CVA Sensitivities, Hedging, and Risk *

S. Crépey[†], B. Li[‡], H. Nguyen[§], B. Saadeddine[¶]

August 16, 2024

Abstract

We present a framework for computing CVA sensitivities, hedging the CVA, and assessing CVA risk, using probabilistic machine learning meant as refined regression tools on simulated data, validatable by low-cost companion Monte Carlo procedures. We identify the sensitivities representing the best practical trade-offs in downstream tasks including CVA hedging and risk assessment.

Introduction

This work illustrates the potential of probabilistic machine learning for pricing and Greeking applications, in the challenging context of CVA computations. By probabilistic machine learning we mean machine learning as refined regression tools on simulated data. Probabilistic machine learning for CVA pricing was introduced in Abbas-Turki, Crépey, and Saadeddine (2023). Here we extend our approach to encompass CVA sensitivities and risk. The fact that probabilistic machine learning is performed on simulated data, which can be augmented at will, does not mean that there are no related data issues. As always with machine learning, the quality of the data is the first driver of the success of the approach. The variance of the training loss may be high and jeopardize the potential of a learning approach. This was first encountered in the CVA granular defaults pricing setup of Abbas-Turki et al. (2023) due to the scarcity of the default events compared with the diffusive scale of the risk factors in the model. Switching from prices to sensitivities in this paper is another case of increased variance. But with probabilistic machine learning we can

Acknowledgement: We are grateful to Moez Mrad, head of XVA, counterparty risk, collateral, and credit derivatives quantitative research at Crédit Agricole CIB, and to an anonymous referee, for inspiring exchanges.

*A long preprint version of this work is available on arXiv. The python code of the paper is available on https://github.com/botaoli/CVA-sensitivities-hedging-risk.

[†]Email: `stephane.crepey@lpsm.paris`. LPSM/Université Paris Cité, France.

[‡]Email: `botaoli@lpsm.paris`. LPSM/Université Paris Cité. The research of B. Li is funded by the Chair *Capital Markets Tomorrow: Modeling and Computational Issues* under the aegis of the Institut Europlace de Finance, a joint initiative of Laboratoire de Probabilités, Statistique et Modélisation (LPSM) / Université Paris Cité and Crédit Agricole CIB, with the support of Labex FCD (ANR-11-LABX-0019-01).

[§]Email: `hdnguyen@lpsm.paris`. LPSM/Université Paris Cité. The research of H.D. Nguyen is funded by a CIFRE grant from Natixis.

[¶]Email: `bouazza.saadeddine2@ca-cib.com`. Quantitative research GMD, Credit Agricole CIB, Paris.

also develop suitable variance reduction tools, namely oversimulation of defaults in Abbas-Turki et al. (2023) and common random numbers in this work. Another distinguishing feature of probabilistic machine learning, key for regulated banking applications, is the possibility to assess the quality of a predictor by means of low-cost companion Monte Carlo procedures.

All equations are written using the risk-free asset as a numéraire and stated under the fininsurance probability measure advocated for XVA computations in Albanese, Crépey, Hoskinson, and Saadeddine (2021, Remark 2.3), with related expectation operator denoted below by $\mathbb{E}$. Unless explicitly stated, we include a ridge regularization term to stabilize trainings and regressions.

1 Fast Bump Sensitivities

Consider a time-0 option price $\Pi_0(\rho) = \mathbb{E}\xi(\rho)$, where the payoff $\xi(\rho) \equiv \xi(\rho;\omega)$ depends on constant model parameters ρ and (implicitly in the shorthand notation $\xi(\rho)$) on the randomness ω of the stochastic drivers of the model risk factors with respect to which the expectation is taken. The model parameters ρ encompass the initial values of the risk factors of the pricing model, as well as all the exogenous (constant, in principle) model parameters, e.g. the value of the volatility in a Black-Scholes model. For each constant ρ, the price $\Pi_0(\rho)$ can be estimated by Monte Carlo. Our problem in this part is the estimation of the corresponding sensitivities $\partial_\rho \Pi_0(\rho_0)$, at a baseline (in practice, calibrated) value $\rho = \rho_0$ of the model parameters. Such sensitivities lie at the core of any related hedging scheme for the option. They are also key in many regulatory capital formulas.

Monte Carlo estimation of sensitivities in finance comes along three main streams: (i) differentiation of the density of the underlying process via integration by parts or more general Malliavin calculus techniques, assuming some regularity of this process; (ii) cash flows differentiation, assuming their differentiability; (iii) Monte Carlo finite differences, biased but generic, which are the Monte Carlo version of the industry standard bump sensitivities. But (i) suffers from intrinsic variance issues. In contemporary technology, (ii) appeals to adjoint algorithmic differentiation (AAD). AAD can quickly represent important implementation and memory costs on complex pricing problems at the portfolio level such as CVA computations: see Capriotti, Jiang, and Macrina (2017). Such an AAD Greeking approach becomes nearly unfeasible in the case of pricing problems embedding numerical optimization subroutines, e.g. the training of the conditional risk measures embedded in the refined CVA and higher-order XVA metrics of Albanese et al. (2021).

1.1 Common Random Numbers

Under the approach (iii), bump sensitivities are computed by relaunching the Monte Carlo pricing engine with common random numbers ω for values bumped by $\pm 1\%$ (typically and in relative terms) of each risk factor and/or model parameter of interest, then taking the accordingly normalized difference between the corresponding $\Pi_0(\rho)$ and $\Pi_0(\bar{\rho})$, where $\bar{\cdot}$ means symmetrization with respect to ρ_0, so

$$\frac{\rho + \bar{\rho}}{2} = \rho_0, \text{ i.e. } \bar{\rho} = 2\rho_0 - \rho. \tag{1}$$

This approach requires two Monte Carlo simulation runs (with common drivers ω) of m paths per sensitivity, making it a robust but heavy procedure dubbed **benchmark bump sensitivities** hereafter. We also compute **smart bump sensitivities**, similar but only using m/p paths each, where $p = |\rho|$. Hence the time of computing all the smart bump sensitivities is of the same order of magnitude as the time of pricing $\Pi_0(\rho_0)$ by Monte Carlo with m paths. More precisely, each smart bump sensitivity uses m/p paths of a Monte Carlo simulation run with m paths as a whole. This is significantly more efficient than doing p Monte Carlo runs of size m/p each, especially in the GPU simulation environment of our CVA computations below.

As another way to accelerate the bump sensitivities with common random numbers as in (iii), a useful trick is to introduce $\varsigma(\rho;\omega) := \xi(\rho;\omega) - \xi(\bar{\rho};\omega)$ (cf. (1)). We can then learn the sensitivity (in the sense here of finite differences) function $\rho \mapsto \Sigma_0(\rho) := \mathbb{E}\varsigma(\rho)$, which satisfies by linearity and chain rule (as $\bar{\rho} = 2\rho_0 - \rho$) $\Sigma_0'(\rho) = \Pi_0'(\rho) + \Pi_0'(\bar{\rho})$, in particular $\partial_\rho \Sigma_0(\rho_0) = 2\partial_\rho \Pi_0(\rho_0)$. For learning the function $\Sigma_0(\cdot)$ locally around ρ_0, with ϱ randomizing ρ as above, we rely on the representation $\Sigma_0(\rho) = \mathbb{E}\big(\varsigma(\varrho) \,\big|\, \varrho = \rho\big)$, i.e.

$$\Sigma_0(\cdot) = \operatorname*{arg\,min}_{\Phi \in \mathcal{B}} \mathbb{E}\Big[\big(\varsigma(\varrho) - \Phi(\varrho)\big)^2\Big], \tag{2}$$

where $\mathcal{B}$ denotes the Borel measurable functions of ρ. Then we replace, in the optimization problem (2), $\mathcal{B}$ by a linear hypothesis space $\Sigma_0^\theta(\rho) = \theta^\top(\rho - \rho_0)$ (noting that $\Sigma_0(\rho_0) = 0$) and $\mathbb{E}$ by a simulated sample mean $\widehat{\mathbb{E}}$, with each draw of ϱ followed by one draw of ω that is implicit in $\varsigma(\varrho)$. This results in a linear least-squares problem for the weights θ, solved by regression. The estimated weights $\theta/2$ are our **linear bump sensitivities** estimate for $\frac{1}{2}\partial_\rho \Sigma_0(\rho_0) = \partial_\rho \Pi_0(\rho_0)$. For simple parametric distributions of ϱ, the covariance matrix that appears in the regression is known and invertible in closed form, which reduces this regression (implemented without ridge regularization in this analytical case) to standard Monte Carlo endowed with the associated confidence intervals, for a computation time comparable to the one of smart bump sensitivities.

The sensitivities $\partial_\rho \Pi_0(\rho_0)$ are sensitivities to model parameters. Practical hedging schemes require sensitivities to calibrated prices of hedging instruments. Hence, for hedging purposes, our sensitivities must be mapped to hedging ratios in market instruments. This is done via the implicit function theorem, the way explained in Antonov et al. (2018).

2 Credit Valuation Adjustment and Its Bump Sensitivities

Since we are also interested in the risk of CVA fluctuations, we now consider the targeted price Π (CVA from now on) as a process. We denote by MtM^c, the counterparty-risk-free valuation of the portfolio of the bank with its client c; τ_c, the client c's default time, with intensity process γ^c; X, with $X_0 \equiv 0$ (componentwise), the vector of the default indicator processes of the clients of the bank; Y, a diffusive vector process of model risk factors such that each MtM_t^c and γ_t^c is a

measurable function of (t, Y_t), for $t \geq 0$ (in the case of credit derivatives with the client c, MtM_s^c would also depend on X_t, which can be accommodated at no harm in our setup). The exogenous model parameters are denoted by ϵ. Let the baseline $\rho_0 = (y_0, \epsilon_0)$ denote a calibrated value of (y, ϵ), where y is used for referring to the initial condition of Y, whenever assumed constant. Let ι denote an initial condition for Y randomized around its baseline y_0, ε be likewise a randomization of ϵ around its baseline ϵ_0, and $\varrho_t = (Y_t, \varepsilon), t \geq 0$. Starting from the (random) initial condition $(0, \iota)$, the model (X, Y) is supposed to evolve according to some Markovian dynamics (e.g. the one of our CVA Lab in the next part) parameterized by ε. This setup encompasses in a common formalism:

- the **baseline mode** of Abbas-Turki et al. (2023, Section 4) where $\varrho_0 \equiv \rho_0$;
- the **risk mode** where $Y_0 \equiv y_0$;
- the **sensis mode**, or general ϱ_0 case (only used for $t = 0$).

The CVA engine in the baseline mode $\varrho_0 \equiv \rho_0$ was introduced in Abbas-Turki et al. (2023). The risk and sensis mode also incorporating a randomization ε of the exogenous model parameters ϵ, and of Y_0 in the sensis mode, are novelties of this work.

We restrict ourselves to an uncollateralized CVA for notational simplicity. Given n pricing time steps of length h such that $nh = T$, the final maturity of the derivative portfolio of the bank, let, at each $t = ih$,

$$\begin{aligned} \mathrm{LGD}_t &= \sum_c \sum_{j=0}^{i-1} (\mathrm{MtM}_{jh}^c)^+ \mathbb{1}_{jh < \tau_c \leq (j+1)h}, \\ \xi_{t,T} &= \sum_c \sum_{j=i}^{n-1} (\mathrm{MtM}_{jh}^c)^+ (e^{-h\sum_{\imath=i}^{j-1} \gamma_\imath^c} - e^{-h\sum_{\imath=i}^{j} \gamma_\imath^c}) \mathbb{1}_{\{\tau_c > ih\}}. \end{aligned} \tag{3}$$

Our computations rely on the following default-based and intensity-based formulations of the (time-discretized) CVA of a bank with clients c, at the pricing time $t = ih$ (cf. Abbas-Turki et al. (2023, Eqns. (25)-(27))):

$$\begin{aligned} \mathrm{CVA}_t(x, \rho) &= \mathbb{E}\Big[\underbrace{\sum_c \sum_{j=i}^{n-1} (\mathrm{MtM}_{jh}^c)^+ \mathbb{1}_{jh < \tau_c \leq (j+1)h}}_{\mathrm{LGD}_T - \mathrm{LGD}_t} \Big| X_{ih} = x, \varrho_{ih} = \rho \Big] \\ &= \mathbb{E}\Big[\underbrace{\sum_c \sum_{j=i}^{n-1} (\mathrm{MtM}_{jh}^c)^+ (e^{-h\sum_{\imath=i}^{j-1} \gamma_\imath^c} - e^{-h\sum_{\imath=i}^{j} \gamma_\imath^c}) \mathbb{1}_{\{\tau_c > ih\}}}_{\xi_{t,T}} \Big| X_{ih} = x, \varrho_{ih} = \rho \Big], \end{aligned} \tag{4}$$

where each coordinate of x is 0 or 1. From a numerical viewpoint, the second line of (4) entails less variance than the first one (see Figure 5 in Abbas-Turki et al. (2023)). Hence we rely for our CVA computations on this second line. At the initial time 0, as $X_0 \equiv 0$, we can restrict attention to the origin $x = 0$ and skip the argument x as well as the conditioning by $X_{ih} = x$ in (4). In the sensis mode, one is in the setup (2) for $\xi = \xi_{0,T}$, which implicitly depends on $\varrho_0 = (\iota, \varepsilon)$, i.e.

$\mathrm{CVA}_0(\varrho_0) = \mathbb{E}(\xi(\varrho_0) \,|\, \varrho_0)$. CVA fast bump sensitivities can thus be computed the ways exposed in the first part of the paper, with CVA and $(y = Y_0, \epsilon)$ here in the role of Π and ρ there. In the baseline mode where $\varrho_0 \equiv \rho_0$, $\mathrm{CVA}_0(\varrho_0)$ is constant, equal to the corresponding

$$\mathrm{CVA}_0(\rho_0) = \mathbb{E}\,\mathrm{LGD}_{nh} = \mathbb{E}\xi_{0,T}, \tag{5}$$

which is computed by Monte Carlo based on the second line of (4) for $t = 0$ there, as a sample mean of $\xi_{0,T}$, along with the corresponding 95% confidence interval.

2.1 CVA Lab

In our numerics, we have 10 economies. For each of them we have a short-term interest rate driven by a Vasicek diffusion and, except for the reference economy, a Black-Scholes exchange rate with respect to the currency of the reference economy. The reference bank has 8 counterparties with corresponding default intensity processes driven by CIR diffusions. We thus have 8 default indicator processes X of the counterparties and $10 + 9 + 8 = 27$ diffusive risk factors Y. This results in a Markovian model (X, Y) of dimension 35, entailing $p = 90$ parameters corresponding to the 27 initial conditions of the Y processes plus their 63 exogenous parameters.

A "reasonable" but arbitrary baseline ρ_0 plays the role of calibrated model parameters in our numerics. The portfolio consists of 500 interest rate swaps with random characteristics (maturity $\leq T = 10$ years, notional, currency and counterparty) and strikes such that the swaps are worth 0 in the baseline model (i.e. for $\varrho_t \equiv \rho_0$) at time 0. The swaps have analytic counterparty-risk-free valuation in our pricing model. Their price processes are converted into the reference currency and aggregated into the corresponding clients MtM^c processes.

We simulate by an Euler scheme $m = 2^{17} \approx 1.3 \times 10^5$ paths of the pricing model (X, Y), with $n = 100$ MtM pricing time steps of length $h = 0.1$ and 25 Euler simulation sub-steps per pricing time step. A Monte Carlo computation of (5) in the baseline mode yields $\mathrm{CVA}_0(\rho_0) \in 5,027 \pm 18$ with 95% probability (computed in about 30s). A randomization of Y_0 and ϵ in the sensis mode is used for deriving CVA linear bump sensitivities the way explained after (2). In Figure 1, the increasing curves in the left panel highlight the almost linear growth of the speedup of the linear and smart bump sensitivities with respect to the benchmark ones when the number p of pricing model parameters increases, but with also increasing errors displayed in the middle panel. The smart bump sensitivities have smaller errors than the linear bump sensitivities for all tested p. The superiority of the smart bump sensitivities over the linear ones is also observed in the right panel of Figure 1, where for all tested p the complexity of linear bump sensitivities is higher than the one of smart bump sensitivities, itself very close (as expected) to the one of the benchmark bump sensitivities.

We specify $q = 256$ market instruments corresponding to zero-coupons, credit defaults swaps and FX forwards of various maturities in the different economies and on the different counterparties of the bank in the model. The model sensitivities are then converted into market sensitivities by the implicit function theorem.

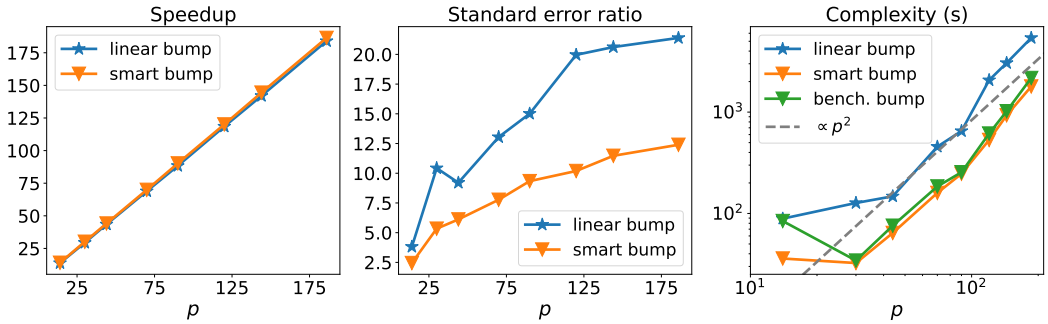

Figure 1: Comparison of performance of linear and smart bump sensitivities as a function of the number of model parameters p fixing $m = 2^{17}$, using benchmark bump sensitivities as references. The speedups displayed in the left panels are obtained by dividing the execution times for benchmark sensitivities by the execution times for linear or smart bump sensitivities. For each value of p, the error ratio in the middle panel denotes the median of the p ratios between the Monte Carlo standard errors of the linear (resp. smart) bump sensitivity and of the benchmark sensitivity. The complexities in the right panels are defined as projected times of reaching a relative standard error of 1%, building on a $1/\sqrt{|\text{sample size}|}$ scaling of the standard errors justified by the central limit theorem, with $|\text{sample size}| = m$ for linear and benchmark bumps and m/p for smart bumps.

2.2 Learning the Future CVA

Equivalently to the second line in (4),

$$\mathrm{CVA}_t(\cdot) = \operatorname*{arg\,min}_{\Phi\in\mathcal{B}} \mathbb{E}\Big[\big(\xi_{t,T} - \Phi(X_t, \varrho_t)\big)^2\Big], \tag{6}$$

where $\mathcal{B}$ is the set of the Borel measurable functions of (x, ρ). We denote by $\mathrm{CVA}_t^\theta(X_t, \varrho_t)$ the conditional CVA at time $t = ih > 0$ learned by a neural network with parameters θ on the basis of simulated pairs (X_t, ϱ_t) and cash flows $\xi_{t,T}$. The conditional CVA pricing function $\mathrm{CVA}_t^\theta(x, \rho)$ is obtained by replacing $\mathbb{E}$ by a simulated sample mean $\widehat{\mathbb{E}}$ and $\mathcal{B}$ by a neural net (or linear as a special case) search space in the optimization problem (6). The latter is then addressed numerically by Adam gradient descent on the basis of simulated pairs (X_t, ϱ_t) as features and $\xi_{t,T}$ as labels.

A key asset of probabilistic machine learning procedures for any conditional expectation such as $\Pi_0(\varrho)$ in the first part of the paper (or $\mathrm{CVA}_t(X_t, \varrho_t)$ above) is the availability of the companion "twin Monte Carlo validation procedure" of Abbas-Turki et al. (2023, Section 2.4), allowing one to assess the accuracy of a predictor. Let $\xi^{(1)}(\varrho)$ and $\xi^{(2)}(\varrho)$ denote two copies of $\xi(\varrho)$ independent given ϱ, i.e. such that $\mathbb{E}[f(\xi^{(1)}(\varrho))g(\xi^{(2)}(\varrho))|\varrho] = \mathbb{E}[f(\xi^{(1)}(\varrho))|\varrho]\mathbb{E}[g(\xi^{(2)}(\varrho))|\varrho]$ holds for any Borel bounded functions f and g. The twin Monte Carlo validation procedure for a predictor $\Phi(\varrho)$ of $\Pi_0(\varrho) = \mathbb{E}(\xi(\varrho)|\varrho)$ consists in computing Monte Carlo estimates *twin-stat* for

$$\mathbb{E}\Big[\Phi(\varrho)^2 - (\xi^{(1)}(\varrho) + \xi^{(2)}(\varrho))\Phi(\varrho) + \xi^{(1)}(\varrho)\xi^{(2)}(\varrho)\Big] = \mathbb{E}\Big[\Big(\Phi(\varrho) - \mathbb{E}(\xi(\varrho)|\varrho)\Big)^2\Big] \tag{7}$$

(as it follows from the tower rule by conditional independence) and *twin-var* for $\mathbb{V}\mathrm{ar}\Big[\Phi(\varrho)^2 - (\xi^{(1)}(\varrho) + \xi^{(2)}(\varrho))\Phi(\varrho) + \xi^{(1)}(\varrho)\xi^{(2)}(\varrho)\Big]$. The ensuing 95% confidence level upper bound on $\sqrt{\mathbb{E}\Big[\Big(\Phi(\varrho) - \mathbb{E}(\xi(\varrho)|\varrho)\Big)^2\Big]}$ (the RMSE of the predictor $\Phi(\varrho)$) is then given by

$$\textit{twin-up} = \sqrt{\textit{twin-stat} + \frac{2}{\sqrt{m}}\sqrt{\textit{twin-var}}}. \tag{8}$$

3 CVA Risk and Hedging

An economic (or "internal") view gained from simulating the movements of model or/and market risk factors and obtaining risk measures of CVA fluctuations is an important dimension of the CVA capital regulatory requirements of a bank, in the context of its supervisory review and evaluation process (SREP): quoting https://www.bankingsupervision.europa.eu/legalframework/publiccons/html/icaap_ilaap_faq.en.html (last accessed June 6 2024), "the risks the institution has identified and quantified will play an enhanced role in, for example, the determination of additional own funds requirements on a risk-by-risk basis."

3.1 Run-off CVA Risk

We first consider

$$\delta\mathrm{CVA}_t^\theta + \mathrm{LGD}_t, \text{ where } \delta\mathrm{CVA}_t^\theta = \mathrm{CVA}_t^\theta(X_t, \varrho_t) - \mathrm{CVA}_0(\rho_0) \tag{9}$$

(cf. (3)-(4)), assessed in the risk mode, where $\varrho_t = (Y_t(y_0, \varepsilon), \varepsilon)$ and ε follows $\mathcal{N}\Big(\epsilon_0, \mathrm{diag}\left(t(1\%)^2\epsilon_0 \odot \epsilon_0\right)\Big)$[1]. The random variable $(\delta\mathrm{CVA}_t^\theta + \mathrm{LGD}_t)$ reflects a dynamic but also run-off view on CVA and counterparty default risk altogether, as opposed to the stationary run-on CVA risk view below. We assess CVA risk, counterparty default risk, and both risks combined, on the basis of value-at-risks (VaR) and expected shortfalls (ES) of $\delta\mathrm{CVA}_t^\theta$ and LGD_t in the risk mode, reported for $t = 0.01, 0.1, 1$ yr and quantile levels 99% for VaR and 97.5% for ES in Table 1. The results of Table 1 emphasize that CVA and counterparty default risks assessed on a run-off basis are primarily driven by client defaults, especially at higher quantile levels.

	$t = 0.01$			$t = 0.1$			$t = 1$		
	$\delta\mathrm{CVA}_t^\theta$	LGD_t	$\delta\mathrm{CVA}_t^\theta$ +LGD_t	$\delta\mathrm{CVA}_t^\theta$	LGD_t	$\delta\mathrm{CVA}_t^\theta$ +LGD_t	$\delta\mathrm{CVA}_t^\theta$	LGD_t	$\delta\mathrm{CVA}_t^\theta$ +LGD_t
Expectation	-9	0.28	-8	56	7	63	75	502	578
VaR 99%	388	0	389	1,347	0	1,389	4,796	11,997	11,757
ES 97.5%	395	0.28	397	1,361	7	1,445	4,953	12,383	12,297

Table 1: Risk measures of $\delta\mathrm{CVA}_t^\theta$, LGD_t and their sum ($\mathrm{CVA}_0(\rho_0) = 5,027$).

[1] $\odot$ is the Hadamard (i.e. componentwise) product between vectors and diag(vec) is a diagonal matrix with diagonal "vec".

3.2 Run-on CVA Risk

With $\varepsilon_{(t)} \sim \mathcal{N}\Big(\epsilon_0, \mathrm{diag}\left(t(1\%)^2\epsilon_0 \odot \epsilon_0\right)\Big)$, denoting by $Y_t(y_0, \varepsilon_{(t)})$ the Y process at time t starting from y_0 at 0 and for model parameters set to $\varepsilon_{(t)}$, by $\varrho_{(t)} = \big(Y_t(y_0, \varepsilon_{(t)}), \varepsilon_{(t)}\big)$, by $Z_0(\varrho_{(t)})$ the time-0 price of market instruments corresponding to the model parameters $\varrho_{(t)}$, and by z_0 the baseline price of the market instruments at time 0, let

$$\begin{aligned} &\delta\varrho_{(t)} = (Y_t(y_0, \varepsilon_{(t)}) - y_0, \varepsilon_{(t)} - \epsilon_0) = \varrho_{(t)} - \rho_0,\ \delta Z_{(t)} = Z_0(\varrho_{(t)}) - z_0, \\ &\delta\mathrm{CVA}_{(t)} = \mathrm{CVA}_0(\varrho_{(t)}) - \mathrm{CVA}_0(\rho_0). \end{aligned} \tag{10}$$

The fact that we consider the time-0 $\mathrm{CVA}_0(\varrho_{(t)})$ (see after (4)) and likewise $Z_0(\varrho_{(t)})$ here is in line with an assessment of risk on a run-on portfolio and customers basis and with a siloing of CVA vs. counterparty default risk, which have both become standard in regulation and market practice. Various predictors of $\delta\mathrm{CVA}_{(t)}$ can be learned directly from the simulated model parameters $\varrho_{(t)}$ and cash flows $\xi_{0,T}(\varrho_{(t)}) - \xi_{0,T}(\rho_0)$ (as opposed to and better than learning $\delta\mathrm{CVA}_{(t)}$ via $\mathrm{CVA}_0(\varrho_{(t)})$, which would involve more variance): nested Monte Carlo estimator, neural net regressor $\delta\mathrm{CVA}^{\theta}_{(t)}$, linear(-diagonal quadratic) regressors against $\delta\varrho_{(t)}$ or $\delta Z_{(t)}$ referred to as LS (for "least squares") below. The neural network used for training $\delta\mathrm{CVA}^{\theta}_{(t)}$ based on simulated data $(\varrho_{(t)} - \rho_0, \xi_{0,T}(\varrho_{(t)}; \omega) - \xi_{0,T}(\rho_0; \omega))$ has one hidden layer with two hundred hidden units and softplus activation functions.

Table 2 displays some twin upper bounds (8) and risk measures of $\delta\mathrm{CVA}_{(t)}$ computed with these different approximations, as well as with linear(-diagonal quadratic) Taylor expansions in $\delta\varrho_{(t)}$ or $\delta Z_{(t)}$ with coefficients estimated as benchmark or smart bump sensitivities. In terms of the twin upper bounds, the nested CVA has the best accuracy, but (for a given risk horizon t) it takes about 2 hours (with 1024 inner paths), versus about one minute of simulation time for generating the labels, plus 30 seconds for training by neural networks and 2-3 seconds for LS regression. The neural network excels at large t, where the non-linearity becomes significant, while being outperformed by the linear methods at small t, where $\delta\mathrm{CVA}_{(t)}$ is approximately linear. With diagonal gamma (Γ) elements taken into account, the performance of the LS regressor improves at large t. The linear quadratic Taylor expansions show relatively good twin upper bounds for small t, but worsen for large t.

Regarding now the risk measures, the nested $\delta\mathrm{CVA}_{(t)}$ and the neural network $\delta\mathrm{CVA}^{\theta}_{(t)}$ provide more conservative VaR and ES estimates than any linear(-quadratic) proxy in all cases. Surprisingly, even though $\mathrm{CVA}_0(\varrho_{(t)})$ in $\delta\mathrm{CVA}_{(t)}$ is a function of $\varrho_{(t)}$, for $t = 1$ the linear(-quadratic) proxies in $\delta Z_{(t)}$ outperform those in $\varrho_{(t)}$, in terms both of twin error and of consistency of the ensuing risk measures with those provided by the nonparametric (neural net and nested) references. Also note that, when compared with the nonparametric approaches again, the smart bump sensitivities proxy in $\delta Z_{(t)}$ yields results almost as good as the much slower benchmark bump sensitivities proxy in $\delta Z_{(t)}$.

t	risk measure	nonparametric		linear(-quadratic) in $\delta\varrho_{(t)}$			linear(-quadratic) in $\delta Z_{(t)}$		
		nested $\delta\mathrm{CVA}_{(t)}$	$\delta\mathrm{CVA}^{\theta}_{(t)}$	bench. bump sensis w/ Γ	smart bump sensis	LS sensis w/ Γ	bench. bump sensis	smart bump sensis	LS sensis w/ Γ
0.01	twin-up	**11**	29	**13**	**15**	18	23	23	18
	VaR 99%	**510**	**431**	429	418	415	**435**	431	424
	ES 97.5%	**514**	**434**	432	421	417	**437**	433	427
0.1	twin-up	**59**	103	110	111	109	**101**	104	**94**
	VaR 99%	**1,686**	**1,618**	**1,562**	1,463	1,539	1,554	1,542	1,556
	ES 97.5%	**1,693**	**1,622**	**1,570**	1,467	1,544	1,559	1,545	1,561
1	twin-up	**325**	**693**	1,244	1,307	1,113	932	943	**743**
	VaR 99%	**10,333**	**9,887**	8,991	6,646	8,812	7,914	7,846	**9,493**
	ES 97.5%	**10,654**	**10,090**	9,309	6,715	9,032	8,075	7,988	**9,792**

Table 2: Risk measures of $\delta\mathrm{CVA}_{(t)}$ computed by Monte Carlo using $\delta\mathrm{CVA}_{(t)}$ simulated by various predictors. The three lowest (i.e. best) twin upper bounds (8) and the three highest (i.e. most conservative) risk estimates on each row are emphasized in bold.

3.3 Run-on CVA Hedging

By loss $L^{\theta}_{(t)}$, we mean the following hedged loss(-and-profit) of the CVA desk over the risk horizon t (cf. (10)):

$$L^{\theta}_{(t)} = \delta\mathrm{CVA}^{\theta}_{(t)} - (\delta Z_{(t)})^{\top}\Delta - c, \tag{11}$$

where the hedging ratio $\Delta \in \mathbb{R}^q$ is treated as a free parameter, while the real constant c is deduced from Δ through the constraint that $\mathbb{E}L^{\theta}_{(t)} = 0$ (or $\widehat{\mathbb{E}}L^{\theta}_{(t)} = 0$ in the numerics). The constant c, which is equal to 0 (modulo the numerical noise) in the baseline mode where CVA + LGD and Z + CF are both martingales, can be interpreted in terms of a hedging valuation adjustment (HVA) in the spirit of Albanese, Benezet, and Crépey (2023), i.e. a provision for model risk. Albanese et al. (2023) develop how, such a provision having been updated in a first stage, the loss $L^{\theta}_{(t)}$, thus centered via the "HVA trend" c (i.e. $c = -\mathbb{E}\delta\mathrm{HVA}_{(t)} = \mathrm{HVA}_0 - \mathbb{E}\mathrm{HVA}_{(t)}$, where $\mathrm{HVA}_{(t)}$ is the HVA analog of $\mathrm{CVA}_{(t)}$), deserves an economic capital, which we quantify below as an expected shortfall of $L^{\theta}_{(t)}$. By **economic capital (EC) and PnL explain (PLE) run-on sensitivities**, we mean (cf. Rockafellar and Uryasev (2000))

$$\begin{aligned}\Delta^{ec} &= \operatorname*{arg\,min}_{\Delta\in\mathbb{R}^q} \mathbb{ES}\left(L^{\theta}_{(t)}\right),\ (\Delta^{ec}, \mathbb{V}a\mathbb{R}(L^{\theta}_{(t)})) = \operatorname*{arg\,min}_{\Delta\in\mathbb{R}^q,k\in\mathbb{R}}\ k + (1-\alpha)^{-1}\mathbb{E}\left[\left(L^{\theta}_{(t)} - k\right)^{+}\right],\\ \Delta^{ple} &= \operatorname*{arg\,min}_{\Delta\in\mathbb{R}^q} \mathbb{E}[(L^{\theta}_{(t)})^2],\end{aligned} \tag{12}$$

where $\mathbb{V}a\mathbb{R}$ and $\mathbb{ES}$ mean the 95% value-at-risk and the corresponding expected shortfall. Once $\delta\mathrm{CVA}^{\theta}_{(t)}$ learned from simulated $\varrho_{(t)}$ and $\xi_{0,T}(\varrho_{(t)}) - \xi_{0,T}(\rho_0)$ the way mentioned after (10), these sensitivities are computed as per (12), by training of Δ, k in the second part of the first line, for Δ^{ec}, and by regression of Δ in the second line, for Δ^{ple}. Simpler (but still optimized) **least squares (LS) run-on sensitivities** are obtained without prior learning of $\delta\mathrm{CVA}_{(t)}$, just by regressing linearly $\xi_{0,T}(\varrho_{(t)}) - \xi_{0,T}(\rho_0)$ against $\delta Z_{(t)}$ (but for a purely linear LS regression

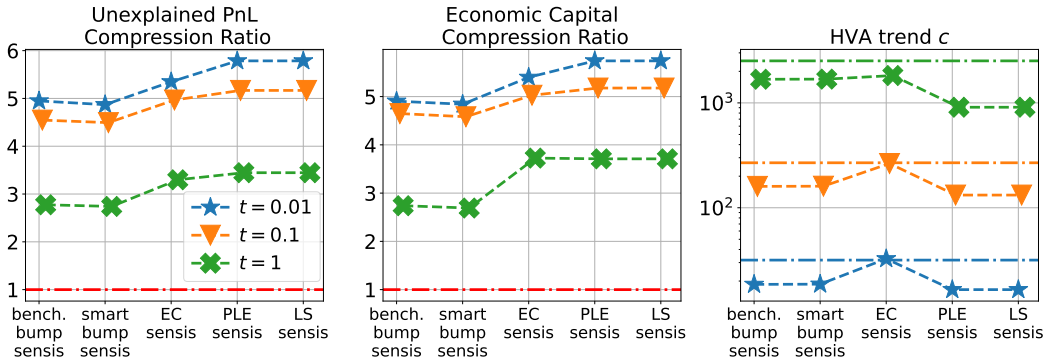

Figure 2: Compression ratios of UPL (*left*), EC (*center*), and HVA trends c (*right*), for various $\delta\text{CVA}^{\theta}_{(t)}$ hedging approaches. Horizontal dash-dot lines correspond to the unhedged case with $\Delta = 0$ in (11).

here as opposed to linear diagonal quadratic LS regression in Table 2, due to the hedging focus of this part). These LS sensitivities are thus obtained much like the linear bump sensitivities (see after (2)), but directly in the market (price) variables, without Jacobian transformations. The computation times of the LS, EC, and PLE run-on sensitivities are reported in Table 3.

By unexplained PnL UPL (resp. economic capital EC), we mean the standard error (resp. expected shortfall) of $L^{\theta}_{(t)}$. As performance metrics, we consider a backtesting, out-of-sample UPL (resp. EC with $\alpha = 95\%$) for $\Delta = 0$, divided by UPL (resp. EC with $\alpha = 95\%$) for each considered set of sensitivities: the higher the corresponding "compression ratios", the better the corresponding sensitivities. For each simulation run below, we use $m = 2^{17}$ paths to estimate the PLE, EC and LS sensitivities and we generate other $m = 2^{17}$ paths for our backtest.

Figure 2 shows the run-on CVA hedging performance of different candidate sensitivities. All the risk compression ratios decrease with the risk horizon t. All sensitivities reduce both the unexplained PnL and economic capital by at least 2.5 times for $t = 0.01$ and 4.5 times for $t = 0.1$ and 1. Since client defaults are skipped in the run-on mode, the efficiency of bump sensitivities hedges is understandable. For each risk horizon and performance metric, the optimized sensitivities always have better results than (benchmark or smart) bump sensitivities. Among those, the PLE and LS sensitivities hedges display the highest risk compression ratios and the lowest HVA trend c. Most sensitivities (except for EC sensitivities when $t = 0.01$ or 0.1) also compress the HVA trend c compared to the unhedged case.

[Smart] bump sensitivities		Optimized sensitivities					Speedup		
Model sensis	Jacobian transforms	MtM simulation	LS	$\delta\text{CVA}^{\theta}_{(t)}$ learning	PnL regression EC	PnL regression PLE	LS	EC	PLE
12min48s [8.5s]	30s	27s	1s	6s	31s	1s	28.5 [1.4]	12.5 [0.6]	23.5 [1.1]

Table 3: Computation times for learning $\delta\text{CVA}^{\theta}_{(t)}$ and the related sensitivities. The speedups measure the ratios of the total time taken by the benchmark bump sensitivities approach to the total time taken by each of the sensitivities.

Conclusion

Table 4 synthesizes our findings regarding CVA sensitivities, as far as their approximation quality to corresponding partial derivatives (for bump sensitivities) and their hedging abilities (regarding also the optimized sensitivities) are concerned. The winner that emerges as the best trade-off for each downstream task in **blue red** in the first row is identified by the same color in the list of sensitivities. **bench. bump** plays the role of market standard. Sensitivities that are novelties of this work are emphasized in yellow (smart bump sensitivities essentially mean standard bump sensitivities with less paths, but with the important implementation caveat mentioned after (1) PLE sensitivities were already introduced in the different context of SIMM computations in Albanese et al. (2017); EC sensitivities were introduced in Rockafellar and Uryasev (2000)).

	sensitivities	speed	stability	**local accuracy**	**CVA run-on hedge**
	bench. bump	very slow	very stable	benchmark	good
fast bump sensis	linear bump	fast	average	good	*good*
	smart bump	fast	stable	good	good
optimized sensis	EC sensis	fast	average	not applicable	very good
	PLE sensis	very fast	stable	not applicable	excellent
	LS w/o Γ	very fast	stable	not applicable	excellent

Table 4: Conclusions regarding sensitivities and hedging. By local accuracy of a bump sensitivity, we mean the accuracy of the approximation it provides for the corresponding partial derivative. *Italics* means tested but not reported in tables or figures in the paper.

Regarding the assessment of CVA risk, in the run-on CVA case (see Table 5 using the same color code as Table 4), we found out that neural net regression of conditional CVA results in likely more reliable (judging by the twin scores of the associated CVA learners) and also faster value-at-risk and expected shortfall estimates than CVA Taylor expansions based on bump sensitivities (such as the ones that inspire certain regulatory CVA capital charge formulas). But an LS proxy, linear diagonal quadratic in market bumps, provides an even quicker (as it is regressed without training) and almost equally reliable view on CVA risk as the neural net CVA. In the **run-off CVA** case (not represented in the table), the **neural net learner of CVA$_t$** in the risk mode (or nested CVA learner alike but in much longer time) allows one to get a consistent and dynamic view on CVA and counterparty default risk altogether.

	$\delta\text{CVA}_{(t)}$ learners	speed	stability	**twin accuracy**	$\delta\text{CVA}_{(t)}$ VaR and ES
nonparametric	nested MC	very slow	stable	very good	very conservative
	neural net	fast	average	good	conservative
linear(-quadratic) in market bumps	**bench. bump**	very slow	very stable	good/average for small/large t	aggressive
	LS w/ Γ in δZ_t	very fast	stable	good	conservative

Table 5: Conclusions regarding run-on CVA risk.

References

Abbas-Turki, L., S. Crépey, and B. Saadeddine (2023). Pathwise CVA regressions with oversimulated defaults. *Mathematical Finance 33*(2), 274–307.

Albanese, C., C. Benezet, and S. Crépey (2023). Hedging valuation adjustment and model risk. arXiv:2205.11834v2.

Albanese, C., S. Caenazzo, and M. Syrkin (2017). Optimising VAR and terminating Arnie-VAR. *Risk Magazine*, October.

Albanese, C., S. Crépey, R. Hoskinson, and B. Saadeddine (2021). XVA analysis from the balance sheet. *Quantitative Finance 21*(1), 99–123.

Antonov, A., S. Issakov, and A. McClelland (2018). Efficient SIMM-MVA calculations for callable exotics. *Risk Magazine*, August.

Capriotti, L., Y. Jiang, and A. Macrina (2017). AAD and least-square Monte Carlo: Fast Bermudan-style options and XVA Greeks. *Algorithmic Finance 6*(1-2), 35–49.

Rockafellar, R. T. and S. Uryasev (2000). Optimization of conditional value-at-risk. *Journal of risk 2*, 21–42.