Efficient valuation of (non-)linear products for xVA

Felix Wolf[1]
Joint work with Griselda Deelstra[1] and Lech Grzelak[2,3]
[1]Department of Mathematics, Université libre de Bruxelles
[2]Financial Engineering, Rabobank [3]Mathematical Institute, Universiteit Utrecht

April 19, 2022

Expected exposure

Let $V_t = \mathbb{E}_t^{\mathbb{Q}}\Big[\sum_j \frac{B_t}{B_{T_j}} H_{T_j}\Big]$ be the discounted value of an asset (or portfolio) with payoffs H_{T_j}.

The *positive exposure* of V at time t is $E^+(t) = \max(0, V_t)$.
At initial time t_0 we can observe the *expected positive exposure* at time t:

$$\mathrm{EE}(t_0, t) = \mathbb{E}_{t_0}^{\mathbb{Q}}\left[\frac{B_{t_0}}{B_t} \max(0, V_t)\right]$$

Expected exposure

Let $V_t = \mathbb{E}_t^{\mathbb{Q}}\Big[\sum_j \frac{B_t}{B_{T_j}} H_{T_j}\Big]$ be the discounted value of an asset (or portfolio) with payoffs H_{T_j}.

The *positive exposure* of V at time t is $E^+(t) = \max(0, V_t)$.
At initial time t_0 we can observe the *expected positive exposure* at time t:

$$\mathrm{EE}(t_0, t) = \mathbb{E}_{t_0}^{\mathbb{Q}}\left[\frac{B_{t_0}}{B_t}\max(0, V_t)\right]$$

Exposures are typically obtained by Monte Carlo simulation and the EE is obtained from the sample mean:

$$\mathrm{EE}(t_0, t) \approx \frac{1}{M}\sum_{j=1}^{M}\max\Big(0, \frac{B_{t_0}(\omega_j)}{B_t(\omega_j)} V_t(\omega_j)\Big).$$

Thus, we are interested in samples $V_t(\omega)$ of the random variable $V_t \mid F_{t_0}$.

Swap portfolio

Consider an interest rate (payer) swap V with price

$$V_t = \bar{N} \sum_{j=1}^{m} \tau_j P(t, T_j) \Big(\frac{P(t, T_{j-1}) - P(t, T_j)}{\tau_j P(t, T_j)} - K \Big).$$

In affine interest rate models, a ZCB at time t is given by

$$\begin{aligned} P(t, T) &= \mathbb{E}_t^{\mathbb{Q}} \big[\exp(- \int_t^T r_s \, \mathrm{d}s) \big] \\ &= \exp\big(A(t, T) + B(t, T) r_t \big), \end{aligned}$$

where r_t is a random variable (e.g. Gaussian).

Thus, at time t_0, V_t is a random variable in one *"risk factor"* r_t, following some distribution $\mathcal{L}$.

$$(V_t \mid \mathscr{F}_{t_0}) = (f(r_t) \mid \mathscr{F}_{t_0}) \sim \mathcal{L}$$

Approximating the portfolio

Let the portfolio Π consist of 1000 swaps $V^1, ..., V^{1000}$. Simulating the price of one portfolio realisation $\Pi_t(\omega)$ requires 1000 swap evaluations:

$$\Pi_t(\omega) = V_t^1(\omega) + \cdots + V_t^{1000}(\omega) =: g\big(r_t(\omega)\big).$$

Approximating the portfolio

Let the portfolio Π consist of 1000 swaps $V^1, ..., V^{1000}$. Simulating the price of one portfolio realisation $\Pi_t(\omega)$ requires 1000 swap evaluations:

$$\Pi_t(\omega) = V_t^1(\omega) + \cdots + V_t^{1000}(\omega) =: g\big(r_t(\omega)\big).$$

MC simulation requires:

- M expensive, exact evaluations $g\big(r_t(\omega_j)\big),\ j = 1, \ldots, M$.

Approximating the portfolio

Let the portfolio Π consist of 1000 swaps $V^1, ..., V^{1000}$. Simulating the price of one portfolio realisation $\Pi_t(\omega)$ requires 1000 swap evaluations:

$$\Pi_t(\omega) = V_t^1(\omega) + \cdots + V_t^{1000}(\omega) =: g\big(r_t(\omega)\big).$$

MC simulation requires:

- M expensive, exact evaluations $g\big(r_t(\omega_j)\big),\ j = 1, \ldots, M$.

Simplify with an approximation $\widetilde{g}_n \approx g$:

- n expensive, exact evaluations at the interpolation points: $\Big(\big(r_t^1, g(r_t^1)\big), \ldots, \big(r_t^n, g(r_t^n)\big)\Big)$
- (Compute the approximation)
- M cheap evaluations of the approximation $\widetilde{g}_n\big(r_t(\omega_j)\big),\ j = 1, \ldots, M$.

How to interpolate between distributions?

Sample transformation

For a continuous random variable Y with cumulative distribution function F_Y, it holds $F_Y(Y) \sim \mathcal{U}[0,1]$. Proof:

$$\begin{aligned} F_{F_Y(Y)}(u) &= \mathbb{P}[F_Y(Y) \leq u] = \mathbb{P}[F_Y^{-1}(F_Y(Y)) \leq F_Y^{-1}(u)] \\ &= \mathbb{P}[Y \leq F_Y^{-1}(u)] = F_Y(F_Y^{-1}(u)) = u \\ &= F_U(u). \end{aligned}$$

Sample transformation

For a continuous random variable Y with cumulative distribution function F_Y, it holds $F_Y(Y) \sim \mathcal{U}[0,1]$. Proof:

$$\begin{aligned} F_{F_Y(Y)}(u) &= \mathbb{P}[F_Y(Y) \leq u] = \mathbb{P}[F_Y^{-1}(F_Y(Y)) \leq F_Y^{-1}(u)] \\ &= \mathbb{P}[Y \leq F_Y^{-1}(u)] = F_Y(F_Y^{-1}(u)) = u \\ &= F_U(u). \end{aligned}$$

"Inverse transform sampling": Sample u from $\mathcal{U}[0,1]$ and set $y = F_Y^{-1}(u)$.

Sample transformation

For a continuous random variable Y with cumulative distribution function F_Y, it holds $F_Y(Y) \sim \mathcal{U}[0,1]$. Proof:

$$\begin{aligned} F_{F_Y(Y)}(u) &= \mathbb{P}[F_Y(Y) \leq u] = \mathbb{P}[F_Y^{-1}(F_Y(Y)) \leq F_Y^{-1}(u)] \\ &= \mathbb{P}[Y \leq F_Y^{-1}(u)] = F_Y(F_Y^{-1}(u)) = u \\ &= F_U(u). \end{aligned}$$

"Inverse transform sampling": Sample u from $\mathcal{U}[0,1]$ and set $y = F_Y^{-1}(u)$.

For two continuous random variables X and Y, we have $F_X(X) \sim F_Y(Y) \sim \mathcal{U}(0,1)$.

From a sample ξ of X we can obtain a sample y of Y via

$$y = F_Y^{-1}(F_X(\xi)).$$

Stochastic collocation sampling[1]

X is a random variable we can easily sample from (e.g. Gaussian), Y is expensive to sample from. We can relate samples:

$$y = F_Y^{-1}(F_X(\xi)).$$

This function $g := F_Y^{-1} \circ F_X$ is computationally expensive (inversion of F_Y).

[1] L.A. Grzelak, J.A.S. Witteveen, M. Suárez-Taboada, and C.W. Oosterlee. The stochastic collocation Monte Carlo sampler: highly efficient sampling from "expensive" distributions. Quantitative Finance, 19(2):339–356, 2019.

Stochastic collocation sampling[1]

X is a random variable we can easily sample from (e.g. Gaussian), Y is expensive to sample from. We can relate samples:

$$y = F_Y^{-1}(F_X(\xi)).$$

This function $g := F_Y^{-1} \circ F_X$ is computationally expensive (inversion of F_Y).

1. Find interpolation points $x_1, ..., x_n$ ("collocation points") and evaluate exactly: $y_i = g(x_i),\ i = 1, \ldots, n$.
2. Build approximation function $\widetilde{g}_n \approx g$ based on these n points.
3. Obtain (approximated) samples $\widetilde{y}_i = \widetilde{g}_n(\xi_i)$ of Y from (cheap) samples ξ_i of X.

---

[1] L.A. Grzelak, J.A.S. Witteveen, M. Suárez-Taboada, and C.W. Oosterlee. The stochastic collocation Monte Carlo sampler: highly efficient sampling from "expensive" distributions. Quantitative Finance, 19(2):339–356, 2019.

Why collocation?

"Classic interpolation" framework:

$$\Pi_t(\omega) = g(r_t(\omega)) \approx \widetilde{g}_n(r_t(\omega))$$

Collocation framework:

$$y = \left(F_Y^{-1} \circ F_X\right)(\xi) = g(\xi) \approx \widetilde{g}_n(\xi).$$

We know the cheap distribution X (e.g. interest rate) and the function g, but we do not need knowledge about the distribution of Y.

Why collocation?

"Classic interpolation" framework:

$$\Pi_t(\omega) = g(r_t(\omega)) \approx \widetilde{g}_n(r_t(\omega))$$

Collocation framework:

$$y = \left(F_Y^{-1} \circ F_X\right)(\xi) = g(\xi) \approx \widetilde{g}_n(\xi).$$

We know the cheap distribution X (e.g. interest rate) and the function g, but we do not need knowledge about the distribution of Y.

Difference to "standard" function interpolation: We evaluate $\widetilde{g}_n$ at random points from the known distribution X.

Why collocation?

"Classic interpolation" framework:

$$\Pi_t(\omega) = g(r_t(\omega)) \approx \widetilde{g}_n(r_t(\omega))$$

Collocation framework:

$$y = \left(F_Y^{-1} \circ F_X\right)(\xi) = g(\xi) \approx \widetilde{g}_n(\xi).$$

We know the cheap distribution X (e.g. interest rate) and the function g, but we do not need knowledge about the distribution of Y.

Difference to "standard" function interpolation: We evaluate $\widetilde{g}_n$ at random points from the known distribution X.

Example Lagrange polynomial over interpolation points $x_1, \dots, x_n$:

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi),$$

where

$$\ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$

Connection to Gaussian Quadrature

Let p_i be an orthogonal, polynomial basis in $L^2(X)$, i.e.

$$\int_a^b p_i(x)p_j(x)f_X(x)dx = \delta_{ij}\mathbb{E}[p_i(X)^2].$$

We want to find weights w_i and collocation points x_i, $i = 1, \ldots, n$, so that

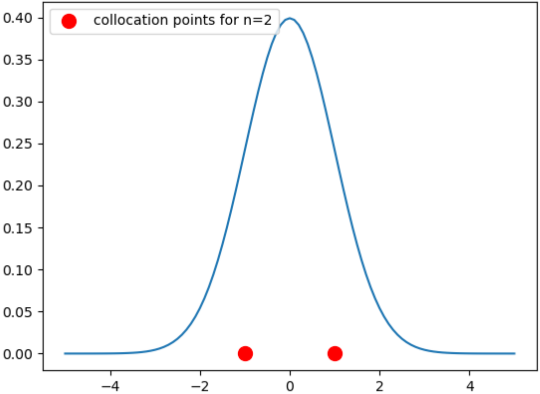
$$\int_a^b g(x)f_X(x)\mathrm{d}x \approx \sum_{i=1}^{n} g(x_i)w_i.$$

Find weights and points by consideration of enough exact integrals:

$$\int_a^b p_0(x)\mathrm{d}x = w_1p_0(x_1) + \cdots + w_np_0(x_n)$$
$$\int_a^b p_1(x)\mathrm{d}x = w_1p_1(x_1) + \cdots + w_np_1(x_n)$$
$$\ldots$$

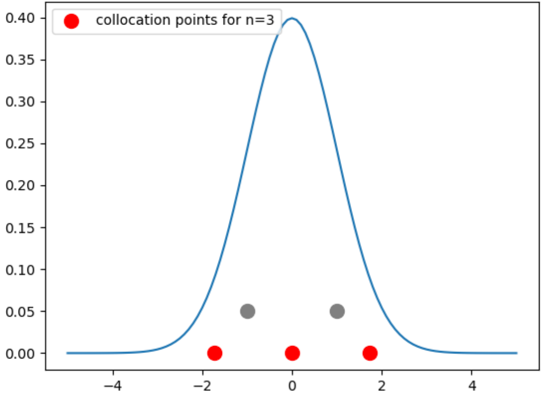
Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



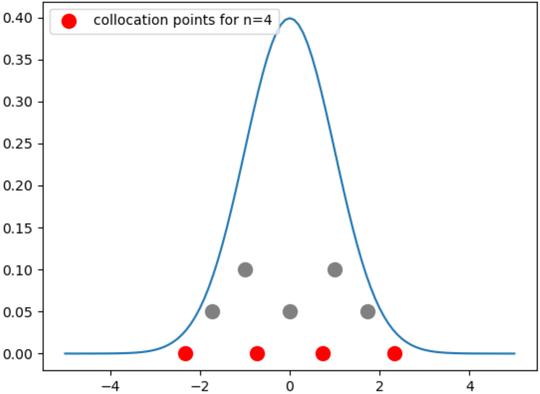
Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



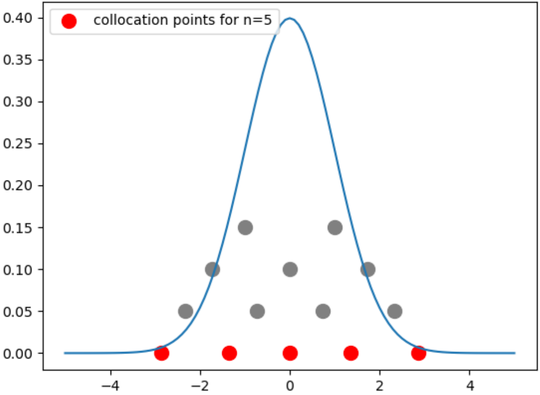
Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



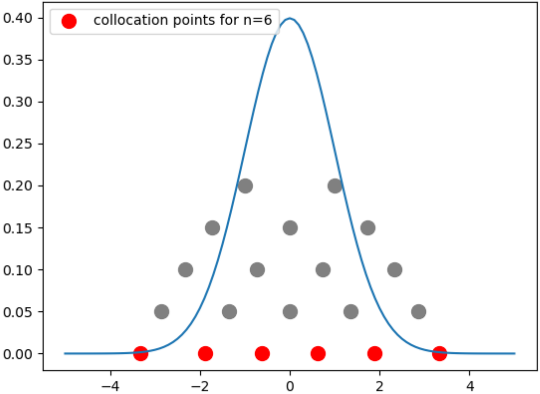
Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



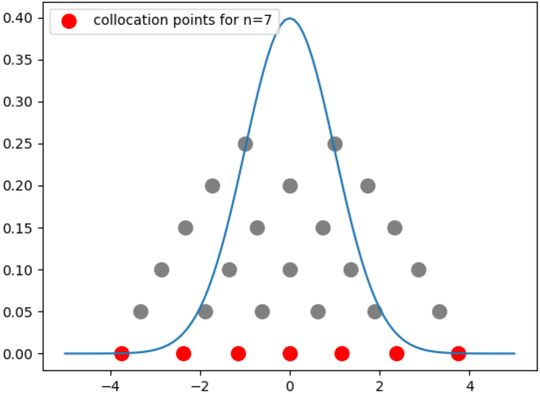
Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



Optimal Collocation Points

$$\widetilde{g}_n(\xi) = \sum_{i=1}^{n} g(x_i)\ell_i(\xi), \;\; \ell_i(\xi) = \prod_{j=1, j\neq i}^{n} \frac{\xi - x_j}{x_i - x_j}$$



Choice of interpolation function

The approximation $\widetilde{g}_n$ ...

- must be cheap to evaluate:
  "n exact $+$ M cheap evaluations $\ll$ M exact valuations"
- must offer high accuracy
- may preserve properties of g (e.g. monotonicity)
- may be differentiable

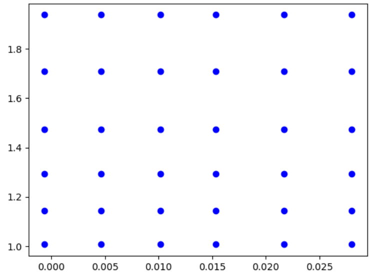
There are many options:

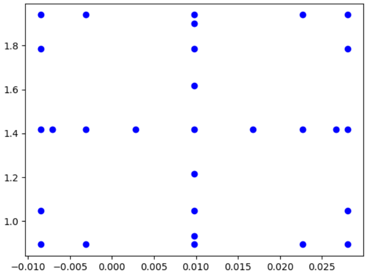
Lagrange polynomials, Chebyshev polynomials, Hermite polynomials, ...

Higher dimensions[2]

The number of interpolation points should not grow too fast.

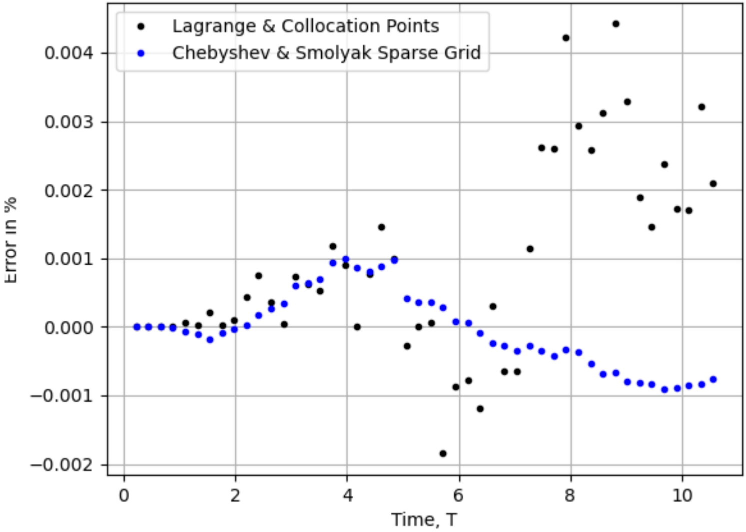
Cartesian grid of (optimal) collocation points vs. Smolyak sparse grid





[2] L.A. Grzelak. Sparse Grid Method for Highly Efficient Computation of Exposures for xVA. arXiv:2104.14319, 2021.

Hybrid portfolio of (many) stock contracts and swaps.



Many directions to investigate

- Error bounds for different interpolation methods (in higher dimensions)
- Interplay between interpolation points and interpolation methods
- Effects on portfolios of non-linear derivatives

Many directions to investigate

- Error bounds for different interpolation methods (in higher dimensions)
- Interplay between interpolation points and interpolation methods
- Effects on portfolios of non-linear derivatives

Thank you for listening!