Open Source Risk Engine

Peter Caspers

Quaternion Risk Management

9 December 2016

Agenda

Open Source Risk Project

QuantLib Extension Library

Data Library

Analytics Library

Agenda

Open Source Risk Project

QuantLib Extension Library

Data Library

Analytics Library

Overview

The *Open Source Risk Project* aims at establishing a transparent peer-reviewed framework for **pricing and risk analysis** that can serve as

- a benchmarking, validation, training, teaching reference
- an extensible foundation for in-house and vendor solutions

Overview

The *Open Source Risk Project*

- is sponsored by Quaternion Risk Management
- is open source software (Modified BSD License)
- permits using, modifying and even commercialising the code
- is based on QuantLib, the open source source library for quantitative finance.

Its main software project is *Open Source Risk Engine* (ORE).

Overview

ORE

1. provides **contemporary risk analytics and pricing**
2. accessible to the **end users** by providing **transparent interfaces** for trade and market data, as well as system configuration
3. well **documented** usage, methodology and codebase

Analytics Scope

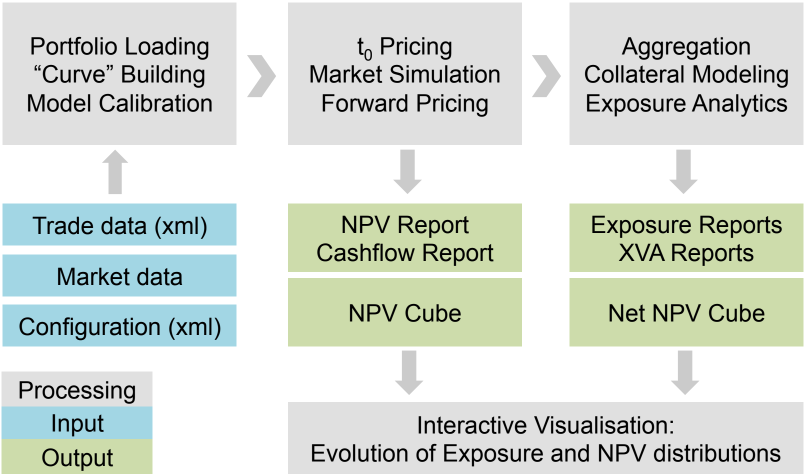
Portfolio pricing and cash flow projection

Derivative portfolio analytics based on a Monte Carlo simulation framework

- Credit exposure evolution with netting and collateral (EE, EPE, EEPE, PFE) supporting regulatory capital charge calculation under internal model methods
- Collateral modeling with Dynamic Initial Margin (DIM)
- Derivative value adjustments (CVA, DVA, FVA, COLVA, MVA)
- Market risk measures

Quaternion
Risk Management

Data Flow



Portfolio Loading
“Curve” Building
Model Calibration

t_0 Pricing
Market Simulation
Forward Pricing

Aggregation
Collateral Modeling
Exposure Analytics

Trade data (xml)

Market data

Configuration (xml)

NPV Report
Cashflow Report

NPV Cube

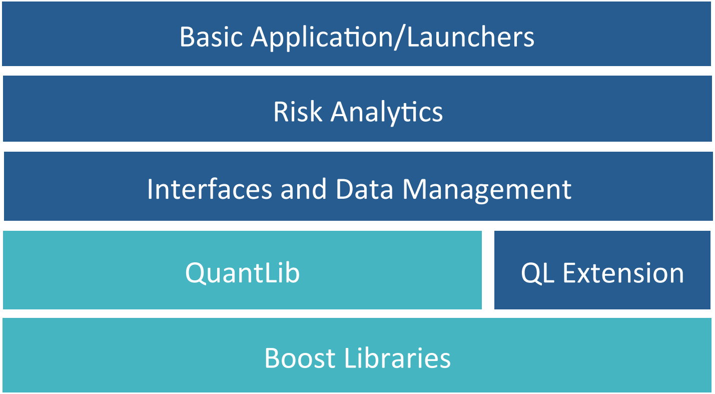
Exposure Reports
XVA Reports

Net NPV Cube

Interactive Visualisation:
Evolution of Exposure and NPV distributions

Processing
Input
Output

Components



Basic Application/Launchers

Risk Analytics

Interfaces and Data Management

QuantLib

QL Extension

Boost Libraries

Quaternion
Risk Management

Products

ORE’s initial product scope comprises Interest Rate and FX products

- Interest Rate Swaps
- Caps/Floors
- Swaptions
- Cross Currency Swaps
- FX Forwards
- FX Options

Features

ORE comes with extensive tests, examples and documentation

- Comprehensive **test suites**
- Various **examples** to demonstrate typical use cases
- Several ways to launch ORE and **visualise results**
- Detailed **user guide** on examples and parametrisation
- Comprehensive **source code documentation**

User Interfaces

- XML file driven **command line application**
- **Jupyter notebook** for NPV Cube and Exposure visualisation
- Excel and LibreOffice Calc **spreadsheets** to kick off jobs and present results
- Web-based risk **dashboard** for interactive presentation of ORE results

Audience

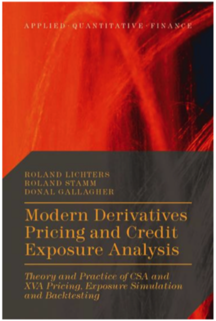
ORE is suited for users in the industry and academia

- **End Users** in Front Office, Risk, Finance and Audit functions interested in quickly parameterising and running analytics,
- **Quants and Developers** interested in inspecting, modifying and extending the code base
- **Students and Academics** interested in practical examples to enhance their course work

Methods

Models and methods applied in ORE are based on

Modern Derivatives Pricing and Credit Exposure Analysis, Palgrave MacMillan 2015



APPLIED · QUANTITATIVE · FINANCE

ROLAND LICHTERS
ROLAND STAMM
DONAL GALLAGHER

Modern Derivatives Pricing and Credit Exposure Analysis

Theory and Practice of CSA and XVA Pricing, Exposure Simulation and Backtesting

Risk Factor Evolution

- IR/FX: Cross Currency **Linear Gauss Markov** model (i.e. Gaussian single factor interest rate models for each currency, Geometric Brownian Motion for each FX component)
- Inflation: **Dodgson-Kainth**, Jarrow-Yildirim
- Credit: **Gaussian**, Cox-Ingersoll-Ross, Black-Karasinski, Peng-Kou
- Equity: **GBM**, Heston
- Commodity: **GBM**, 2-factor Gabillon

IR/FX Model: Cross Currency LGM

$$
\begin{aligned}
dz_0 &= \alpha_0 \, dW_0^z \\
dz_i &= \gamma_i \, dt + \alpha_i \, dW_i^z, \qquad i > 0 \\
\frac{dx_i}{x_i} &= \mu_i \, dt + \sigma_i \, dW_i^x, \qquad i > 0
\end{aligned}
$$

$$
\begin{aligned}
\gamma_i &= -\alpha_i^2 \, H_i - \rho_{ii}^{zx} \, \sigma_i \, \alpha_i + \rho_{i0}^{zz} \, \alpha_i \, \alpha_0 \, H_0 \\
\mu_i &= r_0 - r_i + \rho_{0i}^{zx} \, \alpha_0 \, H_0 \, \sigma_i \\
r_i &= f_i(0,t) + z_i(t) \, H_i'(t) + \zeta_i(t) \, H_i(t) \, H_i'(t), \qquad \zeta_i(t) = \int_0^t \alpha_i^2(s) \, ds
\end{aligned}
$$

Numeraire and zero bond:

$$
N(t) = \frac{1}{P(0,t)} \exp\left\{ H_t \, z_t + \frac{1}{2} H_t^2 \, \zeta_t \right\}
$$

$$
P(t,T,z_t) = \frac{P(0,T)}{P(0,t)} \exp\left\{ -(H_T - H_t) \, z_t - \frac{1}{2} \left(H_T^2 - H_t^2 \right) \zeta_t \right\}.
$$

Quaternion
Risk Management

Exposure Measures

In ORE we use the following exposure definitions

$$EE(t) = EPE(t) = \mathbb{E}^N \left[\frac{(NPV(t) - C(t))^+}{N(t)} \right]$$

$$ENE(t) = \mathbb{E}^N \left[\frac{(-NPV(t) + C(t))^+}{N(t)} \right]$$

where $NPV(t)$ stands for the netting set NPV and C is the posted collateral.

Basel III Exposures

Expected Exposure

$$EE_B(t) = \mathbb{E}[\max(NPV(t) - C(t), 0)] := \frac{EE(t)}{P(0,t)}$$

Expected Positive Exposure

$$EPE_B(T) = \frac{1}{T} \sum_{t<T} EE_B(t) \cdot \Delta t \qquad T = \min(1, \text{maturity})$$

Effective Expected Exposure

$$EEE_B(t) = \max(EEE_B(t - \Delta t), EE_B(t))$$

Effective Expected Positive Exposure

$$EEPE_B(T) = \frac{1}{T} \sum_{t<T} EEE_B(t) \cdot \Delta t$$

CVA and DVA

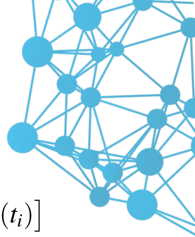
Unilateral discretised CVA and DVA

$$CVA = \sum_i PD(t_{i-1}, t_i) \times LGD \times EPE(t_i)$$

$$DVA = \sum_i PD_{Bank}(t_{i-1}, t_i) \times LGD_{Bank} \times ENE(t_i)$$

with

$EPE(t)$	expected exposure
$ENE(t)$	expected negative exposure
$PD(t_i, t_j)$	counterparty probability of default in $[t_i, t_j]$
$PD_{Bank}(t_i, t_j)$	our probability of default in $[t_i, t_j]$
LGD	counterparty loss given default
LGD_{Bank}	our loss given default



FCA and FBA

$$FVA = \underbrace{\sum_{i=1}^{n} f_b(t_{i-1},t_i)\,\delta_i\,\mathbb{E}^N\left[S_C(t_{i-1})\,S_B(t_{i-1})\,(NPV(t_i))^+\,D(t_i)\right]}_{\text{Funding Benefit Adjustment (FBA)}}$$

$$-\underbrace{\sum_{i=1}^{n} f_l(t_{i-1},t_i)\,\delta_i\,\mathbb{E}^N\left[S_C(t_{i-1})\,S_B(t_{i-1})\,(-NPV(t_i))^+\,D(t_i)\right]}_{\text{Funding Cost Adjustment (FCA)}}$$

$D(t_i)$ stochastic discount factor, $1/N(t_i)$ in LGM

$NPV(t_i)$ portfolio value after potential collateralization

$S_C(t_j)$ survival probability of the counterparty

$S_B(t_j)$ survival probability of the bank

$f_b(t_j)$ borrowing spread for the bank rel. to the coll. comp. rate

$f_l(t_j)$ lending spread for the bank rel. to the coll. comp. rate

COLVA

When the CSA defines a collateral compounding rate that deviates from the overnight rate, this gives rise to another value adjustment labeled COLVA. In the simplest case the deviation is just given by a constant spread Δ:

$$COLVA = \mathbb{E}^N \left[\sum_i C(t_i) \cdot \Delta \cdot \delta_i \cdot D(t_{i+1}) \right]$$

where $C(t)$ is the collateral posted and $D(t)$ is the stochastic discount factor $1/N(t)$ in LGM. Both $C(t)$ and $N(t)$ are computed in ORE's Monte Carlo framework, and the expectation yields the desired adjustment.

CSA Floor

$$\Pi_{Floor,\Delta} = \mathbb{E}^N\left[\sum_i C(t_i)\cdot((r(t_i)+\Delta)^+ - (r(t_i)+\Delta))\cdot\delta_i\cdot D(t_{i+1})\right]$$

Similar to **COLVA** with stochastic spread,

- paying off when overnight rates are negative
- with stochastic notional given by the amount of posted collateral
- with significant correlation between notional and rate

See Burgard-Kjaer, or Lichters/Stamm/Gallagher.

Collateral Model

ORE computes the collateral requirement (aka **Credit Support Amount**) through time along each Monte Carlo path

$$CSA(t_m) = \begin{cases} \max(0, V_{set}(t_m) - I_A - T_{hold}), & V_{set}(t_m) - I_A \geq 0 \\ \min(0, V_{set}(t_m) - I_A + T_{hold}), & V_{set}(t_m) - I_A < 0 \end{cases}$$

where

- $V_{set}(t_m)$ is the value of the netting set as of time t_m
- T_{hold} is the threshold exposure below which no collateral is required (possibly asymmetric)
- I_A is the sum of all collateral independent amounts attached to the underlying portfolio of trades (positive amounts imply that the bank has received a net inflow of independent amounts from the counterparty), assumed here to be cash.

Collateral Model

Collateral account balance: $C(t_m)$

Collateral shortfall: $CSA(t_m) - C(t_m)$

Taking the value $M(t_m)$ of unsettled margin calls into account:

$$\Delta(t) = CSA(t_m) - C(t_m) - M(t_m)$$

The margin **Delivery Amount** $D(t_m)$ is calculated as follows:

$$D(t_m) = \begin{cases} \Delta(t), & |\Delta(t)| \geq MTA \\ 0, & |\Delta(t)| < MTA \end{cases}$$

where MTA is the minimum transfer amount.

$D(t_m)$ is **settled with a delay** specified by the *Margin Period of Risk* (MPOR) which leads to residual exposure and XVA even for daily margining, zero thresholds and minimum transfer amounts.

Exposure and XVA Allocation to Trade Level

XVAs and exposures are typically computed at netting set level.

For accounting purposes it may be required to *allocate* XVAs from netting set to individual trade level such that the allocated XVAs add up to the netting set XVA.

This distribution is not trivial, since due to netting and imperfect correlation single trade (stand-alone) XVAs hardly ever add up to the netting set XVA: XVA is sub-additive similar to VaR.

ORE provides an allocation method (labeled *marginal allocation* in the following) which slightly generalises a method proposed by Pykhtin and Rosen in 2010.

Allocation is done path-wise which first leads to allocated expected exposures and then to allocated CVA/DVA by inserting these exposures into CVA and DVA equations.

Exposure and XVA Allocation to Trade Level

The allocation algorithm in ORE is as follows:

- Consider the netting set's discounted NPV after taking collateral into account, on a given path at time t:

$$E(t) = D(0,t)\,(NPV(t) - C(t))$$

- On each path, compute contributions A_i of the latter to trade i as

$$A_i(t) = \begin{cases} E(t) \times NPV_i(t)/NPV(t), & |NPV(t)| > \epsilon \\ E(t)/n, & |NPV(t)| \leq \epsilon \end{cases}$$

  with number of trades n in the netting set and trade i's value $NPV_i(t)$.

- The EPE fraction allocated to trade i at time t:

$$EPE_i(t) = \mathbb{E}\left[A_i^+(t)\right]$$

By construction, $\sum_i A_i(t) = E(t)$ and hence $\sum_i EPE_i(t) = EPE(t)$.

Dynamic Initial Margin and MVA

Partially covered in ORE, see Roland’s presentation.

Timeline

Release: 7 October 2016

Web site, FAQ, Forum:

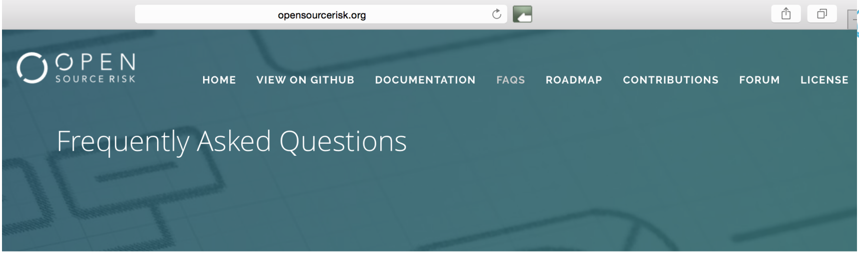
```
http://www.opensourcerisk.org
```

Code base:

```
https://github.com/OpenSourceRisk/Engine
```

Next release: Q1 2017

opensourcerisk.org

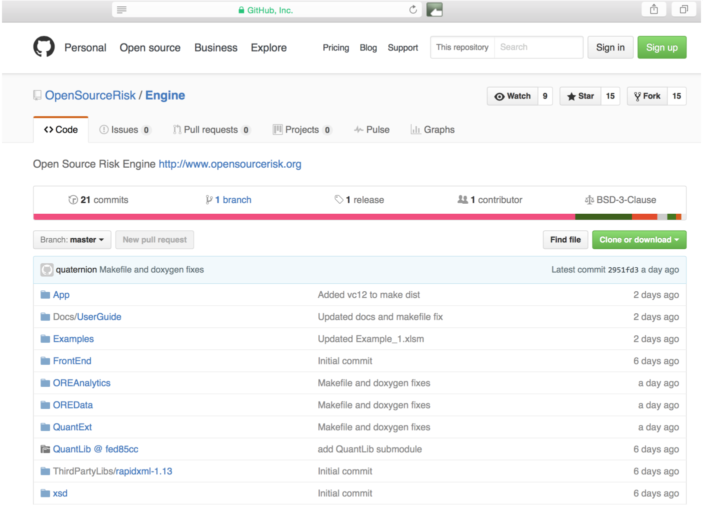


Frequently Asked Questions

General

+ What is Open Source Risk Engine?
+ What is QuantLib?
+ Is there a user guide for ORE?
+ Are there any tutorials for ORE?
+ Is there a technical document describing ORE?
+ What is it written in?

github.com/OpenSourceRisk/Engine



OpenSourceRisk / **Engine**

Open Source Risk Engine http://www.opensourcerisk.org

21 commits | **1** branch | **1** release | **1** contributor | BSD-3-Clause

quaternion Makefile and doxygen fixes — Latest commit 2951fd3 a day ago

App	Added vc12 to make dist	2 days ago
Docs/UserGuide	Updated docs and makefile fix	2 days ago
Examples	Updated Example_1.xlsm	2 days ago
FrontEnd	Initial commit	6 days ago
OREAnalytics	Makefile and doxygen fixes	a day ago
OREData	Makefile and doxygen fixes	a day ago
QuantExt	Makefile and doxygen fixes	a day ago
QuantLib @ fed85cc	add QuantLib submodule	6 days ago
ThirdPartyLibs/rapidxml-1.13	Initial commit	6 days ago
xsd	Initial commit	6 days ago

Roadmap

Analytics:

- SA-CCR, the new standard for derivatives capital
- Sensitivity analysis and stress testing
- Parametric VaR and initial margin methods

Asset classes and simulation models:

- Credit simulation, credit derivatives and loan products
- Default risk modeling and credit portfolio analysis
- Inflation simulation and inflation derivatives
- Equity simulation, equity derivatives
- Commodity simulation, commodity derivatives

Community

Users and developers are invited to contribute to the project:

- Discussions, feedback, enhancement proposals at
  `www.opensourcerisk.org/forum`
- Pull requests, see
  `www.opensourcerisk.org/contributions`

Sponsorship

Quaternion is committed to sponsor the project in several ways. In addition to the initial release, Quaternion provides resources to

- further develop ORE to regularly deliver extensions
- answer community questions in the ORE forum
- administrate and consolidate community contributions submitted via email or pull requests
- promote ORE at conferences and through publications
- seek additional sponsors to help promoting ORE and making it widely known and used

Interest Rate Swaps

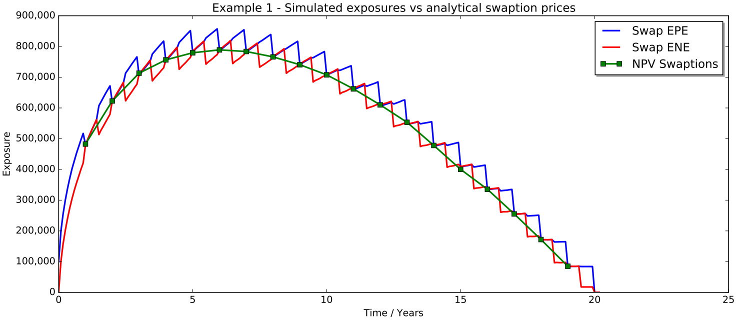


Figure: Vanilla ATM Swap expected exposure in a flat market environment.

Interest Rate Swaps

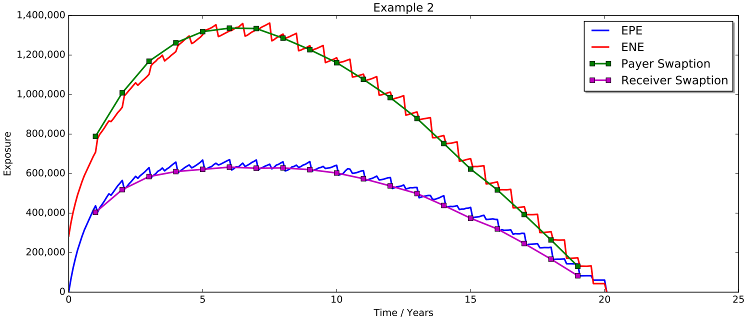


Figure: Vanilla ATM Swap expected exposure in a realistic market environment as of 05/02/2016.

Cross Currency Swaps

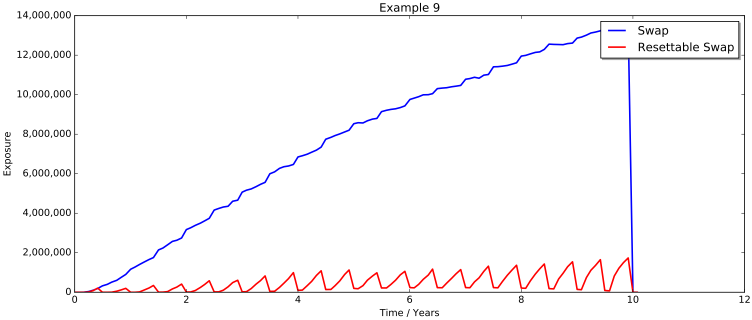


Figure: Cross Currency Swap exposure evolution with and without mark-to-market notional reset.

Netting and Collateral

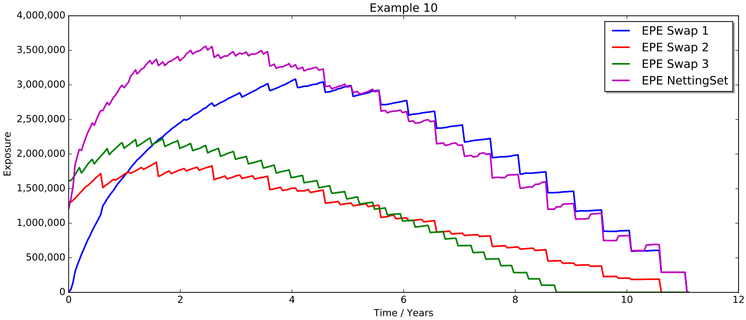


Figure: Three Swaps netting set, no collateral.

Netting and Collateral

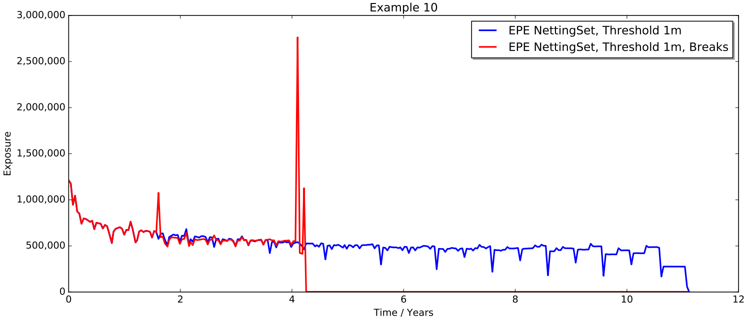


Figure: Three Swaps netting set, THR=1m EUR, MTA=100k EUR, MPR=2w. The red evolution assumes that the each trade is terminated at the next break date. The blue evolution ignores break dates.

Quaternion
Risk Management

Netting and Collateral

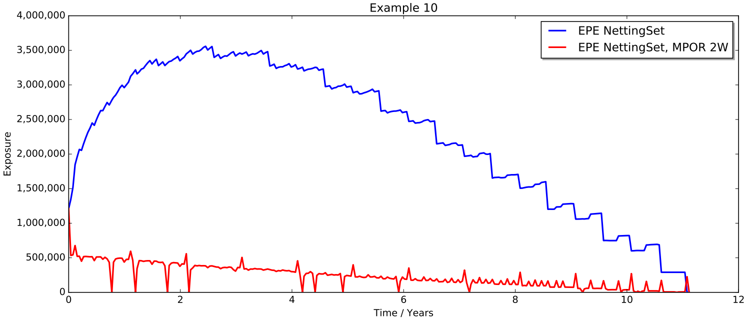


Figure: Three Swaps, THR=MTA=0, MPR=2W.

Exposure Allocation

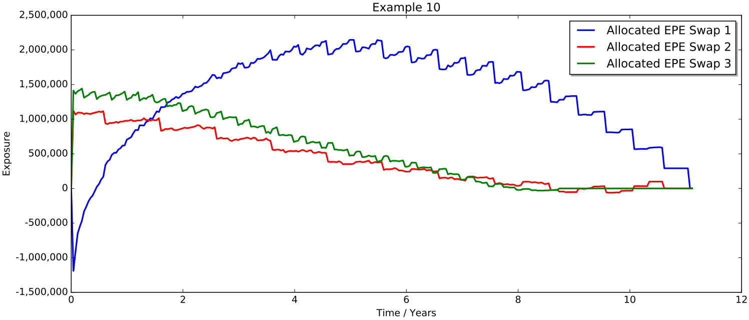


Figure: Exposure allocation without collateral (Pykhtin/Rosen).

Exposure Allocation

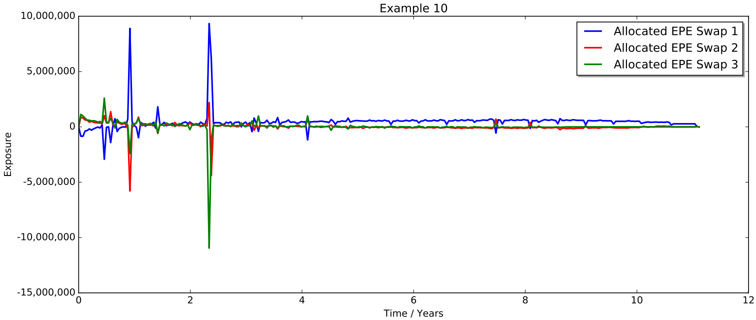


Figure: Exposure allocation with collateral and threshold 1m EUR (Pykhtin/Rosen).

Long-term Simulation

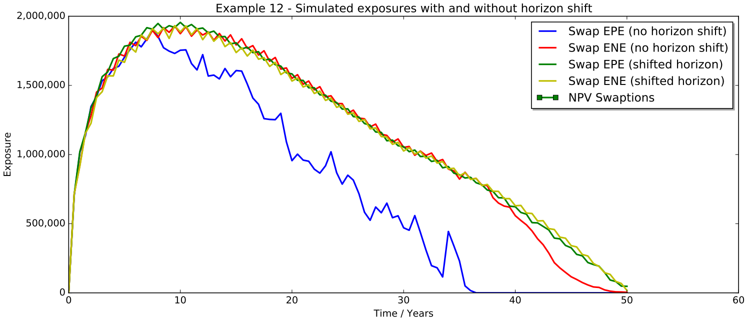


Figure: Long term Swap exposure simulation with and without LGM horizon shift to 30y.

Long-term Simulation

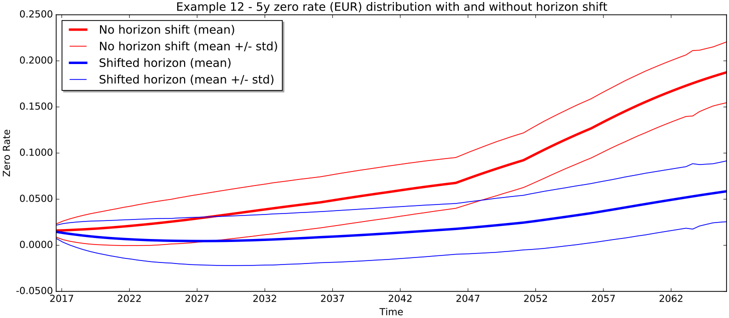


Figure: Long term rate simulation with and without LGM horizon shift to 30y.

Agenda

Open Source Risk Project

QuantLib Extension Library

Data Library

Analytics Library

QuantExt - QuantLib Extension Library

QuantExt adds supplementary building blocks to QuantLib

- a cross asset model and associated pricing engines
- rate helpers for bootstrapping cross currency and tenor basis curves
- a few instruments like currency swaps, basis swaps and average OIS swaps
- additional currencies and indexes

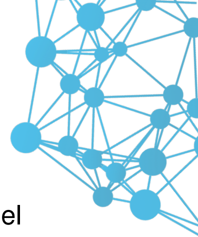
Structure

The directory structure is like in QuantLib

```
QuantExt / qle  / cashflows
                  currencies
                  indexes
                  instruments
                  math
                  methods
                  models
                  pricingengines
                  processes
                  quotes
                  termstructures
QuantExt / test /
```

Code Statistics

Library	Files	Lines of Code	Unit Test Cases
QuantLib	∼ 2400	360k	646
QuantExt	∼ 200	20k	36
OpenRiskEngineData	∼ 160	20k	20
OpenRiskEngineAnalytics	∼ 60	7k	21
Sum	∼ 420	47k	77



CrossAssetModel

QuantExt provides an implementation of a cross asset model

- multi-Gaussian IR-FX (-INF-CR-EQ-COM)[1]
- exact discretization of the underlying stochastic process for large step simulations
- utilizing Joshi’s Sobol Brownian bridge generator provided in QuantLib’s market model implementation
- analytic vanilla option engines for fast calibration
- extensible - other models can be plugged in (Heston, multifactor LGM, stochastic basis models, ...)

Test Suite

Extensive test suite, e.g. for the model part

- consistency with finite difference and Gaussian1D pricing engines in QuantLib
- recovery of analytical moments by Euler Monte Carlo
- martingale property of deflated payoffs
- repricing of calibration baskets with Monte Carlo

QuantLib as a Backbone for XVA Simulations

QuantLib 1.8 can be used for efficient XVA simulations

- no modifications in QuantLib necessary - this is fantastic
- but we use workarounds at some places, which are efficient in practice, but not clean
- in the following we derive proposals for future QuantLib development from this

Proposal #1 Floating Termstructures

We make extensive use of evaluation date shifts during simulation

- provide floating and fixed reference date term structures consistently throughout the library
- expose `TermStructure::moving_` to make fixed and floating term structures distinguishable during run time[2]
- add floating lags for NPV and settlement date parameters in pricing engines, for example and notably in the `DiscountingSwapEngine`
- provide fixed and floating bootstrap helpers

Quaternion
Risk Management

Proposal #2 Quotes

Quotes are the central tool to apply scenarios to term structures during simulation

- support quotes in `ExchangeRateManager`
- provide quote based constructors in term structures consistently

Proposal #3 Observability

Observability is used to propagate quote updates to term structures and instruments during simulation

- a naive use yields correct results, but may be slow
- deferral of notifications[3] does not seem to speed up our simulation or even slows it down in cases
- our workarounds are
  - disable Notifications and manually update term structures and instruments
  - unregister coupons from evaluation date observation
- goal: can we tape the notification graph on a small subset of simulation paths (or one path) and derive a minimal set of objects that needs to be updated from that?

Proposal #4 Simulated Fixings

During simulation, future fixings have to be generated and published

- required fixings are implicitly known from pricing on the original evaluation date.
- no global notification of all observers[4] necessary when adding a simulated fixing
- pathwise generation of future fixings and publishing them can be automated by an extension of `Index`
- no need for changes in pricing engines
- (almost) zero overhead when simulated fixings mode is disabled

Agenda

Open Source Risk Project

QuantLib Extension Library

Data Library

Analytics Library

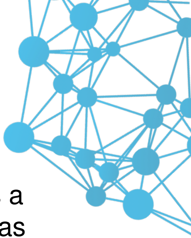
Data Library

- OpenRiskEngineData is a C++11 library that manages market and trade data
- Configured via API or XML (using RapidXML)
- Flexible curve bootstrap can be configured for Libor, OIS, XOIS, etc leaving the choice to users
- Curve configuration defined for all market curves (option surfaces/cubes) which maps to QL TermStructures
- Lightweight portfolio data model
- Again trade XML maps to QL Instruments

XML Example

```
<Trade id="123456">
  <TradeType>Swap</TradeType>
  <Envelope>
    <CounterParty>CP_A</CounterParty>
    <NettingSetId>CP_A_NS_1</NettingSetId>
  </Envelope>
  <SwapData>
    <LegData>
      <LegType>Fixed</LegType>
      <Payer>true</Payer>
      <Currency>EUR</Currency>
      <Notionals>
        <Notional>70000000</Notional>
      </Notionals>
      <DayCounter>30/360</DayCounter>
      <PaymentConvention>
        Following
      </PaymentConvention>
      <FixedLegData>
        <Rates>
          <Rate>0.035000</Rate>
        </Rates>
      </FixedLegData>
```

```
      <ScheduleData>
        <Rules>
          <StartDate>20120530</Sta
          <EndDate>20160704</EndDa
          <Tenor>1Y</Tenor>
          <Calendar>TARGET</Calend
          <Convention>
            Following
          </Convention>
          <TermConvention>
            Following
          </TermConvention>
          <Rule>Forward</Rule>
          <EndOfMonth/>
          <FirstDate>20120704</Fir
          <LastDate/>
        </Rules>
      </ScheduleData>
    </LegData>
  </SwapData>
</Trade>
```



Market and EngineFactory

- Interface `openriskengine::data::Market` defines a complete set of all the market instruments and curves (as Handles to QL objects) needed for pricing

-
  ```
  class Market {
    //...
    virtual Handle<YieldTermStructure> discountCurve(const string& ccy) = 0;
    virtual Handle<IborIndex> iborIndex(const string& indexName) = 0
    virtual Handle<Quote> fxSpot(const string& ccypair) = 0;
  }
  ```
- `TodaysMarket` implements this interface using curves bootstrapped as on previous slide
- `openriskengine::data::EngineFactory` takes a Market and generates QuantLib::PricingEngines for the portfolio (Actual engine choice and parameters are configurable via API/XML)
- TodaysMarket + EngineFactory + Portfolio = T0 pricing

Agenda

Open Source Risk Project

QuantLib Extension Library

Data Library

Analytics Library

Analytics Library

- OpenRiskEngineAnalytics is a smaller library built on top of QuantLib, QuantExt and OpenRiskEngineData.
- Provides a framework for Monte-Carlo simulation of future NPVs, aggregation and (in the future) market risk sensitivities.
- We use the following common definitions:
  - **DateGrid** a set of future dates we wish to calculate exposure on
  - **Cube** the 3-D matrix of trade NPVs for the portfolio on each path and each date in the date grid
  - **Scenario** A set of simulated market data points represented as a set of `QuantLib::Real` values
  - **ScenarioGenerator** A class that combines a model, date grid and PRNG to generate Scenarios.
- Scenarios can be generated by a CrossAssetModel, a Real world model or a set of defined sensitivities.

ScenarioSimMarket

- `analytics::ScenarioSimMarket` is a concrete implementation of the `data::Market` interface that is `Quote` based.
- The method `ScenarioSimMarket::update()` retrieves a Scenario from a ScenarioGenerator and updates the underlying Quotes.
- When a portfolio's `EngineFactory` uses this Market, then all Instruments will be directly linked to the Market's `TermStrutures` and `Quotes`
- Therefore, to price under a scenario we simply call `update()` and then loop over the portfolio calling `Instrument::NPV()`
- This relies heavily on QuantLib's Lazy Object and Observer patterns.

Loop Order

- To compute a Cube, we essentially have three nested loops
- Innermost loop is over portfolio ⇒ two options remain

- Option 1 - Outer Loop over Dates, then Paths

  Pro Minimise date changes - Rebuild static TermStructures at each date and so can use both floating or fixed reference dates.

  Con Need to cache scenarios, creates a memory constraint on the number of paths we can run

  Con Fixings are difficult to do properly

  Con Need to maintain state for path dependent trades

- Option 2 - Outer Loop over Path

  Pro Can price on a path and maintain fixings easily

  Pro Can stream scenarios, no memory constraints

  Con All TermStructures must have a floating reference date

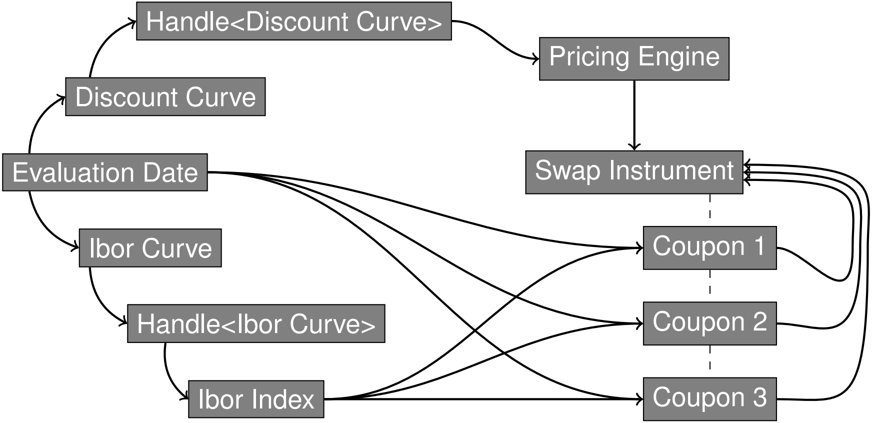
  Con Need to do multiple asof date changes, not cheap

Observations

- `Settings::evaluationDate` is observed a lot.
- Analysis from an early version of OpenRiskEngine:
  - 100 Fixed vs. Floating Swaps, Average maturity = 16.2 years
  - Total of 3,778 Floating Rate Coupons
  - Single call to `Settings::instance().evaluationDate() = d;` takes 1,500 microseconds.
  - 1,000 samples and 80 dates $\Rightarrow 120$ seconds.
  - Update time is all notification, does not change even at later grid dates when trades are expired.
  - `evaluationDate` is observed by 4,915 observers.
  - Total number of notifications is 34,848
  - Each notification is fast (we are doing 24 per microsecond).
  - However total number is massive (over 2.7 Billion)
  - Deepest chain is of depth 6

Observation - Single Swap

- There is a lot of overlap in the notification chain.
- Consider a simple 3 coupon swap.



- Swap is notified 7 times $(2n+1)$ of a change in the eval date

Observation - Solutions

All of the following solutions are available in OpenRiskEngine

1. Do nothing.
   - everything is working as designed and all values are correct
   - it can be slow
2. Minimise notification chain
   - Reduce the notification chain by careful selection or implementation of market data objects
   - remove duplication by unregistering connected observers with common observables (e.g. floating rate coupon and index)
3. Disable all notifications
   - Use `ObservableSettings::disableUpdates(false);`
   - Disable notifications and maintain a separate list of observers that require explicit notification
   - Notification still preformed, but the large chains do not kick in.
4. Defer all notifications
   - Use `ObservableSettings::disableUpdates(true);`
   - Defer notifications until all market quotes and fixings have been updated. This is generally slower than (1)!

Observation - Examples

- Swap exposure benchmark test runs a portfolio of vanilla swaps over 1,000 samples and 80 dates.
- Cube generation time in seconds

Mode	1 Swap	100 Swaps
None	12.64	320.99
Minimise	12.46	227.91
Disable	11.4	229.45
Defer	13.32	349.07

Quaternion
Risk Management

Firm locations and details

Quaternion™ Risk Management is based in four locations:

Ireland
54 Fitzwilliam Square,
Dublin 2,
Ireland,

+353 1 678 7922

London
29th floor
Canada Square,
Canary Wharf,
London E14 5DY.
+44 2077121645

USA
24th floor
World Financial Centre,
200 Vesey Street,
NY 10281-1004.
+1 646 952 8799

Germany
Maurenbrecherstrasse 16
47803 Krefeld,
Germany.

+49 2151 9284 800

quaternion.com

Quaternion™ is a sponsor of opensourcerisk.org along with Columbia University and Tullett Prebon Information

OPEN SOURCE RISK

COLUMBIA UNIVERSITY

tullett prebon information
DATA. IT'S IN OUR DNA.

DISCLAIMER:

This document is presented on the basis of information that is not publicly available. Quaternion™ is not liable for its contents. The presentation is for the named recipient only and is private, confidential and commercially sensitive.