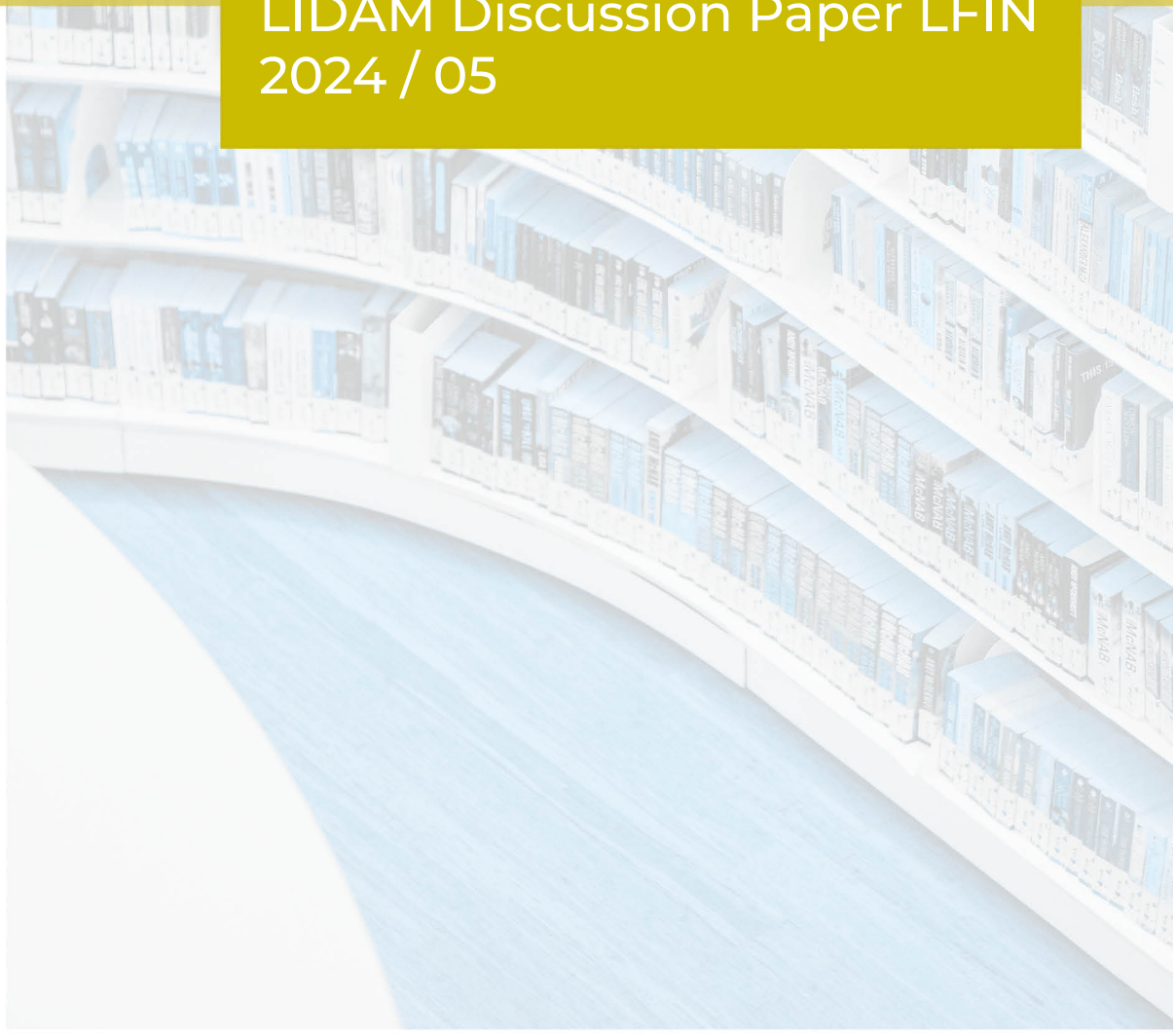
EUROPEAN OPTION PRICING WITH MODEL CONSTRAINED GAUSSIAN PROCESS REGRESSIONS

Donatien Hainaut, Frédéric Vrins

LIDAM Discussion Paper LFIN
2024 / 05



UCLouvain

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

LFIN

Voie du Roman Pays 34, L1.03.01 B-1348 Louvain-la-Neuve

Tel (32 10) 47 43 04

Email: lidam-library@uclouvain.be

https://uclouvain.be/en/research-institutes/lidam/lfin/publications.html

European option pricing with model constrained Gaussian process regressions

Donatien Hainaut*
UCLouvain, ISBA-LIDAM
Frédéric Vrins†
UCLouvain, LFIN-LIDAM

We propose a method for pricing European options based on Gaussian processes. We convert the problem of solving the Feynman-Kac (FK) partial differential equation (PDE) into a model-constrained regression. We form two training sets by sampling state variables from the PDEs inner domain and boundary. The regression function is then estimated to fit the option values on the boundary sample while satisfying the FK PDE on the inner sample. 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 closed-form expressions of approximated option prices and their Greeks. We demonstrate the performance of the procedure on call options in the Black-Scholes and Heston models.

KEYWORDS: Gaussian process regression, Option pricing, Feynman-Kac equation, partial differential equation, Heston model, machine learning

1 Introduction

We address the problem of computing the price of European-style derivative securities in an arbitrage-free market. In such markets, the prices of non-dividend paying assets (including derivative securities up to their expiry) are governed by a partial differential equation (PDE) subject to a terminal condition. In this context, the price of the asset is given by a function f evaluated at the prevailing state variables, $\boldsymbol{x}$. The PDE governing f is known as the Feynman-Kac (FK) equation (see e.g. Jeanblanc et al. 2009, section 2.2.3) and takes the form $\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = z(\boldsymbol{x})$ where $\mathcal{L}_{\boldsymbol{x}}$ is a linear differential operator. The boundary constraint reads as $f(\bar{\boldsymbol{x}}) = h(\bar{\boldsymbol{x}})$ where h is the value of derivative and $\bar{\boldsymbol{x}}$ is on the PDE boundary. The former captures the dynamics of the market model at hand while the latter uniquely characterizes the considered option. As an illustration, the time-t price of a European call with strike K and expiry $T \geq t$ written on a stock whose price process $S = (S_t)_{t\geq 0}$ follows a geometric Brownian motion with volatility σ and assuming a constant risk-free rate leads to $V_t = f(\boldsymbol{x})$ with $\boldsymbol{x} = (t, S_t)$ for $t < T$, $z(\boldsymbol{x}) = 0$,

$$\mathcal{L}_{\boldsymbol{x}} f(t,x) = \frac{\partial}{\partial t} f(t,x) + rx\frac{\partial}{\partial x} f(t,x) + \frac{\sigma^2 x^2}{2}\frac{\partial^2}{\partial x^2} f(t,x) - rf(t,x).$$

*Corresponding author. Postal address: Voie du Roman Pays 20, 1348 Louvain-la-Neuve (Belgium). E-mail to: donatien.hainaut(at)uclouvain.be
†E-mail to: frederic.vrins@uclouvain.be

The option values on the boundary are respectively null for $\bar{\boldsymbol{x}} = (t, 0)$ and equal to $(S_T - K)_+$ for $\bar{\boldsymbol{x}} = (T, S_T)$. Except in some special cases like the Black-Scholes model, it is not possible to solve this equation analytically. Computing the price of simple European derivative securities in a stochastic volatility model, for instance, requires numerical methods such as lattice, Laplace transform or Monte Carlo methods. Although efficient algorithms have been designed for this purpose, the most efficient ones are model-specific (e.g., they require the knowledge of the characteristic function). Moreover, they can suffer from convergence or stability issues, and often provide disappointing estimations of the sensitivities (Greeks). In this article, we propose an efficient method to tackle this problem in its general form by leveraging Constrained Gaussian Process Regression (CGPR).

In this paper we propose to solve the FK equation by considering a constrained regression model. To this end, we assume to have two samples of points, $\mathring{\boldsymbol{X}}$ and $\bar{\boldsymbol{X}}$, respectively in the inner domain of the PDE and on the boundary. We seek for a regression function $\hat{f}(\boldsymbol{x})$, approximating f subject to

$$\mathcal{L}_{\boldsymbol{x}} \hat{f}(\boldsymbol{x}) = z(\boldsymbol{x}) + \epsilon \quad \forall \boldsymbol{x} \in \mathring{\boldsymbol{X}} \quad \text{and} \quad \hat{f}(\bar{\boldsymbol{x}}) = h(\bar{\boldsymbol{x}}) + \bar{\epsilon} \quad \forall \bar{\boldsymbol{x}} \in \bar{\boldsymbol{X}}$$

for some independent Gaussian noises $\epsilon, \bar{\epsilon}$. In a constrained Gaussian process regression, we are looking for the random functions $\widehat{f}$ taking the form of a Gaussian process. One can thus take the expectation of $\widehat{f}$ as an approximation of the true price function, f.

The unconstrained Gaussian process regression (GPR) is a powerful tool for regression that has been successfully applied in many different fields. We refer the reader to Rasmussen and Williams (2006, Chapter 6) and to (2012, Chapter 15) for a general presentation. Gaussian processes (GPs) are also used to regress responses to explanatory variables constrained to satisfy a partial differential equation (PDE). Similarly to Physics-Inspired Neural Networks (PINNs), this PDE represents the physical law governing the experiment for which responses are measured. There is a vast literature on this topic, and we refer to Swiler et al. (2020) for a survey of CGPRs, including constraints provided by linear PDEs and boundary conditions. Among the useful references therein, we highlight Graepel (2003), who approximates the solution of PDEs with noisy responses using a CGPR. Calderhead et al. (2009) present an accelerated sampling procedure to solve nonlinear ordinary and delay differential equations with Gaussian processes. Cialenco et al. (2012) combine an Euler discretization scheme on the time axis with a CGPR. Dondelinger (2013), Barber and Wang (2014), and Cockayne et al. (2019) propose a CGPR that improves the gradient matching technique. Särkkä (2011) and Nguyen and Peraire (2015) develop procedures for handling responses of the PDE solution, given in the form of linear functionals of the state variables. Raissi et al. (2017) use an approach based on GPs to discover governing equations expressed by parametric linear operators. Pförtner et al. (2022) consider physics-informed Gaussian process regression for solving both weak and strong formulations of linear PDEs. Chen et al. (2021) propose a method to solve nonlinear PDEs based on the maximum a posteriori estimator of a Gaussian process.

When it comes to finance, GPR is mainly used for time-series forecasting. Wu et al. (2014) introduce a non-parametric model for time-changing variances based on Gaussian processes. Han et al. (2016) combine a Gaussian process framework with a stochastic volatility (SV) model for stock prices. Petelin et al. (2019) fit a GPR to daily data from U.S. commodity markets. Gonzalvez et al. (2019) present a method using GPR for fitting the U.S. yield curve and portfolio allocation. GPs are also used for the regression of risk measures or derivatives, computed with a model, on their parameters to speed up the valuation procedure. For instance, Crépey and Dixon (2019) use GPR for derivative portfolio modeling and computing credit valuation adjustments. Wilkens (2019) proposes an approach based on GPR for forecasting risk management indicators. Goudenège et al. (2021) fit a GP to regress prices of variable annuities on market risk factors. Closer to our work is De Spiegeleer et al. (2018), who use GPR to

predict option prices valued with trees or fast Fourier transforms from market parameters. Our approach is different, however, because we exploit the FK PDE. In particular, De Spiegeleer et al. (2018) build their regression on a sample of prices produced via other pricing techniques (such as FFT or Monte Carlo). By contrast, we only need to sample the state variables in the inner domain ($\mathring{\boldsymbol{X}}$) and on the boundary ($\bar{\boldsymbol{X}}$), ensuring that the regression function complies with the payoff and the PDE at those points, respectively.

This work contributes to the financial literature in several ways. To our knowledge, this article is among the first to propose a method for pricing derivatives by solving the FK equation with a CGPR. Variants of the CGPR have been successfully applied in mechanics and engineering (see references above) but have not yet been applied to option valuation. Physics-Inspired Neural Networks (PINNs) also aim to solve PDE-constrained regression and have been considered for option pricing (see Sirignano and Spiliopoulos, 2018; Al-Aradi et al., 2022; Hainaut, 2024 a; and Hainaut and Casas, 2024). However, PINNs rely on neural networks instead of GPs. Second, by taking advantage of the FK equation, we do not rely on numerical differentiation, in contrast to PINN-based methods. Third, compared to classical finite difference approaches, GPR naturally justifies adding a Tikhonov regularization factor, which helps tune the robustness of numerical computations. Fourth, we improve upon the approach of Graepel (2003), which is designed for solving PDEs with a nullity constraint on the boundary. For other constraints, Graepel (2003) assumes that the solution is the sum of an a priori chosen function, which fulfills the terminal condition, and a GP that solves a PDE with a nullity constraint. Choosing the function that satisfies the boundary constraints a priori is the critical step of this method. In contrast, we sample state variables on the boundary as well and consider a CGPR model that fits the boundary condition within this sample. Finally, we provide fully analytical expressions for CGPR option prices and their Greeks using a Gaussian kernel.

The outline of the paper is as follows. Section 2 presents a general market model in a Brownian setting and a general version of the FK equation. The next section reviews the classic framework of GPR and introduces the PDE-constrained extension (CGPR). Section 4 provides detailed developments for solving this FK equation using a Bayesian approach. It also presents a general analytical expressions of approximated prices and Greeks when the kernel is Gaussian. Sections 5 and 6 test the efficiency of our method in two particular cases. First, we evaluate the capacity of the CGPR, for pricing a call option in a Black and Scholes market. Next, we price European call options in the Heston stochastic volatility model, using the Monte-Carlo method as benchmark.

2 The Feynman-Kac equation

We consider a financial market driven by $p-1$ risk processes, stored in a vector $(\boldsymbol{y}_t)_{t\geq 0}$. These processes are defined on a complete probability space endowed with a filtration $\mathbb{F} = (\mathcal{F}_t)_{t\geq 0}$ and a risk neutral probability measure, $\mathbb{Q}$. The risk-free rate is function of state variables and denoted by $r_t = r(t, \boldsymbol{y}_t)$. Asset prices and risk factors (e.g. the stochastic variance) are ruled by the following stochastic differential equation (SDE)[1]:

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

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

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

[1]Note that if y is a tradeable asset, the corresponding drift takes the form $\mu_y(t, \boldsymbol{y}_t) = r_t y_t$.

where $\mathbb{E}$ is the expectation operator under $\mathbb{Q}$. We consider a European derivative security expiring at time T with payoff $\mathcal{P}(T, \boldsymbol{y}_T)$. We assume that its price at any time $t \leq T$ is a function of the current value of the state variables, i.e., $f(t, \boldsymbol{y}_t)$ where f is referred to as the *pricing function,* $f(t, \boldsymbol{y}_t)$ of the derivative. Under the assumption of arbitrage-free market, all traded securities paying no cash-flow are $(\mathbb{Q}, \mathbb{F})$- martingales. Hence, the pricing function satisfies

$$\mathbb{E}\left(e^{-\int_s^t r(u,\boldsymbol{y}_u)du} f(t, \boldsymbol{y}_t) \mid \mathcal{F}_s\right) = f(s, \boldsymbol{y}_s) \quad \forall s \leq t \leq T \tag{2}$$

subject to the terminal boundary condition determined by the payoff

$$f(T, \boldsymbol{y}_T) \quad = \quad \mathcal{P}(T, \boldsymbol{y}_T) \quad \forall \boldsymbol{y}_T \,. \tag{3}$$

An immediate consequence of (2) is that the expected rate of return of the derivative under $\mathbb{Q}$ is the risk-free rate:

$$\mathbb{E}\left(df(t, \boldsymbol{y}_t) \mid \mathcal{F}_t\right) = r(t, \boldsymbol{y}_t) f(t, \boldsymbol{y}_t) \; dt \,. \tag{4}$$

Computing the left-hand side of (4) where $df(t, \boldsymbol{y}_t)$ is found using Itô's lemma yields the Feynman-Kac PDE:

$$\partial_t f + \boldsymbol{\mu}_{\boldsymbol{y}}(t, \boldsymbol{y}_t)^\top \nabla_{\boldsymbol{y}} f + \frac{1}{2}\mathrm{tr}\left(\Sigma_{\boldsymbol{y}}(t, \boldsymbol{y}_t)\Sigma_{\boldsymbol{y}}(t, \boldsymbol{y}_t)^\top \mathcal{H}_{\boldsymbol{y}} f\right) = r(t, \boldsymbol{y}_t) f \,, \tag{5}$$

where $\nabla_{\boldsymbol{y}} f$ is the gradient of f with respect to $\boldsymbol{y}$, $\mathcal{H}_{\boldsymbol{y}} f$ is the Hessian and tr(.) is the trace operator. To lighten further developments, we note $\boldsymbol{x} = (t, \boldsymbol{y}_t)$, the $p-$vector of time and state variables at time t and define the linear[2] differential operator associated with the FK equation:

$$\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = \partial_t f - r(\boldsymbol{x}) f + \boldsymbol{\mu}_{\boldsymbol{y}}(\boldsymbol{x})^\top \nabla_{\boldsymbol{y}} f + \frac{1}{2}\mathrm{tr}\left(\Sigma_{\boldsymbol{y}}(\boldsymbol{x})\Sigma_{\boldsymbol{y}}(\boldsymbol{x})^\top \mathcal{H}_{\boldsymbol{y}} f\right) \,. \tag{6}$$

We denote by $\mathcal{X} = [0, T] \times \mathbb{R}_0^{p-1}$ the domain on which the pricing function f is defined. The boundary is $\bar{\mathcal{X}} = \bar{\mathcal{X}}_b \cup \bar{\mathcal{X}}_T$ where

$$\bar{\mathcal{X}}_T = \{(T, s) \mid s \in \mathbb{R}^+\}$$

is the set of $\bar{\boldsymbol{x}}$ such the option is equal to the payoff $f(\bar{\boldsymbol{x}}) = \mathcal{P}(\bar{\boldsymbol{x}})$. Whereas $\bar{\mathcal{X}}_b$ is the set of $\bar{\boldsymbol{x}}$ such that $f(\bar{\boldsymbol{x}}) = b(\bar{\boldsymbol{x}})$ where be is a known function (e.g. $b(\bar{\boldsymbol{x}}) = 0$). $\mathring{\mathcal{X}}$, that we call the interior domain, is such that $\mathcal{X} = \mathring{\mathcal{X}} \cup \bar{\mathcal{X}}$. We infer that the pricing function f of any European derivative with a payoff function h, solves

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = 0 & \boldsymbol{x} \in \mathring{\mathcal{X}} \,, \\ f(\bar{\boldsymbol{x}}) = h(\bar{\boldsymbol{x}}) & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}} \,, \end{cases} \tag{7}$$

where $h(\bar{\boldsymbol{x}})$ is the derivative value function on $\bar{\mathcal{X}}$:

$$h(\bar{\boldsymbol{x}}) \quad = \quad \begin{cases} b(\bar{\boldsymbol{x}}) & \boldsymbol{x} \in \bar{\mathcal{X}}_b \\ \mathcal{P}(\bar{\boldsymbol{x}}) & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}}_T \end{cases} .$$

In later developments, $h(\bar{\boldsymbol{x}})$ is called the boundary function.

3 The constrained Gaussian process regression (CGPR)

Solving the PDE (7) can be viewed as a regression problem constrained by a PDE, with its structure depending on the stochastic model chosen for the market. In this work, we propose an approach based on Gaussian process regression. We will briefly review the main features of this type of regression before introducing the PDE-constrained extension.

[2]By linear operator, we mean that $\mathcal{L}_x(g^{(1)}(\boldsymbol{x}) + g^{(2)}(\boldsymbol{x})) = \mathcal{L}_x g^{(1)}(\boldsymbol{x}) + \mathcal{L}_x g^{(2)}(\boldsymbol{x})$.

3.1 The Gaussian process regression (GPR)

A Gaussian process is a collection noted $\{g(\boldsymbol{x})\,,\ \boldsymbol{x} \in \mathcal{X}\}$ such that for any $d \in \mathbb{N}$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d \in \mathcal{X}$, the random vector $(g(\boldsymbol{x}_1), \ldots, g(\boldsymbol{x}_d))$ has a joint multivariate Gaussian distribution, characterized by a mean and a covariance function. Without loss of generality, the mean is set to zero and the covariance function is defined by a kernel $k(\boldsymbol{x}, \boldsymbol{x}')$:

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

The value of the kernel $k(\boldsymbol{x}, \boldsymbol{x}^{'})$ controls the interaction between the states $(\boldsymbol{x}, \boldsymbol{x}^{'})$. A necessary and sufficient condition for the function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ to be a valid kernel is that the $d \times d$ matrix of $(k(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1,\ldots,d}$ is positive semidefinite for all possible samples, i.e., $\sum_{i=1}^{d} k(\boldsymbol{x}_i, \boldsymbol{x}_j) c_i c_j \geq 0$ holds for any $d \in \mathbb{N}$, $(\boldsymbol{x}_i)_{i=1,\ldots,d} \in \mathcal{X}$ and $c_1, ..., c_d \in \mathbb{R}$.

The general principle of GPR is to provide an approximation of a function $h(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}$ whose functional form is unknown but whose values are observed at certain points. The training set contains covariates, $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1,\ldots,d}$, randomly sampled in $\mathcal{X}$, and noised responses $\boldsymbol{h} = (h_i)_{i=1,\ldots,d}$ with $h_i = h(\boldsymbol{x}_i) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma_*^2)$. From a Bayesian perspective, we encode our belief that instances of $h(\boldsymbol{x})$ are drawn from a Gaussian process g with zero mean and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$, prior to taking into account observations. The covariance matrix of g on the training set is given by the kernel matrix

$$k(\boldsymbol{X}, \boldsymbol{X}) = (k(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1,\ldots,d}\,.$$

Our regression problem thus consists in finding a Gaussian process g such that

$$g(\boldsymbol{x}_i) = h_i \quad \forall \boldsymbol{x}_i \in \boldsymbol{X}. \tag{8}$$

In the framework of the GPR, $\boldsymbol{h}$ is the realization of d-dimensional normal denoted by $\boldsymbol{H}$, which is the Gaussian process at points $(\boldsymbol{x}_i)_{i=1,\ldots,d}$, noised by $\epsilon \sim \mathcal{N}(0, \sigma_*^2)$. Therefore, the a priori joint-distribution of $g(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X} \setminus \boldsymbol{X}$ and $\boldsymbol{H}$ is :

$$\begin{pmatrix} g(\boldsymbol{x}) \\ \boldsymbol{H} \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{0}\,; \begin{pmatrix} k(\boldsymbol{x}, \boldsymbol{x}) & k(\boldsymbol{X}, \boldsymbol{x}) \\ k(\boldsymbol{X}, \boldsymbol{x})^{\top} & \sigma_*^2 I_d + k(\boldsymbol{X}, \boldsymbol{X}) \end{pmatrix}\right), \tag{9}$$

where $k(\boldsymbol{X}, \boldsymbol{x}) = (k(\boldsymbol{x}_i, \boldsymbol{x}))_{i=1,\ldots,d}$. Equation (9) should be regarded as the prior of $(g(\boldsymbol{x}), H)$. We consider as estimator $\widehat{h}(\boldsymbol{x})$ of $h(\boldsymbol{x})$ the conditional expectation of g given $\boldsymbol{H} = \boldsymbol{h}$, which is found thanks to the posterior density of $g(\boldsymbol{x}) \mid \boldsymbol{h}$:

$$\begin{aligned} \widehat{h}(\boldsymbol{x}) &= \mathbb{E}\left(g(\boldsymbol{x}) \mid \boldsymbol{h}\right) \\ &= k(\boldsymbol{x}, \boldsymbol{X})^{\top} \left(\sigma_*^2 I_d + k(\boldsymbol{X}, \boldsymbol{X})\right)^{-1} \boldsymbol{h}\,. \end{aligned}$$

Note that assuming noisy observations of $h(\boldsymbol{x}_i)$ for $\boldsymbol{x}_i \in \boldsymbol{X}$ provides regularization via a linear shrinkage of the covariance matrix $k(\boldsymbol{X}, \boldsymbol{X})$, analogous to Ridge estimators. We also infer from (9) that the conditional variance of the estimator does not depend on $\boldsymbol{h}$, and is equal to:

$$\mathbb{V}\left(g(\boldsymbol{x}) \mid \boldsymbol{h}\right) = k(\boldsymbol{x}, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{X})^{\top} \left(\sigma_*^2 I_d + k(\boldsymbol{X}, \boldsymbol{X})\right)^{-1} k(\boldsymbol{X}, \boldsymbol{x})\,.$$

3.2 GPR constrained by the FK PDE

Applied to derivatives valuation, $g(\boldsymbol{x})$ can be estimated by performing a GPR on a sample of state variables $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1,\ldots,d}$ (e.g., current time, spot price and volatility) using the corresponding prices $\{h_i\}_{i=1,\ldots,d}$ as targets. This approach first requires generating the sample of prices used for training, obtained through other pricing techniques (such as trees, lattice,

Laplace transform or Monte Carlo) to estimate the target values h_i. This is the route taken by De Spiegeleer et al. (2018). Under model (1), this appears to be sub-optimal for two reasons. First, the price function is completely determined by the FK PDE combined with the terminal condition, meaning there is no need to rely on a training set of prices produced by alternative methods. Second, such a regression exercise fails to leverage (2), which is a crucial functional constraint for driving arbitrage-free prices. For these reasons, we adopt a different approach that consists of solving a GPR constrained by the FK PDE (the CGPR), without relying on a set of exogenous prices.

We consider two training sets for the CGPR. The first set contains d combinations of risk factors on the boundary, denoted as $\bar{\boldsymbol{X}} = \{\bar{\boldsymbol{x}}_i\}_{i=1,\dots,d}$, along with the corresponding noised values, $\boldsymbol{h} = (h_i)_{i=1,\dots,d}$. The $\bar{\boldsymbol{x}}_i$'s are randomly sampled from $\bar{\mathcal{X}}$ while $h_i = h(\bar{\boldsymbol{x}}_i) + \bar{\epsilon}$ are observations of the boundary function (e.g. payoffs), noised by $\bar{\epsilon} \sim \mathcal{N}(0, \sigma_*^2)$. The second training set consists of m combinations of risk factors 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 FK PDE are stored in a m-vector, $\boldsymbol{z} = (z_i)_{i=1,\dots,m}$ where $z_i \sim \mathcal{N}(0, \sigma_*^2)$. We seek a Gaussian process g approximating the price function f, and satisfying the following equalities on the two training sets:

$$\begin{cases} \mathcal{L}_{\boldsymbol{x}} g(\mathring{\boldsymbol{x}}_i) &= z_i \quad \forall \mathring{\boldsymbol{x}}_i \in \mathring{\boldsymbol{X}} \ , \quad i = 1, 2, \dots, m \ , \\ g(\bar{\boldsymbol{x}}_i) &= h_i \quad \forall \bar{\boldsymbol{x}}_i \in \bar{\boldsymbol{X}} \ , \quad i = 1, 2, \dots, d \ . \end{cases} \tag{10}$$

Let us make two important points here. Firstly, the price function at expiry coincides with the (noisy) payoff, meaning that $h_i = f(\bar{\boldsymbol{x}}_i) + \bar{\epsilon}$ for $\bar{\boldsymbol{x}}_i \in \bar{\mathcal{X}}_T$. If we ignore the first PDE constraint in Equation (10), we recognize the GPR framework and Equation (8), in which we approximate the price function on the boundary by a Gaussian process. Secondly, the training sets do not contain any market or computed prices. Indeed, $\boldsymbol{h}$ is a vector of noisy boundary values and $\boldsymbol{z}$ is a vector of Gaussian noises. It is important to note that the noise serves to provide regularization.

In the sequel, we denote by $(\boldsymbol{z}, \boldsymbol{h})$ a realization of $(\boldsymbol{Z}, \boldsymbol{H})$, two random d- and m-vectors with a Gaussian prior. We show in the next section that the joint distribution of $(g(\boldsymbol{x}), \boldsymbol{Z}, \boldsymbol{H})$ for $\boldsymbol{x} \notin \mathring{\boldsymbol{X}}$ follows a multivariate normal, similar to $(g(\boldsymbol{x}), \boldsymbol{H})$, as presented in Equation (9).

4 Estimation of the CGPR

We encode a-priori our belief that instances of $g(\boldsymbol{x})$, satisfying Equation (10), are drawn from a Gaussian process with zero mean and covariance or kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$, prior to taking into account observations $\boldsymbol{h}$ and $\boldsymbol{z}$ of the boundary and FK PDE. From a Bayesian perspective, the estimator of $f(\boldsymbol{x})$ is the *a posteriori* expectation of $g(\boldsymbol{x})$ conditional upon $(\boldsymbol{Z}, \boldsymbol{H}) = (\boldsymbol{z}, \boldsymbol{h})$, which we note $\widehat{f}(\boldsymbol{x}) = \mathbb{E}\left(g\left(\boldsymbol{x}\right) \mid \boldsymbol{z}\ , \boldsymbol{h}\right)$. From a mathematical point of view, the function $g(\boldsymbol{x})$ resides in a reproducing kernel Hilbert space spanned by eigenfunctions of the covariance function (see e.g. Chapter 6 of Rasmussen and Williams, 2006).

4.1 Estimator and properties

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. Because we consider a Gaussian prior with null mean and covariance matrix

$\mathbb{C}\left(g(\boldsymbol{x}), g(\boldsymbol{x}')\right) = k(\boldsymbol{x}, \boldsymbol{x}')$ for $g(\boldsymbol{x})$, we rewrite $g(\boldsymbol{x})$ as a weighted sum of functions $\varphi_j(.)$:

$$g(\boldsymbol{x}) = \sum_{j=1}^{n} w_j \varphi_j(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\varphi}(\boldsymbol{x}) \,, \tag{11}$$

where the weight vector is $\boldsymbol{w} \sim \mathcal{N}(0, I_n)$. We define the function $\boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x}) = \left(\mathcal{L}_{\boldsymbol{x}} \varphi_j(\boldsymbol{x})\right)_{j=1,\ldots,n}$ from $\mathcal{X} \to \mathbb{R}^n$. Using the linearity of the differential operator (6), $\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x})$ is the scalar product:

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

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 $(\boldsymbol{Z}, \boldsymbol{H})$, the random vectors of right-hand terms in Equation (10) is a multivariate normal:

$$\left(\begin{array}{c} \boldsymbol{Z} \\ \boldsymbol{H} \end{array} \right) \Bigg| \, \boldsymbol{w} \sim \mathcal{N}\left(\left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w} \\ \bar{\boldsymbol{\varphi}} \boldsymbol{w} \end{array} \right), \left(\begin{array}{cc} \sigma_*^2 I_m & \boldsymbol{0} \\ \boldsymbol{0} & \sigma_*^2 I_d \end{array} \right) \right) . \tag{12}$$

The next proposition provides the posterior distribution of weights.

Proposition 1. *The conditional distribution of regression weights in Equation (11) is:*

$$\boldsymbol{W} | \boldsymbol{z}, \boldsymbol{h} \;\; \sim \;\; \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}} \,, \Sigma_{\boldsymbol{z},\boldsymbol{h}}\right)$$

where the mean $\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}$ *and covariance* $\Sigma_{\boldsymbol{z},\boldsymbol{h}}$ *are equal to*

$$\begin{cases} \Sigma_{\boldsymbol{z},\boldsymbol{h}} & = \left(I_n + \sigma_*^{-2} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} + \sigma_*^{-2} \bar{\boldsymbol{\varphi}}^\top \bar{\boldsymbol{\varphi}} \right)^{-1} \,, \\ \boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}} & = \sigma_*^{-2} \Sigma_{\boldsymbol{z},\boldsymbol{h}} \left(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top \boldsymbol{z} + \bar{\boldsymbol{\varphi}}^\top \boldsymbol{h} \right) . \end{cases} \tag{13}$$

Proof. We infer the posterior density of $\boldsymbol{W}$ from the Bayes theorem:

$$f_{\boldsymbol{W}|\boldsymbol{z},\boldsymbol{h}}(\boldsymbol{w}) \;\; = \;\; \frac{f_{\boldsymbol{Z},\boldsymbol{H}|\boldsymbol{w}}(\boldsymbol{z},\boldsymbol{h}) f_{\boldsymbol{W}}(\boldsymbol{w})}{f_{\boldsymbol{Z},\boldsymbol{H}}(\boldsymbol{z},\boldsymbol{h})} \,,$$

where $f_{\boldsymbol{W}}(.)$ is the standard normal density of dimension n,

$$f_{\boldsymbol{W}}(\boldsymbol{w}) \;\; = \;\; \frac{1}{(2\pi)^{n/2}} \exp\left(-\frac{1}{2} \boldsymbol{w}^\top \boldsymbol{w} \right) ,$$

and $f_{\boldsymbol{Z},\boldsymbol{H}|\boldsymbol{w}}(.)$ is the multivariate normal density (12):

$$\begin{aligned} f_{\boldsymbol{Z},\boldsymbol{H}|\boldsymbol{w}}(\boldsymbol{z},\boldsymbol{h}) = & \frac{1}{(2\pi)^{(m+d)/2} \, \sigma_*^{m+d}} \\ & \times \exp\left(-\frac{1}{2\sigma_*^2} (\boldsymbol{z} - \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w})^\top (\boldsymbol{z} - \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w}) \right) \\ & \times \exp\left(-\frac{1}{2\sigma_*^2} (\boldsymbol{h} - \bar{\boldsymbol{\varphi}} \boldsymbol{w})^\top (\boldsymbol{h} - \bar{\boldsymbol{\varphi}} \boldsymbol{w}) \right) . \end{aligned}$$

Ignoring terms not depending upon $\boldsymbol{w}$, we infer that $f_{\boldsymbol{W}|\boldsymbol{z},\boldsymbol{h}}(\boldsymbol{w})$ is proportional to :

$$\begin{aligned} f_{\boldsymbol{W}|\boldsymbol{z},\boldsymbol{h}}(\boldsymbol{w}) \;\; \propto \;\; & \exp\left(-\frac{1}{2} \boldsymbol{w}^\top \boldsymbol{w} - \frac{1}{2\sigma_*^2} (\boldsymbol{z} - \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w})^\top (\boldsymbol{z} - \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{w}) \right) \\ & \times \exp\left(-\frac{1}{2\sigma_*^2} (\boldsymbol{h} - \bar{\boldsymbol{\varphi}} \boldsymbol{w})^\top (\boldsymbol{h} - \bar{\boldsymbol{\varphi}} \boldsymbol{w}) \right) . \end{aligned}$$

To prove that this expression complies with the claim, we show that the quadratic and linear terms in the above exponential (scaled by $-1/2$) agree with those of

$$\left(\boldsymbol{w}-\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}\right)^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\left(\boldsymbol{w}-\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}\right)=\boldsymbol{w}^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{w}-\boldsymbol{w}^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}-\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{w}+\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}\ .$$

This is indeed the case because the quadratic term in the above equation is equal to

$$\begin{aligned}
&\boldsymbol{w}^{\top}\left(I_n+\frac{1}{\sigma_*^2}\left(k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right)+k\left(\bar{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)\right)\right)\boldsymbol{w}\\
&\quad=\boldsymbol{w}^{\top}\boldsymbol{w}+\frac{1}{\sigma_*^2}\left(\boldsymbol{w}^{\top}(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^{\top}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}})\boldsymbol{w}+\boldsymbol{w}^{\top}(\bar{\boldsymbol{\varphi}}^{\top}\bar{\boldsymbol{\varphi}})\boldsymbol{w}\right)\\
&\quad=\boldsymbol{w}^{\top}\boldsymbol{w}+\frac{1}{\sigma_*^2}\left((\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\boldsymbol{w})^{\top}(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\boldsymbol{w})+(\bar{\boldsymbol{\varphi}}\boldsymbol{w})^{\top}(\bar{\boldsymbol{\varphi}}\boldsymbol{w})\right)\ ,
\end{aligned}$$

while the linear term is the following sum:

$$-\boldsymbol{w}^{\top}\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}-\Sigma_{\boldsymbol{z},\boldsymbol{h}}^{-1}\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}^{\top}\boldsymbol{w}=-\frac{1}{\sigma_*^2}\left((\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\boldsymbol{w})^{\top}\boldsymbol{z}+\boldsymbol{z}^{\top}(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\boldsymbol{w})+(\bar{\boldsymbol{\varphi}}\boldsymbol{w})^{\top}\boldsymbol{h}+\boldsymbol{h}^{\top}(\bar{\boldsymbol{\varphi}}\boldsymbol{w})\right)\ .$$

□

From (11), $\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})$ depends on the weights. Therefore, we infer from Proposition (1) that, given $(\boldsymbol{z},\boldsymbol{h})$, $\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})$ is a random variable distributed as:

$$\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})\,|\,(\boldsymbol{z},\boldsymbol{h})\sim\mathcal{N}\left(\boldsymbol{\varphi}_{\mathcal{L}}^{\top}(\boldsymbol{x})\boldsymbol{\mu}_{\boldsymbol{z},\boldsymbol{h}}\,,\,\boldsymbol{\varphi}_{\mathcal{L}}^{\top}(\boldsymbol{x})\Sigma_{\boldsymbol{z},\boldsymbol{h}}\boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x})\right)\ . \tag{14}$$

The conditional expectation of the differential operator given $(\boldsymbol{z},\boldsymbol{h})$ is obtained by integrating $g(\boldsymbol{x})$ with respect to the conditional density of $\boldsymbol{W}$.

Proposition 2. *Let us define*

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

and the regularized Gram matrix $C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)$,

$$C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)=\sigma_*^2I_{m+d}+\left(\begin{array}{cc}
k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right) & k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)\\
k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{\top} & k\left(\bar{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)
\end{array}\right)\ . \tag{15}$$

The expectation of $\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})$ *given* $(\boldsymbol{z},\boldsymbol{h})$ *is equal to*

$$\begin{aligned}
&\mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})\,|\,\boldsymbol{z},\boldsymbol{h}\right)=\\
&\left(\begin{array}{c}
k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\boldsymbol{x}\right)\\
k_{\mathcal{L}}\left(\bar{\boldsymbol{X}},\boldsymbol{x}\right)
\end{array}\right)^{\top}C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{-1}\left(\begin{array}{c}\boldsymbol{z}\\ \boldsymbol{h}\end{array}\right)\ .
\end{aligned} \tag{16}$$

Proof. From equations (14) and (13), using the properties of conditional expectations of the multivariate normal law, we find that

$$\mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}}g(\boldsymbol{x})\,|\,\boldsymbol{z},\boldsymbol{h}\right)\ =\ \boldsymbol{\varphi}_{\mathcal{L}}^{\top}(\boldsymbol{x})\left(\sigma_*^2I_n+\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^{\top}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}+\bar{\boldsymbol{\varphi}}^{\top}\bar{\boldsymbol{\varphi}}\right)^{-1}\left(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^{\top}\boldsymbol{z}+\bar{\boldsymbol{\varphi}}^{\top}\boldsymbol{h}\right)\ .$$

The matrix inverted in this equation can be rewritten as follows:

$$\sigma_*^2 I_n + \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} + \bar{\boldsymbol{\varphi}}^\top \bar{\boldsymbol{\varphi}} = \sigma_*^2 \left(I_n + \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix}^\top \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix} \right). \tag{17}$$

We next use the Woodbury formula which states that for a $n\times(m+d)$ matrix U and a $(m+d)\times n$ matrix V,

$$(I_n + UV)^{-1} \;=\; I_n - U\left(I_{m+d} + VU\right)^{-1} V\,. \tag{18}$$

This equality allows us to rewrite (17) as:

$$\begin{aligned} &\left(I_n + \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix}^\top \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix} \right)^{-1} = \\ &I_n - \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix}^\top \left(I_{m+d} + \sigma_*^{-2} \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^\top \\ \bar{\boldsymbol{\varphi}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \bar{\boldsymbol{\varphi}}\bar{\boldsymbol{\varphi}}^\top \end{pmatrix} \right)^{-1} \begin{pmatrix} \sigma_*^{-1}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \sigma_*^{-1}\bar{\boldsymbol{\varphi}} \end{pmatrix}. \end{aligned} \tag{19}$$

Injecting this expression into the expectation, one gets

$$\begin{aligned} \mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) = \sigma_*^{-2}\boldsymbol{\varphi}_{\mathcal{L}}^\top(\boldsymbol{x}) \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \bar{\boldsymbol{\varphi}} \end{pmatrix}^\top \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix} - \boldsymbol{\varphi}_{\mathcal{L}}^\top(\boldsymbol{x}) \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \bar{\boldsymbol{\varphi}} \end{pmatrix}^\top \times \\ \left(I_{m+d} + \sigma_*^{-2} \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^\top \\ \bar{\boldsymbol{\varphi}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \bar{\boldsymbol{\varphi}}\bar{\boldsymbol{\varphi}}^\top \end{pmatrix} \right)^{-1} \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^\top \\ \bar{\boldsymbol{\varphi}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \bar{\boldsymbol{\varphi}}\bar{\boldsymbol{\varphi}}^\top \end{pmatrix} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix}. \end{aligned}$$

We obtain (16) with the following relation:

$$\begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^\top \\ \bar{\boldsymbol{\varphi}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \bar{\boldsymbol{\varphi}}\bar{\boldsymbol{\varphi}}^\top \end{pmatrix} = \sigma_*^2 \left[I_{m+d} + \sigma_*^{-2} \begin{pmatrix} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \mathring{\boldsymbol{\varphi}}_{\mathcal{L}}\bar{\boldsymbol{\varphi}}^\top \\ \bar{\boldsymbol{\varphi}}\mathring{\boldsymbol{\varphi}}_{\mathcal{L}}^\top & \bar{\boldsymbol{\varphi}}\bar{\boldsymbol{\varphi}}^\top \end{pmatrix} - I_{m+d} \right].$$

The claim follows from the definition of basis functions. □

Notice that Equations (15) and (16) do not directly depend on basis functions $\varphi_j(.)$, but only on the kernel. Therefore, we do not need to know the analytical expressions of the basis functions to compute $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ and $\mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right)$.

Proposition 3. *The conditional variance of $\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x})$ given $\boldsymbol{z}$ and $\boldsymbol{h}$ does not depend on $(\boldsymbol{z}, \boldsymbol{h})$, and is equal to*

$$\mathbb{V}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) = k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right) - \begin{pmatrix} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) \\ k_{\mathcal{L}}\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right) \end{pmatrix}^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) \\ k_{\mathcal{L}}\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right) \end{pmatrix}. \tag{20}$$

Proof. This expression results from the definition of $\Sigma_{\boldsymbol{x},\boldsymbol{z}}$ and from the application of the Woodbury formula, Equation (19). □

The next proposition presents the closed-form expression for the estimator of the pricing function using the CGPR method.

Proposition 4. *Let us define $k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) = \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_j} k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_j)\right)_{j=1,\dots,m}$ and the $(m+d)$-vector:*

$$\boldsymbol{\beta}(\boldsymbol{x}) = \begin{pmatrix} k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) \\ k\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right) \end{pmatrix}. \tag{21}$$

The CGPR estimator $\widehat{f}(\boldsymbol{x}) = \mathbb{E}\left(g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right)$ of option prices is equal to

$$\widehat{f}(\boldsymbol{x}) \;=\; \boldsymbol{\beta}(\boldsymbol{x})^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix}. \tag{22}$$

Proof. From Eq. (16), the conditional expectation of $\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x})$ is a linear combination of $\mathcal{L}_{\boldsymbol{x}} k(\boldsymbol{x}, \bar{\boldsymbol{x}}_j)$ and $\mathcal{L}_{\boldsymbol{x}} \mathcal{L}_{\boldsymbol{x}_j} k(\boldsymbol{x}, \boldsymbol{x}_j)$:

$$\begin{aligned}\mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) &= \left(\begin{array}{cc} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \boldsymbol{x}\right) & k_{\mathcal{L}}\left(\bar{\boldsymbol{X}}, \boldsymbol{x}\right) \end{array} \right) \widehat{\boldsymbol{\alpha}} \\ &= \sum_{j=1}^{m} \widehat{\alpha}_j \mathcal{L}_{\mathring{\boldsymbol{x}}_j} \mathcal{L}_{\boldsymbol{x}} k(\mathring{\boldsymbol{x}}_j, \boldsymbol{x}) + \sum_{j=1}^{d} \widehat{\alpha}_{m+j} \mathcal{L}_{\boldsymbol{x}} k(\bar{\boldsymbol{x}}_j, \boldsymbol{x}) \,,\end{aligned}$$

where the m-vector $\widehat{\boldsymbol{\alpha}} = (\hat{\alpha}_j)_{j=1,\ldots,m}$ is defined by

$$\widehat{\boldsymbol{\alpha}} = \left(\sigma_*^2 I_{m+d} + \left(\begin{array}{cc} k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right) & k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) \\ k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{\top} & k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) \end{array} \right) \right)^{-1} \left(\begin{array}{c} \boldsymbol{z} \\ \boldsymbol{h} \end{array} \right) . \tag{23}$$

Using the linearity of the operator, the estimator $\widehat{f}(\boldsymbol{x})$ is given by

$$\begin{aligned}\widehat{f}(\boldsymbol{x}) &= \mathcal{L}_{\boldsymbol{x}}^{-1} \mathbb{E}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) \qquad (24) \\ &= \sum_{j=1}^{m} \widehat{\alpha}_j \mathcal{L}_{\mathring{\boldsymbol{x}}_j} k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_j) + \sum_{j=1}^{d} \widehat{\alpha}_{m+j} k(\bar{\boldsymbol{x}}_j, \boldsymbol{x}) \,.\end{aligned}$$

Eq. (22) is the matrix reformulation of this last equation. □

As previously mentioned, adding Gaussian noise to h_i's and z_i's introduces a diagonal matrix $\sigma_*^2 I_{m+d}$ in the definition (22) of $\widehat{f}(\boldsymbol{x})$. This diagonal matrix is a Tikhonov regularization factor that stabilizes a possibly ill-conditioned inverse matrix. The next proposition provides the a posteriori conditional variance of the estimator.

Proposition 5. *Let $\boldsymbol{\beta}(\boldsymbol{x})$ be the $(m+d)$ vector defined in Eq. (21). Then the conditional variance of the CGPR estimator (22) does not depend on $(\boldsymbol{z}, \boldsymbol{h})$ and is equal to*

$$\mathbb{V}\left(g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{\beta}(\boldsymbol{x})^{\top} C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \boldsymbol{\beta}(\boldsymbol{x}) \,. \tag{25}$$

Proof. By definition of $g(\boldsymbol{x})$, the $\boldsymbol{z}, \boldsymbol{h}$-conditional variance of $\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x})$ is equal to:

$$\begin{aligned}\mathbb{V}\left(\mathcal{L}_{\boldsymbol{x}} g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) &= \mathbb{V}\left(\sum_{k=1}^{n} W_k \mathcal{L}_{\boldsymbol{x}} \varphi_k(\boldsymbol{x}) \,\middle|\, \boldsymbol{z}, \boldsymbol{h} \right) \\ &= \boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x})^{\top} \Sigma_{\boldsymbol{z},\boldsymbol{h}} \boldsymbol{\varphi}_{\mathcal{L}}(\boldsymbol{x}) \,,\end{aligned}$$

where $\Sigma_{\boldsymbol{z},\boldsymbol{h}}$ is the a posteriori covariance of weights defined in Equation (13). By comparison with Eq. (20), we deduce an alternative formulation[3] of $\Sigma_{\boldsymbol{z},\boldsymbol{h}}$:

$$\Sigma_{\boldsymbol{z},\boldsymbol{h}} = I_n - \left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \bar{\boldsymbol{\varphi}} \end{array} \right)^{\top} C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \\ \bar{\boldsymbol{\varphi}} \end{array} \right) . \tag{26}$$

On the other hand, the variance of $g(\boldsymbol{x})$ is equal to

$$\begin{aligned}\mathbb{V}\left(g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) &= \mathbb{V}\left(\sum_{i=1}^{n} W_i \varphi_j(\boldsymbol{x}) \middle| \, \boldsymbol{z}, \boldsymbol{h} \right) \\ &= \boldsymbol{\varphi}(\boldsymbol{x})^{\top} \Sigma_{\boldsymbol{z},\boldsymbol{h}} \boldsymbol{\varphi}(\boldsymbol{x}).\end{aligned}$$

Combining this last expression with Eq. (26) allows us to infer that:

$$\mathbb{V}\left(g(\boldsymbol{x}) \,|\, \boldsymbol{z}, \boldsymbol{h}\right) = \boldsymbol{\varphi}(\boldsymbol{x})^{\top} \boldsymbol{\varphi}(\boldsymbol{x}) - \left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{\varphi}(\boldsymbol{x}) \\ \bar{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\boldsymbol{x}) \end{array} \right)^{\top} C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \left(\begin{array}{c} \mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{\varphi}(\boldsymbol{x}) \\ \bar{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\boldsymbol{x}) \end{array} \right) .$$

As $\boldsymbol{\varphi}(\boldsymbol{x})^{\top} \boldsymbol{\varphi}(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x})$, $(\mathring{\boldsymbol{\varphi}}_{\mathcal{L}} \boldsymbol{\varphi}(\boldsymbol{x}))_i = \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)$ and $(\bar{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\boldsymbol{x}))_i = k(\boldsymbol{x}, \bar{\boldsymbol{x}}_i)$, we obtain Equation (25). □

[3] The same expression can be obtained by applying the Woodbury formula.

4.2 Algorithm and the Gaussian kernel

The frame 1 summarizes the algorithm for computing derivative prices by CGPR. In practice, we do not add noises ϵ and $\bar{\epsilon}$ to $(z_i)_{i=1,...,m}$ and $(h_i)_{i=1,...,d}$. This has no impact on calculations and slightly improves the accuracy of the algorithm.

Algorithm 1 CGPR approximation of the price $\widehat{f}(\boldsymbol{x})$ of a derivative with payoff $h(\boldsymbol{x})$.

---

1. Sample m points $\mathring{\boldsymbol{x}}_i = (t_i, y_i) \in \mathring{\mathcal{X}}$ and set $(z_i = 0)_{i=1,...,m}$.

2. Sample d points $\bar{\boldsymbol{x}}_i = (t, y_i) \in \bar{\mathcal{X}}$ and set $(h_i = h(\bar{\boldsymbol{x}}_i))_{i=1...d}$.

3. Compute the regularized Gram matrix

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

   where

$$\begin{cases} \left(k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right)\right)_{i,j} &=& \mathcal{L}_{\mathring{\boldsymbol{x}}_i}\mathcal{L}_{\mathring{\boldsymbol{x}}_j} k(\mathring{\boldsymbol{x}}_i, \mathring{\boldsymbol{x}}_j) & i,j = 1,...,m\,, \\ \left(k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)\right)_{i,j} &=& \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k(\mathring{\boldsymbol{x}}_i, \bar{\boldsymbol{x}}_j) & i = 1,...,m\,,\ j = 1,...,d\,, \\ \left(k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)\right)_{i,j} &=& k(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{x}}_j) & i,j = 1,...,d\,. \end{cases} \tag{27}$$

4. Invert the Gram matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ and compute for an out of sample $\boldsymbol{x} \in \mathcal{X}$:

$$\boldsymbol{\beta}(\boldsymbol{x}) = \begin{pmatrix} \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_i} k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)\right)_{i=1,...,m} \\ \left(k(\bar{\boldsymbol{x}}_i, \boldsymbol{x})\right)_{i=1,...,d} \end{pmatrix}.$$

5. The pricing function is $\widehat{f}(\boldsymbol{x}) = \boldsymbol{\beta}(\boldsymbol{x})^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix}$.

The computation of estimates requires applying the differential operator $\mathcal{L}_{\boldsymbol{x}}$ twice to the kernel function. Many alternatives exist such as the rational quadratic (RQ), Matern 3/2 (M32), Matern 5/2 (M52) kernels, see e.g. Beckers (2021). We opt for a Gaussian kernel (also called radial basis function) in our numerical illustrations. This kernel is easy to differentiate and allows us to infer fully analytical expressions for the vector $\boldsymbol{\beta}(\boldsymbol{x})$ and sub-blocks of the Gram matrix. In a general setting, the Gaussian kernel has the form:

$$k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \;=\; \exp\left(-\frac{\left(t - \tilde{t}\right)^2}{2\,\eta_t^2} - \sum_{k=1}^{p-1} \frac{\left(y_k - \tilde{y}_k\right)^2}{2\,\eta_k^2}\right) \quad \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}\,. \tag{28}$$

The following results depends on the following p-vector:

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

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

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

With a Gaussian kernel, the vector $\boldsymbol{\beta}(\boldsymbol{x})$ admits a closed-form expression as stated in the next proposition. We assume in later developments that the risk-free rate $r(\boldsymbol{x}) = r \in \mathbb{R}$ is constant, without loss of generality.

Proposition 6. *For all $\boldsymbol{x} = (t, \boldsymbol{y}) \in \mathcal{X}$, the approximated derivative price is $\widehat{f}(\boldsymbol{x}) = \boldsymbol{\beta}(\boldsymbol{x})^\top \times C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1}(\boldsymbol{z}, \boldsymbol{h})^\top$. With a Gaussian kernel (28), the $(m+d)$-vector $\boldsymbol{\beta}(\boldsymbol{x}) = (\boldsymbol{\beta_1}(\boldsymbol{x}), \boldsymbol{\beta_2}(\boldsymbol{x}))^\top$ has the following components:*

$$\begin{cases} \boldsymbol{\beta_1}(\boldsymbol{x}) = \Big(\big(\big(t-\mathring{t}_i\big)/\eta_t^2 - r + \boldsymbol{\mu}_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i) \\ \quad + \frac{1}{2} tr\left(\Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)\Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{H}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)\right)\big)\; k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)\Big)_{i=1,\dots,m}, \\ \boldsymbol{\beta_2}(\boldsymbol{x}) = \left(e^{-(\bar{t}_i - t)^2/\left(2\,\eta_t^2\right) - \sum_{k=1}^{p-1}\left(\bar{y}_{i,k} - t\right)^2/\left(2\,\eta_k^2\right)}\right)_{i=1,\dots,d}. \end{cases}$$

Proof. The last d elements of $\boldsymbol{\beta}(\boldsymbol{x})$ are equal to $(k(\bar{\boldsymbol{x}}_i, \boldsymbol{x}))_{i=1,\dots,d}$, directly computable with Equation (28). By direct differentiation, we check that the gradient and Hessian of the kernel are given by

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

We infer from this last equation, that $\mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ depends on $\boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ and $\boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ as follows:

$$\begin{aligned} \mathcal{L}_{\tilde{\boldsymbol{x}}} k(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \Big(\big(t - \tilde{t}\big)/\eta_t^2 - r + \boldsymbol{\mu}_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^\top \boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \\ + \frac{1}{2}\mathrm{tr}\left(\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^\top \boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}})\right)\Big)\, k(\boldsymbol{x}, \tilde{\boldsymbol{x}}). \end{aligned} \tag{29}$$

Equations (29) allows us to retrieve the expression of $\boldsymbol{\beta_1}(\boldsymbol{x})$. □

The construction of the Gram matrix, $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$, requires to compute the sub-blocks: $k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$, $k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ and $k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}}, \mathring{\boldsymbol{X}}\right)$. When the kernel is Gaussian, they admit general closed-form expressions.

Proposition 7. *The sub-blocks $k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ and $k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ are computed with the following analytical formulas*

$$\begin{cases} k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) = \left(e^{-(\bar{t}_i - \bar{t}_j)^2/\left(2\,\eta_t^2\right) - \sum_{k=1}^{p-1}\left(\bar{y}_{i,k} - \bar{y}_{j,k}\right)^2/\left(2\,\eta_k^2\right)}\right)_{i,j=1,\dots,d}, \\ k_{\mathcal{L}}\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right) = \Big(\big(\big(\bar{t}_j - \mathring{t}_i\big)/\eta_t^2 - r + \boldsymbol{\mu}_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\bar{\boldsymbol{x}}_j, \mathring{\boldsymbol{x}}_i) \\ \quad + \frac{1}{2} tr\left(\Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)\Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{H}_{\mathring{\boldsymbol{y}}_i}(\bar{\boldsymbol{x}}_j, \mathring{\boldsymbol{x}}_i)\right)\big)\; k(\bar{\boldsymbol{x}}_j, \mathring{\boldsymbol{x}}_i)\Big)_{i=1,\dots,m\; j=1,\dots,d}. \end{cases}$$

The expression of $k\left(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$ is a direct consequence of the kernel choice. The second equation comes from the symmetry of the kernel and Equation (29). Let us denote by $\boldsymbol{e}_k$'s the canonical vectors of $\mathbb{R}^{p-1}$. We define the following matrix:

$$\boldsymbol{H}_{\boldsymbol{y}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \boldsymbol{H}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + \begin{pmatrix} 2/\eta_1^2 & 0 & 0 \\ 0 & \dots & 0 \\ 0 & 0 & 2/\eta_{p-1}^2 \end{pmatrix}.$$

Proposition 8. *The sub-block* $k_{\mathcal{L}^2}\left(\mathring{\boldsymbol{X}},\mathring{\boldsymbol{X}}\right)$ *of* $C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)$ *is the matrix of* $\mathcal{L}_{\mathring{\boldsymbol{x}}_i}\mathcal{L}_{\mathring{\boldsymbol{x}}_j}k(\mathring{\boldsymbol{x}}_i,\mathring{\boldsymbol{x}}_j)$ *for* $i,j=1,...,m$ *where*

$$\begin{aligned}\mathcal{L}_{\boldsymbol{x}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) &= \partial_t\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \boldsymbol{\mu}_{\boldsymbol{y}}(\boldsymbol{x})^\top\nabla_{\boldsymbol{y}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ &-r\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \frac{1}{2}tr\left(\Sigma_{\boldsymbol{y}}(\boldsymbol{x})\Sigma_{\boldsymbol{y}}(\boldsymbol{x})^\top\mathcal{H}_{\boldsymbol{y}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})\right).\end{aligned} \tag{30}$$

Using Equation (29), the partial derivative in (30) is given by

$$\partial_t\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) \;=\; \frac{1}{\eta_t^2}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) - \left(\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})\right)\frac{\left(t-\tilde{t}\right)}{\eta_t^2}, \tag{31}$$

while the gradient $\nabla_{\boldsymbol{y}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \left(\partial_{y_k}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})\right)_{k=1,...,p-1}$ *is a* $(p-1)$ *vector filled by :*

$$\begin{aligned}\partial_{y_k}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) &= \Big[\boldsymbol{e}_k^\top\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^\top\boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ &+\boldsymbol{e}_k^\top\boldsymbol{\mu}_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})\Big]\frac{k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)}{\eta_k^2} - \mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\boldsymbol{e}_k^\top\boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}),\end{aligned} \tag{32}$$

Finally The Hessian $\mathcal{H}_{\boldsymbol{y}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \left(\partial_{y_l}\partial_{y_k}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})\right)_{k,l=1,...,p-1}$ *in Equation (30) is a* $(p-1)\times(p-1)$ *matrix with elements:*

$$\begin{aligned}&\partial_{y_l}\partial_{y_k}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right) = \left(\boldsymbol{e}_k^\top\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})\Sigma_{\boldsymbol{y}}(\tilde{\boldsymbol{x}})^\top\boldsymbol{e}_l\right)\frac{k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)}{\eta_k^2\eta_l^2} \\ &-\left(\partial_{y_k}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right)\boldsymbol{e}_l^\top\boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) - \left(\partial_{y_l}\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\right)\boldsymbol{e}_k^\top\boldsymbol{d}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ &-\mathcal{L}_{\tilde{\boldsymbol{x}}}k\left(\boldsymbol{x},\tilde{\boldsymbol{x}}\right)\left(\boldsymbol{e}_l^\top\left(\boldsymbol{H}_{\boldsymbol{y}}(\boldsymbol{x},\tilde{\boldsymbol{x}})\right)\boldsymbol{e}_k\right).\end{aligned} \tag{33}$$

In Section 6 and 5, we will present the specific forms of $\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})$ and $\mathcal{L}_{\boldsymbol{x}}\mathcal{L}_{\tilde{\boldsymbol{x}}}k(\boldsymbol{x},\tilde{\boldsymbol{x}})$ in the Heston and Geometric diffusive models.

We conclude this section by discussing the fit of kernel bandwidths. Equations (22) and (25) correspond to the conditional expectation and variance of $g(\boldsymbol{x})\,|\,\boldsymbol{z},\boldsymbol{h}$ with a multivariate normal distribution:

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

Based on this relation, we can tune the bandwidths of Gaussian kernels by maximizing the marginal log-likelihood of $(\boldsymbol{Z},\boldsymbol{H})$. The kernel parameters, $\boldsymbol{\eta} = \{\eta_t,\eta_1,...,\eta_{p-1}\}$ are optimized by maximization of the following expression:

$$\log l(\boldsymbol{z},\boldsymbol{h}) = -\frac{1}{2}\log\left|C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)\right| - \frac{1}{2}\begin{pmatrix}\boldsymbol{z}\\\boldsymbol{h}\end{pmatrix}^\top C\left(\mathring{\boldsymbol{X}},\bar{\boldsymbol{X}}\right)^{-1}\begin{pmatrix}\boldsymbol{z}\\\boldsymbol{h}\end{pmatrix} + \frac{(m+d)}{2}\log 2\pi\,. \tag{34}$$

In practice, this maximization is subject to numerical instabilities because the log-likelihood is a non-convex function of hyper-parameters. We propose an efficient and robust alternative in which the bandwidths minimize the errors of approximation on the inner and boundary domains. For this purpose, we generate two new samples, exclusively used for tuning $\boldsymbol{\eta}$. The first sample set contains m points in the inner domain and is denoted by $\mathring{\boldsymbol{X}}' = \left\{\mathring{\boldsymbol{x}}_i'\right\}_{i=1,...,m}$ with $\mathring{\boldsymbol{x}}_i' \in \mathring{\mathcal{X}}$. We draw a second sample set of d points on the boundary. This set is $\bar{\boldsymbol{X}}' = \left\{\bar{\boldsymbol{x}}_i'\right\}_{i=1,...,d}$ with

$\bar{\boldsymbol{x}}_i' \in \bar{\mathcal{X}}$. We define the mean square errors on the inner and terminal domains as follows:

$$\mathring{Err}(\boldsymbol{\eta}) = \frac{1}{m} \sum_{\mathring{\boldsymbol{x}}_i' \in \mathring{\boldsymbol{X}}'} \left(\mathcal{L}_{\mathring{\boldsymbol{x}}_i'} \widehat{f}\left(\mathring{\boldsymbol{x}}_i'\right) \right)^2 ,$$
$$\bar{Err}(\boldsymbol{\eta}) = \frac{1}{d} \sum_{\bar{\boldsymbol{x}}_i' \in \bar{\boldsymbol{X}}'} \left(\widehat{f}\left(\bar{\boldsymbol{x}}_i'\right) - h\left(\bar{\boldsymbol{x}}_i'\right) \right)^2 .$$

The optimal hyper-parameters $\boldsymbol{\eta}^*$, are obtained by minimization of the total error, e.g. with a Nelder-Mead algorithm:

$$\boldsymbol{\eta}^* = \arg\min_{\boldsymbol{\eta}} \left(\mathring{Err}(\boldsymbol{\eta}) + \bar{Err}(\boldsymbol{\eta}) \right) . \tag{35}$$

We will see that this method yields good results for the two optimization problems analyzed in the following section.

4.3 The Greeks

Since the pricing function $\widehat{f}(\boldsymbol{x})$ is linear in $\boldsymbol{\beta}(\boldsymbol{x})$, we can efficiently approximate the Greeks with analytical expressions. The Greeks are the partial derivatives of the pricing function, representing the sensitivity of the option's value to changes in one or more underlying risk factors. They are essential tools for risk management, but their computation via finite differences is often prone to numerical instabilities. The sensitivity of the option price to time, called the "theta", is in our case given by

$$\partial_t \widehat{f}(\boldsymbol{x}) = \left(\partial_t \boldsymbol{\beta}(\boldsymbol{x})\right)^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix} . \tag{36}$$

The sensitivity of $\widehat{f}(\boldsymbol{x})$ to the $\hat{\imath}^{th}$ state variable, the "delta", is measured by the following derivative:

$$\partial_{y_k} \widehat{f}(\boldsymbol{x}) = \left(\partial_{y_k} \boldsymbol{\beta}(\boldsymbol{x})\right)^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix} . \tag{37}$$

Whereas the second order sensitivity, called "gamma" or "cross-gamma" is the second order derivative of the approximated prices with respect to state processes:

$$\partial_{y_l}\partial_{y_k} \widehat{f}(\boldsymbol{x}) = \left(\partial_{y_l}\partial_{y_k} \boldsymbol{\beta}(\boldsymbol{x})\right)^\top C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)^{-1} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{h} \end{pmatrix} . \tag{38}$$

When the kernel is Gaussian, partial derivatives of $\boldsymbol{\beta}(\boldsymbol{x})$ admits closed-form expressions. The partial derivative of $\boldsymbol{\beta}(\boldsymbol{x})$ w.r.t. time is in this case equal to:

$$\partial_t \boldsymbol{\beta}(\boldsymbol{x}) \quad = \quad \begin{pmatrix} \left(\partial_t \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right)\right)_{i=1,\dots,m} \\ \left(\partial_t k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right)\right)_{j=1,\dots,d} \end{pmatrix} . \tag{39}$$

Using Equation (29), the terms of $\partial_t \boldsymbol{\beta}(\boldsymbol{x})$ are:

$$\begin{cases} \partial_t \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) & = \frac{k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)}{\eta_t^2} - \frac{\left(t - \mathring{t}_i\right)}{\eta_t^2} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) \\ \partial_t k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right) & = \frac{\left(\bar{t}_j - t\right)}{\eta_t^2} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right) . \end{cases}$$

For the "delta's", the derivative of $\boldsymbol{\beta}(\boldsymbol{x})$ w.r.t. y_k in Equation (37), is a $(m+d)$ vector given by

$$\partial_{y_k} \boldsymbol{\beta}(\boldsymbol{x}) \quad = \quad \begin{pmatrix} \left(\partial_{y_k} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right)\right)_{i=1,\dots,m} \\ \left(\partial_{y_k} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right)\right)_{j=1,\dots,d} \end{pmatrix} , \tag{40}$$

with elements equal to

$$\begin{cases} \partial_{y_k} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) & = \left[\boldsymbol{e}_k^\top \Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i) \Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i) \right. \\ & \quad \left. + \boldsymbol{\mu}_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{e}_k \right] \frac{k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)}{\eta_k^2} \\ & \quad - \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) \boldsymbol{e}_k^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)\,, \\ \partial_{y_k} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right) & = \frac{\left(\bar{y}_{j,k} - y_k\right)}{\eta_k^2} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right)\,. \end{cases}$$

The "cross-Gamma's" depend on the $(m+d)$ vectors:

$$\partial_{y_l} \partial_{y_k} \boldsymbol{\beta}(\boldsymbol{x}) \quad = \quad \begin{pmatrix} \left(\partial_{y_l} \partial_{y_k} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right)\right)_{i=1,\ldots,m} \\ \left(\partial_{y_l} \partial_{y_k} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right)\right)_{j=1,\ldots,d} \end{pmatrix}. \tag{41}$$

Using Equation (29) and (32), the elements of this vector have the following closed-form expressions:

$$\begin{cases} \partial_{y_l} \partial_{y_k} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) & = \left(\boldsymbol{e}_k^\top \Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i) \Sigma_{\boldsymbol{y}}(\mathring{\boldsymbol{x}}_i)^\top \boldsymbol{e}_l\right) \frac{k(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i)}{\eta_k^2 \eta_l^2} \\ & \quad - \left(\partial_{y_k} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right)\right) \boldsymbol{e}_l^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i) \\ & \quad - \left(\partial_{y_l} \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right)\right) \boldsymbol{e}_k^\top \boldsymbol{d}_{\mathring{\boldsymbol{y}}_i}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i) \\ & \quad - \mathcal{L}_{\mathring{\boldsymbol{x}}_i} k\left(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i\right) \left(\boldsymbol{e}_l^\top \boldsymbol{H}_{\boldsymbol{y}}(\boldsymbol{x}, \mathring{\boldsymbol{x}}_i) \boldsymbol{e}_k\right) \\ \partial_{y_l} \partial_{y_k} k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right) & = \left(\frac{\left(\bar{y}_{j,k} - y_k\right)\left(\bar{y}_{j,l} - y_l\right)}{\eta_k^2 \eta_l^2} - \mathbf{1}_{\{k=l\}} \frac{1}{\eta_k^2}\right) k\left(\bar{\boldsymbol{x}}_j, \boldsymbol{x}\right)\,. \end{cases}$$

The calculation of Greeks is challenging in absence of analytical formulas due to numerical instabilities when approximating partial derivatives. Having (approximated) closed-form expressions is a real added value for managing risks.

5 European options in a geometric diffusive market

In this first example, we benchmark our approach with the Black & Scholes formula. We consider a financial market with a single risky assets, noted $(S_t)_{t\geq 0}$ ruled by a Brownian motion $(B_t)_{t\geq 0}$. The state variable is $\boldsymbol{y_t} = S_t$. This is ruled by the stochastic differential equation (1) where $\boldsymbol{\mu_y}(.)$ and $\Sigma_{\boldsymbol{y}}(.)$ are scalar functions:

$$\boldsymbol{\mu_y}(t, \boldsymbol{y}_t) = r\, S_t \quad \text{and} \quad \Sigma_{\boldsymbol{y}}(t, \boldsymbol{y}_t) = \sigma S_t\,,$$

where r is the risk-free rate and $\sigma \in \mathbb{R}^+$. $\boldsymbol{x} = (t, s)$ is a bivariate vector ($p = 2$) combining time t and a realization s of $\boldsymbol{y_t} = S_t$. The domain of $\boldsymbol{x}$ is $\mathcal{X} = \mathbb{R}^{2+}$. We evaluate a call option with a strike price, K and a payoff $\mathcal{P}(\bar{\boldsymbol{x}}) = (\bar{s} - K)_+$. We solve the FK equation on the inner domain $\mathring{\mathcal{X}} = [0, T) \times (0, s_{max})$ where s_{max} is large. The boundary set is the union of three subsets, $\bar{\mathcal{X}} = \bar{\mathcal{X}}_0 \cup \bar{\mathcal{X}}_{max} \cup \bar{\mathcal{X}}_T$, where

$$\begin{aligned} \bar{\mathcal{X}}_0 &= \{(t, 0) \mid t \in [0, T)\}\,, \\ \bar{\mathcal{X}}_{max} &= \{(t, s_{max}) \mid t \in [0, T)\}\,, \\ \bar{\mathcal{X}}_T &= \{(T, s) \mid s \in \mathbb{R}^+\}\,. \end{aligned} \tag{42}$$

The boundary function for $\bar{\boldsymbol{x}} = (\bar{t}, \bar{s})$, is in this particular case defined as follows

$$h\left(\bar{\boldsymbol{x}}\right) = \begin{cases} 0 & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}}_0\,, \\ e^{-r(T-\bar{t})}(\bar{s} - K) & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}}_{max}\,, \\ (\bar{s} - K)_+ & \bar{\boldsymbol{x}} \in \bar{\mathcal{X}}_T\,. \end{cases} \tag{43}$$

The value of the option in state $\boldsymbol{x}$, $f(\boldsymbol{x})$, solves the FK equation (5) where the gradient and the Hessian of f with respect to $\boldsymbol{y}$ by $\nabla_{\boldsymbol{y}} f$ and $\mathcal{H}_{\boldsymbol{y}}(f)$ are this time given by:

$$\nabla_{\boldsymbol{y}} f = \partial_s f \quad \text{and} \quad \mathcal{H}_{\boldsymbol{y}}(f) = \partial_{ss} f \,.$$

We consider the Gaussian kernel, $k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right)$, defined by a positive vector of bandwidths, $\boldsymbol{\eta} = (\eta_t, \eta_1) \in \mathbb{R}^2$:

$$k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \;=\; \exp\left(-\frac{\left(t-\tilde{t}\right)^2}{2\,\eta_t^2} - \frac{\left(s-\tilde{s}\right)^2}{2\,\eta_1^2}\right).$$

We compare CGPR prices, obtained with Proposition 6, to those calculated using the Black-Scholes-Merton (BSM) formula (see Black and Scholes, 1976). This special case is interesting because exact prices are available. Table 1 presents the stock parameters, the features of the priced call options, σ_* and the kernel bandwidths. We optimize bandwidths by minimization of errors, see Equation (35). The points in training sets $\mathring{\boldsymbol{X}}$ and $\bar{\boldsymbol{X}}$ are randomly drawn according to a uniform distribution over the intervals reported in Table 1.

Parameter	Value	Parameter	Value
r	0.05	K	100
σ_1	0.25	T	0.5 , 1 , 1.5 , 2 , 2.5 , 3
$s_i^{(1)}$	$\in [0, 200]$, uniform law	t_i	$\in [0, T]$, uniform law
σ_*	0.01		

Table 1: Parameters of the BSM model, features of the call options and boundaries for sampling $\mathring{\boldsymbol{x}}_i = (t_i, s_i^{(1)})$ and $\bar{\boldsymbol{x}}_i = (T, s_i^{(1)})$.

We consider a validation set of $6 \times 50 = 300$ call options with 6 maturities T, from 6 months to 3 years and 50 initial values $S_0^{(1)}$ from 50 to 150, in increments of 2. We use training sets $\mathring{\boldsymbol{X}}$ of size m varying between 200 and 1000 points. We respectively sample $m/5$, $m/4$ and $m/3$ points in $\bar{\mathcal{X}}_0$, $\bar{\mathcal{X}}_{max}$ and $\bar{\mathcal{X}}_T$. The size d of $\bar{\boldsymbol{X}}$ is then equal to $\frac{3}{5}m$, $\frac{3}{4}m$ and m. Table 2 reports the mean absolute errors (MAEs) between CGPR and BSM prices. Regardless the combination of d and m, the MAEs are negligible but the lowest one is achieved with $d = m$. Table 2 also shows the mean relative errors (MREs) for in-the-Money options. The MRE ranges from 11 to 35 basis points, which is acceptable for industrial applications.

	Spread CGPR - BSM prices					
	Absolute			Relative (ITM)		
m	$d = 3/5m$	$d = 3/4m$	$d = m$	$d = 3/5m$	$d = 3/4m$	$d = m$
200	0.0546	0.0643	0.0655	0.0011	0.0017	0.0018
400	0.0380	0.0388	0.0400	0.0016	0.0018	0.0018
600	0.0791	0.0619	0.0282	0.0035	0.0027	0.0012

Table 2: Absolute: MAEs between CGPR and BSM prices. Relative (ITM) : MREs for in-the-Money (ITM) options ($S_0^{(1)} > K$).

Table 3 reports the computation times along with the MAEs and MREs of CGRP with $m = d$. As a numerical method, CGPR is naturally slower than the analytical valuation using the Black–Scholes formula. Nevertheless, the time required to value 300 options does not exceed 13 seconds. Table 3 also presents the average standard deviation of CGPR prices computed using Equation (25). These values are comparable regardless of the sample sizes.

	Spread CGPR - BSM prices			Times (seconds)		
m and $d = m$	Absolute	Relative (ITM)	Std	Opt.	CGPR	BSM
200	0.0655	0.0018	0.0018	5.70	1.32	0.03
400	0.0400	0.0018	0.0023	6.65	1.59	0.03
600	0.0282	0.0012	0.0018	10.58	1.80	0.04

Table 3: Absolute: MAEs between CGPR and BSM prices. Relative : MREs for in-the-Money (ITM) options ($S_0^{(1)} > K$). Std: standard deviation of CGPR estimates. Times CGPR and BSM: computation times with CGPR and BSM formula.

	Spread CGPR - BSM prices			Hyper-parameters	
T	Absolute	Relative (ITM)	St. dev.	η_t	η_1
0.5	0.0217	0.0017	0.0014	0.75	11.96
1.0	0.0248	0.0013	0.0014	0.76	13.48
1.5	0.0246	0.0010	0.0016	0.77	13.96
2.0	0.0272	0.0010	0.0018	0.76	15.19
2.5	0.0299	0.0010	0.0022	0.76	15.03
3.0	0.0409	0.0013	0.0025	0.76	15.94

Table 4: MAEs and MREs for call options and in-the-Money (ITM) options ($S_0^{(1)} > K$) per maturity. CGPR prices are computed with $m = 600$ and $d = 600$. St.dev.: standard deviations of CGPR estimates.

Table 4 presents the MAEs and MREs per maturity for options in the validation sets, and the estimated hyper-parameters. For each maturity, the spread is based on 50 option prices with $S_0^{(1)}$ ranging from 50 to 150, in increments of 2. In relative terms, the largest errors are observed for 0.5 year options (0.17%). The left plot of Figure 1 compares CGPR and BSM prices of 1.5 year options. The right plot displays a heatmap of absolute spreads between BSM and CGPR call prices with respect to T and $S_0^{(1)}$. These graphs confirm the overall accuracy of the CGPR method and reveal that the highest errors occurs for deep in the money long-term options (near the boundary on which the FK equation is solved).

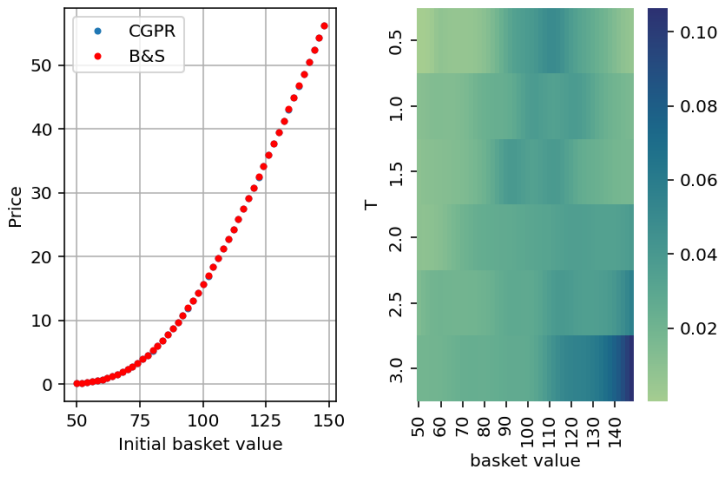


Figure 1: Left: BSM (B&S) and CGPR (Kernel) 1.5-year call prices computed with $m = 400$ and d =240 for $S_0^{(1)}$ ranging from 50 to 150, by step of 2. Right: heatmap of absolute spreads between BSM and CGPR call prices with respect to T and $S_0^{(1)}$.

6 Options valuation in the Heston model

We test the CGPR Algorithm 1 for pricing European options in the Heston model. We consider a market with two assets. The cash account earns a constant risk-free rate $r \in \mathbb{R}^+$. The stock price process $S = (S_t)_{t\geq 0}$, is ruled by a geometric Brownian diffusion with a stochastic variance process, $V = (V_t)_{t\geq 0}$. The price and variance processes are defined on a complete filtered probability space $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{Q})$ generated by two independent Brownian motions $\boldsymbol{B}_t = \left(B_t^{(1)}, B_t^{(2)}\right)^\top_{t\geq 0}$. The vector of state variables $\boldsymbol{y_t} = (S_t, V_t)$ is driven by the SDE (1) with

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

Parameters κ, $\gamma \in \mathbb{R}^+$ are respectively the speed of mean reversion and the long term average of the stock variance while $\sigma \in \mathbb{R}^+$ is the volatility of the stock variance. In this expression, Σ_S and Σ_V are 1×2 vectors such that the matrix

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

i.e., ρ_{SV} is the instantaneous coefficient of correlation between S_t and V_t and Σ is the upper Choleski decomposition of the correlation matrix.

As previously, we note $\boldsymbol{x} = (t, \boldsymbol{y})$, where t is the time and $\boldsymbol{y} = (v, s)$ are the state variables at the corresponding time. The domain of $\boldsymbol{x}$ is $\mathcal{X} = [0, T] \times \mathbb{R}^{+,2}$. In the numerical illustration, we consider a European call option on S with expiry T and strike price K. The boundary set and boundary function are defined by Equations (42) and (43). The Heston model is a popular framework for option valuation, with many extensions. Prices are computed by a Fast Fourier Transform (FFT), Monte-Carlo simulations or finite difference approximation. For instance, Düring and Miles (2017) propose an alternating direction implicite finite difference scheme for pricing options. Hainaut (2023 b) evaluate spread and exchange options in a rough-Heston model. Souto Arias (2025) use the COS method for valuing an extension of the Heston model with a Q-Hawkes jump process. Cuomo et al. (2020) assume that the option price in the Heston model, is a weighted sum of radial basis function on a grid. This is very close to the idea underlying Physics Inspired Neural Network (PINN's). In this article, we opt for the Monte-Carlo (MC) method because of its accuracy.

The price of the derivative is equal to the risk-neutral expectation of the discounted payoff. It solves the FK equation (5) where the gradient and the Hessian of f with respect to $\boldsymbol{y}$ by $\nabla_{\boldsymbol{y}} f$ and $\mathcal{H}_{\boldsymbol{y}}(f)$ are here given by:

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

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

$$k\left(\boldsymbol{x}, \tilde{\boldsymbol{x}}\right) \;=\; \exp\left(-\frac{\left(t-\tilde{t}\right)^2}{2\,\eta_t^2} - \frac{\left(s-\tilde{s}\right)^2}{2\,\eta_s^2} - \frac{\left(v-\tilde{v}\right)^2}{2\,\eta_v^2}\right) \quad \forall \boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}\,. \tag{44}$$

The estimator of the price function, $\widehat{f}(\boldsymbol{x}) = \mathbb{E}\left(g(\boldsymbol{x} \,|\, \boldsymbol{z}, \boldsymbol{h})\right)$ is provided in Proposition 4.

Let us now analyze the performance of our CGPR estimator. To this end, we will benchmark our results against MC option prices. The CGPR algorithm is implemented in Python

and tested on a laptop with an Intel Core Ultra 7, 1.40 Ghz with 32Gb of RAM. Table 5 presents the parameters of the Heston model. We evaluate standard call options with maturities ranging from 6 months to 3 years. The samples in $\mathring{\mathcal{X}}$ and $\bar{\mathcal{X}}$ are obtained by randomly drawing from independent uniform distributions over intervals reported in Table 5.

Parameter	Value	Parameter	Value
r	0.05	K	100
σ	0.30	T	0.5 , 1 , 1.5 , 2 , 2.5 , 3
γ	0.15^2	s_i	$\in [0, 300]$, uniform law
κ	0.95	t_i	$\in [0, T]$, uniform law
ρ_{SV}	-0.30	v_i	$\in [0\ ,\ 8\gamma]$, uniform law

Table 5: Parameters of the Heston model, of the call options, and of the sampling model used for building the training sets $\mathring{\boldsymbol{X}}$ and $\bar{\boldsymbol{X}}$. We draw m samples $\mathring{\boldsymbol{x}}_i = (t_i, s_i, v_i)$ and d samples $\bar{\boldsymbol{x}}_i = (T, s_i, v_i)$. The samples s_i, v_i (and t_i for $\mathring{\boldsymbol{x}}_i$) are drawn independently of each other.

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

Table 6: Features of 2400 options in the validation set.

We compute CGPR and MC prices at time $t = 0$ for $6 \times 20 \times 20 = 2400$ call options with features specified in Tables 5 and 6. The features of the 2400 options are selected to uniformly cover the hypercube $[0, 3] \times [50, 150] \times [\gamma/5\ ,\ 5\gamma]$, representing expiry, stock and variance values, respectively. We consider various sample sizes m for $\mathring{\boldsymbol{X}}$, ranging from 400 to 800 points. Specifically, we respectively sample $m/5$, $m/4$, $m/3$ points in $\bar{\mathcal{X}}_0$, $\bar{\mathcal{X}}_{max}$ and $\bar{\mathcal{X}}_T$. The boundary sample $\bar{\boldsymbol{X}}$ has a size d equal to $\frac{3}{5}m$, $\frac{3m}{4}$ and m. Table 7 presents MAEs and MREs between CGPR and MC prices, obtained using 5000 simulations.The MC stock and volatility paths are generated using Euler discretization with 100 steps per year. In absolute terms, the MAEs between CGPR and MC prices are consistently small across all combinations of of d and m. The lowest MAEs for deep in-the-money options are achieved when $m = d$ and $m = 600$ or 800.For these configurations, the MRE drops to 0.74%, which is acceptable given that the MC benchmark also incurs estimation error due to the Euler discretization.

	Spread CGPR - MC prices					
	Absolute			Relative (ITM)		
m	$d = 3/5m$	$d = 3/4m$	$d = m$	$d = 3/5m$	$d = 3/4m$	$d = m$
400	0.3296	0.3927	0.4555	0.0132	0.0122	0.0117
600	0.3660	0.5516	0.3944	0.0125	0.0228	0.0074
800	0.3781	0.3906	0.3548	0.0140	0.0172	0.0074

Table 7: Absolute: MAEs between CGPR and MC prices. Relative (ITM) : MREs for in-the-Money (ITM) options ($S_0^{(1)} > K$).

Table 8 reports for $d = m$, computation times of CGPR and MC, errors, and standard deviations of CGPR prices, computed with formula (25). We observe that the a posteriori standard deviation falls to 0.46% when $m = 800$. In term of computational time, the CGPR is more efficient than MC simulations. The left plot of Figure 2 compares MC and CGPR values of

0.5-year call options and $\sqrt{V_0} = 24.76\%$. This visually confirms the accuracy of the CGPR in comparison to a standard MC method. The right plot of the same figure, shows the heatmap of MAEs between MC and CGPR call prices with respect to S_0 and $\sqrt{V_0}$. Figure 3 presents the heatmaps[4] for combinations of T with S_0 and of T with $\sqrt{V_0}$. Regardless of the combination of T, S_0 and $\sqrt{V_0}$, the spreads between MC and CGPR prices remain at acceptable levels. The highest spreads are obtained for combinations with either small or large values for S_0 and $\sqrt{V_0}$. We attribute this to the truncation of the domain $\mathcal{X}$ by a closed rectangular parallelepiped. The accuracy of the CGPR method decreases for combinations of state variables that are close to the boundaries delimiting this box.

	Spread CGPR - MC prices			Times (seconds)		
$m = d$	Absolute	Relative (ITM)	St.dev.	Opt.	CGPR	MC
400	0.4555	0.0117	0.0056	14.92	4.02	174.82
600	0.3944	0.0074	0.0046	33.29	7.46	176.15
800	0.3548	0.0074	0.0046	46.74	8.76	173.21

Table 8: Absolute: MAEs between CGPR and MC call prices. Relative (ITM) : MREs for in-the-Money options ($S_0 > K$). St.dev.: standard deviations of CGPR estimates. Opt. : optimization times for computing η_t, η_s and η_v. CGPR and MC: pricing times of 2400 options with CGPR and MC methods.

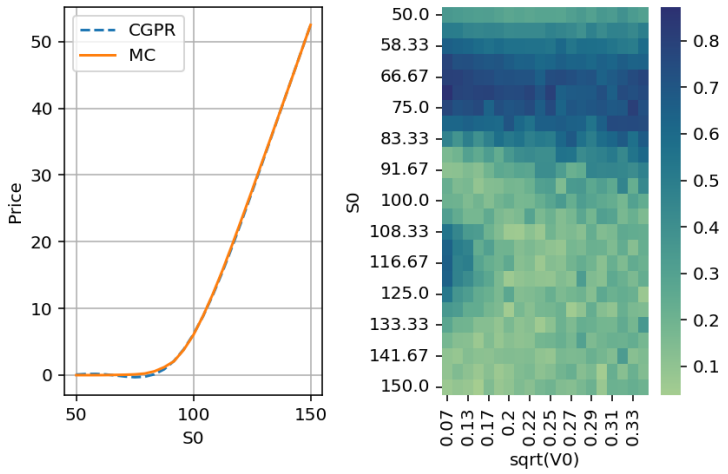


Figure 2: Left: MC and CGPR (kernel) 0.5-year call prices computed with $m = 800$, $d = 800$ and $\sqrt{V_0} = 24.76\%$. Right: heatmap of average absolute spreads between MC and CGPR call prices with respect to S_0 and $\sqrt{V_0}$ (all maturities).

We conclude this section by discussing the impact of σ_* on accuracy and stability of the CGPR prices. In practice, σ_* serves as a Tikhonov regularization parameter that prevents numerical instability when inverting the matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$, defined in Equation (15). If σ_* is too small, the CGPR algorithm may fail to invert $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$. Conversely, increasing σ_* introduces a bias in the CGPR prices. To better understand the impact of σ_*, we evaluate 3-years call options with S_0 and V_0 from Table 6 under various noise levels. Table 9 reports MAEs and MREs (ITM) between MC and CGPR prices for σ_* values ranging from 0.001 to 0.50. The table also presents

[4]The heatmaps present an average of errors with respect to 2 state variables. These errors are averaged over the other simulated state variables.

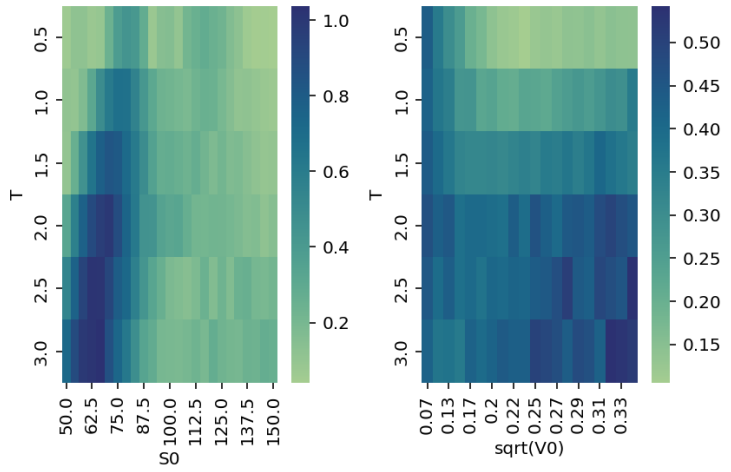


Figure 3: Right: heatmap of average absolute spreads between MC and CGPR (m=800, $d = 800$) call prices with respect to T and $\sqrt{V_0}$. Left: same heatmap but with respect to T and S_0.

the optimal hyper-parameters. We observe that the highest absolute error is obtained with the highest value of σ_*. When $\sigma_* = 0.001$, the Tikhonov regularization factor is not sufficient to prevent some numerical instability that generates higher errors mainly for out-of-the-money options. The minimal absolute error is obtained with $\sigma_* = 0.01$ and increases for larger σ_*. Figure 4 visualizes the gap between MC and CGPR prices as σ_* increases, highlighting that a high σ_* introduces a valuation bias.

	Spread CGPR - MC prices			Hyper-parameters		
σ_*	Absolute	Relative (ITM)	St. dev.	η_t	η_s	η_v
0.001	0.8354	0.0144	0.0037	1.49	45.41	0.12
0.01	0.4825	0.0060	0.0080	1.51	51.56	0.13
0.1	0.5213	0.0069	0.0416	1.55	64.19	0.14
0.25	1.0763	0.0212	0.0839	1.58	80.89	0.15
0.5	1.5301	0.0361	0.1176	2.84	73.95	0.99

Table 9: Absolute values and relative spreads for call options and in-the-Money (ITM) options ($S_0 > K$) per maturity. m=400 and σ_* ranges from 0.001 to 0.5. St.dev.: standard deviations of CGPR estimates.

7 Conclusions

This article presents a novel method for pricing European options using constrained Gaussian process regression (CGPR), leveraging the Feynman-Kac PDE that governs arbitrage-free derivative prices. By reframing the task of solving the Feynman-Kac partial differential equation as a constrained regression problem, this approach provides a robust and efficient alternative to traditional numerical methods. The method's reliance on sampling state variables from both the inner domain of the PDE and the terminal boundary, along with a Bayesian framework, ensures high accuracy and stability in option pricing.

One of the key advantages of this method is its avoidance of numerical differentiation, a common challenge in Physics-Inspired Neural Networks (PINNs). Instead, CGPR harnesses the inherent

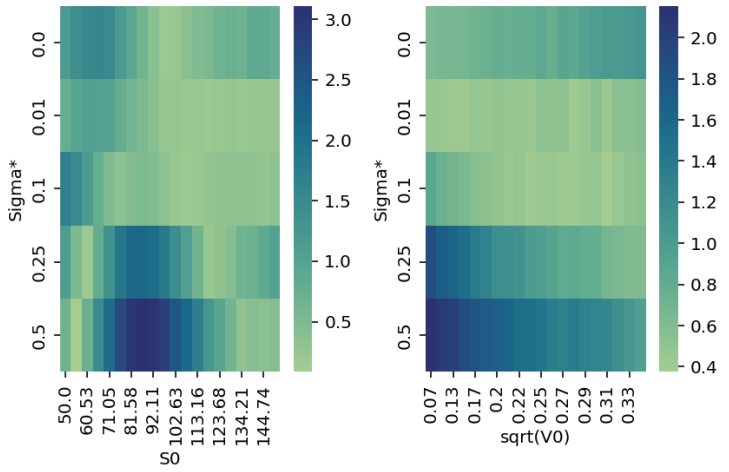


Figure 4: Right: heatmap of average absolute spreads between MC and CGPR (m=800, $d = 800$) 3 years call prices with respect to σ_* and $\sqrt{V_0}$. Left: same heatmap but with respect to σ_* and S_0.

properties of Gaussian processes to deliver a closed-form approximation of option prices. This approach not only enhances numerical robustness but also supports the inclusion of a Tikhonov regularization factor, further improving the numerical stability of the method.

The numerical illustrations, including the evaluation of European call options in the Black and Scholes and Heston models demonstrate the practical applicability and effectiveness of the proposed framework. The results indicate that CGPR achieves accuracy comparable to established techniques such as the Monte Carlo (MC) methods. This positions CGPR as a robust alternative for pricing derivatives, showcasing its potential in both theoretical and practical contexts.

Nevertheless, the CGPR presents a few drawbacks, such as the time required to optimize the kernel bandwidths. Similar to PINNs, CGPR necessitates large training datasets when addressing higher-dimensional problems, complicating the inversion of the Gram matrix $C\left(\mathring{\boldsymbol{X}}, \bar{\boldsymbol{X}}\right)$. Nevertheless, there exists numerical solutions for managing large datasets, such as discussed in Chapter 8 of Rasmussen and Williams (2006).

However, the CGPR framework offers a novel and efficient approach to derivative pricing. By addressing the limitations of existing methods and providing a robust solution to the Feynman-Kac PDE, this framework has the potential to inspire further research and development in this area. In particular, the CGPR can be extended for pricing American options with an iterative penalized method, as studied in Hainaut (2024 b). Another extension of CGPR is presented in Hainaut and Dupret (2025) who combine in this article, the CGPR with policy improvements for solving stochastic control problems.

This work paves the way for further research. For instance, we can consider other kernels or other dynamics for state variables. It is probably possible to apply our framework to jump or switching processes. In this case, the differential operator is non-linear but we can adapt the framework of Hainaut (2024 b) for developing an iterative method.

Acknowledgement

Donatien Hainaut thanks the FNRS (Fonds de la recherche scientifique) for the financial support through the EOS project Asterisk (research grant 40007517). The work of Frédéric Vrins is supported by the F.S.R.-FNRS (Fonds de la Recherche Scientifique) through the CDR grant J.0037.18 and by the ARC project NEMO (grant 18-23/089) of the Belgian Federal Science Policy Office.

Disclosure statement

No potential conflict of interest was reported by authors.

References

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

[2] Barber D., Wang Y. 2014. Gaussian processes for Bayesian estimation in ordinary differential equations. In Proceedings of the 31st International Conference on Machine Learning (ICML-14), pages 1485-1493. JMLR Workshop and Conference Proceedings.

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

[4] Black F., Scholes M. 1976. The Pricing of options and corporate liabilities. Journal of political economy, 81, 637-654.

[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] Cialenco I., Fasshauer G.E., Ye Q. 2012. Approximation of Stochastic Partial Differential Equations by a Kernel-based Collocation Method. International Journal of Computer Mathematics. Vol 89 (18), p. 2543-2561.

[7] Chen Y., Hosseini B., Owhadi H., Stuart A.M. 2021. Solving and Learning Nonlinear PDEs with Gaussian Processes. Journal of Computational Physics, 447, 110668.

[8] Cockayne J., Oates C., Ipsen I., Girolami M. 2019. A Bayesian conjugate gradient method (with discussion). Bayesian Analysis, 14(3), p. 937–1012.

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

[10] Cuomo S., Di Somma V., Sica F. 2020. A note on the numerical resolution of Heston PDEs. Ricerche di Matematica (2020) 69:501–508.

[11] De Spiegeleer J., Madan D.B., Reyners S., Schoutens W. 2018. Machine learning for quantitative finance: fast derivative pricing, hedging and fitting. Quantitative finance, 18 (10), 1635–1643.

[12] Düring B., Miles J. 2017. High-order ADI scheme for option pricing in stochastic volatility models. J. of Comp. and App. Math. 316, 109-121

[13] Dondelinger F. , Rogers S., Husmeier D. 2013. ODE parameter inference using adaptive gradient matching with Gaussian processes. In Sixteenth International Conference on Articial Intelligence and Statistics; AISTATS.

[14] 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, 57–72.

[15] Gonzalvez J., Lezmi E., Roncalli T., Xu J. 2019. Financial Applications of Gaussian Processes and Bayesian Optimization. arXiv:1903.04841.

[16] 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

[17] 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), p. 1015-1028.

[18] Hainaut D. 2023 b. Pricing of spread and exchange options in a rough jump-diffusion market. J. of Comp. and App. Math. 419, 114752.

[19] 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

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

[21] Hainaut D. 2024 b. American option pricing with model constrained Gaussian process regressions. UCLouvain LIDAM working paper. https://dial.uclouvain.be/pr/boreal/object/boreal:292665

[22] Hainaut D., Dupret J.L. 2025. Optimal control by policy improvements and constrained Gaussian process regressions. UCLouvain LIDAM working paper. https://drive.google.com/file/d/1iHK5nEF2TI5cdnAaxeopslS6GtH_wJud/view

[23] Jeanblanc M., Yor M., Chesney M. 2009. Mathematical methods for financial markets. Springer-Verlag London.

[24] Murphy K.P. 2012. Machine learning: a probabilistic perspective, MIT press.

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

[26] 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.

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

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

[29] Sarkka S. 2011. Linear operators and stochastic partial differential equations in Gaussian process regression. In Artificial Neural Networks and Machine Learning - ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part II, pages 151-158. Springer, Berlin, Heidelberg.

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

[31] Souto Arias L.A. , Cirillo P., Oosterlee C.W. 2025 The Heston–Queue-Hawkes process: A new self-exciting jump–diffusion model for options pricing, and an extension of the COS method for discrete distributions, J. of Comp. and App. Math. 454, 116177.

[32] 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) p. 119-156.

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

[34] Wilkens S., 2019. Machine learning in risk measurement: Gaussian process regression for value-at-risk and expected shortfall. Journal of risk management in financial institutions. 12 (4), p. 374-383.

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