Derivatives Credit Risk Management and Measurement

Final Project

Group B

Cristina Primo
Oscar Soppelsa
Bruno Sottile

Bocconi University
Specialized Master in Quantitative Finance and Risk Management

23/06/2020

Contents

1 Introduction

The following report shows a model for the valuation of interest rate swaps (IRS) accounting for counterparty risk. The credit exposure is quantified and priced under three scenarios: uncollateralised IRS, collateralised IRS, and uncollateralised IRS with stochastic volatility. Furthermore, collateralisation as a risk mitigation technique is thoroughly analysed and the concept of Margin Period of Risk is introduced. The main assumptions underlying the analysis are:

- interest rates are modelled under the Hull-White model. It is a no-arbitrage model, able to fit today's term structure;
- a single currency economy is assumed, hence the one-factor Hull-White model is implemented;
- the pricing of interest rate swaps is performed under a single-curve framework;
- constant volatility of interest rates is assumed through Questions 1-7. This assumption is loosened to answer Question 8;
- the Credit Value Adjustment is computed under the hypothesis of absence of wrong-way risk: namely, independence between default, recovery and exposure values is assumed.

All the assignment questions have been addressed.

2 Hull-White model implementation

The Hull-White model for a single currency economy is described by the following stochastic differential equation:

$$\mathrm{d}r(t) = (a\theta(t) - r(t))\mathrm{d}t + \sigma(t)\mathrm{d}W(t)$$

The drift of the process is mean-reverting, moving towards a reversion level, $\theta(t)$, that is time-dependent, with a speed of mean reversion governed by the parameter a. The reversion level reproduces the forward rate curve, $f(0,t)$, and can be written as follows:

$$\theta(t) = \frac{\partial f(0,t)}{\partial t} + af(0,t) + \frac{\sigma^2}{2a}(1 - e^{-2at})$$

The price at time t of a zero coupon bond with maturity T is given by

$$P(t,T) = A(t,T)e^{-B(t,T)r(t)}$$

where

$$B(t,T) = \frac{1}{a}\left(1 - e^{-2at}\right)$$

$$A(t,T) = \frac{P(0,T)}{P(0,t)}\exp\left(B(t,T)f(0,t) - \frac{\sigma^2}{4a}\left(1 - e^{-2at}\right)B(t,T)^2\right)$$

In order to simulate the paths of the short rate, solutions from the Hull-White model are sampled exploiting a discretization scheme that allows to compute the value of the short rate over a discrete time-grid. In particular, the short rate between time step s and t is known beforehand:

$$\mathrm{d}r(t) = r(s)e^{-a(t-s)} + \alpha(t) - \alpha(s)e^{-a(t-s)} + \sigma\int_s^t e^{-a(t-s)}\mathrm{d}W(u)$$

where

$$\alpha(t) = f(0,t) + \left(\frac{\sigma^2}{2a}\left(1 - e^{-2at}\right)\right)^2$$

The function `HullWhiteSim` simulates the paths of the short rate by sourcing the function `MakeScenarios`, which has been coded in C++ in order to increase the computation speed. The compilation of `MakeScenarios.cpp` is automatic when sourcing the function `HullWhiteSim`. Moreover, the function `HullWhiteSim` simulates the paths of the stochastic volatility. However, from Question 1 to Question 7 the volatility is assumed to be constant over time.

The C++ source code `MakeScenarios.cpp` is displayed below.

```
#include <Rcpp.h>

using namespace Rcpp;

double alpha_t(double a,
               double sigma,
               double t,
               double f_0_t)
{
```

```
  double var_factor = sigma * sigma / (2 * a) * (1 - exp(-2 * a * t));
  return f_0_t + var_factor * var_factor;
};

double r_t(double r_s,
           double t,
           double s,
           double a,
           double sigma,
           double f_0_t)
{
  NumericVector w = rnorm(1);
  double r = r_s * exp(-a * (t - s)) +
    alpha_t(a, sigma, t, f_0_t) -
    alpha_t(a, sigma, s, f_0_t) * exp(-a * (t - s)) +
    sigma * exp(-a * (t - s)) * as< double >(w) * sqrt(t - s);
  return r;
}

// [[Rcpp::export]]
List MakeScenarios(double a,
                   double lambda,
                   double k,
                   double xi,
                   double dt,
                   const NumericVector& f_0_t,
                   const NumericVector& my_seq,
                   int n_scenarios,
                   const NumericVector& variance_path)
{
  List scenarios;

  for (int scenario = 0; scenario < n_scenarios; ++scenario)
  {
    NumericVector r, v;
    r.push_back(f_0_t[0]);
    v.push_back(k);

    for (int i = 1; i < my_seq.size(); ++i)
    {
      if (all(variance_path == 0).is_true())
      {
        double dv = lambda * (k - v[i - 1]) * dt + xi * sqrt(v[i - 1]) *
          as< double >(rnorm(1)) * sqrt(dt);
        v.push_back(max(NumericVector::create(0.0, v[i - 1] + dv)));
        r.push_back(r_t(r[i - 1], my_seq[i], my_seq[i - 1], a,
                        sqrt(v[i]), f_0_t[i]));
      }
```

```
      else
      {
        r.push_back(r_t(r[i - 1], my_seq[i], my_seq[i - 1], a,
                        sqrt(variance_path[i]), f_0_t[i]));
      }
    }

    if (any(variance_path != 0).is_true())
    {
      v = variance_path;
    }

    List rate_vol = List::create(Named("rate") = r, Named("variance") = v);
    scenarios.push_back(rate_vol);
  }

  return scenarios;
}
```

The R function `HullWHiteSim` sources `MakeScenarios.cpp` and simulates the paths of the short rate.

```
HullWhiteSim = function(curve,
                        n_scenarios = 10000,
                        a = 0.01,
                        lambda = 1,
                        k = 4e-04,
                        xi = 0,
                        variance_path = NULL)
{
  f_0_t = splinefun(curve$tenor, curve$forward_rate)
  my_seq = curve$tenor
  dt = diff(my_seq)[1]

  if (is.null(variance_path))
  {
    variance_path = rep(0, length(my_seq))
  }

  scenarios = MakeScenarios(a, lambda, k, xi, dt, f_0_t(my_seq),
                            my_seq, n_scenarios, variance_path)
  names(scenarios) = paste0('scenario', 1:n_scenarios)
  return( scenarios )
}
```

Once the short rate has been simulated, it is possible to obtain the numeraire of the risk neutral measure Q:

$$E^Q\left[e^{-\int_t^T r(s)ds}\right]$$

This computation is performed in the function `HullWhiteP`.

```
HullWhiteP = function(curve, t, tau, r, a, sigma)
{
  f_0_t = splinefun(curve$tenor, curve$forward_rate)

  B = function(t, tau)
  {
    1 / a * (1 - exp(-a * (tau - t)))
  }

  P_0 = splinefun(curve$tenor, curve$discount_factor)

  A = function(t, tau)
  {
    P_0(tau) / P_0(t) * exp(B(t, tau) * f_0_t(t) -
                              sigma * sigma / (4 * a) * (1 - exp(-2 * a * t))
                              * B(t, tau) ^ 2)
  }

  P = function(t, tau, r)
  {
    A(t, tau) * exp(-B(t, tau) * r)
  }

  return( unname(P(t, tau, r)) )
}
```

3 Hull-White model: short rate simulations

The function `MakeCurves` builds the zero rate curve with a cubic spline that interpolates the given spot rates with a piecewise third-order polynomial. Then, in a single-curve framework, the forward rates are obtained as follows:

$$f(t_1, t_2) = \frac{1}{\tau(t_1, t_2)}\left(\frac{P(0, t_1)}{P(0, t_2)} - 1\right)$$

The output of the function is a data frame with the zero rates curve, the forward rates curve and the discount factors, a data structure that is exploited in the following steps.

```
Make_curves <- function(given_curve, tenor){

  zero_rate = spline(given_curve$tenor, given_curve$zero_rate, xout = tenor)$y
  curve = data.frame(tenor = tenor,
```

```
                      zero_rate = zero_rate)
  curve$discount_factor = exp(-curve$zero_rate * curve$tenor)
  forward_rate = rep(NA, nrow(curve))
  forward_rate[1] = curve$zero_rate[1]
  df = diff(tenor)[1]

  for (i in 2:length(forward_rate))
      {
        forward_rate[i] = 1 / df * (curve$discount_factor[i - 1]
                                    /curve$discount_factor[i] - 1)
      }
  curve$forward_rate = forward_rate
  return(curve)
}
```

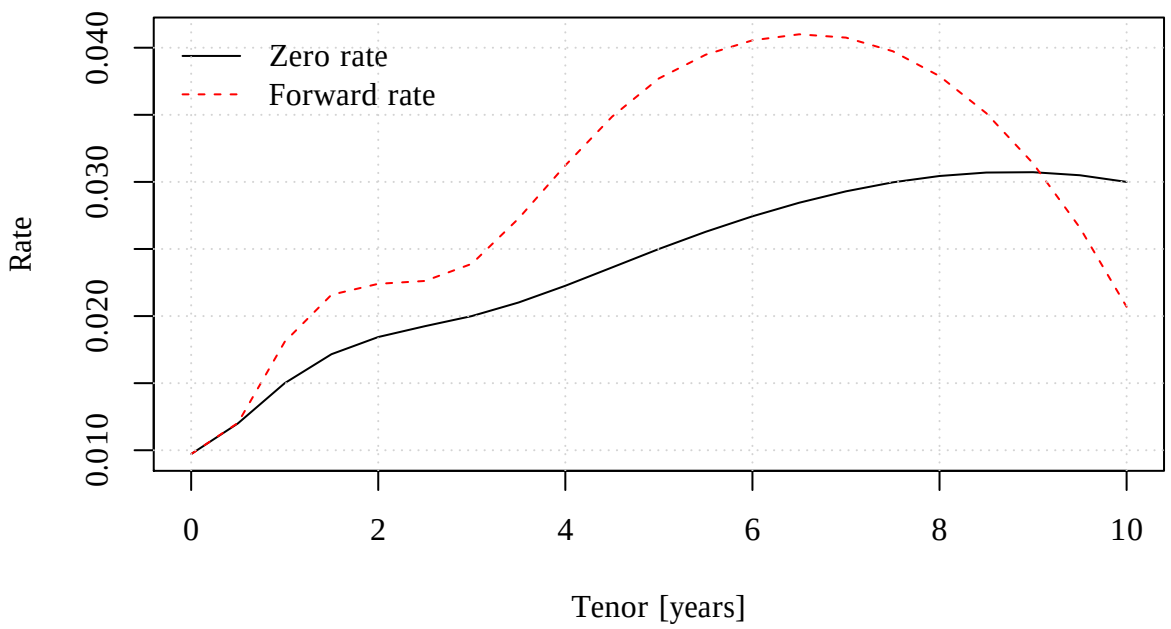


Figure 1: Zero rates and forward rates curves.

The following plot exhibits 10,000 simulated scenarios for the short rate under Hull-White model with these parameters: $a = 0.01$ and $\sigma = 0.02$.

```
paths = HullWhiteSim(curve)
```

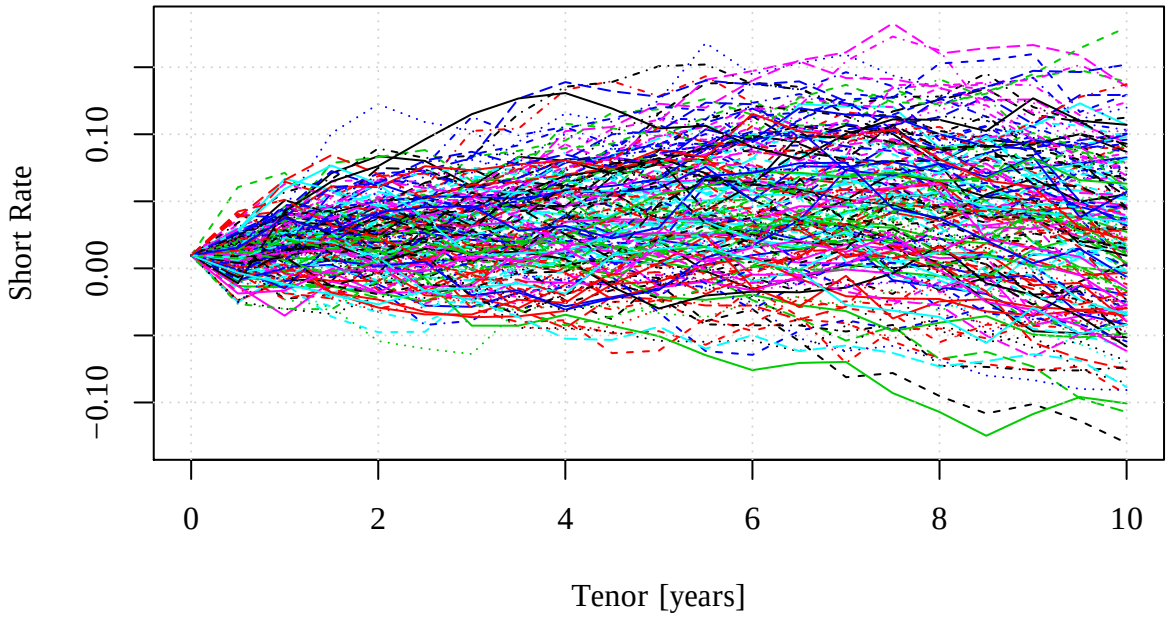


Figure 2: Short rate paths simulated under the Hull-White model.

The Hull-White model is able to reproduce the initial curve observed in $t = 0$. To test whether the model is properly implemented, the simplest strategy is comparing the short rate simulations obtained under low volatility with the current forward rates curve. It is clearly possible to notice that the simulated paths replicate the forward rates curve. By setting the volatility to zero, the two curves would be perfectly overlapping.

```
paths_small_variance <- HullWhiteSim(curve, k = 4e-07)
```

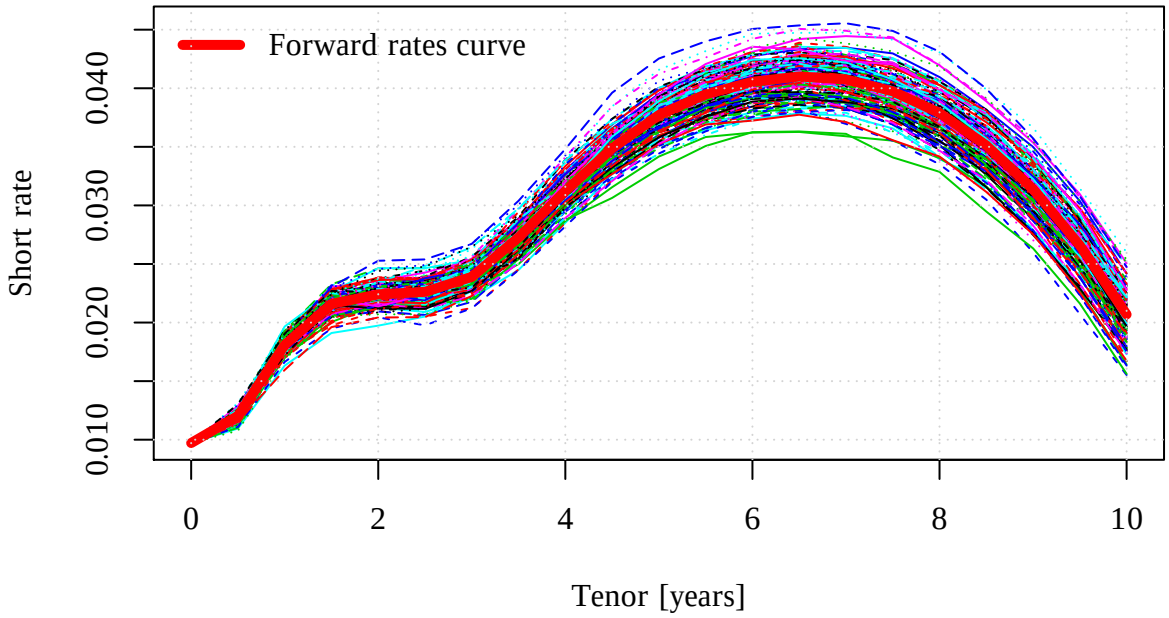


Figure 3: Hull-White short rate simulations with low volatility: comparison with the forward rates curve.

Another possible test to check whether the Hull-White implementation is replicating the forward rates curve consists of comparing the discount curve $P(t,T)$ with the discount factors obtained with the simulations:

$$E^Q\left[e^{-\int_t^T r(s)\mathrm{d}s}\right]$$

The discount factors are obtained with the R function `GetDiscountFactors`.

```
GetDiscountFactors = function(curve,
                              rates,
                              auto_plot = TRUE)
{
  this_path = rates
  dt = diff(curve$tenor)[1]
  stoch_df = matrix(NA, nrow(curve), ncol(rates))
  stoch_df[1, ] = 1 / (1 + dt * this_path[1, ])

  for (i in 2:nrow(this_path))
  {
    stoch_df[i, ] = stoch_df[i - 1, ] / (1 + dt * this_path[i, ])
  }

  if (auto_plot)
  {
    matplot(stoch_df[, 1:100], x = curve$tenor, type = 'l',
```

```
            xlab = 'Tenor [years]',
            ylab = 'Discount factor') ; grid()
    lines(curve$discount_factor ~ curve$tenor, lwd = 5, col = 2)
    legend('bottomleft', legend = 'Discount curve', lwd = 5, col = 2, lty = 1,
           bty = 'n')
  }
}
```

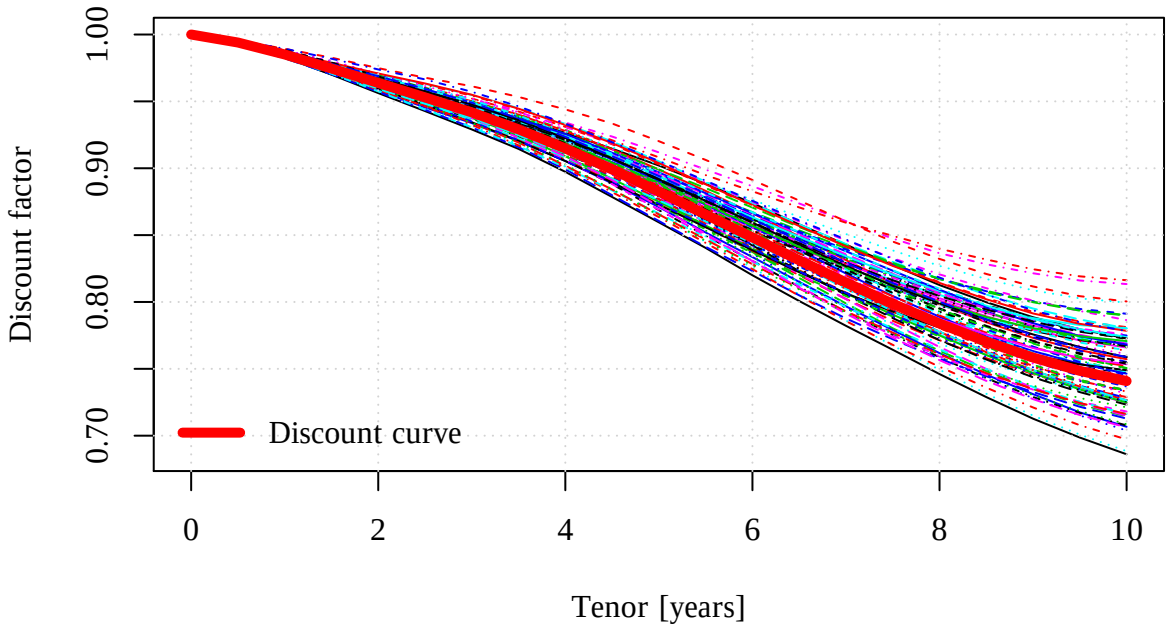


Figure 4: Discount curve simulated under the Hull-White model.

4 Interest rate swap pricing

The function `IRS` prices interest rate swaps. The value of a floating-receiver IRS is obtained by computing the difference between the NPV of the floating leg, L_f, and the NPV of the fixed leg, L_x, of the contract:

$$\text{IRS}(t) = L_f(t) - L_x(t)$$

Hence, the two legs exhibit the following NPVs:

$$L_f(t) = \sum_{i=1}^{n} D(t, T_i) N \tau_i f(t_{i-1}, t_i)$$

$$L_x(t) = \sum_{i=1}^{n} D(t, T_i) N \tau_i K$$

where

- $\tau_i = 0.5$ represents the year fraction between two coupon dates;
- $N = 1$;
- K is the annual coupon of the fixed leg.

Since the coupon dates for the two legs are overlapping, the discounted payoff of the contract can be expressed as:

$$\text{IRS}(t) = \sum_{i=1}^{n} D(t, T_i) N \tau_i (f(t_{i-1}, t_i) - K)$$

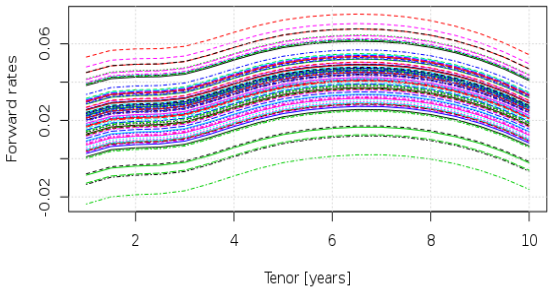
```
IRS = function(evaluation_tenor,
               curve,
               fixed_rate,
               tau,
               rates,
               a,
               variances)
{
  dt = diff(curve$tenor)[1]
  my_curve = subset(curve, tenor > evaluation_tenor)

  if (nrow(my_curve) > 0)
  {

    scenarios = data.frame(volatility = sqrt(variances[
                                          as.character(evaluation_tenor), ]),
                           rate = rates[as.character(evaluation_tenor), ])

    discount_factors = apply(scenarios, 1, function(x){

      x = unlist(x)
      HullWhiteP(curve = curve,
                 t = evaluation_tenor,
                 tau = my_curve$tenor,
                 r = x[2],
                 a = a,
                 sigma = x[1])

    })

    dim(discount_factors) = c(length(my_curve$tenor), ncol(rates))
    discount_factors = t(discount_factors)
    colnames(discount_factors) = paste0('P(', my_curve$tenor, ',', tau, ')')
    rownames(discount_factors) = colnames(rates)

    forward_rates = matrix(NA, nrow = nrow(discount_factors),
```

```
                              ncol = ncol(discount_factors))
    forward_rates[, 1] = 1 / dt * (1 / discount_factors[, 1] - 1)
    if (ncol(discount_factors) > 1)
    {
      for (j in 2:ncol(forward_rates))
      {
        forward_rates[, j] = 1 / dt * (discount_factors[, j - 1] /
                                         discount_factors[, j] - 1)
      }
    }

    if (evaluation_tenor == 0)
    {

      discounted_cf_float = t(t(discount_factors) * my_curve$forward_rate) * dt

    } else {

      discounted_cf_float = t(t(discount_factors) * t(forward_rates)) * dt

    }

    discounted_cf_fixed = discount_factors * fixed_rate * dt
    npv_float = rowSums(discounted_cf_float)
    npv_fixed = rowSums(discounted_cf_fixed)
    npv = npv_float - npv_fixed

  }
  else
  {

    npv = rep(0, ncol(rates))
    names(npv) = colnames(rates)

  }

  return( npv )
}
```

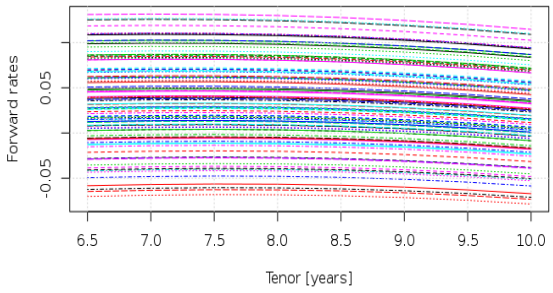
It is possible to check that the inception value of an IRS with the following characteristics is equal to zero:

- 10 years maturity;
- semiannual payment frequency;
- K = 0.02959

The following plots exhibit the value of the forward rates evaluated in 0.5 and 6 years from the evaluation date.



(a) Forward rates at 0.5 years.



(b) Forward rates at 6 years.

Figure 5: Forward rates simulated under Hull-White model.

The NPV of the IRS in $t = 0$ is 0.00178. The value is slightly different from zero owing to the approximation deriving from the zero rates curve interpolation.

5 Mark-to-market of an interest rate swap

The calibration of the Hull-White model is performed on market data in such a way that $\theta(t)$ reproduces today's discount curve, as shown above. The parameter σ must be calibrated on swaptions. Let the calibrated parameters of the Hull-White model are the following:

- $a = 0.01$
- $\sigma = 0.02$

The function `MtMPaths` computes the mark-to-market of the aforementioned IRS on every fixing and coupon date over 10,000 simulated scenarios. In particular, the discount factor is stochastic as it depends on the simulated short rates. Hence, the distribution of the mark-to-market of the contract is obtained on each date.

```
MtM_Paths <- function(evaluation_date, curve, fixed_rate,
                      tau, rates, a, variances){

    MtM = foreach(eval_t = evaluation_date, .combine = rbind.data.frame) %do%
      {
        print(paste0('Evaluating ', eval_t, ' years tenor...'))
        npv = IRS(evaluation_tenor = eval_t, curve, fixed_rate, tau,
                  rates, a, variances)
        print(paste0('>>> Average NPV(', eval_t, ') = ', mean(npv)))
        return( npv )
      }
    colnames(MtM) = colnames(rates)
    rownames(MtM) = evaluation_date
    MtM_t <- t(MtM)
```

```
    return(MtM_t)
}

# Compute the MtM
MtM_paths <- MtM_Paths(evaluation_date = tenor,
                       curve = curve,
                       fixed_rate = 0.02959,
                       tau = 10,
                       rates = rates,
                       a = 0.01,
                       variances = variances)
```

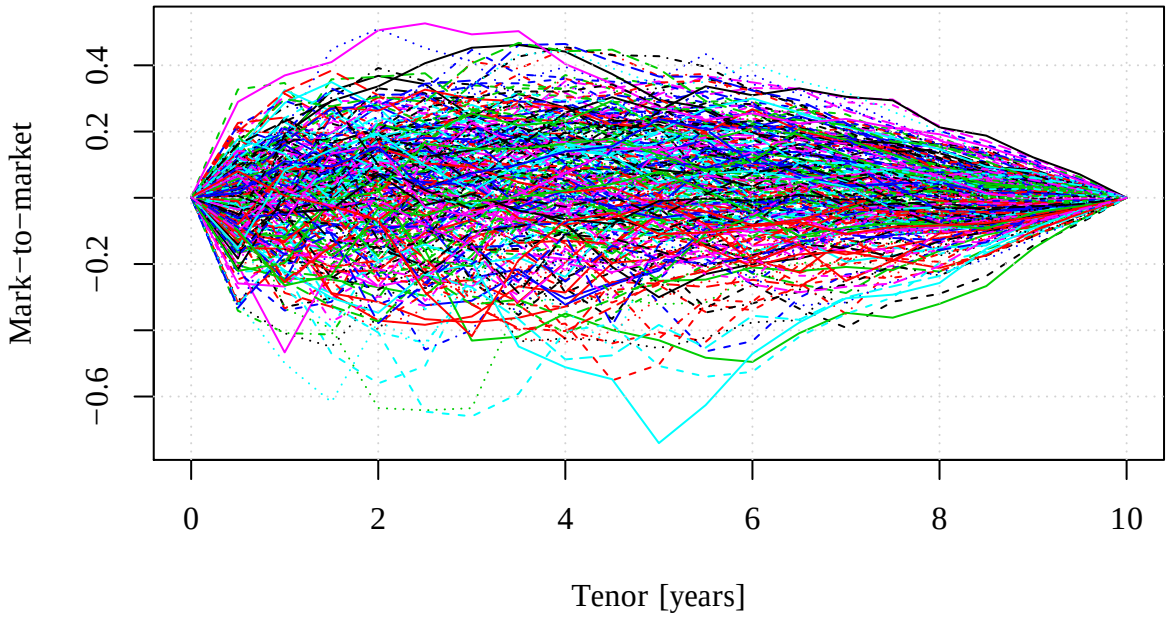


Figure 6: Simulated mark-to-market of an IRS.

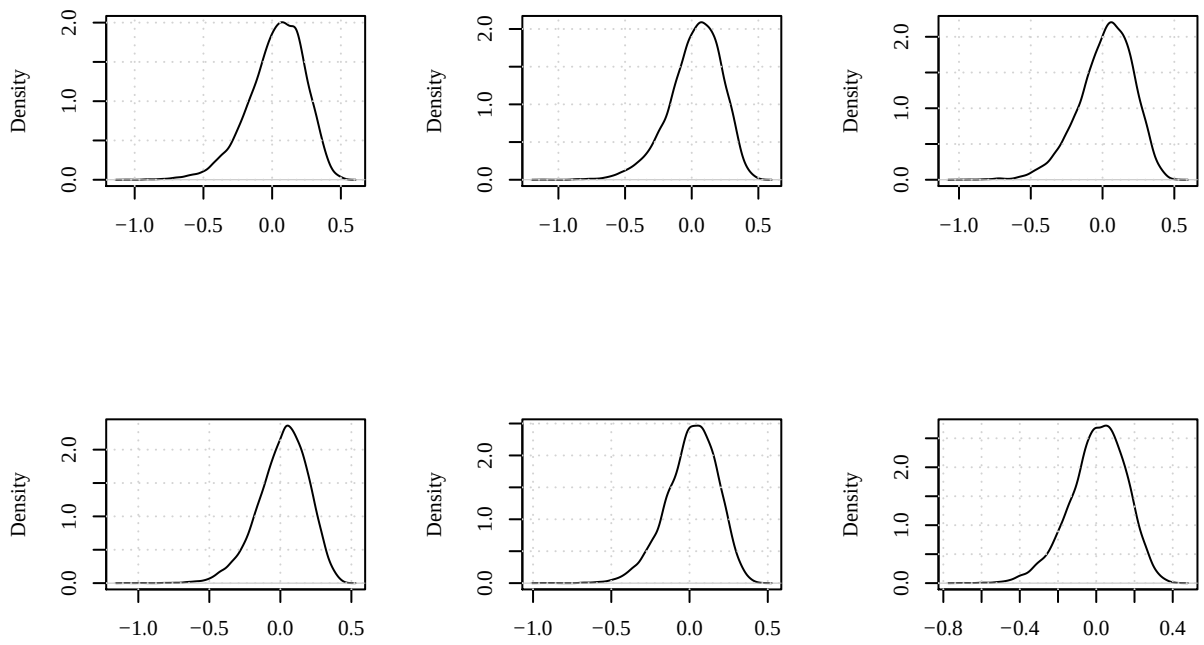


Figure 7: Density of the mark-to-market of an IRS on six different evaluation dates.

6 Expected Exposure and Potential Future Exposure

Once the distribution of the mark-to-market of an IRS has been obtained, it is possible to compute two key counterparty risk measures: Expected Exposure (EE) and Potential Future Exposure (PFE) at 95% confidence level. In counterparty risk, there is a fundamental asymmetry of the potential losses with respect to the mark-to-market. In case of a positive mark-to-market, the amount of the mark-to-market less the recovery rate represents the loss suffered when the counterparty defaults. In this situation, the defaulted counterparty is not able to meet the future commitments and the bank shall pay the positive mark-to-market to enter into an identical contract on the market. On the contrary, in case of negative mark-to-market the bank settles the amount with the defaulted counterparty, but it does not suffer any loss: the position is essentially unchanged.

The risk measures are computed by the function `RiskMeasures`.

```
Risk_Measures <- function(MtM_Paths, conf_level){
    EE <- apply(MtM_Paths, 2, FUN = function(x){
      mean(x[x>0])
    })
    EE[is.nan(EE)] <- 0

    PFE <- apply(MtM_Paths, 2, FUN = function(x){
      quantile(x, conf_level)
    })
```

```
    PFE[PFE<0] <-0

    Risk_Measures <- data.frame(EE, PFE)
    rownames(Risk_Measures) <- colnames(MtM_Paths)
    return(Risk_Measures)
}
```

The following plot depicts the value of EE and PFE for a distribution of 2 years tenor mark-to-market.

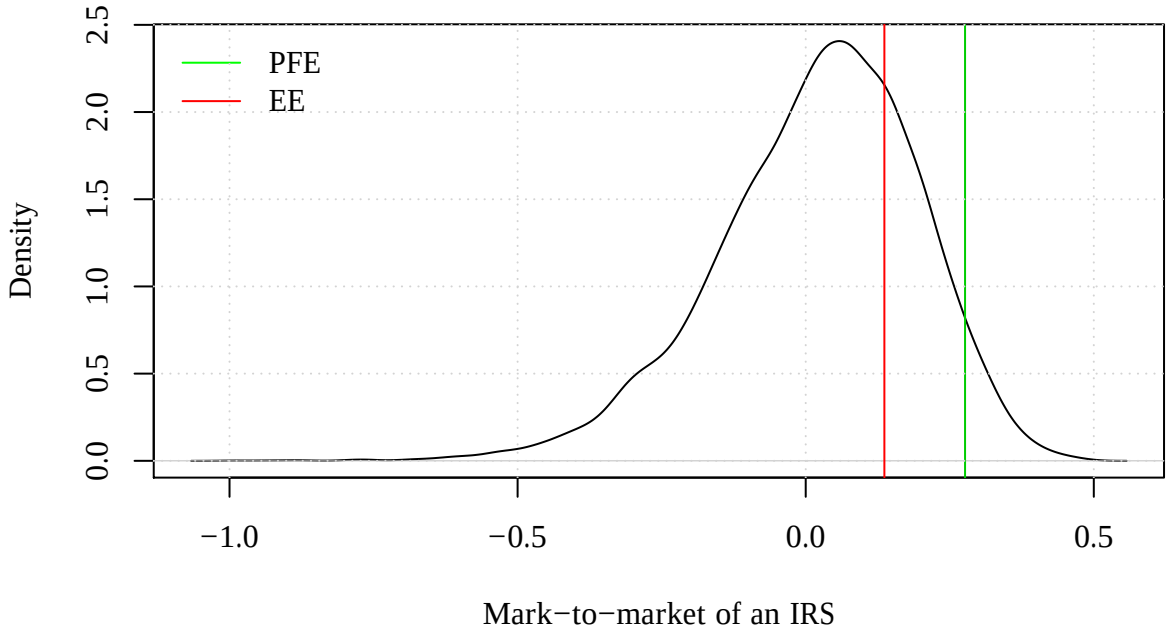


Figure 8: Expected Exposure and Potential Future Exposure.

The Expected Exposure (EE) represents the amount expected to be lost if the counterparty defaults.

$$\mathrm{EE}(t) = E[\max(0, \mathrm{MtM}(t))]$$

The 95% confidence level Potential Future Exposure (PFE) defines the exposure that will be exceeded with a probability smaller than or equal to 5%.

$$\mathrm{PFE}(t) = E[\max(0, q(\mathrm{MtM}(t), 0.95)]$$

The following plot shows the evolution of PFE and EE from the inception to the maturity of an IRS.

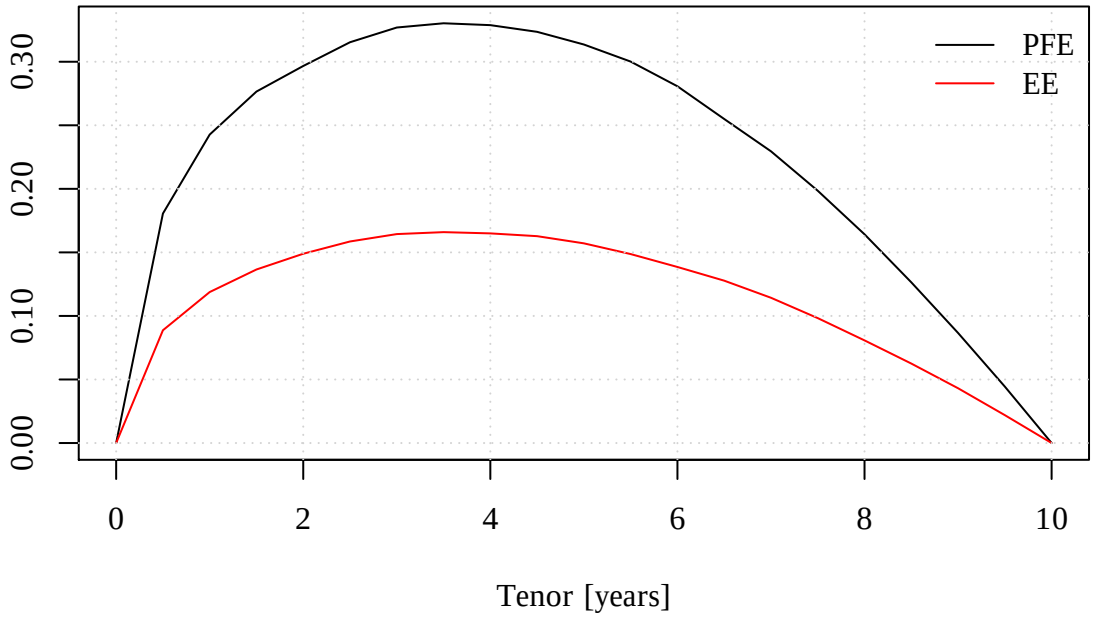


Figure 9: Time series of Expected Exposure and Potential Future Eposure of an IRS.

As expected, the EE profile is non-monotonic: it starts from zero at inception, increases significantly, and then decreases as maturity approaches. This is the result of two main factors:

1. the volatility is larger for longer time horizons: this leads to the initial increase of EE;
2. as time goes by, the trade expires and the cash flows are exchanged, hence the exposure decreases.

The PFE is usually larger than the EE. However, they both start from 0 at inception and take on values equal to zero in the last five coupon dates, because the mark-to-market of the contract in the last two years appears to be negative.

7 Credit Value Adjustment

The function `CVA` computes the unilateral CVA of an uncollateralized IRS, given the CDS spread of the counterparty. Let assume the absence of *wrong way risk*: hence, there is independence between default probability, exposure and recovery values. The expression for unilateral CVA, which is a function of both credit and market risk, is given by

$$\text{CVA} = E_t \left[(1 - R_c) 1_{t < \tau_c \leq T} D(t, \tau_c) \text{MtM}(\tau_c)^+ \right]$$

First, the implied default probabilities in CDS spreads must be derived following these steps:

1. a relationship between the hazard rate (λ) and the CDS spread is derived. The LGD is fixed at 40%:

$$\lambda = \frac{S_{\text{CDS}}}{1 - R_c} = \frac{S_{\text{CDS}}}{\text{LGD}}$$

2. the cumulative default probability is obtained:

$$F(T) = 1 - e^{-\lambda T}$$

3. the marginal default probabilities are computed.

These steps are performed in the function `MakePD`, which is displayed below.

```
Make_PD <- function(given_spread, LGD, valuation_dates){

  interpolate <- splinefun(x = given_spread$t,
                           y =given_spread$CDS_spread)

  Risky_curve = data.frame(tenor = valuation_dates,
                      spread = interpolate(valuation_dates)*0.0001)
  Risky_curve$LGD <- rep(LGD, nrow(Risky_curve))
  Risky_curve$lambda <- Risky_curve$spread/LGD
  Risky_curve$PD_cum <- 1 - exp(-Risky_curve$lambda * Risky_curve$tenor)

  Risky_curve$PD_marg <- rep(NA, nrow(Risky_curve))
  Risky_curve$PD_marg[1] <- Risky_curve$PD_cum[1]
  Risky_curve$PD_marg[2:length(valuation_dates)] <- na.omit(
                                                    diff(
                                                      Risky_curve$PD_cum))
  return(Risky_curve)
}
```

The following function computes the CVA.

```
CVA <- function(tenor, LGD, discount_factor, MtM_Paths, CDS_spread){

    EE <- Risk_Measures(MtM_Paths, conf_level = 0.95)$EE
    PD <- Make_PD(given_spread = CDS_spread, LGD = LGD,
                  valuation_dates = tenor)
    PD <- PD$PD_marg
    LGD = rep(LGD, length(tenor))
    summary_table <- data.frame(tenor,
                                LGD,
                                discount_factor,
                                EE,
                                PD)

    summary_table$CVA_component <- summary_table$LGD *
                                     summary_table$discount_factor *
                                     summary_table$EE *
                                     summary_table$PD

    CVA <- sum(summary_table$CVA_component)
```

```
    results = list(summary_table, CVA)
    names(results) <- c('Summary_table', 'CVA')
    return(results)
}
```

In the following table, the information needed to compute CVA are summarized. The final CVA of the IRS, which is computed as the sum of the single CVA components, is equal to 1.3%.

	Tenor	LGD	Discount factor	EE	PD	CVA components
1	0.0	0.4	1.0000	0.0018	0.0000	0.000000
2	0.5	0.4	0.9940	0.0887	0.0026	0.000090
3	1.0	0.4	0.9851	0.1188	0.0106	0.000496
4	1.5	0.4	0.9746	0.1366	0.0159	0.000844
5	2.0	0.4	0.9638	0.1489	0.0188	0.001079
6	2.5	0.4	0.9530	0.1586	0.0199	0.001205
7	3.0	0.4	0.9418	0.1644	0.0197	0.001219
8	3.5	0.4	0.9291	0.1659	0.0186	0.001145
9	4.0	0.4	0.9148	0.1649	0.0177	0.001065
10	4.5	0.4	0.8991	0.1628	0.0175	0.001022
11	5.0	0.4	0.8825	0.1571	0.0173	0.000960
12	5.5	0.4	0.8654	0.1487	0.0170	0.000873
13	6.0	0.4	0.8482	0.1385	0.0164	0.000772
14	6.5	0.4	0.8312	0.1277	0.0158	0.000670
15	7.0	0.4	0.8146	0.1143	0.0151	0.000564
16	7.5	0.4	0.7987	0.0983	0.0146	0.000460
17	8.0	0.4	0.7839	0.0808	0.0143	0.000363
18	8.5	0.4	0.7703	0.0625	0.0142	0.000274
19	9.0	0.4	0.7584	0.0431	0.0141	0.000184
20	9.5	0.4	0.7485	0.0219	0.0139	0.000091
21	10.0	0.4	0.7408	0.0000	0.0137	0.000000

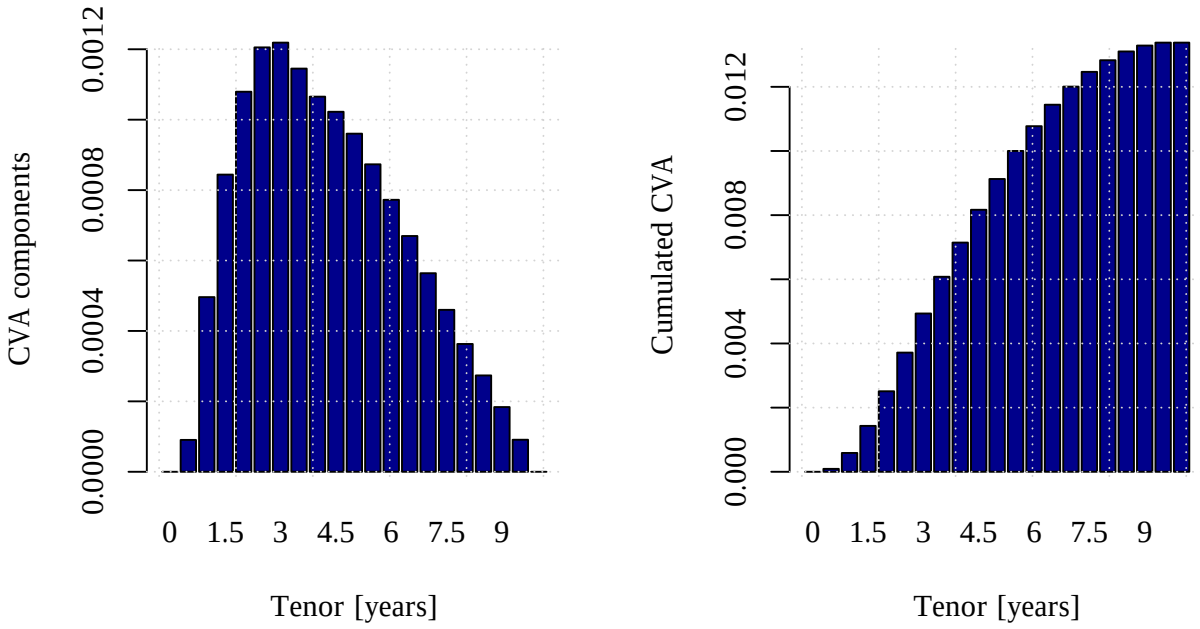


Figure 10: Credit Value Adjustment of an IRS: sigle components and cumulated value.

8 Collateralization of an interest rate swap

Let assume that the trade is perfectly collateralized with cash, with daily margins and immediate collateral transfer. Hence, the mark-to-market is fully collateralized but there is still some residual exposure, generated by the Margin Period of Risk (MPOR). At the moment of default, even though the mark-to-market is collateralized, the exposure is not equal to zero because some days (10, in this case) are required to liquidate the collateral and replicate the position on the market. Clearly, during the MPOR the mark-to-market of an IRS changes, and some collateralized exposure is still present.

In this training case, we are fully collateralized on every repricing date, hence the remaining exposure is from t_i to $t_i + 10$ days. On each repricing date, the value of the mark-to-market must be simulated over 10 days and the collateralized exposure must be evaluated. This would require nesting a Monte Carlo simulation on top of another Monte Carlo simulation, which would be computationally intensive. In order to ease the computations, Least Squares Monte Carlo (LSMC) technique is used.

The idea behind LSMC is to simulate the risk factors forward, from the valuation date until the expiry date, and then price the IRS backward. As a result, it is possible to obtain the price distribution at each time step from inception to maturity. The following steps are implemented in the `LSMC` function:

- the short rate with a 10-days time step is simulated. Short rate values and their squares are the exogenous variables and fill the 2^{nd} order paths matrix $\mathbf{G}_t$;
- the matrix $\mathbf{X}_t$ is obtained by subtracting the fixed leg coupon rate K from $\mathbf{G}_t$;

- starting from the maturity date, the value of an IRS is defined as:

$$\text{IRS}_t = \mathbf{X}_t\beta_t + \varepsilon_t$$

$$\varepsilon_t \sim \mathcal{N}(0, v)$$

- on each valuation date, the aforementioned regression is performed, and the fitted values are obtained for each t:

$$\widehat{\text{IRS}}_t = \mathbf{X}_t\beta_t$$

- the value of an IRS on each $t_i + 10$ day is computed exploiting the regression coefficients:

$$\widehat{\text{IRS}}_{t+10} = \mathbf{X}_{t+10}\beta_t$$

- the collateralized exposure, CE, is derived:

$$\text{CE} = \widehat{\text{IRS}}_{t+10} - \text{IRS}_t$$

```
LSMC = function(mtm_df,
                rates,
                mpor_curve,
                fixed_rate)
{
  my_seq = as.numeric(rownames(mtm_df))
  mpor_eval_tenors = na.omit(
    mpor_curve$tenor[which(mpor_curve$tenor %in% my_seq) + 1])

  irs_lsmc = foreach(i = 1:length(mpor_eval_tenors),
                     .combine = cbind.data.frame) %do%
    {
      X_train = unlist(rates[as.character(my_seq[i]), ]) - fixed_rate
      X_train = cbind(X_train, X_train ^ 2)
      colnames(X_train) = c('r - k', '(r - k)^2')
      y = unlist(mtm_df[as.character(my_seq[i]), ])
      model = lm(y ~ X_train)
      X_pred = unlist(rates[as.character(mpor_eval_tenors[i]), ]) - fixed_rate
      X_pred = cbind(X_pred, X_pred ^ 2)
      colnames(X_pred) = c('r - k', '(r - k)^2')
      irs_fitted = predict(model, data.frame(X_pred))

      return( irs_fitted )
    }

  value = cbind(irs_lsmc, 0)
  colnames(value) = c(mpor_eval_tenors, tail(my_seq, 1))

  return( value )
}
```

Once the collateralized exposure has been obtained, the computation of the EE, PFE, and CVA of the contract is performed. The results are displayed below.

```
# Simulate Hull-White with a 10-days time step
paths = HullWhiteSim(curve_10d)
```

The CVA of the collateralized exposure is 0.133%. It is about ten times smaller than the CVA of the uncollateralized IRS, as expected.

Figure 11: EE and PFE of the Collateralized Exposure.

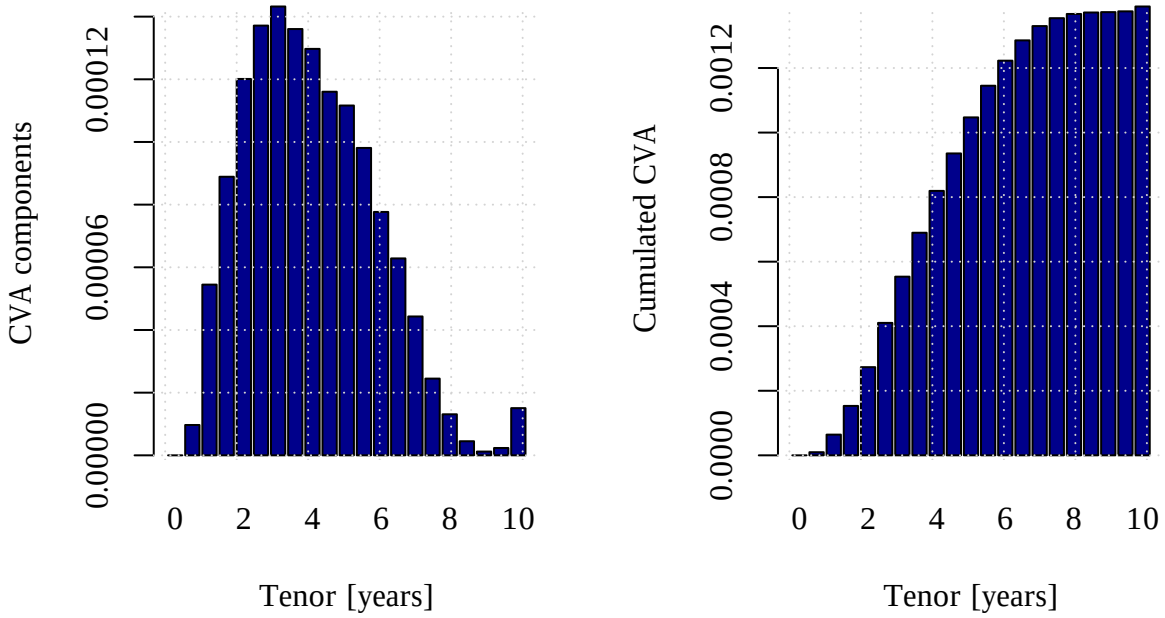


Figure 12: CVA of a collateralized IRS.

The following plot exhibits the comparison between the Expected Exposure of the collateralized and uncollateralized contract. As expected, the collateralized EE appears to be significantly lower than the EE of the uncollateralized IRS. In collateralized trades the CVA is greatly reduced, however it is not equal to zero because the Expected Exposure is still positive, although small, owing to the presence of the Margin Period of Risk.

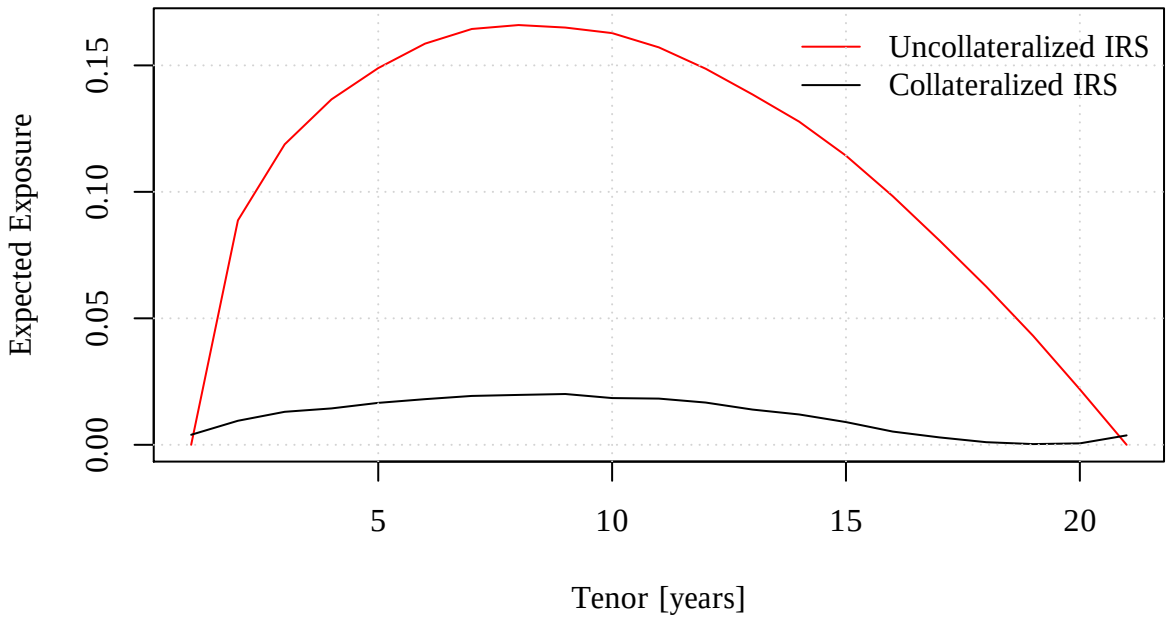


Figure 13: Comparison between the Expected Exposure of a collateralized and an uncollateralized IRS.

9 Hull-White model with stochastic CIR volatility

The assumption of constant interest rate volatility is removed: stochastic volatility is introduced in the model. Let assume that the instantaneous variance follows a Cox-Ingersoll-Ross process, described by the following stochastic differential equation:

$$\mathrm{d}v(t) = \lambda(k - v(t))\mathrm{d}t + \xi\sqrt{v(t)}\mathrm{d}W^v(t)$$

where $\mathrm{d}W^v(t)$ is is a standard Brownian motion independent of $W(t)$. The CIR model is characterised by a mean-reverting drift, where k is the long-term variance and the strictly positive parameter λ represents the speed of the mean-reversion. The coefficient ξ is the volatility of the variance process. The presence of the square root ensures that the volatility does not take on negative values.

9.1 Hull-White model simulations: stochastic volatility

Let the following parameters for the CIR process:

- $\lambda = 1$
- $k = 0.0004$
- $\xi = 0.02$

```
# Compute curves with a 30-days time step
tenor_30days <- seq(0, 10, 1/12)
curve_30days <- Make_curves(given_curve, tenor_30days)
```

The following plot displays the simulated paths of the stochastic volatility:

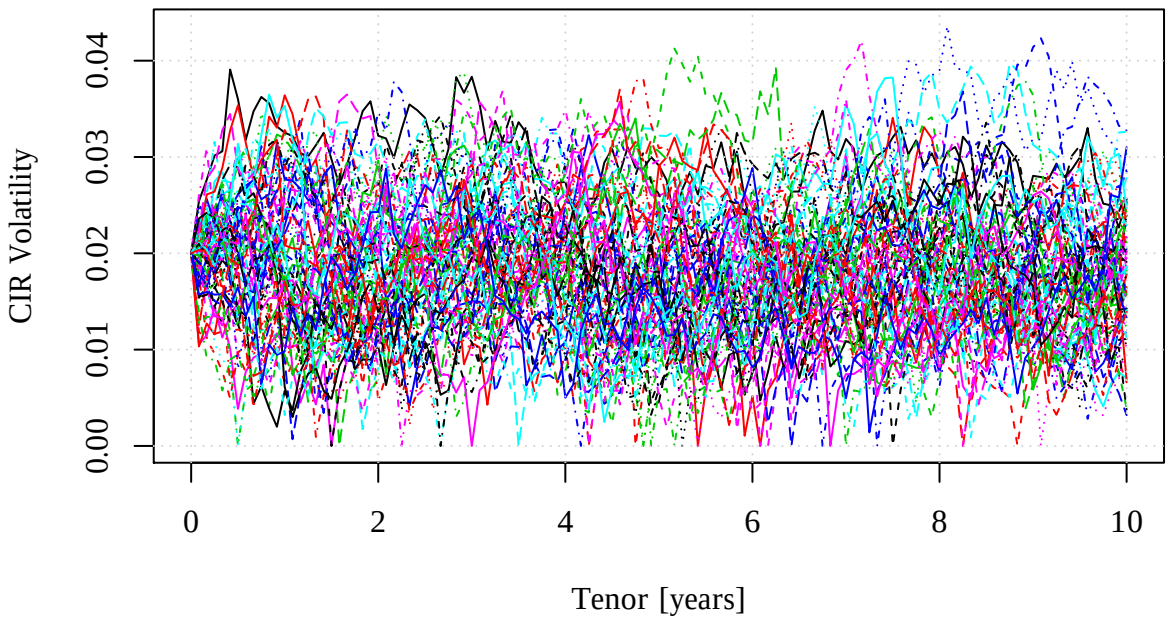


Figure 14: Cox-Ingersoll-Ross stochastic volatility paths.

From the volatility simulated values, a single path is chosen.

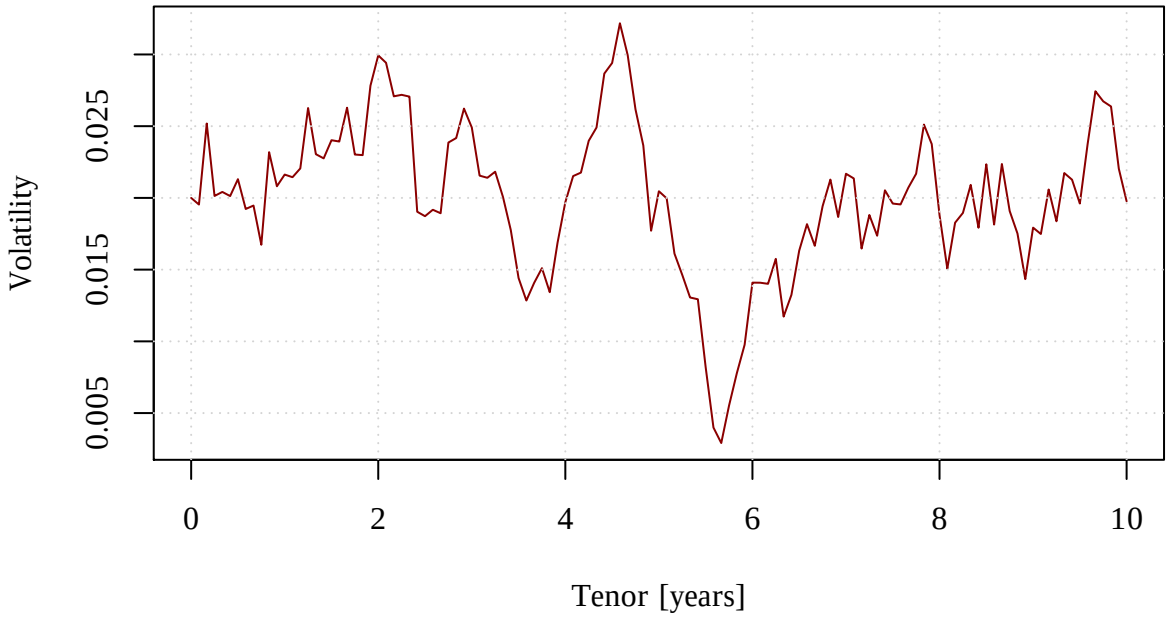


Figure 15: A time series of stochastic volatility.

```
path_CIR <- HullWhiteSim(curve = curve_30days,n_scenarios = n_scenarios,
                         lambda = lambda,k = k,xi = xi,a = a,
                         variance_path = stoch_vol )
```

The simulates short rate under Hull-White model with stochastic volatility are shown in the following Figure:

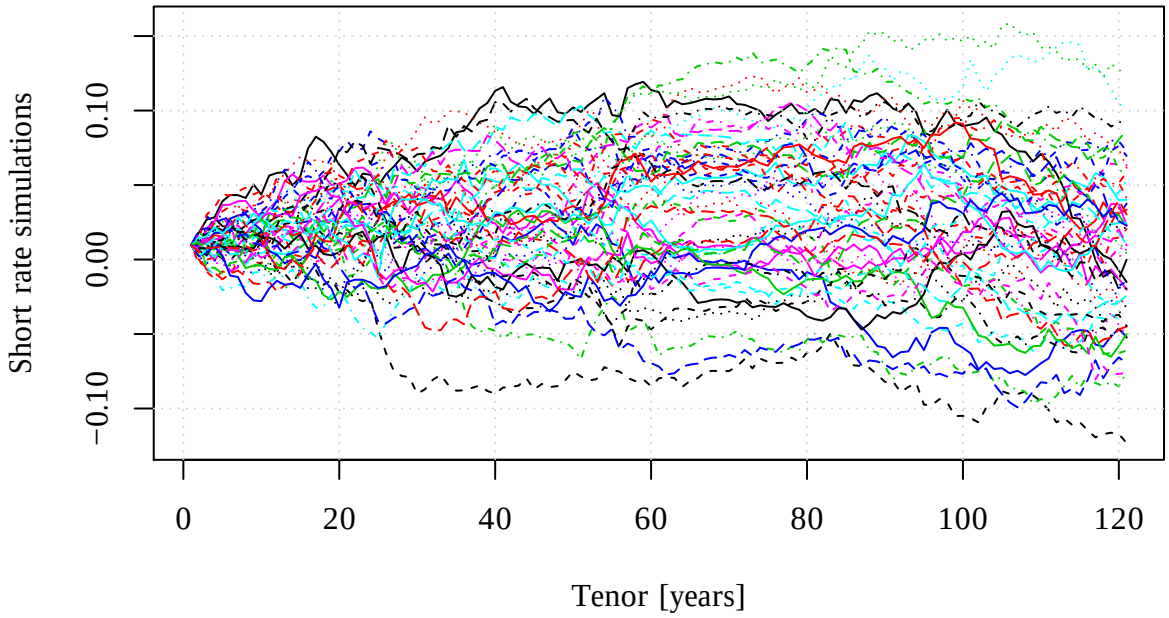


Figure 16: Short rate simulations with stochastic volatility

9.2 Risk measures under stochastic volatility

First, 10,000 paths of the short rate are simulated under the Hull-White model with a 30-days time step, exploiting the time series of volatility obtained under the CIR model. The short rates and discount factors are used to price an interest rate swap. The calculations are performed in the function `IRSStochVol`. The results of the computation of the mark-to-market, EE and PFE of the contract are displayed below.

The function `IRSStochVol.R` computes all the relevant information for Question 8.

```
IRS_StochVol <- function (curve,
                          n_scenarios,
                          lambda,
                          k,
                          xi,
                          a,
                          fixed_rate = 0.02959,
                          conf_level = 0.95,
                          LGD = 0.4,
                          given_spread){

  HWS = HullWhiteSim(curve = curve, n_scenarios = n_scenarios,
                     lambda = lambda, k = k, xi = xi, a = a)

  rates = do.call(cbind, lapply(HWS, function(x){ x$rate }))
  variancess = do.call(cbind, lapply(HWS, function(x){ x$variance }))
  rownames(variancess) = rownames(rates) = curve$tenor
```

```
  best_v = variancess[,18]  # CIR simulated volatility time series

  HWS_CIR = HullWhiteSim(curve = curve,
                         n_scenarios = n_scenarios,
                         lambda = lambda,
                         xi = xi,
                         a = a,
                         variance_path = best_v)

    rates_CIR = do.call(cbind, lapply(HWS_CIR, function(x){ x$rate }))
    variances_CIR = do.call(cbind,
                            lapply(HWS_CIR, function(x){ x$variance}))
    rownames(variances_CIR) = rownames(rates_CIR) = curve$tenor

    MtM_Path_CIR = MtM_Paths(evaluation_date = curve$tenor, curve = curve,
                             fixed_rate = fixed_rate, tau = 10,
                             rates = rates_CIR,
                             a = 0.01, variances = variances_CIR)

    rm = Risk_Measures(MtM_Paths = MtM_Path_CIR,
                       conf_level = conf_level)

    CvA_CIR <- CVA(tenor = curve$tenor,LGD = LGD,
                   discount_factor = curve$discount_factor,
                   MtM_Paths=MtM_Path_CIR,CDS_spread = given_spread)

    df <- data.frame(CVA = diff(CvA_CIR$Summary_table$CVA_component),
                     variances = diff(best_v))
    df_2 <- data.frame(CVA = CvA_CIR$Summary_table$CVA_component,
                       variances = best_v,
                       EE = rm[,'EE'],PFE = rm[,'PFE'])

    reg = lm(CVA~variances,data = df)
    vega <- summary(reg)$coefficients[-1,1]
    names(vega) <- c('Vega regression')
    results = list(HWS, best_v, HWS_CIR, MtM_Path_CIR, vega_2,df_2)
    names(results) <- c('HWS', 'variance_ts', 'HWS_CIR',
                        'MtM_path_CIR', 'Vega','DataFrame')
    return(results)
}
```

```
MtM_paths_CIR <- MtM_Paths(evaluation_date = tenor,
                       curve = curve_30days, fixed_rate = 0.02959,
                       tau = 10,
                       rates = rates_CIR,
                       a = 0.01,
                       variances = variances_CIR_sel)
```

The mark-to-market of the contract exhibits several extreme values, as a consequence of the stochastic volatility.

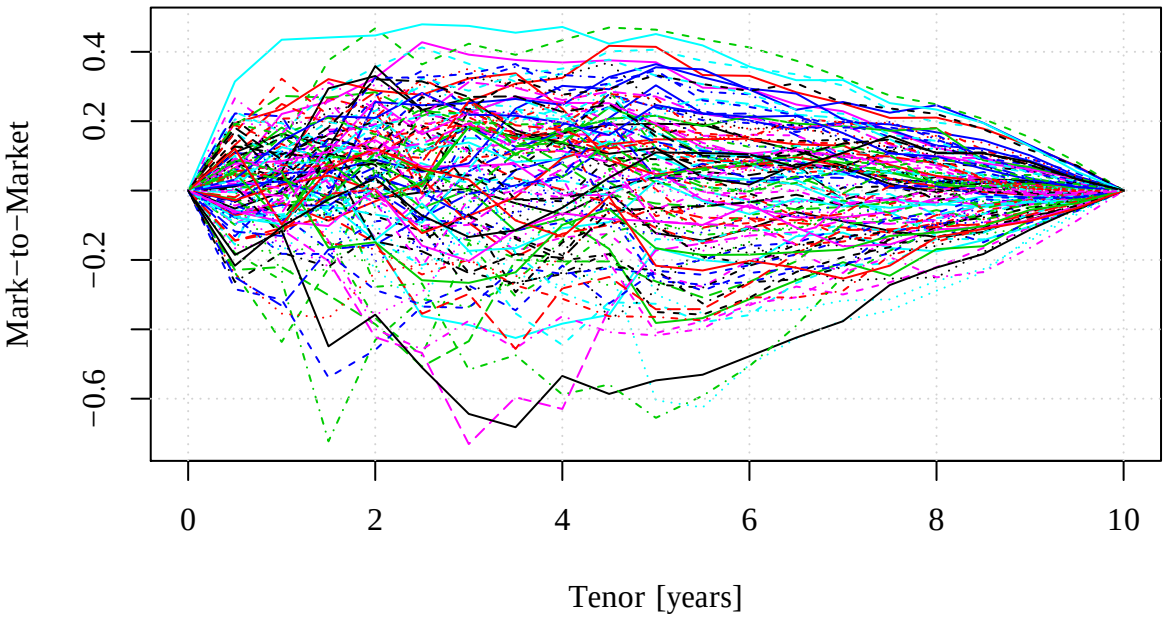


Figure 17: Simulated mark-to-market of an IRS with stochastic volatility.

Furthermore, as a consequence of the more volatile mark-to-market, the PFE appears to be notably larger than the Expected Exposure and its profile looks less smooth, owing to the several spikes in mark-to-market that have been observed in the previous plot.

Figure 18: Expected Exposure and Potential Future Exposure with stochastic volatility.

9.3 CVA: volatility risk

The CVA of the contract is 1.42%. As expected, it is slightly larger than the CVA computed under the assumption of constant volatility.

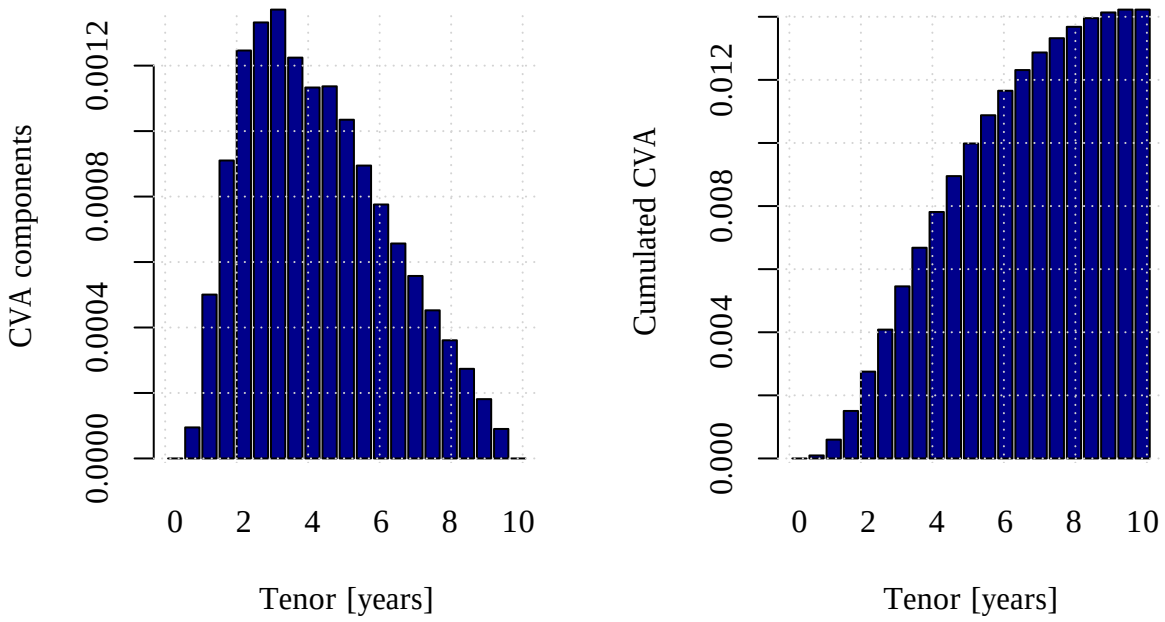


Figure 19: Time series of cumulated CVA.

In order to assess the volatility risk of the CVA, the adjustment is computed by analysing 10 different simulated scenarios, in which the volatility is kept at a constant value that ranges from 0% to 70%. The following plot displays CVA as a function of the interest rates volatility: CVA exhibits susbstantial volatility risk because it clearly increases when the interest rate volatility increases.

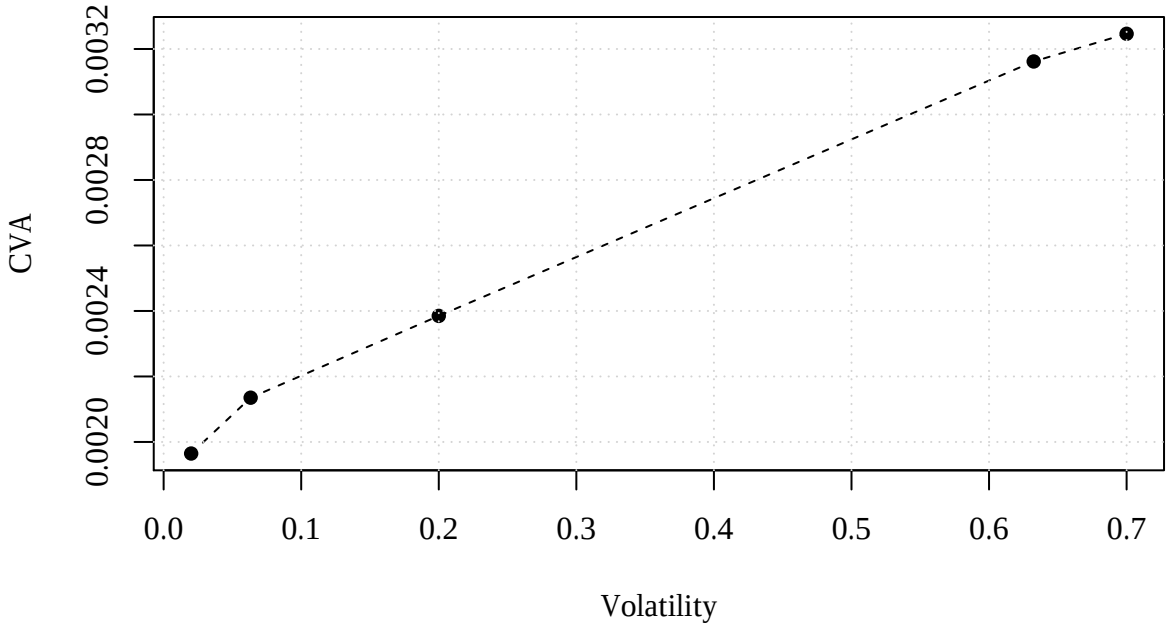


Figure 20: CVA as function of volatility.

In order to hedge the volatility risk of the CVA two solutions are feasible.

1. trading a swaption;

2. trading a variance swap.

The CVA of an interest rate swap can be represented as a series of swaptions written on the reverse swap (Sorensen and Bollier, 1994). The main idea behind this intuition is that, whenever a counterparty defaults, it cancels the non-recovered value of the swap. This situation is very similar to exercising the reverse swaption. Clearly, the sensitivity of CVA to interest rate volatility can be hedged via entering into the (opposite) swaptions.

Furthermore, trading a swaption would allow for an hedge on the volatility term structure. Short and long term swaptions tend to have low values because of the short maturity and the underlying swap low duration. The higher values are on medium-term maturities. Hence, the usual exposure profile that is analysed for IRS holds true for swaptions, too.

10 Conclusions

A final comparison between the CVA in the different analysed frameworks is presented.

IRS	Collateral	Volatility	CVA
1	Uncollateralized	Constant	0.0133
2	Collateralized	Constant	0.0013
3	Uncollateralized	Stochastic	0.0142

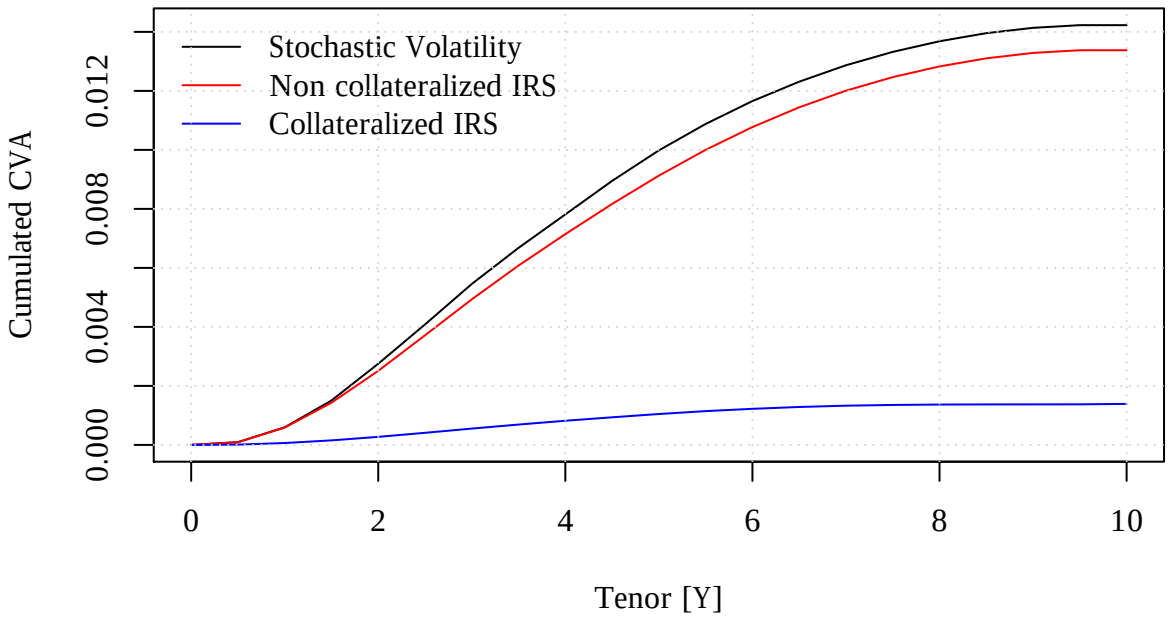


Figure 21: CVA comparison.

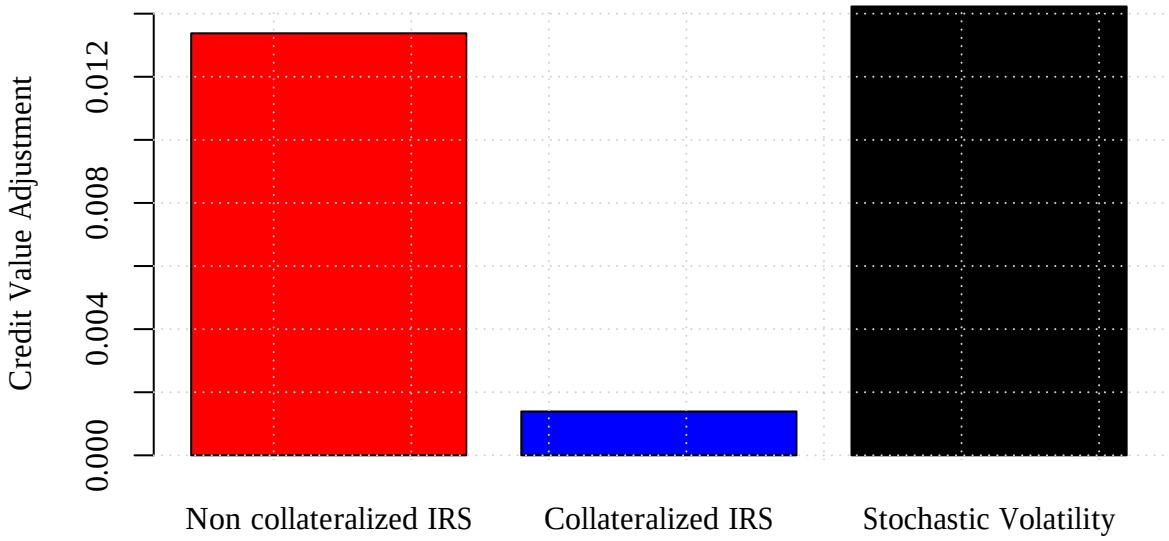


Figure 22: CVA comparison.

Furthermore, a comparison between risk measures is presented.

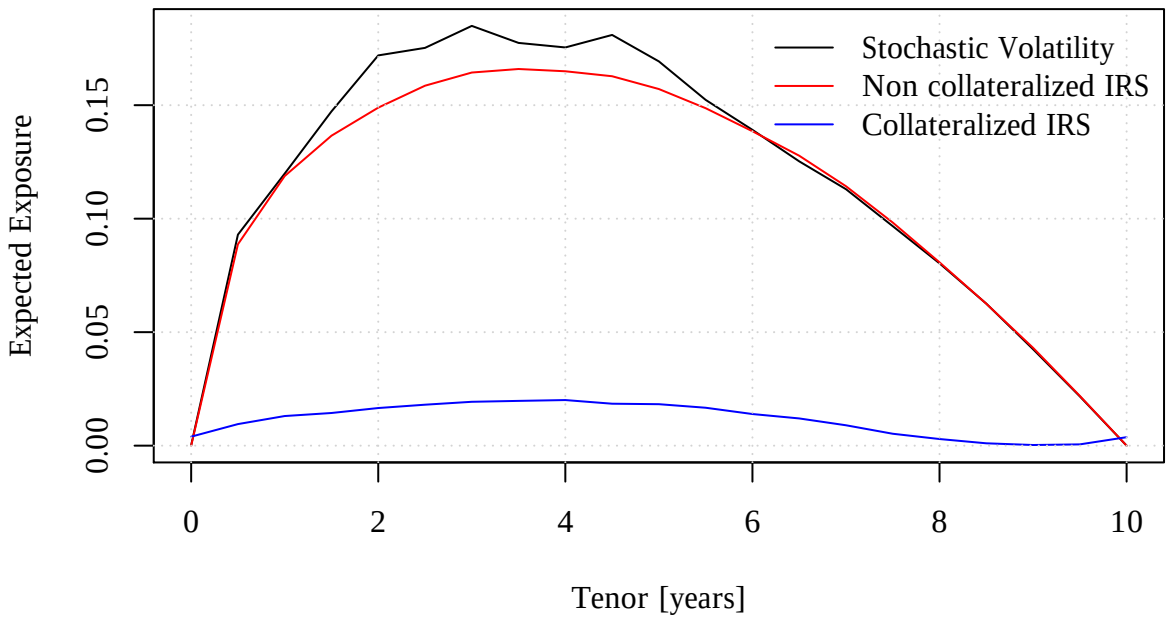


Figure 23: Expected Exposure comparison.

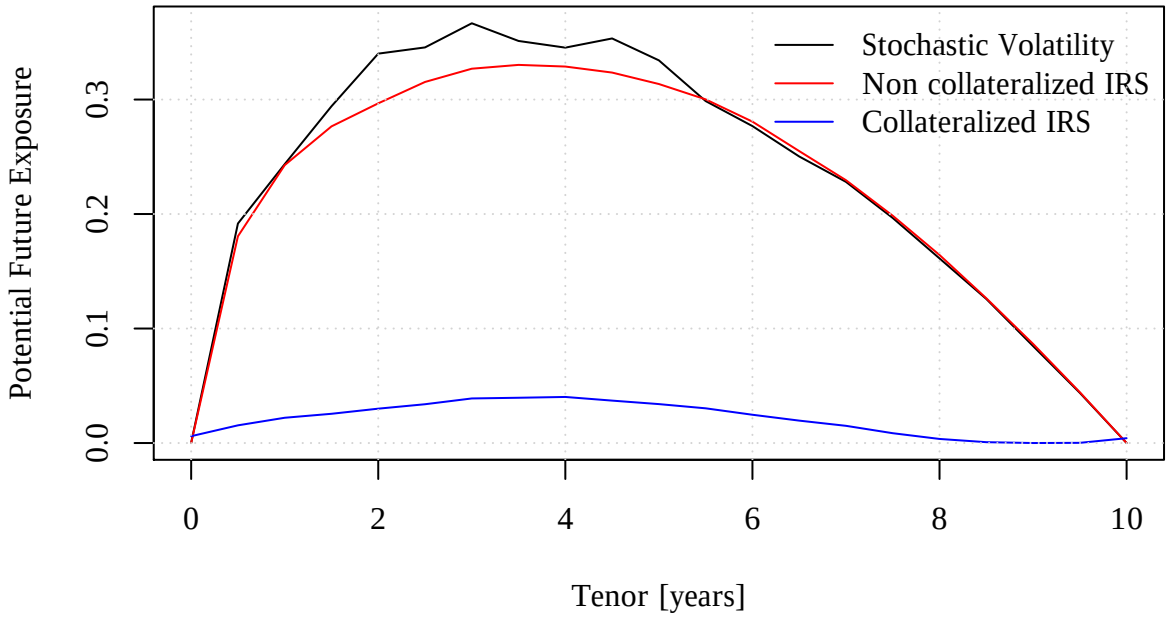


Figure 24: Potential Future Exposure comparison.

The presence of a full collateralisation allows to transform the exposure risk, which should be considered over a 10 years time period, into a risk measure that is computed only over the MPOR. Consequently, the risk measures and the CVA are strongly reduced. However, note that several assumptions have been made through the implementation of the model:

- the collateral has been assumed to be cash. In case of a different collateral, its mark-to-market, adjusted by an haircut, should be modelled as well;
- the remargin period has been set to zero: hence, it has been assumed that the effective time between a collateral call and receiving the collateral is zero. However, the time needed for liquidating the existing collateral and re-entering the market has been properly accounted for;
- no thresholds or Minimum Transfer Amounts have been modelled: any amount of collateral could be posted.

Cleary, these assumptions produce a slight underestimation of the CVA of the collateralized IRS. Nonetheless, it is still possible to perfectly gauge the effect of collateralization on counterparty risk.

11 References

- Karyampas, D. *Lecture Notes and Slides*, 2020.
- Hull, J. *Options, Futures and Other Derivatives*, Boston, Prentice Hall, 2019.
- Gregory, J.*Counterparty Credit Risk and Credit Value Adjustment*, Wiley Finance Series, 2013.

- Sorenson, E. H., Bollier, *Pricing Swap Default Risk*, Financial Analysts Journal, 50, 1994.