R You Ready?

Paul E. Johnson,
Prof., Political Science
Assoc. Dir, Center for Research Methods and Data Analysis

University of Kansas

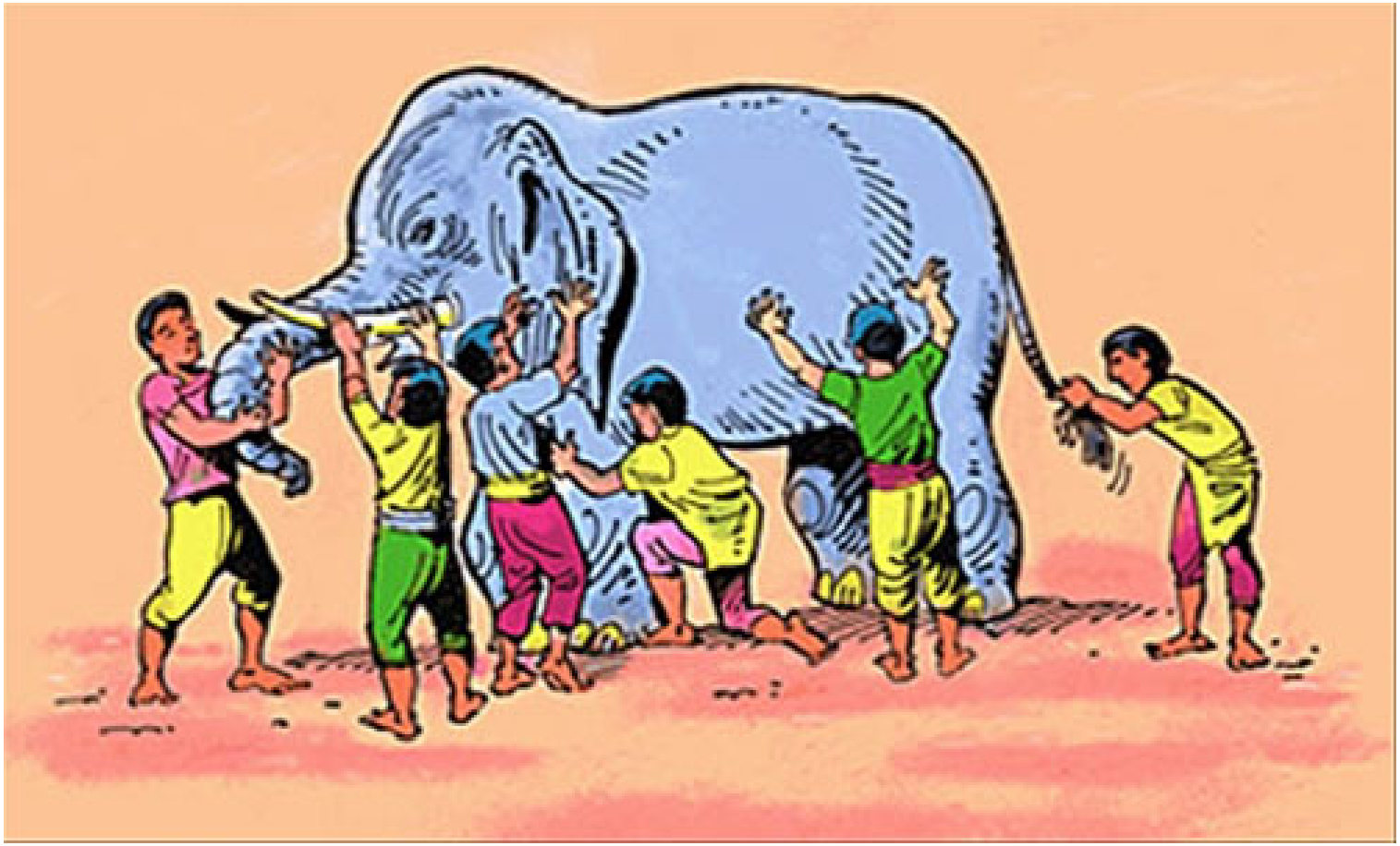
Acknowledgment: Thanks to the r-help crowd, especially Pat Burns, Deepayan Sarkar, John Fox, and Sandy Weisberg, for their useful examples

- Mission for this talk
  - Describe "R"
  - Illustrate some of its uses
- Future "hands-on" computing sessions can be scheduled.
- Alert: KU Summer Stats Camp will offer 1 week-long session on R taught by some well qualified folks :) `http://www.quant.ku.edu`

Outline

Outline

"R is a little bit like an elephant"



Ouch! That's not my Trunk!

R is

a free/open implementation of S.

a SAS/SPSS replacement for stats and graphs (salvation from Excel)

the embodiment of a new philosophy about data analysis, perhaps best exemplified by William Venables and Brian Ripley, *Modern Applied Statistics with S/R*, now in its 4th edition.

a statistical toobench for rapid model development by statisticians.

an open community of scholars who cooperate, exchange, and enhance each other's work product

Outline

What does R Taste Like? Everybody Says "Tastes like S"

- The S Language was developed at Bell Labs (mid 1970s). See Richard Becker's "Brief History of S" about the AT&T years
- *S-plus* is a commercial product that answers to S syntax commands (from the Insightful Corporation).
- There have been 4 generations of the S language.
  - Currently, S3 and S4 are in use
  - In perfect world, transition would not affect users because changes are "under the hood"

What does R Taste Like? Everybody Says "Tastes like S"

- R is a computer language
  - similar to S, but possibly better from a "computer science point of view."

  Ross Ihaka and Robert Gentleman. 1996. "R: A language for data analysis and graphics." *Journal of Computational and Graphical Statistics*, 5(3):299-314.
- R is a program that interprets scripts written in the R language
  - R also can "inter-connect" with other programs.
- R is now the "lingua franca" of research methods development. You Snooze, You Lose.

Does it matter that it is "Open Source"? YES!

- We can inspect, verify, copy, change, fix, and extend R.
- R team also elected to make R available for FREE, without charge.
- R evolves. It is an open, world-wide community of scholars.
- In R-space, nobody can hear (has to listen to) you scream (apologies to *Alien*)

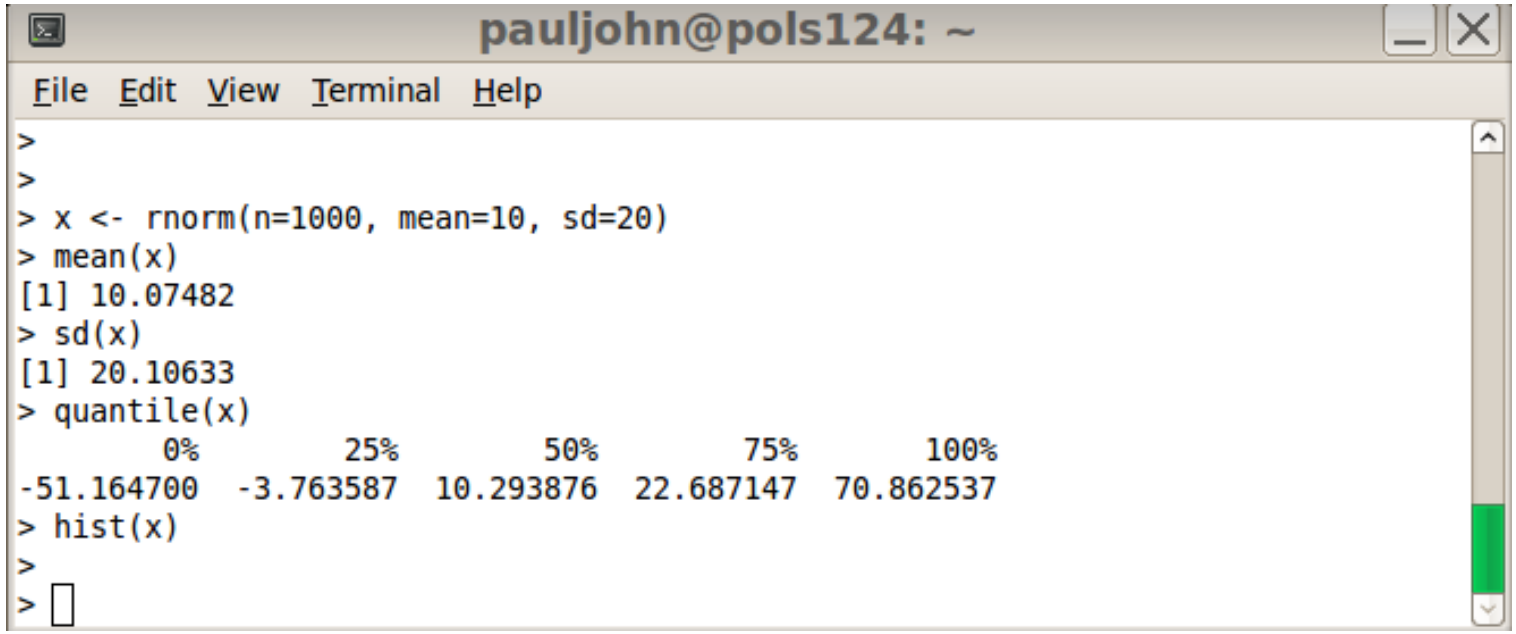
Outline

I Don't Give a Hoot about S. What is R?

- A set of ways to organize data
- All the usual statistical models
- Handy graphs
- Highly "extensible"–open to modular "packages"
- Framework for cooperation with other programs and languages

Its interactive, but not "pointy clicky"

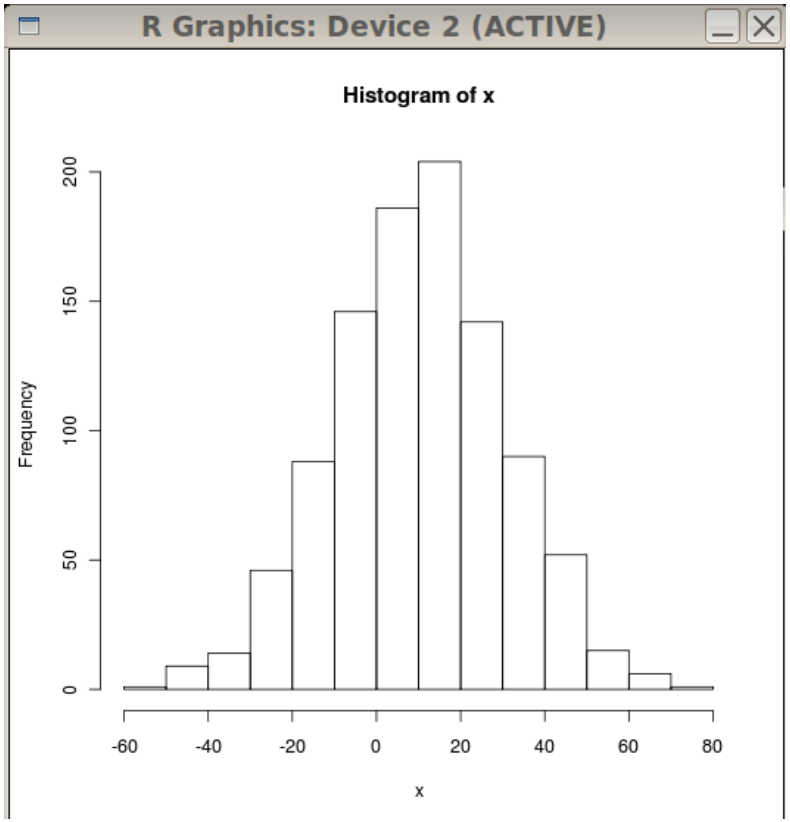
- An interactive session in R looks like this



- > is the "prompt". Type stuff there!

There might be some excitement

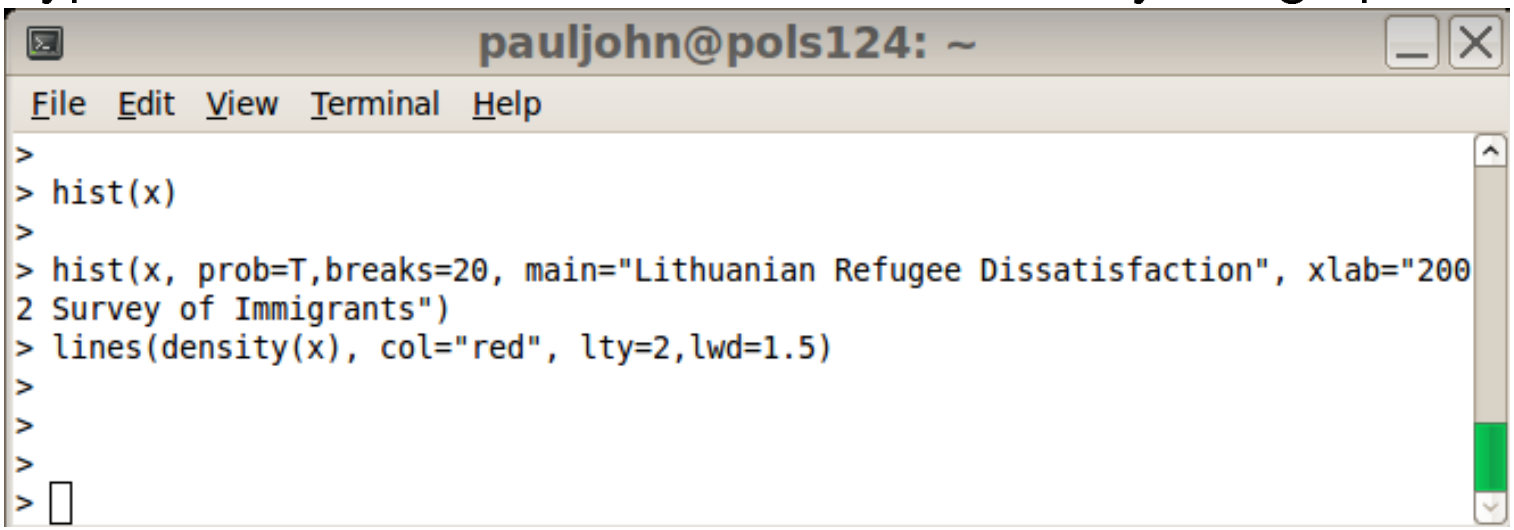
- A graph pop ups when you type "hist(x)"



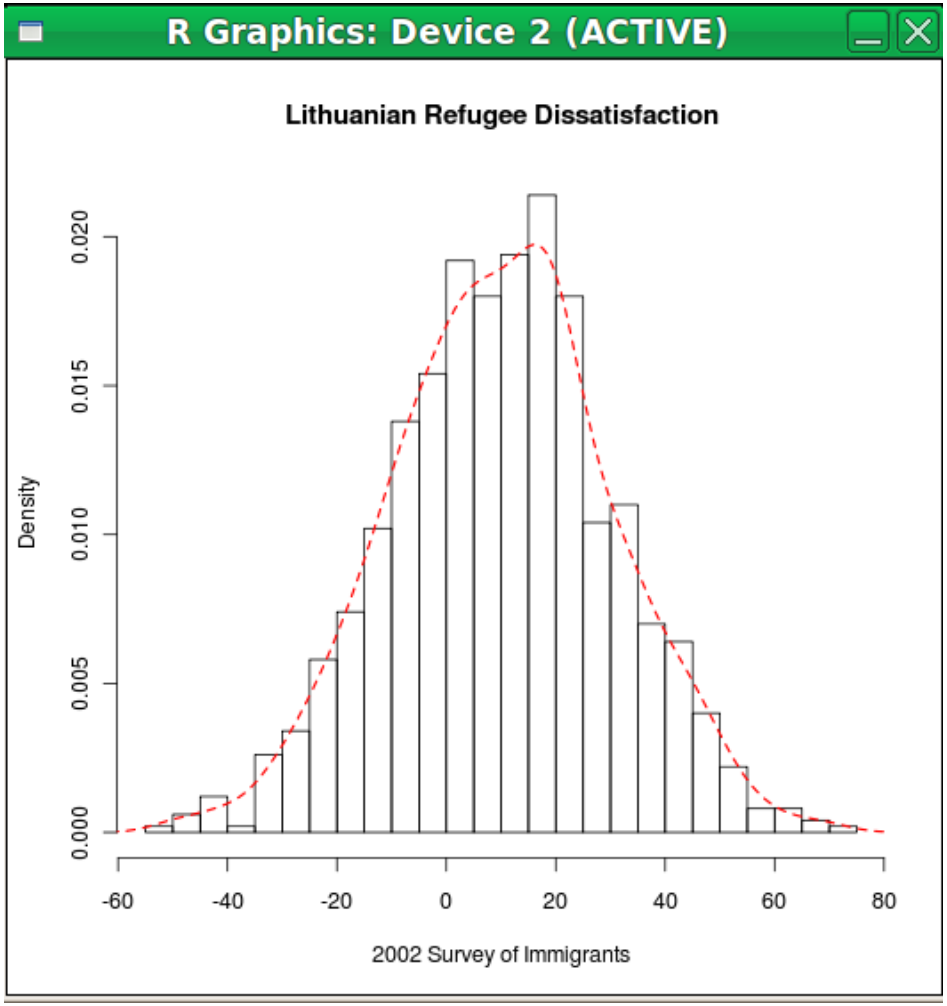
- But clicking on the graph doesn't do anything.

But you do interact with R

- Type more commands to re-draw and beautify the graph.



And a nicer looking histogram pops up



- Some GUI do exist (Rcmdr, jagr, rattle, rkward), but....

Outline

I Use R to Make Line Art

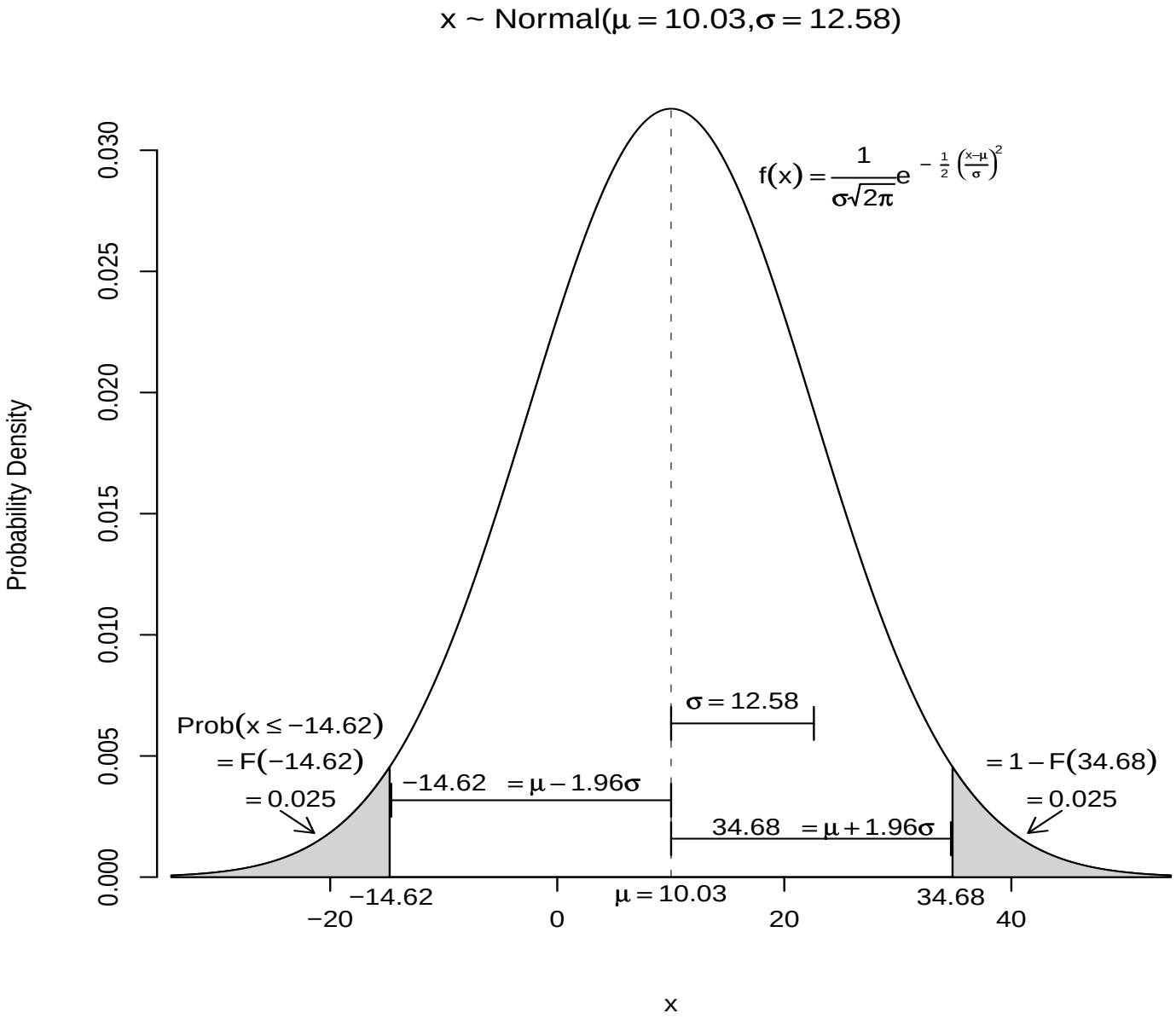
- R can create a "blank canvas"
- Which can then be decorated with subsidiary plotting commands like
  - lines
  - points
  - text
  - polygon

Hold your Seats! Prepare for the Graphic of the Century

Recall the old crowd favorite, the Normal Distribution,

$$x \sim N(\mu, \sigma^2)$$

μ is the center point of x's range, the expected value, or mean
σ is a dispersion parameter, often called the standard deviation



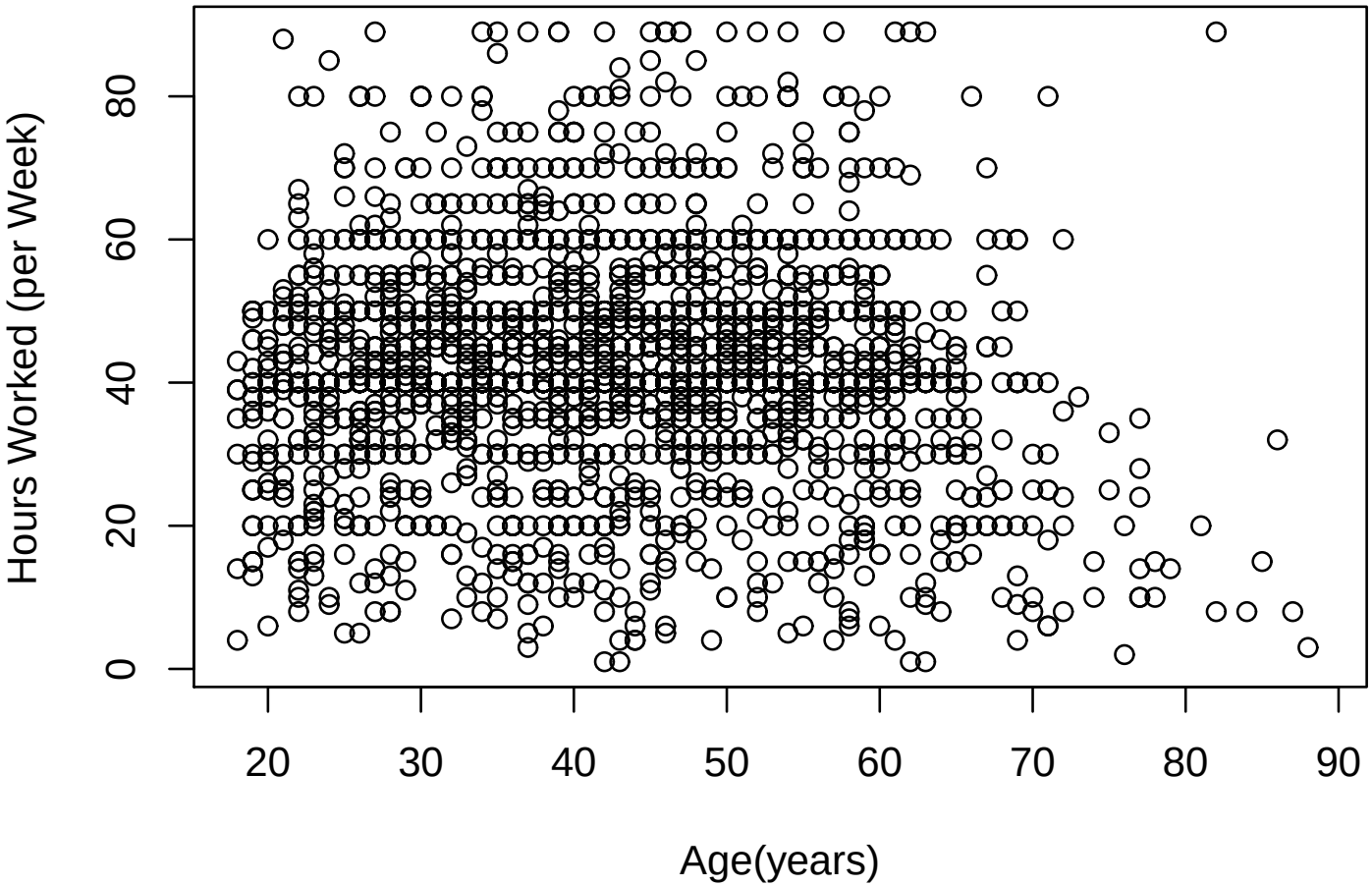
I warned you. This is one awesome figure!

Getting all Computer-science-ey now:

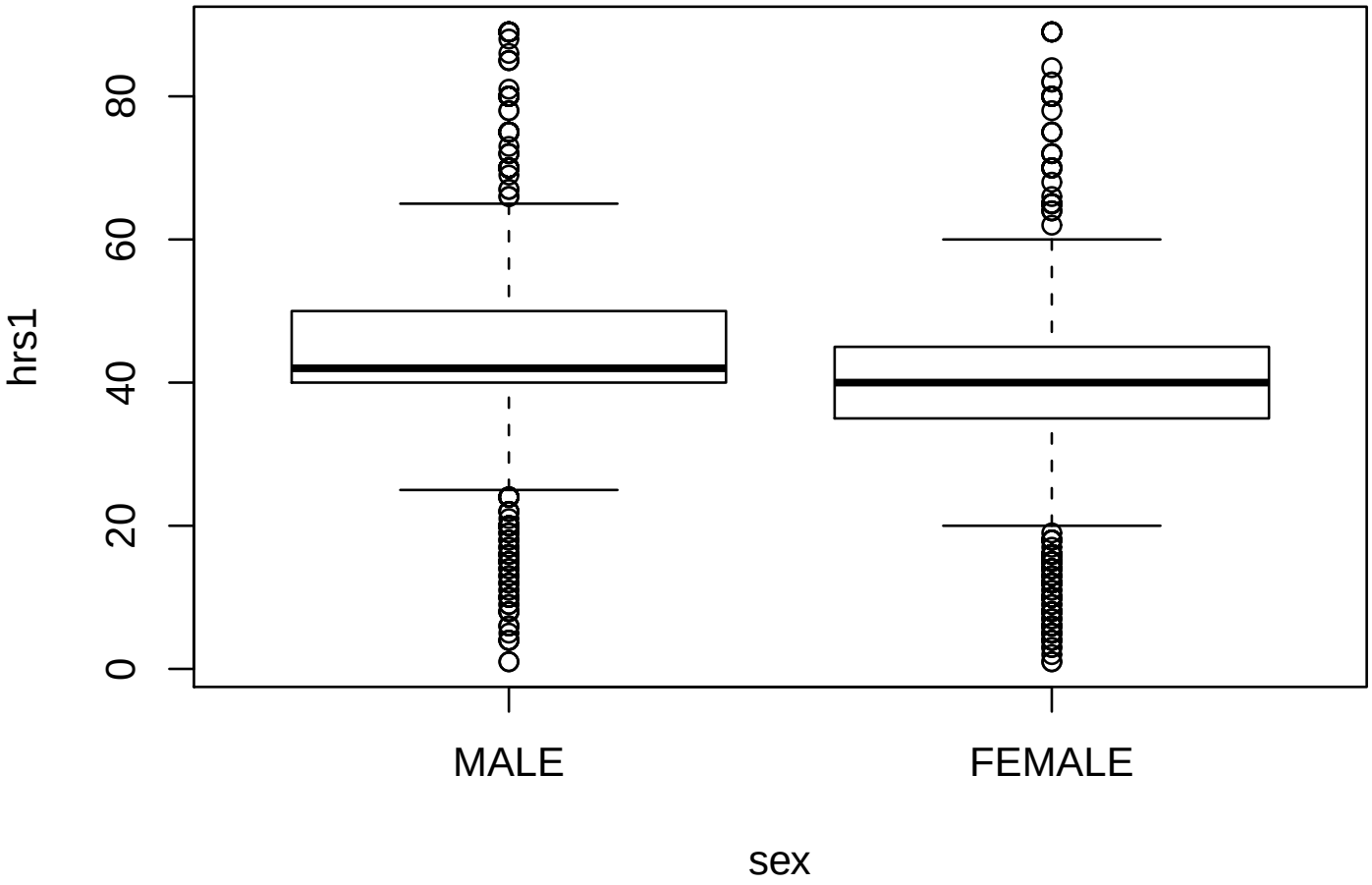
plot() is magic!
It tries to guess what you need, and it gives it to you.
R has separate methods to create

- scatterplots
- barplots
- boxplots
- spinograms
- and so forth

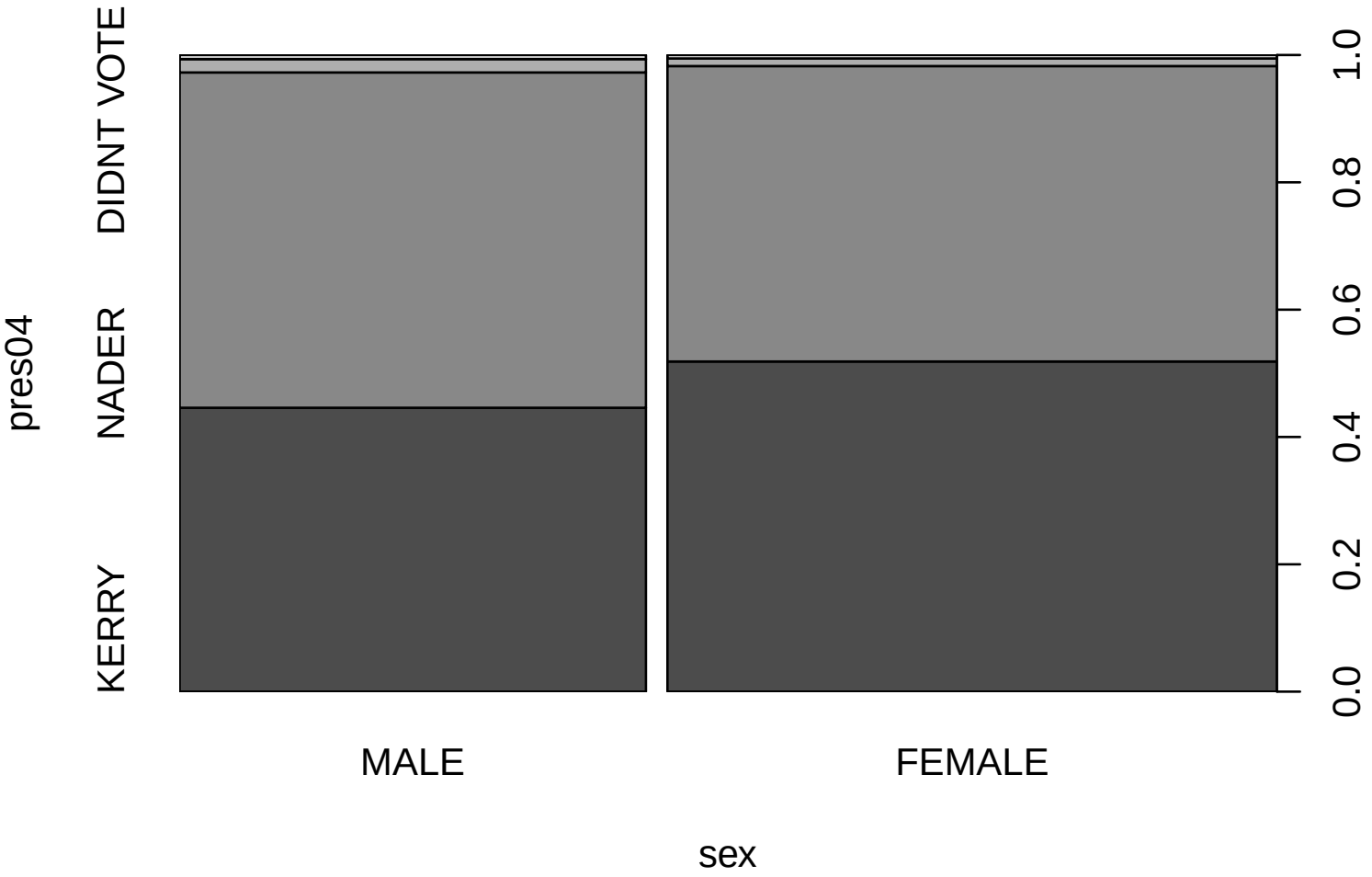
plot of 2 numeric variables $\longrightarrow$ get a scatterplot



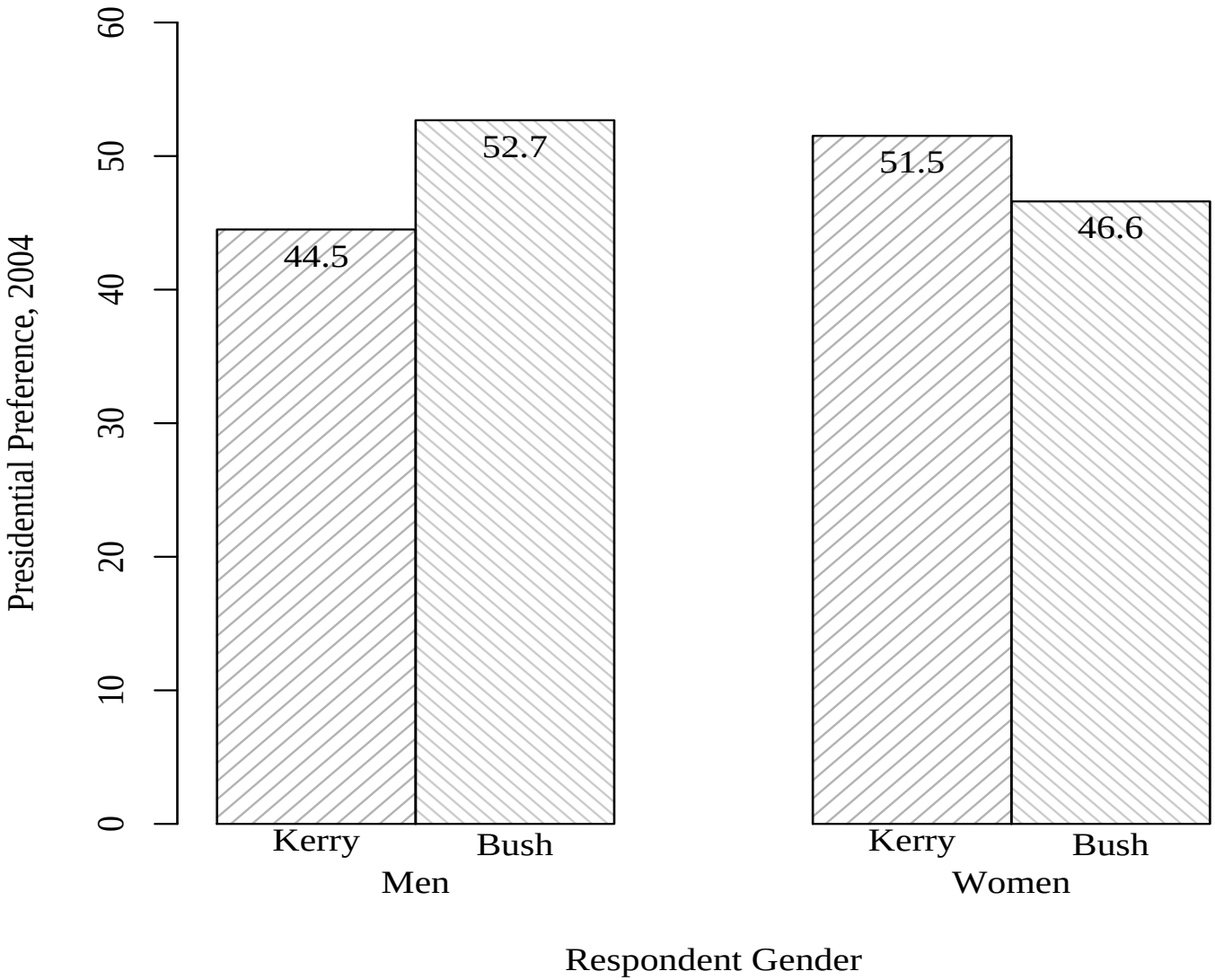
plot 1 numeric by a categorical variable, get boxplot



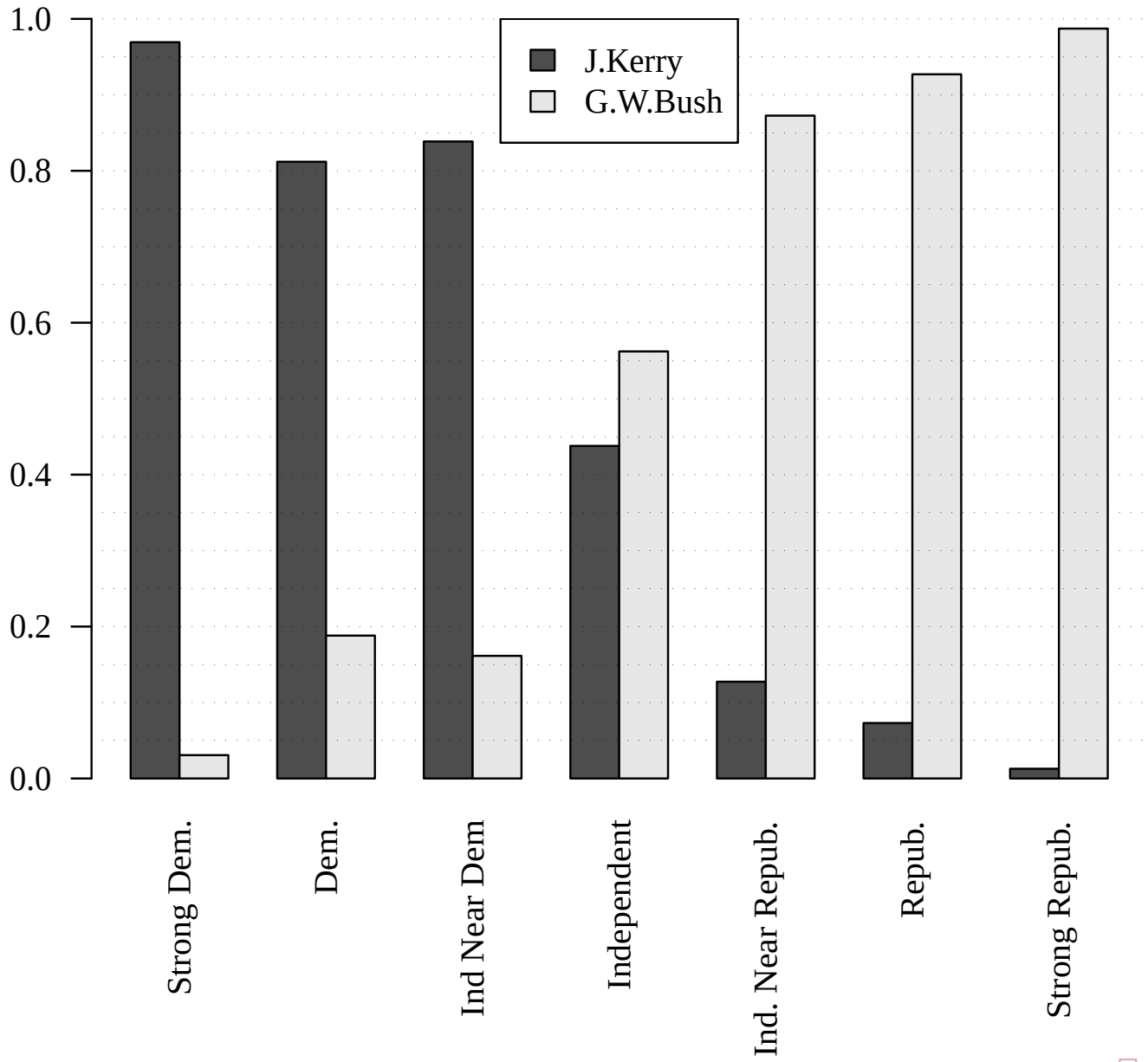
plot 2 categorical variables $\longrightarrow$ spineplot



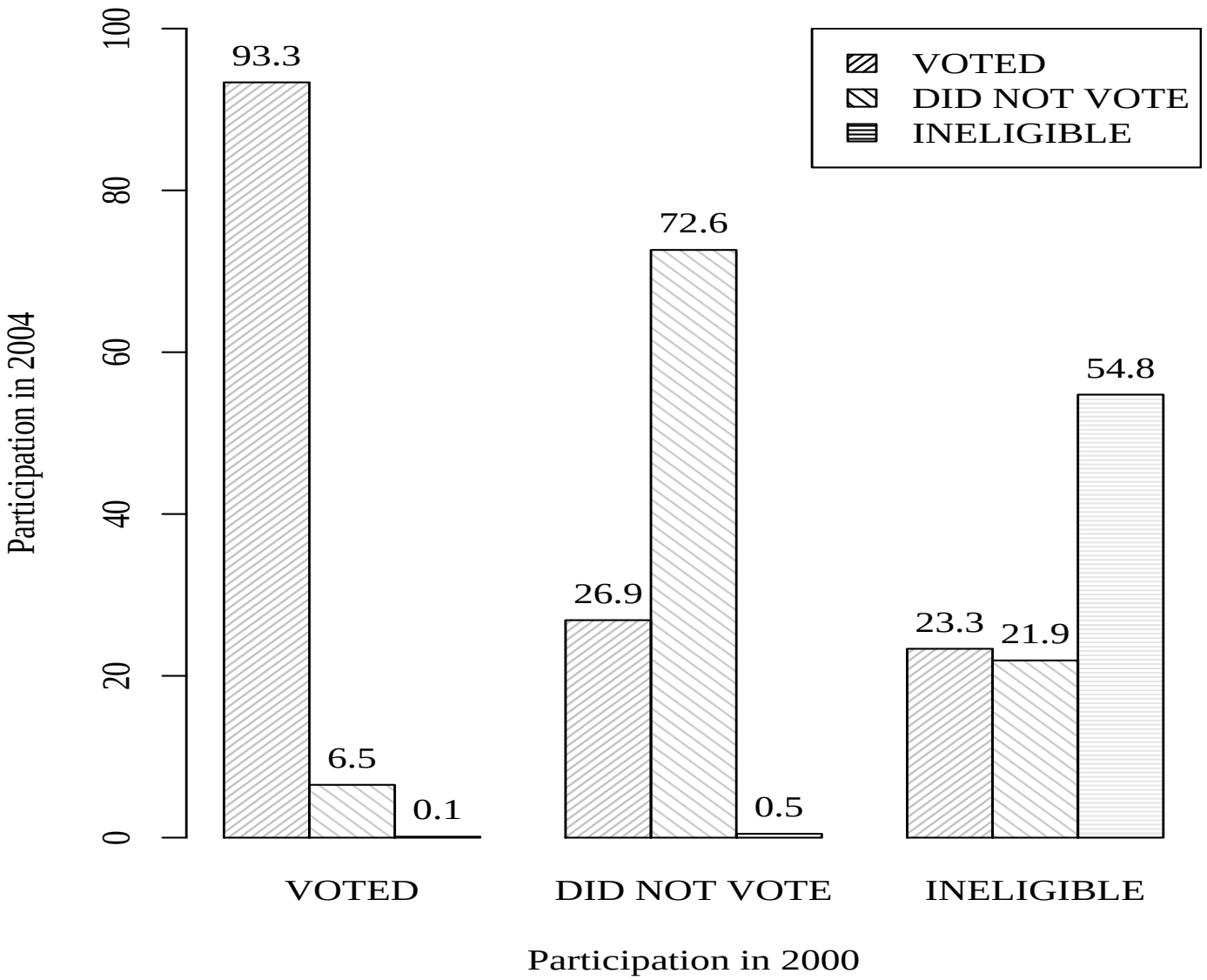
Gender Gap Prettier as a Barplot, IMHO



Best Bar Plot from POLS706 Midterm 2010



My Best Barplot from the POLS706 Midterm, 2009

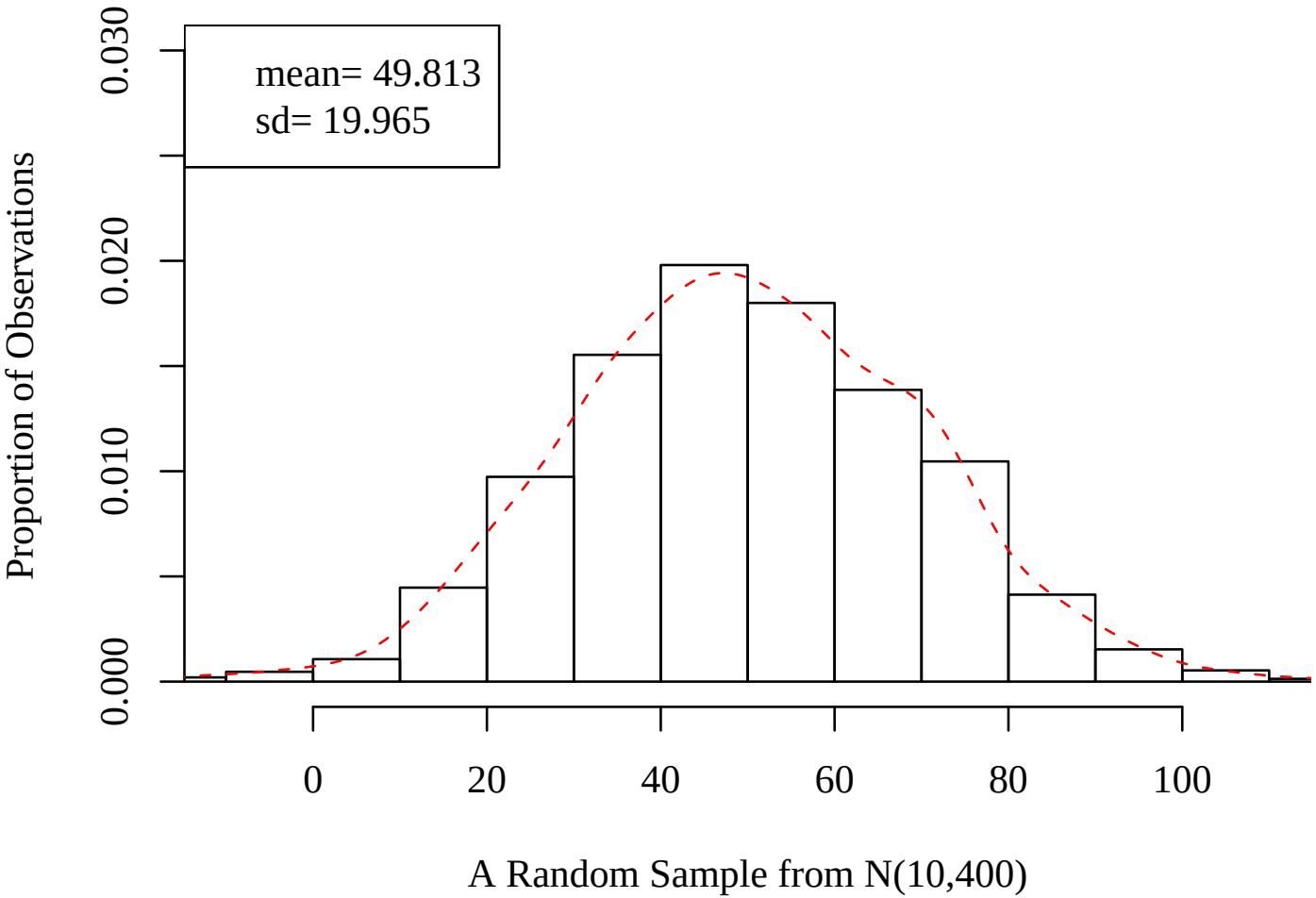


Outline

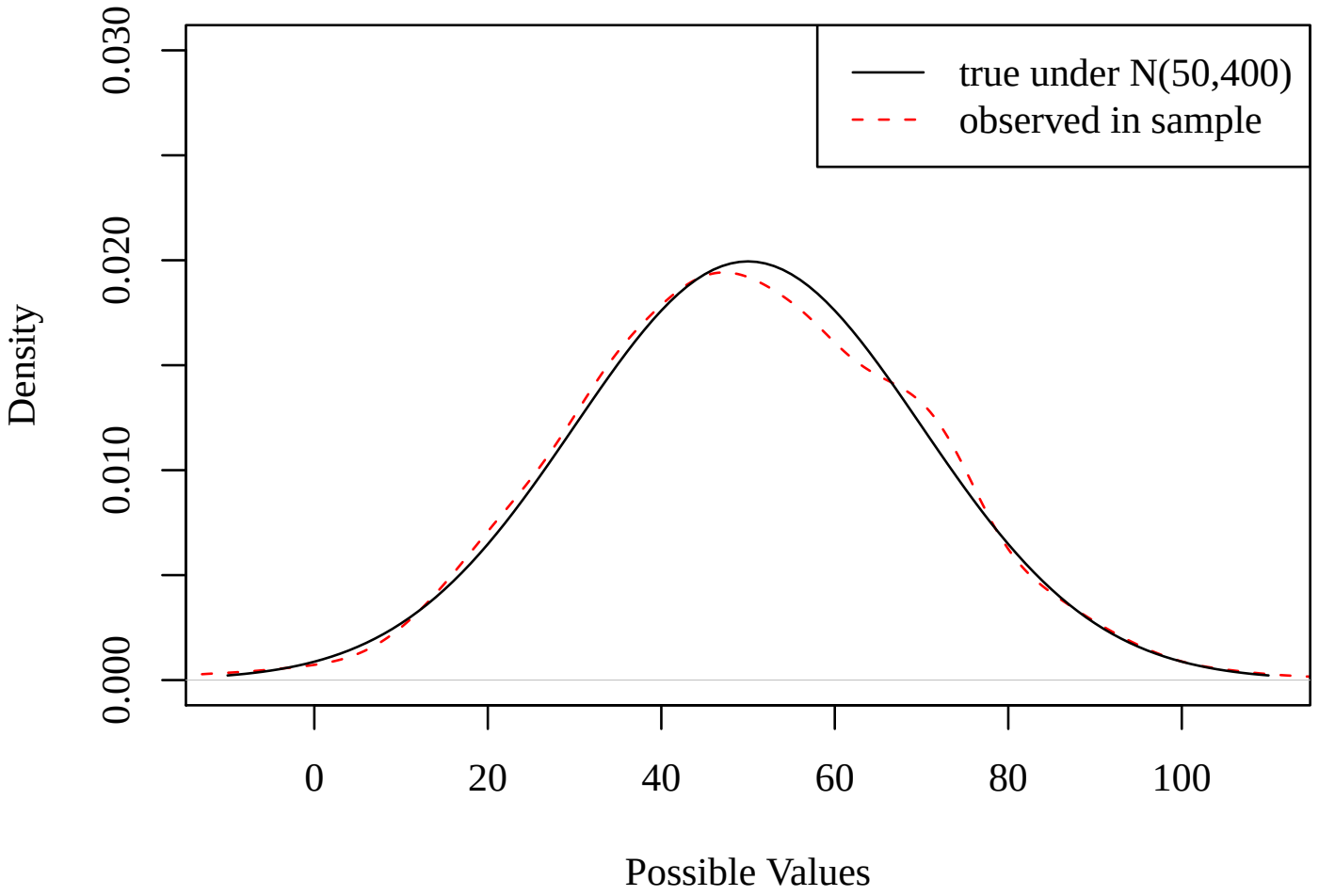
R has random variables

- Types of random variable generators (not just Normal, but also many others)
- Calculate theoretical quantities
  - probability density curves
  - cumulative distribution functions
- Draw samples from these distributions
- Conduct simulations (Monte Carlo experiments) easily
  - R has functions to streamline this work.

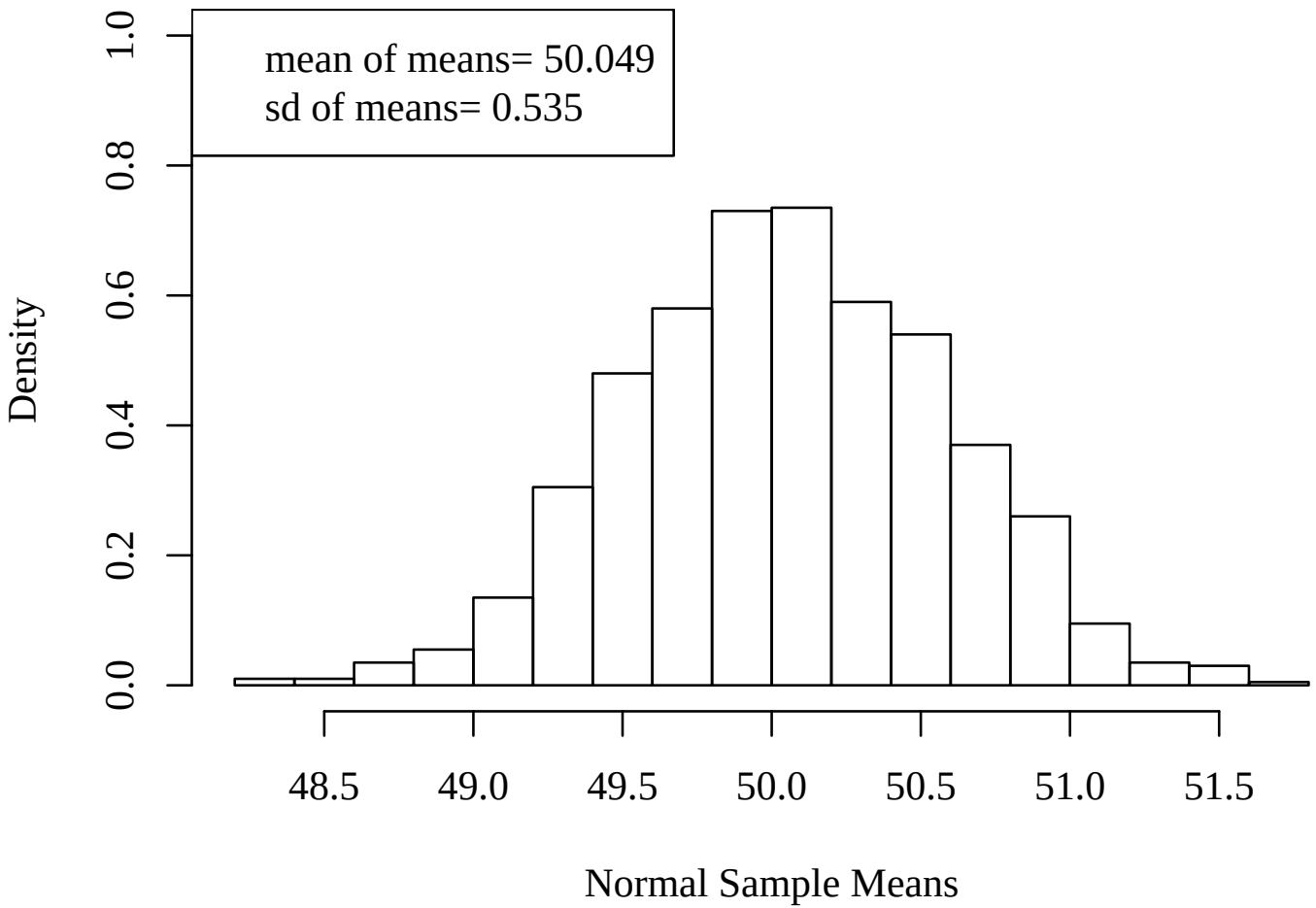
One Normal Variable, μ=50, σ=20



Observed and "True" Probabilties

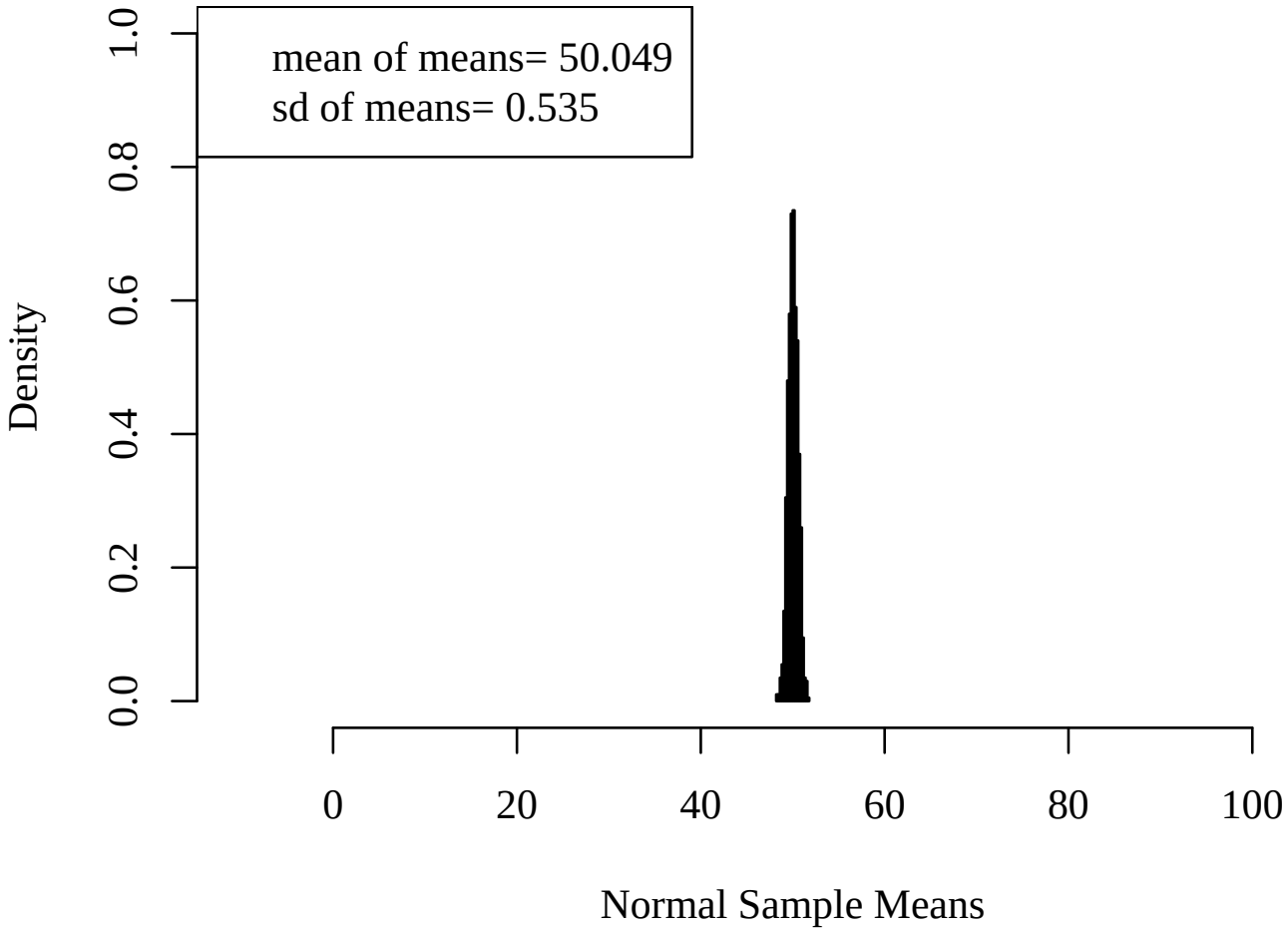


The Sampling Distribution of the Mean

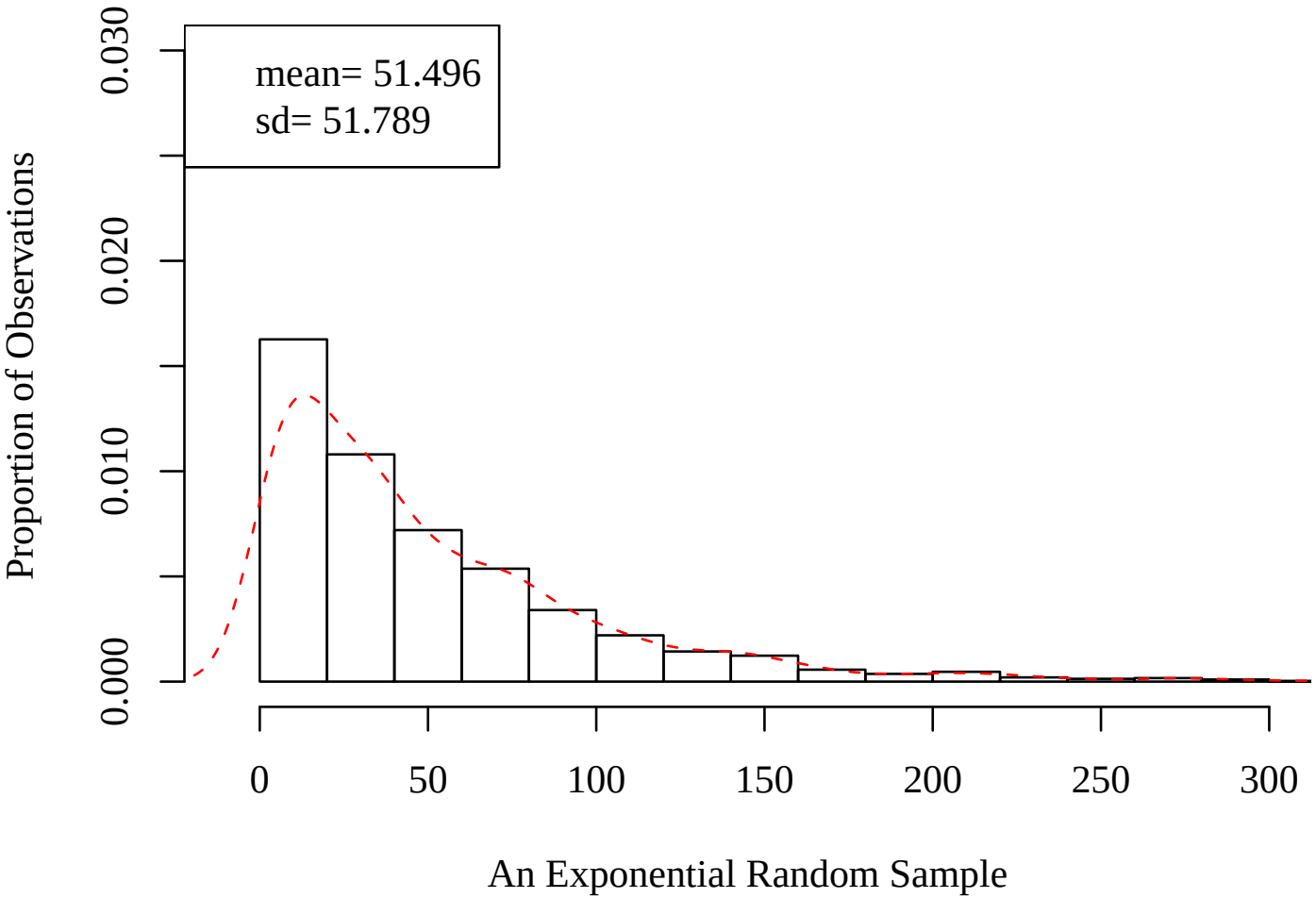


Consistent with theory, means should be Normal(μ=50, σ =20/$\sqrt{1500}$

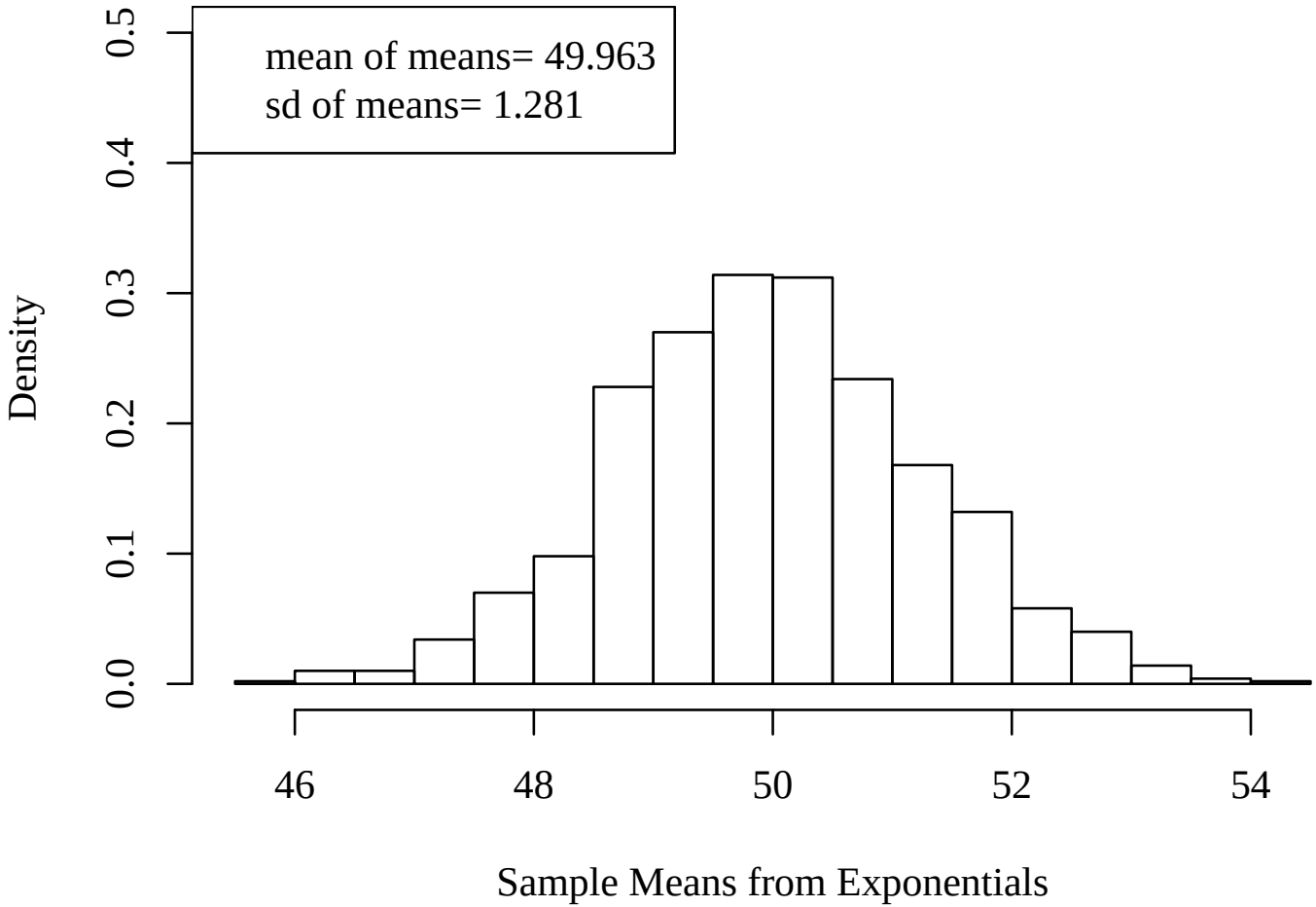
Put On Original Scale!



Sample from Exponential is not Normal



The Means Look Very Normal to ME!



Recall that this is the Central Limit Theorem

Outline

CRAN: a service from the R Core Team

- R Package Writers follow a set of guidelines
- Upload packages to CRAN
- Available after passing checks & tests
- R users can download & install from within R.

```
> install.packages(c("lmtest","car"), dep=T)
```

A Little Introspection, Please

- What packages do you have already?

```
> rownames(installed.packages())
```

  R provides a set of "recommended" packages, every install will have them.

- Wonder what you are missing out on?

```
> rownames(available.packages())
```

  On 2010-03-19, that command returned a list of 2260 packages.

- I want it ALL!
  I wrote a script that installed them all on a Windows system.
  Download and Install took
  - 3 hours
  - 2.7 Gigabytes of storage

- Check for updates periodically

```
> update.packages( ask=F, checkBuilt=T)
```

A Vignette on Sudoku

- I recently learned there is an R package for making and playing SudoKu puzzles.
- At first, I turned my nose up at the frivolity of it, but then
- I installed it

```
> install.packages("sudoku")
```

- After it is installed, run

```
> library(sudoku)
```

What is that Sudoku thing?

The first thing I always do after loading a package is find out what is inside it:

```
> library(help=sudoku)
```

Documentation Included! No Extra Charge!

```
Information on package 'sudoku'
Description:
Package: sudoku
Version: 2.2
Date: 2009-02-02
Title: Sudoku Puzzle Generator and Solver
Author: David Brahm <brahm@alum.mit.edu> and Greg Snow <Greg.
    Snow@intermountainmail.org>, with contributions from Curt
    Seeliger <Seeliger.Curt@epamail.epa.gov> and Henrik
    Bengtsson <hb@maths.lth.se>.
Maintainer: David Brahm <brahm@alum.mit.edu>
Suggests: tkrplot
Description: Generates, plays, and solves Sudoku puzzles. The
     GUI playSudoku() needs package "tkrplot" if you are not
    on Windows.
License: GPL
Packaged: Mon Feb 2 16:28:15 2009; a215020
Built: R 2.10.1; ; 2010-03-19 06:50:35 UTC; unix
```

Index:

fetchSudokuUK	Fetch the daily sudoku puzzle from http://www.sudoku.org.uk/
generateSudoku	Randomly Generate a Sudoku Puzzle Grid
hintSudoku	Give a Hint for a Sudoku Cell
playSudoku	Interactively play a game of Sudoku
printSudoku	Print a Sudoku Grid to the Terminal.
readSudoku	Read a File Containing a Sudoku Grid
solveSudoku	Solve a Sudoku Puzzle
writeSudoku	Write a Sudoku Grid to a File

Documentation Included! No Extra Charge!

- Then I use the help feature to find out more on the interesting-looking ones:

```
> ?generateSudoku
```

- That's the same as:

```
> help(generateSudoku)
```

- Perhaps I run the example that is displayed on the help page:

```
> example(generateSudoku)
```

When you run a function, the parentheses are required, even if you don't add any specific arguments. This tells generateSudoku to use the default settings.

```
> generateSudoku()
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]
[1,]	1	0	0	0	0	0	0	0	0
[2,]	7	0	0	0	1	3	5	8	2
[3,]	8	2	0	0	6	0	0	0	0
[4,]	4	0	1	0	2	8	6	0	0
[5,]	0	5	8	0	0	0	4	0	1
[6,]	0	0	0	3	4	0	0	0	0
[7,]	5	0	2	0	7	9	3	1	4
[8,]	0	0	0	0	0	2	0	0	0
[9,]	0	7	0	0	0	0	0	5	0

A Nicer Looking Sudoku Puzzle

```
> myPuzzle <- generateSudoku(Nblank = 20, print.it = F)
> printSudoku(myPuzzle)
```

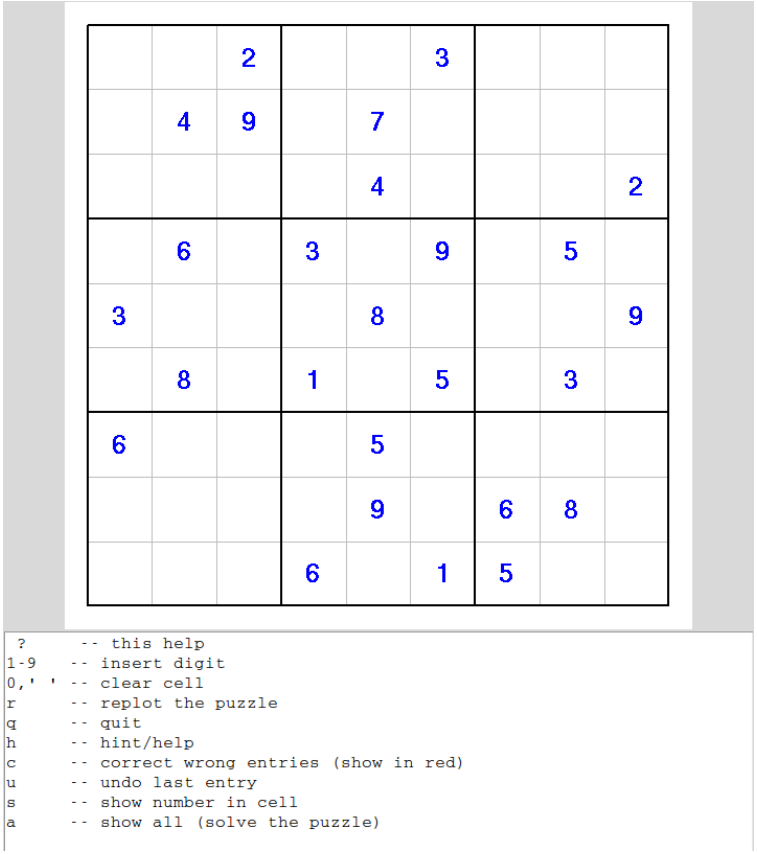
```
+-------+-------+-------+
| 9 3 6 |   1 2 | 5 7 8 |
| 7   5 | 6 9 3 | 2 4 1 |
| 1 2   | 7   5 | 6     |
+-------+-------+-------+
| 8 5   | 9 3 6 |   1 2 |
| 2 4 1 |   5 7 | 9   6 |
| 3 6   | 1   4 | 7   5 |
+-------+-------+-------+
| 5   8 | 3 6   | 1 2   |
| 4   2 |   7 8 |   6 9 |
| 6   3 | 2 4 1 | 8 5 7 |
+-------+-------+-------+
```

Torture Yourself with British Sudoku

```
> printSudoku(fetchSudokuUK())
+-------+-------+-------+
|     2 |     3 |       |
|   4 9 |   7   |       |
|       |   4   |     2 |
+-------+-------+-------+
|   6   | 3   9 |   5   |
| 3     |   8   |     9 |
|   8   | 1   5 |   3   |
+-------+-------+-------+
| 6     |   5   |       |
|       |   9   | 6 8   |
|       | 6   1 | 5     |
+-------+-------+-------+
```

Play Sudoku interactively against R

There is even an interactive on-screen game to be played (with hints for cheaters)



In Some Ways, R is very forgiving

R interprets all of these commands in the same way:

```
> generateSudoku(Nblank=20, print.it = TRUE)
> generateSudoku(20,T)
> generateSudoku(N=20, p=T)
> generateSudoku(p=T, N=20)
```

R will try to match up the options with your arguments, but I try to avoid gambling by explicitly naming options.
This does not give what you want because the arguments are out of order and unnamed

```
> generateSudoku(T, 20)
```

Outline

How do you get that GSS data?

```
> library(memisc)
> idat <- spss.system.file("/home/pauljohn/ps/ps706/DataExample
> idat2 <- as.data.set(idat)
> dat <- as.data.frame(idat2)
> rm(idat2)
> rm(idat)
```

R table() output: boring

```
> table(dat$vote00)

             VOTED       DID NOT VOTE         INELIGIBLE
              1826                715                389
REFUSED TO ANSWER
                 0
```

gmodels package: Tastes Like SPSS in here!

```
> library(gmodels)
> CrossTable(dat$vote00)

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|
```

VOTED	DID NOT VOTE	INELIGIBLE
1826	715	389
0.623	0.244	0.133

gmodels package: Tastes Like SPSS in here!

```
> CrossTable(dat$vote00, dat$sex)

   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  2930
```

	dat$sex		
dat$vote00	MALE	FEMALE	Row Total
VOTED	779	1047	1826
	0.259	0.199	
	0.427	0.573	0.623
	0.612	0.632	
	0.266	0.357	
DID NOT VOTE	317	398	715
	0.130	0.100	
	0.443	0.557	0.244
	0.249	0.240	
	0.108	0.136	
INELIGIBLE	177	212	389
	0.378	0.290	
	0.455	0.545	0.133
	0.139	0.128	

I like memisc's way

```
> gt <- genTable(percent(vote00) ~ sex, data = dat)
> gt
```

	sex	
percent(vote00)	MALE	FEMALE
VOTED	61.19403	63.18648
DID NOT VOTE	24.90181	24.01931
INELIGIBLE	13.90416	12.79421
REFUSED TO ANSWER	0.00000	0.00000
N	1273.00000	1657.00000

mainly because it easily goes to LaTeX

	MALE	FEMALE
VOTED	61%	63%
DID NOT VOTE	25	24
INELIGIBLE	14	13
REFUSED TO ANSWER	0	0
N	1273	1657

Outline

R usage for Dummies

My new policy. I won't help students unless they follow my "Workspace Advice" for R.[1] In essence,

1. Create a "folder"
2. Copy a template R file into that folder
3. Open that R file with the Emacs text editor
4. Launch an R session inside an Emacs window
5. Develop the R code by going back-and-forth between the "program buffer" and the "R buffer"

[1] I put it in the Emacs wiki, it must be right!
`http://www.emacswiki.org/emacs/CategoryESS`

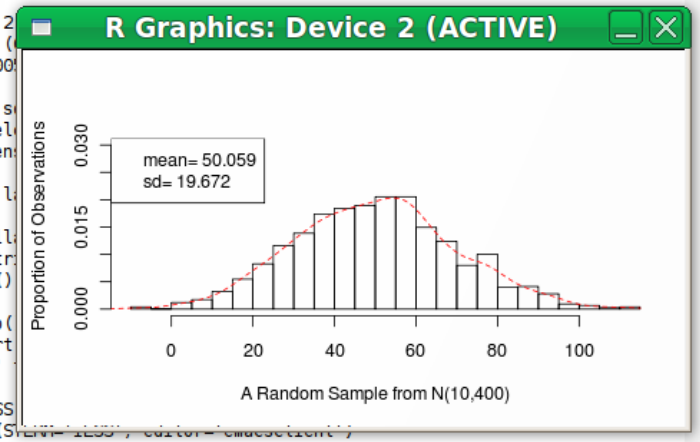
Commands on left, R session on Right

distrodemo.R

```
File  Edit  Options  Buffers  Tools  Imenu-S  ESS  Help

##################################################
### chunk number 2: Roptions
##################################################
options(width=60, continue="  ")
###Leave less white space at top
options(SweaveHooks=list(fig=function() par(mar=c(5.1, 4.1, 0.5, 2.1))))
###Sweave appears to ignore following settings 2010-03-20
ps.options(horizontal=F, onefile=F, family="Times",    paper="special", height=4, width=6 )
pdf.options(onefile=F,family="Times",paper="special",height=4,width=6)
options(papersize="special")



##################################################
### chunk number 3: fig1
##################################################
var1 <- rnorm(n=1500, mean=50, sd=20)
hist(x=var1, prob=T, breaks= 20, xlim=c(-10,110), ylim=c(0,0.03), xlab="A Random Sample from N(10,400)", ylab="Proportion of Observations", main="")
den1 <- density(var1)
lines(den1, lty=2, col="red")
legend("topleft",legend=c(paste("mean=",round(mean(var1),3)),paste("sd=",round(sd(var1),3))))


##################################################
### chunk number 4: fig2
##################################################
plot(den1, xlim=c(-10,110), ylim=c(0,0.03), xlab="Possible Values", type="l", lty=2, col="red",main="")
possValues <- seq(-10,110)
trueProbs <- dnorm(possValues, mean=50, sd=20)
lines(possValues, trueProbs, lty=1, col="black")
legend("topright", legend=c("true under N(50,400)","observed in sample"),lty=c(1,2),col=c("black","red"))


##################################################
### chunk number 5: fig3
##################################################
samp <- replicate(1000, mean(rnorm(n=1500, mean=50, sd=20)))
hist(samp, prob=T, breaks=20, ylim=c(0,1), xlab="Normal Sample Means", main="")
--:---  distrodemo.R   Top L28    (ESS[S] [R] Rox)----------
```

R

```
File  Edit  Options  Buffers  Tools  iESS  Complete  In/Out  Signals  Help

R version 2
Copyright (
ISBN 3-9000

R is free s
You are wel
Type 'licen

  Natural l

R is a coll
Type 'contr
'citation()

Type 'demo(
'help.start
Type 'q()'

> .help.ESS
> options(S
> options(width=60, continue="  ")
> options(SweaveHooks=list(fig=function() par(mar=c(5.1, 4.1, 0.5, 2.1))))
> ps.options(horizontal=F, onefile=F, family="Times",    paper="special", height=4, width=6 )
> pdf.options(onefile=F,family="Times",paper="special",height=4,width=6)
> options(papersize="special")
> var1 <- rnorm(n=1500, mean=50, sd=20)
> hist(x=var1, prob=T, breaks= 20, xlim=c(-10,110), ylim=c(0,0.03), xlab="A Random Sample from N(10,400)", ylab="Proportion of Observations", main="")
>
> den1 <- density(var1)
> lines(den1, lty=2, col="red")
> legend("topleft",legend=c(paste("mean=",round(mean(var1),3)),paste("sd=",round(sd(var1),3))))
>
-U:**-  *R*          All L33    (iESS [R]: run)----------
```



Emacs is like Democracy. Its the worst, except for all of the others that have been tried...

- Emacs
  - Free
  - Available on all platforms
  - Highly configurable
  - Useful for many other kinds of projects.

Outline

Draw a Sample from the Normal, Create a Histogram

```
> var1 <- rnorm(n = 1500, mean = 50, sd = 20)
> hist(x = var1, prob = T, breaks = 20, xlim = c(-10,
      110), ylim = c(0, 0.03), xlab = "A Random Sample from N(
      ylab = "Proportion of Observations", main = "")
> den1 <- density(var1)
> lines(den1, lty = 2, col = "red")
> legend("topleft", legend = c(paste("mean=", round(mean(var1),
      3)), paste("sd=", round(sd(var1), 3))))
```

Compare Theoretical Probabilities and Observed Sample

```
> plot(den1, xlim = c(-10, 110), ylim = c(0, 0.03),
      xlab = "Possible Values", type = "l", lty = 2,
      col = "red", main = "")
> possValues <- seq(-10, 110)
> trueProbs <- dnorm(possValues, mean = 50, sd = 20)
> lines(possValues, trueProbs, lty = 1, col = "black")
> legend("topright", legend = c("true under N(50,400)",
      "observed in sample"), lty = c(1, 2), col = c("black",
      "red"))
```

Draw Lots of Samples, Calculate their Means, and Plot

```
> samp <- replicate(1000, mean(rnorm(n = 1500, mean = 50,
      sd = 20)))
> hist(samp, prob = T, breaks = 20, ylim = c(0,
      1), xlab = "Normal Sample Means", main = "")
> legend("topleft", legend = c(paste("mean of means=",
      round(mean(samp), 3)), paste("sd of means=",
      round(sd(samp), 3))))
```

Re-scale the Previous Histogram

```
> hist(samp, prob = T, breaks = 20, xlab = "Normal Sample Means
      xlim = c(-10, 110), ylim = c(0, 1), main = "")
> legend("topleft", legend = c(paste("mean of means=",
      round(mean(samp), 3)), paste("sd of means=",
      round(sd(samp), 3))))
```

Create and Plot an Exponential Variate

```
> var1 <- rexp(n = 1500, rate = 1/50)
> hist(x = var1, prob = T, breaks = 20, xlim = c(-10,
      300), ylim = c(0, 0.03), xlab = "An Exponential Random Sa
      ylab = "Proportion of Observations", main = "")
> den1 <- density(var1)
> lines(den1, lty = 2, col = "red")
> legend("topleft", legend = c(paste("mean=", round(mean(var1),
      3)), paste("sd=", round(sd(var1), 3))))
```

The Central Limit Theorem is Correct

```
> samp <- replicate(1000, mean(rexp(n = 1500, rate = 1/50)))
> hist(samp, prob = T, breaks = 20, ylim = c(0,
      0.5), xlab = "Sample Means from Exponentials",
      main = "")
> legend("topleft", legend = c(paste("mean of means=",
      round(mean(samp), 3)), paste("sd of means=",
      round(sd(samp), 3))))
```