ECB calls for lighter treatment of high-quality ABS. How QuantLib might help?

Michael von den Driesch / Matthias Groncki
IKB

IKB
Deutsche Industriebank

Motivation

Chart 1. European ABS outstanding (EUR billion)

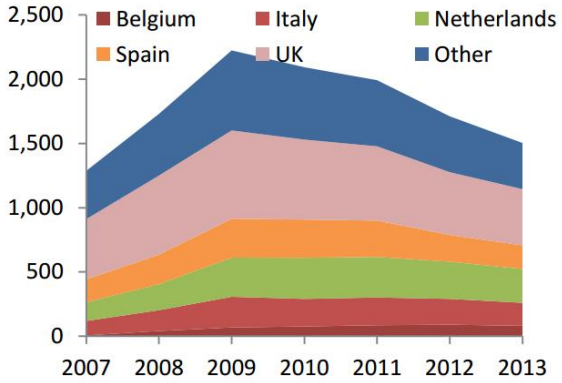
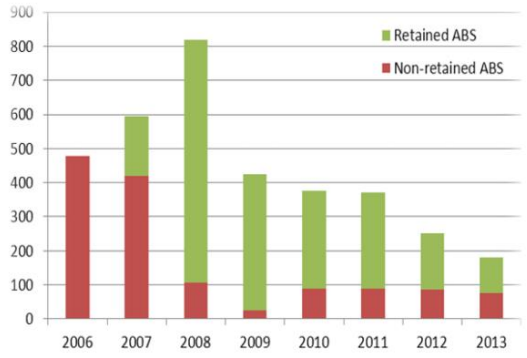


Chart 2. European ABS issuance (EUR billion)



Source: Association for Financial Markets in Europe (AFME). End observation: 2013 Q4.

Yves Mersch states during his speech on the conference at the Deutsche Börse in March 2014 (http://www.ecb.europa.eu/press/key/date/2014/html/sp140407.en.html):

- We have seen that SME ABSs can play a key role in bridging the gap between deleveraging banks and investors seeking to diversify their portfolio
- It is not too late to change course: not all EU securitisations deserve the stigma attached to them for the past few years. There is a need to restore coherence across financial sectors in particular amid the unfavourable regulatory treatment of (high quality) securitisation instruments, without violating prudential principles…

Mario Draghi states during the IMF Spring Meeting :

- The ECB is also contributing towards a revitalisation of the asset-backed securities market in Europe, …

The ECB and Bank of England want to revitalize the ABS market for high-class SME ABS.

Sample SME ABS

- Assume a large (many obligors) pool high-quality SME loans
- Aim: Create an ABS with n tranches with a replenishment period

- Model:
  - Exogene PDs and LGDs (e.g. from an internal model)
  - Copula Model for Default Times
  - Deterministic delayed recoveries
  - No prepayments

- Pricing with Monte-Carlo-Simulation
  - In each simulation step:
    - for each obligor generate a default time from Copula-Model
    - Roll out the cashflows until default and send it through the waterfall
  - Estimate the expected cashflows for each tranche

Problems:
- Simulation of a large pool can be very time consuming
- Setup of the underlying pool should be easy for the end user (Structurer, Risk Analyst, etc..)
- The waterfall should be easy to implement / to read and to adjust, even for non quantivtative developers
- Calculation of sensitivities through monte carlo very time consuming

How can QL help or why have we choosen QL ?

As we all know, QuantLib provides a lot of tools for doing this:

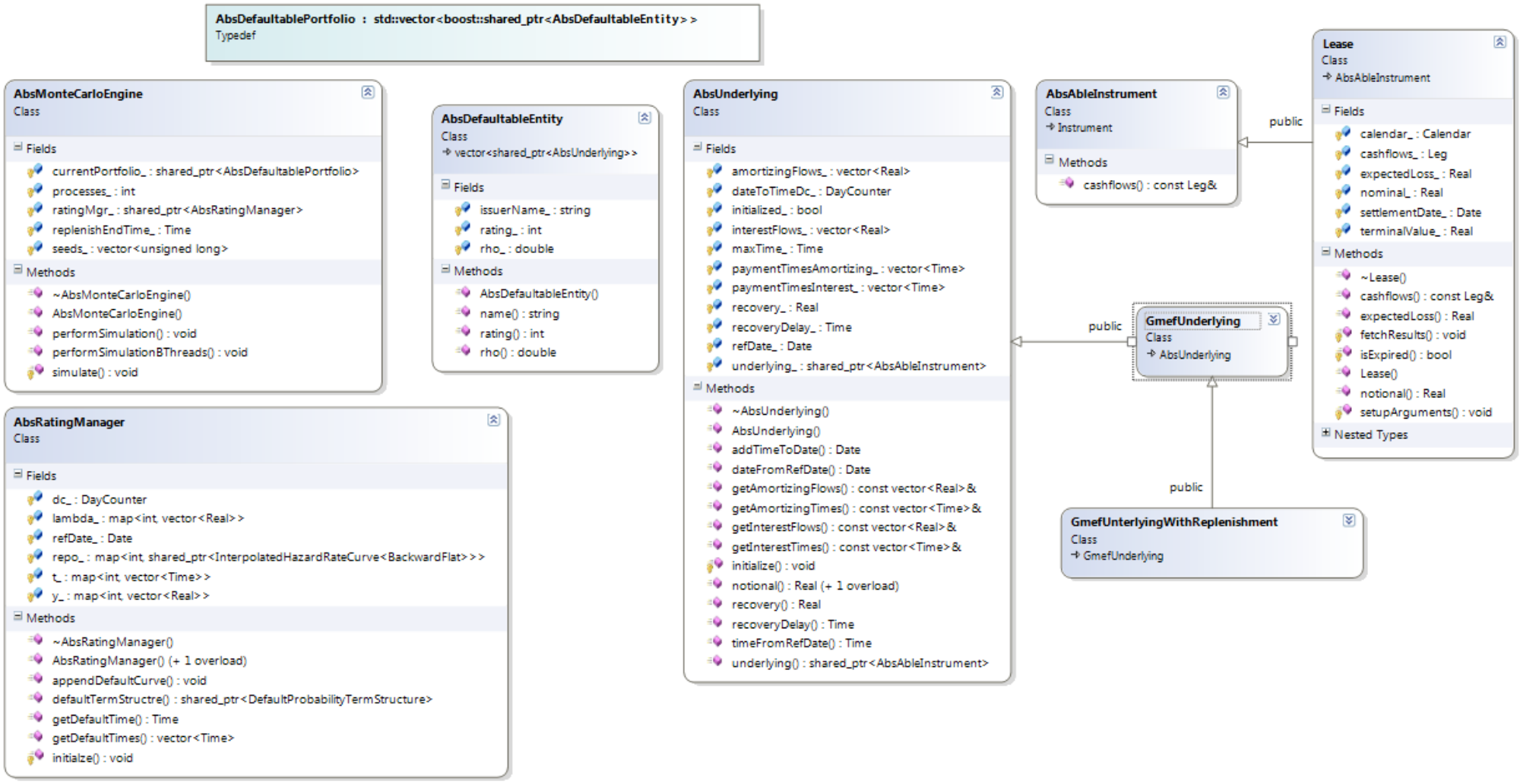
- YieldTermStructures
- DefaultTermStructures
- Random-Number-Generators
- Dates & Calendar & Conventions
- Copulae
- Its written in C++ (fast) and technologies like OpenMP or boost::threads enable us to use multithreading for the Monte-Carlo-Simulation

- But it also has some limitations:
  - Not easy to learn / to use for non-developers
  - Need to compile / no „on the fly“ computations
  - C++ is not that comfortable when it comes to handling large datasets, data analysis and visualation

- To bridge this gap we use the SWIG interface to Python to hide all the „number crunching“ under the hood in the C++ Library and give the user the comfort of a scripting language to setup the portfolio, modelling the waterfall and visualize the results.

QuantLib C++ as back-end and Python as front-end seems for us a good solution

UML class diagramm of „C++ Side“ (work in progress)



Userinterface +
Implementing the waterfall on the liability side (Python)

Switch to Ipython-Notebook