Economic Scenarios Generation for Insurance: ESG package and other tools (Pt. I : ESG)

Rmetrics Workshop on R in Finance and Insurance 2014, Paris

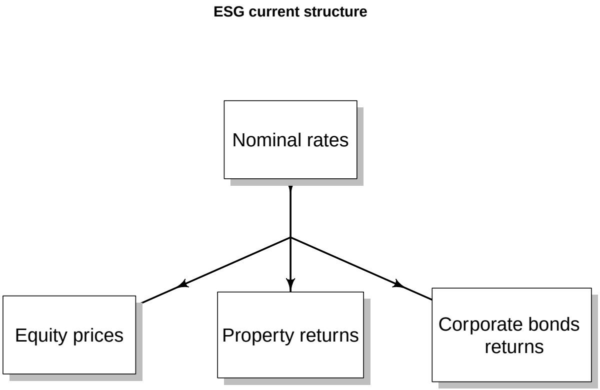
Thierry Moudiki, Frederic Planchet

June 27, 2014

ESG

- R package `ESG` was designed to provide **a minimal Economic Scenarios Generator** (ESG) for valuation and capital requirements calculations in **Solvency II**.
- Currently : projections of risk factors in a **risk-neutral** world.
- Available risk factors are : **nominal rates**, **equity returns**, **property returns**, **corporate bonds returns**.

ESG


ESG current structure
Nominal rates
Equity prices
Property returns
Corporate bonds returns

ESG

Available risk factors

Let $(W_t)_{t \geq 0}$ be a standard brownian motion.

- **nominal rates** : Hull-White Extended Vasicek (HW)

$$dr_t = (\theta(t) - ar_t)\, dt + \sigma dW_t^{(r)}$$

- **equity prices** : Geometric Brownian motion with stochastic Hull-White interest rates (BSHW)

$$dS_t = r_t S_t dt + \sigma S_t dW_t^{(E)}$$

$$dW_t^{(E)} dW_t^{(r)} = \rho dt$$

ESG

Available risk factors (cont'd)

- **property returns** : Geometric Brownian motion with stochastic Hull-White interest rates

$$dS_t = r_t S_t dt + \sigma S_t dW_t^{(P)}$$

- **corporate bonds returns** : HW + intensity of default + liquidity spread = Longstaff-Mithal-Neis (LMN)
    - Intensity of default

$$d\lambda_t = (\alpha - \beta\lambda_t)dt + \sigma\sqrt{\lambda_t}dW_t^{(\lambda)}$$

    - Additional liquidity spread

$$d\eta_t = \eta_t dW_t^{(\eta)}$$

ESG

Simulation/discretization

Example : Vasicek model.

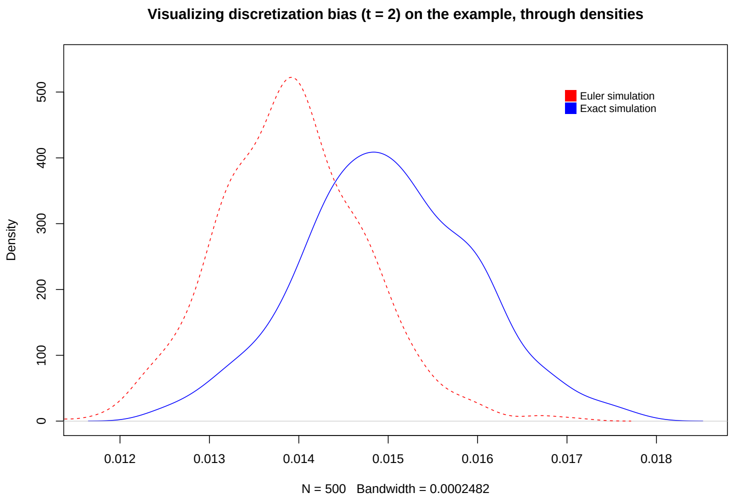
- Most **simple and intuitive** way : **Euler scheme** (1st order Ito-Taylor development) :

$$r_{t_{i+1}} - r_{t_i} = a(\theta - r_{t_i})(t_{i+1} - t_i) + \sigma\epsilon\sqrt{t_{i+1} - t_i}$$

- Another way : 2nd order development, **Milstein scheme**. More precise. But When σ is constant, not necessary. More complicated formula.
- Third way : **exact simulation** of the transition distribution between t_{i+1} and t_i :

$$r_{t_{i+1}} = e^{-a(t_{i+1}-t_i)} r_{t_i} + \theta(1 - e^{-a(t_{i+1}-t_i)}) + \sigma\epsilon\sqrt{\frac{1 - e^{-a(t_{i+1}-t_i)}}{2a}}$$

ESG


Visualizing discretization bias (t = 2) on the example, through densities
Euler simulation
Exact simulation
Density
0
100
200
300
400
500
0.012
0.013
0.014
0.015
0.016
0.017
0.018
N = 500 Bandwidth = 0.0002482

The package's structure

An `S4` **object-oriented** architecture, around 2 classes : `ParamsScenarios` and `Scenarios`, with associated ***getter*** and ***setter*** methods.

`ParamsScenarios`	`Scenarios`
`horizon`	A `ParamsScenarios` attribute
`n`	`ForwardRates` slot
HW, BSHW, LMN parameters	`ZCRates` slot
Equity/short rate correlation	One slot for each model path

ESG

Using the package : 2 ways

- **Step by step approach** : Using successive **getter** and **setter** methods, to know exactly what is done at each step
  - `set/get*Params*Scenarios*`
  - `set/getForwardRates`
  - `set/getZCRates`
  - `set/get*Paths`
- **Through an interface** wrapping the successive **getter** and **setter** methods, which is easier but also safer
  - `rShortRate` : nominal rates (HW)
  - `rStock` : equity (BSHW)
  - `rDefaultSpread` + `rLiquiditySpread` : credit (LMN)
  - `rRealEstate` : property (BSHW)

Examples of use of ESG

```
# loading ESG
library(ESG)
# needed for yield curve interpolation
library(ycinterextra)
# yield to maturities
txZC <- c(0.01422,0.01309,0.01380,0.01549,0.01747,0.01940,
          0.02104,0.02236,0.02348,0.02446,0.02535,0.02614,
          0.02679,0.02727,0.02760,0.02779,0.02787,0.02786,
          0.02776,0.02762,0.02745,0.02727,0.02707,0.02686,
          0.02663,0.02640,0.02618,0.02597,0.02578,0.02563)
# maturities
u <- 1:30
# the yield curve must be interpolated on a monthly basis
ZC <- fitted(ycinter(yM = txZC, matsin = u,
                     matsout = seq(1, 30, by = 1/12),
                     method = "SW"))
```

Examples of use of ESG : **Step by step approach**

```r
# object creation
objScenario <- new("Scenarios")
# Setting the basic scenario's parameters
objScenario <- setParamsBaseScenarios(objScenario,
                                      horizon = 5,
                                      nScenarios = 100)
# Parameters for BSHW
objScenario <- setRiskParamsScenariosS(objScenario,
                                       vol = .1,
                                       k = .2,
                                       volStock = .2,
                                       stock0 = 100,
                                       rho= .5)
```

Examples of use of ESG : **Step by step approach** (cont'd)

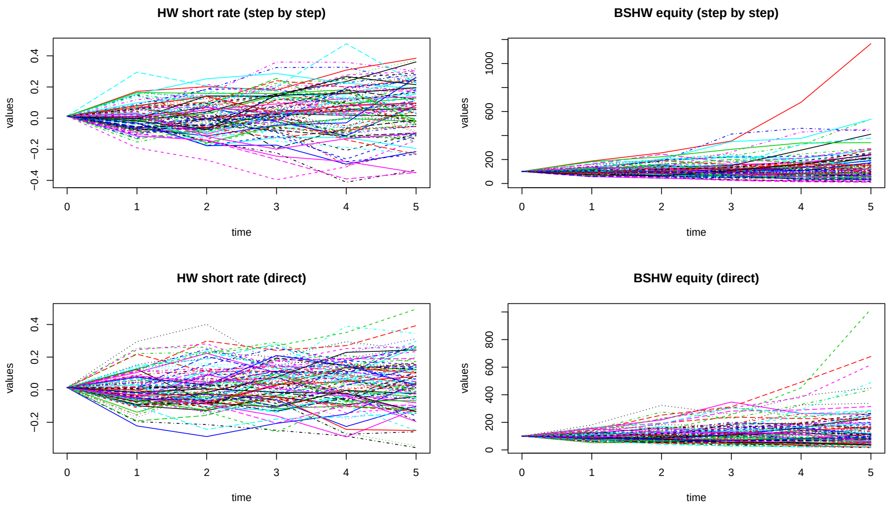
```
#  Setting the forward rates
objScenario <- setForwardRates(objScenario, ZC = ZC,
                               horizon = 5)

# Simulation, setting the Paths' slots
objScenario <- customPathsGeneration(objScenario,
                                     type="stock")
y.step <- getstockPaths(objScenario)
```

Examples of use of ESG : **Through the interface**

```r
# only one function is called, instead of 6
y.interface <- rStock(horizon = 5,
                      nScenarios = 100,
                      ZC = ZC,
                      vol = .1,
                      k = .2,
                      volStock = .2,
                      stock0 = 100,
                      rho= .5)
```

```r
# visualizing the results :
par(mfrow=c(2, 2))
matplot(t(y.step$shortRatePaths), type = 'l')
matplot(t(y.step$stockPaths), type = 'l')
matplot(t(y.interface$shortRatePaths), type = 'l')
matplot(t(y.interface$stockPaths), type = 'l')
```


HW short rate (step by step)
BSHW equity (step by step)
HW short rate (direct)
BSHW equity (direct)
values
time
0.4
0.2
0.0
−0.2
−0.4
1000
600
200
0
800
400
1
2
3
4
5

Example of Best Estimate Liability calculation

We consider an insurance company, offering a **unit-linked** contract

- The insured party pays a premium equal to 1.
- The premium is invested in a stock : the **unit**
- **Maturity** : 10 years
- **Systematic surrender rate** : 2% (unavoidable)
- **Economic surrender rate** : 5% (depends on the economic situation). Added to systematic surrender rate whenever the unit-link falls below the initial value invested in it, which is 1
- The contract is **entirely redeemed at maturity** $\Longrightarrow$ **surrender rate at maturity** : 100%

In Solvency II, the **Best Estimate liability** related to the contract is equal to the average discounted value of its future cash-flows

Example of Best Estimate Liability calculation (cont'd)

- $r^{(s)}$: the **systematic surrender rate** (2%)
- $r^{(e)}$: the **economic surrender rate** (5%)
- $\forall i = 1, \ldots, 10$

$$r_i^{(total)} := \left(r^{(s)} + r^{(e)} \mathbb{1}_{\left(\frac{S_i}{S_{i-1}} < 1\right)} \right) \mathbb{1}_{(i \leq 9)} + 100\% \times \mathbb{1}_{(i=10)}$$

- $(r_t)_{t \geq 0}$: the **instantaneous short rate** (HW)
- $(S_t)_{t \geq 0}$: the **value of the unit** (BSHW)
- $Res_i = Res_{i-1} \times \frac{S_i}{S_{i-1}} \left(1 - r_i^{(total)}\right)$; $Res_0 = 1$: the **reserves**

The **Best Estimate liability** associated to the contract is equal to :

$$BEL = \mathbb{E}^* \left[\sum_{i=1}^{10} e^{-\int_0^i r_u du} Res_i \right]$$

Example of Best Estimate Liability calculation (cont'd)

- **No close formula** for the *BEL*. . .
- . . . Or difficult to derive $\Longrightarrow$ **Monte Carlo simulation** with `ESG`
- An `R` function, `calculFlux`, is defined for the calculation of ALM cash-flows. `calculFlux` depends on projected short rates, projected values of the unit, and the surrender rates, depending on the latter
- Parameters for ALM projection :

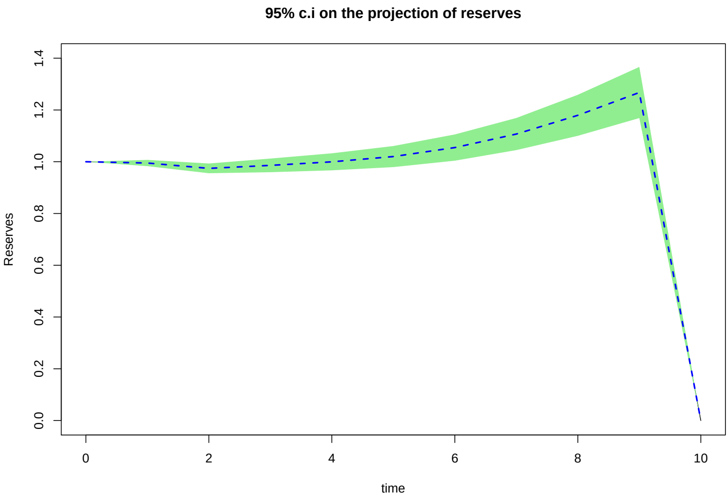
```
k <- 0.12                    # short rates' mean-reversion speed
sTaux <- 0.05                # volatility of short rates
sUC <- .16                   # volatility of the unit
rho_rS <- .5                 # correlation unit vs short rates
H <- 10                      # maturity of the contract
nSimulations <- 1000         # number of simulations
tauxRachatS <- .02           # systematic surrender rates
tauxRachatC <- .05           # economic surrender rates
```

Example of Best Estimate Liability calculation (cont'd)

```
set.seed(10)
# Simulation of the unit and of short rates with rStock
traj <- rStock(horizon=H, nScenarios=nSimulations, ZC=ZC,
               vol=sTaux, k=k, volStock=sUC, stock0=1,
               rho=rho_rS)
# Short rates
trajectoiresTaux <- traj$shortRatePaths
# Unit (a stock)
trajectoiresUC <- traj$stockPaths
# Future cash-flows and discount factors
Flux_futurs <- calculFlux(trajectoiresTaux,trajectoiresUC,
                          tauxRachatS,tauxRachatC)
# discounted cash-flows
ActuFlux_futurs <- Flux_futurs$flux*Flux_futurs$actu
```

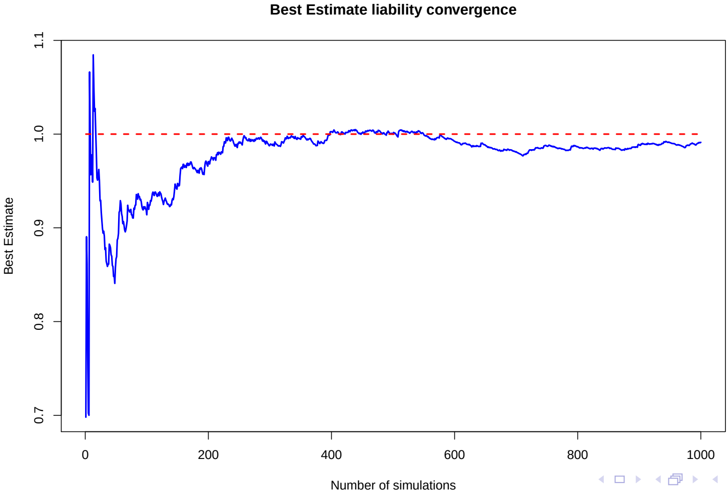
```r
# Future distribution of the reserves
Res <- t(apply(Flux_futurs$PM, 2,
        function(x) summary(x))[c(-1,-6), ])
rownames(Res) <- paste0("Year ", 0:10)
```

```
##          1st Qu. Median  Mean 3rd Qu.
## Year 0     1.000  1.000 1.000    1.00
## Year 1     0.864  0.980 0.995    1.10
## Year 2     0.756  0.929 0.974    1.14
## Year 3     0.668  0.901 0.986    1.23
## Year 4     0.609  0.883 1.000    1.28
## Year 5     0.555  0.849 1.020    1.33
## Year 6     0.487  0.833 1.050    1.41
## Year 7     0.430  0.821 1.110    1.47
## Year 8     0.392  0.783 1.180    1.63
## Year 9     0.360  0.774 1.270    1.69
## Year 10    0.000  0.000 0.000    0.00
```


95% c.i on the projection of reserves
Reserves
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0
2
4
6
8
10
time

```r
(BestEstimate <- sum(ActuFlux_futurs)/nSimulations)
```

```
## [1] 0.9912
```



ESG

Future versions

- More **flexibility on the interpolation of zero-rates** (not only monthly frequency required)
- Projection is annual $\Longrightarrow$ impossible to obtain correct estimations of discount factors $\Longrightarrow$ add an option for **changing the sampling frequency**
- Adding **correlation/dependence between the risk factors**
- Adding **real world models**
- `ESGtoolkit`, will be used in `ESG`