Using **R** to Teach MBA Derivatives

Robert L. McDonald

Kellogg School of Management
Northwestern University

May 19, 2014

Outline

- How I got started with **R**
- Why use **R** in an MBA class?
- Binomial pricing example
- The **R** adoption decision as an option exercise problem
- Examples of **R** in action

Everyone in this audience made a decision to use **R**; my focus will be on persuading *others* to use **R**

Context

- "Derivatives Markets II" is an advanced elective
- Topics include:
  - The lognormal model
  - Brownian motion and Itô's Lemma
  - Black-Scholes-Merton analysis
  - Stochastic volatility
  - Exotic options
  - One-factor fixed income models
  - Credit risk
- Focus on computation. Originally we used Excel/VBA
  - Fully transitioned to **R** this year
- All Kellogg students receive Stata

Educational Goals

In addition to the obvious (for example, learning about different kinds of exotic options), major themes include:

- Simple data skills (e.g., historical volatility)
- Monte Carlo valuation
- Appreciation of Jensen's inequality
- Implications of normality and why it is inadequate for modeling stock returns

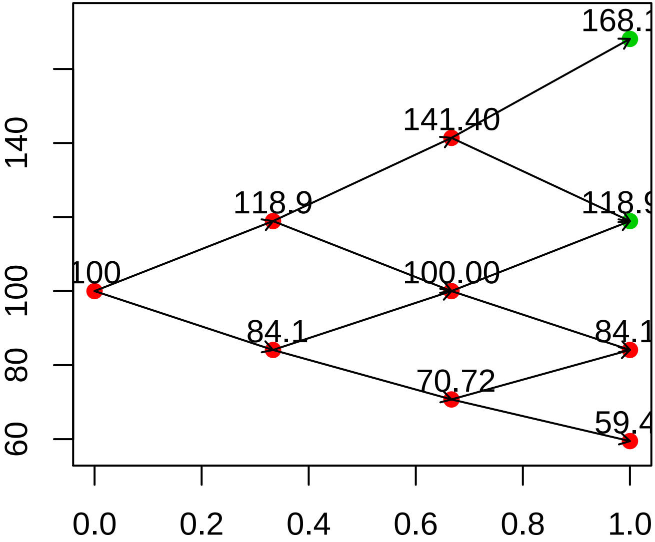
Why **R**?

- Vectorization
  - Hard to overstate pedagogical importance
  - `set.seed` permits comparison of loops and vectors
- High quality graphics
- Integrated statistics
- RStudio: Makes **R** feasible for students
- knitr — a terrific tool for the instructor
- Free and cross-platform

Example: Binomial Option Pricing

- Major topic in an introductory derivatives class
- Assume the stock moves discretely up or down, from S to uS or dS
- Probability of up move is the "risk-neutral probability," p
- u and d are calibrated to match volatility
- The binomial model has great pedagogical value and it can be used to price American options

Binomial Stock Price Tree



American Options

- American options may be exercised by the holder at any time
- When is it optimal to exercise the option: paying K in exchange for the stock, earning the intrinsic value, $S - K$?

The Economics of Early Exercise

When $S > K$, three factors determine whether to exercise a standard American call prior to expiration

- Dividend payments induce exercise (the holder of an unexercised call forgoes dividends)
- Interest leads to delay of exercise (defer payment of the strike price)
- Volatility leads to delay of exercise (with higher volatility, greater chance of later regret for having exercised)

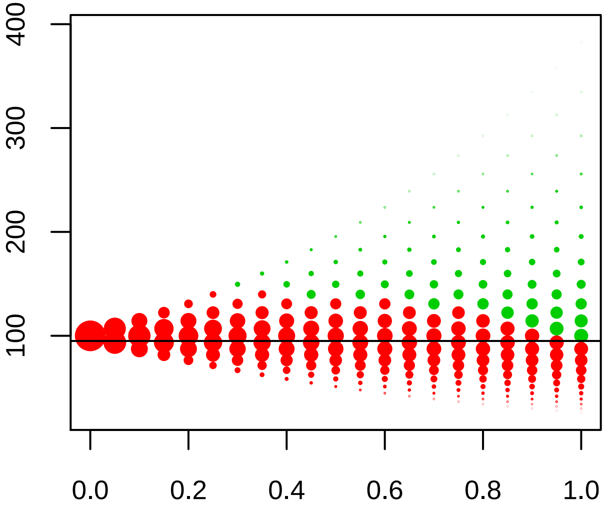
Illustrating Early Exercise

An **R** plot can be used to show simultaneously

- The possible values of future stock prices
- The probability of each price
- Whether the option is exercised at a given node (green means exercise, red means not)

Illustrating Early Exercise: Example

$S = 100$, $K = 95$, $\sigma = 0.3$, $r = 0.08$, $\delta = 0.08$, $T = 1$, $n = 20$:



Drawing the Stock Price Points

- The circles are color-coded to denote early exercise (green) or not (red):

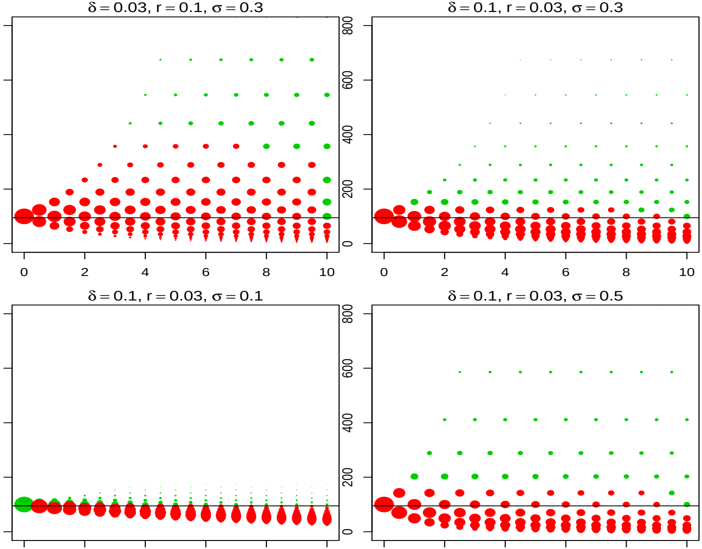
```
exer <- (optflag*(SMat - k) == Vc)
plotcolor <- ifelse(exer,"green3","red")
... ## within the plot command
     ,col=plotcolor[SMat>0]
```

- Their area is proportional to the probability of being at that price in that period:

```
... ## within the plot command
     ,cex= sqrt(probmat[SMat>0])*pointsize
```

Comparative Statics for Early Exercise

$S = 100$, $K = 95$, $T = 10$, $n = 20$:



Application to the **R** Adoption Decision

- Adopting software is analogous to exercising an American call
  - The adoption cost is the strike price
  - The value from adopting is the stock price
  - The flow of benefits lost by not adopting is the dividend yield
  - The cost reduction by waiting is the interest rate
  - Volatility measures uncertainty about the value of adopting

The Adoption Decision, cont.

- Optimal to adopt when
  - The flow of value is great (dividends are large)
  - The cost reduction from waiting is small
    - interest rate is small
    - strike price is constant or increasing
  - Likelihood of regret is small (volatility is low and/or intrinsic value is great)
- All of these must be assessed relative to alternatives…

Adoption Decision

Table : Comparison of Software for MBA Use

	VBA/Excel	**Stata**	**Matlab**	**R**	**SAS**
Programming?	Y	Y/N	Y	Y	Y
Ease of learning	Medium	Easy/Hard	Med/Hard	Med/Hard	Med
Versatility	Low	Medium	High	High	Low
Transferability of Skills	Medium	Low	High	High	Low
Future use likely	High	Low	?	Med	?
Longevity	?	High	High	High	High
Cost	Low	Medium	High	Very Low	High

Questions to Consider. . .

- What backgrounds and capabilities do students have?
- Should students be active or passive users?
- Is the software intended for class only, or for long-term student use (after graduation)
- Student believe that VBA and Matlab are important

Adoption Issues Specific to **R**

- Interface / RStudio
- Necessity of programming
  - Does `source('foo.R')` count as programming?
- Instructor friendly?
- Should one use base **R** or packages?
  - I use base **R** unless there is a compelling reason to do otherwise

The Decision to Adopt

- Large dividend flow from **R**
  - Whether it's greater than for alternatives depends on context
- Prior to RStudio, high gain to waiting
- Volatility (relative to alternatives) was high, now seems low
  - For example, Octave looked like a contender, raising volatility, delaying my adoption of **R**
- Is $S > K$?
  - Depends on course, student, and instructor
- RStsudio made the adoption decision much easier

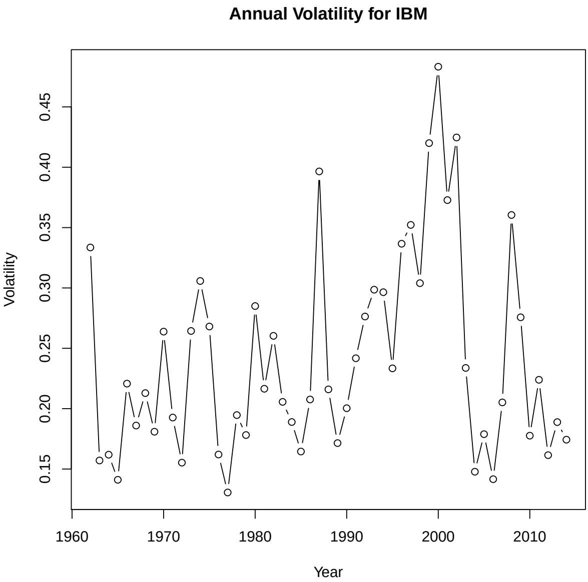
Data: Example

- Even students with some statistics background often have little or no experience dealing with “primary” data
- Using `read.csv` to obtain web data gets their attention
- Following example is not trivial, but
  - It illustrates the power of a script
  - It demonstrates time variation in volatility
  - The same exercise could be done in Excel (students can compare the difficulty and reproducibility)
  - Deciphering the code is a worthwhile exercise
- The example is easy to generalize (looping across tickers, for example)

Data: Example

```r
## Yahoo format is date, open, high, low, close, volume, adj close
URL <- "http://ichart.finance.yahoo.com/table.csv?s=IBM"
ibm <- read.csv(URL)[, c(1, 7)]
ibm$Date <- as.Date(ibm$Date)
ibm <- ibm[complete.cases(ibm),]  ## remove NAs
ibm$ret <- c(-diff(log(ibm$Adj.Close)), NA)
ibm$year <- as.POSIXlt(ibm$Date)$year + 1900
vol <- tapply(ibm$ret, ibm$year, sd, na.rm=TRUE)*sqrt(252)
plot(names(vol), vol, type='b', ylab='Volatility', xlab='Year',  main='Annual Volatility for IBM')
```

Data: Result



Annual Volatility for IBM
Volatility
Year
0.15
0.20
0.25
0.30
0.35
0.40
0.45
1960
1970
1980
1990
2000
2010

Monte Carlo Valuation

- Incredibly important topic
- Vectorization permits clean correspondence between equations and code
  - The transparency of vectorized code is an implicit selling point
  - Vectorized code often leads to an “aha” moment.
- Helpful in developing intuition about specific distributions
- The command `set.seed` always causes discussion

Valuing a European Call

Some cases display a perfect correspondence between equations in the book and equations in **R**:

```
s0 <- 100; k <- 95; v <- 0.30; r <- 0.08;
d <- 0; tt <- 0.25; n <- 1e05
z <- rnorm(n)
st <- s0*exp((r-d-0.5v^2)*tt + v*sqrt(tt)*z)
price <- exp(-r*tt)*mean(pmax(st-k, 0))
```

Valuing a European Call (jump version)

- Merton: lognormal jumps occur as a Poisson process
- Let g be the jump intensity, $J = e^a - 1$ the mean jump, and v_J the jump volatility.
- On paths when no jumps occur, the price is

```
st0 <- s0*exp((r-d-g*J-0.5v^2)*tt +
         v*sqrt(tt)*z1)
```

- We count jumps:

```
m <- rpois(n, g*tt)
```

- The terminal stock price is

```
z2 <- rnorm(m)
st <- st0*exp((a - 0.5*vj^2)*m + vj*sqrt(m)*z2)
```

Simulating the Merton Jump Model

```
s0 <- 100; k <- 95; r <- 0.08; v <- 0.30;
d <- 0; vj <- 0.40; aj <- -0.15; g <- 2.5;
tt <- 2.5;   J <- exp(aj) - 1; n <- 1e05
m <- rpois(n, lambda=g*tt)
z1 <- rnorm(n)
z2 <- rnorm(n)
st <- s0*exp((r-d-g*J-0.5*v^2)*tt + v*sqrt(tt)*z1)*
    exp((aj-0.5*vj^2)*m + vj*sqrt(m)*z2)
price <- exp(-r*tt)*mean(pmax(st-k, 0))
```

Arithmetic Asian Options

- The payoff of an Arithmetic Asian call is

$$\max\left[0, \sum_{i=1}^{n} \frac{S_i}{n} - K\right]$$

  where the S_is are sampled at regular intervals
- Because each subsequent stock price is based on the previous, it is tempting to write a loop.
- This is not necessary, but the correct code is tricky

The Problem

R can be too lenient:

```
> 1:3 + matrix(1, nrow=6, ncol=2)
     [,1] [,2]
[1,]    2    2
[2,]    3    3
[3,]    4    4
[4,]    2    2
[5,]    3    3
[6,]    4    4
> 1:3 + matrix(1, nrow=2, ncol=6)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    2    4    3    2    4    3
[2,]    3    2    4    3    2    4
```

Arithmetic Asian Call: Code

```
s0 <- 100; k <- 95; r <- 0.08; v <- 0.30;
d <- 0; tt <- 1
drift <- r - d - 0.5*v^2
z <- matrix(rnorm(n*m), nrow=n, byrow=TRUE)
zcum <- t(apply(z, 1, cumsum))
hmat <- matrix(1:m, nrow=n, ncol=m, byrow=TRUE)
s <- s0*exp((drift*h*hmat + v*sqrt(h)*zcum))
payoff <- pmax(rowSums(s)/m - k, 0)
price <- exp(-r*tt)*mean(payoff)
```

- The expression `t(apply(z, 1, cumsum))` requires some explanation
- This is a case where a loop would likely be more transparent

Black-Scholes Pricing

```
bsOpt(s=100, k=95, v=0.3, r=0.08, tt=1, d=0,
      greeks=TRUE)

##                    Call         Put
## Price         18.38706    6.083114
## Delta        0.7216145  -0.2783855
## Gamma        0.0111875  0.01119105
## Vega         0.3356785   0.3356785
## Rho          0.5377439  -0.3392167
## Theta      -0.02558117 -0.00636012
## Psi         -0.7216145   0.2783855
## Elasticity    3.924577   -4.576365
```

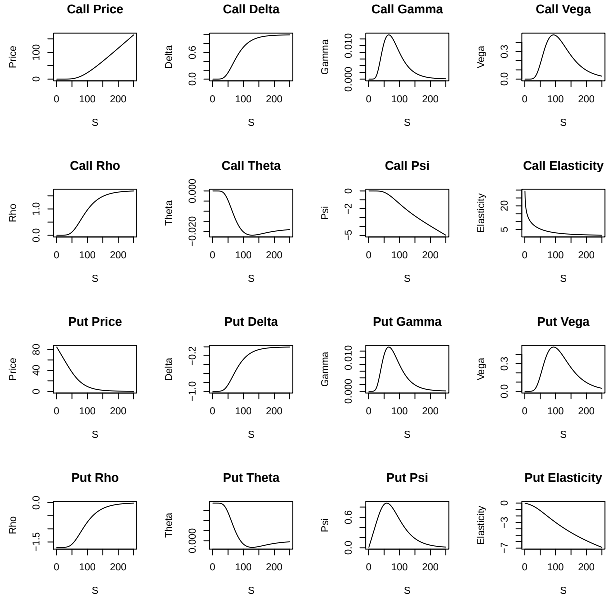
Code to show off **R**

`bsOpt` returns a data frame:

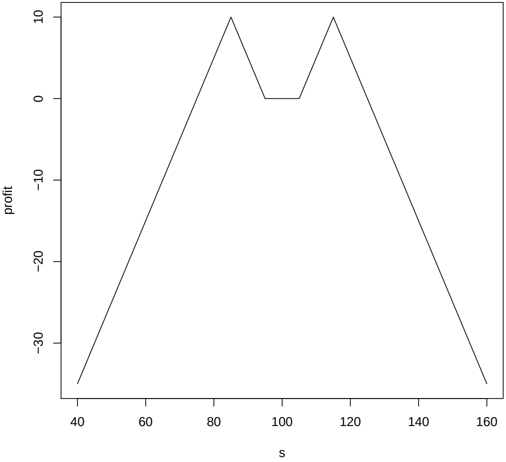
- columns are calls and puts
- rows are price and Greeks

```
k <- 100; v <- 0.30; r <- 0.08; tt <- 2; d <- 0
S <- seq(.5, 250, by=.5)
x <- bsOpt(S, k, v, r, tt, d, greeks=TRUE)
par(mfrow=c(4, 4))
for (i in colnames(x)) {
  for (j in rownames(x)) {
    plot(S, x[[i]][[j]], main=paste(i, j), ylab=j,
   type='l')
  }
}
```

Output

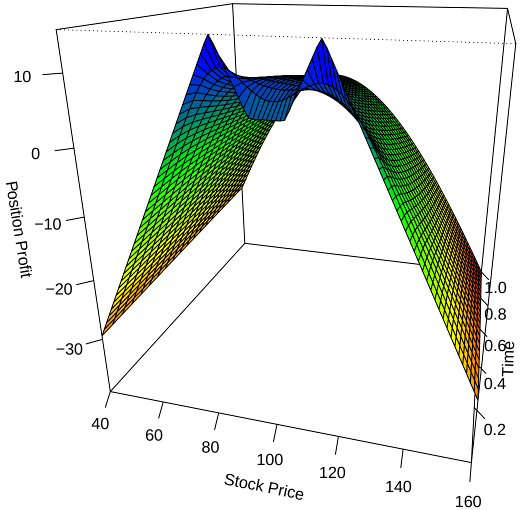

Call Price
Price
0
100
200
S
Call Delta
Delta
0.0
0.6
Call Gamma
Gamma
0.000
0.010
Call Vega
Vega
0.0
0.3
Call Rho
Rho
0.0
1.0
Call Theta
Theta
−0.020
0.000
Call Psi
Psi
−5
−2
0
Call Elasticity
Elasticity
5
20
Put Price
Price
0
40
80
Put Delta
Delta
−1.0
−0.2
Put Gamma
Gamma
0.000
0.010
Put Vega
Vega
0.0
0.3
Put Rho
Rho
−1.5
0.0
Put Theta
Theta
0.000
Put Psi
Psi
0.0
0.6
Put Elasticity
Elasticity
−7
−3
0

The Batman: Expiration

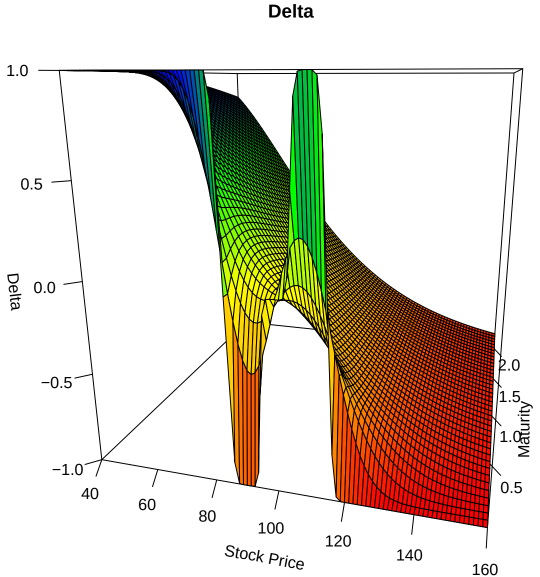


Sell two 85 puts, buy one 95 put, buy one 105 call, sell two 115 calls

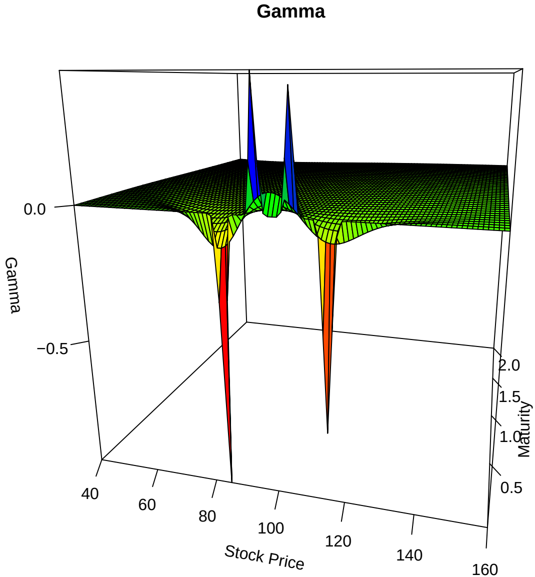
The Batman: Price



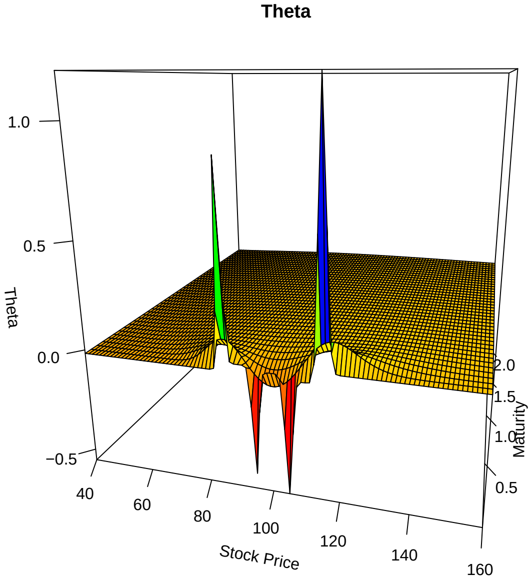
The Batman: Delta



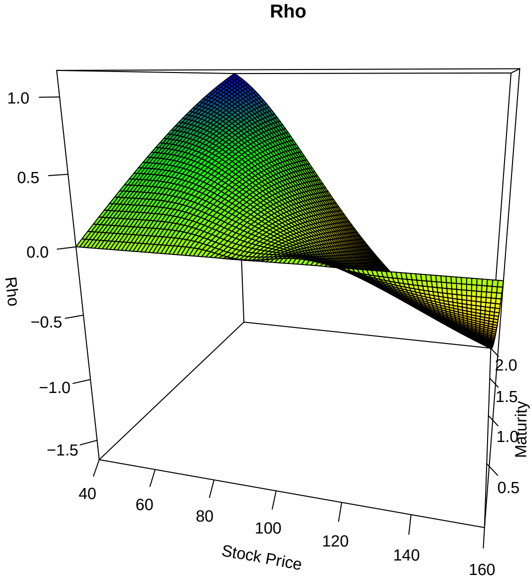
The Batman: Gamma



The Batman: Theta



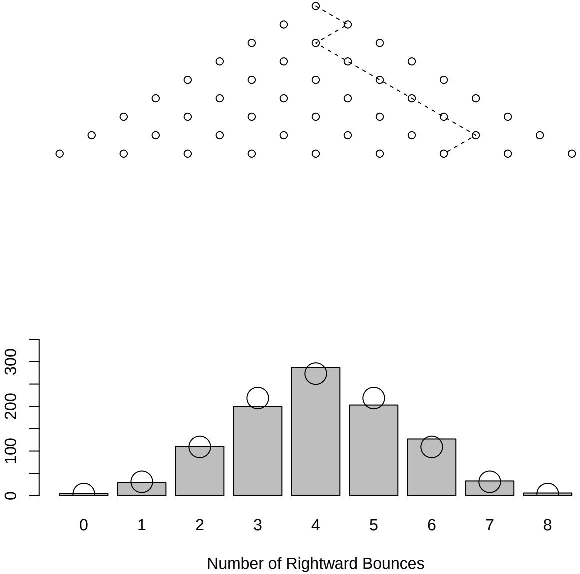
The Batman: Rho



The Quincunx

- A quincunx, or Galton board, is a physical device that illustrates the Central Limit Theorem
- Balls hit a series of pegs, randomly falling left or right
- **R** can (very crudely) simulate the appearance of a quincunx
- With **R**, unlike with a physical quincunx, it is possible to alter the probability from 50-50.

The Quincunx



Thank You

Thanks to all the fantastic folks who push this great software forward:

- R Core Team
- the RStudio team
- the Revolution Analytics team
- Hadley Wickham
- Yihui Xie
- Dirk Eddelbuettel
- R/Finance organizers and UIC
- etc.