CQF Module 5 Exercise Solution

Ran Zhao

1 a To compute the firm's asset value and volatility, set up the Merton type structural model as

$$
\begin{aligned}
E_0 &= V_0 N(d_1) - D\exp(-rT)N(d_2) \\
d_1 &= \frac{1}{\sigma_V}\left[\log\left(\frac{V_0}{D}\right) + \left(r + \frac{1}{2}\sigma_V^2\right)T\right] \\
d_2 &= d_1 - \sigma_V\sqrt{T} \\
\sigma_E &= \sigma_V N(d_1)\frac{V_0}{E_0}
\end{aligned}
$$

To solve the simultaneous equations numerically, I use MATLAB to find the minimum of the penalty function, where the deviations of E_0 and σ_E between what are given in the context and computed results are calculated. The optimization results yield

$$
\begin{cases}
V_0 = & 7.9088 \\
\sigma_V = & 19.12\%
\end{cases}
$$

Substitute the solutions into the simultaneous equations above, we yield back the equity value and equity volatility. The codes solving the equations are provided in the Appendix.

1 b The probability of the default for Merton model is

$$\mathbb{P}[V_t < D] = N(-d_2)$$

whereas in the Black-Cox PD is calculated as

$$\mathbb{P}[\tau \le T|\tau > t] = N(h_1) + \exp\left\{2\left(r - \frac{\sigma_V^2}{2}\right)\log\left(\frac{K}{V_0}\right)\frac{1}{\sigma_V^2}\right\}N(h_2)$$

Using the simultaneous equations in (1a) to solve for V_0 and σ_V, we have the following sensitivity between σ_E and the probability of default.

As shown in Figure 1, the probability of default increases with higher equity volatility. Intuitively speaking, the higher equity volatility makes it more possible for asset value to fall below the debt, and therefore more like to default.

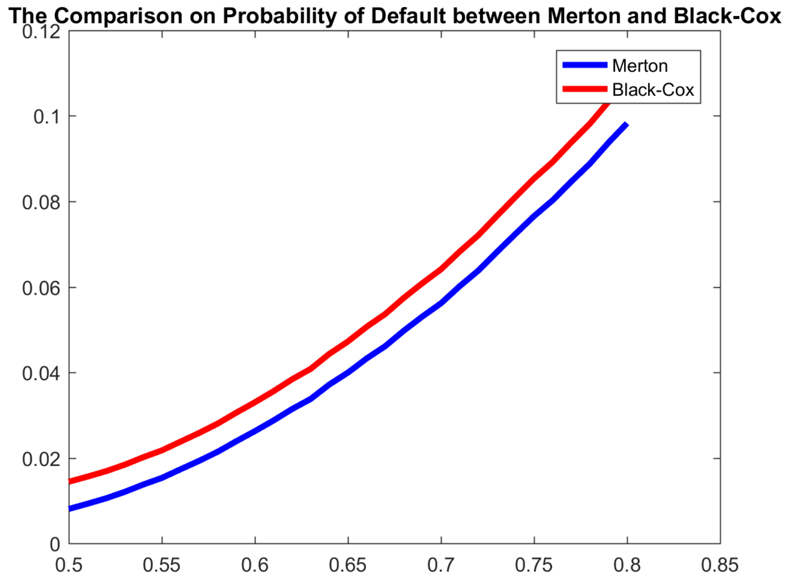

Figure 1: The comparison of probability of default between Merton and Black-Cox Models

The Merton model estimates PD lower than the Black-Cox model, for the reason that the Black-Cox allows for early default than the maturity. Beyond equity volatility as 60%, the probability of default increases monotonically with the equity volatility, and the PD estimated by Black-Cox model is 0.8%.

The codes solving the equations are provided in the Appendix.

2 Before determining the bivariate European binary call option price, we use transformation of Kendall's tau value to solve for the copula parameter α. That is

$$\rho_K = 0.35 = 1 - \frac{4}{\alpha}[D_1(-\alpha) - 1] \qquad D_1(-\alpha) = \frac{1}{\alpha}\int_0^{\alpha} \frac{x}{e^x - 1}dx + \frac{\alpha}{2}$$

Numerically, the α can be solved by equating both sides of the equation in MATLAB. The copula coefficient α is calculated as 0.8777. The code solving the equations is provided in the Appendix.

Alternatively, TSE on the $\frac{x}{e^x-1}$ contributes on solving the equation analytically. That is,

$$\begin{aligned}
\rho_K &= 1-\frac{4}{\alpha}[D_1(-\alpha)-1] \\
&= 1-\frac{4}{\alpha}\left[\frac{1}{\alpha}\int_0^{\alpha}\frac{x}{e^x-1}dx+\frac{\alpha}{2}-1\right] \\
&\cong 1-\frac{4}{\alpha}\left[\frac{1}{\alpha}\int_0^{\alpha}(1-\frac{x}{2}+\frac{x^2}{12})dx+\frac{\alpha}{2}-1\right] \\
&= 1-\frac{4}{\alpha}\left[\frac{1}{\alpha}\ x-\frac{x^2}{4}+\frac{x^3}{36}\Big|_0^{\alpha}+\frac{\alpha}{2}-1\right] \\
&= -\frac{\alpha}{9} \\
\alpha &= -3.15
\end{aligned}$$

The analytical solution is very different with the exact value. Here we use the α calculation from MATLAB. Then, regarding on the two underlyings, we have

$$\begin{cases} u_1 = & Pr^{\mathbb{Q}}(S_1,T>K)=1-N(-d_2^1)=0.0718 \\ u_1 = & Pr^{\mathbb{Q}}(S_1,T>K)=1-N(-d_2^2)=0.3362 \end{cases}$$

Finally, the bivariate European binary call option price is

$$\begin{aligned}
B(S_1,S_2,t) &= e^{-r(T-t)}C(u_1,u_2) \\
&= e^{-r(T-t)}\frac{1}{\alpha}\log\left[1+\frac{\Pi_{i=1}^2(e^{\alpha u_i}-1)}{(e^{\alpha}-1)^{n-1}}\right] \\
&= \frac{1}{\alpha}\log\left[1+\frac{(e^{\alpha u_1}-1)(e^{\alpha u_2}-1)}{(e^{\alpha}-1)^{n-1}}\right] \\
&= 0.0284
\end{aligned}$$

Credit Curve 1 The CDS with accruals has premium leg and default leg as

$$\begin{cases} PL_N = & S_N\sum_{n=1}^{N}D(0,T_n)P(T_n)(\Delta t_n)+S_N\sum_{n=1}^{N}D(0,T_n)[P(T_{n-1})-P(T_n)]\frac{(\Delta t_n)}{2} \\ DL_N = & (1-R)\sum_{n=1}^{N}D(0,T_n)[P(T_{n-1})-P(T_n)] \end{cases}$$

For every $\Delta t=0.25$, I use log-linear interpolation for the discount factor and linear interpolation for the hazard rate provided in Table 1. Equalizing the premium leg and the default at each projection period yields the CDS spread at each time. The construction of the Solver and formulas is shown in the attached Excel spreadsheet (tab '1 price CDS'). The credit spreads in each period are shown below (in basis points).

T	Credit Sprad (bps)
0.25	44.7750
0.5	37.2899
0.75	29.7866
1	37.1648
1.25	51.0404
1.5	57.7789
1.75	60.3492
2	68.1458
2.25	76.4806
2.5	82.5226
2.75	86.8723
3	92.1867

Credit Curve 2 The bootstrapping of the survival probabilities requires the equalization on PL and DL on each period. For the first period we have

$$P(T_1) = \frac{L}{L + \Delta t_1 S_1}$$

where $L = (1 - R)$ using the fact that $P(T_0) = 1$. When $N \geq 2$, we have

$$P(T_N) = \frac{\sum_{n=1}^{N-1} D(0, T_n)[LP(T_{n-1}) - (L + \Delta t_n S_N)P(T_n)]}{D(0, T_N)(L + \Delta t_n S_N)} + \frac{P(T_{N-1})L}{L + \Delta t_N S_N}$$

The implementation is on '2 bootstrap PrSurv' tab of the attached spreadsheet. And the bootstrapped survival probabilities are shown below.

T	DF	$P(T_i)$
0	1.0000	1.0000
1	0.9920	0.9769
2	0.9841	0.9469
3	0.9763	0.9160
4	0.9685	0.8826
5	0.9608	0.8503

Credit Curve 3 The hazard rates are in turn calculated as

$$P(0, T) = \exp\left(-\sum_{t=1}^{T} \lambda_t \Delta t\right) \qquad \lambda_m = \frac{1}{\Delta t} \log \frac{P(0, t_m)}{P(o, t_{m-1})}$$

and we yield

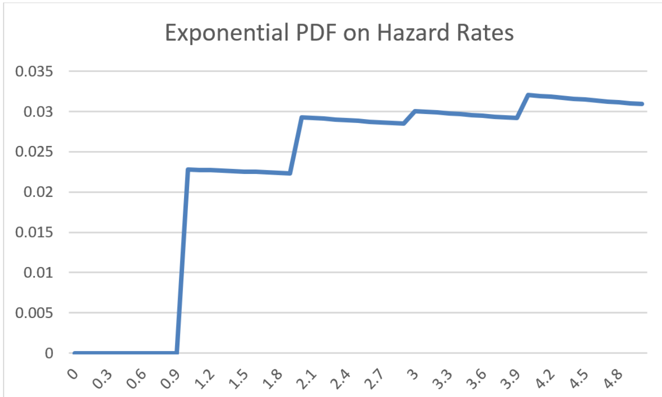

Figure 2: The exponential pdf of hazard rate

T	$P(T_i)$	Hazard Rate
0	1.0000	0.0000
1	0.9769	0.0234
2	0.9469	0.0312
3	0.9160	0.0332
4	0.8826	0.0372
5	0.8503	0.0373

The exponential pdf $f(t) = \lambda e^{-\lambda t}$ is plotted in Figure 2.

From the graph, the density of the hazard rate is very instable due to the piecewise constant assumption on λ. Especially on the year-change, the density jumps up. However, a linear interpolation on the hazard rate will yield a much smoothier exponential density.

Appendix

```
function diff_mse = compute_E0(V0, sigmaV)
%COMPUTE the difference of equity value
% between calculated initial equity value and 3M
%
%INPUTS
%  V0:      the initial asset value
%  sigmaV:  the volaility of assets
%  inputM:  include r, T, D
%
%OUTPUT
%  diff_mse: calculated mse using INPUTS - context

r = 0.02;
D = 5;
T = 1;
```

```
d_1 = (1/(sigmaV*sqrt(T))) * ...
    ( log(V0/D) + (r+0.5*sigmaV^2)*T );
d_2 = d_1 - sigmaV*sqrt(T);

E0 = V0*normcdf(d_1,0,1) - D*exp(-r*T)*normcdf(d_2,0,1);
sigmaE = sigmaV*normcdf(d_1,0,1)*V0/E0;

diff_mse = 10*(E0-3)^3 + (sigmaE-0.5)^2;

end
```

compute_E0.m

```
% solve V0 and sigma_V
[results,fval] = fsolve(@(x) compute_E0(x(1),x(2)),[9;0.25], ...
    optimoptions('fmincon','MaxFunEvals',10000,'MaxIter',10000));
% check answer
compute_E0(results(1), results(2))
```

compute_value_vol_1a.m

```
function diff_mse = solve_asset_vol(V0, sigmaV, givenSigmaE)

r = 0.02;
D = 5;
T = 1;

d_1 = (1/(sigmaV*sqrt(T))) * ...
    ( log(V0/D) + (r+0.5*sigmaV^2)*T );
d_2 = d_1 - sigmaV*sqrt(T);

E0 = V0*normcdf(d_1,0,1) - D*exp(-r*T)*normcdf(d_2,0,1);
sigmaE = sigmaV*normcdf(d_1,0,1)*V0/E0;

diff_mse = 10*(E0-3)^3 + (sigmaE-givenSigmaE)^2;

end
```

solve_asset_vol.m

```
r = 0.02;
D = 5;
T = 1;
K = 5;

sigmaE = 0.50:0.01:0.80;
Merton_PD = zeros(1, length(sigmaE));
BC_PD = zeros(1, length(sigmaE));

for i = 1:length(sigmaE)
    [results,~] = fsolve(@(x) solve_asset_vol(x(1), x(2), sigmaE(i))
        ,[9;0.25], ...
        optimoptions('fmincon','MaxFunEvals',10000,'MaxIter',10000));

    % probability of default Merton
    Merton_PD(i) = normcdf(-((1/(results(2)*sqrt(T))) * ...
        ( log(results(1)/D) + (r+0.5*results(2)^2)*T ) - results(2)*sqrt(T))
            ,0,1);

    % probability of default Black-Cox
```

```
    h_1 = (log(K/(exp(r*T)*results(1))) + (results(2)^2*T/2))/(results(2)*T);
    h_2 = h_1 - results(2)*sqrt(T);
    BC_PD(i) = normcdf(h_1,0,1) + exp(2*(r-results(2)^2/2) ...
        *log(K/results(2))/results(2)^2)*normcdf(h_2,0,1);
end

avg_diff_PD = mean(BC_PD(sigmaE >= 0.60) - Merton_PD(sigmaE >= 0.60));

plot(sigmaE, Merton_PD, '-b', 'LineWidth', 3);
hold on
plot(sigmaE, BC_PD, '-r', 'LineWidth', 3);
hold off
legend('Merton','Black-Cox')
title('The Comparison on Probability of Default between Merton and Black-Cox')
```

compare_PD_1b.m

```
% solve alpha
alpha = fsolve(@(x) correlation_int(x)-0.35,0.85, ...
        optimoptions('fmincon','MaxFunEvals',10000,'MaxIter',10000));
% alpha = 0.8777
correlation_int(0.8777)

% call option price
T = 0.5; K = 120; r = 0; S1 = 90; S2 = 110; sigma1 = 0.3; sigma2 = 0.5;
d2_1 = 1/(sigma1*sqrt(T))*(log(S1/K)+(r-sigma1^2/2)*T);
d2_2 = 1/(sigma2*sqrt(T))*(log(S2/K)+(r-sigma2^2/2)*T);

u1 = 1 - normcdf(-d2_1);
u2 = 1 - normcdf(-d2_2);

B = 1/alpha*log(1+((exp(alpha*u1)-1)*(exp(alpha*u2)-1))/(exp(alpha-1)));
```

solve_alpha.m

```
function result = correlation_int(alpha)

    fun = @(x) x./(exp(x)-1);
    result = 1-4/alpha*(integral(fun,0,alpha)+alpha/2-1);

end
```

correlation_int.m