COHERENT GLOBAL MARKET SIMULATIONS FOR COUNTERPARTY CREDIT RISK

CLAUDIO ALBANESE, TOUFIK BELLAJ, GUILLAUME GIMONET, GIACOMO PIETRONERO

Quantitative Strategies, Investment Banking Division, Credit Suisse Group, One Cabot Square, London, E14 4Q, UK

ABSTRACT. Valuing, hedging and securitizing counterparty credit risk involves analyzing large portfolios of netting sets over time horizons spanning decades. Theory dictates that the simulation measure should be coherent, i.e. arbitrage free. It should also be used consistently both to simulate and to value all instruments.

This article describes the Mathematics and the software architecture of a risk system that accomplishes this task. The usage pattern is based on an offline phase to calibrate and generate model libraries. Valuation and simulation algorithms are planned offline with portfolio specific optimizations. The interactive user-driven phase includes a coherent global market simulation taking a few minutes and a real time data exploration phase with response time below 10 seconds. Data exploration includes 3-dimensional risk visualization of portfolio loss distributions and sensitivities. It also includes risk resolution capability for outliers from the global portfolio level down to the single instrument level and hedge ratio optimization.

The network bottleneck is bypassed by using heterogeneous boards with acceleration. The memory bottleneck is avoided at the algorithmic level by adapting the mathematical framework to revolve around a handful of compute-bound algorithms.

1. Introduction

In the aftermath of the 2007 crisis, new issuance of credit structured products and fixed income derivatives slowed down quite suddenly. However, the crisis certainly didn't reduce the complexity of the valuation tasks at hand. Quite to the contrary, computational complexity in pricing theory has reached all time highs.

The traditional abstraction of pricing individual instruments in isolation revealed its limits. A qualitative shift occurred because of the renewed emphasis on counterparty credit risk (CCR) which motivates the development of pricing methodologies embedding in a consistent fashion global market information. A renewed theoretical research effort is thus needed as the staggering increase in algorithmic complexity prompts a reassessment of the mathematical formalism within the context of current computing technologies.

Date: October 20, 2010.

Email address of corresponding author: claudio@albanese.co.uk.

Acknowledgments: Research on this project was carried out at Credit Suisse. We would like to thank Anton Merlushkin and Steve White for valuable discussions. All remaining errors are our own.

Disclaimer: The opinions and views expressed in this paper are uniquely those of the authors, and do not necessarily represent those of Credit Suisse Group.

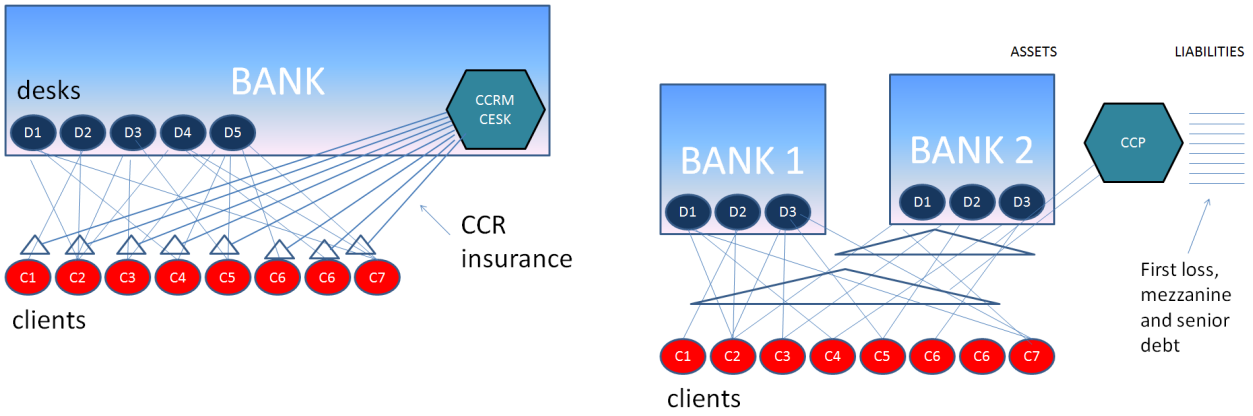


FIGURE 1. Flow charts for CCR management schemes, via an internal desk (left) or by securitization (right)

Business practices for CCR management have also evolved. Leading financial intermediaries aggregate CCR exposures in portfolios of insurance contracts covering the risk of default on derivative positions and hold them at a designated CCR desk. A typical portfolio entails thousands of netting sets. A netting set is a sub-portfolio of CCR insurance contracts sharing the same contractual netting agreement with a specific counterparty.

A netting agreement is a legal contract that regulates collateral posting obligations throughout the life of the referenced deals and liquidation procedures in case one of the parties defaults. It makes reference to two legal entities: a subsidiary of the portfolio holder and either a subsidiary of a counterparty entity or the entity itself. Different subsidiaries are characterized by different collateral allocation while all the subsidiaries corresponding to any given counterparty can safely be assumed to default in a synchronized manner.

An emerging pattern involves externalizing CCR desks into regulated entities acting as Central Counterparty Clearing Houses (CCPs) and operating in open markets. The CCPs would fund themselves by issuing several classes of debt, thus securitizing CCR insurance portfolios to facilitate risk transfer from commercial banks to several classes of investors by tranching. See figure 1 for an illustration.

The challenge is to value and hedge portfolios of netting sets by projecting out at regular time intervals scenarios over time horizons as long as the portfolio life itself, i.e. typically decades. Scenarios must be drawn with probabilities "consistent" with all available market information. Consistency means that scenarios for market factors and credit defaults ought to be generated under the very same measure which is used to value all instruments in the portfolio. The requirement for consistency descends from the Fundamental Theorem of Finance itself according to which the existence of a single unified pricing measure is equivalent to the condition of *global arbitrage freedom*.

We should clarify here the intended meaning of the expression "global arbitrage freedom" in the previous paragraph. The original de Finetti statement and proof of the Fundamental Theorem of Finance in [18] predates the introduction of the modern expression "arbitrage freedom" in the Finance jargon. (See also [7] for a derivation in modern language). In the original paper, arbitrage freedom is referred to as "coherence condition". "Coherence" and "arbitrage freedom" are thus synonyms: both mean that there is no self-financing investment strategy in the securities traded globally which would lead to a certain profit within any fixed time period.

The Black-Scholes-Merton papers [11] [32] re-derive the Fundamental Theorem in the special case of geometric Brownian motion by showing that this result follows from the existence of a dynamic replication strategy. The validity of the Fundamental Theorem does not rest on the possibility of dynamic replication and in fact holds true in the general case in which perfect replication is not possible. The Black-Scholes-Merton formulation however is fascinating as it comes with the promise of replicability and hedgeability of derivative portfolios.

The Black-Scholes-Merton model was then extended into an all-encompassing empirical methodology based on the notion of local valuation according to which portfolio positions should be priced with deal specific arbitrage free models admitting dynamic replication. Having done that, the methodology prescribes that portfolio level hedge ratios are empirically derived by summing over deal specific hedge ratios. This procedure is based on a "local" (i.e. deal specific) notion of arbitrage freedom. It is not rigorous and lacks of theoretical justification, but it has virtues from a practical standpoint. In fact, it fits very well with cluster computing schemes based on large grids of small 32-bit nodes where parallelism does not involve inter-node communication and is brokered transparently by middleware.

Local valuation was endorsed in VaR methodologies [33], banking regulation [10]. It is broadly reflected in derivative valuation schemes based on adjusters to "turn good prices into great prices" [23], some of which are even patented [37]. Hedging strategies in current use are also based on the use of numerous mutually inconsistent local models, see for instance [22].

The domain of applicability of local valuation is confined to short term derivative portfolios subject to one or very few risk factors. In fact, within short time horizons, dynamic models tend to become similar to each other and the impact of model risk is limited. However, model inconsistencies tend to amplify dynamically with the passage of time and lack of coherence becomes particularly worrisome for long-run dynamic portfolio simulations and risk analysis involving numerous factors and large portfolios. Since this is precisely the situation we encounter in the case of CCR management and securitization in which we are interested, local valuation is not a viable option in our case.

A second consideration that motivates us to insist on global valuation is related to technology. Our solution achieves high throughput portfolio processing on a single computational node, not in a grid environment with hundreds of small nodes. Hence local valuation does not present a computational advantage but would rather be a penalty. Using single node technology, we are not confined to trivial parallelism and can orchestrate complex multi-threading schemes. As we explain below, global models are favored as they allow for portfolio-level algorithmic optimizations which would otherwise not be possible. These optimizations are such that performance per instrument for the global market simulation grows with the portfolio size. By persisting in memory all detailed information collected during the simulation phase, one can then enable real time data exploration based on 3-d risk visualization and risk resolution schemes. On the basis of this risk analysis, hedging and replication strategies can be optimized at the portfolio level without having to resort to the theoretically dubious practice of aggregating sensitivities from a myriad of inconsistent models.

The paper is organized as follows. In Section 2, we discuss metrics for CCR exposures. In Section 3, we elaborate on the notion of algorithmic complexity in light of the latest developments in computer engineering. Section 4 outlines design patterns for financial software based on model libraries. Section 5 elaborates on the use of single precision floating point arithmetics, fast exponentiation and the central role played by the CFL condition. Model specifications for interest rates are outlined in 6, for foreign exchange rate models are reviewed in 7 and

credit models are in 9. Section 8 reviews quanto and stochastic interest rate corrections which are relevant when analyzing large multi-currency portfolios. Finally, Section 10 discusses a case study with performance benchmarks and 3-dimensional risk representations for a sizeable counterparty credit risk portfolio.

2. Metrics for Counterparty Credit Risk

Early attempts to tackle CCR valuation rest on the idea of reducing complexity by restricting the focus to the narrow class of additive risk measures given by expected loss. See [21], [15], [12], [14], [13], [16], [28], [31] and [36].

The "Credit Valuation Adjustment" (CVA) of a derivative position is defined as the discounted expected loss due to counterparty default risk. More precisely, the CVA of a portfolio of netting sets is

$$\text{CVA} = E_0\left[\sum_n \int_0^\infty e^{-\int_0^t r_s ds}(P_t^n)_+ d\pi_t^n\right] \tag{2.1}$$

where $(x)_+$ denotes $\max(x, 0)$, n is an index for netting sets, P_t^n is the price process of the portfolio corresponding to the netting set n, r_t is the process for the domestic short rate and π_t^n is the process followed by the cumulative probability of default up to time t for netting set n.

The CVA can thus be decomposed as follows:

$$\text{CVA} = \sum_n \text{CVA}_n \tag{2.2}$$

where

$$\text{CVA}_n = E_0\left[\int_0^\infty e^{-\int_0^t r_s ds}(P_t^n)_+ d\pi_t^n\right], \tag{2.3}$$

i.e. the CVA of a portfolio of netting sets is given by the sums of the CVAs of each individual netting set. Netting set specific CVAs also admit the following time decomposition:

$$\text{CVA}_n = \int_0^\infty \gamma_n(t)dt. \tag{2.4}$$

The function $\gamma_n(t)$ is the *term structure of point-in-time CVA* and is defined as follows

$$\gamma_n(t)dt = E_0\left[e^{-\int_0^t r_s ds}(P_t^n)_+ d\pi_t^n\right], \tag{2.5}$$

More informative decompositions can be formulated in terms of loss distributions

$$\text{CVA} = \int_0^\infty \int_0^\infty \Lambda(t, l)dtdl \tag{2.6}$$

where $\Lambda(t, l)$ is the density of the measure such that

$$\int_0^T \int_0^L \Lambda(t, l)dtdl = E_0\left[\int_0^T e^{-\int_0^t r_s ds} \min((P_t^n)_+, L)d\pi_t^n\right]. \tag{2.7}$$

The function $\Lambda(t, l)$ is called *term structure of loss distributions* and is our main focus of attention. Given the term structure of loss distributions, not only one can reconstruct the CVA of

a portfolio, but one can also derive more general risk metrics such as the *tranche point-in-time CVA* defined as follows:

$$\gamma(t, l_0, l_1)dt = \int_{l_0}^{l_1} \Lambda(t, l)dtdl \tag{2.8}$$

where $l_0 \geq 0$ and $l_1 > l_0$ are the tranche attachment and detachment points, respectively.

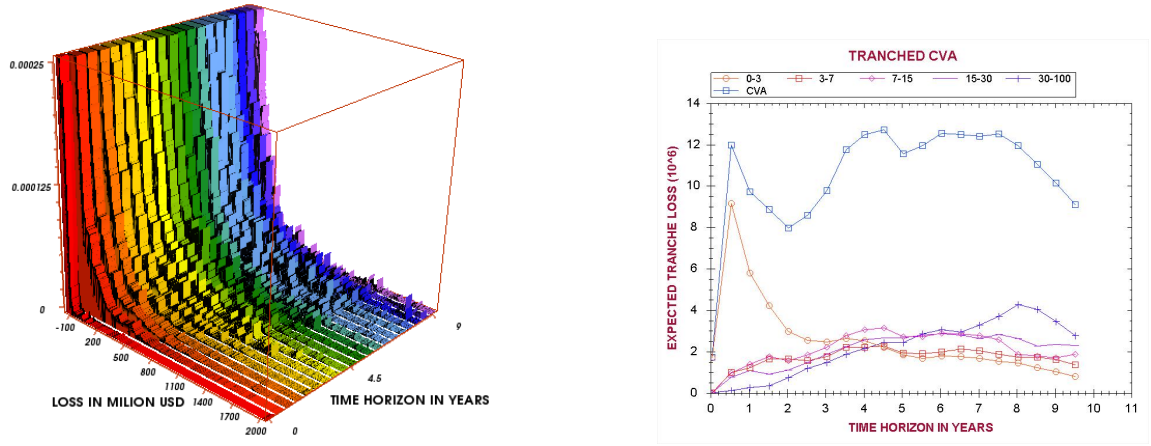


FIGURE 2. Portfolio loss distribution and tranched point-in-time CVA for a portfolio of 302 netting sets. Tranches are given as percentages of expected exposures.

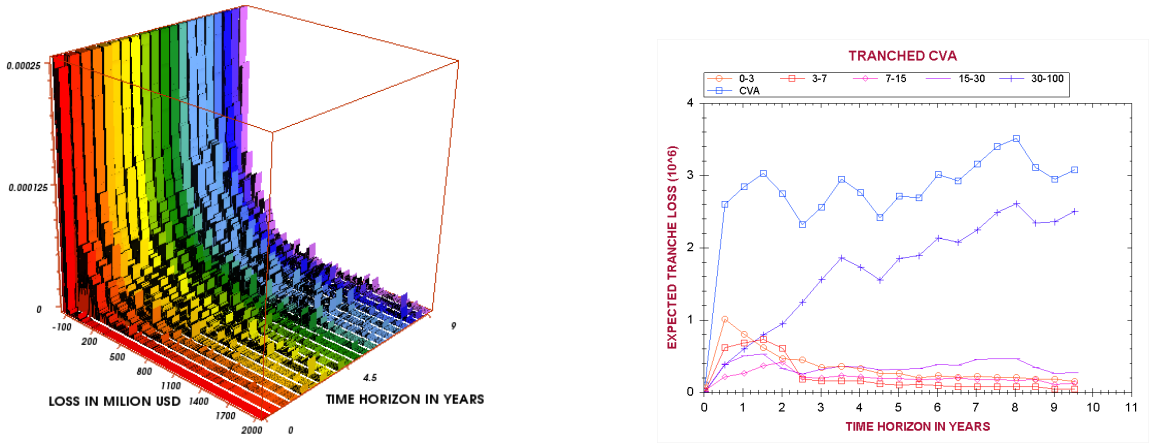


FIGURE 3. Portfolio loss distribution and tranched point-in-time CVA for a portfolio of 62 netting sets in the financial sector. Tranches are given as percentages of expected exposures.

See Figure 2 for examples of loss distributions for a portfolio of 302 netting sets in the case study example discussed in Section 10. By comparison, Figure 3 shows the graphs relative to the smaller and less diversified sub-portfolio containing only the 62 netting sets corresponding to counterparties in the financial sector. Notice how the lesser degree of diversification is reflected in a much higher relative value for the tranche CVA corresponding to the range above 30% of expected loss. This is an indication that, if losses were to occur in this sector, they would appear as more highly clustered than in the global portfolio.

Neither loss distributions nor tranched CVAs are additive across netting sets. Hence they need to be computed while processing the portfolio of netting sets as a whole. The plain CVA instead can be evaluated by firstly carrying out the calculation independently for each netting set and then aggregating results by taking sums over netting sets. Additivity of the plain CVA descends from the fact that this measure is independent of default correlation. Thanks to additivity, the plain CVA can be evaluated using grid computing as the task is trivially parallelizable. Instead, the calculation of loss distributions and other non-additive risk metrics necessitates compute boards which are sufficiently powerful and capable to process global portfolios in their entirety without partitioning them and to support high-throughput data exploration and visualization.

Default arrival times of counterparties have a tendency to cluster during crisis periods. This clustering is perhaps the single most important risk factor that can potentially deplete the economic capital provisioned to CCR portfolios and is not reflected in a measure of expected loss. Additive risk metrics such as the CVA may appear conservative but are neither prudent nor very useful and in fact give a distorted representation of risk as they neglect credit correlations. In our opinion, effective hedging requires a process of data exploration and risk resolution of the outliers that ought to be based on the visualization of the full loss distribution.

Notwithstanding all its limitations, an agreement was reached between market participants and regulators to use the CVA as a metric to set capital adequacy requirements. The sole justification behind this choice is to lower the algorithmic complexity down to a level at which it could be handled with the traditional design patterns involving a combination of stochastic calculus, grids and middleware. Had market participants reviewed the full potential of new technologies, perhaps they would have considered other options.

In addition to being a poor risk measure, the CVA is sometimes valued using uncontrollable empirical approximations such as the following ones:

(i) Sometimes the CVA is assessed at the individual instrument level and added across positions, although this measure is only sub-additive and not additive unless it is applied at the level of netting sets.

(ii) CVA valuation at the netting set level is sometimes not coherent, i.e. not truly arbitrage free, because individual instruments are priced using locally calibrated models with instrument specific parameters.

(iii) It is a common practice to infer the CVA out of expected positive exposure (EPE) vectors over future time periods assuming zero correlation between credit and market risk factors, i.e. to use the following approximation:

$$\mathrm{CVA}_n \approx \int_0^\infty E_0\left[e^{-\int_0^t r_s ds}(P_t^n)_+\right] E_0\left[d\pi_t^n\right], \tag{2.9}$$

This assumption neglects the bias toward "wrong way risk" in intermediaries' portfolios, due to the tendency to deal in maturity transformation, i.e. to provide cash upfront to counterparties in exchange for promises of future payments discounted for credit risk.

(iv) Scenario generation under a measure very different from the one used for valuation is theoretically inconsistent for pricing purposes. For risk management purposes, this choice would be admissible only if the market price of risk was carefully and parsimoniously estimated, which is often not the case.

The situation is further complicated by the regulatory and accounting requirement that market participants also assess the "debt value adjustment" (DVA) defined as the putative "expected gain" due to the default of the portfolio holder himself. The DVA is often treated as additive

across netting sets although it is not: in case of default of the portfolio holder, assets and liabilities across all counterparties will be netted by the appointed liquidator. Nevertheless, the common practice is to aggregate the DVA linearly across netting sets. Assuming but certainly not granting that DVA adjustments are justified and useful, aggregating DVAs linearly across netting sets is devoid of economic meaning and can induce wrong behaviours.

The approach we advocate here is to capture credit correlations by working directly at the aggregate level of portfolios of netting sets. We don't place primary emphasis on the CVA but instead evaluate 3-dimensional risk metrics such as fully detailed loss distributions. Emerging computing technologies, if correctly interpreted, are quite sufficient for this purpose, even within the confines imposed by real time usability for visualization and exploratory data analysis.

3. Silicon Economics and Algorithmic Complexity

Traditionally, algorithms could be understood by means of a Turing machine abstraction, [38], i.e. imagining that computational task are carried out as a sequence of bitwise operations. In this context, algorithmic complexity is a function of the number of bitwise operations the arithmetic logic unit (ALU) needs to perform in order to complete the calculation.

Real computing machines don't operate on isolated bits but on either single precision or double precision floating point numbers or on integers of various size. Types do matter and affect performance. But more importantly, real machines have complex physical architectures which greatly impact performance. The complexities of modern hardware imply that the traditional notion of algorithmic complexity defined as the number of bitwise operations to execute in order to accomplish a given task can be quite misleading. In fact, the sort of algorithms we find optimal necessitate a massively greater number of floating point operations.

To explain how greater performance can be achieved by doing more calculations, let us recall a few traits of the silicon economics affecting microchip and board designs. In recent times, there was in fact a radical shift in this landscape.

It used to be that:

(i) Computing capabilities were limited by the ability of ALUs to execute floating point and integer arithmetics
(ii) Memory was expensive and a scarce resource
(iii) Most algorithms were single-threaded and parallelism was best brokered transparently by middleware layers dispatching jobs to large grid farms
(iv) Code was best written in native C++ optimized in such a way to speed up the execution of a great variety of bespoke algorithms.

Although these practices are still widespread in the transition period we are living, the underling technology has now shifted quite radically.

(a) Nowadays, it is relatively cheap to populate microchips with highly capable ALU cores. The 8-socket CPU boards of the emerging generation entail as many as 80 cores capable of hyperthreading in the case of Intel or 96 cores in the case of AMD. Even more extreme ALU counts are seen in the GPU space where the AMD Firepro GPUs have 1600 cores and nVidia Fermis have 512.
(b) Memory is relatively cheap and readily available up to terabyte scale, thus enabling single node technology for portfolio processing as a viable alternative to grid computing.

(c) The clock frequency and bandwidths of data paths are not keeping pace with the compute power of ALUs and the massive memory available, rendering the memory bottleneck tighter than ever within the bounds of cost effective designs.

(d) Vastly different microchip architectures have emerged, including SIMD multiprocessors with up to 16-32 data registers located in discrete GPU parts as in the nVidia Fermi and ATI Firestream, the multicore MIMD designs on CPU boards by Intel and AMD and the emerging MIMD-SIMD hybrid fusion architectures, the Intel Sandybridge and AMD Booldozer.

(e) MIMD and SIMD designs are characterized by radically different threading models: SSE2/SSE3/AVX primitives rule with CPUs while the lightweight, no-frills threading models in CUDA/OpenCL are used for GPUs.

(f) Cache hierarchies for MIMD architectures are complex and involve up to 2 MB per core. GPUs instead are nearly cacheless except for a modest amount of shared memory located on individual SIMD microprocessors.

(f) On the programming language side we see the merit a bifurcation away from catch-all C++ coding. On the one hand, the variety of architectures motivates a revival of interest in low-level optimization of basic building block algorithms. On the other hand, the complexity of multi-threaded orchestration in shared memory designs using large scale in-memory processing motivates the use of higher level languages. Features such as garbage collection, managed thread pools and support for service oriented architectures are in fact essential for complexity management.

Mathematical Finance is founded upon frameworks of Applied Mathematics and Probability Theory which have roots in the pre-computer era when floating point calculations were carried out manually. Keeping the number of floating point operations to a minimum was the leading guiding principle underlying the invention of those techniques based on analytic closed form solutions and sparse matrix methods which lie at the heart of virtually all named valuation models for derivatives. The shift in cost structure for board design has now tightened the memory bottleneck to a point that it actually throws off balance the ecosystem of numerical algorithms and renders closed form valuations and sparse matrix methods highly inefficient.

To take the innovations in computer engineering to fruition one needs to refresh the formalism of Mathematical Finance, starting from the meta-mathematical question of defining relevance of mathematical inventions in terms of hardware performance. The traditional definition of algorithmic complexity measured in terms of flop count is valid in a world where algorithms are compute bound, i.e. the bottleneck lies in the ability of the ALU to carry out floating point operations. Until a decade ago, this used to be the case for most algorithms. However, with the turn of the third millennium, the relative cost structure and performance characteristics of computer parts have diverged at exponential rate. ALUs and memory have become very cheap and easily available in massive quantities, while the data paths connecting the two has became much costlier in relative terms.

In the changed ecosystem of computer engineering where the memory bottleneck is tighter than ever, the objective of optimizing execution time can be reached by architecting software solutions around the precious few algorithms which are known to admit compute bound implementations. The ones of particular relevance to our application are of four types:

- Full matrix-matrix multiplication and tensor variations thereof (the standard Level 3 and the Level 4 BLAS extensions in [7]). To understand why these can be made compute bound, consider a square matrix of dimension N has N^2 elements and requires $O(N^3)$

operations to be squared: by choosing N large enough, the data to flops ratio can be arbitrarily reduced.
- Discrete convolutions of a generic function times a function supported in two points.
- Discrete Fourier transforms.
- Carefully orchestrated Monte Carlo scenario generation schemes based on look up tables which can be persisted in third level cache long enough to marginalize memory access times while also avoiding cache snooping.

As we explain in [7], we find that the SIMD microprocessors in GPUs are ideally suited to perform linear algebra, convolution and Fourier transform logic. MIMD cores in CPUs excel at implementing the threading schemes needed for Monte Carlo scenario generation as they are endowed with large caches and are not slowed down by asynchronous branching.

4. A System Architecture based on Model Libraries

Modeling flexibility has paramount importance in applications of Mathematics. The traditional approach has been to design models around techniques to solve them. Found a hammer, a mathematician would go look for a nail to bang. As Jacobi stated in his 1847 lectures [27]:

> "The main difficulty in integrating a given differential equation lies in introducing convenient variables, which there is no rule for finding. Therefore we must travel the reverse path and after finding some notable substitution, look for problems to which it can be successfully applied."

This is the traditional approach of Mathematics aimed at solving complex problems by carrying out the few operations required to evaluate special functions. Named valuation models in current use have been built around the same principles, by adapting well known techniques from 19th century Physics literature. However efficient for manual calculations and on last century's hardware, this approach gives rise to severe limitations on the modeling side. Hardy expresses this concept quite bluntly in the short essay *A Mathematician's Apology* [24]:

> "Most of the finest products of an applied mathematician's fancy must be rejected, as soon as they have been created, by the brutal but sufficient reason that they do not fit the facts."

The availability of GPUs as powerful matrix engines changes all this and redefines financial modeling as GPU computing finally allows one bypass the need of using analytical shortcuts. Since generic Markov processes can now be effectively solved numerically, there is neither the need nor the justification for using named models built around ingenious mathematical shortcuts to exploit analytic solutions.

The motivation for using sparse matrix algebra is also greatly diminished as these algorithms tend to be memory bound as the have a high data/flop count ratio. As we explain in the next section 5, sparse matrix methods are typically used to implement unconditionally marginally stable schemes which are less robust and less flexible from the modeling viewpoint.

This shift has far reaching repercussions on several levels. We were led to the design of a software architecture for valuation and portfolio simulation based on libraries of global models that are calibrated offline. A global market model is built by correlating dynamically marginal processes for individual risk factors. Whenever one is tasked with a given portfolio to value, be it a single swap or a large portfolio of netting sets, the valuation engine queries the model library for the appropriate marginal specification and correlation dynamics. The valuation engine also queries the portfolio database for pre-processed contractual information and cash

flow generators. Then assembles these two sources of information and executes a pricing function, combining multi-pass backward induction with simple or nested Monte Carlo simulations.

The main traits of our system can be summarized as follows (see also Figure 4):

(i) **Realistic marginal distributions**: Models for marginal single factor processes can be specified on the basis of the study of econometric time series aimed at achieving a parsimonious and economically realistic representation. This is possible because the constraint of analytic solvability has become irrelevant.

(ii) **Aggregate portfolio pricing**: The most effective strategy to parallelize portfolio pricing is to organize valuation tables into matrices and distribute the load according to complex multi-threading patterns. This is contrary to the widespread strategy of valuing assets one by one in isolation on grid computing equipment.

(iii) **Global calibration layer**: In order to take advantage of the ability to formulate econometrically realistic models and to achieve a coherent specification of a global market simulation engine, models need to be of high quality and be calibrated against a large number of liquid derivatives comprising hundreds of data points. (A technical problem in this respect is that the very large number of targets gives rise to high frequency disturbances and lack of smoothness in the objective function. We tackled this difficulty by developing a new optimization algorithm called "rotating frames" that is documented in [4]).

(iv) **Bidirectional character:** As explained in Section 5, conditionally strongly stable methods such as fast exponentiation on GPUs yield an unprecedented degree of smoothness in kernel calculations. Since kernels are now directly available, it is possible to use models consistently in both time directions for forward induction, backward induction and Monte Carlo simulations. Named models instead privilege one temporal direction over the other: some are best solved backwards and others are best used for forward Monte Carlo simulations.

(v) **Global Monte Carlo simulations:** multi-factor correlations are modeled most naturally by means of dynamic copulas. For performance reasons, it is most efficient to simulate all market factors at once on a single high density CPU node on the basis of kernels and valuation tables produced by a GPU matrix factory.

(vi) **Correlation modeling:** In a global market simulation, correlation strengths don't have to be necessarily stationary but may and should depend on the realized scenarios for collective market behavior.

(vii) **FX polygonal relationships:** Correlations between foreign exchange crosses are often highly non-stationary because of the triangular and more generally polygonal relationships tieing up the various crosses with each other. It is thus essential to select a proper sufficient basis of anchor currencies as a reference frame to maximize stationarity, see also Section 7.

(viii) **Unified valuation and simulation engine:** One of the benefits of not having to rely on analytic shortcuts is that software implementations for financial models can have a high degree of polymorphism. The basic algorithms of forward induction, backward induction and scenario generation can and should be implemented just once in a highly efficient manner and applied to all market factors across all asset classes. The only bit of information that a unified valuation and simulation engine needs to be aware of is a boolean flag indicating wether the model dynamics refers to an interest rate or to some other factor, as there is an impact on discounting rules.

(ix) **Instrument capture:** contractual specifications need to be captured and encoded as payoff generation objects which are not only containers of instrument attribute lists but also embed methods giving a full description of risk factor dependencies and conditional payoffs. These objects provide thus a semantically isomorphic representation of contractual information. Most importantly, these objects should also not involve any valuation logic. Finally, the valuation engine should be designed to interface with generic specifications of payoff logic. These principles were first pioneered in [29] a decade ago and can now come to fruition to the full spectrum of valuation tasks thanks to the technical feasibility of payoff agnostic valuation engines.

(x) **Data-bound pre-processing:** to achieve real time performance, portfolio parsing tasks and all other memory-bound pre-processing chores must be attached to the data itself and executed offline.

(xi) **High throughput data-processing:** While the input data channels used for calibration need to comprise as much data as possible, the output data bandwidth must yield the most detailed risk information and allow for real time exploratory data analysis. For counterparty risk applications, this means having loss distributions available dynamically for large portfolios of netting sets and all sub-portfolios thereof, along with sensitivities.

(xii) **In-memory visualization database** To prepare for real time data exploration, it is useful to envisage a user driven simulation stage executing in a 3-10 minutes time frame which persists fine grained simulation information into an in-memory database, out of which one can then reconstruct an answer to all the questions of interest.

(xii) **Real time 3-d visualization and data exploration layers:** Having formed an in-memory database, a user-driven risk analytics layer should execute with a 1-10 seconds response time and allow for exploratory analysis of risk profiles. The user would likely want to perform the analysis in cycles, starting from a global portfolio view, identifying features and outliers and then resolve them by drilling down to the lower levels of sub-portolios of netting sets, individual netting sets and individual deals. In addition, the user will want the ability to incrementally add positions interactively, experiment with definitions for tranche attachment and detachment points to gauge the impact on equilibrium tranche spreads, query for risk sensitivities and hedging information, etc...

5. Fast Exponentiation, the CFL Condition and Single Precision Arithmetics

To bring to fruition the emerging computing technologies where full matrix algebra is a privileged algorithm, we make use of a formulation of Mathematical Finance based on operator methods. The intent of this framework is to make use of matrix-multiplication engines to numerically evaluate transition probability kernels as large matrices and then obtain all quantities of financial interest out of multiplying, differentiating, Fourier transforming and taking tensor products of these matrices. In this section, we give a brief description of the key ideas, referring the reader to [7] and references therein for more details.

To model the evolution of a risk factor, its values are associated to a state variable $x = 0, ..d-1$ by means of a (possibly time dependent) map. We find that the most convenient values for the total number of state variables d are the multiples of 64 between 512 and 1280: the divisibility by 64 maximizes efficiency of memory read-write transactions, the upper bound at 1280 is useful not to exceed GPU memory and the lower bound of 512 is necessary to have a sufficient resolution.

To describe a process, we give a matrix $u_{\delta t}(x, y; t)$ whose elements are interpreted as the transition probabilities to evolve from the state variable x to the state variable y at any given

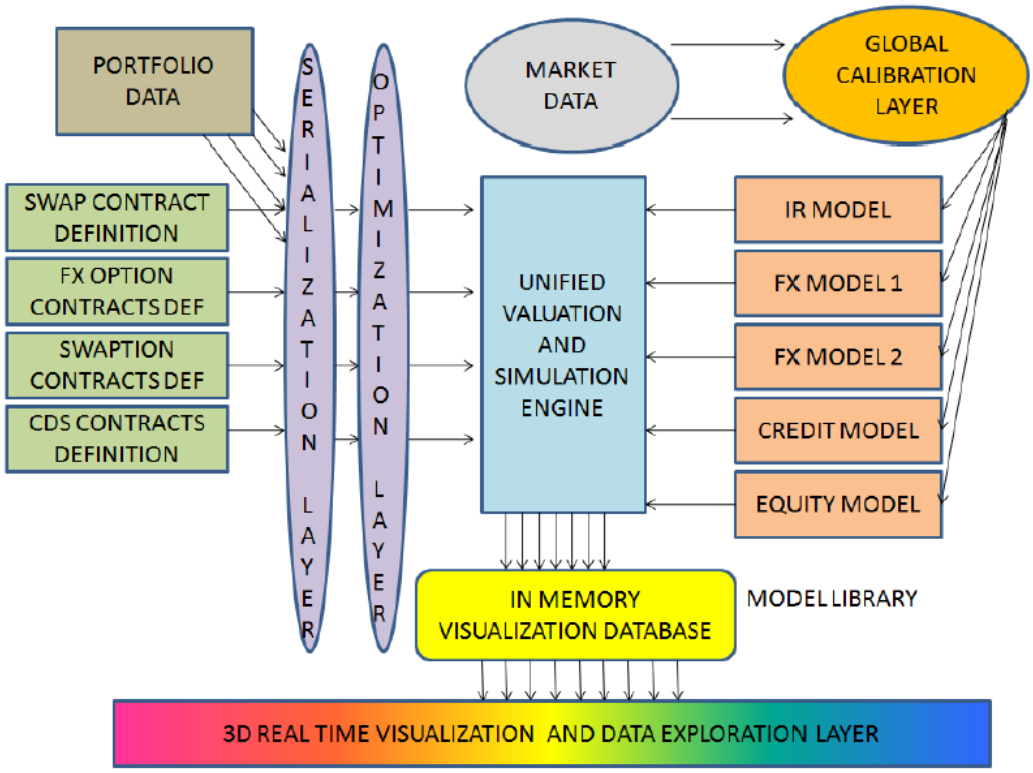


FIGURE 4. Flow chart describing the general architecture.

date t and over a fixed but small time interval δt. In practical applications, δt is equal to one day for interest rate models and a fraction thereof such as 12 or 6 hours for foreign exchange models. This matrix is called *elementary transition probability kernel*. To be valid, an elementary transition probability kernel must satisfy the following two properties:

(i) **Positivity:** $u_{\delta t}(x, y; t) \geq 0$,
(ii) **Probability conservation:** $\sum_y u_{\delta t}(x, y; t) = 1 \quad \forall x$.

The first property states that transition probabilities are positive, the second that they add up to 1. A convenient way of expressing an elementary transition probability kernel is via a Markovian matrix $\mathcal{L}(x, y; t)$ such that

$$u_{\delta t}(x, y; t) = \delta_{xy} + \delta t \mathcal{L}(x, y; t). \tag{5.1}$$

In order for the representation to be acceptable, the following three conditions need to be satisfied for all t:

(i') **Positivity:** $\mathcal{L}(x, y; t) \geq 0$, for all $x \neq y$
(ii') **Probability conservation:** $\sum_y \mathcal{L}(x, y; t) = 0 \quad$ for all x
(iii') **CFL condition:** $\delta t \leq \frac{1}{\max_x |\mathcal{L}(x,x;t)|}$

The third condition is named after Courant, Friedrichs and Lewy as they first proposed it in [17].

To find the kernel over time horizons longer than δt, one can make use of matrix multiplications. In fact, the law of compounded probability indicates that the transition probability

kernel over the double length interval $2\delta t$ is given by

$$u_{2\delta t}(x,y) = \sum_z u_{\delta t}(x,z)u_{\delta t}(z,y). \tag{5.2}$$

Notice that this rule amounts to the rule of matrix multiplication. Iterating this equation, we then arrive at kernels over time steps of length $4\delta t$, $8\delta t$, etc.... In formulas, we iterate as follows:

$$u_{2\delta t} = u_{\delta t}^2, \;\; u_{4\delta t} = u_{2\delta t}^2, \;\; ..... \;\; u_{2^n\delta t} = u_{2^{n-1}\delta t}^2. \tag{5.3}$$

Fast exponentiation uses repeated squaring and has ancient roots. Egyptians used extensively repeated doublings as they carried out arithmetic calculations in a basis 2 representation for rational numbers and then translated into basis 10 hieroglyphics, [1]. In the 10-th century, al-Biruni succeeded in the feat of expressing 2^{64} in basis 10 by using repeated squarings. Nowadays, fast modular exponentiation algorithms are used in public-key cryptosystems, [35], while the use for probabilistic modeling that we advocate is still not widely adopted.

When using fast exponentiation, the length of the elementary time interval δt can be chosen very small. In fact, halving this interval has only the marginal impact of adding one more step in the iteration. It turns out that there is a critical threshold given by the so called Courant condition [17] such that whenever the elementary time interval is below that threshold, the resulting long step kernels are very smooth.

Smoothness is of great practical importance because operator methods are all about manipulating kernel matrices: if these operations were affected by a high level of noise deriving from floating point errors, errors would propagate exponentially fast.

Very remarkably, this is not the case: the roundoff errors one inevitably incurs at each step largely compensate against each other, the positive errors offsetting the negative errors, thus cleaning off the signal to a surprising degree.

The mathematical mechanism is subtle. In [2], Albanese attempts a rigorous explanation by considering the toy model of one-dimensional diffusion processes of Markovian

$$\mathcal{L} = \mu(x)\nabla_h + \frac{\sigma(x)^2}{2}\Delta_h \tag{5.4}$$

where ∇_h and Δ_h are the discrete differentiation operators of first and second order, respectiviley, on the lattice $h\mathbb{Z}$. Assuming that the coefficients $\mu(x)$ and $\sigma(x)$ are Hölder continuous and under minor technical assumptions, the conclusion is that transition probability kernels $u_h(x,y;T)$ converge in the limit $h \to 0$ so that

$$\sup_x \sup_y |(u_h - u_0)(x,y;T)| + |(\mathcal{L}u_h - u_0)(x,y;T)| \leq ch^{\min(2,\alpha)}. \tag{5.5}$$

The result also admits a version for joint transition probability kernels between a one-dimensional stochastic process and a stochastic integral over it, see [3].

The proof is based on a technique from Mathematical Physics called "renormalization group". In this toy model for floating point errors, the CFL condition guarantees that kernels converge pointwise along with first and second order derivatives, notwithstanding the lack of smoothness in the coefficients.

Without the CFL condition, convergence estimates are much weaker. Until a decade ago, the CFL condition could not be enforced because of limitations in technology and the inability to multiply matrices. Hence there has traditionally been a focus on so called "unconditionally stable" methods, such as the Crank-Nicolson method, see for instance [34]. In this case, one can

only show that, assuming the coefficients $\mu(x)$ and $\sigma(x)$ are smooth functions and if $\phi \in L^2$ is a test function, then

$$\left(\sum_{x \in h\mathbb{Z}} \left(\sum_{y \in h\mathbb{Z}} u_h(x,y)\phi(y)h - \int dy u_0(x,y)\phi(y) \right)^2 \right)^{1/2} \leq ch^c. \tag{5.6}$$

If ϕ has two derivatives or more, then $c = 2$. If ϕ is smooth except for a jump discontinuity $c = \frac{1}{2}$. Otherwise in general one can only claim that $c > 0$.

For comparison, in the case of 5.5 instead, coefficients are just Hölder continuous as opposed to being smooth, the test function ϕ is allowed to contain singularities as hard as that of a delta function along with its first and second derivatives as opposed to be in L^2 and the estimate is in the uniform norm, which is much stronger than the L^2 norm in 5.6. The observed qualitative jump in terms of accuracy is so great that the kind of smoothness and robustness we can achieve nowadays in single precision surpasses greatly what could be achieved using Crank-Nicholson methods in double precision.

The use of single precision also has major consequences on the technology side as GPU chipsets have grown out of the graphics market and are thus based primarily on 32-bit floating point arithmetic engines. Combining two 32-bit ALUs to make a 64-bit ALU does not require a huge effort. However, data paths to feed the ALUs are expensive and dictate that the optimal performance ratio between single and double precision is in excess of a factor of two. The highest single precision performance nowadays is achieved on AMD microchips like the Firepro V9800 which offer a performance in matrix multiplication of 1450 GF/sec with OpenCL code and a performance ratio of 5:1 between single and double precision. As we make a transition to hardware platforms where single precision floating point arithmetics is at such great advantage, robust numerical methods such as fast exponentiation come to play a privileged role.

6. Interest rate models

To model interest rate processes, we use a stochastic monetary policy short rate model. In this model, the short rate is confined to be positive and has a stochastic mean reversion level. To use stochastic calculus notations, the process could be loosely described by means of a "pseudo-equation" of the form

$$d\rho_t = \mu_{a_t} dt + \kappa(t)(\theta(t) - \rho_t)dt + \sigma(t) s_{a_t} \rho^{\beta(t)} dW + \text{jumps}, \tag{6.1}$$

The short rate process itself is an affine transformation of the process ρ_t given by

$$r_t = \lambda(t)\rho_t + \phi(t). \tag{6.2}$$

The function $\lambda(t)$ depends on time deterministically, is assumed positive and is calculated in such a way to fit the term structure of rates precisely at a discrete set of time points in the future. The function $\phi(t)$ is then added in such a way to achieve a daily fit with any given input discount curve. The regime variable a_t denotes monetary policy and has the effect of shifting the mean reversion level for rates, thus affecting the steepness of the curve. Jumps are added to ensure that whenever there is a change in monetary policy there is also a sizeable change in the short rate.

The short rate process depends on parameters which are a function of the monetary regime variable and of time. Parameters are assumed to be piecewise constant and are allowed to

change only at preset time points $t_i, i = 0, ...N$. We also restrict variability to be of exponential type, i.e.

$$s(t_i) = s_\infty + (s_0 - s_\infty)e^{-\gamma t_i}. \tag{6.3}$$

This restricts the flexibility of the optimizer and avoids irregular time dependencies. Also the other functions in 6.1, i.e. $\kappa(t_i), \theta(t_i)$ etc., have the same exponential form.

The pseudo-equation in 6.1 should not be interpreted literally as in fact it is discretized on a lattice. To precisely reflect our model specification, we need to use operator notations and express the dynamics in terms of a Markovian generator. The generator in the period $[t_i, t_{i+1})$ is

$$\begin{aligned}\mathcal{L}_i(x, a; x', a') = &\mathcal{L}_i^d(x, x'; a)\delta_{a,a'} \\ &+ \mathcal{L}_i^{j+}(x, x')\delta(\min(a + 1, n_a - 1), a') \\ &+ \mathcal{L}_i^{j-}(x, x')\delta(\max(a - 1, 0), a') + c(x, a)\delta_{x,x'}\delta_{aa'}\end{aligned} \tag{6.4}$$

where

$$\mathcal{L}_i^d(x, x'; a) = (\mu(a) + \kappa(t_i)(\theta(t_i) - \rho(x)))\nabla_x(x, x') + \frac{1}{2}\sigma(t_i)^2 s(a)^2 \rho(x)^\beta \Delta_x(x, x'). \tag{6.5}$$

Furthermore,

$$\mathcal{L}_i^{j+}(x, x'; a) = \exp(-(\rho(x') - \rho(x))/\rho_0(t_i))\delta(x' > x) \tag{6.6}$$

and

$$\mathcal{L}_i^{j-}(x, x'; a) = \exp(-(\rho(x) - \rho(x'))/\rho_0(t_i))\delta(x' < x). \tag{6.7}$$

Finally, $c(x, a)\delta_{x,x'}\delta_{aa'}$ is a diagonal term added in such a way to ensure probability conservation.

To avoid unreasonable local minima, the optimization algorithm is first run assuming that all time dependent functions are constant. Exponential time dependencies are then allowed in a second phase of the optimization procedure.

This model calibrates well to swaptions, typically to within errors below 3 basis points in normal volatility. Bermuda swaptions can be added directly to the calibration basket as they are only marginally more complex to price than their European counterparts. Moreover, regime switching confers volatility to rate spreads, introducing a rotation mode that leads to steepening or flattening of the yield curve. This feature makes it possible to add also CMS spread options to the calibration basket.

Globally calibrated stochastic monetary policy models are of high quality and can be used for model validation. Figure 5 shows an example of vetting exercise whereby a real portfolio of CMS spread options is valued with three methods: the Hull-White model with adjusters which is computationally fast but relatively imprecise, a 50-factor BGM model which is slow but of high quality and our stochastic monetary policy model. Regarding performance, the stochastic drift model is much faster than the Hull-White model and (assuming that the globally calibrated model has already been obtained) succeeds to valuing a real portfolio of 930 CMS spreads in 25 seconds as opposed to over an hour time. The BGM model has execution times ranging from 2 to 5 minutes per instruments and thus requires overnight execution times or grid deployment. When valuations are compared, the stochastic drift model agrees with the BGM model quite remarkably, with over 90% of the instruments within two standard deviations of the BGM simulation noise. Also, on an aggregate portfolio basis, relative errors for Net Asset Value (NAV) are below 2 basis points. The differences with the Hull-White model with adjusters

FIGURE 5. Scatter plots of relative price differences for a portfolio of callable CMS spread options as a function of the option strike. The date is October 30th 2008 and options are mostly JPY denominated. The left figure shows differences between our stochastic drift model and the Hull-White model with adjusters and the right figure shows differences with a high quality 50-factor BGM model on the same scale.

instead are quite substantial and, in the particular case at hand, show a visible skew bias as a function of option strikes.

7. Foreign exchange rate models

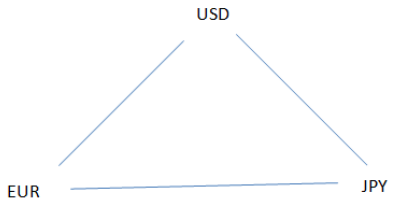
We make use of two types of FX models: local and global. Global models are used for simulation and for exact valuation. Local models are used for approximate valuation and variance reduction.

Local models are defined for each of the crosses that are referred to in our CCR portfolio. They are used to create approximate valuation tables for FX options to use as control variates for the purpose of variance reduction. Although these local models are mutually inconsistent with each other, the approximation is useful and can be controlled, i.e. errors can be removed by running a nested Monte Carlo simulations based purely on the global model. These are more precise but also more time consuming.

Global models are designed to be consistent with both the stochastic interest rate processes and with each other by obeying polygonal constraints.

One example of polygonal constraint is the following triangular relation

$$\frac{USD}{EUR}\frac{EUR}{JPY} = \frac{USD}{JPY}. \tag{7.1}$$



More generally, if one considers a graph having world currencies at each vertex and where the links represent the stochastic processes for crosses, there is a similar polygonal relation for each cycle in the graph. If one models only the crosses in a subgraph corresponding to a minimal spanning tree, by chaining these models and accounting for stochastic interest rates, one obtains a global model for all other crosses as well.

The question then arises: what is the best minimum spanning tree? Our answer is to identify the spanning tree which minimizes the volatility of each cross that is contemplated. This consideration led us to the graph in 6. Here, we added an extra vertex designated with the symbol SDR that denotes the IMF global currency but should be interpreted as a modeling device to

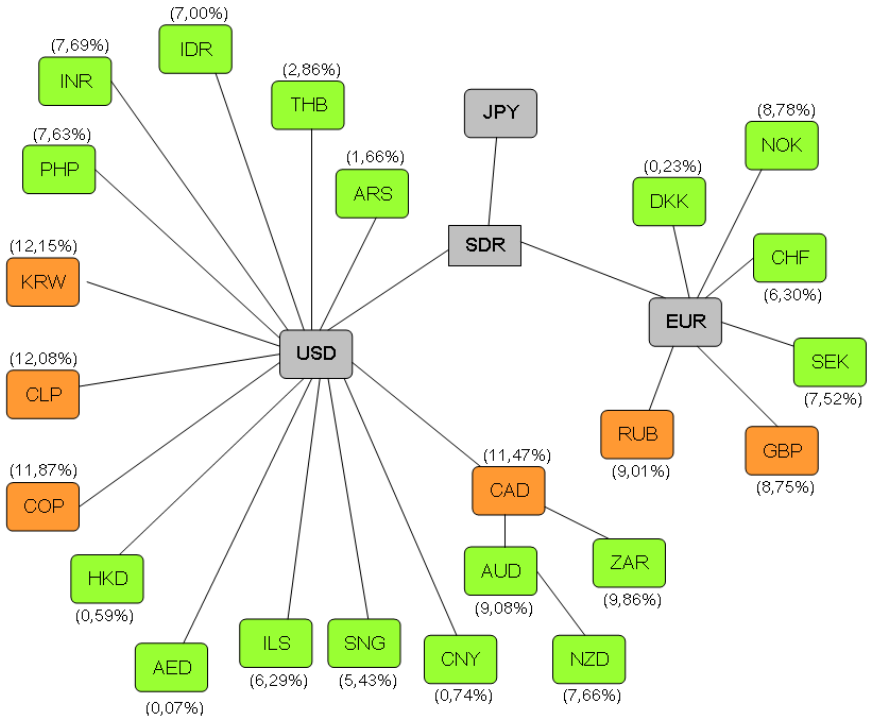


FIGURE 6. Minimum volatility spanning tree in the global currency graph

generate arbitrage free processes for the crosses in the main FX triangle USD-EUR-JPY. The other currencies are divided into the ones linked to the dollar and the ones linked to the EUR. No currency appears to be linked to the JPY. Furthermore, our cross-volatility minimization procedure shows that emerging market currencies such as the MXN, TRY and BRL and the Commonwealth block CAD-AUD-ZAR-NZD are best grouped together.

Discounted martingale conditions with respect to stochastic rates are imposed in two phases. In the first phase, which is reviewed in this section, we pretend interest rates are deterministic and formulate our models. In the second phase, we correct for the error and restore the exact martingale conditions as is explained in Section 8.

The FX process with deterministic interest rates we choose for the approximate local models is the of the same type we also use for the links in the minimum spanning volatility tree in the global FX graph. Namely, we use a "local correlation" model. Since this is the first article where such model is discussed, a few words of motivation are needed.

The most common choices in the FX domain, are to either (i) use the Black-Scholes model with adjusters [37], or (ii) use local volatility models [19] of the type

$$dX_t = (r^d(t) - r^f(t))X_t dt + \sigma(X_t, t)dW \tag{7.2}$$

or (iii) have Heston style stochastic volatility model [25] like

$$\begin{aligned} dX_t &= (r^d(t) - r^f(t))X_t dt + \sigma_t X_t dW \\ d\sigma_t^2 &= \kappa(\theta - \sigma_t^2) + \nu\sigma_t dW' \\ dW\,dW' &= \rho dt. \end{aligned} \tag{7.3}$$

The Black-Scholes model is the most widely used and it involves empirical adjusters for the implied Black-Scholes volatility treated as the unique, deal specific adjustable parameter. This

essentially means that the Black-Scholes formula is used as a convenient way of formulating an interpolation-extrapolation methodology out of unprocessed prices driven exclusively by offer and demand. The local volatility and the Heston model attempt to explain the skew and, in that sense, are less local than the Black-Scholes formula.

Over short time horizons, the local volatility and the Heston model behave in a similar way. Actually, over short time horizons all diffusion models can be approximated by a local volatility model. But for the purpose of CCR valuation, we are interested in long time horizons where this property fails.

Over long time horizons, the local volatility model is not adequate as it links directly the volatility to the level of cross rates. Conditional to a substantial change in level, within short time horizons there can be a change in volatility. However, when the change is consolidated, typically log-normal volatility reverts back to long run averages. Local volatility models are unable to reproduce the volatility relaxation process.

The Heston model instead allows for the volatility to mean revert. However, it has other shortcomings. Firstly, the square root process driving the instantaneous variance is unrealistic as it allows for scenarios whereby the volatility falls to nearly zero and stays very low for very long times. Depending on the strenght of the mean reversion parameter κ, the volatility process may admit even absorption at zero.

Secondly and perhaps more importantly, the Heston model has difficulties handling the mismatch between time scales: the process for the cross rate evolves on time scales of about one day, much shorter than the typical time scales for the volatility process which are of the order of a month. It is notoriously difficult to correlate effectively two processes characterized by different time scales. This often leads modelers to calibrate the Heston model with unrealistically high values of the volatility-of-volatility parameter ν and for the correlation parameter ρ.

The reason why local volatility models and the Heston model have been selected by the community are linked to the ability to solve them numerically with traditional means: sparse linear algebra is used for local volatility models and exact solutions are used for the Heston model. From our viewpoint, since we can use fast exponentiation to obtain transition probability kernels efficiently and robustly, these solutions are not the most natural.

We favor a process that we named "local correlation" and involves modeling correlations between the cross rate and volatility to produce a process which is only Markovian on discrete time intervals, not in continuous time. The process is built in two phases by means of kernel manipulations. In a first phase, we consider a time horizon ΔT of 5 business days and an elementary time step δt of the form $2^{-n}\Delta T$ where $n = 4, 5, 6$. In the second phase, the 5-day kernel is manipulated to insert correlations. Having done that, the kernel is further exponentiated to reach longer time horizons.

State variables are again given by pairs (x, a) where the variable x would typically take 128 values and the variable a would take $n_a \approx 10$. In the first phase, we fast exponentiate a simple block-diagonal Markovian of the form

$$\mathcal{L}_i(x, a; x', a') = \mathcal{L}_i^d(x, x'; a)\delta_{aa'} \tag{7.4}$$

from the elementary time step δt to the weekly time horizon ΔT. Here, the block Markovian has the form

$$\mathcal{L}_i(x, x'; a) = \frac{1}{2}\sigma(t_i)^2 s(a)^2 X(x)\Delta_x(x, x') \tag{7.5}$$

where $X(x)$ is the value of the FX cross corresponding to the state variable x and we assume for simplicity that both the domestic and the foreign interest rates are deterministic and equal to zero. The assumption of zero rates can easily be relaxed by means of a coordinate transformation, as explained below. Jumps can be added without any impact on performance and robustness.

The ΔT kernel is block-diagonal, with blocks denoted by $u_i(x, x'; a)$. The ΔT kernels $U_i(x, a; x', a')$ for the local correlation model we are interested in, are obtained by manipulating such block diagonal kernels to model rate-volatility correlations. We set

$$U_i(x, a; x', a') = \rho(x)U_i^1(x, a; x', a') + (1 - \rho(x))U_i^0(x, a; x', a'). \tag{7.6}$$

Here, $\rho(x)$ is a state dependent correlation function and acts as an interpolation parameter. In the fully correlated limit, the kernel is

$$U_i^1(x, a; x', a') = \begin{cases} u_i(x, x'; a)\delta(a' - \min(n_a, a + 1)) & \text{if } x' > x \\ u_i(x, x'; a)\delta(a' - a) & \text{if } x' = x \\ u_i(x, x'; a)\delta(a' - \max(a - 1, 0)) & \text{if } x' < x \end{cases} \tag{7.7}$$

In the uncorrelated limit, the kernel is instead

$$U_i^0(x, a; x', a') = u_i(x, x'; a) \cdot \tilde{u}_i(a, a') \tag{7.8}$$

where $u_i(a, a')$ is the ΔT-kernel for a mean reverting volatility process,

$$da_t = k(t)(\bar{a} - a_t)dt + s(t)dW. \tag{7.9}$$

Finally, we device a time dependent shift transformation similar in spirit to the shift transformation we use for the short rate model

$$\bar{X}(x_t) \to e^{\phi(t)}\bar{X}(x_t), \tag{7.10}$$

so that

$$\bar{X}(x_0) = \frac{Z_0^f(0)}{Z_0^f(T)} E_k\left[\frac{Z_0^d(T)}{Z_0^d(0)}\bar{X}(x_T)\right] \tag{7.11}$$

for all times $T > 0$ which are an integer multiple of ΔT. This yields the correct martingale condition for the local model of an FX-cross, which assumes deterministic rates. Stochastic interest rates are discussed in the next section.

8. Stochastic Rates and Quanto Corrections

Consistent valuation requires the choice of a base currency and the consistent generation of scenarios under the corresponding risk neutral measure.

Consider a portfolio of netting agreements, select a number of risk metrics to run, let (t_k) be an increasing sequence of epoch dates for the Monte Carlo simulation and let (y_k) be a scenario represented by a sequence of global state variables. We work under the discrete risk neutral measure whose numeraire is given by

$$B_k = \prod_{j=0}^{k-1} < y_j | e^{-\int_{t_j}^{t_{j+1}} r^d(s)ds} | y_{j+1} >^{-1} \tag{8.1}$$

where

$$< y_j | e^{-\int_{t_j}^{t_{j+1}} r^d(s)ds} | y_{j+1} >= E\left[e^{-\int_{t_j}^{t_{j+1}} r^d(s)ds}\delta(y_{t_{j+1}} - y_{j+1}) | y_{t_j} = y_j\right]. \tag{8.2}$$

The discrete measure of numeraire B_k is consistent with the risk neutral measure defined on the shortest time scale. In fact

$$E\left[\frac{B_j}{B_k}\Phi(y_j)|y_{t_k}=y_k\right]=E\left[e^{-\int_{t_k}^{t_j} r^d(s)ds}\Phi(y_j)|y_{t_k}=y_k\right]. \tag{8.3}$$

Next consider generating foreign exchange scenarios in two separate stages: firstly, one generates scenarios for an approximate process which assumes deterministic interest rates and, secondly, one corrects these paths to establish consistency.

By construction, the approximate process $\bar{X}_k$ with deterministic rates satisfies the following condition:

$$\frac{Z_0^f(k+1)}{Z_0^f(k)}\bar{X}_k=E_k\left[\frac{Z_0^d(k+1)}{Z_0^d(k)}\bar{X}_{k+1}\right] \tag{8.4}$$

Instead, the foreign exchange rate process X_k is such that

$$Z_k^f(k+1)X_k=E_k\left[<y_k|e^{-\int_{t_k}^{t_{k+1}} r^d(s)ds}|y_{k+1}>X_{k+1}\right] \tag{8.5}$$

Otherwise stated, the approximate process obeys the equation

$$E_k\left[\frac{\bar{X}_{k+1}}{\bar{X}_k}\right]=\frac{Z_0^f(k+1)}{Z_0^f(k)}\frac{Z_0^d(k)}{Z_0^d(k+1)} \tag{8.6}$$

while the exact process should satisfy

$$E_k\left[\frac{X_{k+1}}{X_k}<y_k|e^{-\int_{t_k}^{t_{k+1}} r^d(s)ds}|y_{k+1}>\right]=Z_k^f(k+1). \tag{8.7}$$

To adjust and achieve consistency, we set

$$\frac{X_{k+1}}{X_k}=e^{q_k^X}\frac{\bar{X}_{k+1}}{\bar{X}_k}. \tag{8.8}$$

where

$$e^{q_k^X}=\frac{Z_0^f(k)}{Z_0^f(k+1)}\frac{Z_0^d(k+1)}{Z_0^d(k)}\frac{Z_k^f(k+1)}{<y_k|e^{-\int_{t_k}^{t_{k+1}} r^d(s)ds}|y_{k+1}>}. \tag{8.9}$$

Foreign assets require a quanto correction. By construction, the approximate process $\bar{S}_k$ for a foreign asset assuming deterministic interest rates and foreign exchange rates, satisfies the following condition:

$$E_k\left[\frac{\bar{S}_{k+1}}{\bar{S}_k}\right]=\frac{Z_0^f(k)}{Z_0^f(k+1)} \tag{8.10}$$

while a consistently specified process should satisfy

$$E_k\left[<y_k|e^{-\int_{t_k}^{t_{k+1}} r^d(s)ds}|y_{k+1}>\frac{S_{k+1}X_{k+1}}{S_kX_k}\right]=1 \tag{8.11}$$

Let q_k^S be defined so that

$$\frac{S_{k+1}}{S_k}=e^{q_k^S}\frac{\bar{S}_{k+1}}{\bar{S}_k}. \tag{8.12}$$

Let us introduce the random variables

$$\xi_k(y_{k+1}) = \frac{< y_k | e^{-\int_{t_k}^{t_{k+1}} r^d(s)ds} | y_{k+1} >}{Z_k^f(k+1)} \frac{\bar{X}_{k+1}}{\bar{X}_k} e^{q_k^X(y_{k+1})} \tag{8.13}$$

and

$$\eta_k(y_{k+1}) = \frac{Z_0^f(k+1)}{Z_0^f(k)} \frac{\bar{S}_{k+1}}{\bar{S}_k}. \tag{8.14}$$

We have that

$$E_k\big[\xi_k\big] = 1, \quad \text{and} \quad E_k\big[\eta_k\big] = 1 \tag{8.15}$$

The condition to be satisfied by the exact process is

$$1 = E_k\left[\xi_k Z_k^f(k+1) \frac{Z_0^f(k)}{Z_0^f(k+1)} e^{q_k^S} \eta_k\right]. \tag{8.16}$$

Hence the condition for the quanto correction q_k^S is

$$e^{-q_k^S} \frac{Z_0^f(k+1)}{Z_0^f(k) Z_k^f(k+1)} = E\big[\xi_k \eta_k\big]. \tag{8.17}$$

Notice that

$$E\big[\xi_k\eta_k\big] = 1 + \Big(E\big[\xi_k\eta_k\big] - E\big[\xi_k\big]E\big[\eta_k\big]\Big) = 1 + \sigma(\xi_k)\sigma(\eta_k)\rho(\xi_k,\eta_k), \tag{8.18}$$

where $\sigma(\xi_k) = \sqrt{E\big[\xi_k^2\big]}$, $\sigma(\eta_k) = \sqrt{E\big[\eta_k^2\big]}$ and

$$\rho(\xi_k,\eta_k) = \frac{E\big[\xi_k\eta_k\big] - E\big[\xi_k\big]E\big[\eta_k\big]}{\sigma(\xi_k)\sigma(\eta_k)}. \tag{8.19}$$

The quanto correction is thus given as follows:

$$q_k^S = -\log\Big(1 + \sigma(\xi_k)\sigma(\eta_k)\rho(\xi_k,\eta_k)\Big) + \log\left(\frac{Z_0^f(k+1)}{Z_0^f(k) Z_k^f(k+1)}\right). \tag{8.20}$$

9. Credit Process

Credit processes can be modeled either with a distance-to-default model or by a defaultable equity model. Either way, we are confronted with a discrete state space parameterized by a variable $y = 0, ..., d-1$ such that $y = 0$ corresponds to the state of default. Let $\mathcal{L}(y_1, y_2; t)$ be the corresponding time dependent Markovian.

To find CDS spreads for the purpose of calibration one needs to evaluate conditional expectations of integrals over stochastic processes of the following form:

$$E_{T_0}\left[\int_{T_1}^{T_2} 1(y_t > 0)dt \;\middle|\; y_{T_1} = y_1, y_{T_2} = 0\right], \tag{9.1}$$

where $T_0 < T_1 < T_2$.

Let

$$u_{T_1}(y_0, y_1) = P\exp\left(\int_{T_0}^{T_1} \mathcal{L}(t)dt\right)(y_0, y_1). \tag{9.2}$$

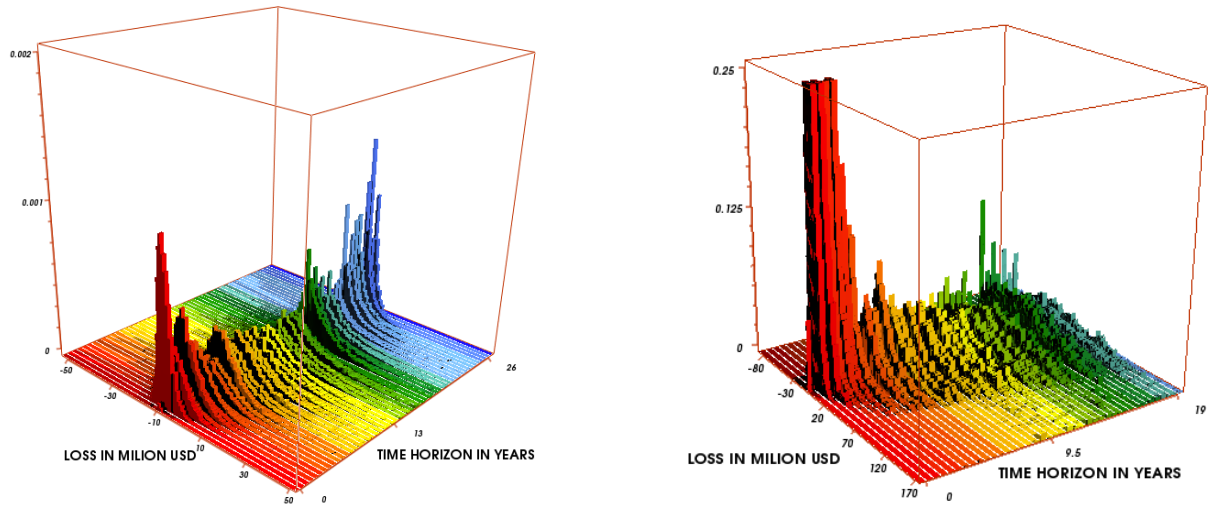


FIGURE 7. Portfolio loss distribution a EUR denominated fix-for-float portfolio (left) compared to the distribution for a cross-currency EUR-USD swap with nominal exchange at maturity (right).

This path-ordered exponential can be evaluated by repeated fast exponentiation in case the Markovian is a piecewise constant operator, as described above. Consider the kernel conditional to survival:

$$\tilde{u}_{T_1}(y_0, y_1) = \begin{cases} \frac{u_{T_1}(y_0,y_1)}{1-u_{T_1}(y_0,0)} & \text{if} \quad y_0, y_1 > 0 \\ 0 & \text{if} \quad y_0 > 0, y_1 = 0 \\ 1 & \text{if} \quad y_0 = y_1 = 0. \end{cases} \tag{9.3}$$

Consider the one-parameter family of perturbed kernels:

$$u^{\epsilon}_{T_2}(y_0, y_2) = \sum_{y_1} \tilde{u}_{T_1}(y_0, y_1) P \exp\left(\int_{T_1}^{T_2} \mathcal{L}^{\epsilon}(t)dt\right)(y_1, y_2) \tag{9.4}$$

where

$$\mathcal{L}^{\epsilon}(y_1, y_2) = \mathcal{L}(y_1, y_2) + \epsilon 1(y_1 > 0)\delta_{y_1 y_2}. \tag{9.5}$$

We have that

$$E_{T_0}\left[\int_{T_1}^{T_2} 1(y_t > 0)dt\delta(y_{T_2}) \;\middle|\; y_{T_0} = y_0\right] = \frac{d}{d\epsilon}\bigg|_{\epsilon=0} u^{\epsilon}_{T_2}(y_0, 0) \tag{9.6}$$

This formula for T_0 equal to the current date allows one to price a CDS in any given generic distance-to-default or credit-equity model. In the simulation stage, the same formula is useful when computed for $T_0 = T_1$ whereby T_1 is a generic epoch date for the global Monte Carlo simulation and it gives the expected survival time conditioned to being in a state of default by time T_2.

10. Counterparty Credit Risk

Counterparty credit risk management is a multilayered procedure whereby exposures are analyzed at various degrees of resolution, ranging from an individual instruments to global portfolios of netting sets.

Phases	Execution times
(i) Portfolio parsing and serialization	≈ 2 minutes
(ii) Algorithmic planning and optimization	≈ 6 minutes
(iii) Model library calibration	$\approx 30 - 60$ minutes per factor
(iv) Calculation of kernels	≈ 2 minutes
(v) Filling valuation tables	≈ 10 seconds
(vi) Simulation	≈ 1 minute
(vii) User driven 3d-visualization	$\approx 3 - 10$ seconds

TABLE 1. Processing phases and performance timings.

Figure 7 shows the loss distribution of a EUR denominated fixed-for-float swap in the left graph. Notice that tails are the fattest for intermediate maturities, while the spikes are concentrated near zero, both at the valuation date (when the swap is nearly at equilibrium) and at maturity. The graph on the right hand side instead refers to a cross-currency swap with nominal exchange at maturity. Since the foreign exchange risk associated to the final nominal exchange is substantial, risk profiles have increasingly fatter tails as one approaches the swap maturity.

Calculation times for risk profiles of these individual swaps vary from 28 seconds for the single currency swap to 65 seconds for the dual currency one. These times are acceptable for interactive use. One should also add that, in case portfolios are considered, very substantial economies of scale are possible as one can recycle kernel calculations across instruments.

We consider here a case study with a global portfolio of 302 netting sets, 6 currencies and 3572 swaps. We run a simulation up to a ten-year time horizon with quarterly epoch dates. The stragegy for simulating a large multi-currency portfolio of netting sets of interest rate derivatives of European or Bermudan type involves the phases in Table 10. The reported time concerns a dual socket system with 6-core Westmere processors X5690, 4 Tesla 1060 GPU cards and 36 GB of RAM memory. This amounts to a specification for a quite affordable high end workstation costing a small fraction of typical grid computing equipment used for portfolio risk management.

Phases (i) and (ii) can be conducted offline by means of procedures invoked to maintain portfolio databases. The global calibration phase (iii) is by far the most expensive one from the computational standpoint. Maintenance of model libraries needs to be carried out offline by running optimization procedures overnight at a total daily cost of several exaflops just to maintain a basic set of risk factors. These tasks largely consume GPU resources. On 4 GPU equipment, we observe a sustain performance of 1050 GF/sec when using Tesla 1060 cards.

Phase (iv) is user driven and GPU based while the subsequent phase (v) is CPU based as it requires complex deal specific information. The are collectively referred to as "KVT phase", where KVT stands for "kernels and valuation tables" and are subdivided into the following steps:

(A) Query payoff descriptor objects to obtain a list of all discount factors needed for all instruments and each one of the currencies involved.
(B) Run a primary backward induction pass to form tables for all needed discount factors for each currency.
(C) Evaluate stochastic integrals on bridges to account for path dependencies as is done in section 9.

(D) Run a secondary backward induction step to value callable swaps and other derivatives of European and Bermudan type.
(E) Populate valuation tables for all sub-portfolios corresponding to each individual counterparty and specific to a single currency.
(F) Generate transition probability kernels for all interest rate, credit and foreign exchange rate processes required.
(G) Obtain discounted transition probability kernels needed to model the money market account in all currencies of interest and to calculate quanto adjustments as explained in section 8.

The next step consists of a global simulation across all factors to persist in memory the values of all single counterparty positions for all scenarios and all epoch dates in the simulation and create an in-memory database to be used in the subsequent real-time data exploration phase. Using hyper-threading, we launch 22 threads in such a way to leave a full core available for the operating system. Each threads generates 4096 scenarios. The total time for generating 90112 scenarios and persisting all relevant information in memory is 68 seconds. The scenario generation is carried out mostly using CPU resources. A further optimization can be achieved by obtaining all the needed correlated normal deviates CPU side in parallel with the KVT phase. The KVT and simulation phases thus require less than 3 minutes in total and can thus be invoked interactively by the end user for any portfolio she wishes to analyze.

Having formed an in-memory database with simulation data, the end user can begin exploratory risk analysis. This phase involves cycling through the following tasks:

(G) Visualize in 3d the loss distributions for the entire portfolio or any given sub-portfolio. Figures 2 and 3 show examples of loss distributions. Another example of analysis is in 10 where the EUR denominated sub-portfolio is compared with the USD denominated sub-portfolio.
(H) Zoom-in on loss distributions to resolve risk concentrations
(I) Find the impact on loss distributions due to first order sensitivities whenever a state variable is changed to bump an interest rate curve, a CDS curve or a foreign exchange rate. Figure 9 shows the sensitivities of the loss distribution to a parallel shift in the USD curve for the entire portfolio and the USD denominated sub-portfolio.
(J) Find secondary 2-dimensional and 1-dimensional analytics such as tranche specific expected losses, expected positive exposures (EPE), credit valuation adjustments (CVA) and the corresponding tranched versions
(K) Add additional instruments and assess the marginal impact on loss distributions and on lower dimensional analytics for hedging purposes

In our case study example, each of these tasks takes a maximum of approximately 10 seconds to execute. We thus think that this application qualifies as a real time tool, especially if compared to standard grid computing schemes which require overnight processing to obtain only 2d analytics.

To explain the portfolio simulation in more mathematical detail, let $t_k, k >= 0$ be an increasing sequence of epoch dates for the simulation where t_0 is the valuation date. For each epoch $[t_k, t_{k+1})$, for each scenario s and for each counterparty c, we persist in memory the default probability $PD(k, c, s)$. We also persist in memory the valuation $V(k, c, s)$ of the position held by counterparty c.

Assuming that transitions to default are uncorrelated, the loss distribution in the $k-$th epoch period is given by the convolution product

$$\Lambda(k,l,s) = \big(\Lambda_{1s} * \Lambda_{2s} * .... * \Lambda_{n_c s}\big)(k,l) \tag{10.1}$$

where n_c is the total number of counterparties and

$$\Lambda_{cs}(k,l) = (1 - PD(k,c,s))\delta(l) + PD(k,c,s)\delta(l - V(k,c,s)). \tag{10.2}$$

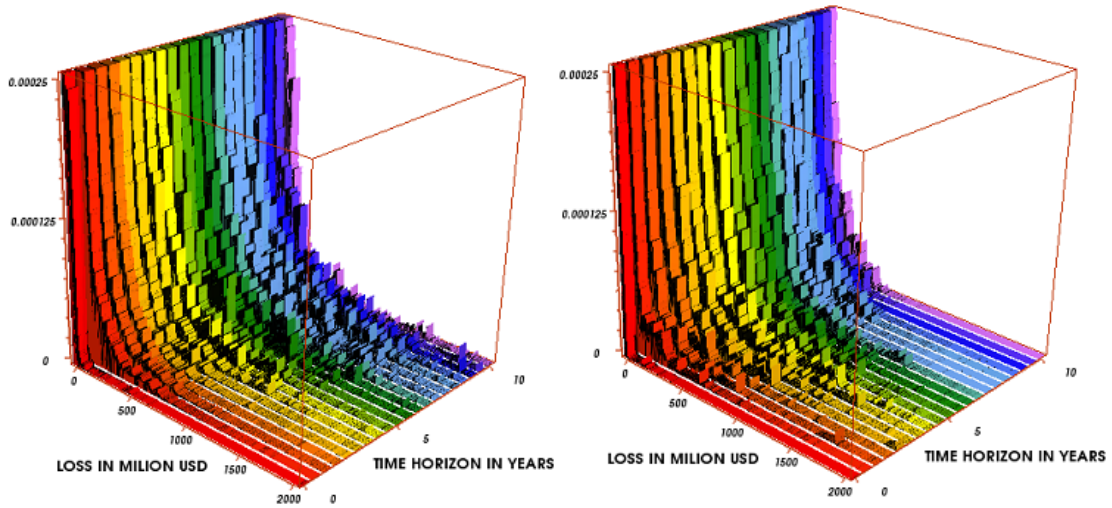
The calculation of the convolution product is best carried out on GPU equipment as the operation can be implemented efficiently.

The assumption of lack of correlation for transitions to default can be relaxed by modeling also events of perfectly correlated defaults among groups of borrowers. Accounting for these events is easier than evaluating correlation products, so the impact on overall performance is just marginal. This hybrid correlation model is similar to the one in [8] and [9].

Sensitivities are evaluated using the likelihood ratio method in [7]. Since transition probability from the valuation date to any epoch date in the future are are available, the signal-to-noise ratio decreases rapidly with the time horizon to the point that one can obtain clear resolutions even of 3d loss distributions with as few as 90112 scenarios. Figure 9 contains a graph for sensitivities of the loss distribution with respect to bumps in the USD curve as applied to the global portfolio and to the sub-portfolio of USD based counterparties.

11. Conclusion

Valuation theory is increasingly shifting its focus from the task of pricing instruments in isolation to a combined valuation-risk analysis within a global market and portfolio context. We find that the challenge can be met by means of coherent global market simulations. It is indeed quite possible to provide end users with real time calculators and 3d risk visualization tools ranging from individual deals to global portfolios of netting sets.



Portfolio USD *Portfolio EUR*

Figure 8. Comparison of loss distributions for the USD and EUR denominated subportfolios.

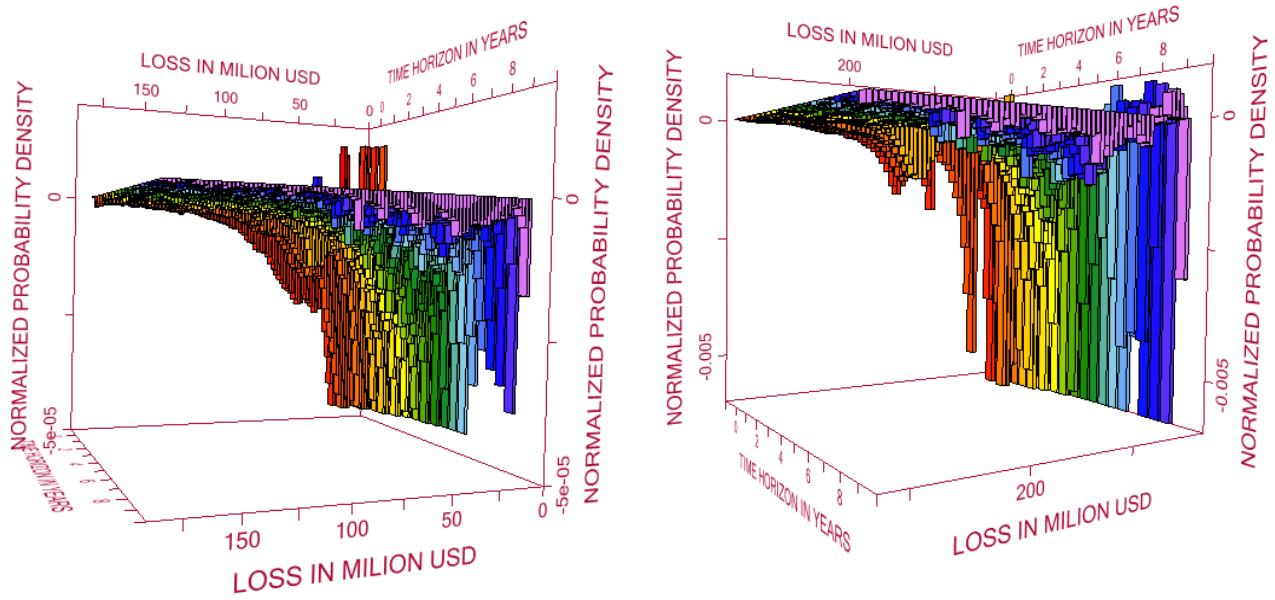


FIGURE 9. Comparison of sensitivities to a parallel shift in the USD curve for the loss distributions corresponding to the entire portfolio (left) and the USD denominated sub-portfolio (right).

We find that, to meet the challenge, it is essential to correctly interpret hardware architectures and orchestrate solutions on heterogeneous boards combining the strengths of MIMD and SIMD microprocessors. It is also essential to understand the impact of the narrow memory bottleneck due to the current limitations of data-path designs and resolve this difficulty at the algorithmic level.

We find that matrix multiplication and its tensorial variations are by far the most important compute bound algorithm for financial applications. One reason of interest is that they allow the modeler broad degrees of flexibility in designing stochastic processes which faithfully capture econometric evidence. Moreover, by adapting the mathematical formalism around matrix multiplication and the other precious few algorithms which are compute bound on current hardware, one can achieve highly polymorphic software designs and take performance to otherwise unattainable levels.

REFERENCES

1. Ahmes, *Rhind Mathematical Papyrus*, ed. Gay Robins and Charles Shute, British Museum Press (Nov 1987), 1650BC.
2. C. Albanese, *Kernel Convergence Estimates for Diffusions with Continuous Coefficients*, arxiv **0711.0132v2** (2007).
3. C. Albanese, *Stochastic Integrals and Abelian Processes*, arxiv **0711.2980v1** (2007).
4. C. Albanese, T. Bellaj, and P. Papathomas, *The Rotating Frames Algorithm for Robust Optimization and Applications to Global Calibration*, (forthcoming).
5. C. Albanese, G. Gimonet, and S. White, *Global Valuation and Dynamic Risk Management*, (2010).
6. _____, *An Introduction to Global Valuation*, (2010).
7. C. Albanese and H. Li, *Monte Carlo Pricing Using Operator Methods and Measure Changes*, SSRN **http://ssrn.com/abstract=1484556** (2009).
8. C. Albanese and A. Vidler, *A Structural Model for Bespoke CDOs*, Willmott Magazine (2007).

9. C. Albanese and A.. Vidler, *Dynamic Conditioning and CDO Modelling*, The Definitive Guide to CDOs, edited by Gunter Meissner, Risk Publications (2008).
10. Basle Committee on Banking Supervision, *Amendment to the capital accord to Incororate Market Risks*, Tech. report, Bank for International Settlements, 1996.
11. F. Black and M. Scholes, *The pricing of options and corporate liabilities*, Journal of Political Economy **81** (1973), 637–59.
12. D. Brigo and A. Capponi, *Bilateral Counterparty Risk Valuation with Stochastic Dynamical Models and Application to Credit Default Swaps*, arXiv **0812.3705v4 [q-fin.RM]** (18 Nov 2009).
13. D. Brigo and M. Masetti, *A Formula for Interest Rate Swap Valuation under Counterparty Risk in Presence of Netting Agreements*, Counterparty Credit Risk Modelling: Risk Management, Pricing and Regulation **London : Risk Books, 2005** (2010).
14. D. Brigo and A. Pallavicini, *Counterparty Risk and Contingent CDS under Correlation*, Risk **February** (2010).
15. G. Cesari, J. Aquilina, N. Charpillon, Z. Filipovic, G. Lee, and I. Manda, *Modelling, Pricing, and Hedging Counterparty Credit Exposure: A Technical Guide*, Springer, 2010.
16. Ian A. Cooper and Antonio S. Mello, *The Default Risk of Swaps*, Journal of Finance **46** (1991).
17. R. Courant, K. Friedrichs, and H. Lewy, *Ueber die partiellen differenzengleichungen der mathematischen physik*, Mathematische Annalen **100** (1928), 3274.
18. B. de Finetti, *Sul Significato Soggettivo della Probabilita'.*, Fondamenta Mathematicae (1931), 298–329.
19. B. Dupire, *Pricing with a Smile*, Risk Magazine (1994).
20. Robert F. Engle, *Autoregressive conditional heteroscedasticity with estimates of variance of united kingdom inflation*, Econometrica **50** (1982), 987–1008.
21. J. Gregory, *Counterparty Credit Risk: The New Challenge for Global Financial Markets*, Wiley Finance, 2010.
22. P. Hagan, D. Kumar, A. Lesniewski, and D. Woodward, *Managing Smile Risk*, Wilmott Magazine **September** (2002), 84–108.
23. Patrick S. Hagan, *Adjusters: Turning good prices into great prices*, The Best of Wilmott, John Wiley and Sons, Ltd, England, 2005 **1** (2005).
24. G. H. Hardy, *A mathematician's apology*, Cambridge: University Press, 1940.
25. S. Heston, *A Closed-form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options*, The Review of Financial Studies (1993), 327–343.
26. J. Hull and A. White, *Pricing Interest Rate Derivative Securities*, Review of Financial Studies **4** (1990).
27. C. Jacobi, *Vorlesungen ueber dynamik*, Encyclopedia Britannica Online Edition, 2010, 1847.
28. Robert A. Jarrow and Fan Yu, *Counterparty risk and the pricing of defaultable securities*, Journal of Finance **LVI** (2001).
29. S. P. Jones, J.M. Eber, and J. Seward, *Composing contracts: an adventure in financial engineering*, Proceedings of the fifth ACM SIGPLAN international conference on Functional Programming (2000).
30. David X. Li, *On Default Correlation: A Copula Function Approach*, Journal of Fixed Income **9** (2000).
31. A. Lipton and Artur Sepp, *Credit Value Adjustment for Credit Default Swaps via the Structural Default Model*, The Journal of Credit Risk **5** (2009).
32. R. Merton, *Theory of Rational Option Pricing*, Bell Journal of Economics and Management Science **4** (1973), 141–183.
33. J.P. Morgan/Reuters, *Riskmetrics—technical document*, 4th edition ed., J.P. Morgan, 1996.
34. R.D. Richtmyer and K.W. Morton, *Difference methods for initial-value problems*, Wiley-Interscience, 1967.
35. B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Wiley, 1996.
36. H. Stein and Kin Pong Lee, *Counterparty Valuation Adjustments*, The Handbook of Credit Derivatives, Bielecki, Tomasz; Damiano Brigo, and Frederic Patras, Eds. (2009).
37. Superderivatives, *Method and system for pricing options*, US Patent 7315838 (2008).
38. A. Turing, *Intelligent machinery*, Cybernetics: Key Papers. Ed. C.R. Evans and A.D.J. Robertson. Baltimore: University Park Press, 1968 (1948).