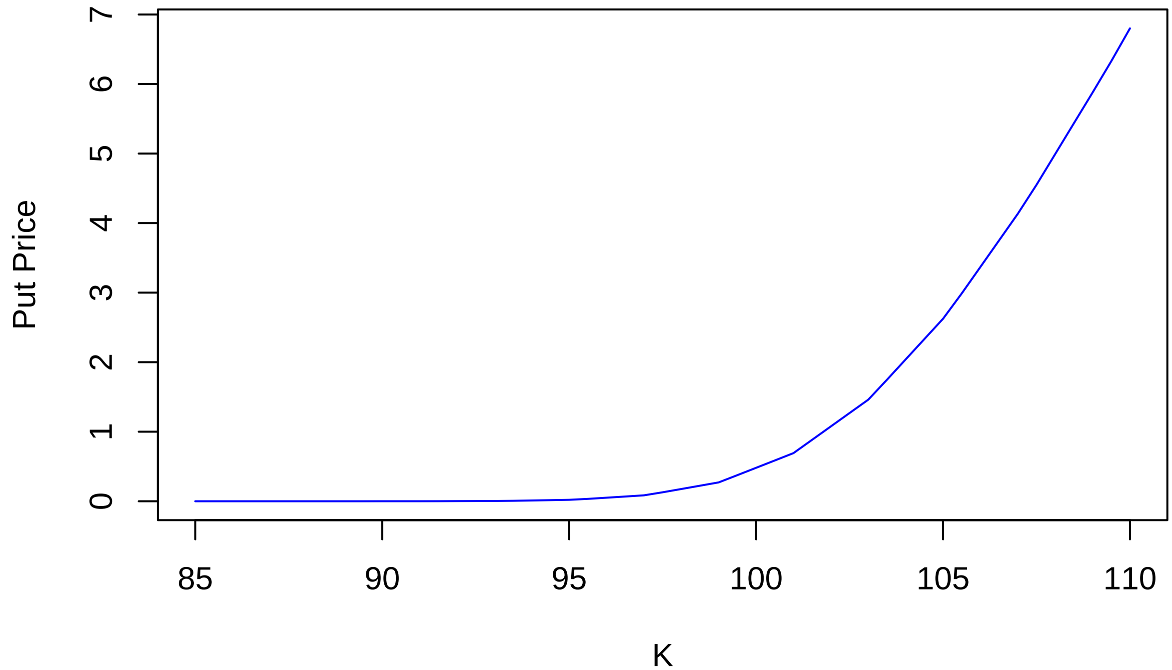
170Homework4

2022-11-16

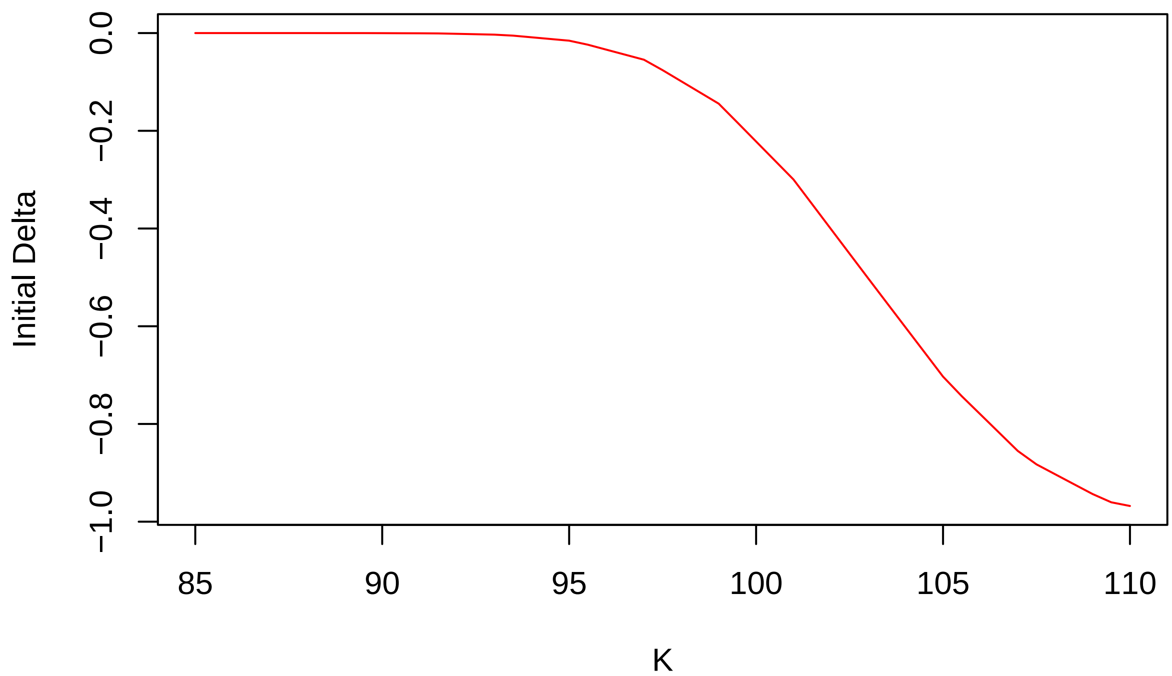
Question 3

```r
# Put function based off given r code
binomPut <- function(n,T,S0,K,r,delta)
{
  h <- T/n
  d<-100/101; u<-1.01; q<- (exp((r-delta)*h)-d)/(u-d)
  Sbin <- S0*d^n*(u/d)^(0:n)  # possible values of stock price
  qbin <- dbinom(0:n,n,q)  # corresponding probabilities using q
  Payoff <- pmax(K-Sbin,0)
  return(exp(-r*T)*sum(qbin*Payoff))
}
# Plot
K=seq(85,110,0.5)
priceE=c()
for (i in seq(85,110,0.5))
{
  put=binomPut(15,T=1,S=100, K=i,r=0.03,delta=0)
  priceE=append(priceE,put)
}
plot(K, priceE,type="l",col="blue", xlab="K", ylab="Put Price")
```




```
# Now for corresponding delta
binTree <- function(N,T,S0,K,r,delta) {
  h <- T/N
  u <- 1.01
  d <- 100/101
  q <- (exp( (r-delta)*h)-d)/(u-d)
  V <- array(0, dim=c(N+1,N+1)) # matrix for storing all the option values
  Del <- array(0, dim=c(N+1,N+1))
  for (i in 0:N) { # terminal payoff
    curS <- S0*(u)^i*(d)^(N-i)
    V[N+1,i+1] <- max( K-curS, 0) # Put payoff
  }
  for (j in N:1) {
    for (i in 0:(j-1)) {
      V[j,i+1] <- exp(-r*h)*(q*V[j+1,i+2]+(1-q)*V[j+1,i+1])
      curS <- S0*(u)^i*(d)^(j-i-1)
      Del[j,i+1] <- exp(-delta*h)*(V[j+1,i+2]-V[j+1,i+1])/(curS*u-curS*d)
    }
  }
  return (list(price = V[1,1],Delta0 = Del[1,1],allDelta=Del,allV = V) )
}
# Plot
del=c()
for (i in seq(85,110,0.5))
  {
  put=binTree(N=15,T=1,S=100, K=i,r=0.03,delta=0)
```

```
  del=append(del, put$Delta0)
}
plot(K, del, type="l", col="red", xlab="K", ylab="Initial Delta")
```

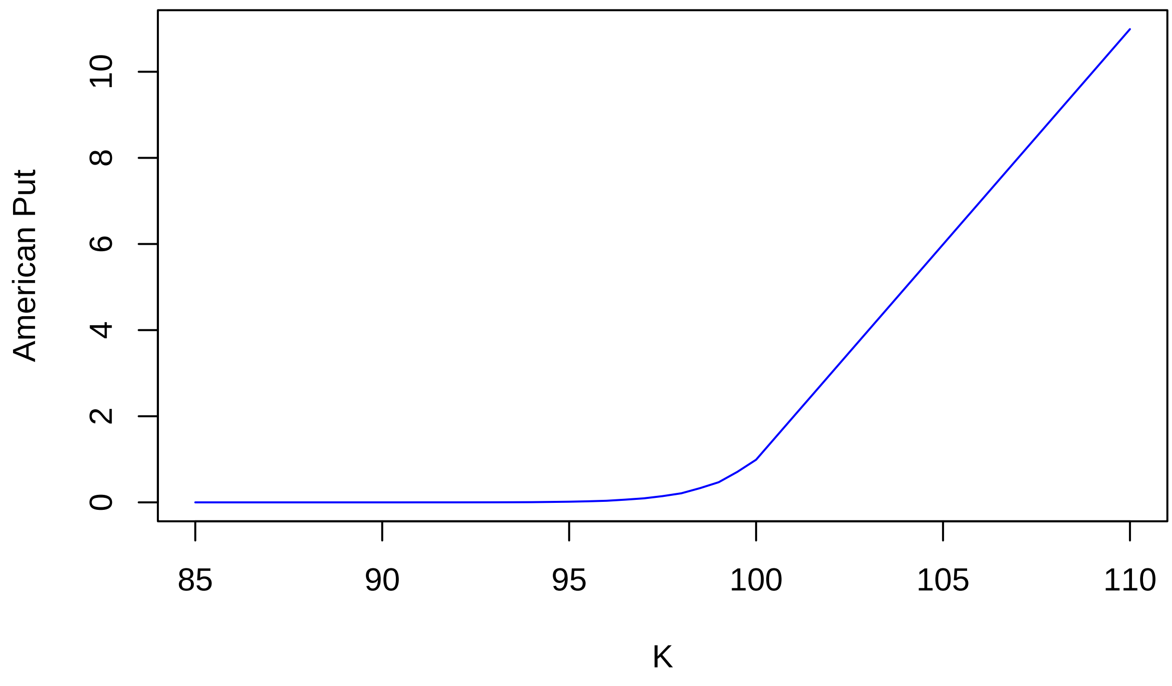


We see that the slope of the put price is convex and has a slope less than e^{-rt} recalling Chapter 9. From the binomial tree, we know the smallest stock price is 86.13 (d fifteen times). Therefore at all K less than 86.5, the put price is 0. Considering the delta, the slope is decreasing for K greater than 86.5, suggesting that we should continuously short some amount of stock in the replicating portfolio. The slope of the put price is increasing as the strike price increases, while the Delta price decreases as the strike price increases.

Question 4

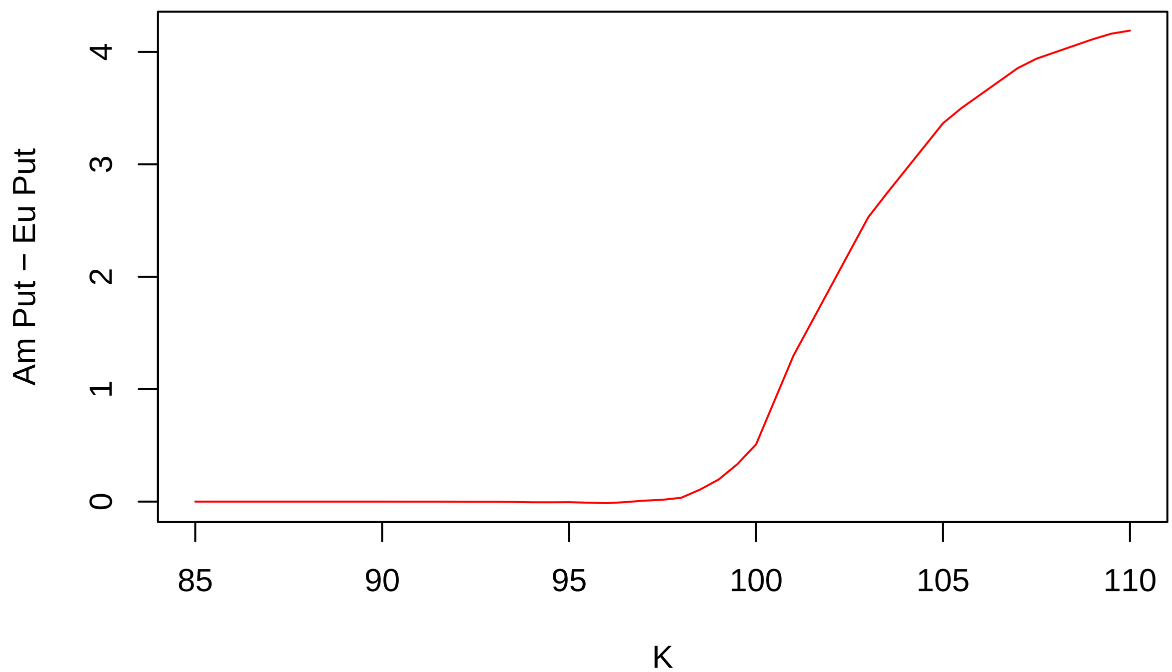
```
binTree <- function(N,T,S0,K,r,delta) {
  h <- T/N
  u <- 1.01
  d <- 100/101
  q <- (exp( (r-delta)*h)-d)/(u-d)
  V <- array(0, dim=c(N+1,N+1))
  Del <- array(0, dim=c(N+1,N+1))

  for (i in 0:N) { # terminal payoff
    curS <- S0*(u)^i*(d)^(N-i)
    V[N+1,i+1] <- max( K-curS, 0) # Put payoff
  }
```

```
  for (j in (N-1):1) {
    for (i in 0:j) {
      V[j,i+1] <-max(K-S0*(u)^i*(d)^(j-i),exp(-r*h)*(q*V[j+1,i+2]+(1-q)*V[j+1,i+1]))

    }
  }

  return (list(price = V[1,1],Delta0 = Del[1,1],allDelta=Del,allV = V) )
}


# Plot
K=seq(85,110,0.5)
priceA=c()
for (i in seq(85,110,0.5))
{
  put=binTree(15,T=1,S=100, K=i,r=0.05,delta=0)
  priceA=append(priceA,put$price)
}
plot(K, priceA,type="l",col="blue", xlab="K", ylab=" American Put ")
```




```
# Plot
K=seq(85,110,0.5)
priceA=c()
for (i in seq(85,110,0.5))
```

```r
{
  put=binTree(15,T=1,S=100, K=i,r=0.05,delta=0)
  priceA=append(priceA,put$price)
}
plot(K, (priceA-priceE),type="l",col="red", xlab="K", ylab="Am Put - Eu Put")
```



The American Put price is convex and has a slope less than 1. Similarly to the European put, the American put is not worth anything when K is small, however for a large strike, the American put price follows a linear slope. Comparing with the second plot, the American put is always worth at least as much as the European put.For large strikes, we can see that the American put is much more valuable than the European put and that there are instances where it would be wise to exercise, whereas with the European put this is not an option.

Question 5

```r
binomCall <- function(n,T,S0,K,r)
{
  h <- T/n
  d<-exp(-0.25*sqrt(h)); u<-exp(0.25*sqrt(h));
  q<- (exp((r)*h)-d)/(u-d)
  Sbin <- S0*d^n*(u/d)^(0:n)  # possible values of stock price
  qbin <- dbinom(0:n,n,q)  # corresponding probabilities using q
  Payoff <- pmax(Sbin-K,0)
```

```r
  return(exp(-r*T)*sum(qbin*Payoff))
}
```

a) Adjusting for n, we can compute for the price of the calls

```r
binomCall(4,1,100,100,0.05) # n = 4
```

```
## [1] 11.74255
```

```r
binomCall(8,1,100,100,0.05) # n = 8
```

```
## [1] 12.03253
```

```r
binomCall(15,1,100,100,0.05)
```

```
## [1] 12.48756
```

```r
binomCall(30,1,100,100,0.05)
```

```
## [1] 12.25395
```

```r
binomCall(60,1,100,100,0.05)
```

```
## [1] 12.29489
```

```r
binomCall(90,1,100,100,0.05)
```

```
## [1] 12.30857
```

```r
binomCall(120,1,100,100,0.05) # n = 120
```

```
## [1] 12.31542
```