AMERICAN OPTION PRICING WITH MODEL CONSTRAINED GAUSSIAN PROCESS REGRESSIONS

Donatien Hainaut

LIDAM Discussion Paper ISBA
2024 / 23

UCLouvain

LIDAM
ISBA | Louvain Institute of Data Analysis and Modeling in economics and statistics

ISBA

Voie du Roman Pays 20 - L1.04.01

B-1348 Louvain-la-Neuve

Email : lidam-library@uclouvain.be

https://uclouvain.be/en/research-institutes/lidam/isba/publication.html

American option pricing with model constrained Gaussian process regressions

Donatien Hainaut*
UCLouvain, ISBA-LIDAM

December 11, 2024

This article introduces a novel method based on Gaussian process regression for pricing American options. The variational partial differential equation (PDE) governing option prices is converted into a non-linear penalized Feynman-Kac equation (PFK). We propose an iterative algorithm to manage the non-linearity of the PFK operator. We sample state variables in the PDE's inner domain and on the terminal boundary. At each step, we fit a constrained regression function approximating the option price. This function matches the option payoffs on the boundary sample while satisfying the PFK PDE on the inner sample. The non-linear term in this PDE is frozen and valued with the price estimate from the previous iteration. We adopt a Bayesian framework in which payoffs and the value of the FK PDE in the boundary and inner samples are noised. Assuming the regression function is a Gaussian process, we find a closed-form approximation of option prices. In the numerical illustration, we evaluate American put options in the Heston model and in the two-factor Hull-White model.

KEYWORDS: Gaussian process regression, American option pricing, Feynman-Kac equation, Heston model

1 Introduction

This work introduces a novel iterative algorithm that relies on constrained Gaussian process regressions (CGPR) for pricing American options. We demonstrate that this method is an accurate and numerically efficient alternative to existing approaches, such as the least squares Monte Carlo method. Its methodological foundation is Gaussian process regression.

Gaussian processes are used for regressing a response y, on a vector of covariates $\boldsymbol{x}$, in various scientific fields. The reader may, for instance refer to Rasmussen and Williams (2006, chapter 6) or to (2012, chapter 15) for an introduction. The Gaussian process regression is a Bayesian method in which we encode our belief that responses y are drawn from a Gaussian process (GP), with zero mean and covariance function $k(\boldsymbol{x}, \boldsymbol{x}\prime)$, called kernel, prior to taking into account observations. The regression function is the a-posteriori expectation of this GP, conditioned by observations.

In financial econometrics, Gaussian process regressions are used for time-series forecasting, as

*Corresponding author. Postal address: Voie du Roman Pays 20, 1348 Louvain-la-Neuve (Belgium). E-mail to: donatien.hainaut(at)uclouvain.be in Wu et al. (2014), Han et al. (2016), or Petelin et al. (2019). In quantitative finance, GPs serve to regress derivative or asset prices on market state variables, as in De Spiegeleer et al. (2018), Goudenège et al. (2021). Crépey and Dixon (2019) use a Gaussian process regression for computing credit valuation adjustement (CVA) in a similar manner to Andersson and Oosterlee (2021) who instead use a neural network regression.

In a similar manner to Physics-Inspired Neural Networks (PINNs), a Gaussian process regression can be constrained by a partial differential equation (PDE). This PDE usually represents the physical law governing the experiment for which responses are measured. The literature on constrained Gaussian process regression (CGPR) is vast, and we refer the reader to Swiler et al. (2020) for a survey. Among the useful references therein, Graepel (2003) uses a CGPR and sampling to solve PDEs with noisy responses. Calderhead et al. (2009) propose a method for accelerating the sampling to solve nonlinear ordinary differential equations. Nguyen and Peraire (2015) develop procedures for handling responses of the PDE solution, given in the form of linear functions of state variables. Raissi et al. (2017) adapt the CGPR for discovering governing equations with parametric linear operators. Pförtner et al. (2022) use a CGPR for solving weak and strong formulations of linear PDEs.

The CGPR is a powerful tool that is underexploited in economics and finance. In a recent article, Hainaut and Vrins (2024) adapt it for pricing European options. In an arbitrage-free market, derivative prices are governed by the Feynman-Kac (FK) equation, which is a PDE of the form $\mathcal{L}_{\boldsymbol{x}} g = z(\boldsymbol{x})$ where g, $\mathcal{L}_x$ and $\boldsymbol{x}$ are, respectively, the value, a linear differential operator and a vector of risk processes. The boundary constraint is $g(\boldsymbol{x}) = h(\boldsymbol{x})$, where $h(\boldsymbol{x})$ is the derivative payoff at expiry. To solve this, Hainaut and Vrins (2024) fit a constrained regression on two samples of points, $\mathring{\boldsymbol{X}}$ and $\bar{\boldsymbol{X}}$, respectively in the inner and boundary domains of the PDE. In this setting, the price is a regression function $g(\boldsymbol{x})$ with $g(\boldsymbol{x}) = h(\boldsymbol{x})$ for all $\boldsymbol{x} \in \bar{\boldsymbol{X}}$,and a PDE constraint, $\mathcal{L}_{\boldsymbol{x}} \widehat{g} = z(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathring{\boldsymbol{X}}$. Hainaut and Vrins (2024) propose a regression function based on a GP regression and illustrate its efficiency for pricing options in the Heston model and basket options in a Black-Scholes market. There is a clear parallel with pricing based on PINNs, in which the CGPR is replaced by a neural network. For examples of pricing with PINNs, we refer e.g. to Sirignano and Spiliopoulos (2018), Al-Aradi et al. (2022), Glau and Wunderlich (2022), Hainaut (2024) and Hainaut and Casas (2024). The main drawback of PINNs compared to CGPR is that they rely on numerical differentiation for solving the FK equation.

This article extends the method of Hainaut and Vrins (2024) for pricing American options. These derivatives satisfy a variational inequality that is converted into a penalized Feynman-Kac (PFK) equation, as suggested in Zvan et al. (1998) or Forsyth and Vetzal (2002). The differential operator involved in the PFK equation is non-linear, and therefore we cannot directly apply the CGPR. To remedy this issue, we propose an iterative algorithm using a local linear version of the PFK equation. At each epoch, the option price is approximated by a CGPR. We show that the algorithm converges under mild conditions.

The paper is organized as follows. The next section presents the general valuation framework and the conversion of the variational equation into a penalized FK equation. We prove that under mild conditions, the solution of this equation can be found by iteratively solving intermediate linear PDEs. Section 3 explains how these intermediate PDEs are solved with a constrained Gaussian process regression. In Section 4, we apply our approach to pricing American put options in the Heston model. As a benchmark, we use the least squares Monte Carlo method. In the next section, we evaluate American puts on zero-coupon bonds in the two-factor Hull-White model.

2 Valuation by iterative approach

We consider a financial market ruled by $p-1$ risk processes, stored in a vector $(\boldsymbol{y}_t)_{t\geq 0} = \left(\left(y_t^{(1)},...,y_t^{(p-1)}\right)\right)_{t\geq 0}$. These processes are for instance, stock prices, an interest rate or a stochastic variance. They are defined on a complete probability space, Ω, endowed with a filtration $\mathbb{F} = (\mathcal{F}_t)_{t\geq 0}$ and a risk neutral measure, $\mathbb{Q}$. The risk-free rate, denoted by $r_t = r(t, \boldsymbol{y}_t)$, is a function of state variables and time. We assume that risk processes are ruled by a system of stochastic differential equation (SDE):

$$\mathrm{d}\boldsymbol{y}_t \quad = \quad \boldsymbol{\mu}_{\boldsymbol{y}}(t, \boldsymbol{y}_t)dt + \Sigma_{\boldsymbol{y}}(t, \boldsymbol{y}_t)d\boldsymbol{B}_t \,, \tag{1}$$

where $\boldsymbol{B} = (\boldsymbol{B}_t)_{t\geq 0}$ is a $(p-1)$-vector of independent Brownian motions, $\boldsymbol{\mu}_{\boldsymbol{y}}(.)$ is a vector of dimension and $\Sigma_{\boldsymbol{y}}(.)$ is a $(p-1)\times(p-1)$ matrix such that

$$\mathbb{E}\left(\int_0^t \boldsymbol{\mu}_{\boldsymbol{y}}(s, \boldsymbol{y}_s)\, ds\right) \leq \infty \quad ,\forall\, t < \infty \,,$$
$$\mathbb{E}\left(\int_0^t \Sigma_{\boldsymbol{y}}(s, \boldsymbol{y}_s)\Sigma_{\boldsymbol{y}}(s, \boldsymbol{y}_s)^\top\, ds\right) \leq \infty \quad ,\forall\, t < \infty$$

where $\mathbb{E}$ is the expectation under $\mathbb{Q}$. We aim to evaluate an American option expiring at time T with a payoff $h(t, \boldsymbol{y}_t)$, for $t \in [0, T]$. We denote the risk-free rate by $r(t, \boldsymbol{y}_t)$.

The price of this derivative at any time $t \leq T$ is a function of risk processes, i.e., $f\,(t, \boldsymbol{y}_t)$ where f is referred to as the *pricing function,* $f(t, \boldsymbol{y}_t)$. If the market is arbitrage-free, $f\,(t, \boldsymbol{y}_t)$ is equal to the expected discounted payoff:

$$f\,(t, \boldsymbol{y}_t) = \mathbb{E}\left(e^{-\int_t^{\tau\wedge T} r(u, \boldsymbol{y}_u)du}\, h(\tau\wedge T, \boldsymbol{y}_{\tau\wedge T}) \,|\, \mathcal{F}_t\right), \tag{2}$$

where τ is a $\mathcal{F}_t$-adapted stopping time defined by:

$$\tau = \inf\{s \geq 0 \,|\, h(s, \boldsymbol{y}_s) \geq f\,(s, \boldsymbol{y}_s)\}\,.$$

We denote by $\mathcal{X} = [0, T]\times\mathbb{R}^{p-1}$, the domain on which is defined the pricing function f. The inner and terminal boundary domains are respectively $\mathring{\mathcal{X}} = [0, T) \times \mathbb{R}^{p-1}$, $\bar{\mathcal{X}} = \{(T, \boldsymbol{y}) \,|\, \boldsymbol{y} \in \mathbb{R}^{p-1}\}$ and are such $\mathcal{X} = \mathring{\mathcal{X}} \cup \bar{\mathcal{X}}$. The gradient and the Hessian of $f(.)$ with respect to $\boldsymbol{y}$ are noted $\nabla_{\boldsymbol{y}} f$ and $\mathcal{H}_{\boldsymbol{y}}(f)$. To lighten further developments, we adopt the notation $\boldsymbol{x} = (t, \boldsymbol{y})$ for the $p-$vector of time and risk factors. In absence of arbitrage, the price of an American option is solution of a variational equation involving the following linear[1] differential operator

$$\mathcal{L}_{\boldsymbol{x}}\cdot = \partial_t \cdot - r(\boldsymbol{x})\cdot + \boldsymbol{\mu}_{\boldsymbol{y}}(\boldsymbol{x})^\top\nabla_{\boldsymbol{y}}\cdot + \frac{1}{2}\mathrm{tr}\left(\Sigma_{\boldsymbol{y}}(\boldsymbol{x})\Sigma_{\boldsymbol{y}}(\boldsymbol{x})^\top\mathcal{H}_{\boldsymbol{y}}\cdot\right)\,. \tag{3}$$

Using standard arguments, we can prove that $f(\boldsymbol{x})$, the derivative value, is solution of the variational equation:

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x})\ (h(\boldsymbol{x}) - f(\boldsymbol{x})) = 0 & \boldsymbol{x} \in \mathring{\mathcal{X}}, \\ h(\boldsymbol{x}) \leq f(\boldsymbol{x}) & \boldsymbol{x} \in \mathring{\mathcal{X}}, \\ \mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) \leq 0 & \boldsymbol{x} \in \mathring{\mathcal{X}}, \\ f(\boldsymbol{x}) = h(\boldsymbol{x}) & \boldsymbol{x} \in \bar{\mathcal{X}}. \end{cases} \tag{4}$$

Instead of solving this variational inequality, we propose a penalty approximation as described in Paragraphs 4.5.4 and 7.2 of Seydel (2017). We also refer the reader to Zvan et al. (1998) or

[1] $\mathcal{L}_x$ is linear if $\mathcal{L}_x(g^{(1)}(\boldsymbol{x}) + g^{(2)}(\boldsymbol{x})) = \mathcal{L}_x g^{(1)}(\boldsymbol{x}) + \mathcal{L}_x g^{(2)}(\boldsymbol{x})$.

Forsyth and Vetzal (2002) for a detailed analysis of this method. We set a penalty weight $\xi > 0$ and consider a non-linear PDE of the form:

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} f_\xi(\boldsymbol{x}) + \xi \left(h(\boldsymbol{x}) - f_\xi(\boldsymbol{x})\right)_+ = 0 & \boldsymbol{x} \in \mathring{\mathcal{X}} \\ f_\xi(\boldsymbol{x}) = h(\boldsymbol{x}) & \boldsymbol{x} \in \bar{\mathcal{X}} \end{cases} . \tag{5}$$

We call this equation the penalized Feynman-Kac (PFK) equation. If $h(\boldsymbol{x}) \geq f_\xi(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X}$, from Eq. (5), we infer that $\mathcal{L}_{\boldsymbol{x}} f_\xi(\boldsymbol{x}) \leq 0$. If $h(\boldsymbol{x}) < f_\xi(\boldsymbol{x})$ then $\left(h(\boldsymbol{x}) - f_\xi(\boldsymbol{x})\right)_+ = 0$ and from Eq. (5), $\mathcal{L}_{\boldsymbol{x}} f_\xi(\boldsymbol{x}) = 0$.

The left side of Eq. (5) is a non-linear operator and therefore we cannot directly apply the constrained Gaussian process regression (CGPR) proposed by Hainaut and Vrins (2024) for solving the PFK equation. Instead, we develop an iterative procedure based on intermediate linear problems and show the convergence. Let us denote by $f_\xi^{(u-1)}(\boldsymbol{x})$, the approximated American option price after $u-1$ iterations, with $u \in \mathbb{N}$. At the u^{th} step, $f_\xi^{(u-1)}(\boldsymbol{x})$ is obtained by solving the following intermediate PFK (IPFK) equation:

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} f_\xi^{(u)}(\boldsymbol{x}) = z^{(u-1)}(\boldsymbol{x}) & \boldsymbol{x} \in \mathring{\mathcal{X}} \\ f_\xi^{(u)}(\boldsymbol{x}) = h(\boldsymbol{x}) & \boldsymbol{x} \in \bar{\mathcal{X}} \end{cases} , \tag{6}$$

where $z^{(u)}(\boldsymbol{x})$ are updated as follows:

$$z^{(u)}(\boldsymbol{x}) \quad = \quad z^{(u-1)}(\boldsymbol{x}) + \theta \left[-\xi \left(h(\boldsymbol{x}) - f_\xi^{(u)}(\boldsymbol{x}) \right)_+ - z^{(u-1)}(\boldsymbol{x}) \right] . \tag{7}$$

We initialize the loop with $z^{(0)}(\boldsymbol{x}) = 0$. The parameter $\theta \in (0,1)$ is a learning rate tuning the convergence of the process. When $\theta = 1$, we retrieve the classical penalty approach. The next proposition states that $f_\xi^{(u)}$ converges toward the solution of the PFK equation under mild conditions.

Proposition 1. *If $f_\xi^{(u)}(\boldsymbol{x})$ admits a lower bound, $f_\xi^{(u)}(\boldsymbol{x})$ converges toward $f_\xi(\boldsymbol{x})$.*

Proof. By induction and as $z^{(0)} = 0$, we find that $\boldsymbol{z}^{(u)}(\boldsymbol{x})$ is a negative weighted sum of positive differences between the payoff and the approximated price computed at previous iterations:

$$\boldsymbol{z}^{(u)}(\boldsymbol{x}) = -\theta\xi \sum_{l=0}^{u-1} (1-\theta)^l \left(h(\boldsymbol{x}) - f_\xi^{(u-l)}(\boldsymbol{x}) \right)_+ .$$

By construction, the difference between $\mathcal{L}_{\boldsymbol{x}} f_\xi^{(u+1)}(\boldsymbol{x})$ and $\mathcal{L}_{\boldsymbol{x}} f_\xi^{(u)}(\boldsymbol{x})$ is equal to

$$\begin{aligned} \mathcal{L}_{\boldsymbol{x}} \left(f_\xi^{(u+1)}(\boldsymbol{x}) - f_\xi^{(u)}(\boldsymbol{x}) \right) &= z^{(u)}(\boldsymbol{x}) - z^{(u-1)}(\boldsymbol{x}) \\ &= \theta\xi \left(\sum_{l=0}^{u-2} (1-\theta)^l \left(h(\boldsymbol{x}) - f_\xi^{(u-l-1)}(\boldsymbol{x}) \right)_+ - \sum_{l=0}^{u-1} (1-\theta)^l \left(h(\boldsymbol{x}) - f_\xi^{(u-l)}(\boldsymbol{x}) \right)_+ \right) . \end{aligned}$$

Under the assumption that $f_\xi^{(u)}(\boldsymbol{x})$ is lower bounded, $\exists\, b \in \mathbb{R}^+$ such that $\left(h(\boldsymbol{x}) - f_\xi^{(u-l-1)}(\boldsymbol{x}) \right)_+ < b$.

$$\begin{aligned} \left| \mathcal{L}_{\boldsymbol{x}} \left(f_\xi^{(u+1)}(\boldsymbol{x}) - f_\xi^{(u)}(\boldsymbol{x}) \right) \right| &\leq \theta\xi \left(\sum_{l=0}^{u-2} (1-\theta)^l - \sum_{l=0}^{u-1} (1-\theta)^l \right) b \\ &= \theta\xi \left(1-\theta\right)^{u-1} b \,. \end{aligned}$$

Therefore, the limit when $u \to \infty$ is null:

$$\lim_{u\to\infty}\left|\mathcal{L}_{\boldsymbol{x}}\left(f_\xi^{(u+1)}(\boldsymbol{x})-f_\xi^{(u)}(\boldsymbol{x})\right)\right|=0\,.$$

Since $f_\xi^{(u+1)}(\boldsymbol{x})$ and $f_\xi^{(u)}(\boldsymbol{x})$ have the same terminal conditions, we infer that $\lim_{u\to\infty} f_\xi^{(u)}(\boldsymbol{x}) = f_\xi^{(\infty)}(\boldsymbol{x})$. As $f_\xi^{(u)}(\boldsymbol{x})$ admits a limit, $z^{(u)}(\boldsymbol{x})$ also converges:

$$\begin{aligned} z^{(\infty)}(\boldsymbol{x}) = \lim_{u\to\infty} \boldsymbol{z}^{(u)}(\boldsymbol{x}) &= -\theta\xi \lim_{u\to\infty}\left(\sum_{l=0}^{u-1}(1-\theta)^l\left(h(\boldsymbol{x})-f_\xi^{(u-l)}(\boldsymbol{x})\right)_+\right) \\ &= -\theta\xi\left(h(\boldsymbol{x})-f_\xi^{(\infty)}(\boldsymbol{x})\right)_+\left(\lim_{u\to\infty}\sum_{l=0}^{u-1}(1-\theta)^l\right) \\ &= -\xi\left(h(\boldsymbol{x})-f_\xi^{(\infty)}(\boldsymbol{x})\right)_+ \,. \end{aligned}$$

Therefore, the limit $f_\xi^{(\infty)}(\boldsymbol{x})$ solves the penalized FK Eq. (6) and is then equal to $f_\xi(\boldsymbol{x})$. □

The condition that $f_\xi^{(u)}(\boldsymbol{x})$ is lower bounded is in practice satisfied (as it is a positive price). We will apply this iterative procedure and solves the IPFK equation with a constrained Gaussian process regression (CGPR).

3 The constrained Gaussian process regression (CGPR)

In this section, we find an approximated solution to the IPFK Eq. (6) by estimating a constrained Gaussian process regression. We start by briefly reviewing the general principles of the CGPR.

We consider a series of Gaussian processes $\{g^{(u)}(\boldsymbol{x})\,,\,\boldsymbol{x}\in\mathcal{X}\}$, where $u\in\mathbb{N}$. By definition, a Gaussian process is a collection such that for any $d\in\mathbb{N}$ and $\boldsymbol{x}_1,\dots,\boldsymbol{x}_d\in\mathcal{X}$, the vector $\left(g^{(u)}(\boldsymbol{x}_1),\dots,g^{(u)}(\boldsymbol{x}_d)\right)$ is a multivariate Gaussian random variable. Without loss of generality, the mean is set to zero and the covariance function is defined by a kernel $k(\boldsymbol{x},\boldsymbol{x}')$:

$$\mathbb{C}\left(g^{(u)}(\boldsymbol{x}),g^{(u)}(\boldsymbol{x}')\right)=k(\boldsymbol{x},\boldsymbol{x}^{'})\quad\forall(\boldsymbol{x},\boldsymbol{x}^{'})\in\mathcal{X}\times\mathcal{X}\,.$$

A necessary and sufficient condition for the function $k:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ to be a valid covariance kernel is that the $d\times d$ matrix of $(k(\boldsymbol{x}_i,\boldsymbol{x}_j))_{i,j=1,\dots,d}$ is positive semidefinite for all possible samples.

The CGPR provides an approximation $\widehat{f_\xi^{(u)}}(\boldsymbol{x})$, of the pricing function $f_\xi^{(u)}(\boldsymbol{x}):\mathcal{X}\to\mathbb{R}$, after u iterations. We will see that $\widehat{f_\xi^{(u)}}(\boldsymbol{x})$ is defined as the a posteriori conditional expectation of $g^{(u)}(\boldsymbol{x})$. Let us detail this. In the CGPR framework, the functional form of $f_\xi^{(u)}(\boldsymbol{x})$ is unknown, but its values are observed at certain points of $\bar{\mathcal{X}}$ and are ruled by the IPFK Eq. (6) at other points of $\mathring{\mathcal{X}}$. For this reason, we consider two random samples. We store in a first set, d combinations of risk factors at expiry T, denoted as $\bar{\boldsymbol{X}}=\{\bar{\boldsymbol{x}}_i\}_{i=1,\dots,d}$, and the corresponding noised payoffs, $\boldsymbol{h}=(h_i)_{i=1,\dots,d}$. The $\bar{\boldsymbol{x}}_i$'s are randomly sampled from $\bar{\mathcal{X}}$ while $h_i=h(\bar{\boldsymbol{x}}_i)+\bar{\epsilon}$ are option payoffs, noised by $\bar{\epsilon}\sim\mathcal{N}(0,\sigma^{*2})$. The second training set contains m combinations of state variables before expiry, denoted as $\mathring{\boldsymbol{X}}=\{\mathring{\boldsymbol{x}}_i\}_{i=1,\dots,m}$ with $\mathring{\boldsymbol{x}}_i\in\mathring{\mathcal{X}}$. The noised values of the IFPK PDE (6) are grouped in a m-vector, $\boldsymbol{z}^{(u-1)}=\left(z_i^{(u-1)}\right)_{i=1,\dots,m}$ with $z_i^{(u-1)}=z^{(u-1)}(\mathring{\boldsymbol{x}}_i)+\mathring{\epsilon}$ with noise $\mathring{\epsilon}\sim N(0,\sigma^{*2})$.

As detailed by Hainaut and Vrins (2024), assuming noisy observations of $h(\bar{\boldsymbol{x}}_i)$ for $\bar{\boldsymbol{x}}_i \in \bar{\boldsymbol{X}}$ and $\mathcal{L}_{\boldsymbol{x}} g^{(u)}(\mathring{\boldsymbol{x}}_i)$ for $\mathring{\boldsymbol{x}}_i \in \mathring{\boldsymbol{X}}$, is a way to introduce a regularization term via a linear shrinkage of the covariance matrix $k(\boldsymbol{X}, \boldsymbol{X})$, analogous to Ridge estimators. We will come back to this point later. At each iteration $u \in \mathbb{N}$, we estimate a Gaussian process $g^{(u)}$ approximating the price function $f_{\xi}^{(u)}$ over $\bar{\boldsymbol{X}}$ and $\mathring{\boldsymbol{X}}$. The Gaussian process satisfies the following equalities on the two training sets:

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} g^{(u)}(\mathring{\boldsymbol{x}}_i) = z_i^{(u-1)} = z^{(u-1)}(\mathring{\boldsymbol{x}}_i) + \mathring{\epsilon} & \forall \mathring{\boldsymbol{x}}_i \in \mathring{\boldsymbol{X}}, \\ g^{(u)}(\bar{\boldsymbol{x}}_i) = h_i = h(\bar{\boldsymbol{x}}_i) + \bar{\epsilon} & \forall \bar{\boldsymbol{x}}_i \in \bar{\boldsymbol{X}}. \end{cases} \tag{8}$$

Eq. (8) is a discrete version of the the system (6). We recognize the structure of a constrained regression problem: we fit $g^{(u)}$ such that $\mathbb{E}\left(g^{(u)}(\bar{\boldsymbol{x}})\right) = h(\bar{\boldsymbol{x}})$ for $\bar{\boldsymbol{x}} \in \bar{\boldsymbol{X}}$, under the constraint that $\mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}} g^{(u)}(\mathring{\boldsymbol{x}})\right) = z^{(u-1)}(\mathring{\boldsymbol{x}})$ for $\mathring{\boldsymbol{x}} \in \mathring{\boldsymbol{X}}$. We denote by $\boldsymbol{H}$ and $\boldsymbol{Z}^{(u-1)}$, the random d- and m-vectors whose realizations are $\boldsymbol{h}$ and $\boldsymbol{z}^{(u-1)}$, the right hand term in Eq. (8).

To summarize, we a priori encode our belief that instances of $g^{(u)}(\boldsymbol{x})$, satisfying Eq. (8), are drawn from a Gaussian process with null mean and covariance $k(\boldsymbol{x}, \boldsymbol{x}')$, prior to taking into account observations $\boldsymbol{h}$ and $\boldsymbol{z}^{(u-1)}$ of payoffs and IPFK PDEs. In this Bayesian approach, the estimator of $f_{\xi}^{(u)}(\boldsymbol{x})$ is the a posteriori expectation of $g^{(u)}(\boldsymbol{x})$ conditioned by realizations of $\boldsymbol{Z}$ and $\boldsymbol{H}$:

$$\widehat{f_{\xi}^{(u)}}(\boldsymbol{x}) = \mathbb{E}\left(g^{(u)}(\boldsymbol{x}) \mid \boldsymbol{Z} = \boldsymbol{z}^{(u-1)}, \boldsymbol{H} = \boldsymbol{h}\right).$$

To estimate $g^{(u)}$, we develop it as a sum a basis functions weighted by random normal weights. This is feasible because the function $g^{(u)}(\boldsymbol{x})$ belongs to a reproducing kernel Hilbert space spanned by eigenfunctions of the kernel (see e.g. Chapter 6 of Rasmussen and Williams, 2006). Without loss of generality, we consider kernels defined by a finite number of eigenfunctions. In this case, there exist n basis functions $\boldsymbol{\varphi}(\boldsymbol{x}) = (\varphi_j(\boldsymbol{x}))_{j=1,\dots,n}$ such that

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\varphi}(\boldsymbol{x})^{\top} \boldsymbol{\varphi}(\boldsymbol{x}'), \ \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}.$$

This assumption is not restrictive as any kernel can be approximated by a finite sum for a given accuracy. As $g^{(u)}(\boldsymbol{x})$ is a-priori normal with null mean and $\mathbb{C}\left(g_{\xi}^{(u)}(\boldsymbol{x}), g_{\xi}^{(u)}(\boldsymbol{x}')\right) = k(\boldsymbol{x}, \boldsymbol{x}')$, $g_{\xi}^{(u)}(\boldsymbol{x})$ is a sum of functions $\varphi_j(.)$ weighted by a vector $\boldsymbol{w} \sim N(0, I_n)$:

$$g_{\xi}^{(u)}(\boldsymbol{x}) = \sum_{j=1}^{n} w_j \varphi_j(\boldsymbol{x}) = \boldsymbol{w}^{\top} \boldsymbol{\varphi}(\boldsymbol{x}). \tag{9}$$

We define the function $\boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x}) = (\mathcal{L}_{\boldsymbol{x}} \varphi_j(\boldsymbol{x}))_{j=1,\dots,n}$ from $\mathcal{X} \to \mathbb{R}^n$. Using the linearity of the differential operator (3), $\mathcal{L}_{\boldsymbol{x}} g_{\xi}^{(u)}(\boldsymbol{x})$ is the scalar product:

$$\mathcal{L}_{\boldsymbol{x}} g^{(u)}(\boldsymbol{x}) = \sum_{j=1}^{n} w_j \left(\mathcal{L}_{\boldsymbol{x}} \varphi_j(\boldsymbol{x})\right) = \boldsymbol{w}^{\top} \boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x}),$$

Next, we denote by $\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}$ the $m \times n$ matrix of $(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}})_{i,j} = \mathcal{L}_{\mathring{\boldsymbol{x}}_i} \varphi_j(\mathring{\boldsymbol{x}}_i)$ where $\mathring{\boldsymbol{x}}_i \in \mathring{\boldsymbol{X}}$ and by $\bar{\boldsymbol{\varphi}}$, the $d \times n$ matrix of $(\bar{\boldsymbol{\varphi}})_{i,j} = \varphi_j(\bar{\boldsymbol{x}}_i)$ with $\bar{\boldsymbol{x}}_i \in \bar{\boldsymbol{X}}$. Conditionally to $\boldsymbol{w}$, the joint distribution of $(Z^{(u-1)}, \boldsymbol{H})$, the random vectors of right-hand terms in Eq. (8) is a multivariate normal:

$$\left(\begin{array}{c} Z^{(u-1)} \\ \boldsymbol{H} \end{array}\right) \Bigg| \boldsymbol{w} \sim \mathcal{N}\left(\left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w} \\ \bar{\boldsymbol{\varphi}} \boldsymbol{w} \end{array}\right), \left(\begin{array}{cc} \sigma^{*2} I_m & \mathbf{0} \\ \mathbf{0} & \sigma^{*2} I_d \end{array}\right)\right). \tag{10}$$

From the properties of the multivariate normal distribution, the regression weights, conditionally to $\boldsymbol{z}^{(u-1)}$ and $\boldsymbol{h}$, have a normal distribution:

$$\boldsymbol{W}|\boldsymbol{z}^{(u-1)},\boldsymbol{h} \quad \sim \quad N\left(\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{(u-1)}\,;\,\Sigma_{\boldsymbol{z},\boldsymbol{h}}\right)\,,$$

where $\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{(u-1)}$, and $\Sigma_{\boldsymbol{z},\boldsymbol{h}}$ are respectively equal to

$$\begin{cases}\Sigma_{\boldsymbol{z},\boldsymbol{h}} & = \left(I_n + \sigma^{*-2}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^{\top}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} + \sigma^{*-2}\bar{\boldsymbol{\varphi}}^{\top}\bar{\boldsymbol{\varphi}}\right)^{-1}\,, \\ \boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{(u-1)} & = \sigma^{*-2}\Sigma_{\boldsymbol{z},\boldsymbol{h}}\left(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^{\top}\boldsymbol{z}^{(u-1)} + \bar{\boldsymbol{\varphi}}^{\top}\boldsymbol{h}\right)\,.\end{cases} \tag{11}$$

In later developments, we adopt the following notations:

$$\begin{cases} k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\boldsymbol{x}\right) & = \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\boldsymbol{\varphi}(\boldsymbol{x}) = \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_i}k(\boldsymbol{x},\mathring{\boldsymbol{x}}_i)\right)_{i=1,\ldots,m}\,, \\ k\left(\bar{\boldsymbol{X}},\boldsymbol{x}\right) & = \left(k\left(\bar{\boldsymbol{x}}_j,\boldsymbol{x}\right)\right)_{j=1,\ldots,d}\,, \\ k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right) & = \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^{\top} = \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_i}k(\bar{\boldsymbol{x}}_j,\mathring{\boldsymbol{x}}_i)\right)_{i=1,\ldots,m,j=1,\ldots.,d}\,, \\ k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right) & = \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_i}\mathcal{L}_{\mathring{\boldsymbol{x}}_j}k(\mathring{\boldsymbol{x}}_i,\mathring{\boldsymbol{x}}_j)\right)_{i,j=1,\ldots,m}\,, \\ k\left(\bar{\boldsymbol{X}},\bar{\boldsymbol{X}}\right) & = \left(k(\bar{\boldsymbol{x}}_i,\bar{\boldsymbol{x}}_j)\right)_{i,j=1,\ldots,d}\,. \end{cases}$$

The next propositions are directly adapted from Hainaut and Vrins (2024) (Propositions 4 and 5), and we refer the reader to this article for the proofs.

Proposition 2. *Let us define $\boldsymbol{\beta}(\boldsymbol{x})$, the $(m+d)$-vector:*

$$\boldsymbol{\beta}(\boldsymbol{x}) = \begin{pmatrix} k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\boldsymbol{x}\right) \\ k\left(\bar{\boldsymbol{X}},\boldsymbol{x}\right)\end{pmatrix}\,, \tag{12}$$

and the Gram matrix

$$C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right) \quad = \quad \sigma^{*2}I_{m+d} + \begin{pmatrix} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right) & k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right) \\ k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{\top} & k\left(\bar{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)\end{pmatrix}\,. \tag{13}$$

The estimator $\widehat{f}_{\xi}^{(u)}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{W}|\boldsymbol{z}^{(u-1)},\boldsymbol{h}}\left(g^{(u)}(\boldsymbol{x})\right)$ of $f_{\xi}^{(u)}(\boldsymbol{x})$ solving Eq. (6) is given by

$$\widehat{f}_{\xi}^{(u)}(\boldsymbol{x}) \quad = \quad \boldsymbol{\beta}(\boldsymbol{x})^{\top}C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{-1}\begin{pmatrix}\boldsymbol{z}^{(u-1)} \\ \boldsymbol{h}\end{pmatrix}. \tag{14}$$

As previously mentioned, the Gaussian noises added to h_i's and z_i's introduce a regularization term, that is the diagonal matrix $\sigma^{*2}I_{m+d}$, in the Gram matrix (13). This term stabilizes a possibly ill-conditioned matrix. The standard deviation σ^* may then be viewed as a parameter tuning the numerical robustness of the CGPR. Of course, this affects the accuracy of the solution but we will see in the numerical illustrations that its impact is limited. The next proposition provides the conditional variance of the estimator. For a proof, we refer the reader to Proposition 5 of Hainaut and Vrins (2024).

Proposition 3. *Let $\boldsymbol{\beta}(\boldsymbol{x})$ be the $(m+d)$ vector defined in Eq. (12). The conditional variance of $g^{(u)}(\boldsymbol{x})$ is equal to*

$$\mathbb{V}_{\boldsymbol{W}|,\boldsymbol{z}^{(u-1)},\boldsymbol{h}}\left(g^{(u)}(\boldsymbol{x})\right) = k(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\beta}(\boldsymbol{x})^{\top}C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{-1}\boldsymbol{\beta}(\boldsymbol{x})\,. \tag{15}$$

Frame 1 presents the iterative algorithm for computing American option prices. This algorithm converges if $\boldsymbol{g}^{(u)}(\boldsymbol{x})$ is lower bounded. This condition may in theory be breached but is satisfied in practice if the learning rate is small enough.

Algorithm 1 Computation of the approximated price $\widehat{f_{\xi}^{(n)}}(\boldsymbol{x})$ of a derivative with payoff $h(\boldsymbol{x})$.

```
1. Sample $m$ points $\mathring{\boldsymbol{x}}_i = (\mathring{t}_i, \mathring{\boldsymbol{y}}_i) \in \mathring{\mathcal{X}}$
2. Sample $d$ points $\bar{\boldsymbol{x}}_i = (T, \bar{\boldsymbol{y}}_i) \in \bar{\mathcal{X}}$ and set $(h_i = h(\bar{\boldsymbol{x}}_i) + \epsilon)_{i=1...d}$ where $\epsilon \sim N(0, \sigma^*)$.
3. Draw a sample $z_i^{(0)} \sim N(0, \sigma^*)$ for $i = 1, ..., m$.
4. Compute the Gram matrix, and its inverse:
   $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) = \sigma^{*2} I_{m+d} + \begin{pmatrix} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right) & k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) \\ k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{\top} & k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) \end{pmatrix}$
   Calculate vectors $\boldsymbol{\beta}(\boldsymbol{x}_i) = \begin{pmatrix} k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) \\ k\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right) \end{pmatrix}$ for $i = 1, ..., m$.
5. Loop on $u \in \mathbb{N}$ until convergence
   a) $\forall\ \mathring{\boldsymbol{x}}_i \in \mathring{\boldsymbol{X}}$ evaluate $\widehat{f_{\xi}^{(u)}}(\mathring{\boldsymbol{x}}_i) = \boldsymbol{\beta}(\mathring{\boldsymbol{x}}_i)^{\top} C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z}^{(u-1)} \\ \boldsymbol{h} \end{pmatrix}$ and $h(\mathring{\boldsymbol{x}}_i)$ and store
      them in vectors $\widehat{\boldsymbol{f}_{\xi,\mathring{\boldsymbol{X}}}^{(u)}}$ and $\boldsymbol{h}_{\mathring{\boldsymbol{X}}}$.
   b) Update the vector $\boldsymbol{z}^{(u)}$ as follows:
      $\boldsymbol{z}^{(u)} = \boldsymbol{z}^{(u-1)} + \theta \left[ -\xi \left( \boldsymbol{h}_{\mathring{\boldsymbol{X}}} - \widehat{\boldsymbol{f}_{\xi,\mathring{\boldsymbol{X}}}^{(u)}} \right)_+ - \boldsymbol{z}^{(u-1)} \right].$
6. If convergence after $n$ iterations, return $\widehat{f_{\xi}^{(n)}}(\boldsymbol{x})$ and $\boldsymbol{z}^{(n-1)}$.
```

Fitting the CGPR requires applying the differential operator $\mathcal{L}_{\boldsymbol{x}}$ twice to the kernel function. Although many variants exist (see e.g. Beckers, 2021), we chose a Gaussian kernel in the numerical illustrations. The main motivation being that it is easy to differentiate and yields good results. Let us denote $\boldsymbol{x} = (t, \boldsymbol{y})$, $\boldsymbol{y} = (y_1, ..., y_{p-1})$, $\tilde{\boldsymbol{x}} = (\tilde{t}, \tilde{\boldsymbol{y}})$ and $\tilde{\boldsymbol{y}} = (\tilde{y}_1, ..., \tilde{y}_{p-1})$. The Gaussian kernel has the general form:

$$k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \quad = \quad \exp\left(-\frac{\left(t - \tilde{t}\right)^2}{2\,\eta_t^2} - \sum_{k=1}^{p-1} \frac{(y_k - \tilde{y}_k)^2}{2\,\eta_k^2} \right) \quad \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}, \tag{16}$$

where $\boldsymbol{\eta} = (\eta_t, \eta_1, ..., \eta_{p-1})^{\top}$is a p-vector in $\mathbb{R}_0^{p,+}$ of hyper-parameters. The coefficients $\boldsymbol{\beta}(\boldsymbol{x})$ of the CGPR have, in this case, a closed-form expression. The last d elements of $\boldsymbol{\beta}(\boldsymbol{x})$ are indeed equal to $(k(\bar{\boldsymbol{x}}_i, \boldsymbol{x}))_{i=1,...,d}$. The first m items of $\boldsymbol{\beta}(\boldsymbol{x})$ require the calculation of $\mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x}, \tilde{\boldsymbol{x}})$. For this purpose, we introduce the following p-vector:

$$\boldsymbol{d}_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \begin{pmatrix} \left(t - \tilde{t}\right) / \eta_t^2 \\ (y_1 - \tilde{y}_1) / \eta_1^2 \\ \vdots \\ (y_{p-1} - \tilde{y}_{p-1}) / \eta_{p-1}^2 \end{pmatrix},$$

and the $(p-1)\times(p-1)$ matrix:

$$\boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \;=\; \begin{pmatrix} \frac{(y_1-\tilde{y}_1)^2-\eta_1^2}{\eta_1^4} & \frac{(y_1-\tilde{y}_1)(y_2-\tilde{y}_2)}{\eta_1^2\eta_2^2} & \cdots & \frac{(y_1-\tilde{y}_1)(y_{p-1}-\tilde{y}_{p-1})}{\eta_1^2\eta_{p-1}^2} \\ \frac{(y_1-\tilde{y}_1)(y_2-\tilde{y}_2)}{\eta_1^2\eta_2^2} & \ddots & \ddots & \frac{(y_2-\tilde{y}_2)(y_{p-1}-\tilde{y}_{p-1})}{\eta_2^2\eta_{p-1}^2} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{(y_1-\tilde{y}_1)(y_{p-1}-\tilde{y}_{p-1})}{\eta_1^2\eta_{p-1}^2} & \frac{(y_2-\tilde{y}_2)(y_{p-1}-\tilde{y}_{p-1})}{\eta_2^2\eta_{p-1}^2} & \cdots & \frac{(y_{p-1}-\tilde{y}_{p-1})^2-\eta_{p-1}^2}{\eta_{p-1}^4} \end{pmatrix}.$$

By direct differentiation, we obtain the gradient and Hessian of the kernel:

$$\begin{cases} \nabla_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x},\tilde{\boldsymbol{x}}) & = \boldsymbol{d}_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\, k(\boldsymbol{x},\tilde{\boldsymbol{x}})\,, \\ \mathcal{H}_{\tilde{\boldsymbol{y}}} k(\boldsymbol{x},\tilde{\boldsymbol{x}}) & = \boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\, k(\boldsymbol{x},\tilde{\boldsymbol{x}})\,. \end{cases}$$

We infer from these expressions, that $\mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x},\tilde{\boldsymbol{x}})$ can be rewritten in terms of $\boldsymbol{d}_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})$ and $\boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})$:

$$\mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \left(\frac{1}{2}\mathrm{tr}\left(\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^{\top}\boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\right) + \left(1\,,\,\boldsymbol{\mu}_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^{\top}\right)\boldsymbol{d}_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) - r\right) k(\boldsymbol{x},\tilde{\boldsymbol{x}})\,, \tag{17}$$

which allows us to evaluate the first m elements of $\boldsymbol{\beta}(\boldsymbol{x})$. To conclude this section, we remark that Eq. (14) and (15) correspond to the conditional expectation and variance of $g^{(u)}(\boldsymbol{x}) \mid \boldsymbol{z}^{(u-1)}, \boldsymbol{h}$. All variables being Gaussian, the triplet $\left(g^{(n)}(\boldsymbol{x}), \boldsymbol{Z}^{(n-1)}, \boldsymbol{H}\right)$ where n is the last iteration of Algorithm 1, has a multivariate normal distribution:

$$\begin{bmatrix} g^{(n)}(\boldsymbol{x}) \\ \boldsymbol{Z}^{(n-1)} \\ \boldsymbol{H} \end{bmatrix} \;\sim\; N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x},\boldsymbol{x}) & \boldsymbol{\beta}(\boldsymbol{x})^{\top} \\ \boldsymbol{\beta}(\boldsymbol{x}) & C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right) \end{bmatrix}\right).$$

This observation allows us to eventually optimize hyper-parameters $\boldsymbol{\eta}$ by maximizing the marginal log-likelihood of $(\boldsymbol{Z}^{(n-1)}, \boldsymbol{H})$:

$$\begin{aligned} \log l(\boldsymbol{z}^{(n-1)}, \boldsymbol{h}) = \frac{m}{2}\log 2\pi - \frac{1}{2}\log\left|C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)\right| \\ -\frac{1}{2}\begin{pmatrix} \boldsymbol{z}^{(n-1)} \\ \boldsymbol{h} \end{pmatrix}^{\top} C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{-1}\begin{pmatrix} \boldsymbol{z}^{(n-1)} \\ \boldsymbol{h} \end{pmatrix}. \end{aligned} \tag{18}$$

This step is nevertheless computationally intensive and subject to numerical instability. An alternative consists in selecting $\boldsymbol{\eta}$ such hat CGPR prices are close to prices obtained with another method, such as the least square Monte-Carlo method.

We will compare option prices obtained by the CGPR and by the Least-Square Monte-Carlo (LSMC) method of Longstaff, and Schwartz (2001). This method uses a sample of N paths of risk factors, simulated by an Euler discretization of Eq. (1). At each time step, the option price is approximated by a polynomial regression of cash-flows on state variables. This procedure is described in Algorithm 2. An alternative method consists of numerically solving the variational pricing equation. For an illustration of this approach, we refer the reader, e.g., to Madi et al. (2018), who use a finite element method and implicit time steps to determine the price of an American option.

Algorithm 2 Least-Square Monte-Carlo algorithm for the valuation of an American option.

```
1. Initialization:
    a) Set $\Delta_t = \frac{T}{M}$ and simulate $N$ paths of $y_t^{(1)}, ..., y_t^{(p-1)}$ stored in $N \times M$ matrix $Y_1$,..., $Y_{p-1}$.
    b) Initialize the payoff at maturity: $P = h\left(T, Y_1[:, M], ..., Y_{p-1}[:, M]\right)$
    c) Set initial cash-flows: $CF = P$

2. Iterate backwards in time from $t = M - 1$ to $t = 0$:
    a) Calculate the $N$ vector of discount factors, $D = \exp\left(-r\left(t, (Y_1[:, t], ..., Y_{p-1}[:, t])\right)\Delta_t\right)$
    b) Update cash-flows by discounting: $CF = D\,CF$.
    c) Identify in-the-money paths:
           $$\text{ITM} = \{i : h\left(t, (Y_1[i, t], ..., Y_{p-1}[i, t])\right) > 0\}$$
       If no paths are in-the-money, continue to the next time step.
    d) For in-the-money paths:
        i. Set $X_1 = Y_1[\text{ITM}, t]$, ... , $X_{p-1} = Y_{p-1}[\text{ITM}, t]$ ) and $Z = CF[\text{ITM}]$, in-the-money CF's.
        ii. Construct the regression matrix $A = \begin{bmatrix} 1 & X_1 & X_1^2 & X_2 & X_2^2 & X_1X_2, \ldots \end{bmatrix}$
        iii. Solve for regression coefficients using least squares: $\boldsymbol{\alpha}^* = \text{argmin}_{\boldsymbol{\alpha}} \left\|A\boldsymbol{\alpha} - Z\right\|_2$
        iv. Estimate the continuation value: $CV = A\boldsymbol{\alpha}^*$
    e) Calculate the exercise value: $EV = h\left(t, (X_1, ..., X_{p-1})\right)$
    f) Determine whether to exercise: $\text{E} = (EV > CV)$
    g) Update cash-flows: $CF[\text{E}] = EV[\text{E}]$

3. Final Step: return the discounted average of cash-flows:
           $$\text{Option Price} = \frac{1}{N}\sum_{i=1}^{N} CF[i]\,.$$
```

4 American options valuation in the Heston model

In this section, we evaluate American options in the Heston model with the Algorithm 1. The stock price, $(S_t)_{t\geq 0}$, is a geometric Brownian diffusion with a stochastic variance process, $(V_t)_{t\geq 0}$. These processes are defined on a probability space $(\Omega, \mathcal{F}, \mathbb{Q})$ generated by two independent Brownian motions $\boldsymbol{B}_t = \left(B_t^{(1)}, B_t^{(2)}\right)_{t\geq 0}^{\top}$. The risk-free rate is assumed constant, $r \in \mathbb{R}$. Risk factors $\boldsymbol{y_t} = (S_t, V_t)$, are ruled by the stochastic differential equation (1) where $\boldsymbol{\mu_y}(.)$ is a vector of dimension 2 and $\Sigma_{\boldsymbol{y}}(.)$ is a 2×2 matrix:

$$\boldsymbol{\mu_y}(t, \boldsymbol{y}_t) = \begin{pmatrix} r\,S_t \\ \kappa\left(\gamma - V_t\right) \end{pmatrix}, \; \Sigma_{\boldsymbol{y}}(t, \boldsymbol{y}_t) = \begin{pmatrix} S_t\sqrt{V_t}\Sigma_S^{\top} \\ \sigma\sqrt{V_t}\Sigma_V^{\top} \end{pmatrix}.$$

Parameters κ, $\gamma \in \mathbb{R}^+$ are the speed of reversion, and mean-reversion level of the variance. $\sigma \in \mathbb{R}^+$ is the volatility of the stock variance. Σ_S and Σ_V are 2-dimensional vectors such that the matrix

$$\Sigma = \begin{pmatrix} \Sigma_S^\top \\ \Sigma_V^\top \end{pmatrix} = \begin{pmatrix} \sqrt{1-\rho_{SV}^2} & \rho_{SV} \\ 0 & 1 \end{pmatrix}, \; \Sigma\Sigma^\top = \begin{pmatrix} 1 & \rho_{SV} \\ \rho_{SV} & 1 \end{pmatrix}.$$

Σ is the upper Choleski decomposition of the correlation matrix between the price and its variance (coefficient of correlation ρ_{SV}). We refer the reader to the Chapter 3 of Hainaut (2023) for other features of this model. We note $\boldsymbol{x} = (t, s, v)$, the vector of time t, and $\boldsymbol{y_t} = (S_t, V_t)$, the vector of risk processes. We focus on an American Put option of maturity T and strike price, K. The payoff function is defined by:

$$h(\boldsymbol{x}) \;\; := \;\; (K-s)_+ \; \forall \boldsymbol{x} \in \mathring{\mathcal{X}} \cup \bar{\mathcal{X}},$$

where the inner and boundary domains are respectively $\mathring{\mathcal{X}} = [0,T) \times \mathbb{R}^{+,2}$ and $\bar{\mathcal{X}} = T \times \mathbb{R}^{+,2}$. The value of the derivative is the expected discounted payoff under the risk neutral measure, such as defined in Eq. (2). The approximated prices after u iterations of Algorithm 1, $g^{(u)}$, solve the IPFK Eq. (6), where the gradient and the Hessian of $g^{(u)}(.)$ with respect to $\boldsymbol{y}$ by $\nabla_{\boldsymbol{y}} g^{(u)}$ and $\mathcal{H}_{\boldsymbol{y}}(g^{(u)})$ are equal to:

$$\nabla_{\boldsymbol{y}} g^{(u)} = \begin{pmatrix} \partial_s g^{(u)} \\ \partial_v g^{(u)} \end{pmatrix} \quad , \quad \mathcal{H}_{\boldsymbol{y}}(g^{(u)}) = \begin{pmatrix} \partial_{ss} g^{(u)} & \partial_{sv} g^{(u)} \\ \partial_{sv} g^{(u)} & \partial_{vv} g^{(u)} \end{pmatrix}.$$

We consider a Gaussian kernel defined by a positive vector of bandwidth parameters $\boldsymbol{\eta} = (\eta_t, \eta_s, \eta_v) \in \mathbb{R}^{+,3}$:

$$k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \;\; = \;\; \exp\left(-\frac{\left(t-\tilde{t}\right)^2}{2\,\eta_t^2} - \frac{(s-\tilde{s})^2}{2\,\eta_s^2} - \frac{(v-\tilde{v})^2}{2\,\eta_v^2} \right) \quad \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}. \tag{19}$$

In order to compute $k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right)$, $k_{\mathcal{L}}\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right)$ and $k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ involved in the definition of $\boldsymbol{\beta}(\boldsymbol{x})$ and $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$, we develop the expression of the FK operator applied to the kernel:

$$\begin{aligned} \mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \;\; &= \;\; \Big(\partial_{\tilde{t}} - r + r\tilde{s}\partial_{\tilde{s}} + \kappa(\gamma - \tilde{v})\,\partial_{\tilde{v}} + \frac{1}{2}\tilde{s}^2\tilde{v}\,\partial_{\tilde{s}\tilde{s}} \\ &\quad + \frac{1}{2}\sigma^2\tilde{v}\,\partial_{\tilde{v}\tilde{v}} + \tilde{s}\,\tilde{v}\,\sigma\,\rho_{SV}\partial_{\tilde{s}\tilde{v}} \Big)\; k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \\ &= \;\; l_{\tilde{x}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})\, k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \;, \end{aligned} \tag{20}$$

where $l_{\tilde{x}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ is the following function

$$\begin{aligned} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \Bigg[& \frac{\left(t-\tilde{t}\right)}{\eta_t^2} - r + r\tilde{s}\frac{(s-\tilde{s})}{\eta_s^2} + \kappa(\gamma-\tilde{v})\frac{(v-\tilde{v})}{\eta_v^2} \\ & + \frac{1}{2}\tilde{s}^2\tilde{v}\left(\frac{(s-\tilde{s})^2}{\eta_s^4} - \frac{1}{\eta_s^2} \right) + \frac{1}{2}\sigma^2\tilde{v}\left(\frac{(v-\tilde{v})^2}{\eta_v^4} - \frac{1}{\eta_v^2} \right) \\ & + \tilde{s}\,\tilde{v}\,\sigma\rho_{SV}\frac{(s-\tilde{s})(v-\tilde{v})}{\eta_s^2\eta_v^2} \Bigg]. \end{aligned} \tag{21}$$

To calculate $k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right)$ and $k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right)$, we apply a second time the FK operator to $\mathcal{L}_{\tilde{\boldsymbol{x}}} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$. A relatively long but simple calculation leads to the following expression:

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{x}} \mathcal{L}_{\tilde{\boldsymbol{x}}} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = {} & k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_t l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_t \, k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \\
& - r\, l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})\, k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) + r\, s\, \left(k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_s l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_s k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \\
& + \kappa\left(\gamma - v\right)\left(k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_v l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_v k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \\
& + \frac{1}{2} s^2 v \, \left(k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_{ss} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + 2 \partial_s l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_s k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_{ss} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \\
& + \frac{1}{2} \sigma^2 v \, \left(k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_{vv} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + 2 \partial_v l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_v k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_{vv} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \\
& + s\, v\, \sigma\, \rho_{SV} \, \left(\partial_s l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_v k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) + \partial_v l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_s k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \\
& + s\, v\, \sigma\, \rho_{SV} \left(k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \partial_{sv} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \partial_{sv} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)\right) \, ,
\end{aligned} \tag{22}$$

where the partial derivatives are given in Tables 1 and 2.

$\partial_s l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{1}{\eta_s^2} \begin{pmatrix} r\tilde{s} + \tilde{s}^2 \tilde{v} \frac{(s-\tilde{s})}{\eta_s^2} \\ + \tilde{s}\, \tilde{v}\, \sigma \rho_{SV} \frac{(v-\tilde{v})}{\eta_v^2} \end{pmatrix}$	$\partial_{ss} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{\tilde{s}^2 \tilde{v}}{\eta_s^4}$
$\partial_v l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{1}{\eta_v^2} \begin{pmatrix} \kappa\left(\gamma - \tilde{v}\right) + \sigma^2 \tilde{v} \frac{(v-\tilde{v})}{\eta_v^2} \\ + \tilde{s}\, \tilde{v}\, \sigma \rho_{SV} \frac{(s-\tilde{s})}{\eta_s^2} \end{pmatrix}$	$\partial_{vv} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{\sigma^2 \tilde{v}}{\eta_v^4}$
$\partial_t l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{1}{\eta_t^2}$	$\partial_{sv} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \tilde{s}\, \tilde{v}\, \frac{\sigma \rho_{SV}}{\eta_s^2 \eta_v^2}$

Table 1: Partial derivatives of $l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$, in Eq. (22).

$\partial_{ss} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = \left(\frac{(s-\tilde{s})^2}{\eta_s^4} - \frac{1}{\eta_s^2}\right) k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$	$\partial_t k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = -\frac{\left(t-\tilde{t}\right)}{\eta_t^2} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$
$\partial_{vv} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = \left(\frac{(v-\tilde{v})^2}{\eta_v^4} - \frac{1}{\eta_v^2}\right) k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$	$\partial_s k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = -\frac{(s-\tilde{s})}{\eta_s^2} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$
$\partial_{sv} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = \frac{(s-\tilde{s})(v-\tilde{v})}{\eta_s^2 \eta_v^2} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$	$\partial_v k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) = -\frac{(v-\tilde{v})}{\eta_v^2} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$

Table 2: Partial derivatives of $k(\boldsymbol{x}, \tilde{\boldsymbol{x}})$, in Eq. (22).

Table 3 provides the market, kernel, and algorithm parameters used in the numerical illustration. The points in $\mathring{\boldsymbol{X}}$ and $\bar{\boldsymbol{X}}$ are randomly drawn according to a uniform law on intervals reported in Table 3. The values chosen for kernel parameters η and σ^*, offer a good trade-off between accuracy and robustness. We discuss at the end of this section the influence of the shrinkage parameter σ^* on the accuracy.

<table>
<tr><td colspan="5">Market parameters</td></tr>
<tr><td>r</td><td>0.05</td><td>σ</td><td colspan="2">0.30</td></tr>
<tr><td>γ</td><td>0.15^2</td><td>s_i</td><td colspan="2">$\in [0, 300]$, uniform law</td></tr>
<tr><td>κ</td><td>0.95</td><td>t_i</td><td colspan="2">$\in [0, T]$, uniform law</td></tr>
<tr><td>ρ_{SV}</td><td>-0.30</td><td>v_i</td><td colspan="2">$\in [0\,,\,8\gamma]$, uniform law</td></tr>
<tr><td colspan="5">Kernel parameters</td></tr>
<tr><td>η_t</td><td>2.00</td><td>η_S</td><td colspan="2">20.00</td></tr>
<tr><td>η_v</td><td>0.40</td><td>σ^*</td><td colspan="2">0.025</td></tr>
<tr><td colspan="5">Algorithm 1 parameters</td></tr>
<tr><td>ξ</td><td>20</td><td>θ</td><td colspan="2">0.03</td></tr>
<tr><td colspan="3">Stopping criterion:</td><td>$\widehat{f_\xi^{(u)}}(\boldsymbol{x}) - \widehat{f_\xi^{(u-1)}}(\boldsymbol{x})$</td><td>$\leq 0.01$</td></tr>
</table>

Table 3: Market, kernel, algorithm parameters and boundaries for sampling $\boldsymbol{x}_i = (t_i, s_i, v_i)$ in $\mathring{\mathcal{X}}$ and $\bar{\mathcal{X}}$.

t	0
T	0.5 , 1 , 1.5 , 2 , 2.5 , 3
K	100
S_0	20 equispaced values from 50 to 150
V_0	20 equispaced values from $\frac{\gamma}{5}$ to 5γ.

Table 4: Features of 2400 options in the test set.

Table 5 reports computation times and spreads between 2400 American puts (with features in Table 4), priced with the CGPR and LSMC algorithms. Notice that for a single maturity, the inverted Gram matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ only needs to be computed once. This considerably reduces the valuation time, compared to the LSMC.

The LSMC is run with $M = 2000$ simulations and hundred time steps per year, $N = 100\,T$. We use a second order regression for determining the continuation region. For the GP method, the size m of the sample $\mathring{\boldsymbol{X}}$ varies from 300 to 500 points. The boundary sample, $\bar{\boldsymbol{X}}$, has the same size, $d = m$.

The spreads in absolute values between CGPR and LSMC prices are small and seem inversely proportional to m. The average relative spread for in-the-Money (ITM) options[2] varies from 1.90% to 3.36% and is acceptable given that both LSMC and CGPR yield numerical approximated prices.

The left plot of Figure 1 compares LSMC and CGPR prices of 1-year American puts with $m = 500$ and $\sqrt{V_0} = 24.76\%$. This emphasizes the accuracy of the CGPR approach compared to a standard LSMC. The right plot of the same figure, shows the heatmap of average absolute spreads between LSMC and CGPR call prices with respect to S_0 and $\sqrt{V_0}$. Figure 2 presents similar heatmaps with respect to T, S_0 and to T, $\sqrt{V}_0$. Whatever the combination of T, S_0 and $\sqrt{V}_0$, the spreads between LSMC and CGPR prices remain very small.

[2] We limit the calculation of relative spreads to in-the-Money options to avoid interpretability and numerical issues caused by out of the money options, whose prices are nearly null.

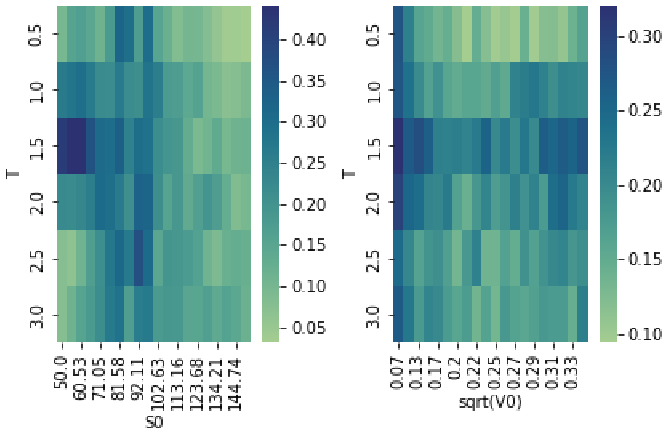

Figure 2: Right: heatmap of average absolute spreads between LSMC and CGPR (m=500) put prices with respect to T and S_0. Left: same heatmap but with respect to T and $\sqrt{V_0}$.

	Spread CGPR - LSMC prices		Times (seconds)	
m	Absolute	Relative (ITM)	CGPR	LSMC
300	0.3719	0.0315	1.61	232.26
350	0.2346	0.0214	1.44	234.64
400	0.2239	0.0180	1.78	240.00
450	0.2321	0.0192	2.11	233.76
500	0.1904	0.0147	2.15	237.42

Table 5: Absolute: average absolute values of the spread between put prices. Relative : average relative spreads for in-the-Money (ITM) options ($S_0 < K$). CGPR and LSMC pricing times of 2400 options with CGPR and LSMC methods.

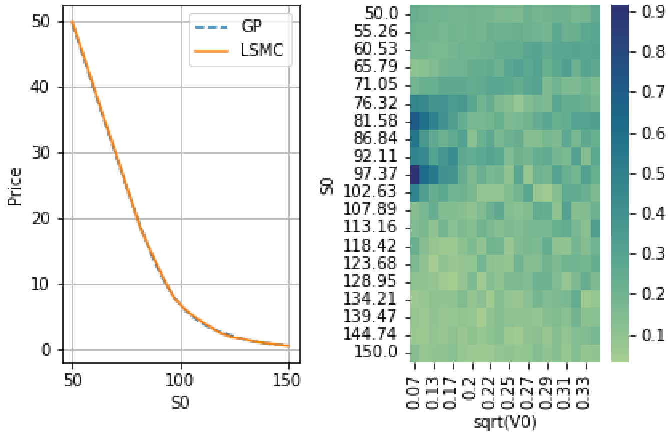

Figure 1: Left: LSMC and CGPR (GP) 1-year put prices computed with $m = 500$ and $\sqrt{V_0} = 24.76\%$. Right: heatmap of average absolute spreads between LSMC and CGPR put prices with respect to S_0 and $\sqrt{V_0}$.

Table 6 reports the average absolute and relative (ITM) spreads between CGPR and LSMC prices, per maturity. The average absolute and relative (ITM) spreads remain relatively stable regardless the time horizon. We also see that the pricing algorithm needs on average around 30 iterations to converge.

	Spread CGPR - LSMC prices		Number of
T	Absolute	Relative (ITM)	iterations
0.5	0.1377	0.0147	17.0
1.0	0.1932	0.0145	26.0
1.5	0.2417	0.0176	26.0
2.0	0.2103	0.0153	30.0
2.5	0.1804	0.0142	31.0
3.0	0.1790	0.0119	30.0

Table 6: . Average absolute and relative spreads for put options and in-the-Money (ITM) options ($S_0 > K$) per maturity. GP prices are computed with $m = 500$. Number of iterations: average number of loops in Algorithm 1.

We conclude this section by discussing the influence of σ^* on the accuracy and stability of the CGPR. In practice, σ^* is a Tikhonov regularization parameter that prevents numerical instability when inverting the Gram matrix, $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$. If σ^* is too small, we may fail to numerically invert the matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$, while a large σ^* generates biased prices. To better understand the impact of σ^*, we evaluate 1-year American puts with S_0 and V_0 from Table 4. Table 7 reports average absolute and relative (ITM) spreads between LSMC and CGPR prices for σ^* ranging from 0.015 to 0.115. The kernel parameters are those of Table 3. We clearly see that spreads increase with σ^*. Setting σ^* to 0.015 leads to an average relative (ITM) spread of 1.41%. This spread climbs to 1.62% with $\sigma^* = 0.115$.

	Spread CGPR - LSMC prices		Number of
σ^*	Absolute	Relative (ITM)	iterations
0.015	0.1913	0.0141	25.0
0.040	0.181	0.0143	25.0
0.065	0.168	0.0144	23.0
0.090	0.1616	0.0151	21.0
0.115	0.1612	0.0162	19.0

Table 7: . Absolute value and relative spreads for put options and in-the-Money (ITM) options ($S_0 < K$) per maturity. m=500 and σ^* ranges from 0.015 to 0.115. Number of iterations: average number of loops in Algorithm 1.

5 American options in a 2 factors Hull & White model

In this second example, we evaluate American options on a zero-coupon bond, in the interest rate model of Hull and White (1994), with two factors. In this setting, the risk-free rate $(r_t)_{t\geq 0}$ is a mean-reverting process driven by two independent Brownian motions, $\boldsymbol{B}_t = \left(B_t^{(1)}, B_t^{(2)}\right)^\top_{t\geq 0}$. We define a vector $\Sigma_r = \left(\rho\,,\, \sqrt{1-\rho^2}\right)^\top$ where ρ is the coefficient of correlation between the two factors ruling r_t. The interest rate has the following dynamic under the risk neutral measure:

$$dr_t = \kappa\left(\gamma(t) - r_t\right)dt + \sigma_r \Sigma_r^\top d\boldsymbol{B}_t\,,$$

where $\gamma(t)$ is a function of time. We denote by $P(0,t) = \mathbb{E}\left(e^{-\int_0^t r_u du} | \mathcal{F}_0\right)$, the value of a discount bond and by $f(0,t) = -\partial_t \ln P(0,t)$, the instantaneous forward rate. From Brigo and Mercurio (2006), Chapter 4, the model matches the initial yield curve of interest rate if $\gamma(t)$ is related to forward rates by:

$$\gamma(t) \quad = \quad \frac{1}{\kappa}\partial_t f(0,t) + f(0,t) + \frac{\sigma_r^2}{2\kappa^2}\left(1 - e^{-2\kappa t}\right).$$

In later developments, we denote by $B_\kappa(t,T)$, the following function

$$B_\kappa(t,T) = \frac{1}{\kappa}\left(1 - e^{-\kappa(T-t)}\right).$$

The price at time $t \leq S$ of a discount bond of maturity S is linked to the initial yield curve by the relation

$$\begin{aligned} P(t,S) \quad &= \quad \exp\left(-\left(r_t - f(0,t)\right) B_\kappa(t,S) + \ln\frac{P(0,S)}{P(0,t)}\right) \\ &\qquad \times \exp\left(-\frac{\sigma_r^2}{4\kappa}\left(\left(1 - e^{-2\kappa t}\right) B_\kappa(t,S)^2\right)\right). \end{aligned} \tag{23}$$

Using the Itô's lemma, we can show that the bond price dynamic is geometric:

$$\frac{dP(t,S)}{P(t,S)} = r_t\, dt - B_\kappa(t,S)\sigma_r \Sigma_r^\top d\boldsymbol{B}_t. \tag{24}$$

As we evaluate American put options on $P(t,S)$, we select as risk factors, the interest rate and the zero-coupon bond of maturity S. They are stored in a vector $\boldsymbol{y_t} = (r_t, P(t,S))$, which is ruled by the stochastic differential equation (1) where $\boldsymbol{\mu_y}(.)$ is a vector of dimension 2 and $\Sigma_{\boldsymbol{y}}(.)$ is a 2×2 matrix:

$$\boldsymbol{\mu_y}(t,\boldsymbol{y}_t) = \begin{pmatrix} \kappa\left(\gamma(t) - r_t\right) \\ P(t,S)\, r_t \end{pmatrix}, \; \Sigma_{\boldsymbol{y}}(t,\boldsymbol{y}_t) = \begin{pmatrix} \sigma_r \Sigma_r^\top \\ -P(t,S) B_{\kappa_r}(t,S)\sigma_r \Sigma_r^\top \end{pmatrix}.$$

We denote by $\boldsymbol{x} = (t, r, P)$, the vector of time, interest rate $r_t = r$ and bond price $P(t,S) = P$. For an option of maturity T and strike price P, the payoff function is defined by:

$$h\left(\boldsymbol{x}\right) \quad := \quad \left(K - P(t,S)\right)_+ \; \forall \boldsymbol{x} \in \mathring{\mathcal{X}} \cup \bar{\mathcal{X}},$$

where the inner and boundary domains are $\mathring{\mathcal{X}} = [0,T) \times \mathbb{R} \times \mathbb{R}^+$ and $\bar{\mathcal{X}} = T \times \times \mathbb{R} \times \mathbb{R}^+$. The price of the derivative is equal to the expected discounted payoff under the risk neutral measure such as defined in Equation (2). The approximated prices after u iterations of Algorithm 1, $g^{(u)}$, solve the penalized FK equation (6), where the gradient and the Hessian of $g^{(u)}(.)$ with respect to $\boldsymbol{y}$ by $\nabla_{\boldsymbol{y}} g^{(u)}$ and $\mathcal{H}_{\boldsymbol{y}}(g^{(u)})$ are equal to:

$$\nabla_{\boldsymbol{y}} g^{(u)} = \begin{pmatrix} \partial_r g^{(u)} \\ \partial_P g^{(u)} \end{pmatrix} \quad , \quad \mathcal{H}_{\boldsymbol{y}}(g^{(u)}) = \begin{pmatrix} \partial_{rr} g^{(u)} & \partial_{rP} g^{(u)} \\ \partial_{Pr} g^{(u)} & \partial_{PP} g^{(u)} \end{pmatrix}.$$

The Gaussian kernel is defined by a three bandwidth parameters $\boldsymbol{\eta} = (\eta_t, \eta_r, \eta_P) \in \mathbb{R}^{+,3}$:

$$k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \quad = \quad \exp\left(-\frac{\left(t - \tilde{t}\right)^2}{2\,\eta_t^2} - \frac{(r - \tilde{r})^2}{2\,\eta_r^2} - \frac{\left(P - \tilde{P}\right)^2}{2\,\eta_P^2}\right) \quad \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}. \tag{25}$$

As in the previous example, we first develop the expression of the FK operator $\mathcal{L}_{\tilde{\boldsymbol{x}}}\cdot$, applied to the Gaussian kernel:

$$\begin{aligned} \mathcal{L}_{\tilde{\boldsymbol{x}}} k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \quad &= \quad \Big(\partial_{\tilde{t}} - \tilde{r} + \tilde{r}\tilde{P}\partial_{\tilde{P}} + \kappa\left(\gamma(\tilde{t}) - \tilde{r}\right)\partial_{\tilde{r}} + \frac{1}{2}\tilde{P}^2\sigma_r^2 B_\kappa(\tilde{t},S)^2\partial_{\tilde{P}\tilde{P}} \\ &\qquad + \frac{1}{2}\sigma_r^2\,\partial_{\tilde{r}\tilde{r}} - \tilde{P}\,B_\kappa(\tilde{t},S)\,\rho\,\sigma_r^2\,\partial_{\tilde{P}\tilde{r}}\Big)\; k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \\ &= \quad l_{\tilde{x}}\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right), \end{aligned} \tag{26}$$

where $l_{\tilde{x}}\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$ is the following function:

$$\begin{aligned} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = &\left[\frac{\left(t-\tilde{t}\right)}{\eta_t^2} - \tilde{r} + \tilde{r}\tilde{P}\frac{\left(P-\tilde{P}\right)}{\eta_P^2} + \kappa\left(\gamma(\tilde{t})-\tilde{r}\right)\frac{(r-\tilde{r})}{\eta_r^2}\right. \\ &+\frac{1}{2}\tilde{P}^2\sigma_r^2 B_\kappa(\tilde{t},S)^2\left(\frac{\left(P-\tilde{P}\right)^2}{\eta_P^4}-\frac{1}{\eta_P^2}\right)+\frac{1}{2}\sigma_r^2\left(\frac{(r-\tilde{r})^2}{\eta_r^4}-\frac{1}{\eta_r^2}\right) \\ &\left.-\tilde{P}\,B_\kappa(\tilde{t},S)\,\rho\,\sigma_r^2\frac{\left(P-\tilde{P}\right)(r-\tilde{r})}{\eta_r^2\eta_P^2}\right]. \end{aligned} \tag{27}$$

To evaluate the matrix $k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right)$, we need $\mathcal{L}_{\boldsymbol{x}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$. A direct calculation leads to the following expression:

$$\begin{aligned} \mathcal{L}_{\boldsymbol{x}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = &\ k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_t l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_t\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) \\ &-r\, l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) + r\,P\,\left(k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_P l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_P k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right) \\ &+\kappa\left(\gamma(t)-r\right)\left(k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_r l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_r k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right) + \frac{1}{2}P^2\sigma_r^2 B_\kappa(t,S)^2 \\ &\times\left(k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_{PP} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + 2\partial_P l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_P k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_{PP} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right) \\ &+\frac{1}{2}\sigma_r^2\ \left(k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_{rr} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + 2\partial_r l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_r k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_{rr} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right) \\ &-P\,B_\kappa(t,S)\,\rho\,\sigma_r^2\left(\partial_P l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_r k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) + \partial_r l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_P k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right) \\ &-P\,B_\kappa(t,S)\,\rho\,\sigma_r^2\left(k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\partial_{Pr} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\partial_{Pr} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right)\ , \end{aligned} \tag{28}$$

where the partial derivatives are given in Tables 8 and 9. Equations (26) and (28) allows us to calculate the Gram matrix $C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)$ and the vectors $\boldsymbol{\beta}(\boldsymbol{x}_i)$ in Algorithm 1.

$\partial_P l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{1}{\eta_P^2}\begin{pmatrix} r\tilde{P} + \tilde{P}^2\sigma_r^2 B_\kappa(\tilde{t},S)^2\frac{\left(P-\tilde{P}\right)}{\eta_P^2} \\ -\tilde{P}\,B_\kappa(\tilde{t},S)\,\rho\,\sigma_r^2\frac{(r-\tilde{r})}{\eta_r^2}\end{pmatrix}$	$\partial_{PP} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\tilde{P}^2\sigma_r^2 B_\kappa(\tilde{t},S)^2}{\eta_P^4}$
$\partial_r l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{1}{\eta_r^2}\begin{pmatrix} \kappa\left(\gamma(\tilde{t})-\tilde{r}\right) + \sigma_r^2\frac{(r-\tilde{r})}{\eta_r^2} \\ -\tilde{P}\,B_\kappa(\tilde{t},S)\,\rho\,\sigma_r^2\frac{\left(P-\tilde{P}\right)}{\eta_P^2}\end{pmatrix}$	$\partial_{rr} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\sigma_r^2}{\eta_r^4}$
$\partial_t l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{1}{\eta_t^2}$	$\partial_{Pr} l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = -\frac{\tilde{P}\,B_\kappa(\tilde{t},S)\,\rho\,\sigma_r^2}{\eta_r^2\eta_P^2}$

Table 8: Partial derivatives of $l_{\tilde{\boldsymbol{x}}}(\boldsymbol{x},\tilde{\boldsymbol{x}})$, in Equation (28).

$\partial_{PP} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = \left(\frac{\left(P-\tilde{P}\right)^2}{\eta_P^4} - \frac{1}{\eta_P^2}\right) k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$	$\partial_t k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = -\frac{\left(t-\tilde{t}\right)}{\eta_t^2}\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$
$\partial_{rr} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = \left(\frac{(r-\tilde{r})^2}{\eta_r^4} - \frac{1}{\eta_r^2}\right) k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$	$\partial_P k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = -\frac{\left(P-\tilde{P}\right)}{\eta_P^2}\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$
$\partial_{Pr} k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = \frac{\left(P-\tilde{P}\right)(r-\tilde{r})}{\eta_P^2\eta_r^2}\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$	$\partial_r k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = -\frac{(r-\tilde{r})}{\eta_r^2}\, k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)$

Table 9: Partial derivatives of $k(\boldsymbol{x},\tilde{\boldsymbol{x}})$, in Equation (28).

In the numerical illustration, we consider a Nelson-Siegel model for modeling instantaneous forward rates. Details and parameters of this model are provided in Appendix A. Other market, kernel and algorithm parameters are reported in Table 10.

Market parameters			
σ_r	0.03	κ	0.50
ρ	-0.30	P_i	$\in [0, 200]$, uniform law
t_i	$\in [0, T]$, uniform law	r_i	$\in [-0.05\,,\,0.10]$, uniform law
Kernel parameters			
η_t	0.85	η_P	20.00
η_r	0.10	σ^*	0.05
Algorithm 1 parameters			
ξ	20	θ	0.03
Stopping criterion:		$\widehat{f_\xi^{(u)}}(\boldsymbol{x}) - \widehat{f_\xi^{(u-1)}}(\boldsymbol{x})$	≤ 0.01

Table 10: Interest rate, kernel, algorithm parameters and boundaries for sampling $\boldsymbol{x}_i = (t_i, P_i, r_i)$ in $\mathring{\mathcal{X}}$ and $\bar{\mathcal{X}}$.

t	0
T	0.5 , 1 , 1.5 , 2 , 2.5 , 3
S	15
K	100
$P(0, S)$	20 equispaced values from 50 to 150
r_0	20 equispaced values from -0.02 to 0.08.

Table 11: Features of 2400 options in the test set.

We price 2400 American put options with features reported in Table 11, for various initial values of r_0 and $P(0, S)$. The CGPR is benchmarked to the LSMC with $M = 2000$ simulations and $N = 100T$. We again use a second order polynomial for determining the continuation region. For the GP method, the size m of the sample $\mathring{\boldsymbol{X}}$ varies from 300 to 500 points. The boundary sample, $\bar{\boldsymbol{X}}$, has the same size, $d = m$.

Table 12 shows average absolute and relative (ITM) spreads as a function of the sample sets size, m. Increasing m globally reduces the valuation discrepancies between the CGPR and LSMC. With $m = 500$, the average relative (ITM) spread falls to 2%, which is acceptable as both LSMC and CGPR yield numerical approximated prices.

	Spread CGPR - LSMC prices		Times (seconds)	
m	Absolute	Relative (ITM)	CGPR	LSMC
300	0.2626	0.0256	1.69	135.37
350	0.2863	0.0335	1.91	131.21
400	0.2172	0.0320	2.04	130.87
450	0.1900	0.0278	2.38	132.93
500	0.1901	0.0199	2.47	132.02

Table 12: Absolute: absolute value of spreads between put prices. Relative : relative spreads for in-the-Money (ITM) options ($P(0, S) < K$). CGPR and LSMC pricing times of 2400 options with CGPR and LSMC methods.

The left plot of Figure 3 compares LSMC to CGPR prices of 1-year American puts with $m = 500$ and $r_0 = 3.26\%$ for $P(0, S)$ ranging from 50 up to 150. In this setting, the CGPR and LSMC prices are nearly indiscernible. The heatmaps in Figure 3 and 3 emphasize that the accuracy of the CGPR depends on the triplets $(r_0, P(0, S), T)$. The largest gaps between CGPR

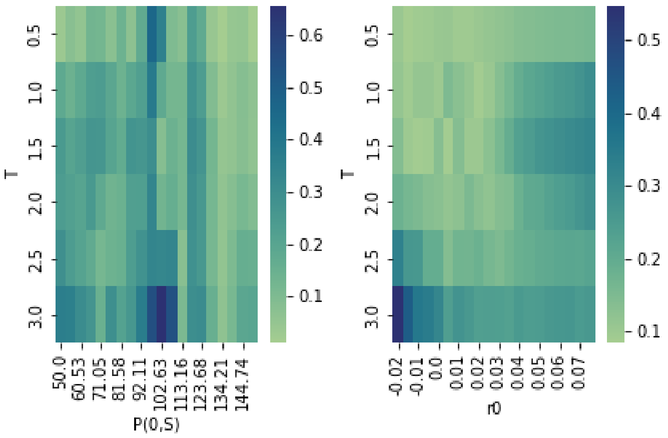

Figure 4: Right: heatmap of average absolute spreads between LSMC and CGPR (m=500) put prices with respect to T and $P(0,S)$. Left: same heatmap but with respect to T and r_0.

and LSMC are obtained for options with the longest maturities, around-the-money with r_0 close to boundaries -0.02 or 0.07.

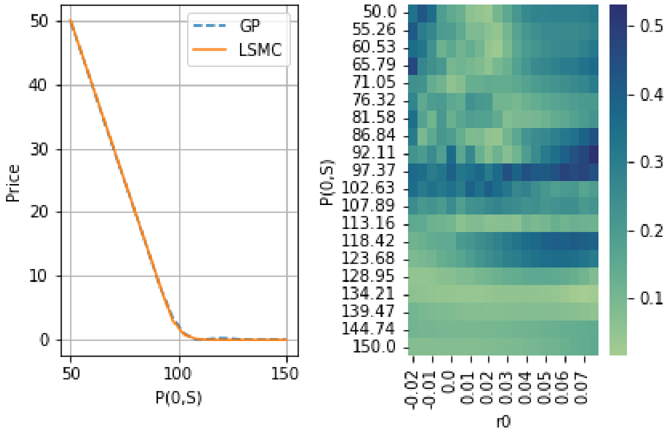

Figure 3: Left: LSMC and CGPR (GP) 1-year put prices computed with $m = 500$ and $r_0 = 3.26\%$. Right: heatmap of average absolute spreads between LSMC and CGPR put prices with respect to r_0 and $P(0,S)$.

Table 13 presents the average absolute and relative spreads between CGPR and LSMC prices for each option maturity. The absolute spreads slightly increase with T, whereas the relative ITM spreads are smaller for 2 and 2.5-year options. We also observe that the algorithm 1 needs more simulations to converge when T increases.

	Spread CGPR - LSMC prices		Number of
T	Absolute	Relative (ITM)	iterations
0.5	0.1215	0.0211	20.0
1.0	0.1677	0.0214	25.0
1.5	0.1849	0.0201	26.0
2.0	0.1777	0.0182	28.0
2.5	0.2014	0.0171	32.0
3.0	0.2872	0.0218	34.0

Table 13: . Average absolute and relative spreads for put options and in-the-Money (ITM) options ($P(0,S) < K$) per maturity. CGPR prices are computed with $m = 500$. Number of iterations: number of loops in Algorithm 1.

Finally, we analyze in Table 14 the impact of the shrinkage parameter σ^* on the discrepancies between 1-year CGPR and LSMC put prices with r_0 and $P(0,S_0)$, randomly drawn according to Table 11. Similar to Heston American options, increasing σ^* improves the numerical stability of the algorithm but it also raises the bias between SGPR and LSMC prices. We also see that σ^* has no impact on the number of algorithm iterations.

	Spread CGPR - LSMC prices		Number of
σ^*	Absolute	Relative (ITM)	iterations
0.01	0.1473	0.0170	25.0
0.02	0.1607	0.0185	25.0
0.03	0.1654	0.0198	25.0
0.04	0.1665	0.0207	25.0
0.05	0.1677	0.0214	25.0
0.06	0.1694	0.0220	25.0
0.07	0.1716	0.0226	25.0
0.08	0.1741	0.0231	25.0

Table 14: . Absolute value and relative spreads for put options and in-the-Money (ITM) options ($P(0,S) < K$) per maturity. m=500 and σ^* ranges from 0.01 to 0.08. Number of iterations: number of loops in Algorithm 1.

6 Conclusions

This work extends the framework of constrained Gaussian process regression (CGPR) of Hainaut and Vrins (2024) to the valuation of American options. The variational PDE governing option prices in the absence of arbitrage is approximated by a penalized Feynman-Kac (PFK) equation. The differential operator in this equation being non-linear, we propose an iterative algorithm using a local linear operator. At each epoch, we fit a CGPR approximating the option price.

Our method inherits the advantages of the CGPR with respect to other numerical approaches. It only relies on the sampling of risk processes in the inner and boundary domains of the PDE and does not require numerical differentiation as finite difference methods do. Along with a Bayesian framework, which adds a Tikhonov regularization factor to the numerical scheme, this ensures good accuracy and stability in option pricing.

The numerical illustrations focus on the pricing of American put options in the Heston and two-factor Hull-White models. The comparison with the least squares Monte Carlo method reveals that the CGPR achieves comparable accuracy and is much less computationally intensive.

Nevertheless, the CGPR for American options presents a few drawbacks. It is based on kernels whose hyperparameters may be difficult to tune. The Tikhonov regularization parameter, which stabilizes the numerical framework, introduces a small bias. Finally, when the number of risk factors is high, the CGPR needs large training datasets, which complicates the inversion of the Gram matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$.

Acknowledgement

Donatien Hainaut thanks the FNRS (Fonds de la recherche scientifique) for the financial support through the EOS project Asterisk (research grant 40007517).

Appendix A

In the second numerical illustration, we model the initial yield curve with the Nelson-Siegel (NS) model. In this framework, initial instantaneous forward rates are provided by the following function:

$$f(0,t) := -\partial_t \ln P(0,t) \quad = \quad b_0 + (b_{10} + b_{11} t) \exp\left(-c_1 t\right) \, .$$

Parameters $\{b_0, b_{10}, b_{11}, c_1\}$ are estimated by minimizing the quadratic spread between market and model zero-coupon yields:

$$P(0,t) \quad = \quad \exp\left(b_0 + \frac{1}{t}\frac{b_{10}}{c_1}\left(1 - e^{-c_1 t}\right) + \frac{1}{t}\frac{b_{11}}{(c_1)^2}\left(1 - (c_1 t + 1)\, e^{-c_1 t}\right)\right) \, .$$

We fit the NS model to the yield curve of Belgian state bonds observed on the 1/9/2023 and obtain estimates reported in Table 15.

Parameter	Value
b_0	0.040535
b_{10}	0.003652
b_{11}	-0.022349
c_1	0.449268

Table 15: Nelson-Siegel parameters, Belgian state bonds, 1/9/23.

References

[1] Al-Aradi A., Correia A., Jardim G., de Freitas Naiff D., Saporito Y. 2022. Extensions of the deep Galerkin method. App. Math. and Computation, 430, 127287.

[2] Andersson K., Oosterlee C.W. 2021. Deep learning for CVA computations of large portfolios of financial derivatives. App. Math. and Computation, 409, 126399

[3] Beckers T. 2021. An introduction to Gaussian process models. arXiv:2102.05497v1.

[4] Brigo D., Mercurio F. 2006. Interest rate models :theory and practice. Second edition. Springer-Verlag, Berlin-Heidelberg.

[5] Calderhead B. , Girolami M., Lawrence N.D. 2009. Accelerating Bayesian inference over nonlinear differential equations with Gaussian processes, in Advances in Neural Information Processing Systems, pp 217-224.

[6] Crépey S. Dixon M. 2019. Gaussian Process Regression for Derivative Portfolio Modeling and Application to CVA Computations. arXiv:1901.11081

[7] Forsyth P.A., Vetzal K.R. 2002. Quadratic Convergence for Valuing American Options Using a Penalty Method. SIAM Journal on scientific computing, 23 (6), pp 2095–2122.

[8] Glau K., Wunderlich L. 2022. The deep parametric PDE method and applications to option pricing. App. Math. and Computation, 432, 127355

[9] Goudenège, L., Molent, A., Zanette, A., 2021. Gaussian process regression for pricing variable annuities with stochastic volatility and interest rate. Decisions in Economics and Finance, 44, pp 57–72.

[10] Graepel T. 2003. Solving Noisy Linear Operator Equations by Gaussian Processes: Application to Ordinary and Partial Differential Equations. ICML'03: Proceedings of the Twentieth International Conference on International Conference on Machine Learning Pages 234 - 241

[11] Hainaut D. 2023. Continuous time processes for finance: switching, self-exciting, fractional and other recent dynamics. Springer & Bocconi series in mathematics, statistics, finance and economics. Springer nature Switzerland.

[12] Hainaut D., Casas A., 2024.Option pricing in the Heston model with physics inspired neural networks. Annals of finance. https://doi.org/10.1007/s10436-024-00452-7

[13] Hainaut D. 2024. Valuation of guaranteed minimum accumulation benefits (GMABs) with physics-inspired neural networks. Annals of actuarial sciences. https://doi.org/10.1017/S1748499524000095

[14] Hainaut D., Vrins F., 2024. European option pricing with model constrained Gaussian process regressions. UCLouvain ISBA discussion paper. https://dial.uclouvain.be/pr/boreal/object/boreal:292395

[15] Han J., Zhang X.P., Wang F., 2016. Gaussian process regression stochastic volatility model for financial time series, in IEEE Journal of Selected Topics in Signal Processing, 10 (6), pp 1015-1028.

[16] Hull J., White A., 1994. Numerical Procedure for Implementing Term Structure Models II: Two Factor Models. Journal of Derivatives, 2, pp 37-47.

[17] Longstaff F.A., Schwartz E.S., 2001. Valuing American options by simulation: a simple least squares approach. Review of Financial Studies, 14, pp 113–148.

[18] Madi S., Cherif Bouras M., Haiour M, Stahel A., 2018. Pricing of American options, using the Brennan–Schwartz algorithm based on finite elements. App. Math. and Computation, 339, 846-852.

[19] Nguyen N.C., Peraire J. 2015. Gaussian functional regression for linear partial differential equations. Computational methods in applied mechanics and engineering, 287, pp 69-89.

[20] Petelin D., Sindelar J., Prikryl J., Kocijan J., 2011. Financial modeling using Gaussian process models. Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems, Prague, Czech Republic, 2011, pp 672-677.

[21] Pförtner, M., Steinwart I., Hennig P., Wenger J. 2022. Physics-Informed Gaussian Process Regression Generalizes Linear PDE Solvers. arXiv:2212.12474

[22] Raissi M., Perdikaris P., Em Karniadakis E. 2017. Machine learning of linear differential equations using Gaussian processes. Journal of Computational Physics 348, pp 683-693.

[23] Rasmussen C.E., Williams C.K.I. 2006. Gaussian processes for machine learning, MIT press.

[24] Seydel R. 2017. Tools for computational finance. Sixth edition. Springer textbook. Springer-Verlag Berlin Heidelberg New-York.

[25] Sirignano J., Spiliopoulos K., 2018. DGM: a deep learning algorithm for solving partial differential equations, Journal of Computational Physics, 375, 1339–1364.

[26] Swiler L., Gulian M., Frankel A., Safta C., Jakeman J. 2020. A survey of constrained Gaussian process regression: approaches and implementation challenges. Journal of Machine Learning for Modeling and Computing 1(2) pp 119-156.

[27] Wu Y., HernÃ¡ndez-Lobato J.M., Ghahramani Z. 2014. Gaussian process volatility model. Advances in Neural Information Processing Systems 27, proceedings NIPS 2014.

[28] Zvan R., Forsyth P.A., Vetzal K.R. 1998. Penalty methods for American options with stochastic volatility. Journal of Comp. and App. Math., 91, pp 199-218