Bilateral CVA with Mathematica 10

CVA measure, as explained in the previous demonstration, is not the final word in the valuation adjustment methodology currently applied in the market. A closely related metric is the Bilateral CVA – also known as **BCVA**. In practice, this is the CVA extension in the dealer (financial institution) direction where creditworthiness of the bank is taken into consideration. BCVA is therefore the sum of CVA and DVA, which stands for debit valuation adjustment and reflect the mirror profile of the CVA. It is worth noting that BCVA is still a controversial measure – if the bank's credit worthiness deteriorates, it earns money as it reduces its CVA. Mathematically BCVA is complex: $BCVA(u) = \int_t^T (1 - Rc(u))DF(u)EPE(u)dPDc(u)Sb(u)du + \int_t^T \big(1 - Rb(u)\big)\, DF(u)ENE(u)dPDb(u)Sc(u)du$ where R is the recovery rate, EPE is the positive exposure, $dPDc$ is the counterparty conditional default probability, Sb is the Bank's survival probability, ENE is the expected negative exposure. Subscript c indicates counterparty and b the Bank. Since the BCVA is more involved calculation, **Mathematica 10** comes as a handy platform to solve this problem quickly and efficiently. We will use Monte Carlo method to solve this problem.

BCVA formula is more complicated – we have two recovery rates (the counterparty and the bank), two conditional default probabilities and two exposure profiles – positive for CVA and negative for DVA. There is also a new term in the formula – survival probability$S$. Why? Because CVA will be chargeable only if the Bank is still functioning and the similar reasoning applies to DVA – counterparty must be alive if the DVA is to be applied.

BCVA calculation can be really complex – we have two recovery variables and three processes (swap, counterparty hazard rate and bank's hazard rate). They can be correlated. We present the following , complex case:

(i) Recovery rates – make them stochastic (driven by beta process) and correlated to each other but independent from swap and hazard rates

(ii) Swap rate process and hazard rate processes for the counterparty and the bank are correlated

We repeat there the data from the CVA case : the instrument is a 5Y S/A IR swap with 2.5% fixed rate. Counterparty 5Y CDS rate trades at 1.25% and the Bank CDS stands at 1%.

1) Swap rate = GBM process with $\mu = 0.5\%$,$\sigma = 20\%$
2) CP CDS = CIR process $\vartheta = 0.5\%$,$\beta = 1.5\%$,$\sigma 2 = 5\%$
3) BK CDS = CIR process $\kappa = 3.2\%$,$\alpha = 1.2\%$,$\sigma 3 = 4\%$
4) Correlation matrix (Exp-CP-BK):

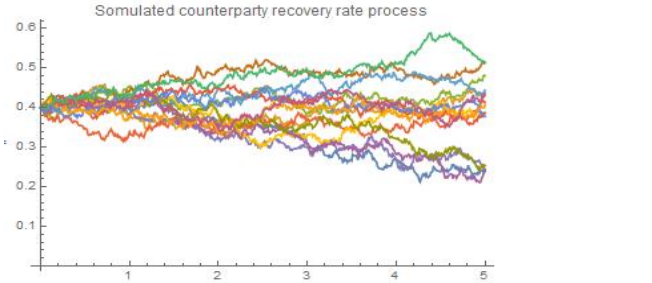
$$\begin{matrix} 1 & 0.2 & 0.35 \\ 0.2 & 1 & 0.17 \\ 0.35 & 0.17 & 1 \end{matrix}$$

Define two recovery rate processes:

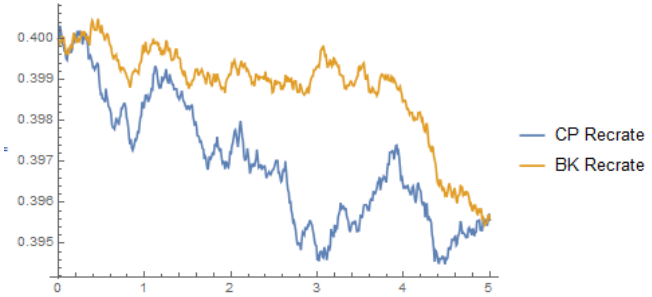
```
cW[ρ_] := Refine[ItoProcess[{{0, 0}, IdentityMatrix[2]}, {{w1, w2}, {0, 0}},
    t, {{1, ρ}, {ρ, 1}}], -1 < ρ < 1];

rec2proc = ItoProcess[{dx[t] == -λ (μ - x[t]) dt + η * Sqrt[1 - x[t]] dw1[t],
    dy[t] == -ν (π - y[t]) dt + ξ * Sqrt[1 - y[t]] dw2[t]},
   {x[t], y[t]}, {{x, y}, {x0, y0}}, t, {w1, w2} ≈ cW[ρ]];

rec2sim =
  RandomFunction[
   rec2proc /. {λ → -0.015, μ → 0.35, η → 0.055, x0 → 0.4, ν → 0.01, π → 0.44,
     ξ → 0.04, y0 → 0.4, ρ -> 0.27}, {0, z, dt}, 1000, Method → "StochasticRungeKutta"] ;
```

This is the simulated CP RR example:

```
ListLinePlot[cprec["Path", Range[15]], PlotRange → All,
 PlotLabel → "Somulated counterparty recovery rate process"]
```



And these are the expected RR values – for CP and BK:



Now define three correlated processes (swap and two CDS):

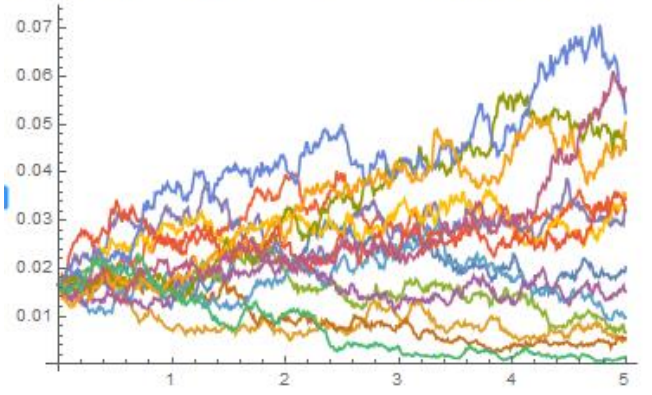
```
cW[n_] := ItoProcess[{{0, 0, 0}, IdentityMatrix[n]}, {{n1, n2, n3}, {0, 0, 0}}, t, sig]

triproc =
  ItoProcess[{ds[t] == μ s[t] dt + σ1 * s[t] dn1[t],
    dc[t] == σ (β - c[t]) dt + σ2 * Sqrt[Abs[c[t]]] dn2[t],
    dr[t] == κ (α - r[t]) dt + σ3 * Sqrt[Abs[r[t]]] dn3[t]
   }, {s[t], c[t], r[t]}, {{s, c, r}, {s0, c0, r0}}, t, {n1, n2, n3} ≈ cW[3]];
```

Break them down into components and get HR:

```
swapproc = simtriproc["PathComponent", 1];
cpproc = simtriproc["PathComponent", 2];
ourproc = simtriproc["PathComponent", 3];
cphazrate = TimeSeriesMap[#/(1 - mmcrec) &, cpproc];
ourhzrate = TimeSeriesMap[#/(1 - mmbrec) &, ourproc];
```

this is the Bank's hazard rate evolution:

```
ListLinePlot[ourhzrate["Path", Range[15]]]
```

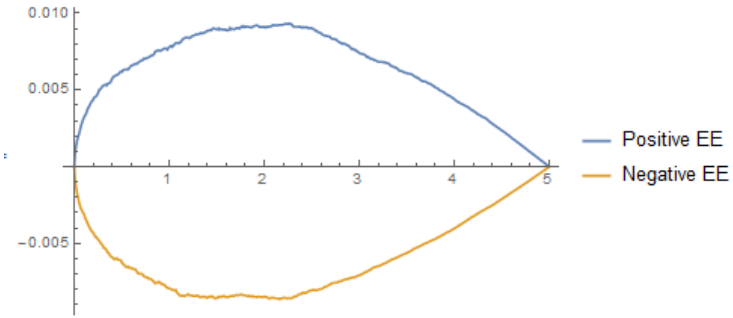


Define (i) positive and (ii) negative swap exposure functions:

```
PExposure[T_, S_, t_, v_] := (T - t) * Max[v - S, 0];
NExposure[T_, S_, t_, v_] := (T - t) * Min[v - S, 0];

pospath = TimeSeriesMapThread[PExposure[5, 0.025, #1, #2] &, swapproc];
negpath = TimeSeriesMapThread[NExposure[5, 0.025, #1, #2] &, swapproc];

eepos = TimeSeriesThread[Mean, pospath];
eeneg = TimeSeriesThread[Mean, negpath];
```

And display both EPE and ENE profile:

```
ListLinePlot[{eepos, eeneg},
 PlotLegends → {"Positive EE", "Negative EE"}]
```

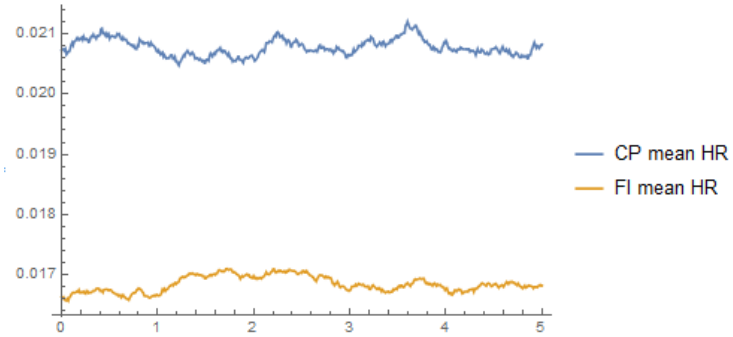


Both exposures are reasonably symmetric.

In a similar way, we define the mean hazard rate – both for the counterparty and the bank.

```
meancphaz = TimeSeriesThread[Mean, cphazrate];
meanourhaz = TimeSeriesThread[Mean, ourhzrate];
```

The two mean hazard rates look as follows:



The counterparty expected hazard rate exceeds the bank hazard rate by approx. 0.40%

The last component we need to define is the survival function: $S(u) = Exp[-\int h(u)du]$

```
survfunc[x_, i_] :=
 Exp[-If[i == "CP", meancphaz["PathFunction"][x], ourhzrate["PathFunction"][x]] * x]
```

Since the hazard rate is now path-dependent, we use path-wise integral (or sum in discrete case) to get the path-dependent survival function.

Having obtained all critical components, we can define the BCVA function as follows:

```
BCVA[n_, fr_, r_] :=
 Module[{f, g, h, m, s},
  f = TemporalData[Table[{i, survfunc[i, "CP"]}, {i, 0, n, fr}]];
  g = MovingMap[First[#] - Last[#] &, f, {2}];
  h = TemporalData[Table[{i, survfunc[i, "BK"]}, {i, 0, n, fr}]];
  m = MovingMap[First[#] - Last[#] &, h, {2}];
  s =
   Sum[(1 - mcprec["PathFunction"][i]) * DF[i] * eepos["PathFunction"][i] *
     survfunc[i, "BK"] * g["PathFunction"][i], {i, fr, n, fr}] +
   Sum[(1 - mbkrec["PathFunction"][i]) * DF[i] * eeneg["PathFunction"][i] *
     survfunc[i, "CP"] * m["PathFunction"][i], {i, fr, n, fr}];
  s]
```

The unction is a discrete version of the continuous function shown above. With cached vectors it provide instant calculation response:

```
= BCVA[5, 1/2, 0.4] * 10^4
= 0.0540738
```

The BCVA value with provided parameters is negligible = 0.06 bp. This is much lower than the CVA alone and shows what BCVA does – it adjusts CVA for its own credit risk. We can compare components:

```
CVACP[n_, fr_, r_] :=
 Module[{f, g, s}, f = TemporalData[Table[{i, survfunc[i, "CP"]}, {i, 0, n, fr}]];
  g = MovingMap[First[#] - Last[#] &, f, {2}];
  s = Sum[(1 - mcprec["PathFunction"][i]) * DF[i] * eepos["PathFunction"][i] *
     g["PathFunction"][i], {i, fr, n, fr}];
  s]

CVABK[n_, fr_, r_] :=
 Module[{f, g, s}, f = TemporalData[Table[{i, survfunc[i, "BK"]}, {i, 0, n, fr}]];
  g = MovingMap[First[#] - Last[#] &, f, {2}];
  s = Sum[(1 - mbkrec["PathFunction"][i]) * DF[i] * eeneg["PathFunction"][i] *
     g["PathFunction"][i], {i, fr, n, fr}];
  s]
```

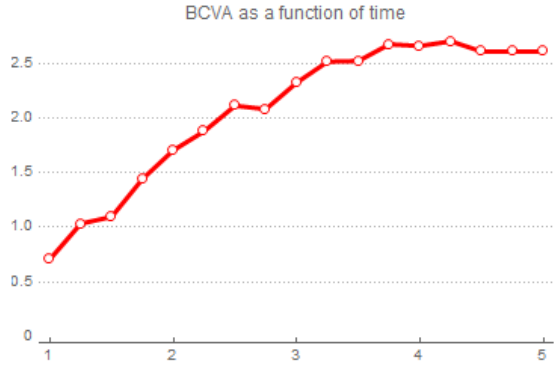
And compute individual CVA and DVA:

```
= CVACP[5, 1/2, 0.4] * 10^4
= 3.54098

= CVABK[5, 1/2, 0.4] * 10^4
= -3.46452
```

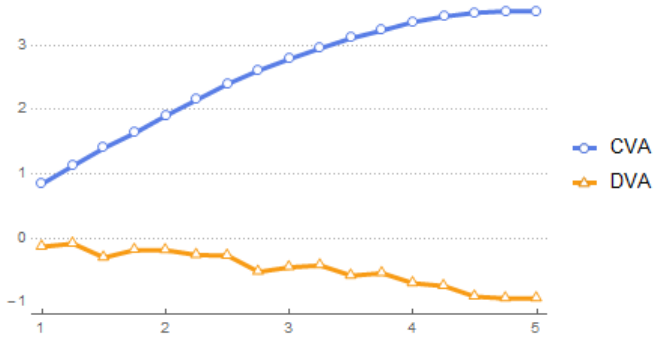
The component values are consistent. Please note that mathematically $BVCA \neq CVA + DVA$ since each component in the BCVA formula is 'weighted' by the survival probability of each other. This term is absent in stand-alone CVA or DVA definition.

We can now visualise the BCVA evolution over time:

```
Table[{i, BCVA[i, 1/4, 0.4] * 10^4}, {i, 1, 5, 1/4}];
ListLinePlot[%, PlotStyle → Red, PlotTheme → "Business",
 PlotLabel → "BCVA as a function of time"]
```



Having defined the component parts, we can also examine how the BCVA is impacted by each part:



The shape of the BCVA in our case is clearly driven by the CVA component, however the flattening of the curve in the long-end comes from the DVA part.

In short: Quick and practical implementation with few functional calls. Well suited approach for testing of modelling assumptions and parameters setting.