Nicolas Saleille
ENSAE ParisTech
Année 2014-2015

Stage long

Counterparty Credit Risk

Efficient statistical methods to compute CVA sensitivities

Non Confidential

April 2, 2015

Option :	Applied Mathematics
Company :	BNP Paribas
Supervisor :	Arnaud Tisseyre
Dates :	04/07/2013 - 04/07/2014
Adress :	10 Harewood Avenue NW6AA, London, UK

Note de synthèse

Stagiaire en recherche quantitative au sein du département des risques de BNP Paribas à Londres, j'ai travaillé sur la valorisation et la gestion du *risque de contrepartie* lié aux produits dérivés échangés par la branche Corporate and Investment Banking. L'organisation du stage et de l'équipe que j'ai intégré sont détaillés dans le résumé de mes missions joint à la fin de ce rapport.

Mon travail de recherche porte sur l'optimisation statistique et numérique du calcul des sensibilités du Credit Valuation Adjustment (CVA) aux différents paramètres de marché. Le CVA est une mesure du risque de contrepartie attaché à un produit dérivé. Dès lors qu'elle s'engage dans une transaction, la banque supporte un risque lié au défaut de sa contrepartie qu'elle souhaite lui facturer. Ce risque est valorisé comme la perte moyenne en cas de défaut, pondérée par la probabilité de défaut, et agrégée sur l'ensemble de la durée de vie du produit. Le résultat de cette valorisation est appelé Credit Valuation Adjustment et est retiré du prix du produit financier pour obtenir un prix "net" du risque de contrepartie. En pratique, les unités de trading sont facturées par la banque du montant de CVA correspondant aux opérations dans lesquelles elles s'engagent. Ce montant est ensuite transféré à une unité spécialisée qui se charge de le couvrir (il représente alors une sorte de provision). La CVA est également suivie par le département des risques qui calcule et analyse les montants agrégés par contrepartie. En particulier le stock de CVA entre en jeu dans le calcul du montant des fonds propres réglementaires de la banque.

Les principales difficultés liées à la CVA sont d'ordre computationnels. Son calcul repose sur l'estimation de l'exposition (i.e. la perte moyenne en cas de défaut) à chaque instant futur jusqu'à maturité du produit (ou du portefeuille si l'on raisonne au niveau de la contrepartie). Hormis quelques cas triviaux, il n'existe pas de formule fermée pour l'expression de l'exposition. Il est donc nécessaire de recourir à la simulation de facteurs de risques dans un espace temps discrétisé, puis d'évaluer la valeur du produit financier à chaque instant futur (généralement à l'aide d'une approximation) afin d'estimer le profil d'exposition. Ce processus, puisqu'il porte potentiellement sur des portefeuilles très larges, est numériquement coûteux.

Le problème spécifique qui m'a été posé concerne l'optimisation du calcul des *sensibilités* de CVA. Dans ce domaine la méthode standard est celle dite des différences finies, qui permet d'approximer une dérivée de premier ordre en négligeant les effets d'ordres supérieurs. Cette méthode est biaisée (la taille du biais dépendant des effets d'ordre supérieurs à un) et convergente; elle est surtout numériquement coûteuse puisqu'il faut évaluer la quantité d'intérêt en deux points différents. Ma mission a ainsi consisté à explorer les méthodes alternatives d'approximation des dérivés proposées par la littérature statistique et financière et à comparer leurs propriétés dans le cas spécifique de la CVA. J'ai implémenté différents estimateurs en Python (différences finies, ratio de vraisemblance, poids de Malliavin, méthode pathwize, Vibrato Monte-Carlo), testé leur performances, et rédigé un document de synthèse dont le présent rapport reprend les éléments principaux. Mes résultats ont ensuite été présenté à des équipes d'analystes et à des équipes modèles à Londres, New-York, Paris, Hong-Kong et Tokyo.

Les principales difficultés que j'ai rencontrées sont d'ordre théoriques puisque ce problème fait intervenir des concepts financiers avancés, des méthodes numériques complexes, et des modèles de simulation variés. Il m'a fallu faire preuve de persévérance avant de finalement être à l'aise avec, par exemple, le calcul Malliavin ou les méthodes avancées de Monte-Carlo. La compréhension de tous ces concepts a été longue, et n'a été possible que via l'expérimentation sur Python et C++.

Mes conclusions confirment l'intérêt de la méthode des différences finies. Malgré un coût numérique important, cette méthode est robuste à la complexité des modèles de diffusion et des modèles de valorisation.

Synthetic summary

As a quantitative research intern within BNP Paribas Risk-IM in London, I worked on the valuation and management of counterparty risk arising from capital market activities. During the internship I had the opportunity to work on several research topics. More details on my schedule and missions are presented in the mission summary enclosed with this report.

The project presented in this report deals with the numerical and statistical optimization of CVA's sensitivity computation. Credit Valuation Adjustement (CVA) measures the counterparty risk embedded in any financial derivative. By entering a transaction, the bank bears a risk linked to the default event of its counterpaty. This risk is, in most cases, valued as the expected loss conditional to the counterparty defaulting, weighted by its default probability, and aggregated on the whole life of the derivative. The resulting quantity is called CVA and is included in the pricing of the derivative in order to get a price net of any counterparty risk. In practice, trading desks pay CVA corresponding to the deals they enter to a specific desk (CVA desk) in charge of hedging this amount (which corresponds to a provision). CVA is also monitored by risk divisions at the counterparty level. In particular, aggregated CVA plays a role in the determination of regulatory capital requirements.

The main difficulties linked to CVA are of computational order. Its computation rely on the estimation of the *positive exposure* (i.e. the average loss in the event of default) at each future point until the maturity of the derivative (or the maturity of the portfolio from a counterparty perspective). Putting apart some trivial cases, there is no closed formulas for the positive exposure. It is thus necessary to simulate risk factors on discrete time spans and to price the derivative (usually through approximations) on each future time point and in each simulated scenario in order to compute the exposure profile. This processes, since it potentially involves very large portfolios and numerous scenarios, is numerically costly.

The specific issue I worked on was the optimization of the computation of CVA *sensitivities*. In this field, the standard method is finite difference, where first-order derivatives are approached neglecting higher order effects. This method provides a biased but consistent estimate. However it is numerically intensive since the function of interest (in our case the exposure) has to be evaluated in two separate points. My mission consisted in the exploration of derivative approximation methods in the statistical and financial literature, and in the comparison of their properties in the specific case of CVA. I implemented several estimators in Python (finite difference, likelihood ratio, Malliavin weights, pathwise estimator, Vibrato Monte-Carlo), tested their performance, and wrote this report. My results have been presented to analysts and quants located in London, New-York, Paris, Hong-Kong and Tokyo.

The main difficulties I encountered are theoretical since the issue of sensitivity estimation involves advanced concepts in mathematical and computational finance and in simulation. Thus, it took me several months to be at ease with, for instance, Malliavin calculus or advanced Monte-Carlo methods. A crucial aspect of the comprehension process has been experimentation on Python and C++. My results are somehow deceiving since I didn't find any optimization technique in a general framework. This result falls from the complexity of models used to generate exposures and the diversity of asset classes involved. However, my work still validates the current methodology in use within BNPP.

Acknowledgments

I am extremely grateful to all my former colleagues within BNP Paribas - Risk-IM / Counterparty Team for their constant help and support.

In particular I wish to express my sincere thanks to Arnaud Tisseyre, Natalie Perrier, Christian Keegan, and Beau Osselaere, without whom this internship wouldn't have been possible.

Last but not least, I warmly thank Guillaume Hermet for the time he spent with me on this research project, for his advices, and for his friendship.

Contents

Introduction

In the context of new regulations impulsed by the Basel III committee, financial institutions are facing new risk management challenges. In particular, counterparty credit risk and valuation adjustments (VAs) arising from derivative activities are a pregnant matter. In the aftermath of the credit crisis and the failure of major financial institutions previously considered as "too-big-to-fail", such as Lehman Brothers, Fannie Mae, and Freddy Mac, counterparty credit risk has become a hot topic. Multiple forms of adjustments are currently investigated and subject to a debate regarding their relevancy and computation. Among them, the popular ones are the Credit Valuation Adjustment (CVA), the Debit Valuation Adjustment (DVA), the Liquidity Valuation Adjustment (LVA), and the Funding Valuation Adjustment (FVA). After the credit crisis of 2009, CVA became the standard metric to account for counterparty credit risk embedded in derivative transactions. Derivative dealers are now actively managing it internally. At the deal level, CVA is charged to the relevant trading desks, so as to transfer the corresponding amount of cash to a specialized desk which focus is on hedging counterparty risk (CVA desk). At the counterparty level, the different VAs are computed and monitored by reinforced risks divisions. As a basis for CVA VaR and for the capital charge arising from counterparty risk, CVA has a particular importance among the different VAs.

Financial institutions are now legally entitled to price counterparty risks and valuation adjustments on a regular basis. To achieve this, one has to start with the central concept of *credit exposure*, defined as the loss in the event of the counterparty default. Since credit exposure exists during the whole life of trades, their estimation requires the simulation of risk factors on a grid of future time points and heavy Monte-Carlo methods. To be more specific, this process implies that portfolios have to be priced for each scenario and for each future time point, a task that can be numerically intensive and time consuming.

This report presents the challenges linked to the estimation and computation of counterparty credit risk. In part I, we define counterparty risk, explain important mitigation techniques, and present the new regulatory framework. In part II, we present the core part of this report, where we try to find an optimisation technique to price CVA sensitivities.

Monitoring CVA requires the regular computation sensitivities (the *Greeks*). To price CVA, one first have to estimate the credit exposure by Monte-Carlo means, which is, as underlined before, a costly process. Deriving CVA sensitivities is potentially even more costly. Indeed the standard first difference method relies on a second pricing of exposure profiles with "stressed" parameters. We therefore investigate alternative methods to compute sensitivities, and try to identify the most efficient ones in the context of CVA. Monte-Carlo methods only provides an estimator for expectations. Thus, we need to find techniques to traduce multiple derivatives of expectations in terms of simple expectations:

$$\frac{\partial}{\partial\theta}\mathbb{E}[f(X,\theta)] \xrightarrow{\text{method ?}} \mathbb{E}[g(X,\theta)]$$

This challenge can be achieved mainly by two means. On the one hand, first difference makes an extensive use of Taylor's approximation for continuous functions. On the other hand, weights methods use the statistical information contained in underlying models to build estimators. Our main conclusion is that, in the specific context of CVA at the counterparty level, first-difference is much more precise and easy to implement. Despite the potentially huge gains of weights methods, they are often too complex and numerically instable.

Part I
Counterparty credit risk: definition and measures

Counterparty credit risk refers to credit risk between counterparties involved into *financial derivatives*. For many years, its size and scale has been understated because of the myth around the creditworthiness of "too-big to fail" institutions. Since the credit crisis of 2007 and the failure of large financial institutions such as Bear Sterns, Lehman Brothers, Fannie Mae and Freddie Mac, this risk has been considered as a key financial risk. Today, counterparty credit risk is actively monitored by financial institutions and clearly defined by regulators.

1. Counterparty credit risk

1.1 Definition

Counterparty credit risk arise from a positive probability that one of the counterparties involved into a *derivative* transaction will fail to pay some or all of the cash-flows as specified by the derivative contract. According to the Basel II rules,

> *"Counterparty credit risk is the risk that the counterparty to a transaction could default before the final settlement of the transaction's cash flows. An economic loss would occur if the transactions or portfolio of transactions with the counterparty has a positive economic value at the time of default."*

Counterparty Credit Risk is a certain form of credit risk. However it differs from the classic lending risk because derivatives are complex and uncertain instruments:

(a) The value of a derivative at a potential default date will be the net value of all future cash-flows to be made under that contract. This future value can be positive or negative and is typically highly uncertain (as seen from today).

(b) Since the value of a derivative contract can be positive or negative, counterparty risk is typically bilateral. In other words, in a derivative transaction, each counterparty has risk to the other.

Counterparty risk is a mixture of **credit** risk (the deterioration of the credit quality of the counterparty) and **market** risk (the variation in the derivative value due risk factors' volatility). Unlike pure market risks, traditionally measured with a short-term Value-at-Risk (VaR), counterparty risk is measured through the entire life of the derivative. Before defining counterparty risk metrics, we present the central concept of credit exposure.

1.2 Credit exposure

Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space. We describe any financial underlying (also called risk factor in a risk management approach) as a continuous stochastic process $X : [0, T] \times \Omega \to (E, \mathcal{E})$ such that $X = (X_t : t \geq 0) \in \mathcal{L}^1(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathcal{P})$ where $\{\mathcal{F}_t\}_{t\geq 0}$ is the filtration associated to X, i.e. $\forall t \quad \mathcal{F}_t = \sigma(X_0, \ldots, X_t)$.

In the rest of this report, we will consider a portfolio of derivatives characterized by a set of cash-flows to be exchanged on $[0, T]$ with a given counterparty. We assume that all cash-flows are based on the underlying process X, where X belongs to a given asset class (interest and FX rates, equities, commodities, indices, etc). We denote the net discounted cash-flows of the financial derivative on the time interval $[u, v]$ by $\Pi_X(u, v) := \Pi(X_u, \ldots, X_v)$. The Net Present Value (NPV) of the derivative (or Mark-to-Market value as referred in [Gre10]) is defined as

$$\mathrm{NPV}(t) = \mathbb{E}_{\mathcal{Q}}\big[\Pi_X(t, T)\big|\mathcal{F}_t\big]$$

where $\mathcal{Q}$ denotes the risk-neutral measure. The NPV represents the *replacement cost* of the derivative, i.e. the cost of entering an equivalent transaction with another counterparty under the assumption of no trading costs. When measuring counterparty credit risk, the central concept to be considered is *credit exposure*.

Definition 1 (Credit Exposure)**.** *Credit exposure on a specific derivative transaction is defined as the loss incurred by the bank in case its counterparty defaults in t. It is given by the **positive** net present value of the portfolio with the counterparty:*

$$E(t) = \max\Big(NPV(t), 0\Big) = NPV^{+}(t) \quad \forall t \in [0, T]$$

Credit exposure at time t depends on future unknown cash-flows to be received on $[t, T]$ and is thus random as of today.

If the NPV of a derivative can typically be positive or negative, exposure represents a loss in the event of default and is defined as a positive amount. Negative NPVs means that the bank owes future cash-flows to its counterparty. In this case there is no proper counterparty risk (if B defaults and B has a positive claim on A, A still owes B the amount of the claim). The next subsection defines standard counterparty risk metrics, namely the Expected Positive Exposure (EPE) and the Potential Future Exposures (PFE).

1.3 EPE and PFE

The traditional metrics to measure counterparty credit risk are the Expected Positive Exposure (EPE) and the Potential Future Exposure (PFE). The EPE indicates the average loss incurred by the bank in case the counterparty defaults. It is the basis for the calculation of Credit Valuation Adjustments (CVA) and the regulatory capital figures (via CVA VaR).

Definition 2 (Expected Positive Exposure (EPE))**.** *The EPE gives the expected value - as of today - of the loss incurred (assuming no recovery) if the counterparty defaults at time t.*

$$EPE(t) = \mathbb{E}[B(0, t)NPV^{+}(t)] \tag{1}$$

where $B(0, t)$ is the spot price of the zero-coupon bond with maturity t. Notice that the external expectation is taken under the physical measure, not the risk-neutral one.

Definition 3 (Expected Exposure Profile)**.** *The expected exposure profile is defined as the curve*

$$t \to EPE(t) \quad \forall t \in [0, T]$$

In risk divisions of financial institutions, EPE profiles are computed at the deal level and aggregated by type of underlying (IR, FX, equities, etc.) at the portfolio and counterparty level. In this context Π_X can be seen as the net discounted cash flows of the portfolio on a given counterparty, rather than the ones of a single derivative. Consequently, the NPV and EPE rely on the value of a portfolio containing k trades, with k being adapted depending on the level of granularity we need.

The estimation of the EPE profile on $[0, T]$ is costly and challenging, in particular in the case of exotic derivatives. Since the EPE involves the *future* NPV of the portfolio, two steps are needed in the estimation process. First (outer step), one has to draw realizations of all risk factors up to the horizon T. In the second step (inner step), one re-prices each instrument in the portfolio at the horizon conditional on the drawn risk factors. It involves potentially two layers of Monte-Carlo simulation. The following algorithm is commonly used:

EPE simulation algorithm

1. For $j \in [\![0; n]\!]$ (**outer step**):
   (a) Draw a path for the risk factors X^j over the dates $(t_1, \dots, t_k)$ under the physical measure, i.e. generate the filtration $\mathcal{F}^j_{t_k}$.
   (b) At each time point t_i, *compute* (closed formula) or *simulate* (**inner step**) the conditional expectation:
   $$\text{NPV}_j(t_i) = \mathbb{E}_{\mathcal{Q}}\left[\Pi_X(t_i, t_k) \middle| \mathcal{F}^j_{t_i}\right]$$
2. Average over the n paths to get the estimated EPE profile
   $$\widehat{\text{EPE}}(t_i) = \frac{1}{n} \sum_{j=1}^{n} B(t_0, t_i) \text{NPV}_j(t_i)^+$$

For vanilla and linear products, the NPV can often be estimated through a closed formula (conditionally to the filtration $\mathcal{F}^j_{t_k}$). However, for exotic derivatives where there isn't any closed formula for this quantity, the inner step is computationally demanding since a second layer of Monte-Carlo is required (nested simulation). Methodologies have been developed to save computing costs, most of them coming from the most studied application of nested Monte-Carlo in finance, i.e. American option pricing. For instance it is possible to use the Longstaff-Schwartz algorithm to price exotic exposures (see [LS01]): a small number of inner step samples are used to estimate a parametric relationship between the underlying and the value of the option (least square Monte-Carlo).

In trading applications, techniques such as least-square Monte-Carlo are useful to price the EPE in a fast and accurate manner for a given exotic trade. However in risk-management

applications, the large amount of counterparties, portfolios and trades to be considered makes it unrealistic to run nested or least-square Monte-Carlo simulations on a regular, industrialized basis. Typically, risk valuation systems propose a set of pricers available at step 1.(b) for which closed formulas exists. Exotic trades are usually represented using conservative approximations, in order to avoid the inner step in the EPE estimation process (override process).

Definition 4 (Potential Future Exposure (PFE))**.** *The potential future exposure at a given confidence level α is defined as an exposure that would not be exceeded with a probability of nor more than $(1-\alpha)$.*

$$PFE(t,\alpha) = \min\left\{x \in E | \mathcal{P}\big(B(0,t)NPV(t) \geq x\big) \leq \alpha\right\} \quad \forall\, t \in [0,T] \tag{2}$$

The PFE is a percentile on the exposure distribution defined for each time point in the future. It is similar to a VaR defined on long time intervals. PFE is estimated taking the α-percentile of the empirical distribution of the exposure.

EPE and PFE are the two main metrics used to supervise counterparty credit risk. On the one hand, the EPE profile indicates the average loss the bank would suffer in the event of default of its counterparty. On the other hand, the PFE gives the loss up to a given level of confidence. Typically, credit lines are internally granted by "credit officer" teams to each counterparty. New trades with the counterparty are then agreed by the risk division as long as the aggregated PFE stays within the limit defined by the credit line.

The next section briefly discuss mitigation techniques commonly used to address counterparty credit risk, namely netting, and collateralisation.

2. Counterparty risk mitigation

Large derivative dealers are involved into thousands of derivative trades at the same time. These trades are typically made with many clients including some other big players, and involve various asset classes (interest rates, foreign exchange, equities, commodities, credit derivatives, etc.). Consequently they have set in place various simple techniques so as to mitigate counterparty credit risk. Before detailing them, we explain relationships between derivative dealers in a standard framework.

2.1 Trading relationships on derivative markets

In a typical OTC derivative transaction, three parties are involved: a dealer (D), a counterparty (C) and a hedging counterparty (HC). C initiates the transaction by entering a trade with D. D then hedges the position by entering the opposite trade against HC, *so as to be neutral to market risk*. D acts as a market maker, i.e. it creates supply and demand for financial products. In practice, D splits the transaction into several parts and hedges separate cash flows to different HCs.

From a counterparty risk perspective, the dealer's position is not neutral. He faces (C) and (HC) and will have a positive exposure with at least one of these two agents. The two main mitigation instruments are netting and collateralisation, defined from a legal standpoint in Master Agreements and Credit Support Annexes.

Definition 5 (Master Agreement)**.** *A Master Agreement is a document agreed between two parties that sets out standard terms that apply to all the transactions entered into between those parties. For each new trade, the terms of the master agreement do not need to be re-negotiated and apply automatically.*

Master agreements are a simple way to reduce transaction costs with a specific counterparty. Since they define netting sets, they are also a way to reduce the bilateral exposure (see next subsection). The ISDA master agreement is a standardized master agreement published by the International Swaps and Derivatives Association. It is the most commonly used master agreement for OTC derivative transactions internationally. Let's now define credit support annexes.

Definition 6 (Credit Support Annexes)**.** *Credit Support Annexes (CSAs) are legal documents embedded in Master Agreements. They define the rules under which collateral is posted between counterparties to mitigate credit risk arising from "in the money" derivative positions.*

A large derivative dealer such as BNPP typically sets master agreements and CSAs with all regular counterparties (large corporates, financial institutions, sovereign counterparties). The next subsections details their impact in terms of counterparty risk.

2.2 Netting of exposures

Derivative dealers often engage into numerous bi-directional transaction with a given counterparty, in particular in the context of hedging. A netting agreement is a legal agreement that comes into force in the event of a bankruptcy. It enables one to net the value of trades with a defaulted counterparty before settling the claims. Consider a financial institution that holds a portfolio of k OTC derivative contracts with its counterparty. Without netting, counterparty level exposure is given by

$$E(t) = \sum_{i=1}^{k} \max\Big(\mathrm{NPV}^i(t), 0\Big)$$

In the presence of netting agreements, it becomes

$$\widetilde{E}(t) = \max\Big(\sum_{i=1}^{k} \mathrm{NPV}^i(t), 0\Big) \leq E(t)$$

Netting agreements are thus risk reducing if there is a non-null probability that the NPV of some trades in the netting set (the set of trades impacted by netting) becomes negative in the future.

2.3 Collateralisation of exposures

A CSA defines the rules under which collateral is posted between counterparties to mitigate credit risk arising from "in the money" derivative positions. More specifically, it sets the respective maximum exposures that trading counterparties accept to take on each other (thresholds). At

regular time intervals (margin call frequency), the actual exposure is compared to the threshold. If one of the counterparty is more exposed than agreed, collateral is exchanged to reduce the exposure back to acceptable levels. The collateral must meet the eligibility criteria in the agreement, e.g., which currencies it may be in, what types of bonds are allowed, and which haircuts are applied.

Collateral typically reduces credit exposure and modifies the nature of risk. Since counterparties have a legal right to keep the collateral in case the other defaults, credit exposure can be reduced from the amount of collateral received (collateralised exposure):

$$\widetilde{E}_c(t) = \max\Big(\sum_{i=1}^{k} \mathrm{NPV}^i(t), 0\Big) - C(t)$$

where C_t is the stock of collateral as of t. Since the amount of collateral is adjusted at each margin date, the remaining risk comes from the volatility of the exposure on the re-margining period: a large increase in the exposure (for instance because of market movements) means more counterparty risk; a large decrease (opposite movements) give rise to a liquidity funding risk (since the bank will probably have to post collateral to the counterparty). Collateralisation transforms a long-term counterparty credit risk into a mixture of short-term risks.

3. Counterparty risk in the Basel III regulatory framework

Since the credit crisis of 2007, the G20 leaders agreed that OTC derivative markets should be more transparent and that counterparty risk should be subject to better risk management practices and higher capital charges. Published in December 2010, the Basel III capital framework sets rules to ensure the risk coverage of bank's counterparty credit risk exposures arising from OTC derivatives, Repo, and other securities financing transactions. On average, the new rules double bank's capital requirements for exposures to bilateral counterparty credit risk (on this matter see [Ing13]). The measures also provides incentives to move transactions to risk-reducing Central Counterparties (CCPs), as the higher capital requirements for bilateral OTC derivatives increases the cost of bilateral transactions.

3.1 Mandatory clearing of standardised derivatives

In September 2009, the G20 set an objective of introducing mandatory clearing for standardized derivatives. The European Union, the U.S. and other jurisdictions in the G20 are currently developing and implementing regulations to achieve this objective. In addition, international regulators developed more demanding international standards for CCPs.

In the US, the Dodd-Frank Act implements the mandatory clearing of swaps (interest rate swaps, cds, currency swaps, commodity swaps, etc.) and, where appropriate, trading of swaps on exchanges or electronic trading platforms. In Europe, the European Markets Infrastructure Regulations (EMIR) is currently defining the OTC derivatives contracts that will be subject to mandatory clearing and to reporting into trade repositories.

3.2 New calibration of credit exposures

The Basel II capital framework did not reflect counterparty credit risk in an appropriate way. In particular wrong-way-risks, CVA losses, and correlation between the exposures of large financial

institutions where not adequately captured. The Basel III committee proposed a new approach to credit exposure to tackle these issues.

Wrong-way risk (i.e. a positive correlation between the creditworthiness of trading counterparties and the level of exposures) now have to be carefully followed, and exposures should reflect the value of instruments *in the event of default*. The computation of the exposure amount must also account for a period of stress (a VaR component) so that capital charges won't fall too low during periods of compressed market volatility. Lastly, standards for collateral management and initial margining have been raised. Slippage periods for very large netting sets and netting sets with illiquid collateral or exotic trades has been doubled compared to Basel II. Re-securitisation are no longer eligible collateral, and haircuts have been increased.

3.3 New capitalization rules

Basel III also define a new approach to the capitalization of credit exposures. Two thirds of the counterparty risk related losses during the credit crisis were actually from CVA volatility rather than defaults (see [Ing13]). Consequently, a new capital charge against the *volatility* of CVA is introduced (CVA VaR) for non-centrally cleared derivatives. Exposures to CCPs will also have to be capitalized (even if subject to a low risk weight) to recognize that they are not completely risk-free.

****

As presented in this first part, counterparty credit risk became a pregnant matter on derivative markets. The financial crisis and new regulations made it crucial for financial institutions to develop internal systems and procedures dedicated to counterparty risk management. This processes involves a large computation power as underlined in I to compute regularly exposure profiles, valuation adjustments, and their sensitivities. In the next part we review and compare statistical and numerical methods to compute sensitivities, in the particular case of CVA.

****

Part II
Numerical computation of CVA sensitivities

From a risk management perspective, CVA sensitivities are critical indicators. New regulations impulsed their production at the counterparty level on a regular, industrialized basis in most financial institutions. The estimation of CVA makes an extensive use of Monte-Carlo simulation and thus comes at a high computational cost. The most evident way of estimating CVA sensitivities is to use a first difference approximation. However this process is very costly since EPE profiles are heavy to estimate and have to be priced twice on the whole portfolio for each counterparty. In this part, we investigate alternative methods, such as weights methods, adapted to sensitivity computation, and compare them in the specific case of CVA. Our main conclusion validates the use of first-difference in the case of CVA. In particular, weights methods may have a smaller computational cost, but they are complex and give poor variance result for smooth functions such as CVA.

1. Credit Valuation Adjustment

Credit Valuation Adjustment on a given counterparty is defined as the difference between the price of a portfolio with default risk-free counterparties and that with default risky counterparties. It reflects the market value of counterparty credit risk. The computation of CVA rely (1) on the EPE relative to a specific trade, portfolio, or counterparty, (2) on the default probability of the relevant counterparty, and (3) on the recovery rate associated with the counterparty. The EPE is typically computed for each trade by risk valuation systems, and valued on a regular basis to account for market moves. The two other components can be either market implied or internally assessed. However, new regulations tend to promote market implied parameters.

1.1 Definition

In order to precisely define CVA, we need a pricing formula. In presence of counterparty risk, the net discounted cash-flows of the derivative writes

$$\begin{aligned}\Pi_X^D(t,T) &= \Pi_X(t,T)\mathbb{1}\{\tau_c > T\} \\ &+ \Big[\Pi_X(\tau_c,T) + B(t,\tau_c)\big((1-R_c)\cdot NPV(\tau_c)^+ \\ &- (-NPV(\tau_c))^+\big)\Big]\mathbb{1}\{t \le \tau_c \le T\}\end{aligned}$$

where τ_c and R_c are respectively the default time and the recovery rate of the counterparty. Taking the expected value of this last expression, we get the following equation:

$$\mathbb{E}[\Pi_X^D(t,T)|\mathcal{F}_t] = \mathbb{E}[\Pi_X(t,T)|\mathcal{F}_t] - (1-R_c)\,\mathbb{E}[B(t,\tau_c)NPV(\tau_c)^+\mathbb{1}\{t \le \tau_c \le T\}|\mathcal{F}_t] \quad (3)$$

The value of the derivative in presence of counterparty risk at time t is given by its non-risky net present value, minus an adjustment term traducing the probability that the counterparty defaults before the maturity date. This last term defines CVA (see [Gre10] for the full demonstration).

Definition 7 (Credit Valuation Adjustment)**.** *CVA is the market value of counterparty credit risk embedded in the derivative* Π_X*. Rewriting the adjustment term in equation* (3) *and integrating over time, we get the following formula:*

$$CVA(t) = (1 - R_c) \int_t^T \mathbb{E}\big[B(t,s) NPV(s)^+ \mathbb{1}\{\tau_c = s\} | \mathcal{F}_t\big] ds$$

where R_c *is the recovery rate of the counterparty, and* $B(t,s)$ *is the discount factor in* t *with maturity* s*. If we assume that the default time and the credit exposure are independant (**no wrong-way risk assumption**), then we can split the conditional expectation and CVA (seen as of today) simplifies to*

$$\begin{aligned} CVA(0) &= (1 - R_c) \int_0^T \mathbb{E}\big[B(0,s) NPV(s)^+\big] d\lambda_c(s) \quad (4) \\ &= (1 - R_c) \int_0^T EPE(s)\, d\lambda_c(s) \quad (5) \end{aligned}$$

where $\lambda_c(s)$ *is the cumulative default probability of the counterparty at horizon* s*.*

From equation (5) we note that CVA depends on every future values taken by the EPE. This means that any future market movement that will affect the exposure profile will have a direct impact on CVA. Estimating CVA thus requires to perform heavy computations as described in subsection 1.1.3 of part I. Once the EPE computed, CVA is estimated through an approximation of the integral term using the k time points:

$$\widehat{\text{CVA}} \approx (1 - R_c) \sum_{i=1}^{k} \widehat{\text{EPE}}(t_i)\big(\lambda(t_i) - \lambda(t_{i-1})\big)$$

When computed at the counterparty level, CVA potentially relies on large portfolios of trades containing all types of derivatives (vanilla and exotic on various underlying asset classes: rates, equities, indexes, etc). Since it highly depends on the composition of the considered portfolio, it is difficult to isolate properties that are specific to this metric.

1.2 Sensitivities

In this subsection, we focus on the estimation of CVA's derivatives with respect to several parameters of the underlying process $\{X_t\}_{t \leq 0}$. We denote by θ the parameter of interest. There are two ways to look at the dependency to θ. In the first one, the underlying is a function of θ, i.e. $X_t = X(\omega_t, \theta)$. In the second view (which, in the end, is equivalent), θ is a parameter of the law in which we sample the underlying process, i.e $X_t \in \mathcal{L}^1(\Omega, \mathcal{F}, \{\mathcal{F}_t\}, \mathcal{P}_\theta)$. Depending on the view we adopt, we shall use different notations. In the first view we will denote variables as $X_t^\theta := X(\omega_t, \theta)$, $\Pi_X(t, T, \theta)$, $\text{CVA}(t, \theta)$, etc. In the second view, we will use the expectation to emphasize this dependency; we define

$$\mathbb{E}_\theta\left[f(X_t)\right] := \int_\Omega f\big(X(\omega, t)\big) d\mathcal{P}_\theta(\omega)$$

We are looking for efficient estimators and procedures to estimate the derivative of CVA with respect to θ. We are interested by the spot CVA, so we set $t = 0$ and we drop the time index in notations, i.e. $\text{CVA}(\theta) = \text{CVA}(0, \theta)$. We have:

$$\frac{d}{d\theta}\text{CVA}(\theta) = \frac{d}{d\theta}\left((1 - R_c) \int_0^T \text{EPE}(s, \theta)\, d\lambda(s)\right)$$

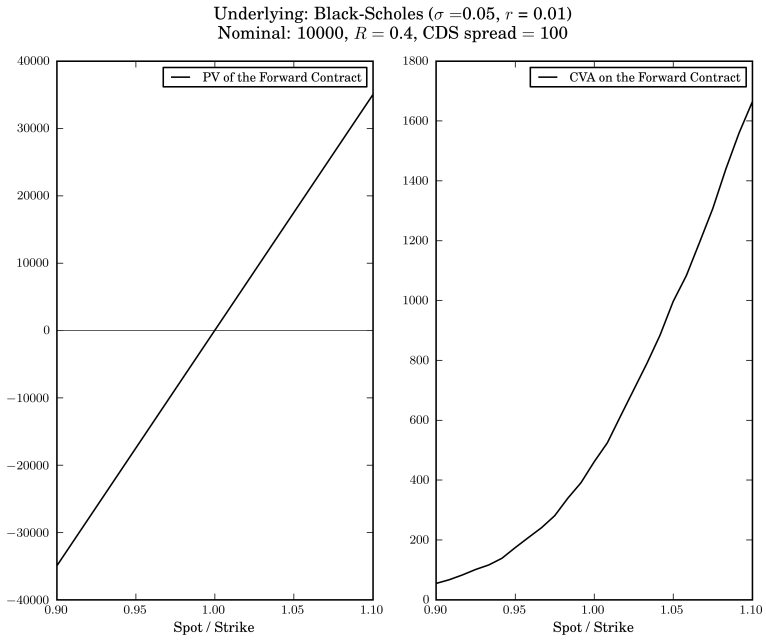
Assuming that $\frac{\partial}{\partial\theta}\text{EPE}(t,\theta)$ exists and is continuous with respect to θ, then we can apply the Leibniz integral rule and differentiate under the integral sign, i.e.

$$\frac{d}{d\theta}\text{CVA}(\theta) = (1-R_c)\int_0^T \frac{d}{d\theta}\text{EPE}(s,\theta)\,d\lambda(s) \tag{6}$$

Differentiating CVA is equivalent to differentiating the EPE at each time point in the future. In the rest of this work we suppose that the necessary conditions for equation (6) to hold are statisfied. In most models, diffusion processes are differentiable with respect to the initial condition, drift and volatility parameters. Most derivatives contracts will preserve differentiability (only some exotic options are problematic, e.g. binary options).

Figures (1) and (2) display the spot CVA on a forward and on a digital option for in-the-money and out-of-the-money positions. From these graphs we can infer three ideas. First, CVA includes an optionality component since negative NPVs of the portfolio are floored to zero. Second, it can be either convex or concave with respect to the parameter of interest. Third, CVA is smooth with respect to the initial condition in these simple examples, even in the case of the digital option, where the payoff is discontinuous. Smoothness comes from the integral term in CVA's expression.

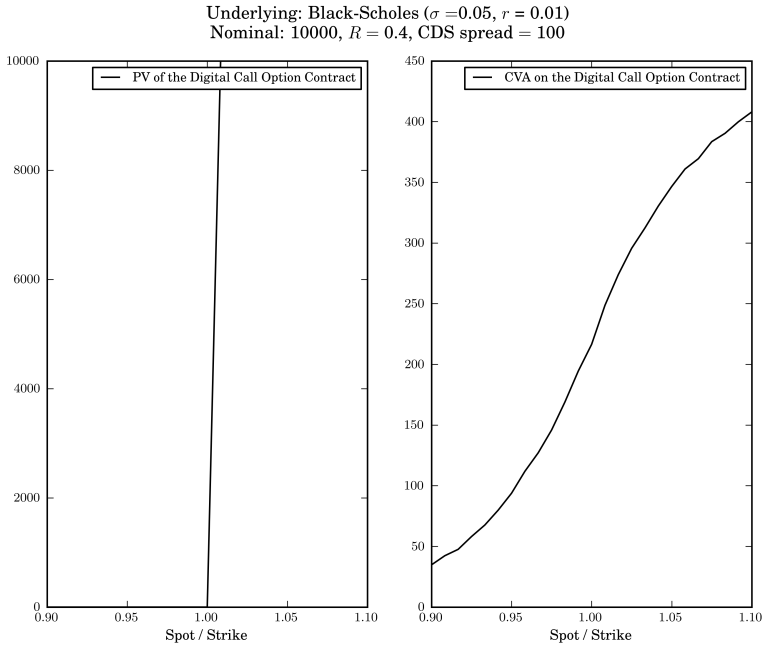
Figure 1: Spot CVA vs Spot PV on a 5Y Forward Contract



Deriving analytical solutions for CVA sensitivities is impossible in most cases since it is computed at the counterparty level on portfolios that aggregate many trades, and that have complex dependencies to θ. Even at the trade level, CVA introduces a option component that makes it difficult to get closed formulas. The next example illustrates this complexity on the simple example of a forward contract.

Example 1 (CVA sensitivities on a forward in Black-Scholes)**.** *Let's consider CVA on a portfolio containing a forward with maturity T contracted in $t = 0$. We take a one-dimensional*

Figure 2: Spot CVA vs Spot PV on a 5Y Digital Call Option



geometric Brownian motion for the underlying process X and we suppose that every hypothesis of the standard Black-Scholes framework are verified. The forward is contracted at the spot price $F(0,T) = e^{rT}x_0$. The payoff of this portfolio at time t is

$$\Pi_X(t,T) = e^{-r(T-t)}(X_T^\theta - e^{rT}x_0)$$

Since the reactualised underlying process is a martingale under the risk-neutral measure, the net present value of the portfolio at t is given by

$$\begin{aligned} NPV(t) &= \mathbb{E}_{\mathcal{Q}}\left[e^{-r(T-t)}(X_T^\theta - e^{rT}x_0)\middle|\mathcal{F}_t\right] \\ &= \mathbb{E}_{\mathcal{Q}}\left[e^{-r(T-t)}X_T^\theta\middle|\mathcal{F}_t\right] - e^{rt}x_0 \\ &= X_t^\theta - e^{rt}x_0 \end{aligned}$$

The EPE profile of this portfolio is defined at each time time $t \in [0,T]$ by the expectation

$$EPE(t) = \mathbb{E}\Big[e^{-rt}\big(X_t^\theta - e^{rt}x_0\big)^+\Big]$$

At each date in the future, the EPE generated by the forward corresponds to the spot price of a European call option with strike $K(t) = e^{rt}x_0$. An optionality has been added by the simple fact that we are considering the EPE. Assuming that we can differentiate without issues under the integral sign, CVA's sensitivity to θ on this portfolio is given by

$$\frac{d}{d\theta}CVA(\theta) = (1-R_c)\int_0^T \frac{d}{d\theta}\mathbb{E}\Big[e^{-rs}(X_s^\theta - e^{rs}x_0)^+\Big]d\lambda_c(s)$$

CVA's sensitivity is close to the sum of the portfolio's sensitivities weighted by the default probability. If we set $\theta = x_0$ then using Black-Scholes closed formula for the Δ we get:

$$\Delta_{CVA} = (1 - R_c) \int_0^T \mathcal{N}\left(\frac{\ln(e^{-rs}) + (r + \sigma^2/2)s}{\sigma\sqrt{s}} \right) d\lambda_c(s)$$

Appart from this very simple example, closed formulas for CVA sensitivities are not available. Consequently, the estimation of CVA sensitivities can be achieved mainly through two different approaches: (1) the Taylor approximation for continuous functions enables to derive a generic estimator for sensitivities, (2) the density function of the underlying process can be characterized so as to compute weights used in sensitivity estimators. Both methods require a Monte-Carlo scheme.

2. First difference approximation

The general issue of sensitivities estimation of expectations can be addressed using a first difference approximation. This technique is based on the Taylor approximation for continuous functions.

2.1 Definition of the estimator

We are interested in estimating the derivative of an expectation with respect to a given parameter θ. Going back to equation (6), we have

$$\mathrm{CVA}^{'}(\theta) = (1 - R_c) \int_0^T \frac{\partial}{\partial\theta} \mathrm{EPE}(s, \theta) d\lambda(s)$$

If we go further and suppose that the EPE is twice differentiable with respect to θ, we can apply the Taylor approximation at order 2 for a small shift $h > 0$:

$$\mathrm{EPE}(t, \theta + h) = \mathrm{EPE}(t, \theta) + \mathrm{EPE}^{'}_{\theta}(t, \theta)h + \frac{1}{2}\mathrm{EPE}^{''}_{\theta}(t, \theta)h^2 + o(h^2) \tag{7}$$

Rearranging the terms of in equation (7), we get the following expression:

$$\frac{\mathrm{EPE}(t, \theta + h) - \mathrm{EPE}(t, \theta)}{h} = \mathrm{EPE}^{'}_{\theta}(t, \theta) + \frac{1}{2}\mathrm{EPE}^{''}_{\theta}(t, \theta)h + o(h)$$

where the left-hand side is composed of expectations that we can easily estimate through the standard Monte-Carlo estimator.

Definition 8 (First Difference Estimator)**.** *The first difference estimator is obtained by simulating n independent pathes $(X_1^{\theta}, \ldots, X_n^{\theta})$ at θ on $(t_1, \ldots, t_k)$ and n additional replications $(X_1^{\theta+h}, \ldots, X_n^{\theta+h})$ at $\theta + h$, $h > 0$. Based on the Taylor approximation we get the expression:*

$$\widehat{CVA^{'}_{\theta}}(n, h) = (1 - R_c) \sum_{i=1}^{k} \left(\frac{\widehat{EPE}(t_i, \theta + h) - \widehat{EPE}(t_i, \theta)}{h} \right) \big(\lambda(t_i) - \lambda(t_{i-1})\big)$$

where $\widehat{EPE}(t_i,\theta) = \frac{1}{n}\sum_{j=1}^{n} NPV_j(t_i,\theta)^+$. *In terms of convergence,*

$$\widehat{CVA'_\theta}(n,h) \xrightarrow{n\to\infty} \frac{CVA(\theta+h) - CVA(\theta)}{h} = CVA'(\theta) + \frac{1}{2}CVA''(\theta)h + o(h)$$

This estimator cumulates two errors, one in $o(n^{-\frac{1}{2}})$ *coming from the Monte-Carlo procedure, and the other one coming from the Taylor approximation. This second error is a statistical bias (i.e. a systematic error in the estimation):*

$$\mathbb{E}\left[\widehat{CVA'_\theta}(n,h)\right] - CVA'(\theta) = \frac{1}{2}CVA''(\theta)h + o(h) \tag{8}$$

The size of this second error depends on CVA's convexity and on the increment parameter h. *Based on the assumption that the pairs* $(X_i^\theta, X_i^{\theta+h})$ *are independent* $\forall i$,

$$\mathbb{V}\left[\widehat{CVA'_\theta}(n,h)\right] = (n^{-1}h^{-2}) \cdot \mathbb{V}\left[\Pi^i_{cva}(\theta+h) - \Pi^i_{cva}(\theta)\right] \tag{9}$$

where $\Pi^i_{cva}(\theta) = (1-R_c)\sum_{j=1}^{k} NPV_i(t_j,\theta)^+\big(\lambda(t_j)-\lambda(t_{j-1})\big)$ *represents CVA's "payoff", i.e. Credit Value Adjustement on a single path.*

This estimator is generic. As long as we know how to estimate EPE, we can compute it at two values of θ and get the CVA estimator. The estimation procedure *does not depend on the underlying portfolio of trades.* When computing CVA sensitivities at the counterparty level, this feature is of primary importance since the EPE typically aggregates many different products.

Looking at the variance in equation (9), we see that simulation settings will have an impact on the convergence rate of the estimator. Using independent simulations for $X^\theta_{t,1}$ and $X^{\theta+h}_{t,1}$ will produce

$$\mathbb{V}\left[\Pi^i_{\text{cva}}(\theta+h) - \Pi^i_{\text{cva}}(\theta)\right] \xrightarrow{h\to 0} 2\mathbb{V}[\Pi^i_{\text{cva}}(\theta)]$$

which is in $o(1)$. It is possible to reduce this term to $o(h)$ using common random numbers (i.e. taking the same seed $\omega = (\omega_1,\ldots,\omega_t)$ for the computation of pathes $X_1(\omega,\theta+h)$ and $X_1(\omega,\theta)$ rather than independent simulations). If $\Pi_{\text{cva}}(\theta)$ is almost surely continuous with respect to θ, the variance of the first difference estimator is further reduced to $o(h^2)$ (see [Gla04] for more details).

As discussed in [Gla04], the size of the stress parameter h has an impact on the statistical properties of the first difference estimator. In particular, when Π_{cva} isn't continuous in θ, decreasing h produce a smaller bias but a larger variance. In this case minimising the Mean Square Error provides an optimal h (which is a function of n). However as long as Π_{cva} is almost surely continuous in θ, there is no arbitrage and h should be as small as possible. As we've seen that CVA is a smooth operator, a small h will give better results.

2.2 Controlling the truncation error

The first-difference estimator gives an approximation of sensitivities based on simulation at θ and $\theta+h$. By simulating CVA(.) at additional points, it is possible to get estimators with smaller bias. Going back to the approximation in equation (7) and assuming that CVA is almost surely

C^k with $k > 2$ in θ, we can expand the Taylor development and try to remove part of the bias with some modified estimators. For instance, let's take two Taylor approximations in $\theta + h$ and $\theta - h$:

$$\mathrm{CVA}(\theta + h) = \mathrm{CVA}(\theta) + \mathrm{CVA}^{'}(\theta)h + \mathrm{CVA}^{''}(\theta)h^2/2 + o(h^2)$$
$$\mathrm{CVA}(\theta - h) = \mathrm{CVA}(\theta) - \mathrm{CVA}^{'}(\theta)h + \mathrm{CVA}^{''}(\theta)h^2/2 + o(h^2)$$

Subtraction eliminates the second order terms, leaving the *central approximation*:

$$\frac{\mathrm{CVA}(\theta + h) - \mathrm{CVA}(\theta - h)}{2h} = \mathrm{CVA}^{'}(\theta) + o(h)$$

Definition 9 (Central Difference Estimator)**.** *The central difference estimator is obtained by simulating n independent pathes $(X_1^{\theta}+h, \dots, X_n^{\theta}+h)$ at $\theta+h$ on $(t_1, \dots, t_k)$ and n additional replications $(X_1^{\theta-h}, \dots, X_n^{\theta-h})$ at $\theta - h$, $h > 0$. Based on the Taylor approximation we get the expression:*

$$\widehat{\widehat{CVA}}'_{\theta}(n, h) = (1 - R_c) \sum_{i=1}^{k} \left(\frac{\widehat{EPE}(t_i, \theta + h) - \widehat{EPE}(t_i, \theta - h)}{2h} \right) \big(\lambda(t_i) - \lambda(t_{i-1})\big)$$

This estimator has a lower bias than the first difference estimator since

$$\widehat{\widehat{CVA}}'_{\theta}(n, h) \xrightarrow{n\to\infty} \frac{CVA(\theta + h) - CVA(\theta - h)}{2h} = CVA'(\theta) + o(h)$$

Central difference removes the second order term in the approximation. As a result the gain of using central difference rather than first difference is huge when the second order term is large (i.e. when the gamma of CVA is large). Figure (3) illustrate this gain and its dependency to CVA's second order term. When the underlying portfolio is highly convex (for instance when looking at a European call option with small volatility and maturity), then using central difference is much more accurate (even when the number of simulation is corrected to keep a constant number of simulation points). With no convexity the gain dissapears. The second order term is small, and h must remain as small as possible to remove the bias in the Taylor approximation.

Based on this idea, we can add well chosen simulation points in our sample to remove higher order terms driving the size of the bias. Let's define $\Theta_{\mathrm{c}}(k)$ the central difference operator characterized by a stress of size kh. Using Taylor approximations at order $\tilde{n} = 2n + (n \mod 2)$, we get

$$\Theta_{\mathrm{c}}(k) := \frac{\mathrm{CVA}(\theta + kh) - \mathrm{CVA}(\theta - kh)}{2kh} = \sum_{i=0}^{n} \mathrm{CVA}(\theta)^{(2i+1)} \frac{(kh)^{2i}}{(2i+1)!} + o(h^{\tilde{n}})$$

Odd powers of h have been removed in $\Theta(k)$. We can now use linear algebra to find the linear combinations of $\Theta_{\mathrm{c}}(1), \dots, \Theta_{\mathrm{c}}(k)$ leading to an estimator in $o(h^{\tilde{n}})$. Using matrix formulation,

$$\overbrace{\begin{pmatrix} \Theta_{\mathrm{c}}(1) \\ \vdots \\ \Theta_{\mathrm{c}}(k) \end{pmatrix}}^{\bar{\Theta}_{\mathrm{c}}} = \overbrace{\begin{pmatrix} 1 & \frac{h^2}{3!} & \dots & \frac{h^{2n}}{(2n+1)!} \\ \vdots & \vdots & & \vdots \\ 1 & \frac{(kh)^2}{3!} & \dots & \frac{(kh)^{2n}}{(2n+1)!} \end{pmatrix}}^{H} \cdot \overbrace{\begin{pmatrix} \mathrm{CVA}^{(1)} \\ \vdots \\ \mathrm{CVA}^{(2n+1)} \end{pmatrix}}^{\bar{\alpha}(\theta)} + \overbrace{\begin{pmatrix} o(h^{\tilde{n}}) \\ \vdots \\ o(h^{\tilde{n}}) \end{pmatrix}}^{\bar{o}(h^{\tilde{n}})}$$

where H is a $(k \times (n+1))$ matrix. In practice, we need H to be invertible, so we set $k = n+1$. Finally:

$$\bar{\Theta}_{\mathrm{c}} = H\bar{\alpha}(\theta) + \bar{o}(h^{\tilde{n}}) \Rightarrow H^{-1}(\bar{\Theta}_{\mathrm{c}} - \bar{o}(h^{\tilde{n}})) = \bar{\alpha}(\theta)$$

The first line of H^{-1} gives the coefficients that can be used to combine linearly the components of $\bar{\Theta}_{\mathrm{c}}$. The resulting estimator has a reduced bias in $o(h^{\tilde{n}})$.

Example 2. *Taking for instance* $k = n+1 = 2$ $(\Rightarrow \tilde{n} = 2 \times 1 + (1 \mod 2) = 3)$*,*

$$H^{-1} = \begin{pmatrix} \frac{4}{3} & -\frac{1}{3} \\ -\frac{2}{h^2} & \frac{2}{h^2} \end{pmatrix} \Rightarrow \begin{pmatrix} \frac{4}{3} & -\frac{1}{3} \\ -\frac{2}{h^2} & \frac{2}{h^2} \end{pmatrix} \begin{pmatrix} \Theta_c(1) - o(h^3) \\ \Theta_c(2) - o(h^3) \end{pmatrix} = \begin{pmatrix} CVA^{(1)} \\ CVA^{(3)} \end{pmatrix}$$

So finally we get the following combined estimator

$$\frac{4}{3}\Theta_c(1) - \frac{1}{3}\Theta_c(2) = CVA^{(1)}(\theta) + o(h^3)$$

with a bias in $o(h^3)$ *whereas* $\Theta(1)$ *and* $\Theta(2)$ *have a bias respectively in* $o(h)$ *and* $o(h^2)$*. Extrapolation has almost removed the error coming from the Taylor approximation. The remaining error mostly comes from the Monte-Carlo approximation of expectations.*

2.3 Conclusion on the use of Taylor's approximation

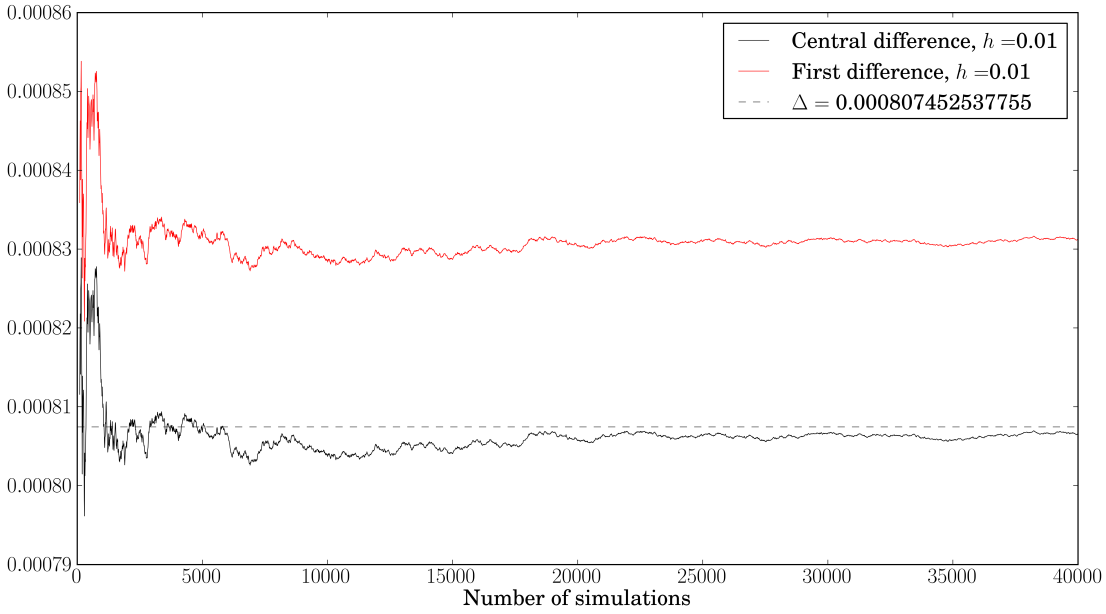
In practice the size of the bias rely on the convexity of the exposure with respect to θ. Overall, it is not easy to say if central difference and combined estimators have a smaller Mean Square Error on average. Central difference reduces the bias for convex products; CVA introduces an optionality and is convex at least for the delta. However, central difference comes at a cost, namely CVA's computation at $\theta + h$ on the whole simulation grid. For non convex exposures, central difference have a larger cost and no comparative advantage. Similarly, combined estimators have an increasing cost and, in most cases, no comparative advantage compared to central difference (high order terms in the Taylor development are usually small).

Optimizing the standard first difference approach seems difficult in a general framework where we don't have information on the general shape of the exposure profile. When producing a large number of sensitivities at once, it seems easier to use a first order approximation. However, if ones needs a very precise estimate of sensitivities, it might be interesting to investigate convexity and central difference.

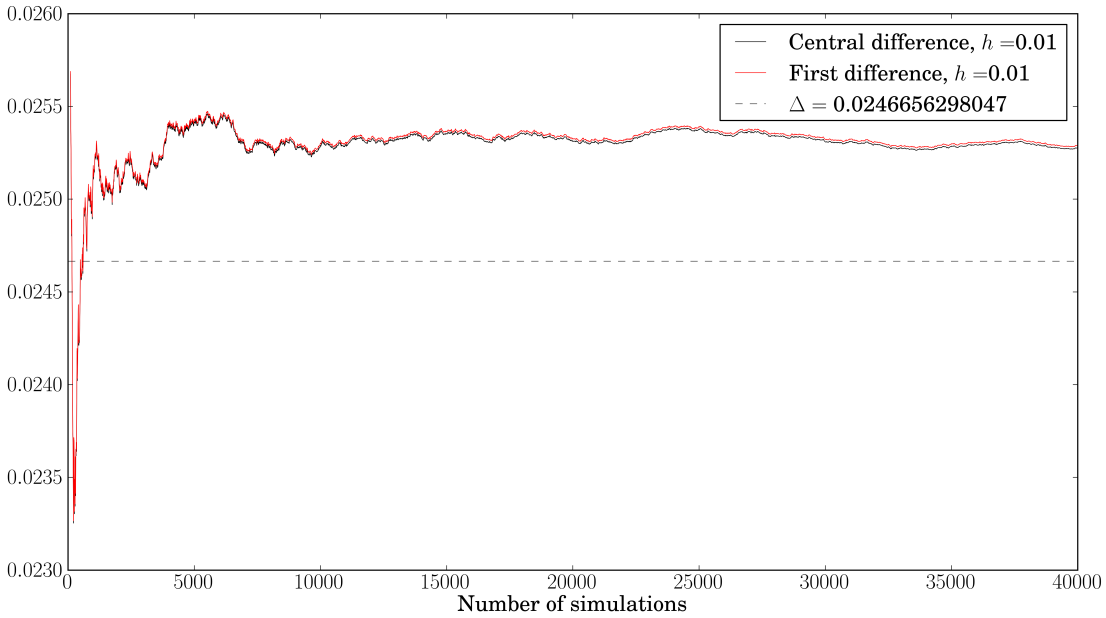
In any Monte-Carlo scheme, the underlying process is simulated using a known model, and we have information on its distribution. The next section presents methods taking part of these statistical properties.

Figure 3: Estimated Δ of CVA on a unique European Call Option in Black-Scholes (Strike=30, Spot/Strike=1, $r = 0$, Spread=100, $R = 0.4$)

(a) $\sigma = 0.01$, $T = 0.025$



(b) $\sigma = 0.1$, $T = 0.75$



3. Sensitivity estimation using weights methods

Weights methods rely on the density of diffusion processes. The central idea is to find the appropriate framework in which we will be able to write:

$$\begin{aligned} \mathrm{CVA}^{'}(\theta) &= (1-R_c)\int_0^T \frac{\partial}{\partial\theta}\mathbb{E}\big[B(0,s)\mathrm{NPV}(s,\theta)^+\big]d\lambda_c(s) \\ &= (1-R_c)\int_0^T \mathbb{E}\big[B(0,s)\mathrm{NPV}(s,\theta)^+ \times \pi(s,\theta)\big]d\lambda_c(s) \end{aligned}$$

where $\pi(s,\theta)$ is a random variable reffered to as a *weight*. The most interesting property of these methods is to produce generic estimators for any function of the underlying process, since weights depends on the underlying process but are independent of the integrand. In this area important results can be found in [Gla04], [FLL$^+$99] and [FLL$^+$99].

We first present the origin of weights using an intuitive framework. We then move to likelihood ratios and Malliavin weights.

3.1 General intuition

In a risk-neutral environment, the observed variations of prices can be interpreted as a change in the risk neutral measure $\mathcal{Q}$. Looking at the prices on $[t, t+\Delta t]$,

$$\begin{aligned} \textit{variation of prices} &= \textit{new price} - \textit{old price} \\ &= \mathbb{E}_{\mathcal{Q}_{t+\Delta t}}[\textit{payoff}] - \mathbb{E}_{\mathcal{Q}_t}[\textit{payoff}] \\ &= \mathbb{E}_{\mathcal{Q}_t}[\textit{payoff} \times \pi(t)] \end{aligned}$$

where the weight $\pi(t)$ is the sensitivity of the risk-neutral measure to the stressed parameter:

$$\pi(t) = \frac{d\mathcal{Q}_{t+\Delta t} - d\mathcal{Q}_t}{d\mathcal{Q}_t} \xrightarrow{\Delta t \to 0} \frac{\partial}{\partial t}\ln \mathcal{Q}_t$$

The weights are linked to the dynamic of the underlying process and their expression is not related to the payoff or the exposure we are looking at. In practice this means that they can be used to price sensitivities for any trade based on a given underlying. Compared to first difference, where every trade has to be priced at least twice, the computational gain might be substantial. In the context of a Monte-Carlo scheme and assuming a one-dimensional underlying, the following procedure could apply:

1. Simulate n independent replications of the underlying at T dates.
2. Compute the weights associated to the simulated underlying.
3. Compute the NPV of any portfolio based on the simulated underlying.
4. Compute the matrix product NPV $\times$ weight and average to estimate sensitivities.

The weight matrix provides a fast and generic way to compute sensitivities on any derivative based on the same underlying. In particular there is no need to re-run a Monte-Carlo procedure with stressed parameters. In multidimensional applications, weights methods also give sensitivities by tenor, whereas first difference only produce aggregated sensitivities.

Two methodologies have been developed in the literature to derive analytical formulas for the weights. The likelihood ratio method requires the konowledge of an explicit density, whereas the method developed in [FLL$^+$99] and [FLLL01], based on Malliavin's calculus, starts from a specific functional form for the diffusion process.

3.2 Likelihood ratio method

The Likelihood ratio method (LRM) provides an alternative approach to derivative estimation requiring no smoothness at all of integrande inside the expectation. It accomplishes this by differentiating probabilities rather than integrands, and relies on the absolute continuity of the diffusion's density.

Definition 10 (Likelihood Ratio estimator)**.** *Let's recall that $X \in \mathcal{L}^1(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathcal{P})$, and assume that $\mathcal{P} := \mathcal{P}(x,\theta) : \Omega \times \Theta \to [0,1]$. If $\mathcal{P}$ admits a density $f := f(x,\theta)$ with respect to the Lebesgue measure, and if f is absolutely continuous with respect to θ, then*

$$\begin{aligned}\frac{\partial}{\partial\theta}\mathbb{E}_\theta\big[B(0,t)NPV(t,X)^+\big] &= \frac{\partial}{\partial\theta}\int_\Omega B(0,t)NPV(t,x)^+ f(x,\theta)dx \\ &= \int_\Omega B(0,t)NPV(t,x)^+ \frac{f_\theta^{'}(x,\theta)}{f(x,\theta)} f(x,\theta)dx \\ &= \mathbb{E}_\theta\big[B(0,t)NPV(t,X)^+ \frac{\partial}{\partial\theta}\ln f(X,\theta)\big] \qquad (10)\end{aligned}$$

The random variable $\pi(X,\theta) = \frac{\partial}{\partial\theta}\ln f(X,\theta)$ is called "likelihood ratio" and must be analytically computed in each underlying model. The statistical estimator for n independent replications of the risk factor is thus given by

$$\widetilde{CVA_\theta^{'}}(n) = (1-R_c)\sum_{i=1}^{k}\left(\frac{1}{n}\sum_{j=1}^{n} B(0,t_i)NPV(t_i,X^j)^+ \frac{\partial}{\partial\theta}\ln f(X^j,\theta)\right)\big(\lambda(t_i)-\lambda(t_{i-1})\big)$$

This estimator is generic and unbiased, but its variance properties rely on the density function f and are difficult to predict in a general framework.

The main limitation of this method is that one has to know the explicit density function of the diffusion process. The next example provides an example of likelihood ratios computation is the uni-unidimensional Black-Scholes framework. In multi-dimensional applications we also need the factor's covariance matrix to be invertible. This assumption is problematic in important models such as the 3-factor HJM model, where only three factors are used to simulate many different rates.

Example 3 (Likelihood ratio with a unidimensional Black-Scholes underlying)**.** *In the unidimensional Black-Scholes framework with constant volatility σ and short rate r, we know from Ito's lemma that*

$$X_t = x_0 e^{(r-\frac{\sigma^2}{2})t+\sigma W_t} \qquad (11)$$

where x_0 is the spot value of the underlying. Consequently the distribution of the underlying is

$$X_t \sim \ln -\mathcal{N}\left(\ln x_0 + \left(r-\frac{\sigma^2}{2}\right)t\,;\,\sigma\sqrt{t}\right) \qquad (12)$$

In order to get the correct likelihood ratio, we then have to distinguish between the path-dependent and path-independent cases.

If we take a derivative with a path-independent payoff, we have

$$\frac{\partial}{\partial x_0}\mathbb{E}_{\mathcal{Q}}[\Pi(X_T)] = \mathbb{E}_{\mathcal{Q}}[\Pi(X_T)\frac{\partial}{\partial x_0}f_T(X_T, x_0)]$$

where $f_T(x)$ is the density function associated to the law of X_T given by equation (12)*. For the Δ, the resulting likelihood ratio is*

$$\frac{\partial}{\partial x_0}\ln f_T(X_T, x_0) = -\frac{1}{2}\frac{\ln\big(X_T/x_0\big) - (r - \frac{\sigma^2}{2})T}{x_0\sigma^2 T} = \frac{W_T}{x_0\sigma T}$$

When taking the derivative of the EPE at time t, we first have to look at the dependency of the NPV in t to the underlying process:

$$NPV(t) = \mathbb{E}_{\mathcal{Q}}[\Pi_X(t,T)|\mathcal{F}_t] = h(X_1, \ldots, X_t)$$

since the conditional expectation is the least-squares best predictors of $\Pi_X(t,T)$ using all the available information as of t. Hence we want to compute the likelihood ratio for the derivative

$$\frac{\partial}{\partial x_0}\mathbb{E}[B(0,t)NPV(t)^+] = \frac{\partial}{\partial x_0}\mathbb{E}[B(0,t)\,h(X_1, \ldots, X_t)^+]$$

The likelihood ratio must account for the density of the whole path $(X_1, \ldots, X_T)$. Since we simulate a finite number of dates, we can consider this last vector as a Markov chain path. Whereas Brownian motion paths have a null density, it is possible to compute the joint density of Markov Chains paths using transition densities. For a geometric Brownian motion, the transition density from state x_l to x_{l+1} is given by

$$G\big(x_l, x_{l+1}\big) = \big(x_{l+1}\sigma\sqrt{2\pi(t_{l+1} - t_l)}\big)^{-1}\exp\left\{-\frac{1}{2}\left(\frac{\ln(\frac{x_{l+1}}{x_l}) - (r - \frac{\sigma^2}{2})(t_{l+1} - t_l)}{\sigma\sqrt{t_{l+1} - t_l}}\right)\right\}$$

and the joint distribution of observations is

$$f(x_0, \ldots, x_T) = \prod_{l=0}^{T-1} G\big(x_l, x_{l+1}\big)$$

We have only one term in this last expression depending on the initial condition x_0, so taking the likelihood ratio for the Δ gives:

$$\frac{\partial}{\partial x_0}\ln f(x_0, \ldots, x_T) = \frac{\partial}{\partial x_0}\ln G\big(x_0, x_1\big) = \frac{\ln(x_1/x_0) - (r - \frac{\sigma^2}{2})t_1}{x_0\sigma^2 t_1} = \frac{W_1}{x_0\sigma t_1}$$

3.3 Sensitivity estimation with Malliavin's calculus

As exposed for the first time in [FLL$^+$99], it is possible to use Malliavin calculus to find generic estimators for sensitivities. Malliavin calculus develops an approach to stochastic differentiation that produces rules slightly different than the ones in Ito's calculus. With a general assumption on the dynamic followed by the underlying process, it is possible to analytically compute weights formulas using Malliavin's integration by part formula.

Definition 11 (Malliavin's estimator)**.** *Suppose we model a multidimensional underlying process using the following functionnal form:*

$$\begin{cases} dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t \\ x_0 \in \mathbb{R}^p \end{cases}$$

The first-difference process $\{Y_t\}_{t\leq 0}$ *associated to* $\{X_t\}_{t\leq 0}$ *is defined by the stochastic differential equation:*

$$\begin{cases} dY_t = \mu^{'}(t, X_t)Y_t dt + \sum_{i=1}^{p} \sigma_i^{'}(t, X_t)Y_t dW_t^i \\ Y_0 = I_p \in M_{p,p}(\mathbb{R}) \end{cases}$$

where $\mu^{'}(t, X_t)$ *is the Jacobian matrix associated to* $\mu(t, X_t)$*, and* σ_i' *is the Jacobian matrix associated to the i-th column of* $\sigma(t, X_t)$*.* Y_t *describes the dynamic of the variations of* X_t*. Then for any function* $\Pi(X) \in \mathcal{L}^2(\Omega, \mathcal{F}, \{\mathcal{F}_t\}, \mathcal{P})$*, [FLL+99] shows that*

$$\frac{d}{d\theta}\mathbb{E}_{\mathcal{Q}}[\Pi(X)] = \mathbb{E}_{\mathcal{Q}}[\Pi(X) \times \pi(X, \theta)]$$

where $\pi(X, \theta)$ *is a random variable called Malliavin weight. Closed weight formulas are given in [FLL+99] for derivatives with respect to the initial condition* x_0*, the drift function* $\mu(t, X_t)$ *and the volatility function* $\sigma(t, X_t)$*. For instance, taking* $\theta = x_0$ *gives*

$$\pi(X, x_0) = \int_0^T a(s)\sigma^{-1}(X_s)Y_s dW_s$$

where $a(t)$ *is any scaling factor such that* $\int_0^T a(t)dt = 1$*.*

Example 4 (Malliavin's Δ in the case of a geometric Brownian motion)**.** *We take a one-dimensional geometric Brownian motion for the underlying, i.e. we set*

$$\begin{cases} dX_t = \mu(t)X_t dt + \sigma(t)X_t dW_t \\ x_0 \in \mathbb{R}^+ \end{cases} \tag{13}$$

Using Ito's lemma the underlying can be simply written

$$X_t = x_0 \exp\Big\{ \int_0^t \big(\mu(s) - \frac{1}{2}\sigma^2(s)\big)ds + \int_0^t \sigma(s)dW_s \Big\}$$

In this case the first difference process associated to (X_t) *is simply* $Y_t = \frac{X_t}{x}$*, and the Malliavin's weight for the* Δ *has a simple expression:*

$$\pi^{\Delta} = \frac{1}{xT}\int_0^T \frac{1}{\sigma(s)}dW_s$$

Using Ito's lemma we get:

$$\int_0^T \frac{1}{\sigma(s)}dW_s = \frac{W_T}{\sigma(T)} - \frac{W_0}{\sigma(0)} + \int_0^T \frac{d}{ds}\left(\frac{1}{\sigma(s)}\right)W_s ds$$

Setting $W_0 = 0$ and assuming a constant volatility parameter $\sigma(t) = \sigma$, the weight simplifies to

$$\pi^{\Delta} = \frac{W_T}{x_0 \sigma T} \tag{14}$$

In this simple framework, we get the exact same weight as in the Likelihood ratio methodology in the case of path independent derivatives.

3.4 Statistical properties of weights estimators

Weights methods potentially brings huge savings in terms of computation power. Whereas first difference implies to price at least twice every portfolio on a regular basis, here one just have to compute a matrix of weights for each underlying. However, the statistical properties of weight estimators have to be investigated further.

Likelihood ratios and Malliavin weights are based on the same idea but start with different hypothesis. Whereas LRM requires knowledge of the underlying's density function, Malliavin's methodology specifies a functional form for its dynamic. The resulting weights have to be found analytically in LRM when Malliavin's method provides a general formula. However the two methods are very close: taking the example of the Black-Scholes framework, they provide the exact same estimator (see example 4). As a result they also have similar variance features:

(a) **Weights formulas do not depend on the integrand** Π_X. Only the underlying dynamic matters. Some important difficulties appear to find the analytical weight in complex models. For instance LRM weights can't be computed in most classic rates models (such as 3-factor HJM) since the covariance matrix of the underlying is not invertible (we simulate k tenors with only 3 factors, generally with $k > 3$, see [GZ99] for more details).

(b) But *the variance of weight estimators depend on* Π_X. Numerically, the balance between the value of the weight and the value of the integrand matters. Weights are functionals of the Wiener process used in the simulation. This dependency introduces an incompressible noise that might be amplified depending on the value that takes the integrand. In a nutshell, this feature is a first source of numerical instability.

(c) An other issue with weights comes from some parametric and model related instabilities. Going back to equation (14), we see that

$$\mathbb{V}\left[\Pi_X(t,T)\pi^{\Delta}(W_T^i)\right] = \left(\frac{1}{x_0 \sigma T}\right)^2 \mathbb{V}\left[\Pi_X(t,T)W_T^i\right]$$

The mean square error of the weight estimator is proportional to $(x\sigma T)^{-2}$; this term will be very large for small maturities and small volatilities. If it is possible to control this error using techniques such as antithetic variates (see [Cap08]), our general approach suffers from this additional instability.

Figure 4 displays the results of our estimations for the Δ of CVA in the Black-Scholes framework. The portfolio here is only made of a single European call option. The weight estimator is very noisy, and illustrates the numerical instability described earlier; convergence is slow and the incompressible noise is visible. On the other hand the first difference estimator converges in $n^{-1/2}$ to a (somehow biased) value.

In practice, for a given number of simulation, weight estimators provide a variance reduction versus first difference **only when the integrand** Π **is discontinuous**. In this case, Taylor's

approximation gives very poor results and first difference is very bad whereas weights estimators are not affected. Figure (5) displays the results of our estimations for the Δ of a European call option in the Black-Scholes framework. Once again, in the standard case where the integrand is continuous with respect to θ, weight estimators are very noisy whereas first difference converges faster to the theoretical value of the Δ. However in the case of a digital option first difference is very unstable and doesn't converges anymore. In this case, weights provides better results.

Figure 4: Estimation of CVA's Δ (portfolio: single European Call) in Black-Scholes (Spot = 30, Spot/Strike = 1, $r = 0.01$, $\sigma = 0.1$, $T = 5$)

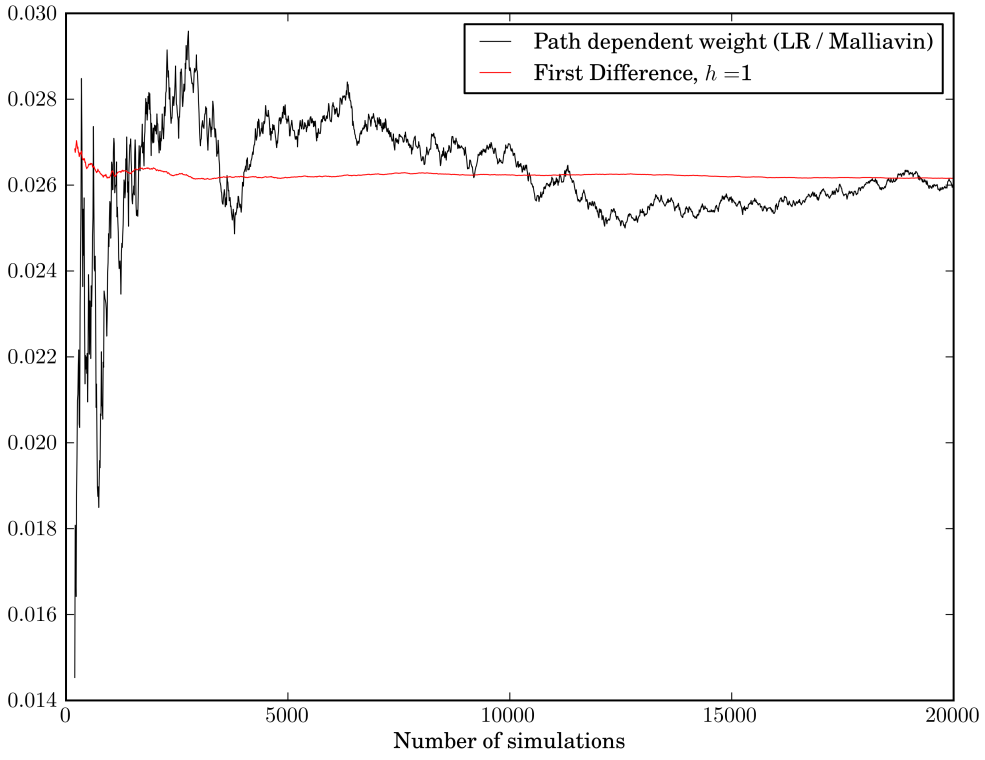
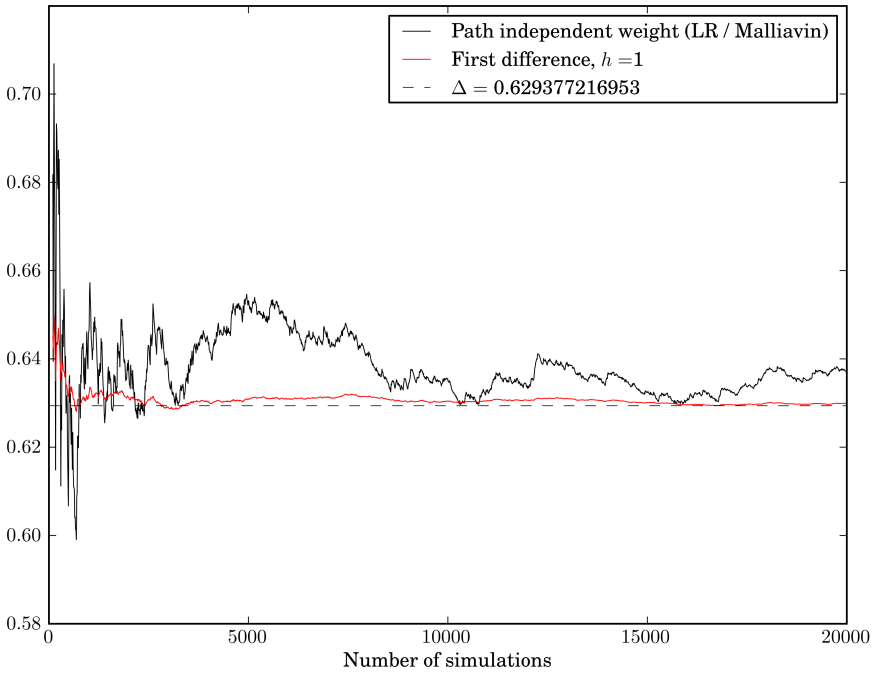


Figure 5: Estimation of the Δ in Black-Scholes for continuous and discontinuous payoff

(a) European call's Δ (Spot $= 30$, Spot/Strike $= 1$, $r = 0.01$, $\sigma = 0.1$, $T = 5$)



(b) Digital Option's Δ (Spot $= 30$, Payoff: , $r = 0.01$, $\sigma = 0.2$, $T = 1$)

Conclusion

Sensitivities are crucial quantities for trading desks that need to hedge their positions. Most underlying models are now based on complex multi-dimensional SDEs, and sensitivities have to be priced using Monte-Carlo methods since no closed formula is available. As a result, sensitivity estimation became a classic issue in quantitative finance. In recent years, the literature focused on methodologies adapted to exotic options. In particular this need came from the inefficient first difference estimation in the case of discontinuous payoffs.

Managing counterparty risk at the counterparty level requires the computation of sensitivities on large heterogeneous portfolios. The most general approach to sensitivity computation is finite difference. Advanced techniques such as path-wise differentiation (see [Gla04]), vibrato Monte-Carlo (see [Gil09]), multilevel Monte-Carlo (see [BG12]), or the use of the Fourier transform (see [EGP10]) can't be used because they produce estimators specific to a payoff function (i.e. that aren't generic). Here we have investigated the properties of weights estimators, that verify the generic property.

Weights estimators have a low computation cost (since each derivative in the portfolio only has to be priced once and weights are only related to the underlying), but they have poor variance properties unless the payoff function is discontinuous. Furthermore, analytical weights are difficult to obtain in most models; they do not even exist in some multi-dimensional applications such as the 3-factor Heath-Jarrow-Morton framework. We conclude that these methods are not adapted to the computation of sensitivities on large heterogeneous portfolios, hence they are not adapted for the computation of CVA sensitivities.

Finite difference is a general method and provides good statistical results for smooth variables. As CVA is smooth with respect to most parameters of the underlying, this method is adapted. Despite a large computational cost, this is the best method available in the context of large heterogeneous portfolios. Consequently, optimization should concentrate on efficient simulation algorithms for the EPE.

References

[BG12] Sylvestre Burgos and Michael B Giles. Computing greeks using multilevel path simulation. pages 281–296, 2012.

[Cap08] Luca Capriotti. Reducing the variance of likelihood ratio greeks in monte carlo. pages 587–593, 2008.

[EGP10] Ernst Eberlein, Kathrin Glau, and Antonis Papapantoleon. Analysis of fourier transform valuation formulas and applications. *Applied Mathematical Finance*, 17(3):211–240, 2010.

[FLL+99] Eric Fournié, Jean-Michel Lasry, Jérôme Lebuchoux, Pierre-Louis Lions, and Nizar Touzi. Applications of malliavin calculus to monte carlo methods in finance. *Finance and Stochastics*, 3(4):391–412, 1999.

[FLLL01] Eric Fournié, Jean-Michel Lasry, Jérôme Lebuchoux, and Pierre-Louis Lions. Applications of malliavin calculus to monte-carlo methods in finance. ii. *Finance and Stochastics*, 5(2):201–236, 2001.

[Gil09] Michael B Giles. Vibrato monte carlo sensitivities. pages 369–382, 2009.

[Gla04] Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer, 2004.

[Gre10] Jon Gregory. *Counterparty credit risk: the new challenge for global financial markets*, volume 470. John Wiley & Sons, 2010.

[GZ99] Paul Glasserman and Xiaoliang Zhao. Fast greeks by simulation in forward libor models. *Journal of Computational Finance*, 3(1):5–39, 1999.

[Ing13] Stephan Ingves. Regulatory reforms for otc derivatives: past, present and future. *Financial Stability Review*, page 19, 2013.

[LeV98] Randall J LeVeque. Finite difference methods for differential equations. *Draft version for use in AMath*, 585(6), 1998.

[LS01] Francis A Longstaff and Eduardo S Schwartz. Valuing american options by simulation: A simple least-squares approach. *Review of Financial studies*, 14(1):113–147, 2001.

Appendices

A. Presentation of the module

This document presents the functions and classes of the Python module *"module_ct_risk_v05.py"*. From now on we will refer to the module as *ct_risk*. This module has been developed in the context of my project on the optimization of CVA sensitivities. It is very simple but can be used as a source of inspiration for future developments within Counterparty Team using Python. The main objectives of this module are as follow:

1. Simulate n replications of a unidimensional Geometric Brownian Motion process on a set of discrete dates $(t_1, \ldots, t_k)$.
2. Simulate the future NPV of a European Call option as defined in the Black-Scholes-Morton framework.
3. Compute counterparty credit risk measures: EE, EPE, ENE, PFE.
4. Estimate default probabilities from credit spreads using the standard model (default events modeled as Poisson process).
5. Compute CVA on a given deal / counterparty.
6. Compute collateral calls on a given deal / counterparty.
7. Compute the greeks using different estimators (first difference, weights, etc.)

The theoretical framework is the Black-Scholes-Morton (BS) model.

1.1 Python modules

The first step is to load useful Python modules:

```
import numpy as np # linear algebra
import scipy.stats as st # statistical functions
import pylab as pl # plotting library
import time # time features
```

Pandas is the most useful module since it proposes very efficient classes and methods to manage series (class pandas.Series), dataframes (pandas.DataFrame), time index (pandas.DatetimeIndex), panels (pandas.Panel). Within *ct_risk*, we deal a lot with dataframes where the lines represents the dates $(t_1, \ldots, t_k)$ and the columns represent the simulations $(1, \ldots, n)$:

$$\begin{pmatrix} x_{t_1}^1 & \ldots & x_{t_1}^n \\ \vdots & & \vdots \\ x_{t_k}^1 & \ldots & x_{t_k}^n \end{pmatrix}$$

In order to well represent this kind of grids, I designed a specific class called *ct.PV_grid* which inherits from the super class *pandas.DataFrame*. This class adds convenient attributes and methods.

class *ct.**PV_grid**(data=None, index=None, columns=None, dtype=None, copy=False, base=365.25)*

Attributes:

- data: an instance of pandas.DataFrame or numpy.array.

- index: a datetimeindex.
- columns: columns index (the index of the simulation).
- tfs: time for start, i.e. the vector of time deltas $(t_1 - t_0, \ldots, t_k - t_0)$ where $t_0 = 0$.
- risk_metrics: a grid containing the output of the method *get_risk_metrics*.

Methods:

- *get_risk_metrics*(pfe_level=0.99, VaR_level=0.05)
  Takes the data attribute and returns a pandas.DataFrame containing the EE, EPE, ENE, PFE, median, VaRs.
- *get_collateral*(counterparty=None)
  Computes the collateral amount associated to the counterparty.
- *plot_sample*(save_path=None)
  Takes three random columns in the data and plots them. Also displays the standard deviation of the data through time.

The other packages are used less frequently. *Numpy* is designed to handle linear algebra and complex operations on n-dimensional arrays. *Scipy.stats* contains statistical functions and is used to get the values of $\mathcal{N}(d_1)$ and $\mathcal{N}(d_2)$ in the Black-Scholes pricing formulas. *Pylab* is a plotting library and *time* is used to get today's date.

1.2 Risk factor generation

The first step in our analysis is to define a set of futures dates and a risk factor to be diffused on n paths for these dates. For that we define the following class based on the Black-Scholes framework:

class *ct.**Black_Scholes_Underlying**(datetimeindex, spot=1, mean=0, vol=0.01, base=365.25)*

Attributes:

- datetimeindex: an object of the class pandas.DatetimeIndex.
- spot: the spot value x_0 of the underlying in the BS.
- mean: the drift μ of the underlying under the physical measure.
- vol: the constant volatility σ of the underlying.
- base: the scaling factor for timedeltas.
- data: stores the last simulated DataFrame.
- seed: a dataframe containing the paths of the Wiener process W_t used in simulations.

Methods:

- *simulate(sim_number=100, new_seed=True, spot=None)*
  Returns a *PV_grid* containing n simulations of the underlying. The simulated equation is:

$$X_t = x_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma W_t}$$

- *append_datetimeindex(self, datetimeindex=None, replace=False)*
  Adds dates to the existing datetimeindex.

We choose the Black-Scholes framework for its simplicity. In this model we have closed formulas to price vanilla derivatives such as European call and put options, binary options, and forwards.

1.3 Pricing of forward PVs

The second step in the script consists in the definition of pricers. The main difficulty here is to price derivatives on each path and for each date in the future:

$$\mathrm{NPV}^i(t) = \mathbb{E}_{\mathcal{Q}}\left[\pi(X_T^i)|\mathcal{F}_t^i\right]$$

To avoid a nested Monte-Carlo scheme, we consider products for which we have a theoretical closed formula. We begin to define a super class called *ct.Risk_ Neutral_ Pricer* containing the attributes and methods shared by every pricers. We then define a sub-class (that inherits from all the methods and attributes of the super class) called *ct.Euro_ Call_ Pricer*, designed to price European call options. Obviously other subclasses such as *ct.Euro_ Put_ Pricer*, *ct.Forward_ Pricer* or *ct.Binary_ Pricer* could also be implemented using the same methodology.

class *ct.**Risk_ Neutral_ Pricer**(underlying = default_ underlying, short_ rate = 0, strike = None)*

Attributes:

- underlying: a ct.Black_Scholes_Underlying object.
- datetimeindex: the datetimeindex of the underlying.
- short_rate: the short rate r.
- strike: the strike K of the option.
- discount_curve: the discount curve implied by the dates and the short rate.
- nd1: the value of $\mathcal{N}(d_1)$ implied by model parameters.

Methods:

- *set_ underlying*(underlying=None)
  Replaces the underlying by a new one.
- *get_ bs_ dx*()
  Compute the values of $d1$, $d2$, $\mathcal{N}(d_1)$, $\mathcal{N}(d_2)$.

class *ct.**Euro_ Call_ Pricer**(underlying = default_ underlying, short_ rate = 0, strike = None)*

Attributes:

- Inherited attributes frome the super-class ct.Risk_Neutral_Pricer.
- npv: a PV_grid containing the simulated NPVs.

Methods:

- *get_ forward_ npv*(sim_number = 100, new_seed = True, spot = None)
  Computes the price of a European Call option for every scenario $i \in (0, \ldots, n)$

  $$C_t = \mathbb{E}[e^{-r(T-t)}(X_T^i - K)^+|\mathcal{F}_t^i] \quad \forall t \in (t_1, \ldots, t_k)$$

  The pricing formula is the one from BS:

  $$C_t = X_t\mathcal{N}(d_1) - Ke^{-r(T-t)}\mathcal{N}(d_2)$$

- *get_ spot_ greeks*(sim_number=100, new_seed=False, spot=None)
  Returns the spot values of the delta and of the gamma.

1.4 Counterparty risk computation

The third part of the script defines counterparties and derivatives. We make the assumption that a counterparty can be represented by a single class and a set of characteristics: a CDS spread,

a GRR, and CSA parameters.

class *ct.**Counterparty**(cds_spread = 100, grr = 0.4, threshold = 0, mta = 0, independant_amount = 0, margin_period = '60D')*

Attributes:

- cds_spread: the cds spread in bps.
- grr: the recovery rate of the counterparty expressed in absolute value.
- threshold: the threshold attached to the counterparty.
- mta: minimum transfer amounts.
- independant_amounts.
- margin_periods: time interval between two margin calls.

Methods:

- *get_margin_call_datetimeindex*(datetimeindex)
  Returns a datetimeindex containing margin call dates. the input datetimeindex is used to determine the spot date and the maturity of the derivative.
- *get_default_probability*(datetimeindex, base=365.25)
  Returns the cumulative default probability implied by the credit spread. We used the model:

$$\lambda_c(t) = \mathcal{P}\Big(\{\tau_c \leq t\}\Big) = 1 - e^{-\frac{Z_c}{1-R_c}t}$$

  where Z_c is the x-years cds spread of the counterparty.

The class *ct.Derivative* links all the previously defined classes together. It is defined by an underlying, a pricer, and a counterparty. Once the *simulate*() method has been run, it is easy to use this class to get counterparty risk metrics and CVA.

class *ct.**Derivative**(underlying = default_underlying, pricer = default_pricer, counterparty = default_counterparty)*

Attributes:

- pricer: an object of the class ct.Euro_Call_Pricer.
- underlying: an object of the class ct.Black_Scholes_Underlying.
- counterparty: an object of the class ct.Counterparty.
- npv: a PV_grid containing the simulated NPVs.

Methods:

- *simulate*(sim_number = 100, new_seed = True, spot = None, collateral = False)
  Simulate the underlying values, compute future NPVs using the pricer attribute, and fills the npv attribute.
- *compute_cva*()
  Computes cva using the npv and counterparty attributes.
- *plot_risk_metrics*()
  Plot the EE, EPE, ENE and PFE.

1.5 Example

Here is a short example of a script using the module. The first step is to import modules:

```
import module_ct_risk_v05 as ct
```

```
import pandas as pd # time series see http://pandas.pydata.org/
import time # time features see https://docs.python.org/2/library/time.html
import pylab as pl

##########################################################################
##################### FIRST DIFFERENCE WITH THE CT MODULE #################
##########################################################################

# initialisation of the underlying, pricer, counterparty, derivative objects

dates = pd.DatetimeIndex(start=time.strftime("%d/%m/%Y"), freq='1D', periods=30)
underlying = ct.Black_Scholes_Underlying(datetimeindex = dates, spot = 1, mean = 0.0, vol
    = 0.1, base = 365.25)
pricer = ct.Euro_Call_Pricer(underlying = underlying, short_rate = 0.0, strike =
    underlying.spot)
counterparty = ct.Counterparty(cds_spread = 100, grr = 0.4, threshold = 0, mta = 0,
    independant_amount = 0, margin_period = '60D')
trade = ct.Derivative(underlying = underlying, pricer = pricer, counterparty =
    counterparty)
pv_grid1 = trade.simulate(sim_number = 20000, new_seed = True, spot = underlying.spot,
    collateral = False)

# ploting samples to check the simulations are right

trade.underlying.data.plot_sample()
trade.npv.plot_sample()
trade.plot_risk_metrics()

# compute values of the first difference estimator

pv_grid2 = trade.simulate(sim_number = 20000, new_seed = False, spot = 1.0001*underlying.
    spot , collateral = False)
first_difference_grid = ((pv_grid2 - pv_grid1) / 0.0001*underlying.spot)#.mul(pricer.
    discount_curve, axis=0)

sim = [x for x in range(100,first_difference_grid.shape[1]) if x%10==0]
delta = pd.Series(index=sim)

for i in sim:

   delta.ix[i] = first_difference_grid.ix[-1,:i].mean()

#  plot the convergence of the first difference estimator using matplotlib library

fig, ax = pl.subplots(nrows=1, ncols=1, sharex=True)
fig.subplots_adjust(left=None, bottom=None, right=None, top=0.9, wspace=0.2, hspace=None)
ax.set_title('Estimation of the delta on a European call Option')
ax.set_xlabel('Number of simulations', fontsize=11)
ax.plot(sim,delta, color='red', label='First Difference Estimator', linestyle='-',
    linewidth=0.5)
ax.axhline(pricer.get_spot_greeks()[0], color='black', label='Black-Scholes Delta',
    linestyle='dashed', linewidth=0.5)
ax.legend(loc=0, prop={'size':11})
ax.tick_params(axis='both', labelsize=11)
pl.show()
```

B. Script - Counterparty Risk module for Python

```
#########################################################
################## CT RISK MODULE V05 ##################
#########################################################

# This module contains classes and methods to :
# 1. Simulate a unidimensionnal Geometric Brownian Motion process on [0,T].
# 2. Simulate the future NPV of a European Call option as defined in the Black-Scholes-Morton
    framework.
# 3. Compute counterparty credit risk measures: EE, EPE, ENE, PFE.
# 4. Estimate default probabilities from credit spreads using the standard model (default events
     modelled as Poisson process).
# 5. Compute CVA on a given deal / counterparty.
# 6. Compute collateral calls on a given deal / counterparty.

# The theoretical framework is Black-Scholes-Morton with a risky and a non risky asset.

#########################################################
################## 00 PACKAGES  ########################
#########################################################

import numpy as np
import pandas as pd
import scipy.stats as st
import pylab as pl
import time
import os

#########################################################
################## 01 GENERIC FUNCTIONS ################
#########################################################

def compute_integral(integrande, integrator, axis=0):

   if axis==1:
      integrande = integrande.T

   dintegrator = (integrator - integrator.shift(1)).shift(-1)
   integral = integrande.mul(dintegrator,axis=0).ix[::-1,:].cumsum(axis=0).ix[::-1,:]

   return(PV_grid(integral, index=integrande.index, columns=integrande.columns))

def compute_scaled_timedeltas(datetimeindex, referenceindex, base, reverse=False):

   """compute the length between timesteps of a DatetimeIndex
   scaled by the base parameter
   returns a pandas.series indexed by datetimeindex"""

   timedelta_array = (np.array(pd.Series(datetimeindex)-referenceindex).astype('timedelta64[D]')
        / np.timedelta64(1, 'D')) / base

   if reverse==True:
      timedelta_array = timedelta_array[::-1]

   return(pd.Series(timedelta_array, index=datetimeindex))

#########################################################
################## 02 RISK FACTOR GENERATION ############
#########################################################
```

```
default_datetimeindex = pd.DatetimeIndex(start=time.strftime("%d/%m/%Y"), freq='30D', periods
    =60) # 60 periods of 30 days = 5y

class Black_Scholes_Underlying(object):

   """docstring for Black_Scholes_Underlying"""

   def __init__(self, datetimeindex=default_datetimeindex, spot=1, mean=0, vol=0.01, base
       =365.25):

      self.datetimeindex = datetimeindex
      self.spot = spot
      self.mean = mean
      self.vol = vol
      self.base = base
      self.data = None
      self.seed = None
      self.tfs = compute_scaled_timedeltas(datetimeindex, datetimeindex[0], base, reverse=False)
      self.ttm = abs(compute_scaled_timedeltas(datetimeindex, datetimeindex[-1], base, reverse=
          False))

      def __repr__(self):

         print(type(self))
         return repr(self.data)

   def simulate(self, sim_number = 100, new_seed=True, spot=None):

      if spot is not None:
         self.spot = spot

      if new_seed==True:
         Z = st.norm.rvs(size=self.datetimeindex.shape+(sim_number,),scale=1)
         self.seed = pd.DataFrame(Z,index=self.datetimeindex,
            columns=['path n'+str(x) for x in range(sim_number)]).mul(np.sqrt(self.tfs-self.tfs.
                shift(1).fillna(value=0)),axis=0).cumsum(axis=0)

      elif new_seed==False:

         self.seed = self.seed.ix[:sim_number,:]

      self.data = self.spot*np.exp(self.seed.ix[:,:sim_number].mul(self.vol, axis=0).add((self.
          mean-0.5*(self.vol**2))*self.tfs,axis=0))

      self.seed = PV_grid(self.seed, index = self.datetimeindex, columns = self.seed.columns)
      self.data = PV_grid(self.data, index = self.datetimeindex, columns = self.seed.columns)

      return(self.data)

   def append_datetimeindex(self, datetimeindex=None, replace=False):

      if datetimeindex is not None:

         if replace==True:
            self.datetimeindex = datetimeindex
         else:
            self.datetimeindex = self.datetimeindex.union(datetimeindex)
         self.tfs = compute_scaled_timedeltas(self.datetimeindex, self.datetimeindex[0], self.
             base, reverse=False)
         self.data = self.data.reindex(self.datetimeindex)
```

```
################################################################
################## 03 FUTURE NPV PRICING ######################
################################################################

default_underlying = Black_Scholes_Underlying()

class Risk_Neutral_Pricer(object):

   """docstring for Risk_Neutral_Pricer"""

   def __init__(self, underlying = default_underlying, short_rate = 0, strike = None):

      self.underlying = underlying
      self.datetimeindex = underlying.datetimeindex
      self.short_rate = short_rate
      self.discount_curve = pd.Series(np.exp(-short_rate*self.underlying.tfs), index = self.
          underlying.datetimeindex)
      self.nd1 = None

      if strike is None:
         self.strike = underlying.spot
      elif strike is not None:
         self.strike = strike

   def set_underlying(self,underlying=None):

      if underlying is not None:
         self.underlying = underlying
         self.discount_curve = pd.Series(np.exp(-self.short_rate*underlying.tfs), index =
             underlying.datetimeindex)

   def get_bs_dx(self):

      if self.underlying.data is None:
         print('Empty: use Black_Scholes_Underlying.simulate()')
         return None

      self.d1 = (np.log(self.underlying.data/self.strike).add(self.short_rate+0.5*(self.
          underlying.vol**2)*self.underlying.ttm, axis=0)).mul(1/(self.underlying.vol*np.sqrt(
          self.underlying.ttm)), axis=0)
      self.d2 = self.d1 - self.underlying.vol*np.sqrt(self.underlying.ttm)
      self.nd1 = pd.DataFrame(st.norm.cdf(self.d1), index = self.datetimeindex, columns = self.
          underlying.seed.columns)
      self.nd2 = pd.DataFrame(st.norm.cdf(self.d2), index = self.datetimeindex, columns = self.
          underlying.seed.columns)

class Euro_Call_Pricer(Risk_Neutral_Pricer):

   """docstring for Euro_Call_Pricer"""

   def __init__(self, underlying = default_underlying, short_rate = 0, strike = None):

      super(Euro_Call_Pricer, self).__init__(underlying = underlying, short_rate = short_rate,
          strike = strike)
      self.npv = None

   def get_forward_npv(self, sim_number = 100, new_seed = True, spot = None):

      self.underlying.mean = self.short_rate # simulation sous la mesure risque neutre
      self.underlying.simulate(sim_number, new_seed, spot)
      self.get_bs_dx()
```

```
      self.npv = (self.underlying.data.values * self.nd1).add(-(self.strike * self.nd2).mul(self
          .discount_curve, axis=0), axis=0)
      self.npv = PV_grid(self.npv, index = self.datetimeindex, columns = self.underlying.seed.
          columns)

      return(self.npv)

   def get_spot_greeks(self, sim_number=100, new_seed=False, spot=None):

      if self.nd1 is None:
         self.underlying.simulate(sim_number, new_seed, spot)
         self.get_bs_dx()

      self.delta = self.discount_curve.ix[-1]*self.nd1.ix[0,0]
      self.gamma = self.nd1.ix[0,0]*self.discount_curve.ix[-1]*(1/(self.underlying.spot*self.
          underlying.vol*np.sqrt(self.underlying.ttm.ix[0])))

      return(pd.Series([self.delta,self.gamma], index = ['Spot Delta', 'Spot Gamma']))

#####################################################################
################## 04 CT RISK COMPUTATION  ######################
#####################################################################

class Counterparty(object):
   """docstring for Counterparty"""
   def __init__(self, cds_spread = 100, grr = 0.4, threshold = 0, mta = 0, independant_amount =
       0, margin_period = '60D'):

      self.cds_spread = cds_spread
      self.grr = grr
      self.threshold = threshold #expressed in percentage of the PV
      self.mta = mta
      self.independant_amount = independant_amount
      self.margin_period = margin_period

   def get_margin_call_datetimeindex(self, datetimeindex):

      self.margin_call_datetimeindex = pd.DatetimeIndex(start=datetimeindex[0], end=
          datetimeindex[-1],freq=self.margin_period)
      return(self.margin_call_datetimeindex)

   def get_default_probability(self, datetimeindex, base=365.25):

      tfs = compute_scaled_timedeltas(datetimeindex, datetimeindex[0], base, reverse=False)
      prob = 1-np.exp((-0.0001*self.cds_spread/(1-self.grr))*tfs)
      return(prob)

default_pricer = Euro_Call_Pricer()
default_counterparty = Counterparty()

class Derivative(object):
   """docstring for Derivative"""
   def __init__(self, underlying = default_underlying, pricer = default_pricer, counterparty =
       default_counterparty):

      self.pricer = pricer
      self.pricer.set_underlying(underlying)
      self.underlying = underlying
      self.counterparty = counterparty
      self.npv = None

   def simulate(self, sim_number = 100, new_seed = True, spot = None, collateral = False):
```

```
      if collateral == True:
         mc_dates = self.counterparty.get_margin_call_datetimeindex(self.underlying.
              datetimeindex)
         self.underlying.append_datetimeindex(mc_dates)

      self.npv = self.pricer.get_forward_npv(sim_number = sim_number, new_seed = new_seed, spot
          = spot)

      if collateral == True:
         self.collat = self.npv.get_collateral(self.counterparty)
         self.npvc = self.npv-self.collat

      return(self.npv)

   def compute_cva(self):

      if self.npv is None:

         print('Empty: use Derivative.simulate()')
         return None

      dp = self.counterparty.get_default_probability(self.underlying.datetimeindex)
      integral = compute_integral(self.npv.mul(self.pricer.discount_curve, axis=0)*(self>0), dp)
      cva = (1-self.counterparty.grr)* integral
      return(cva)

   def plot_risk_metrics(self):

      if self.npv is None:
         print('Empty: use Derivative.simulate()')
         return None

      self.npv.get_risk_metrics().ix[:,:4].plot()
      pl.show()

########################################################################
################## 05 CONVENIENT CLASS  ##############################
########################################################################

class PV_grid(pd.DataFrame):
   """docstring for PV_grid"""

   def __init__(self, data=None, index=None, columns=None, dtype=None, copy=False, base=365.25):
      super(PV_grid, self).__init__(data, index, columns, dtype, copy)
      self.tfs = compute_scaled_timedeltas(self.index, self.index[0], base, reverse=False)
      self.risk_metrics = None

   def get_risk_metrics(self, pfe_level=0.99, VaR_level=0.05):

      metrics_names = ['Expected Exposure', 'Expected Positive Exposure', 'Expected Negative
          Exposure', 'PFE ('+str(pfe_level)+'%)', 'Median',  'VaR ('+str(100-VaR_level)+')', '
          VaR ('+str(VaR_level)+')','VaR_tilde ('+str(100-VaR_level)+')', 'VaR_tilde ('+str(
          VaR_level)+')' ]
      ee = self.mean(axis=1)
      epe = (self*(self>0)).mean(axis=1)
      ene = (self*(self<0)).mean(axis=1)
      pfe = self.quantile(pfe_level, axis=1)
      median = self.median(axis=1)
      VaR_sup = (self-self.shift(1)).quantile(1-VaR_level, axis=1)
      VaR_inf = (self-self.shift(1)).quantile(VaR_level, axis=1l)
```

```
        VaR_tilde_sup = (self-self.shift(1)).quantile(1-VaR_level, axis=1).mul(1/(self.tfs-self.
            tfs.shift(1)))
        VaR_tilde_inf = (self-self.shift(1)).quantile(VaR_level, axis=1).mul(1/(self.tfs-self.tfs.
            shift(1)))

        self.risk_metrics = pd.concat([ee, epe, ene, pfe, median, VaR_sup, VaR_inf, VaR_tilde_sup,
             VaR_tilde_inf],axis=1)
        self.risk_metrics.columns = metrics_names

        return(self.risk_metrics)

    def get_collateral(self, counterparty=None):

        collat_flow = pd.DataFrame(np.zeros(self.data.shape),index=self.data.index,columns=self.
            data.columns)
        pv_above_thr = np.sign(self.data)*(np.abs(self.data) -
           self.csa.threshold*self.nominal)*(np.abs(self.data) > self.csa.threshold*self.nominal)+
                self.csa.independant_amount

        last_call = self.csa_dates[0]
        for t in self.csa_dates:

           current_collateral = collat_flow.cumsum(axis=0).T[last_call]
           collat_flow.ix[t,:] =  pv_above_thr.ix[t,:] - current_collateral
           collat_flow.ix[t,:] *= (np.abs(pv_above_thr.ix[t,:] - current_collateral)>self.csa.mta)
           last_call = t

        self.collateral = PV_grid(collat_flow.cumsum(axis=0), index=self.datetimeindex, columns=
            self.data.columns)
        self.margin_calls = (self.collateral - self.collateral.shift(1))*(self.data<0)*(self.
            collateral < self.collateral.shift(1))
        return(self.collateral)
```

Résumé de la mission

Cette note présente les missions de l'équipe que j'ai intégré, les sujets de recherche qui m'ont été proposés, ainsi que ma méthodologie de recherche.

1. Le risque de contrepartie au sein du groupe BNP-Paribas

1.1 La gestion des risques liés aux marchés de capitaux: Risk-IM

Intégré à la division "Group Risk Management" de BNP Paribas, le département "Risk-Investment & Markets" (Risk-IM) supervise tous les risques liés aux activités de la banque sur les marchés de capitaux. Son domaine de compétence couvre les risques de marché et de liquidité, le risque de contrepartie (transactions de produits dérivés et transaction "REPO"), le risque de crédit, et les risques liés aux activités d'assurance.

Les départements de gestion des risques se sont récemment développés au sein des institutions financières, afin d'accompagner la forte croissance des activités sur les marchés de produits dérivés, source de risques complexes. Risk-IM est la structure spécialisée de BNP-Paribas et rassemble des analystes de risque, des équipes modèles ("quants"), et des informaticiens. Cette entité a deux responsabilités essentielles:

- La définition d'un cadre et d'outils adaptés au calcul et à la mesure des risques. Les équipes modèles s'occupent de définir et calibrer des modèles pertinents (modèles de diffusion des facteurs de risque, modèles de valorisation, mesures de risques). L'implémentation, le design et l'optimisation des systèmes de calcul est ensuite réalisée par les équipes informatiques.
- L'identification, l'analyse et le suivi quotidien des risques et des contreparties. Les équipes d'analystes sont scindées en fonctions de la nature du risque (risque de marché, risque de contrepartie, risque de crédit) et des classes d'actifs considérées (fixed income, equities, commodities, etc.)

Au total, Risk-IM emploie environ 200 personnes à Londres. Bien qu'indépendant des activités de la branche "Corporate and Investment Banking", ce département est en relation permanente avec les équipes du front-office qui opèrent sur les marchés. Les activités sont d'ailleurs physiquement regroupées.

1.2 L'équipe en charge du risque de contrepartie: Risk-IM/CT

L'équipe "Risk-IM/CT" est en charge du suivi du risque de contrepartie liés aux États, institutions financières, et entreprises avec lesquelles la banque échange des produits financiers dérivés. Seuls les hedge-funds sont hors de son périmètre. CT est divisé en deux sous équipes:

- CT/ETAP (Exotic Trading and Portfolios) est en charge de la modélisation du risque de contrepartie (en particulier des profils d'exposition) pour tous les produits exotiques, i.e. qui ne peuvent pas être modélisés directement dans le système standard de risque (calculateur développé en interne par Risk-IM, appelé Valrisk). L'équipe est aussi en charge de la gestion du wrong-way risk (risque que l'exposition soit corrélée positivement à la probabilité de défaut de la contrepartie) et des sensibilités des expositions aux mouvements de marché. Elle conçoit, calcule et analyse des "stress tests", en particulier afin de respecter la réglementation récente. Son principal domaine d'expertise réside dans la représentation des expositions au niveau de la transaction, du portefeuille, ou de la contrepartie.

- CT/XVA calcule et assure le suivi des différents ajustements de valorisation (valuation adjustments) qui sont calculés afin de mesurer le risque de contrepartie. Les principaux ajustements sont les "Credit" and "Debit" Valuation Adjustments (CVA et DVA). L'équipe s'assure que la CVA et le capital réglementaire lié au risque de contrepartie sont correctement calculés, et est en charge de la supervision de la couverture de la CVA par les équipes de trading. Elle analyse les expositions de marché des portefeuilles de CVA et évalue l'efficacité des stratégies de couvertures.

CT est une équipe pro-active et lance de nombreux projets sur le thème du risque de contrepartie (mon stage en étant une illustration). Deux réunions d'équipe sont organisées par semaine pour faire le point. J'ai également pu participer en cours d'année à un séminaire de deux jours organisé à Paris par l'équipe sur le "futur du risque de counterpartie".

2. Organisation du stage

2.1 Progression du stage

Au jour le jour, j'ai travaillé de manière assez autonome même si les membres de l'équipe étaient toujours disponible pour répondre à mes questions. La majorité de mon temps (environ 80%) a pu être consacré aux problèmes théoriques qui m'ont été posés; je n'ai pas eu de tâches opérationnelles à réaliser quotidiennement. Le premier mois de stage c'est déroulé à Paris, où le précédent stagiaire m'a présenté le travail qu'il a réalisé sur l'optimisation du calcul des sensibilités de CVA. Les cinq mois suivants ont été l'occasion de me focaliser sur ce projet que j'ai repris et où j'ai cherché à préciser les résultats. Pendant cette période il m'a fallu me plonger dans les mécanismes qui régissent le risque de contrepartie (fondements théoriques, procédures de calculs, intérêt des mesures pour la banque). J'ai également pu me familiariser avec le travail de mon équipe, et, plus généralement, avec l'ensemble des missions de Risk-IM et de CIB.

Au cours des six mois suivants, j'ai pu utiliser ces connaissances afin de m'impliquer plus dans le travail d'équipe et les réunions. L'équipe travaille en continu sur plusieurs projets dont j'ai pu suivre l'évolution: lancement d'un nouveau système de calcul de risque utilisant le calcul parallèle, analyse de la concentration du risque de contrepartie, modélisation de transactions exotiques, etc.

2.2 Sujets de recherche

Mon stage a été l'occasion d'aborder quatre sujets de recherche distincts, qui ont chacun donné lieu à la rédaction d'un rapport et à une ou plusieurs présentations.

L'optimisation du calcul des sensibilités de CVA (présentée dans le rapport de stage). Mon principal résultat est qu'il est difficile d'optimiser ces calculs. En particulier il est difficile de se passer de la méthode des différences finies malgré son coût computationel. Les méthodes alternatives reposent trop souvent sur des propriétés statistiques spécifiques à un modèle, qui ne tiennent pas dans un cadre général. D'un point de vue pédagogique ce travail a été très enrichissant; j'ai ainsi travaillé sur des aspects théoriques complexes (calcul de Malliavin, méthodes d'estimation de dérivées dans un cadre aléatoire), sur plusieurs modèles de simulation et de valorisation (modèles de Black-Scholes, de Heath-Jarrow-Morton à 3 facteurs), et sur l'implémentation d'un modèle simple en Python. En particulier la phase d'implémentation c'est révélé très utile et m'a permis de bien comprendre les difficultés techniques qui surviennent dans les problèmes

de statistique et de simulation (problèmes de calibration, de convergence des estimateurs et des procédures de Monte-Carlo, d'estimation de l'erreur à l'aide du bootstrap).

Une analyse des sensibilités de la CVA au spread de crédit, sous l'hypothèse d'indépendance entre la probabilité de défaut et le montant de l'exposition (absence de risque "wrong-way"). L'enjeu de ce travail était de comprendre la dynamique de la CVA lorsque les spreads de crédit varient, avec un intérêt particulier pour le gamma (i.e. la variation de second ordre). La principale conclusion de ce travail est la suivante: lorsque le spread associé au CDS d'une contrepartie augmente, la fonction de répartition associée au temps de défaut (modèle standard de Poisson) de la contrepartie est modifiée de manière à ce que le défaut soit plus probable dans un temps proche et moins probable dans un temps long. Comme la CVA intègre l'exposition par rapport à cette probabilité, l'ampleur et le signe de sa variation dépendent de la répartition de l'exposition dans le temps. Ce travail m'a en particulier permis d'utiliser les systèmes de valorisation du front-office afin d'évaluer la rentabilité d'une stratégie de couverture de la CVA par le CDS de la contrepartie.

Le design d'une nouvelle mesure de risque tenant compte de la variabilité de l'exposition. La compensation ("clearing") obligatoire de produits dérivés auparavant échangé sans intermédiaire ("Over-the-Counter") transforme le risque de contrepartie. La collateralisation de ces transactions au travers des chambres de compensation mène en théorie vers une réduction importante des montants d'exposition. Les chambres centralisent le risque de contrepartie et le gèrent de manière autonome. Pour chaque nouvelle transaction compensée, les parties transfèrent le montant initial de l'exposition à la chambre de compensation. Ce montant est ensuite ajusté régulièrement par des appels de marges, afin de prendre en compte les variations des paramètres de marché. Le risque résiduel auquel fait face la banque comprend l'exposition à la chambre de compensation (dont les probabilités de défaut sont théoriquement faibles), et un risque nouveau lié au financement des appels de marges ("funding liquidity risk"). En effet, ces appels de marges sont d'ampleur aléatoire et la banque se doit de les honorer. Il m'a donc fallu réfléchir à une mesure de la *variabilité* de l'exposition sur un intervalle de temps donné. La principale proposition de ce travail est le calcul d'un percentile sur les appels de marges (qui sont simulés régulièrement par le système de risque interne pour l'ensemble du portefeuille de la banque). Cette mesure présente l'avantage d'être simple à mettre en place et de donner une information rapide sur la variabilité du profil d'exposition au cours de la vie du produit. Ce travail m'a permis de travailler sur un sujet qui intéresse directement les équipes opérationnelles et qui ont pu m'aider plus que sur les autres projets.

Une étude d'impact d'un "Oui" au référendum britannique sur l'indépendance de l'Ecosse. Ce travail m'a permis d'entrer en contact avec les équipes de macroéconomistes du département CIB, et d'analyser leurs scénarios. J'ai ensuite synthétisé leurs résultats et déterminé les implications possibles en terme de risque de contrepartie. Ce travail a été présenté à mon équipe et inclus dans ses publications, diffusées à l'ensemble du département des risques.

2.3 Methodologie de travail

Mon travail de recherche a été structuré tout au long du stage par les étapes suivantes:

- Définition du problème de recherche avec mon encadrant (Guillaume Hermet) et le reste de l'équipe.
- Revue de littérature, réflexion et expérimentation. Pour chaque problème la littérature existante en finance computationnelle, statistique, et Monte-Carlo m'a été utile. Après une

étape de réflexion "à la main", l'implémentation informatique m'a permis de sélectionner les méthodes intéressantes où non.

- Synthèse et présentation des résultats. Une fois le projet finalisé, les résultats de chaque projet ont été présentés à mon équipe mais aussi à un public plus large: analystes, quants, credits officers, et informaticiens; et ce dans différentes localisation: London, Paris, New-York, Hong-Kong et Tokyo.

Bilan sur le stage

De mon point de vue le stage a été très enrichissant. L'équipe s'est montrée accueillante et j'ai rapidement été intégré comme un membre à part entière. Mon tuteur m'a laissé une marge de manœuvre dans l'organisation de mon travail et j'ai pu me concentrer sur les aspects théoriques qui m'intéressent le plus. Il m'a également fait confiance, et nous avons présenté nos résultats ensemble à des équipes expérimentées tout au long de l'année.

J'ai pu aborder un énorme éventail de techniques et j'ai du faire face à des défis tout au long de l'année. J'ai ainsi largement amélioré mes connaissances en calcul stochastique, théorie de la mesure et des probabilités, statistique computationnelle et méthodes de Monte-Carlo, programmation en langage orienté objet, modèles de valorisation et de simulation appliqués en finance, et en calcul différentiel.

Par ailleurs, j'ai découvert l'industrie du risque et plus largement le fonctionnement d'une banque d'investissement. Vu le niveau de complexité des opérations et de l'activité dans ce milieu, l'année de stage me parait être un bon format: six mois auraient été trop courts.

En sortant de deuxième année, mes compétences en informatique et en statistique computationnelle étaient un peu légères vu la complexité du problème posé. Cependant, après un an avec "les mains dans le cambouis", j'ai acquis un recul qui me sera extrêmement utile en troisième année (voie Data-Science / Statistique et Apprentissage).