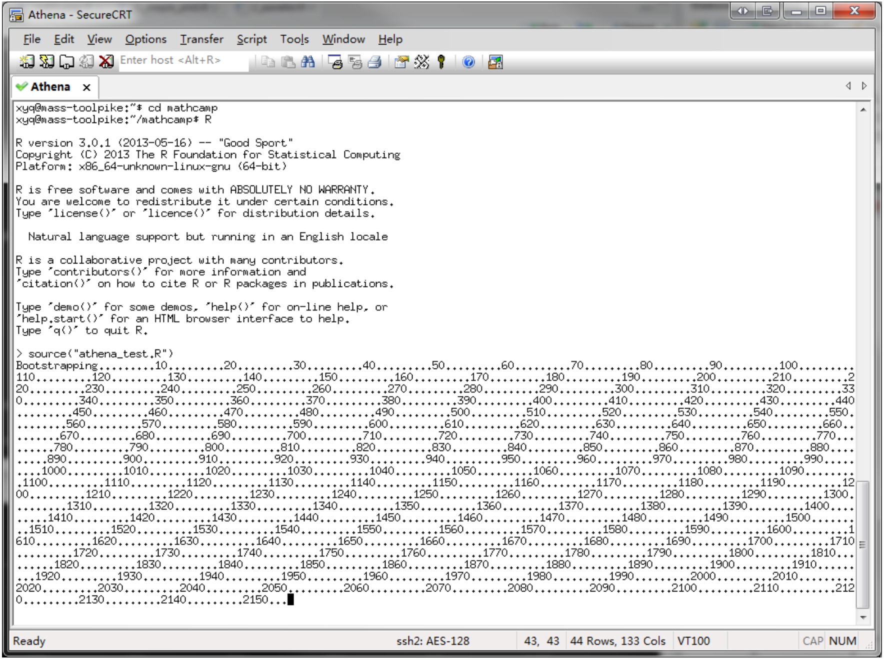
1. Create an R source file and thoroughly test it
2. Transfer the source file (and data files) to Athena using SecureFX
3. Type the following codes in the console

```
$ cd mathcamp
$ R
$ source("athena_test.R")
```

   - Remember to install packages on the server if necessary
   - Be careful about using multiple nodes; the server manager may call you

4. Copy back the stored file or analyse it in the console
5. That's it!


SecureFX
File Edit View Options Tools Window Help
Enter host <Alt+R>
Local (YIQINGXU-PC)
Athena
Filter <Alt+L>
/afs/athena.mit.edu/user/x/y/xyq/mathcamp
Name
..
.Rhistory
1_simple_plot.R
2_parallel.R
athena_test.R
diamonds.csv
edu.RData
eff_hetero.RData
fg_expvalue.pdf
fg_hetero.pdf
fg_treat.pdf
treat.RData
/
afs
athena.mit.edu
user
x
y
xyq
Desktop
Documents
MacData
Mail
mathcamp
Music
OldFiles
Pictures
Private
Public
R
Research
Name Size Type
.. Directory
athena_test.R 797 R_auto_file
diamonds.csv 46525 Microsoft Excel Comma Separat...
i SEND : RealPath, base=/afs/athena.mit.edu/user/x/y/xyq/mathcamp
i Resolved RealPath: /afs/athena.mit.edu/user/x/y/xyq/mathcamp//
i Opened directory: /afs/athena.mit.edu/user/x/y/xyq/mathcamp//
< drwxr-xr-x 2048 Fri 22-Aug-2014 19:05:25 . (S)
< drwxr-xr-x 4096 Fri 22-Aug-2014 18:48:16 .. (S)
< -rw-r--r-- 797 Fri 22-Aug-2014 19:03:10 athena_test.R (S)
< -rw-r--r-- 46525 Tue 08-Oct-2013 21:50:57 diamonds.csv (S)
11 entries
2 entries
Transfer Queue
For Help, press F1



Athena - SecureCRT
File Edit View Options Transfer Script Tools Window Help
Enter host <Alt+R>
Athena
xyq@mass-toolpike:~$ cd mathcamp
xyq@mass-toolpike:~/mathcamp$ R
R version 3.0.1 (2013-05-16) -- "Good Sport"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-unknown-linux-gnu (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
> source("athena_test.R")
Bootstrapping.........10.........20.........30.........40.........50.........60.........70.........80.........90.........100.........
110.........120.........130.........140.........150.........160.........170.........180.........190.........200.........210.........2
20.........230.........240.........250.........260.........270.........280.........290.........300.........310.........320.........33
0.........340.........350.........360.........370.........380.........390.........400.........410.........420.........430.........440
.........450.........460.........470.........480.........490.........500.........510.........520.........530.........540.........550.
........560.........570.........580.........590.........600.........610.........620.........630.........640.........650.........660..
.......670.........680.........690.........700.........710.........720.........730.........740.........750.........760.........770...
......780.........790.........800.........810.........820.........830.........840.........850.........860.........870.........880....
.....890.........900.........910.........920.........930.........940.........950.........960.........970.........980.........990.....
....1000.........1010.........1020.........1030.........1040.........1050.........1060.........1070.........1080.........1090........
.1100.........1110.........1120.........1130.........1140.........1150.........1160.........1170.........1180.........1190.........12
00.........1210.........1220.........1230.........1240.........1250.........1260.........1270.........1280.........1290.........1300.
........1310.........1320.........1330.........1340.........1350.........1360.........1370.........1380.........1390.........1400....
.....1410.........1420.........1430.........1440.........1450.........1460.........1470.........1480.........1490.........1500.......
..1510.........1520.........1530.........1540.........1550.........1560.........1570.........1580.........1590.........1600.........1
610.........1620.........1630.........1640.........1650.........1660.........1670.........1680.........1690.........1700.........1710
.........1720.........1730.........1740.........1750.........1760.........1770.........1780.........1790.........1800.........1810...
......1820.........1830.........1840.........1850.........1860.........1870.........1880.........1890.........1900.........1910......
...1920.........1930.........1940.........1950.........1960.........1970.........1980.........1990.........2000.........2010.........
2020.........2030.........2040.........2050.........2060.........2070.........2080.........2090.........2100.........2110.........212
0.........2130.........2140.........2150...
Ready
ssh2: AES-128
43, 43
44 Rows, 133 Cols
VT100
CAP NUM

`screen`

Finally, what if you want to shut down your computer (to save energy for humanity) and keep Athena working for you?

- Use `screen` command in Linux!
- Start a new screen: `screen -S xxxx`
- Kill an active screen: `Ctrl -d`
- Detach a screen without stopping it: `Ctrl -a d`
- List screens: `screen -ls`
- Reattach to an existing screen: `screen -x xxxx`
- Kill a detached screen: `screen -X -S xxxx quit`

Once you detach the working screen, you can go watching The Simpsons. Let's try!