Module 5.7

Geometric Brownian Motion
Compound Option Valuation Model
(GBM COVM)

1

Overview

- Review assumptions of GBM-based COVM
- Explore two versions of dividends
- Explain the COVM
- Review selected plots and related R code

2

Central Finance Concepts

- Compound option (CO) – an option on an option
- Notation and expiration dates
- Selected plots

3

Compound Option (CO)

- An option on an option
- Two expiration dates: $t < T_1 < T_2$,
  - t – valuation date
  - T_1 – compound option's expiration date
  - T_2 – underlying option's expiration date
- Two strike prices: $0 < X_C$ and $0 < X_U$
  - X_C – strike price of the compound option
  - X_U – strike price of the underlying option

4

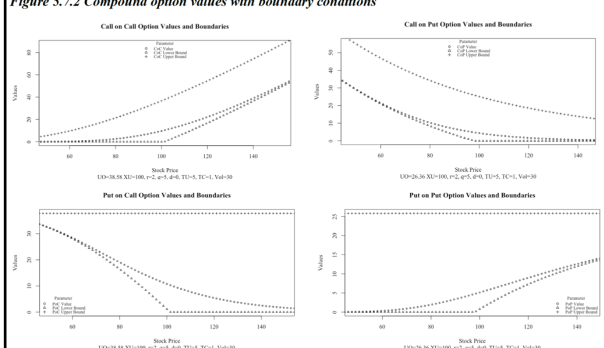
Time Line Illustrated

Figure 5.7.1. Three Important Dates for Compound Options with Notation

Compound Option Trade Date (t)	Compound Option Expiration Date (T_1)	Underlying Option Expiration Date (T_2)

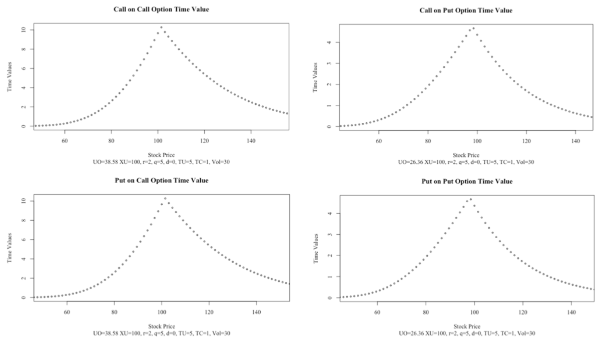
5

Figure 5.7.2 Compound option values with boundary conditions



6

Figure 5.7.3 Compound option time values



Quantitative Finance Material

- Compound option basics
- Underlying assumptions same as GBMOVM
- Boundaries and parities
- COVM
- Selected R code

Underlying Options at Maturity

- Plain Vanilla Call

$$c_{T_2}(S,X_U,T_2) = \max(0, S_{T_2} - X_U)$$

- Plain Vanilla Put

$$p_{T_2}(S,X_U,T_2) = \max(0, X_U - S_{T_2})$$

CO at Maturity

- Cacall: $CoC_{T_1}\left[c(S,X_U,T_2),X_C,T_1\right] = \max\left[0, c_{T_1}(S,X_U,T_2) - X_C\right]$
- Caput: $CoP_{T_1}\left[p(S,X_U,T_2),X_C,T_1\right] = \max\left[0, p_{T_1}(S,X_U,T_2) - X_C\right]$
- Pucall: $PoC_{T_1}\left[c(S,X_U,T_2),X_C,T_1\right] = \max\left[0, X_C - c_{T_1}(S,X_U,T_2)\right]$
- Puput: $PoP_{T_1}\left[p(S,X_U,T_2),X_C,T_1\right] = \max\left[0, X_C - p_{T_1}(S,X_U,T_2)\right]$

Selected Notation

- ι_C – compound option indicator function
  - +1 if CO is call
  - –1 if CO is put
- ι_U – underlying option indicator function
  - +1 if option is call
  - –1 if option is put

Selected Notation

- S_t – underlying instrument at time t
- σ – volatility of underlying instrument
- δ – dividend yield underlying instrument
- $\hat{q}$ – yield on underlying option
  - Firm value is underlying, stock is option
  - δ does not equal $\hat{q}$ as debt requires payment
- $B_{t,T,x} = e^{-x(T-t)}$ – present value of $1 at rate *x*

Underlying Boundaries

- Call LB: $c_t \geq \max\left(0, B_{t,T_2,\delta} S_t - B_{t,T_2,r_c} X_U\right)$
- Call UB: $c_t \leq B_{t,T_2,\delta} S_t$
- Put LB: $p_t \geq \max\left(0, B_{t,T_2,r_c} X_U - B_{t,T_2,\delta} S_t\right)$
- Put UB: $p_t \leq B_{t,T_2,r_c} X_U$

13

Call on Call/Put Boundaries

- Cacall LB: $CoC_t \geq \max\left(0, B_{t,T_1,\hat{q}} c_t - B_{t,T_1,r_c} X_C\right)$
- Cacall UB: $CoC_t \leq B_{t,T_1,\hat{q}} c_t$
- Caput LB: $CoP_t \geq \max\left(0, B_{t,T_1,\hat{q}} p_t - B_{t,T_1,r_c} X_C\right)$
- Caput UB: $CoP_t \leq B_{t,T_1,\hat{q}} p_t$

14

Put on Call/Put Boundaries

- Pucall LB: $PoC_t \geq \max\left(0, B_{t,T_1,r_c} X_C - B_{t,T_1,\hat{q}} c_t\right)$
- Pucall UB: $PoC_t \leq B_{t,T_1,r_c} X_C$
- Puput LB: $PoP_t \geq \max\left(0, B_{t,T_1,r_c} X_C - B_{t,T_1,\hat{q}} p_t\right)$
- Puput UB: $PoP_t \leq B_{t,T_1,r_c} X_C$

15

Put-Call Parities

$$B_{t,T_2,\hat{q}_c} c_t - B_{t,T_2,\hat{q}_p} p_t = B_{t,T_2,\delta} S_t - B_{t,T_2,r_c} X_C$$

$$CoC_t - PoC_t = B_{t,T_1,\hat{q}} c_t - B_{t,T_1,r_c} X_C$$

$$CoP_t - PoP_t = B_{t,T_1,\hat{q}} p_t - B_{t,T_1,r_c} X_C$$

$$CoC_t - PoC_t - \left(CoP_t - PoP_t\right) = B_{t,T_2,\delta} S_t - B_{t,T_2,r_c} X_U$$

16

COVM

$$CO\left(S,t,T_1,T_2,\iota_C,\iota_U\right) = \iota_C \iota_U S_t B_{t,T_2,\delta} B_{T_1,T_2,-\hat{q}} N_2\left(\iota_C \iota_U d_{11}, \iota_U d_{12}; \iota_C \rho\right)$$
$$-\iota_C \iota_U X_U B_{t,T_2,r} B_{T_1,T_2,-\hat{q}} N_2\left(\iota_C \iota_U d_{21}, \iota_U d_{22}; \iota_C \rho\right) - \iota_C X_C B_{t,T_1,r} N\left(\iota_C \iota_U d_{21}\right)$$

$$N_2\left(a,b;\rho\right) \equiv \int_{-\infty}^{a} \int_{-\infty}^{b} \frac{\exp\left\{-\frac{z_1^2 - 2\rho z_1 z_2 + z_2^2}{2\left(1-\rho^2\right)}\right\}}{2\pi\sqrt{1-\rho^2}} dz_1\, dz_2$$

$$d_{21} \equiv \frac{\ln\left(\frac{S_t B_{t,T_1,-(r-\delta)}}{S_{T_1}^*}\right) - \frac{\sigma_{t,T_1}^2}{2}}{\sigma_{t,T_1}} \qquad d_{11} \equiv \frac{\ln\left(\frac{S_t B_{t,T_1,-(r-\delta)}}{S_{T_1}^*}\right) + \frac{\sigma_{t,T_1}^2}{2}}{\sigma_{t,T_1}} = d_{21} + \sigma_{t,T_1}$$

17

COVM

$$d_{22} \equiv \frac{\ln\left(\frac{S_t B_{t,T_2,-(r-\delta)}}{X_U}\right) - \frac{\sigma_{t,T_2}^2}{2}}{\sigma_{t,T_2}} \qquad d_{12} \equiv \frac{\ln\left(\frac{S_t B_{t,T_2,-(r-\delta)}}{X_U}\right) + \frac{\sigma_{t,T_2}^2}{2}}{\sigma_{t,T_2}} = d_{22} + \sigma_{t,T_2}$$

$$\iota_U S_{T_1}^* B_{T_1,T_2,\delta-\hat{q}} N_1\left(\iota_U d_{1,T_1,T_2}^*\right) - \iota_U X_U B_{T_1,T_2,r-\hat{q}} N_1\left(\iota_U d_{2,T_1,T_2}^*\right) - X_C = 0$$

$$d_{2,T_1,T_2}^* = \frac{\ln\left(\frac{S_{T_1}^* B_{T_1,T_2,-(r-\delta)}}{X_U}\right) - \frac{\sigma_{T_1,T_2}^2}{2}}{\sigma_{T_1,T_2}} \qquad d_{1,T_1,T_2}^* = \frac{\ln\left(\frac{S_{T_1}^* B_{T_1,T_2,-(r-\delta)}}{X_U}\right) + \frac{\sigma_{T_1,T_2}^2}{2}}{\sigma_{T_1,T_2}} = d_{2,T_1,T_2}^* + \sigma_{T_1,T_2}$$

18

Critical Stock Price Example

```
# Critical stock price for compound option
COCriticalStockPrice <- function(C, UOV, L, U){
  with(C, {
    inputUnderlying <- S
    inputUnderlyingStrikePrice <- XU
    inputInterestRate <- r
    inputUnderlyingYield <- d
    inputOptionYield <- q
    inputVolatility <- v
    inputTimeToMaturity <- TU - TC
    inputType <- iU
    GBMInputData <- list(inputUnderlying, inputUnderlyingStrikePrice,
      inputInterestRate, inputUnderlyingYield,  inputOptionYield,
      inputVolatility, inputTimeToMaturity, inputType)
    names(GBMInputData) <- c("StockPrice", "StrikePrice", "InterestRate",
      "DividendYield", "OptionYield", "Volatility", "TimeToMaturity", "Type")
    return(GBMDYOptionImpliedStockPrice(GBMInputData, UOV, L, U))
  })
}
```

19

19

d_{11} Calculation

```
# d11
COd11 <- function(C, L, U){
  with(C, {
    r <- r/100
    d <- d/100
    v <- v/100
    CSP <- COCriticalStockPrice(C, XC, L, U)
    B1Nrd <- exp((r - d)*TC)
    v1 <- v*sqrt(TC)
    d11 <- (log( (S*B1Nrd) /CSP) + ((v1^2)/2) )/v1
    return(d11)
  })
}
```

 20

20

```
COValue <- function(C, L, U){
  with(C, {
    r <- r/100
    d <- d/100
    q <- q/100
    v <- v/100
    B2d <- exp(-d*TU)
    B12q <- exp(-q*(TU - TC))
    B12Nq <- exp(q*(TU - TC))
    B2r <- exp(-r*TU)
    B1r <- exp(-r*TC)
    B2q <- exp(-q*TU)
    B12d <- exp(-d*(TU - TC))
    d11 <- COd11(C, L, U)
    d21 <- COd21(C, L, U)
    d12 <- COd12(C)
    d22 <- COd22(C)
    mean1 <- rep(0,2)
    lower1 <- rep(-Inf,2)
    corr1 <- diag(2)
    corr1[lower.tri(corr1)] <- iC*sqrt(TC/TU)
    corr1[upper.tri(corr1)] <- iC*sqrt(TC/TU)
    upper1 <- c(iC*iU*d11,iU*d12)
    N2d11d12 <- pmvnorm(lower=lower1, upper=upper1, mean=mean1, corr=corr1)[1]
    upper1 <- c(iC*iU*d21, iU*d22)
    N2d21d22 <- pmvnorm(lower=lower1, upper=upper1, mean=mean1, corr=corr1)[1]
    CO <- iC*iU*S*B12Nq*B2d*N2d11d12
    CO <- CO - iC*iU*XU*B12Nq*B2r*N2d21d22 - iC*XC*B1r*NAp(iC*iU*d21)
    return(CO)
  })
}
```

21

21

Sample Inputs

```
> inputiC <- 1                                  # 1-CO call, -1-CO put
> inputiU <- 1                                  # 1-UO call, -1-UO put
> inputUnderlying <- 100.0                      # Must be positive
> inputUnderlyingStrikePrice <- 100.0           # Must be positive
> inputCompoundStrikePrice <- 10                # Must be positive
> inputInterestRate <- 2.0                      # Must be positive
> inputUnderlyingYield <- 0.0                   # Payout of underlying instrument
> inputOptionYield <- 0.0                       # Payout of option
> inputVolatility <- 30.0                       # Must be positive
> inputUnderlyingTimeToMaturity <- 5.0          # Must be positive
> inputCompoundTimeToMaturity <- 1.0            # Must be positive
```

 22

22

Base Outputs

```
> UCallValue <<- GBMOptionValue(GBMInputData)
> GBMInputData$Type <- -1
> UPutValue <<- GBMOptionValue(GBMInputData)
> UCallValue; UPutValue
[1] 30.04362
[1] 20.52736
```

 23

23

```
> COInputData$iC <- 1
> COInputData$iU <- 1
> CoCCSP <- COCriticalStockPrice(COInputData, UCallValue, LowerBound, UpperBound)
> CoC <- COValue(COInputData, LowerBound, UpperBound)
> COInputData$iU <- -1
> CoPCSP <- COCriticalStockPrice(COInputData, UPutValue, LowerBound, UpperBound)
> CoP <- COValue(COInputData, LowerBound, UpperBound)
> COInputData$iC <- -1
> COInputData$iU <- 1
> PoCCSP <- COCriticalStockPrice(COInputData, UCallValue, LowerBound, UpperBound)
> PoC <- COValue(COInputData, LowerBound, UpperBound)
> COInputData$iU <- -1
> PoPCSP <- COCriticalStockPrice(COInputData, UPutValue, LowerBound, UpperBound)
> PoP <- COValue(COInputData, LowerBound, UpperBound)
> CoCCSP; CoPCSP; PoCCSP; PoPCSP
[1] 104.8807
[1] 95.66102
[1] 104.8807
[1] 95.66102
> CoC; CoP; PoC; PoP
[1] 13.14699
[1] 4.039534
[1] 3.224262
[1] 3.633066
```

 24

24

Set x-axis Generically

```
# Set x range: Find break point on x-axis
z <- -99
for(i in 2:NumberOfObservations){
  if(CoCLowerBound[i] > CoCLowerBound[i-1] && z < 0){
    MidPoint <- StockPrice[i]
    z <- 99
  }
}
Range <- 0.5
MaxXValue = MidPoint * (1 + Range)
MinXValue = MidPoint * (1 - Range)
xlim1 = c(1:2)
xlim1[1] = MinXValue
xlim1[2] = MaxXValue
```


Set y-axis Generically

```
# Set y range: Find break point on y-axis
Maxz <- -99
Minz <- -99
for(i in 1:NumberOfObservations){
  if(StockPrice[i] > MaxXValue && Maxz < 0){ # S just greater than S(Max)
    MaxXi <- i
    Maxz <- 99
  }
  if(StockPrice[i] > MinXValue && Minz < 0){ # S just less than S(Min)
    MinXi <- i
    Minz <- 99
  }
}
```


Summary

- Reviewed assumptions of GBM-based COVM
- Explored two versions of dividends
- Explained the COVM
- Reviewed selected plots and related R code