Module 5.4: Geometric Brownian Motion-Based Option Valuation Models
R Commentary

See *Ch 5.4 Valuation GBM CT OVM*. The test program is in *GBMOVM Test.R*. The functions are contained in GBMOVM Functions.R.

GBMOVM Test.R (Selected Excerpts and Output)

The test program explores a variety of issues, not all of which are covered here.

```
# 5.4 GBMOVM Test.R
# Illustrating functions in R (function definitions in separate file)
# rmarkdown::render("5.4 GBMOVM Test.R", "word_document")
rm(list = ls())  # Take out the Environment "trash"
cat("\014") # Clear Console, making error checking easier.
while (!is.null(dev.list()))dev.off() # Clear old plots
par(family = 'Times New Roman') # Globally set fonts for graphs
# Generic test inputs
inputStockPrice = 100.0
inputStrikePrice = 100.0
inputInterestRate = 5.0        # In percent
inputDividendYield = 0.0       # In percent
inputVolatility = 30.0         # In percent
inputTimeToMaturity = 1
inputType = 1 # 1 for call, -1 for put
NumberOfObservations = 200
LowerBound = 0.5*inputStockPrice # Note centering on original values
UpperBound = 1.5*inputStockPrice
LowerBoundX = 0.5*inputStrikePrice # Note centering on original values
UpperBoundX = 1.5*inputStrikePrice
LowerBoundDY = 0.0 # Note not centering on original values
UpperBoundDY = 20.0
# Plot footers
TX = paste0('X=', inputStrikePrice)
TR = paste0(', r=', inputInterestRate)
Td = paste0(', d=', inputDividendYield)
TV = paste0(', Vol=', inputVolatility)
TT = paste0(', T=', inputTimeToMaturity)
sTitle = paste0(TX, TR, Td, TV, TT)
#
# Available functions
#
#  GBMInputData - list of inputs with associated names
#  PV1(Maturity, Rate) - present value of $1
#  B = GBMInputData
#  d1(B) - value of d1
#  d2(B) - value of d2
#  n(d) - standard normal PDF, given scalar d
#  N(d) - standard normal CDF, given scalar d
#  GBMOptionValue(B) - option value, type = 1 is call, type = -1 is put
#  OptionLowerBound(B) - option lower bounds
#  OptionUpperBound(B) - option upper bounds
#
#  Input matrix
#  GBMInputData - list of inputs with associated names
GBMInputData <- list(inputStockPrice, inputStrikePrice, inputInterestRate,
  inputDividendYield, inputVolatility, inputTimeToMaturity, inputType)
names(GBMInputData) <- c("StockPrice", "StrikePrice", "InterestRate",
  "DividendYield", "Volatility", "TimeToMaturity", "Type")
source("GBMOVM Functions.R")
#
# Test functions
#
UIValue = GBMInputData$StockPrice
TestPV1 = PV1(inputTimeToMaturity, inputInterestRate)
```

```
Testd1 = d1(GBMInputData)
Testd2 = d2(GBMInputData)
Testn = n(d1(GBMInputData))
TestN = N(d1(GBMInputData))
GBMInputData$Type = 1
CallUpperBound = OptionUpperBound(GBMInputData)
CallLowerBound = OptionLowerBound(GBMInputData)
CallValue = GBMOptionValue(GBMInputData)
GBMInputData$Type = -1
PutUpperBound = OptionUpperBound(GBMInputData)
PutLowerBound = OptionLowerBound(GBMInputData)
PutValue = GBMOptionValue(GBMInputData)
# Canonical case
GBMInputData$StockPrice = 100
GBMInputData$StrikePrice = 100
GBMInputData$InterestRate = 5.0
GBMInputData$DividendYield = 0.0
GBMInputData$Volatility = 30
GBMInputData$TimeToMaturity = 1.0
GBMInputData$Type = 1
CallValue = GBMOptionValue(GBMInputData)
GBMInputData$Type = -1
PutValue = GBMOptionValue(GBMInputData)
CallValue; PutValue
# # Homogeneity of degree one illustrated
# GBMInputData$StockPrice = 1
# GBMInputData$StrikePrice = 1
# GBMInputData$Type = 1
# CallValue = GBMOptionValue(GBMInputData)
# GBMInputData$Type = -1
# PutValue = GBMOptionValue(GBMInputData)
# CallValue; PutValue
# # Further illustration
# GBMInputData$StrikePrice = 0.0000001
# GBMInputData$Type = 1
# CallValue = GBMOptionValue(GBMInputData)
# GBMInputData$Type = -1
# PutValue = GBMOptionValue(GBMInputData)
# CallValue; PutValue
#
# Illustrations with plots
#
StepSize = (UpperBound - LowerBound)/NumberOfObservations
StockPrice <- c(1:NumberOfObservations)
CallValue <- c(1:NumberOfObservations)
PutValue <- c(1:NumberOfObservations)
CallLowerBound <- c(1:NumberOfObservations)
PutLowerBound <- c(1:NumberOfObservations)
CallUpperBound <- c(1:NumberOfObservations)
PutUpperBound <- c(1:NumberOfObservations)
CallTimeValue <- c(1:NumberOfObservations)
PutTimeValue <- c(1:NumberOfObservations)
for(i in 1:NumberOfObservations){
  StockPrice[i] <- as.double(LowerBound + (i-1)*StepSize)
  GBMInputData$StockPrice = StockPrice[i]
  GBMInputData$Type = 1
  CallLowerBound[i] <- OptionLowerBound(GBMInputData)
  CallUpperBound[i] <- OptionUpperBound(GBMInputData)
  CallValue[i] <- GBMOptionValue(GBMInputData)
  CallTimeValue[i] <- CallValue[i] - CallLowerBound[i]
  GBMInputData$Type = -1
  PutLowerBound[i] <- OptionLowerBound(GBMInputData)
  PutUpperBound[i] <- OptionUpperBound(GBMInputData)
  PutValue[i] <- GBMOptionValue(GBMInputData)
```

```
  PutTimeValue[i] <- PutValue[i] - PutLowerBound[i]
}
GBMInputData$StockPrice = inputStockPrice
# Option value plots
MaxValue = max(CallValue, CallLowerBound, CallUpperBound)
MinValue = min(CallValue, CallLowerBound, CallUpperBound)
ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
MaxValue = max(StockPrice); MinValue = min(StockPrice)
xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
legtxt = c("Call Value","Lower Bound", "Upper Bound")
mTitle = "GBM Call Value"
xTitle = "Stock Price"
yTitle = "Call Value"
lTitle = "Parameter"
plot(StockPrice, CallValue, type = "b", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(StockPrice, CallLowerBound, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
lines(StockPrice, CallUpperBound, type = "b", col ="black", xlim = xlim1,
      ylim = ylim1, pch = 3, cex = 0.5)
legend("right", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1,2,3), bty = "n", title = lTitle)
# Put value
MaxValue = max(PutValue, PutLowerBound, PutUpperBound)
MinValue = min(PutValue, PutLowerBound, PutUpperBound)
ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
MaxValue = max(StockPrice); MinValue = min(StockPrice)
xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
legtxt = c("Put Value","Lower Bound", "Upper Bound")
mTitle = "GBM Put Value"
xTitle = "Stock Price"
yTitle = "Put Value"
lTitle = "Parameter"
plot(StockPrice, PutValue, type = "b", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(StockPrice, PutLowerBound, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
lines(StockPrice, PutUpperBound, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 3, cex = 0.5)
legend("right", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1,2,3), bty = "n", title = lTitle)
# All together now
MaxValue = max(CallValue, CallLowerBound, PutValue, PutLowerBound)
MinValue = min(CallValue, CallLowerBound, PutValue, PutLowerBound)
ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
MaxValue = max(StockPrice); MinValue = min(StockPrice)
xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
legtxt = c("Call Value", "Call Lower Bound", "Put Value", "Put Lower Bound")
mTitle = "GBM Option Values"
xTitle = "Stock Price"
yTitle = "Option Value"
lTitle = "Parameter"
plot(StockPrice, CallValue, type = "b", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(StockPrice, CallLowerBound, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
lines(StockPrice, PutValue, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 3, cex = 0.5)
lines(StockPrice, PutLowerBound, type = "b", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 4, cex = 0.5)
legend("top", legtxt, cex = 0.75, lwd = c(1, 1, 1, 1), lty = c(1, 1, 1, 1),
```

```
  col = c("black","black","black","black"), pch = c(1, 2, 3, 4),
  bty = "n", title = lTitle)
#
# Option time value plots
#
MaxValue = max(CallTimeValue); MinValue = min(CallTimeValue)
ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
MaxValue = max(StockPrice); MinValue = min(StockPrice)
xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
legtxt = c("Call Value","Lower Bound")
mTitle = "GBM Call Time Value"
xTitle = "Stock Price"
yTitle = "Call Time Value"
lTitle = "Parameter"
plot(StockPrice, CallTimeValue, type = "b", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
# Put time value
MaxValue = max(PutTimeValue); MinValue = min(PutTimeValue)
ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
MaxValue = max(StockPrice); MinValue = min(StockPrice)
xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
legtxt = c("Put Value","Lower Bound")
mTitle = "GBM Put Time Value"
xTitle = "Stock Price"
yTitle = "Put Time Value"
lTitle = "Parameter"
plot(StockPrice, PutTimeValue, type = "b", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
# #
# # Option boundary plots
# #
# MaxValue = max(CallValue, CallLowerBound, CallUpperBound)
# MinValue = min(CallValue, CallLowerBound, CallUpperBound)
# ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
# MaxValue = max(StockPrice); MinValue = min(StockPrice)
# xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
# legtxt = c("Call Value", "Call Lower Bound", "Call Upper Bound")
# mTitle = "GBM Option Values"
# xTitle = "Stock Price"
# yTitle = "Value"
# lTitle = "Parameter"
# plot(StockPrice, CallValue, type = "b", main = mTitle,
#   sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
#   ylim = ylim1, pch = 1, cex = 0.5)
# lines(StockPrice, CallLowerBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 2, cex = 0.5)
# lines(StockPrice, CallUpperBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 3, cex = 0.5)
# legend("topleft", legtxt, cex = 0.75, lwd = c(1, 1, 1), lty = c(1, 1, 1),
#   col = c("black","black","black"), pch = c(1, 2, 3), bty = "n", title = lTitle)
# # Put bounds
# MaxValue = max(PutValue, PutLowerBound, PutUpperBound)
# MinValue = min(PutValue, PutLowerBound, PutUpperBound)
# ylim1 = c(1:2); ylim1[1] = MinValue; ylim1[2] = MaxValue
# MaxValue = max(StockPrice); MinValue = min(StockPrice)
# xlim1 = c(1:2); xlim1[1] = MinValue; xlim1[2] = MaxValue
# legtxt = c("Put Value", "Put Lower Bound", "Put Upper Bound")
# mTitle = "GBM Option Values"
# xTitle = "Stock Price"
# yTitle = "Value"
# lTitle = "Parameter"
# plot(StockPrice, PutValue, type = "b", main = mTitle,
```

```
#   sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
#   ylim = ylim1, pch = 1, cex = 0.5)
# lines(StockPrice, PutLowerBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 2, cex = 0.5)
# lines(StockPrice, PutUpperBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 3, cex = 0.5)
# legend("bottomleft", legtxt, cex = 0.75, lwd = c(1, 1, 1), lty = c(1, 1, 1),
#   col = c("black","black","black"), pch = c(1, 2, 3), bty = "n", title = lTitle)
# # Open a pdf file
# pdf("Put Values w Boundaries.pdf")
# plot(StockPrice, PutValue, type = "b", main = mTitle,
#   sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
#   ylim = ylim1, pch = 1, cex = 0.5)
# lines(StockPrice, PutLowerBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 2, cex = 0.5)
# lines(StockPrice, PutUpperBound, type = "b", col ="black", xlim = xlim1,
#   ylim = ylim1, pch = 3, cex = 0.5)
# legend("bottomleft", legtxt, cex = 0.75, lwd = c(1, 1, 1), lty = c(1, 1, 1),
#   col = c("black","black","black"), pch = c(1, 2, 3), bty = "n", title = lTitle)
#
# Normalized analysis wrt strike price
#
StrikePrice <- c(1:NumberOfObservations)
StepSize = (UpperBoundX - LowerBoundX)/NumberOfObservations
for(i in 1:NumberOfObservations){
  StrikePrice[i] <- as.double(LowerBoundX + (i-1)*StepSize)
  GBMInputData$StrikePrice = StrikePrice[i]
  GBMInputData$Type = 1
  CallLowerBound[i] <- OptionLowerBound(GBMInputData)
  CallUpperBound[i] <- OptionUpperBound(GBMInputData)
  CallValue[i] <- GBMOptionValue(GBMInputData)
  CallTimeValue[i] <- CallValue[i] - CallLowerBound[i]
  GBMInputData$Type = -1
  PutLowerBound[i] <- OptionLowerBound(GBMInputData)
  PutUpperBound[i] <- OptionUpperBound(GBMInputData)
  PutValue[i] <- GBMOptionValue(GBMInputData)
  PutTimeValue[i] <- PutValue[i] - PutLowerBound[i]
}
GBMInputData$StrikePrice = inputStrikePrice
x <- 100*(StrikePrice/GBMInputData$StockPrice)
MaxValuex = max(x); MinValuex = min(x)
xlim1 = c(1:2); xlim1[1] = MinValuex; xlim1[2] = MaxValuex
# Plot: Normalized option values wrt normalized strike price
yC <- 100*(CallValue/GBMInputData$StockPrice)
yP <- 100*(PutValue/GBMInputData$StockPrice)
MaxValuey = max(yC, yP); MinValuey = min(yC, yP)
ylim1 = c(1:2); ylim1[1] = MinValuey; ylim1[2] = MaxValuey
legtxt = c("Normalized Call Value", "Normalized Put Value")
mTitle = "Normalized Option Values"
xTitle = "Strike Price/Stock Price (%)"
yTitle = "Normalized Option Values (%)"
lTitle = "Parameter"
plot(x, yC, type = "p", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(x, yP, type = "p", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
legend("top", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1, 2), bty = "n", title = lTitle)
# Plot: Normalized time values wrt normalized strike price
yC <- 100*(CallTimeValue/GBMInputData$StockPrice)
yP <- 100*(PutTimeValue/GBMInputData$StockPrice)
MaxValuey = max(yC, yP); MinValuey = min(yC, yP)
ylim1 = c(1:2); ylim1[1] = MinValuey; ylim1[2] = MaxValuey
```

```
legtxt = c("Normalized Call Time Value", "Normalized Put Time Value")
mTitle = "Normalized Option Time Values"
xTitle = "Strike Price/Stock Price (%)"
yTitle = "Normalized Option Time Values (%)"
lTitle = "Parameter"
plot(x, yC, type = "p", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(x, yP, type = "p", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
legend("topleft", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1, 2), bty = "n", title = lTitle)
#
# Normalized analysis wrt dividend yield
#
DividendYield <- c(1:NumberOfObservations)
StepSize = (UpperBoundDY - LowerBoundDY)/NumberOfObservations
for(i in 1:NumberOfObservations){
  DividendYield[i] <- as.double(LowerBoundDY + (i-1)*StepSize)
  GBMInputData$DividendYield = DividendYield[i]
  GBMInputData$Type = 1
  CallLowerBound[i] <- OptionLowerBound(GBMInputData)
  CallUpperBound[i] <- OptionUpperBound(GBMInputData)
  CallValue[i] <- GBMOptionValue(GBMInputData)
  CallTimeValue[i] <- CallValue[i] - CallLowerBound[i]
  GBMInputData$Type = -1
  PutLowerBound[i] <- OptionLowerBound(GBMInputData)
  PutUpperBound[i] <- OptionUpperBound(GBMInputData)
  PutValue[i] <- GBMOptionValue(GBMInputData)
  PutTimeValue[i] <- PutValue[i] - PutLowerBound[i]
}
GBMInputData$DividendYield = inputDividendYield
x <- DividendYield
MaxValuex = max(x); MinValuex = min(x)
xlim1 = c(1:2); xlim1[1] = MinValuex; xlim1[2] = MaxValuex
# Plot: Normalized option values wrt normalized strike price
yC <- 100*(CallValue/GBMInputData$StockPrice)
yP <- 100*(PutValue/GBMInputData$StockPrice)
MaxValuey = max(yC, yP); MinValuey = min(yC, yP)
ylim1 = c(1:2); ylim1[1] = MinValuey; ylim1[2] = MaxValuey
legtxt = c("Normalized Call Value", "Normalized Put Value")
mTitle = "Normalized Option Values"
xTitle = "Dividend Yield (%)"
yTitle = "Normalized Option Values (%)"
lTitle = "Parameter"
plot(x, yC, type = "p", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
  ylim = ylim1, pch = 1, cex = 0.5)
lines(x, yP, type = "p", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
legend("topleft", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1, 2), bty = "n", title = lTitle)
# Plot: Normalized time values wrt normalized strike price
yC <- 100*(CallTimeValue/GBMInputData$StockPrice)
yP <- 100*(PutTimeValue/GBMInputData$StockPrice)
MaxValuey = max(yC, yP); MinValuey = min(yC, yP)
ylim1 = c(1:2); ylim1[1] = MinValuey; ylim1[2] = MaxValuey
legtxt = c("Normalized Call Time Value", "Normalized Put Time Value")
mTitle = "Normalized Option Time Values"
xTitle = "Dividend Yield (%)"
yTitle = "Normalized Option Time Values (%)"
lTitle = "Parameter"
plot(x, yC, type = "p", main = mTitle,
  sub = sTitle, xlab = xTitle, ylab = yTitle, col = "black", xlim = xlim1,
```

```
  ylim = ylim1, pch = 1, cex = 0.5)
lines(x, yP, type = "p", col ="black", xlim = xlim1,
  ylim = ylim1, pch = 2, cex = 0.5)
legend("topright", legtxt, cex = 0.75, lwd = c(1, 1), lty = c(1, 1),
  col = c("black","black"), pch = c(1, 2), bty = "n", title = lTitle)
```

GBMOVM Functions.R

The key function in GBMOVM Functions.R is the valuation function that is written generically to be able to value either the call or put option.

```
# GBMOVM Functions.R
#
#  INPUT STRUCTURE
#  GBMInputData - list of inputs with associated names; percent, not decimal
#  GBMInputData <- list(inputStockPrice, inputStrikePrice, inputInterestRate,
#    inputDividendYield, inputVolatility, inputTimeToMaturity, inputType)
#  names(GBMInputData) <- c("StockPrice", "StrikePrice", "InterestRate",
#    "DividendYield", "Volatility", "TimeToMaturity", "Type")
#
# Available functions
#  PV1(Maturity, Rate) - present value of $1
#  B = GBMInputData
#  d1(B) - value of d1
#  d2(B) - value of d2
#  n(d) - standard normal PDF, given scalar d
#  N(d) - standard normal CDF, given scalar d
#  GBMOptionValue(B) - option value, type = 1 is call, type = -1 is put
#  OptionLowerBound(B) - option lower bounds
#  OptionUpperBound(B) - option upper bounds
#
# Functions for GBMOVM-related calculations
#
PV1 = function(Maturity, Rate){
  return(exp( -(Rate/100.0) * Maturity) )
}
# d - functions used in GBMOVM
# with: Evaluates R expressions based on a set of data (B here)
d1 = function(B){
  with(B, {
    Num = ( ((InterestRate - DividendYield) / 100) + ((Volatility/100)^2)/2) *
      TimeToMaturity
    Num = log(StockPrice/StrikePrice) + Num
    Den = (Volatility/100)*sqrt(TimeToMaturity)
    return( Num/Den )
  })
}

d2 = function(B){
  with(B, {
    return( d1(B) - (Volatility/100)*sqrt(TimeToMaturity) )
  })
}
# Normal distribution functions
# n - probability density function of standard normal (0,1)
n = function(d){
  return( exp( -(d^2) / 2.0 ) / ( sqrt( 2.0 * pi ) ) )
}
# N - cumulative distribution function of standard normal (0,1)
N = function(d){
  return( as.numeric(integrate(n,-Inf,d)[1]) )
}
```

Key function to produce option values.

```
# GBMOVM
```

```
GBMOptionValue = function(B){
  with(B, {
    OptionValue = Type * StockPrice * PV1(TimeToMaturity, DividendYield) *
      N(Type * d1(B)) - Type * StrikePrice * PV1(TimeToMaturity, InterestRate) *
      N(Type * d2(B))
    LowerBound = Type * StockPrice * PV1(TimeToMaturity, DividendYield) -
      Type * StrikePrice * PV1(TimeToMaturity, InterestRate)
    return( max(OptionValue, LowerBound) )
  })
}
# Lower bound
OptionLowerBound = function(B){
  with(B, {
    LowerBound = Type * StockPrice * PV1(TimeToMaturity, DividendYield) -
      Type * StrikePrice * PV1(TimeToMaturity, InterestRate)
    LowerBound = max(0, LowerBound)
    return( LowerBound )
  })
}
# Upper bound
OptionUpperBound = function(B){
  with(B, {
    if(Type == 1)UpperBound = StockPrice * PV1(TimeToMaturity, DividendYield)
    if(Type == -1)UpperBound = StrikePrice * PV1(TimeToMaturity, InterestRate)
    return( UpperBound )
  })
}
```

With these user-defined functions built in R, several plots are produced.