Valuation of Contingent Convertibles with Derivatives

Nicolay Schmitt

Frankfurt School of Finance and Management

A thesis submitted for the degree of

Master of Science in Finance

August 2016

This thesis is dedicated to my parents for their love and support.

Thank you!

Abstract

Financial crises have led to higher regulatory standards on the capital adequacy of banks. They are required to hold more capital with loss absorbance capacities on their balance sheet. In conjunction with this development, contingent convertible bonds (CoCos) have become an attractive instrument for banks to seek new capital. The defining characteristic of CoCos is the automatic conversion into common equity when a predetermined trigger is met. Loss absorbing capital is created, which instantly improves the capital structure of distressed banks.

The thesis scrutinizes the valuation of CoCos. Three major approaches are examined: the structural approach in accordance to Pennacchi (2010) and both the credit derivatives approach respectively the equity derivatives approach pursuant to De Spiegeleer and Schoutens (2011). The application covers sensitivity analysis to further understand the dynamics of the different methodologies. Based on a case study the viability of those approaches is evaluated.

Contents

List of Figures

Chapter 1

Introduction and Motivation

1.1 Introduction

1.2 Literature Overview

1.3 Motivation

1.4 Methodology

Chapter 2

Structure of CoCos

2.1 Description of CoCos

2.2 Payoff and Risk Profile

2.3 Conversion Trigger

2.3.1 Market Trigger

2.3.2 Accounting Trigger

2.3.3 Regulatory Trigger

2.3.4 Multivariate Trigger

2.4 Conversion Details

2.4.1 Conversion Fraction

- conversion fraction α
- face value N
- conversion amount $N \times \alpha$
- amount remaining in case of partial equity conversion $N \times (1 - \alpha)$

2.4.2 Conversion Price and Ratio

- conversion rate C_r
- conversion price C_p

- recovery rate R_{CoCo}
- stock price at trigger event S_T^*
- loss attributable to CoCo holders L_{CoCo}

$$C_p = \frac{\alpha N}{C_r} \tag{2.1}$$

$$C_r = \frac{\alpha N}{C_p} \tag{2.2}$$

$$R_{CoCo} = \frac{S_T^*}{C_p} \tag{2.3}$$

$$L_{CoCo} = N - (1 - R_{CoCo}) = N\left(1 - \frac{S_T^*}{C_p}\right) \tag{2.4}$$

$$P_T = \begin{cases} (1-\alpha)N + C_r S_T^* & \text{if converted} \\ N & \text{if not converted} \end{cases} \tag{2.5}$$

Chapter 3

Theory of Pricing

3.1 Credit Derivative Approach

3.1.1 Intensity-based Approach

$$p^* = 1 - \exp\left(-\lambda_{Trigger} \times T\right) \tag{3.1}$$

$$s_{CoCo} = (1 - R_{CoCo}) \times \lambda_{Trigger} = Loss_{CoCo} \times \lambda_{Trigger} \tag{3.2}$$

$$Loss_{CoCo} = N - C_r \times S^* = N\left(1 - \frac{S^*}{C_P}\right) \tag{3.3}$$

$$R_{CoCo} = \frac{S^*}{C_p} \tag{3.4}$$

$$p^* = \Phi\left(\frac{\log\left(\frac{S^*}{S}\right) - \mu T}{\sigma\sqrt{T}}\right) + \left(\frac{S^*}{S}\right)^{\frac{2\mu}{\sigma^2}} \Phi\left(\frac{\log\left(\frac{S^*}{S}\right) + \mu T}{\sigma\sqrt{T}}\right) \tag{3.5}$$

$$\lambda_{Trigger} = -\frac{\log\left(1 - p^*\right)}{T} \tag{3.6}$$

$$s_{CoCo} = -\frac{\log\left(1 - p^*\right)}{T} \times \left(1 - \frac{S^*}{C_p}\right) \tag{3.7}$$

3.1.2 Application to CoCos

3.1.3 Data Requirements and Calibration

3.1.4 Pricing Example

3.2 Equity Derivative Approach

Sources: Erismann (2015), De Spiegeleer and Schoutens (2011)

$$
\begin{aligned}
P_T &= \mathbb{1}_{\{\tau>T\}} N + \left[(1-\alpha)N + \frac{\alpha N}{C_p S^*}\right] \mathbb{1}_{\{\tau\leq T\}} \\
&= N + \left[\frac{\alpha N}{C_p} S^* - \alpha N\right] \mathbb{1}_{\{\tau\leq T\}} \\
&= N + \left[C_r S^* - \alpha N\right] \mathbb{1}_{\{\tau\leq T\}} \\
&= N + C_r \left[S^* - \frac{\alpha N}{C_r}\right] \mathbb{1}_{\{\tau\leq T\}} \\
&= N + C_r \left[S^* - C_p\right] \mathbb{1}_{\{\tau\leq T\}}
\end{aligned}
$$

$$
V_t^{ed} = V_t^{cb} - V_{t_i}^{dibi} + V_t^{difwd} \tag{3.8}
$$

3.2.1 Corporate Bonds

$$
V_t^{cb} = \sum_{i=t}^{T} c_i \exp\left(-r t_i\right) + N \exp\left[-r\left(T-t\right)\right] \tag{3.9}
$$

3.2.2 Binary Options

$$
\begin{aligned}
V_t^{dibi}\left(c_i, S^*, t\right) = \alpha \sum_{i=1}^{k} c_i \exp\left(-r t_i\right) \Bigg[& \Phi\left(-x_{1i} + \sigma\sqrt{t_i}\right) \\
& + \left(\frac{S^*}{S_t}\right)^{2\lambda-2} \Phi\left(y_{1i} - \sigma\sqrt{t_i}\right)\Bigg]
\end{aligned} \tag{3.10}
$$

with

$$
\begin{aligned}
x_{1i} &= \frac{\log\left(\frac{S_t}{S^*}\right)}{\sigma\sqrt{t_i}} + \lambda\sigma\sqrt{t_i} \\
y_{1i} &= \frac{\log\left(\frac{S^*}{S_t}\right)}{\sigma\sqrt{t_i}} + \lambda\sigma\sqrt{t_i} \\
\lambda &= \frac{r - q + \frac{\sigma^2}{2}}{\sigma^2}
\end{aligned}
$$

3.2.3 Down-And-In Forward

$$
\max\left(S_T - K\right) \text{ if } \min_{0 \leq t \leq T}\left(S_T\right) \leq S^* \tag{3.11}
$$

$$
\max\left(K - S_T\right) \text{ if } \min_{0 \leq t \leq T}\left(S_T\right) \leq S^* \tag{3.12}
$$

$$
\begin{aligned}
V_t^{dic}\left(S_t, S^*, K\right) = {} & S_t \exp\left[-q\left(T-t\right)\right]\left(\frac{S^*}{S_t}\right)^{2\lambda}\Phi\left(y\right) \\
& - K\exp\left[-r\left(T-t\right)\right]\left(\frac{S^*}{S_t}\right)^{2\lambda-2}\Phi\left(y - \sigma\sqrt{T-t}\right)
\end{aligned} \tag{3.13}
$$

with

$$
\begin{aligned}
K &= C_p \\
y &= \frac{\log\left(\frac{S^{*2}}{S_t K}\right)}{\sigma\sqrt{T-t}} + \lambda\sigma\sqrt{T-t} \\
\lambda &= \frac{r - q + \frac{\sigma^2}{2}}{\sigma^2}
\end{aligned}
$$

$$
\begin{aligned}
V_t^{dip}\left(S_t, S^*, K\right) = {} & S_t \exp\left[-q\left(T-t\right)\right]\left(\frac{S^*}{S_t}\right)^{2\lambda}\left[\Phi\left(y\right) - \Phi\left(y_1\right)\right] \\
& - K\exp\left[-r\left(T-t\right)\right]\left(\frac{S^*}{S_t}\right)^{2\lambda-2}\left[\Phi\left(y - \sigma\sqrt{T-t}\right) - \Phi\left(y_1 - \sigma\sqrt{T}\right)\right] \\
& + K\exp\left[-r\left(T-t\right)\right]\Phi\left(x_1 + \sigma\sqrt{T-t}\right) \\
& - S_t\exp\left[-q\left(T-t\right)\right]\Phi\left(-x_1\right)
\end{aligned} \tag{3.14}
$$

with

$$x_1 = \frac{\log\left(\frac{S_t}{S^*}\right)}{\sigma\sqrt{T-t}} + \lambda\sigma\sqrt{T-t}$$
$$y_1 = \frac{\log\left(\frac{S^*}{S_t}\right)}{\sigma\sqrt{T-t}} + \lambda\sigma\sqrt{T-t}$$

$$\min(S_t) \leq S^* : P_T = S_T - K = \max(S_T - K) - \max(K - S_T) \tag{3.15}$$

$$\min(S_t) > S^* : P_T = 0 \tag{3.16}$$

$$\begin{aligned} V_t^{difwd} = C_r \Bigg[& S_t \exp[-q(T-t)] \left(\frac{S^*}{S_t}\right)^{2\lambda} \Phi(y_1) \\ & - K \exp[-r(T-t)] \left(\frac{S^*}{S_t}\right)^{2\lambda-2} \Phi\left(y_1 - \sigma\sqrt{T-t}\right) \\ & - K \exp[-r(T-t)] \Phi\left(-x_1 - \sigma\sqrt{T-t}\right) \\ & + S_t \exp[-q(T-t)] \Phi(-x_1) \Bigg] \end{aligned} \tag{3.17}$$

with

$$C_r = \frac{\alpha N}{C_p} \tag{3.18}$$

3.2.4 Data Requirements and Calibration

3.2.5 Pricing Example

3.3 Structural Approach

3.3.1 Synthetic Balance Sheet

3.3.2 Data Requirements and Calibration

3.3.3 Pricing Example

Chapter 4

Dynamics and Sensitivity Analysis

4.1 Credit Derivative Approach

4.2 Equity Derivative Approach

4.3 Structural Approach

Chapter 5

Empirical Analysis and Model Comparison

5.1 Data Description

5.1.1 Deutsche Bank

5.2 Model Parametrization

5.3 Model Comparison

5.3.1 Qualitative Analysis

5.3.2 Quantitative Analysis

Chapter 6

Conclusion

Appendix A

Sample Title

Appendix B

Code - Credit Derivative Approach

The following source code is an implementation of the Credit Derivative Approach written in R.

```r
# Price of Contingent Convertible Bond
price_coco_cd <- function(t, T, S_t, S_star, C_p, c_i, r, N, q, sigma){

  spread_coco <- calc_spread_coco(t, T, S_t, S_star, C_p, r, q, sigma)
  V_t_coco <- N * exp(-(r + spread_coco) * (T - t))

  for (t in 1:T){
    V_t_coco <- V_t_coco + c_i * exp(-(r + spread_coco) * t)
  }
  V_t_coco
}

# Calculation of Trigger Probability
calc_p_star <- function(t, T, S_t, S_star, r, q, sigma){
  p_star <- pnorm((log(S_star / S_t) - calc_mu(r, q, sigma) * (T - t)) / (sigma * sqrt(T - t))) + (S_star / S_t)^(2 * calc_mu(r, q, sigma) / sigma^2) * pnorm((log(S_star / S_t) + calc_mu(r, q, sigma) * (T - t)) / (sigma * sqrt(T - t)))
  p_star
}

# Calculation of Drift of Underlying
calc_mu <- function(r, q, sigma){
  mu <- r - q - sigma^2 / 2
  mu
}

# Spread of CoCo Bond
calc_spread_coco <- function(t, T, S_t, S_star, C_p, r, q, sigma){
  spread_coco <- - log(1 - calc_p_star(t, T, S_t, S_star, r, q, sigma)) / (T - t) * (1 - S_star / C_p)
  spread_coco
}

# Pricing Example
```

```
price_coco_cd(t <- 0, T <- 5, S_t <- 40, S_star <- 20, C_p <- 25, c_i <-
    7, r <- 0.03, N <- 100, q <- 0, sigma <- 0.3)
```

Appendix C

Code - Equity Derivative Approach

The following source code is an implementation of the Equity Derivative Approach written in R.

```
# Price of Contingent Convertible Bond
price_coco_ed <- function(t, T, S_t, S_star, C_p, c_i, r, N, q, sigma, alpha){
  V_t_ed <- price_cb(t, T, c_i, r, N) - price_dibi(t, T, S_t, S_star, c_i, r, q, sigma, alpha) + price_difwd(t, T, S_t, S_star, C_p, r, N, q, sigma, alpha)

  return(V_t_ed)
}

# Price of Corporate Bond
price_cb <- function(t, T, c_i, r, N){
  V_t_cb <- N * exp(-r * (T - t))

  for (t in 1:T){
  V_t_cb <- V_t_cb + c_i * exp(-r * t)
  }

  return(V_t_cb)
}

# Price of Binary Option
price_dibi <- function(t, T, S_t, S_star, c_i, r, q, sigma, alpha){
  V_t_dibi <- 0

  i <- t
  k <- T

  for (i in 1:k) {
  V_t_dibi <- V_t_dibi + c_i * exp(- r * i) * (pnorm(- calc_x_1_i(S_t, S_star, sigma, r, q, i) + sigma * sqrt(i)) + (S_star / S_t)^(2 * calc_lambda(r, q, sigma) - 2) * pnorm ( calc_y_1_i(S_t, S_star, sigma, r, q, i) - sigma * sqrt(i)))
  }

```

```r
  V_t_dibi <- alpha * V_t_dibi

  return(V_t_dibi)
}

# Price of Down-And-In Forward
price_difwd <- function(t, T, S_t, S_star, C_p, r, N, q, sigma, alpha){
  V_t_difwd <- calc_conversion_rate(C_p, N, alpha) * (S_t * exp(- q * (T - t)) * (S_star / S_t) ^ (2 * calc_lambda(r, q, sigma)) * pnorm(calc_y_1(t, T, S_t, S_star, r, q, sigma)) - C_p * exp(- r * (T - t)) * (S_star / S_t)^(2 * calc_lambda(r, q, sigma) - 2) * pnorm(calc_y_1(t, T, S_t, S_star, r, q, sigma) - sigma * sqrt(T - t)) - C_p * exp(- r * (T - t)) * pnorm(- calc_x_1(t, T, S_t, S_star, r, q, sigma) + sigma * sqrt(T - t)) + S_t * exp(- q * (T - t)) * pnorm(- calc_x_1(t, T, S_t, S_star, r, q, sigma)))

  return(V_t_difwd)
}

# Calculation of Conversion Rate
calc_conversion_rate <- function(C_p, N, alpha){
  C_r <- alpha * N / C_p

  return(C_r)
}

# Calculation of additional Parameters
calc_x_1_i <- function(S_t, S_star, sigma, r, q, t_i){
  x_1_i <- log(S_t / S_star) / (sigma * sqrt(t_i)) + calc_lambda(r, q, sigma) * sigma * sqrt(t_i)

  return(x_1_i)
}

calc_y_1_i <- function(S_t, S_star, sigma, r, q, t_i){
  y_1_i <- log(S_star / S_t) / (sigma * sqrt(t_i)) + calc_lambda(r, q, sigma) * sigma * sqrt(t_i)

  return(y_1_i)
}

calc_lambda <- function(r, q, sigma){
  lambda <- (r - q + sigma^2 / 2) / sigma^2

  return(lambda)
}

calc_x_1 <- function(t, T, S_t, S_star, r, q, sigma){
  x_1 <- log(S_t / S_star) / (sigma * sqrt(T - t)) + calc_lambda(r, q, sigma) * sigma * sqrt(T - t)

  return(x_1)
}

```

```
calc_y_1 <- function(t, T, S_t, S_star, r, q, sigma){
  y_1 <- log(S_star / S_t) / (sigma * sqrt(T - t)) + calc_lambda(r, q,
    sigma) * sigma * sqrt(T - t)

  return(y_1)
}

# Pricing Example
price_coco_ed(t <- 0, T <- 5, S_t <- 40, S_star <- 20, C_p <- 25, c_i <-
    7, r <- 0.03, N <- 100, q <- 0, sigma <- 0.3, alpha <- 1)
```

Appendix D

Code - Structural Approach

The following source code is an implementation of the Structural Approach written in R.

```
# Price of Contingent Convertible Bond
price_coco_sa <- function(T , npath , rho , kappa , r_bar, r0, sigma_r,
    mu_Y, sigma_Y, lambda, g, x_hat, b0, p, e_bar, sigma_x, x0_low, x0_
    high, x0_nint, B, c_low, c_high, c_nint){
  n <- T * 250
  dt <- T / n

  result <- sim_corrProcess(T, npath, rho, n, dt)
  dW_1 <- result$dW_1
  dW_2corr <- result$dW_2corr

  r <- sim_interestrate(kappa, r_bar, r0, sigma_r, dW_2corr, n, npath,
    dt)

  V_t_sa <- get_price(npath, n, dt, dW_1, dW_2corr, r, mu_Y, sigma_Y,
    lambda, g, x_hat, b0, p, e_bar, sigma_x, x0_low, x0_high, x0_nint, B
    , c_low, c_high, c_nint) * 100
  return(V_t_sa)
}

sim_corrProcess <- function(T, npath, rho, n, dt){
  vect <- c(1, rho, rho, 1)
  RHO <- matrix(vect, nrow = 2)
  chol_RHO <- t(chol(RHO))

  # Create two Brownian Motions
  dW_1 <- matrix(1, n, npath)
  dW_2 <- matrix(1, n, npath)

  for(j in 1:npath)
  {
    dW_1[ , j] <- rnorm(n) * sqrt(dt)
    dW_2[ , j] <- rnorm(n) * sqrt(dt)
  }

```

```r
  # Create Correlated Process based on Brownian Motions using Cholesky-
    Decomposition
  dW_2corr <- matrix(1, n, npath)
  for(j in 1:npath)
  {
    for(i in 1:n)
    {
      dW_2corr[i, j] <- dW_1[i, j] * chol_RHO[2, 1] + dW_2[i, j] * chol_
    RHO[2, 2]
    }
  }

  return(list("dW_1" = dW_1, "dW_2corr" = dW_2corr))
}

# Create Interest Rate Process
sim_interestrate <- function(kappa, r_bar, r0, sigma_r, dW_2corr, n,
    npath, dt){
  r <- matrix(r0, n + 1, npath)

  for(j in 1:npath)
  {
    for(i in 1:n)
    {
      r[i + 1, j] <- r[i, j] + kappa * (r_bar - r[i, j]) * dt + sigma_r
    * sqrt(r[i, j]) * dW_2corr[i, j]
    }
  }

  return(r)
}

get_price <- function(npath, n, dt, dW_1, dW_2corr, r, mu_Y, sigma_Y,
    lambda, g, x_hat, b0, p, e_bar, sigma_x, x0_low, x0_high, x0_nint, B
    , c_low, c_high, c_nint){

  c_fit_matrix <- matrix(0, x0_nint, length(lambda))

  for(w in 1:length(lambda))
  {
    # Create parametres for jump process
    phi <- matrix(rbinom( n%*%npath, 1, dt * lambda[w]), n, npath)
    ln_Y <- matrix(rnorm(n%*%npath, mu_Y, sigma_Y), n, npath)

    b <- matrix(b0, n + 1, npath)
    x_bar0 <- 1 + e_bar + p * b0
    x_bar <- matrix(x_bar0, n + 1, npath)

    h <- matrix(1, n, npath)

    k <- exp(mu_Y + 0.5 * sigma_Y^2) - 1

    c <- seq(c_low, c_high, length = c_nint)
    x0 <- seq(x0_low, x0_high, length = x0_nint)
```

```
    for(l in 1:x0_nint) # Wieso?
    {
      for(m in 1:c_nint) # Wieso?
      {
        x <- matrix(x0[l],n+1,npath)
        ln_x0 <- matrix(log(x0[l]),n+1,npath)
        ln_x <- ln_x0
        binom_c <- matrix(1,n+1,npath)

        for(j in 1:npath)
        {
          for(i in 1:n)
          {
            d_1 <- (ln_x[i, j] + mu_Y) / sigma_Y
            d_2 <- d_1 + sigma_Y

            h[i, j] <- lambda[w] * (pnorm( - d_1) - exp(ln_x[i, j]) * exp(mu_Y + 0.5 * sigma_Y^2) * pnorm(-d_2))

            b[i + 1, j] <- b[i, j] * exp(- g[w] * (exp(ln_x[i, j]) - x_hat) * dt)

            ln_x[i + 1, j] <- ln_x[i, j] + ( (r[i, j] - lambda[w] * k) - (r[i, j] + h[i, j] + c[m] * b[i, j]) / exp(ln_x[i, j]) - g[w] * (exp(ln_x[i, j]) - x_hat) - 0.5 * sigma_x^2) * dt + sigma_x * sqrt(dt) * dW_1[i, j] + ln_Y[i,j] * phi[i, j]

            x[i + 1, j] <- exp(ln_x[i + 1, j])

            x_bar[i + 1, j] <- 1 + e_bar + p * b[i + 1, j]

            if(x[i + 1, j] >= x_bar[i + 1, j] && binom_c[i, j] > 0.5)
            {
              binom_c[i + 1, j] <- 1
            }else
            {
              binom_c[i + 1, j] <- 0
            }
          }
        }

        payments <- matrix(c(rep(c[m] * dt, n - 1), B), n, npath) * binom_c[1:n, ]

        for(j in 1:npath){
          for(i in 2:n){
            if(payments[i, j] == 0 && p * b[sum(binom_c[ , j]) + 1, j] <= x[sum(binom_c[ , j]) + 1, j] - 1 ){
              payments[i, j] <- p * B
              break
            }
            else if(payments[i, j] == 0 && 0 < x[sum(binom_c[ , j]) + 1, j] - 1 && x[sum(binom_c[ , j]) + 1, j] - 1 < p * b[sum(binom_c[ , j
```

```
]) + 1, j]){
              payments[i, j] <- (x[sum(binom_c[ , j]) + 1, j] - 1) * B / b[sum(binom_c[ , j]) + 1, j]
              break
            }
            else{
              payments[i, j] <- payments[i, j]
            }
          }
        }
        vec_disc_v <- rep(0, npath)
        for(j in 1:npath)
        {
          disc_v <- 0
          int_r <- 0

          for(i in 1:n)
          {
            int_r <- int_r + r[i, j] * dt
            disc_v <- disc_v + exp(- int_r) * payments[i, j]
          }
          vec_disc_v[j] <- disc_v
        }

        V_t_sa <- mean(vec_disc_v)

        return(V_t_sa)
      }
    }
  }
}

# Pricing Example
price_coco_sa(T = 5, npath = 2, rho = - 0.2, kappa = 0.114, r_bar = 0.069, r0 = 0.035, sigma_r = 0.07, mu_Y = -0.01, sigma_Y = 0.02, lambda = c(1), g = c(0.5), x_hat = 1.1, b0 = 0.04, p = 1, e_bar = 0.02, sigma_x = 0.02, x0_low = 1.15, x0_high = 1.15, x0_nint = 10, B = 1, c_low = 0.05, c_high = 0.05, c_nint = 10)
```

References

Per Alvemar and Philip Ericson. Modelling and pricing contingent convertibles. *University of Gothenburg*, 2012.

Jan De Spiegeleer and Wim Schoutens. Pricing contingent convertibles: a derivatives approach. *Available at SSRN 1795092*, 2011.

Marc Erismann. *Pricing Contingent Convertible Capital-A Theoretical and Empirical Analysis of Selected Pricing Models*. PhD thesis, University of St. Gallen, 2015.

George Pennacchi. A structural model of contingent bank capital. Working Paper 1004, Federal Reserve Bank of Cleveland, 2010. URL `https://ideas.repec.org/p/fip/fedcwp/1004.html`.