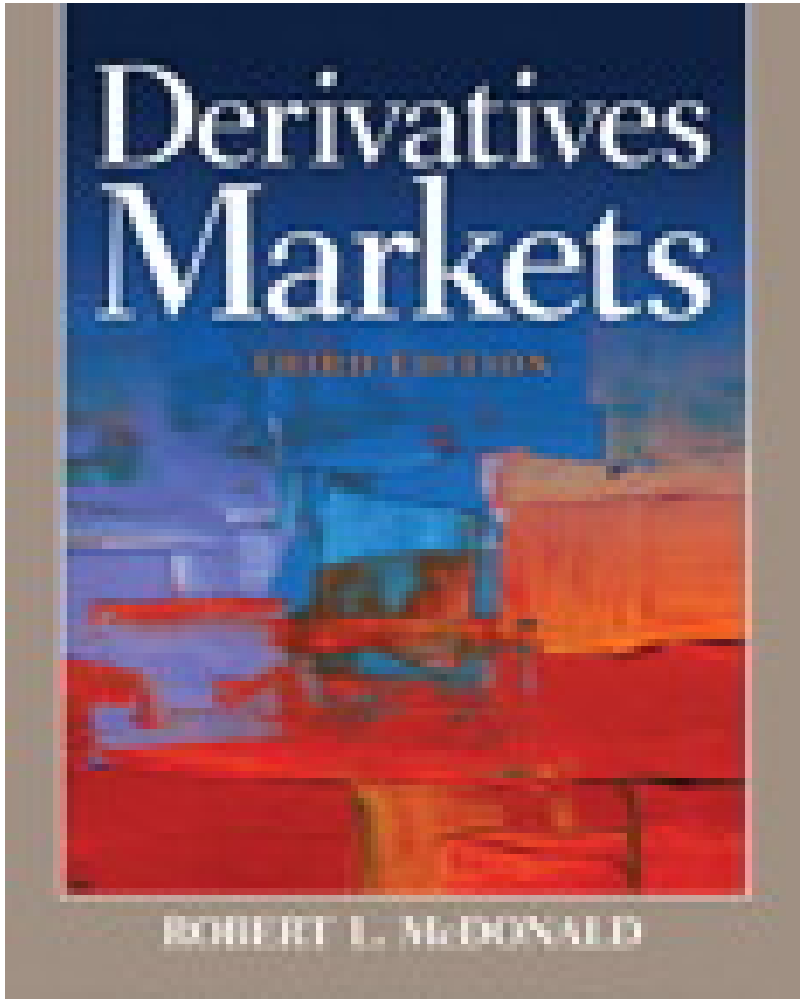
The derivmkts Package

Robert L. McDonald
Kellogg School of Management
Northwestern University

May 19, 2016

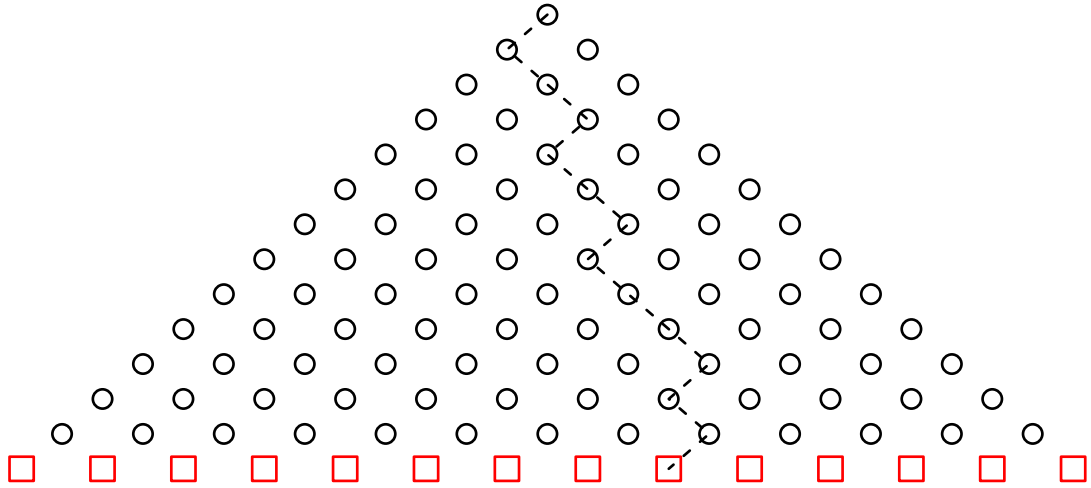
Derivatives Markets: The Book

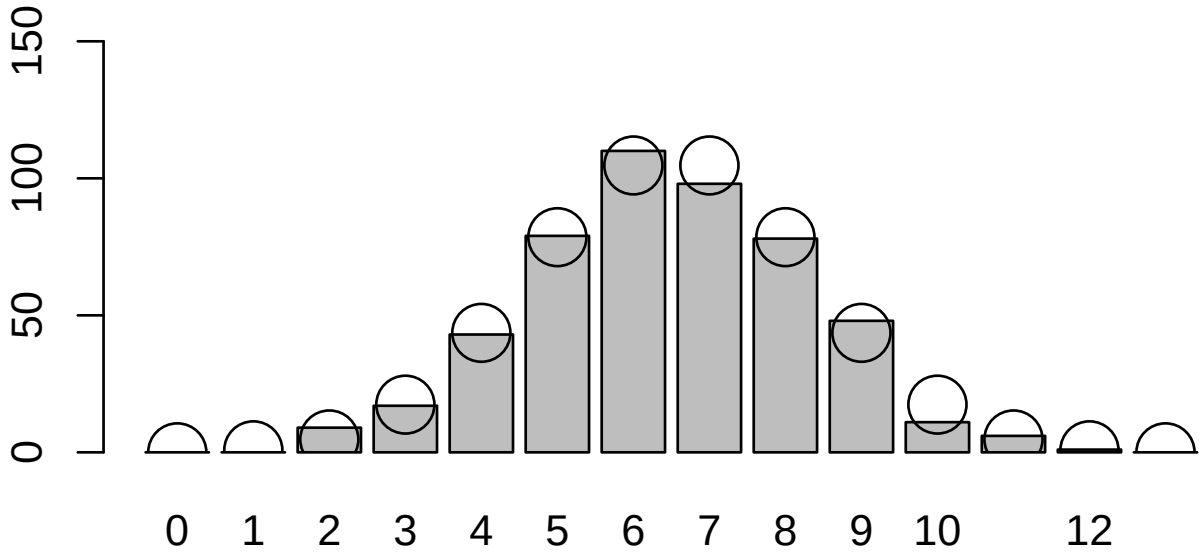


Contents of the package

- Basic option pricing functions
  - Vanilla
  - Barrier
  - Asian
  - Merton jump model
- Of special note:
  - The Quincunx
  - Binomial calculations and plotting
  - Greeks made easy

Quincunx





Quincunx

```
set.seed(1)
quincunx(n=13, numballs=500, delay=0, probright=0.5)
```

Output is controlled with **`delay`**`=tt`:

- `tt>0` runs one trial every `tt` seconds
- `tt=0` runs the entire simulation at once
- `tt<0` runs one trial every keypress

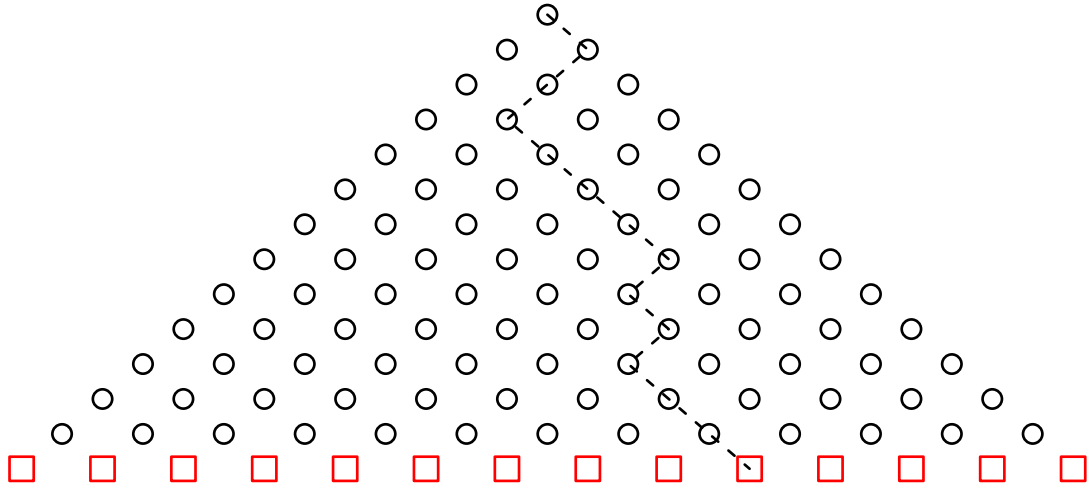
Quincunx, $p = 0.7$

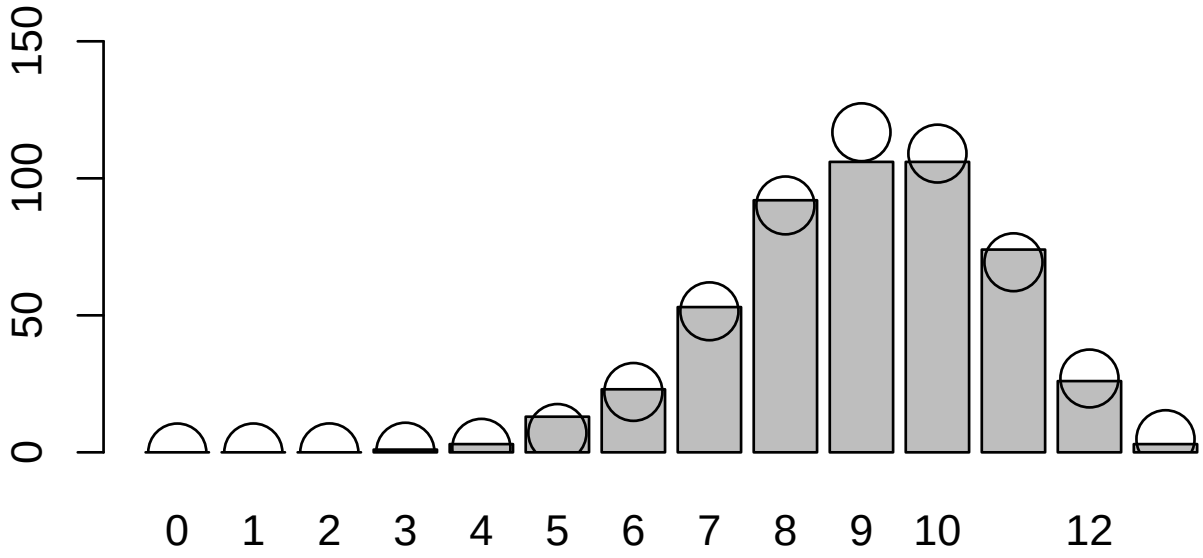
What if the probability of the ball going right or left is not 50-50?

```
set.seed(1)
quincunx(n=13, numballs=500, delay=0, probright=0.7)
```

The Central Limit Theorem to the rescue...

Quincunx, $p = 0.7$





Binomial Option Pricing

Lots of control over output:

```
binomopt <- function(s, k, v, r, tt, d, nstep=10,
                     american = TRUE,
                     putopt=FALSE,
                     specifyupdn=FALSE,
                     up=1.5, dn=0.5,
                     crr=FALSE,
                     jarrowrudd=FALSE,
                     returntrees=FALSE,
                     returnparams=FALSE,
                     returngreeks=FALSE
                ) {}
```

Binomial Pricing Output

```
s <- 100; k <- 100; r <- 0.08; v <- 0.30
tt <- 2; d <- 0.03
binomopt(s, k, v, r, tt, d, nstep=3) ## option price
binomopt(s, k, v, r, tt, d, nstep=3,
         returntrees=TRUE, putopt=TRUE)
```

Trees returned include

- stock price
- value of option
- whether exercised
- probability
- replicating portfolio (Δ and B at each node)

Binomial parameters and greeks

```
binomopt(s, k, v, r, tt, d, nstep=3,
         returnparams=TRUE, returngreeks=TRUE,
         putopt=TRUE)


        price         delta         gamma         theta
 12.936045164  -0.335721679   0.010614110  -0.007598759
            s             k             v             r
100.000000000 100.000000000   0.300000000   0.080000000
           tt             d         nstep             p
  2.000000000   0.030000000   3.000000000   0.439067117
           up            dn             h
  1.320859033   0.809275690   0.666666667
```

binomopt: Return parameters

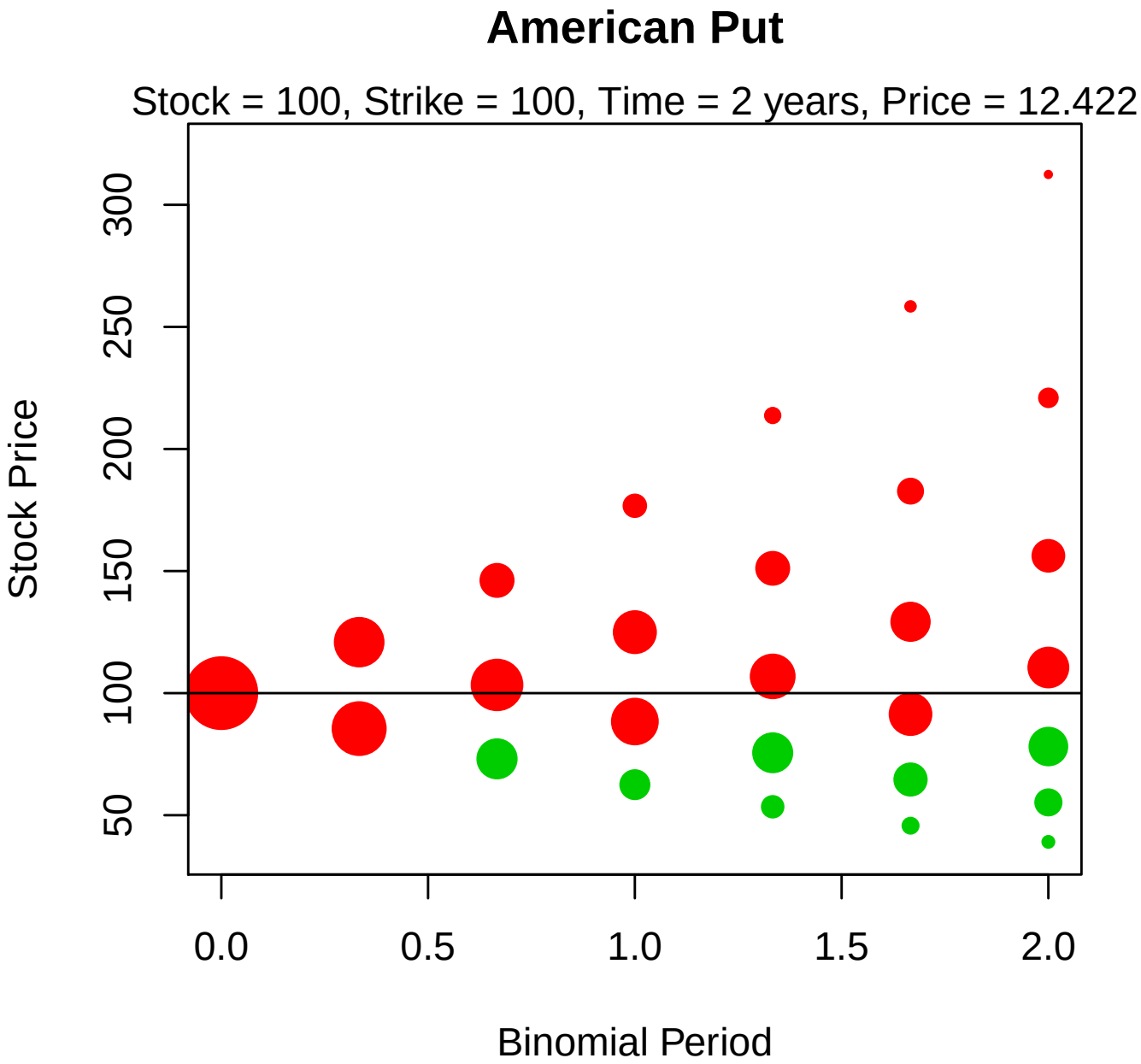
```
str(binomopt(s, k, v, r, tt, d, nstep=3,
             returntrees=TRUE, putopt=TRUE))

List of 9
 $ price       : Named num 12.9
  ..- attr(*, "names")= chr "price"
 $ greeks      : Named num [1:3] -0.3357 0.0106 -0.0076
  ..- attr(*, "names")= chr [1:3] "delta" "gamma" "theta
 $ params      : Named num [1:11] 1e+02 1e+02 3e-01 8e-02
  ..- attr(*, "names")= chr [1:11] "s" "k" "v" "r" ...
 $ oppricetree: num [1:4, 1:4] 12.94 0 0 0 3.82 ...
 $ stree       : num [1:4, 1:4] 100 0 0 0 132 ...
 $ probtree    : num [1:4, 1:4] 1 0 0 0 0.439 ...
 $ exertree    : logi [1:4, 1:4] FALSE FALSE FALSE FALSE
 $ deltatree   : num [1:3, 1:3] -0.336 0 0 -0.104 -0.647
 $ bondtree    : num [1:3, 1:3] 46.5 0 0 17.6 73.7 ...
```

Binomial Plotting

```
binomplot(s, k, v, r, tt, d, nstep=6,
          american=TRUE,
          putopt=TRUE)
```

The dot size in the plot is proportional to the probability of reaching the node

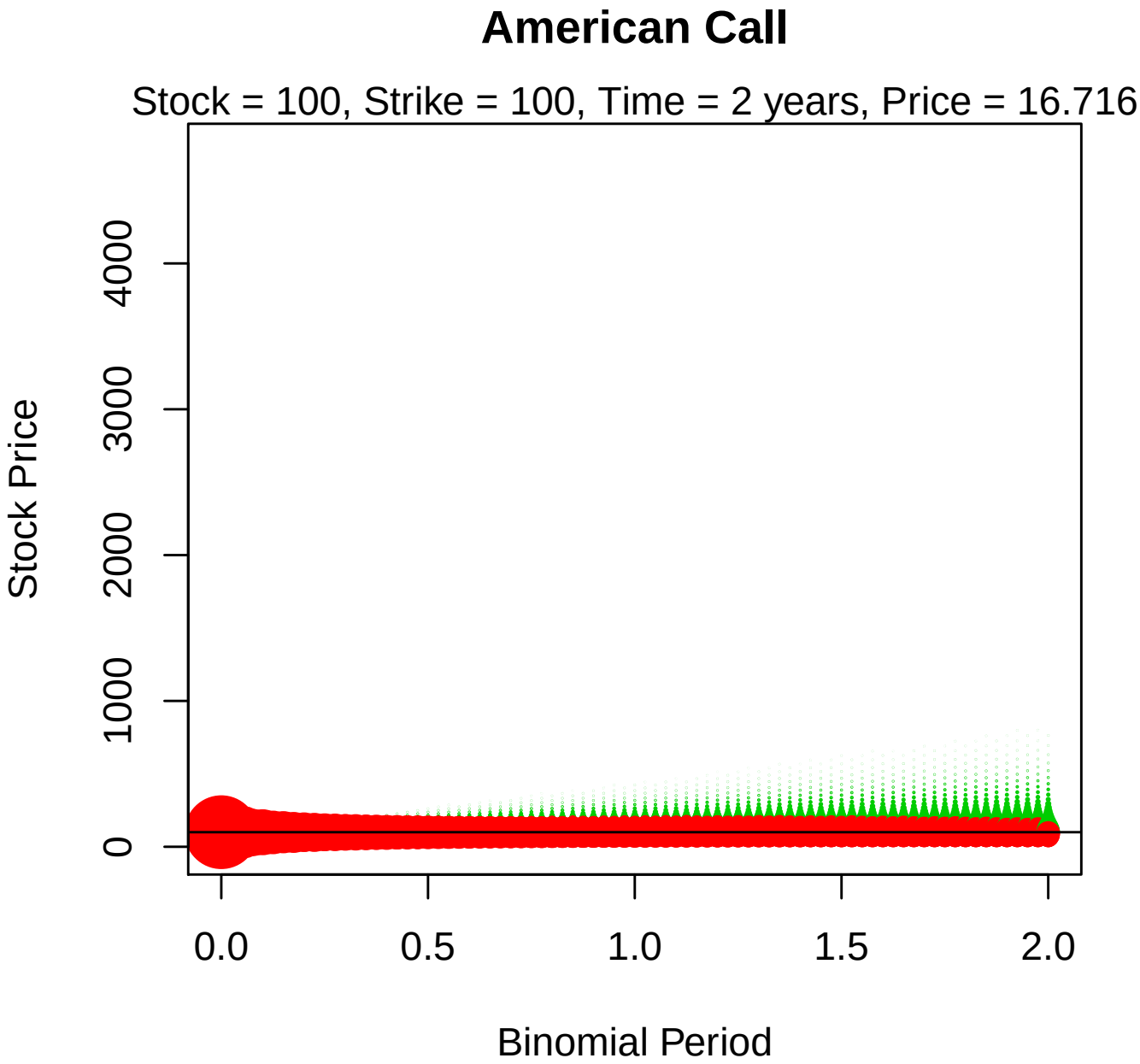
Binomial Plotting



Binomial Plotting, 80 Steps

```
d <- 0.06
binomplot(s, k, v, r, tt, d,
          nstep=80, american=TRUE)
```

Binomial Plotting, 80 Steps

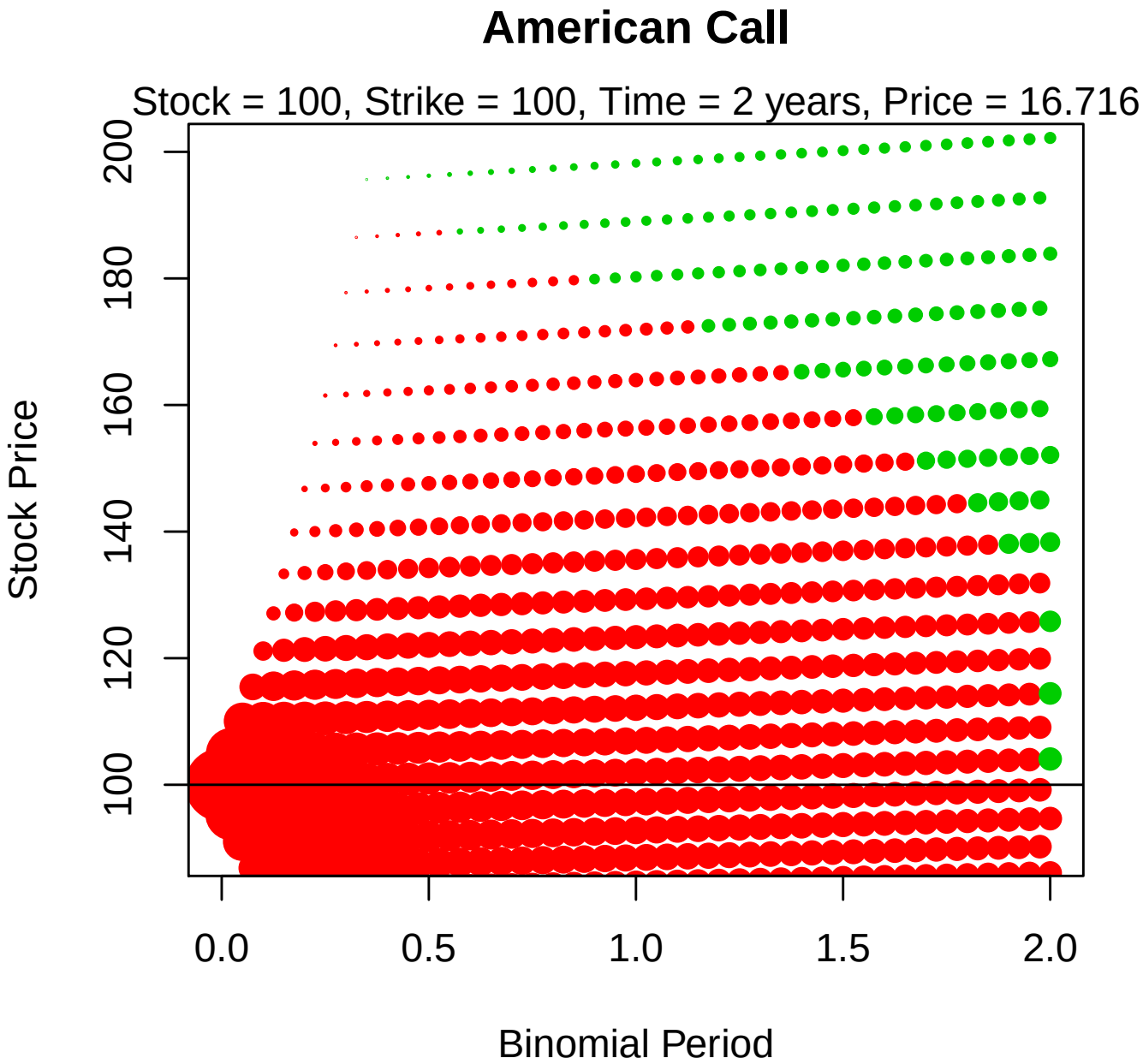


Binomial Plotting, 80 Steps

Make the plot more useful by evaluating a slice using `ylimval`:

```
d <- 0.06
binomplot(s, k, v, r, tt, d, nstep=80,
          american=TRUE, ylimval=c(90, 200))
```

Binomial Plotting, 80 Steps



Greeks

The `greeks()` function evaluates the greeks of any pricing function defined using specific parameter names:

```
bscall(s=100, k=c(95, 100, 105), v=0.3, r=0.08, tt=1, d=0)


[1] 18.38706 15.71131 13.33973


greeks(bscall(s=s, k=c(95, 100, 105), v=0.3, r=0.08, tt=1, d=0))


             bscall_95  bscall_100  bscall_105
Price      18.38706134 15.71131255 13.33973159
Delta       0.72161447  0.66153888  0.60026489
Gamma       0.01118931  0.01219245  0.01287585
Vega        0.33567849  0.36577236  0.38627533
Rho         0.53774385  0.50442575  0.46686757
Theta      -0.02558117 -0.02608765 -0.02610704
Psi        -0.72161447 -0.66153888 -0.60026489
Elasticity  3.92457749  4.21058953  4.49982731
```

Greeks for user-defined functions

Functions to price pre-paid forwards for S^a and for option spreads:

```
powercontract <- function(s, v, r, tt, d, a) {
    price <-
        exp(-r*tt)*s^a*
            exp((a*(r-d) + 1/2*a*(a-1)*v^2)*tt)
}

bullspread <- function(s, v, r, tt, d, k1, k2) {
    bscall(s, k1, v, r, tt, d) -
        bscall(s, k2, v, r, tt, d)
}
```

Note the same variable names in `powercontract`.

Greeks for powercontract()

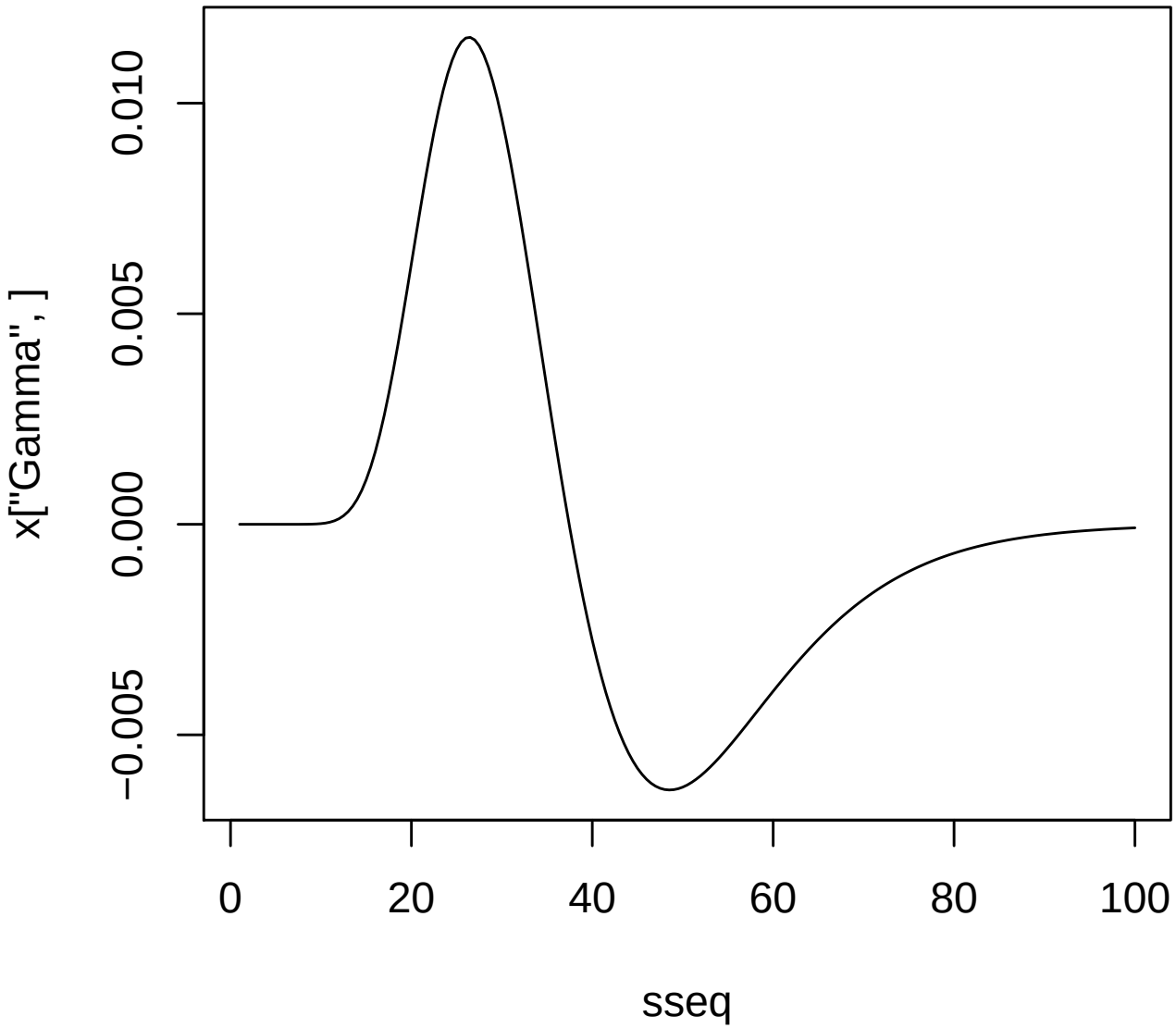
```
greeks(powercontract(s=100, v=c(0.2, 0.3, 0.4),
                     r=0.08, tt=1, d=0, a=2))
```

```
           powercontract_0.2 powercontract_0.3
Price           11274.968516      11853.048513
Delta             225.499370        237.060970
Gamma               2.254992          2.370598
Vega               45.099875         71.118292
Rho               112.749685        118.530485
Theta              -3.706839         -5.520598
Psi              -225.499372       -237.060972
Elasticity          2.000000          2.000000
           powercontract_0.4
Price           12712.491503
Delta             254.249830
Gamma               2.542492
Vega              101.699933
Rho               127.124915
Theta              -8.358899
Psi              -254.249832
Elasticity          2.000000
```

Greeks for a bull spread

```
sseq <- seq(1, 100, by=0.5)
x <- greeks(bullspread(sseq, .3, .08, 1, 0,
                       k1=40, k2=45))
plot(sseq, x['Gamma',], type='l')
```

Bull spread gamma



Conclusion

- At a minimum, I hope this makes it easier to teach a derivatives class
- The package is on cran and on github at rmcd1024/derivmkts
- Comments welcome

Conclusion

- At a minimum, I hope this makes it easier to teach a derivatives class
- The package is on cran and on github at rmcd1024/derivmkts
- Comments welcome

Thanks!