Performance Enhancing Market Risk Calculation through Gaussian Processes Regression and Multi-Fidelity Modeling

N. Lehdili*, P. Oswald†, H. Nguyen‡

November 14, 2023

Abstract

With the implementation of new regulations (Basel IV, FRTB), market risk measurement has become an increasingly time-consuming task for banks. Indeed, the market risk measurement, specifically the practical implementation of the value-at-risk (VaR) and expected shortfall (ES) models, of a bank's trading portfolio involves intensive recalls of its pricing engine. Machine learning algorithms offer a solution to this challenge. In this study, we investigate the application of Gaussian processes regression (GPR) and its variant, namely multi-fidelity Gaussian processes regression (mGPR), powerful algorithms for both regression and classification, as alternatives for the pricing engine. More precisely, multi-fidelity modeling combines models of different fidelity levels, defined as the degree of detail and precision offered by a predictive models or simulation, to achieve rapid yet precise predictions. We use these regression models to approximate the prices of portfolios containing numerous equity derivatives, referring to canonical high-dimensional problems. In our numerical experiments conducted with limited data, we demonstrate that GPR and mGPR outperform both the traditional approaches used in banks and the well-known neural network model in term of pricing accuracy and computational efficiency of risk calculation.

Keywords: value-at-risk, expected shortfall, Basel III, FRTB, option pricing, multi-asset option, Gaussian processes regression, multi-fidelity model, machine learning, neural networks.

Mathematics Subject Classification: 91G20, 91B05, 62G08, 60G15, 65D05.

1 Introduction

Market risk refers to the risk of losses resulting from adverse movements in market prices, encompassing both on and off-balance sheet positions. From a regulatory standpoint, market risk emanates from positions held in a bank trading book, as well as from commodity and

*noureddine.lehdili@natixis.com. Expert Leader Market & Counterparty Risks Modelling.

†pascal.oswald@natixis.com. Expert Leader Market & Counterparty Risks Modelling.

‡hoangdung.nguyen@natixis.com. LPSM/Université Paris Cité. The research of H.D. Nguyen is funded by a CIFRE grant from Natixis.

foreign exchange risk positions within the entire balance sheet. Counterparty credit risk, on the other hand, is the risk associated with the possibility that the counterparty involved in a transaction may default before the transaction's cash flows are finally settled. Credit valuation adjustment (CVA) risk pertains to the risk of incurring losses due to fluctuations in CVA values triggered by changes in counterparty credit spreads and market risk factors affecting the prices of derivative transactions and securities financing transactions. Managing financial risks has always been a central activity in banking industry but since the 2008 global financial crisis it has increased in importance: there is a more and more focus on how financial risks are being assessed, monitored and managed. In the last three decades, extensive research efforts, spanning both academic and financial industry domains [Dowd, 2003, 2007; Culkin and Das, 2017; Saunders et al., 2021], have been dedicated to advancing banking practices and refining risk calculations to meet existing and forthcoming challenges.

The European Banking Authority (EBA) plays a crucial role in ensuring consistent implementation of the new regulations across the European Union, particularly with respect to the Fundamental Review of the Trading Book (FRTB) as part of the Basel III reforms [Basel Committee on Banking Supervision, 2013], by developing technical standards, guidelines, and reports. Specifically, the Basel Committee on Banking Supervision introduced the FRTB framework to enhance accuracy and consistency in trading book capital requirements. This initiative has compelled banks with trading operations to swiftly implement significant changes. Banks had to reinforce their models, data management, information technology infrastructure, and processes. Additionally, capital markets businesses had to optimize product structures, adjust hedging strategies, and reprice products. For some banks, unprofitable business lines due to FRTB led to downsizing or exiting their operations.

The impact of FRTB on banks is multifaceted, affecting methodologies, data management, processes, and systems. These changes pose significant challenges, particularly in areas like non-modellable risk factors and the development of default risk charge (DRC) methodologies. Simultaneously, the implementation of FRTB necessitates substantial upgrades in banking systems, including computational capacities and alignment of risk factors. FRTB also emphasizes the importance of market and reference data quality, demanding a coherent framework for sourcing and mapping positions accurately. These interconnected challenges highlight the intricate demands faced by banks in adapting to FRTB regulatory requirements. Banks must increase computational capacities, revalue positions daily, and develop new platforms to compute standardized capital charges: significantly higher calculation capacity will be required compared to that for current value-at-risk (VaR) calculations and banks estimate an increase in number of full revaluation runs by a factor of 5 to 10 [Basel Committee on Banking Supervision, 2019].

Simultaneously, the banking industry's response to these complex demands involves exploring the potential of machine learning, a subset of artificial intelligence, which has gained prominence for handling big data in various sectors. In finance, machine learning applications in , such as pricing, fraud detection, credit scoring, and portfolio management, have become increasingly significant [Financial Stability Board, 2017; Ferguson and Green, 2018; Liu et al., 2019]. This technological advancement offers a potential solution to the intricate challenges posed by FRTB, offering innovative ways to adapt and streamline banking processes in the face of regulatory requirements.

A recent study by the Financial Stability Board [2017] highlights the potential of machine learning in enhancing risk models by detecting complex patterns in large datasets. This

technology opens a new field of scientific research in regulation and risk management. While there is extensive research in market risk modeling, machine learning algorithms are not widely used in traditional quantitative finance and risk management yet. However, the neural network family, famous for its universal approximation and its customization flexibility, is applied to various financial tasks ranging from derivative pricing and hedging [Hutchinson et al., 1994; Buehler et al., 2019] to calibration [Horvath et al., 2021] and partial differential equations solving [Huré et al., 2020]. Zhang et al. [2017] show improvements in volatility forecasting using ANN models and innovative approaches like the GELM model, combining GARCH and Extreme Machine Learning [Huang et al., 2006] enhancing accuracy and efficiency in value-at-risk calculations. Ruf and Wang [2019] make a survey on the application of neural network models in finance quantitative. Supervised algorithms such as Gaussian Process regression is also applied in yield curve forecasting and derivatives pricing [Mu et al., 2018], reducing computation times. De Spiegeleer et al. [2018] apply GPR modeling to the fast derivatives pricing and hedging where the market is modelled with the limited set of parameters. They give the numerical test of European call and American put as well as barrier option in the framework of both variance gamma and Heston models. Ludkovski [2018]; Goudenège et al. [2021] price Bermudan options by Gaussian processes regression under the framework of Longstaff and Schwartz [2001] model. Hence, at each backward evaluation time step, one GPR is used for estimating the time value of the option. Crépey and Dixon [2019] study the approximation of European call in Heston model, they then extend the research to the use of multi-output Gaussian processes to predict various option prices underwritten on the same asset. In Lehdili et al. [2019] the GPR is trained to approximate the market value of the whole trading portfolio of different options on a common underlying. In a nutshell, the choice between neural networks and Gaussian processes regression depends on the specific task requirements. The latter is preferred when uncertainty analysis in predictions is crucial, or when there are limitations in the number of training points available. Despite these advancements, machine learning methods have not fully integrated into the financial risk measurement framework yet.

This paper follows the research flow of our preprint Lehdili et al. [2019], which yielded encouraging results for the VaR and ES calculation of the portfolio consisting of financial instruments written on a single asset. However, one drawback of the approach is the curse of dimensionality. As the dimension of the input space of the pricing function increases, the number of sampling points required to train the machine learning algorithm, used as a proxy pricing function, increases substantially. This becomes practically challenging when evaluating multi-asset options such as best-of, worst-of and basket options. The GPR techniques applied to fixed income derivatives like Bermudan swaptions also suffer from the curse of dimensionality. At this stage, the main difficulty lies in the limitation of training data, rather than the scalability of Gaussian processes model for learning large datasets (cf. Section 3.2). While the direct use of machine learning model may be efficient enough, some enhancement techniques can be applied to achieve optimal results. To perform the risk calculation of an interest rate derivatives portfolio, Ruiz and Zeron [2021] use dimensional reduction techniques, e.g. principal component analysis (PCA) or learning from diffusive risk factors [Abbas-Turki et al., 2023], to reduce the number of market risk factors before interpolating the portfolio value by Chebyshev tensors. When products are highly nonlinear and depend on many diffusive risk factors,e.g. multi-asset derivatives, the dimensional reduction techniques proposed in Ruiz and Zeron [2021] may not be applicable. This leads us to explore other enhancement techniques such as multi-fidelity modeling. Fidelity in modeling refers the degree of detail and precision offered by a predictive model or simulation. Logically, high-fidelity models provide highly

accurate outcomes but require substantial computational resources, i.e. the pricing engine in our financial case. Due to its high cost, the usage of high-fidelity models is often limited in practice. By contrast, low-fidelity models, resulting to less accurate outcomes, are cheaply accessible and often offer some useful insights of the learning problem. By combining different levels of fidelity models, multi-fidelity modeling exploit the information along these models and their correlation in order to provide quick yet accurate predictions. Fernández-Godino et al. [2016] do a survey of multi-fidelity modeling. Indeed, the latter can be used along with any machine learning models such as neural networks [Meng and Karniadakis, 2020; Li et al., 2020] or Gaussian process regression [Le Gratiet, 2013; Kennedy and O'Hagan, 2000; Brevault et al., 2020]. In the paper, we investigate the latter modelling, known as multi-fidelity Gaussian process regression (mGPR). While mGPR is well-known and largely applied in geostatistics and physics under the name cokriging[1], to the best of our knowledge we haven't seen any of its application in finance quantitative. This hence motivates us to study this modeling for the financial application, in which we consider the derivative pricing engine as high-fidelity model and lower fidelity models can be either its simpler counterparts or fast pricing engines of correlated products.

The rest of the paper is organized as follows. Section 2.1 introduces the principle of market risk according to the regulation. The computational challenge involved in the practical implementation of VaR and ES model is highlighted and the definition of the financial products of interest is provided. Section 3 reviews Gaussian processes regression and its application in the risk calculation of the trading portfolios. Section 4 describes multi-fidelity modeling and how the latter works with Gaussian processes regression. We depict some ideas of the application of multi-fidelity modeling in finance quantitative. In Section 5, we sketch the setup and the training specification for our numerical experiments, which are reported in Section 6. Section 7 concludes our findings and provides perspectives for future research.

2 Market Risk Assessment and Computational Challenge

This section introduces the principle of market risk calculation used by banks to monitor their risk that is as well demanded by regulatory. The computational challenge is highlighted and finally the scope of interested products is provided.

2.1 Market Risk Assessment

According to Basel Committee on Banking Supervision [2019], market risk is defined as *"the risk of losses (in on- and off-balance sheet positions) arising from movements in market prices"*. Market risks include default risk, interest rate risk, credit spread risk, equity risk, foreign exchange (FX) risk commodities risk, and so on. Regulatory separates market risk with other kinds of financial risk, for example, credit risks referring to the risk of loss resulting from counterparty[2] default or operational risk referring the risk of loss resulting from the failures of internal banking system. Under FRTB framework, banks can choose either the standardized

[1] see lecture notes in https://geostatisticslessons.com and references therein.

[2] the party with whom the bank makes engagement in the contract

measurement method (SMM) or the internal model-based approach (IMA) to determine market risks. The first approach, which can be easily implemented, is not interesting for banks, because it may overestimate their capital required to cover market risks. The second approach reflects more accurately the risk sensitivities and postulates better the economic risk carried by the banking balance sheet. However, to apply their proposed internal model, banks must gain the regulatory approval through a prudent procedure of backtesting.

Under the internal model approach the calculation of capital charge is based on tail risk measures, namely value-at-risk (VaR) and expected shortfall (ES), of the possible loss of the holding portfolio during a certain trading period. VaR and ES are measured at typically high confidence levels $\alpha = 97.5\%$ (for ES) or 99% (for VaR) in a short liquidity horizon (holding period), i.e. h =10 days. The value of the portfolio V_t at date t is the function of risk factors RF_t, denoted by $p_t(RF_t)$. Function $p_t(RF_t)$ corresponds to the pricing engine in the bank mentioned in the introduction. At date t, the possible loss of the portfolio (or profit and loss, denoted by PnL) during $[t, t+h]$ reads

$$\begin{aligned} L_t(h) = -(V_{t+h} - V_t) &= -(p_{t+h}(RF_{t+h}) - p_t(RF_t)) \\ &\approx -(p_t(RF_t + \delta RF_h) - p_t(RF_t)), \end{aligned} \tag{1}$$

where $V_{t+h} = p_{t+h}(RF_{t+h})$ the value of portfolio at date $t+h$ and the second line is the convention is used in then bank when the considered period h is small, e.g. one day or ten days. By the second formula, $L_t(h)$ in (1) can be interpreted as the potential loss of the portfolio when the risk factors RF_t are shocked by a quantity δRF_h. The value-at-risk at a confidence level α represents the minimum capital required to cover the market risk. Furthermore, the expected shortfall (ES), which is defined as the average loss given the loss exceeds the VaR, needs to be computed as well:

$$\begin{aligned} \mathrm{VaR}(L, \alpha, h) &= \min\{q \in \mathbb{R} : \mathrm{P}(L_t(h) \leq q) = \alpha\}, \\ \mathrm{ES}(L, \alpha, h) &= \mathrm{E}[L_t(h) | L_t(h) \geq \mathrm{VaR}(L, \alpha, h)] = \frac{1}{1-\alpha} \int_\alpha^1 \mathrm{VaR}(L, \gamma, h) d\gamma. \end{aligned} \tag{2}$$

Statistical methods for calculating (2) involve analytical, historical and Monte-Carlo approaches [Roncalli, 2020, Section 2.2 page 61]. In this paper we are interested in Monte-Carlo method described as follows. Numerically, (2) is computed by simulating a large number of possible L at time horizon h, then $\mathrm{VaR}(L, h, \alpha)$ is the α-percentile while $\mathrm{ES}(L, h, \alpha)$ is the empirical average of PnLs above the $\mathrm{VaR}(L, h, \alpha)$. Shocked scenarios of risk factors at date h can be generated using synthetic dynamic models which are calibrated by the historical market data. The market risk calculation procedure by Monte-Carlo is summarized in Algorithm 1, see also Hong et al. [2014] for a review of risk calculation by the Monte Carlo methods.

Remark 2.1. In FRTB, when calculating the risk for a trading book using the internal model-based approach, it is necessary to account for various liquidity horizons associated with different risk factors. To be more specific, for a considered portfolio, banks need to classify involved risk factors into five categories of liquidity horizons: 10, 20, 40, 60 and 120 days. FRTB requires banks to assess market risk for these liquidity horizons by considering shocks to risk factors at equal or higher horizons. For example, risk assessments for a 10-day horizon need to consider shocks to risk factors at all five horizons, while a 20-day horizon only needs to consider shocks at 20 days and above, and so on. This framework provides a more comprehensive and accurate assessment of market risk, but requires a complex and time-consuming calculation process.

Algorithm 1. Calculation of VaR and ES by full repricing approach

input : Calibrated diffusion model, pricing engine p_t, an actual vector of risk factors RF_t, a confidence level α, time step h and a large number N

output: $\text{VaR}(\alpha, h)$ and $\text{ES}(\alpha, h)$

1 Compute $p_t(RF_t)$
2 Simulate N scenarios of shock δS_h using calibrated diffusion model
3 Compute N corresponding prices $p_t(RF_t + \delta RF_h)$
4 Compute N scenarios of PnL: $L_t(h) = -\left(p_t(RF_t + \delta RF_h) - p_t(RF_t)\right)$
5 Compute VaR_α and ES_α by Monte-Carlo

2.2 Computational Challenge and Applications of Machine Learning

The market risk calculation appears to be doable by Algorithm 1. Nevertheless, we have not discussed yet about the computational complexity of the pricing engine. In banks, the pricing engine of a trading derivative portfolio involves various numerical algorithms depending on the nature of the portfolio. Some products or portfolios can be rapidly priced, such as using analytical formulas or accurately fast approximations. However, when financial models and/or products of the portfolio are complex as usually the case in practice, some heavily computational algorithms need to be referenced including Monte-Carlo schemes, tree methods, see for example [Crépey, 2013, Chapter 6 and 7]. The risk calculation using any of the algorithms mentioned above is referred to as the full repricing approach (i.e. full revaluation approach). The latter is the most accurate but very costly (for a portfolio of complex products) as the measurement of the tail risk, i.e. VaR and ES, of the portfolio needs to recall intensively the pricing engine.

In the context of implementing the VaR and ES calculation using Monte-Carlo simulation, banking institutions tend to avoid direct use of pricers, as it is time consuming and makes them inefficient. Certain banks prefer to use proxies based on sensitivity calculation and second-order Taylor developments to compute the profit and loss (PnL) of derivative portfolios [Britten-Jones and Schaefer, 1999], see Appendix B. Although these approaches offer performance gains, their precision is often questioned. Recently, alternative approaches have emerged that aim to strike a balance between performance and precision in risk calculation, such as value-at-risk and expected shortfall. Among these methods, the works of Crépey and Dixon [2019], and De Spiegeleer et al. [2018] explore the application of Gaussian process regression in finance, and Ruf and Wang [2019] discuss the use of neural networks in the same context. The common idea behind these studies is to use Gaussian processes, neural networks or other machine learning models as interpolation tools, also known as surrogate models, to estimate derivative prices. More recently, Ruiz and Zeron [2021] use Chebyshev polynomials as interpolators, combined with principal component analysis techniques (PCA), for financial risk calculation, aligning with the same line of though. Figure 1 illustrates the risk calculation process that employs these innovative machine learning approaches, involving only a few complete revaluations of a derivative portfolio.

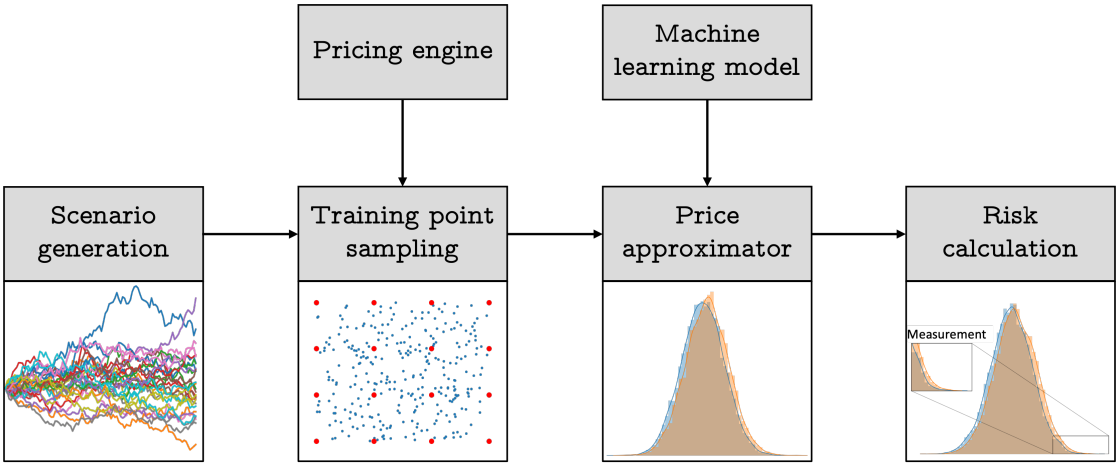

Figure 1: Risk calculation procedure.

2.3 Equity Options

In the remainder of this article, the objective is to evaluate and manage the market risk associated with a trading portfolio consisting of equity derivatives. These products are written on a stock whose the price at date t is denoted by S_t. Among these derivative products, vanilla options, barrier options, and American options on a single underlying asset can be mentioned. Within the framework of the Black-Scholes model, the values of these options are simply determined by analytical formulas depending on the underlying asset price S_t, except the American option values for which a numerical method, i.e. binomial tree, is required. The portfolio also include multi-underlying options such as basket options, best-of options and worst-off options. In this context, we consider d underlying assets whose the prices are denoted by $S_t = [S_t^1, \dots S_t^d]$ and governed by lognormal processes:

$$\begin{aligned} dS_t^j &= rS_t^j dt + \sigma^i S_t^j dW_t^j, \quad \text{for } j = 1 \dots d, \\ d\langle W^j, W^j \rangle_t &= \rho_{jj'} dt, \end{aligned} \tag{3}$$

where r is a risk-free rate, σ^i denotes the volatility of the i-th asset, and the W^i are d Brownian motions supposed with correlation matrix $\mathbf{R} = (\rho_{jj'})_{j,j' \in \{1,\dots,d\}}$. Without loss of generality, we assume the constant risk-free rate and volatilities in what follows.

Let $\eta(S_T)$ denote the value of the European multi-asset derivative at the maturity T, also called the payoff. Within the above framework, it is well-known that the value of the European multi-asset option V_t at evaluation date t is the mathematical expectation under the risk neutral probability of the discounted payoff, denoted by E^*,

$$p_t(S_t) = \mathrm{E}^*[e^{-r(T-t)}\eta(S_T)|S_t]. \tag{4}$$

Hereafter we are particularly interested in four options whose the payoff can be written as a vanilla option payoff ψ of the underlying $\tilde{S}_t$ given in Table 1,

$$\eta(S_T) = \psi(\tilde{S}_T), \tag{5}$$

where

$$\psi(\cdot) = \begin{cases} (\cdot - K)^+ & \text{for a call against strike } K, \\ (K - \cdot)^+ & \text{for a put against strike } K. \end{cases}$$

Table 1: Multi-asset options covered in our numerics.

Option	Geometric average	Basket	Best-of	Worst-of
Underlying	$\tilde{S}_T = \sqrt[d]{\prod_{j=1}^d S_T^j}$	$\tilde{S}_T = \sum_{j=1}^d \alpha_j S_T^j$ [1]	$\tilde{S}_T = \max_j S_T^j$	$\tilde{S}_T = \min_j S_T^j$

[1] with $\sum_{i=1}^d \alpha_j = 1$. The cases where $\alpha_j = \frac{1}{d}$ define arithmetic average options.

Assuming (5), (4) yields

$$p_t(S_t) = \mathrm{E}[e^{-r(T-t)}\psi(\tilde{S}_T)|S_t]. \tag{6}$$

Given (3), the price of geometric average options are fast referenced by the Black-Scholes formula, and the price of basket options can be accurately approximated by Black-Scholes formula thanks to the convention that the sum of lognormal variables is approximately a lognormal variable. Otherwise, the price of other options in Table 1 are evaluated by Monte-Carlo simulation.

3 Gaussian Processes Regression for Option Pricing

Gaussian processes regression (GPR) is a powerful machine learning model for interpolation (possibly modeling interpolation error). GPR is the canonical method for Bayesian modeling of spatial functions. The method is highly recommended in cases where data is sparse because of its generalization of Gaussian distributions from finite-dimensional vector spaces to infinite-dimensional functional spaces. Due to this principle, the GP is considered as a non-parametric model. The objective of this section is to briefly introduce Gaussian processes regression and its application to model financial derivative products. The more background of Gaussian processes regression can be referenced in Rasmussen and Williams [2006] and [Murphy, 2012, Chapter 15].

Throughout the paper, bold lowercase letters, e.g. $\mathbf{x}$, stands for vector notation, whereas bold uppercase letters, e.g. $\mathbf{X}$, are used for matrix notation.

3.1 Gaussian Processes Regression and Prediction

Let $(\Omega, \mathcal{A}, \mathrm{P})$ be a probability space, consider a random vector $\mathbf{x} : \Omega \to \mathcal{X} \subset \mathbb{R}^d$ for an integer d corresponding to the dimension of input, and a response variable $y \in \mathbb{R}$. In supervised learning, the goal is to identify the relation between input and response variable. To do this, we assume the following regression

$$y = f(\mathbf{x}) + \epsilon, \tag{7}$$

where $f : \mathcal{X} \to \mathbb{R}$ is the target function that we aim to learn and ϵ is i.i.d Gaussian white noise with zero mean and variance σ^2. Different from other regression models which usually assume a parameterization form of f, the Gaussian processes place directly a multivariate normal prior on the space of functions. Hence, the Gaussian process generalizes the Gaussian distribution from finite-dimensional vector spaces to infinite dimensional functional spaces. To

be implicit, for any set of input points $[\mathbf{x}_1, \ldots, \mathbf{x}_N]$ in $\mathcal{X}$, the random vector $[f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)]^\top$ is multivariate normally distributed.

More precisely, given a data set of N observations $\mathcal{D} = (\mathbf{X}, \mathbf{y}) = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, N\}$, where $\mathbf{x}_i$ is taken from set $\mathcal{X}$, $\mathbf{X}$ is the concatenated matrix in which each row represents to one observation of the row-vector input and $y_i \sim \mathrm{P}(y|f(\mathbf{x}_i))$. Let $\mathbf{f}$ denote $[f(\mathbf{x}_1) \ldots f(\mathbf{x}_N)]^\top$. In GPR, we have

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{X}, K_{\mathbf{X},\mathbf{X}}).$$

That implies

$$\mathbf{y} \mid \mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{X}, K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_N),$$

where $\mathbf{I}_N$ is the identity matrix of size N, $\boldsymbol{\mu}_\mathbf{X} = [\mu(\mathbf{x}_1), \ldots, \mu(\mathbf{x}_N)]^\top$ is a prior mean vector and $K_{\mathbf{X},\mathbf{X}}$ denotes the covariance matrix characterized by a kernel function k such that $K_{\mathbf{X},\mathbf{X}} = [k(\mathbf{x}_i, \mathbf{x}_{i'})]_{i,i'=1\ldots N}$.

Unless some extra knowledge about the prior mean function, for simplicity one can choose $\mu(\cdot) = 0$. Note that the convention is related only to the prior distribution and does not imply that the posterior distribution (the prediction) has zero mean. Similarly, one impose some knowledge of the target function on the choice of the kernel function. For example, the Exp-Sine-Squared kernel takes into account the periodic characteristic of function [Rasmussen and Williams, 2006, pp. 92]. The Matern kernel and the squared exponential kernel are the most frequently used stationary kernel [Rasmussen and Williams, 2006, pp. 83-84]. The latter reads

$$k\left(\mathbf{x}_i, \mathbf{x}_{i'}\right) = \frac{1}{\Gamma(\nu) 2^{\nu-1}} \left(\frac{\sqrt{2\nu}}{l} \left\| \mathbf{x}_i - \mathbf{x}_{i'} \right\| \right)^\nu K_\nu \left(\frac{\sqrt{2\nu}}{l} \left\| \mathbf{x}_i - \mathbf{x}_{i'} \right\| \right), \tag{8}$$

where Γ is the gamma function and K_ν is the modified Bessel function of the second kind. The smoothness of the approximate function is controllable through a positive parameter ν, e.g. (8) converges to the squared exponential kernel when ν tends to infinity. In addition, (8) is once differentiable when $\nu = 3/2$ and twice differentiable when $\nu = 5/2$. The length-scale parameter $l > 0$ can be a vector with the same number of dimension as the input X. Once the input X is standardized, the vector length-scale parameter modelling detects relevant variables by measuring their variance impact, see [Rasmussen and Williams, 2006, Section 5.1 and 6.6]. Other variants of Matern kernel as well as other kernel functions can be found in [Rasmussen and Williams, 2006, Section 4]. When y_i in $\mathcal{D}$ is observed directly from the groundtruth function f, i.e. $y_i = f(\mathbf{x}_i)$, the white noise ϵ in (7) and its variance σ in above equations can be removed and this corresponds to the interpolation with noise-free observations. In brief, Gaussian processes provide a lot of flexibility to integrate extra knowledge of the learning problem, apart from data set $\mathcal{D}$, into the prior model hypothesis $p(f)$. That is particularly useful for financial applications.

Once the Gaussian processes model is learned (see Section 3.2), the posterior distribution of the target at new inputs is predicted as follows. For matrix of N' new test points, denoted by $\mathbf{X}^* = [\mathbf{x}_1^*, \ldots, \mathbf{x}_{N'}^*]^T$, the joint prior distribution of the response reads

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_\mathbf{X} \\ \boldsymbol{\mu}_{\mathbf{X}^*} \end{bmatrix}, \begin{bmatrix} K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n & K_{\mathbf{X},\mathbf{X}^*} \\ K_{\mathbf{X}^*,\mathbf{X}} & K_{\mathbf{X}^*,\mathbf{X}^*} \end{bmatrix} \right), \tag{9}$$

where $\mathbf{f}^* = [f(\mathbf{x}_1^*), \ldots, f(\mathbf{x}_{N'}^*)]^\top$, $\boldsymbol{\mu}_{\mathbf{X}^*} = [\mu_{\mathbf{x}_1^*}, \ldots, \mu_{\mathbf{x}_{N'}^*}]^\top$, and $K_{\mathbf{X}^*,\mathbf{X}^*} = [k(\mathbf{x}_i^*, \mathbf{x}_{i'}^*)]_{i,i'=1\ldots N'}$ and $K_{\mathbf{X},\mathbf{X}^*} = K_{\mathbf{X}^*,\mathbf{X}}^\top$ are respectively covariance matrix of $\mathbf{X}^*$, and of $\mathbf{X}$ and $\mathbf{X}^*$ such that $K_{\mathbf{X},\mathbf{X}^*} = [k(\mathbf{x}_i, \mathbf{x}_{i'}^*)]_{i=1\ldots N, i'=1\ldots N'}$. The predictive (posterior) distribution of $\mathbf{f}^*$ is

$$\mathbf{f}^* \mid \mathbf{X}, \mathbf{y}, \mathbf{X}^* \sim \mathcal{N}(\mathrm{E}[\mathbf{f}^* \mid \mathbf{X}, \mathbf{y}, \mathbf{X}^*], \mathrm{Var}[\mathbf{f}^* \mid \mathbf{X}, \mathbf{y}, \mathbf{X}^*]), \tag{10}$$

where the posterior mean and variance are given as follows

$$\begin{aligned} \bar{\mathbf{f}}^* = \mathrm{E}[\mathbf{f}^* \mid \mathbf{X}, \mathbf{y}, \mathbf{X}^*] &= \mu(\mathbf{X}^*) + K_{\mathbf{X}^*,\mathbf{X}} \left[K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n\right]^{-1} \mathbf{y}, \\ \mathrm{Var}[\mathbf{f}^* \mid \mathbf{X}, \mathbf{y}, \mathbf{X}^*] &= K_{\mathbf{X}^*,\mathbf{X}^*} - K_{\mathbf{X}^*,\mathbf{X}} \left[K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n\right]^{-1} k(\mathbf{X}, \mathbf{X}^*). \end{aligned} \tag{11}$$

Remark 3.1. As we will discuss in Section 3.2, the repeated computation of the inverse matrix $\left[K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n\right]^{-1}$ in the learning procedure is a bottleneck of GPR. However, in the prediction phase (cf. (11)) the matrix inversion is required only one time and it is even already computed during the learning phase. In particular, the posterior mean is numerically computed using the following linear expression, for $i' = 1 \ldots N'$,

$$\bar{\mathbf{f}}_{i'}^* = \mu(\mathbf{x}_{i'}^*) + \sum_{i=1}^{N} \omega_i k(\mathbf{x}_i, \mathbf{x}_{i'}^*),$$

where $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_N]^\top = \left[K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n\right]^{-1} \mathbf{y}$.

Remark 3.2. The posterior mean can be also written by the following linear combination, without the loss of generality, suppose the zero mean prior,

$$\bar{\mathbf{f}}_{i'}^* = \sum_{i=1}^{N} \gamma_i y_i, \tag{12}$$

where $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_N]^\top = K_{\mathbf{X}^*,\mathbf{X}} \left[K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_n\right]^{-1}$. Equation (12) interprets the GP prediction of function f at a new point, say $\mathbf{x}_{i'}^*$, as a linear combination from a set of N observable points $(y_i)_{i=1\ldots N}$. However, the GPR is not a linear interpolation since the non-linear term is incorporated in γ_i. Moreover, in the general case where exact observations are not accessible, the model will interpolate from the noisy observations taking into account the variance of noise. Hence GPR is considered as a powerful non-linear interpolation tool.

3.2 Estimation of Model Parameters

The learning procedure of Gaussian processes model involves finding the best suited parameters of the mean function $\boldsymbol{\mu}_{\mathbf{X}}$, the covariance kernel function k, jointly denoted by $\boldsymbol{\theta}$ and the white noise variance σ. The set of GP parameters can be estimated either by maximum likelihood or maximum a posteriori, or cross-validation method (see [Rasmussen and Williams, 2006, Section 5.4] and [Murphy, 2012, Section 15.2.4]). By mean of computational facility, maximum likelihood is the the most used among these and this method is chosen to be a default method for the standard Gaussian processes. The criterion $\mathcal{L}$ that we aim to maximize is the marginal likelihood, also called by the (model) evidence reads

$$\begin{aligned} \mathcal{L}(\mu, \sigma, \boldsymbol{\theta}) = \log \mathrm{P}(\mathbf{y}|\mathbf{X}) = -\Big[&(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{X}})^\top \left(K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_N\right)^{-1} (\mathbf{y} - \boldsymbol{\mu}_{\mathbf{X}}) \\ &+ \log\det\left(K_{\mathbf{X},\mathbf{X}} + \sigma^2 \mathbf{I}_N\right)\Big] + const. \end{aligned} \tag{13}$$

Notice that the first term of (13) correspond to the data-fit can be seen as the the negative squared loss with a kernel-based weight in linear regression whereas the second term relates to the penalty of the model complexity (see [Rasmussen and Williams, 2006, Section 2.4 and 5.4.1]).

Maximizing the evidence (or minimizing the negative evidence) is usually solved by a global optimization method, namely BFGS [Nocedal and Wright, 1999, Chapter 6], as in Scikit-learn Python package or by any gradient-based method such as Adam [Kingma and Ba, 2015] using Gpytorch package. In any case, the numerical implementation must be done by full batch and involves reversing the covariance matrix of size $N \times N$ whose the complexity is $\mathcal{O}(N^3)$ (using Cholesky decomposition). Contrarily, the prediction step (cf. (11)) costs only $\mathcal{O}(N^2)$. Hence, even Gaussian processes model is a great interpolation tool, the classical model is not appropriate for high-dimensional problem that requires an important amount of training points N to explore the target surface. Many innovative approaches are proposed to tackle this issue and in general they are developed through the use of low-rank approximation (or possibly decomposition) of the covariance matrix [Titsias, 2009; Gardner et al., 2018] and/or the use of other learning methods such as (stochastic) variational inference [Blei et al., 2017; Hoffman et al., 2013; Titsias, 2009]. We refer the reader to the documentation of Gpytorch (Python, Gardner et al. [2018]) package for the detail as well as numerical implementation of these approaches.

3.3 Application to Derivative Portfolio Valuation

The application of Gaussian process regression is not restricted to evaluating mono-asset derivatives as illustrated below, but also applies to multi-asset derivatives in the same manner. However, when the number of underlying assets is high, one needs to use Monte-Carlo sampling instead of the grid point sampling technique used in low-dimensional problems. The regression model can directly approximate the value of the entire portfolio without any additional computational cost for learning and prediction, regardless of the portfolio size and complexity. This is hence favorable for risk calculation applications [Broadie et al., 2015, Remark 3].

When the considered portfolio V_t consists of a possibly large number of mono-asset derivatives, one can still use efficiently GPR by the decomposition technique provided in our previous preprint Lehdili et al. [2019]. More precisely, the portfolio may depend on a vector of d assets (risk factors) $S = [S^1, \dots, S^d]$ but each of its derivatives is underwritten on single stock. The portfolio price V_t is the conditional expectation of the total derivative payoffs given the stock asset vector S_t, hence, a function of S_t. By the nature of the considered portfolio, we are able to categorize V_t by risk factor

$$V_t = \sum_{j=1}^{d} V_t^j(S_t^j), \tag{14}$$

where V_t^j is the sub-portfolio corresponding to the j-th stock. We build the approximate of each V_t^j by one-dimensional Gaussian processes model. To this end, for $j = 1, \dots, d$, we sample N equidistant training points of S_t^j, say $\mathbf{s}^j = [s_1^j, \dots, s_N^j]$, in a determined range $[\underline{s}^j, \bar{s}^j]$, and the corresponding price $\mathbf{v}^j = [V_t^j(s_1^j), \dots, V_t^j(s_N^j)]$. Since this is the one-dimensional interpolation case and the price functions in finance are usually well-behaved, we only need to recall the pricing engine few times for sampling training data. For example, in the mono-asset portfolio

experiments [Lehdili et al., 2019], the GPR can reach the good results using only $N = 10$. Once the learning is done, the GP price of a sub-portfolio at a new point s^* is the predictive mean of the posterior distribution

$$\mathrm{E}[V_t^j(s^*) \mid \mathbf{s}^j, \mathbf{v}^j, s^*] = \sum_{i=1}^{N} \gamma_j V_t^j(s_i^j), \tag{15}$$

where γ_j is determined in Remark 3.2 (with the adaptation of notations). The GP price of the whole portfolio is then deduced by (14). The market risk calculation is straightforward by using the GPR pricing model instead of the pricing engine in the Monte-Carlo method (cf. Algorithm 1).

Remark 3.3. For a derivative which depends on several market risk factors such as asset volatility, dividend and interest rate and so on, the bank sometimes uses the Black model for the pricing engine. In this case, we can train bring it back to the two-dimensional interpolation problem, i.e. the discounted future price and the volatility, and efficiently apply GPR.

Remark 3.4. In the case of basket options, one can use the current price of the underlying asset $\tilde{S}_t$ formulated in (5) as the unique learning feature, leading to one-dimension Gaussian processes regression.

Remark 3.5. When the derivative depends on many risk factors, i.e. multi-asset products. The algorithm may require a large number of training data for a good approximation. In our financial application, the time to get pricing training points (sampled by costly pricing engine) is more relevant than the model learning difficulty of GPR mentioned in Section 3.2.

4 Multi-fidelity Gaussian Processes Regression

In the previous section, we describe how to apply GPR to interpolate the surface of price function using a few number of price points. However, the number of required points increases dramatically with the number of risk factors. Suppose that we have access to other kinds of information, e.g. a weak pricing engine based on a lower number of Monte-Carlo simulations or using few discretization time steps in the tree model, the question arises how to take into benefit of this knowledge in modeling. For this purpose, we investigate multi-fidelity Gaussian processes regression (mGPR) that allows us to learn the ground truth target from several information sources, each with a different level of fidelity. Any detail of this model can be found in Kennedy and O'Hagan [2000]; Le Gratiet [2013]; Brevault et al. [2020].

Assuming that we have s data sets, denoted by $\mathcal{D}_1 = (\mathbf{X}_1, \mathbf{y}_1), \ldots, \mathcal{D}_s = (\mathbf{X}_s, \mathbf{y}_s)$, sorted by increasing level of fidelity (i.e. $\mathcal{D}_s$ is the most accurate and expensive data whereas $\mathcal{D}_1$ is the most simplified and abundant data). In Kennedy and O'Hagan [2000], the autoregressive model is used to model the cross-correlation structure between different levels:

$$\begin{aligned} f_j(\mathbf{x}) &= \rho_{j-1} f_{j-1}(\mathbf{x}) + \xi_j(\mathbf{x}), \quad j = 2 \ldots s, \\ [\mathbf{y}_1]_i &= f_1(\mathbf{x}_i) + \epsilon, \quad \mathbf{x}_i \in \mathbf{X}_1, \end{aligned} \tag{16}$$

where ρ_{j-1} are scalar correlation parameters between $\mathbf{y}_{j-1}$ and $\mathbf{y}_j$, $\xi_j(\mathbf{x})$ measures the mismatch between $\mathbf{f}_j$ and $\rho_{j-1} f_{j-1}(\mathbf{x})$ and ϵ is the while noise of the lowest fidelity model. In mGPR, ξ_j is supposed independent of $f_{j-1}, \ldots, f_1$. The goal of the model is to lean f_s which is the relationship between the input and the highest fidelity response.

4.1 Two-fidelity Gaussian Process Regression Model

With the interest of the financial application (see Section 4.2 for an illustrative example), we focus on the mGPR with two fidelity levels, detailed as follows. Let assume a data set of N_l low-fidelity points, denoted by $\mathcal{D}_l = (\mathbf{X}_l, \mathbf{y}_l) = (\mathbf{x}_i, y_{l_i})_{i=1\dots N_l}$ and a second set of $N_h (\ll N_l)$ high-fidelity points, denoted by $\mathcal{D}_h = (\mathbf{X}_h, \mathbf{y}_h) = (\mathbf{x}_i, y_{h_i})_{i=N_l-N_h+1\dots N_l}$, such that N_h last points of low-fidelity data coincides with N_h high-fidelity data, i.e.

$$\mathbf{X}_l = \begin{pmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_{N_l-N_h+1}^\top \\ \vdots \\ \mathbf{x}_{N_l}^\top \end{pmatrix} \qquad \text{and} \qquad \mathbf{X}_h = \begin{pmatrix} \mathbf{x}_{N_l-N_h+1}^\top \\ \vdots \\ \mathbf{x}_{N_l}^\top \end{pmatrix} \tag{17}$$

The multi-fidelity modeling considers the following regression functions

$$\begin{cases} f_h(\mathbf{x}) = \rho f_l(\mathbf{x}) + \xi(\mathbf{x}), & \mathbf{x} \in \mathcal{X}, \\ f_l(\mathbf{x}) \perp \xi(\mathbf{x}) \mid \mathbf{x}, \end{cases} \tag{18}$$

where the parameter ρ describes the correlation of the high- and low-fidelity data, f_l is a Gaussian processes regression of $\mathbf{y}_l$ against $\mathbf{X}_l$ and δ is the second Gaussian process conditionally independent with the former. In what follows, we consider GP model with constant prior mean [Forrester et al., 2007; Kennedy and O'Hagan, 2000] parameterized by coefficient β, Marten kernel covariance function $k_{\boldsymbol{\theta}}$ with $\nu = 2.5$ and parameterized by $\boldsymbol{\theta}$ as per (8), and a multiplicative form of the marginal variance σ^2. In particular, the prior model is

$$\mathbf{f}_l \sim \mathcal{N}(\beta_l, \sigma_l^2 K_{\boldsymbol{\theta}_l}(\mathbf{X}_l, \mathbf{X}_l)), \tag{19}$$

and

$$\boldsymbol{\xi} \sim \mathcal{N}(\beta_h, \sigma_h^2 K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)), \tag{20}$$

where $\mathbf{f}_l$ and $\boldsymbol{\xi}$ are mean vectors $[f_l(\mathbf{x}_1) \dots f_l(\mathbf{x}_{N_l})]^\top$ and $[\xi(\mathbf{x}_{N_l-N_h+1}) \dots \xi(\mathbf{x}_{N_l})]^\top$, and the covariance matrices $K_{\theta_{l(h)}}(\mathbf{X}_{l(h)}, \mathbf{X}_{l(h)}) = [k_{\theta_{l(h)}}(\mathbf{x}_i, \mathbf{x}_{i'})]_{\mathbf{x}_i, \mathbf{x}_{i'} \in \mathbf{X}_{l(h)}}$. Hence the multi-fidelity model is characterized by the parameters of the first and second Gaussian processes: $(\beta_l, \sigma_l, \boldsymbol{\theta}_l)$ and $(\beta_h, \sigma_h, \boldsymbol{\theta}_h)$, and the correlation parameter ρ.

4.2 Illustrative Application of Multi-Fidelity Model in Option Pricing

For the pricing and risk calculation problem, our proposed approach is to consider the pricing engine as the high fidelity model, while the less precise price values as lower fidelity data. For instance, if the pricing engine involves Monte-Carlo simulation of $100,000$ paths, we can sample weak price values calculated by only, e.g. 100 paths as low-fidelity data (see Section 6.2 for numerical experiments). In the American option pricing case, where the accurate price value is calculated using a binomial tree with 100 time steps, the weaker price can be generated by a sparser tree with only, e.g. 10 time steps. Conversely, if the target is American option, which is expensively computed by tree model, its European counterpart, which is quickly evaluated using the Black-Scholes formula, can serve as lower fidelity data.

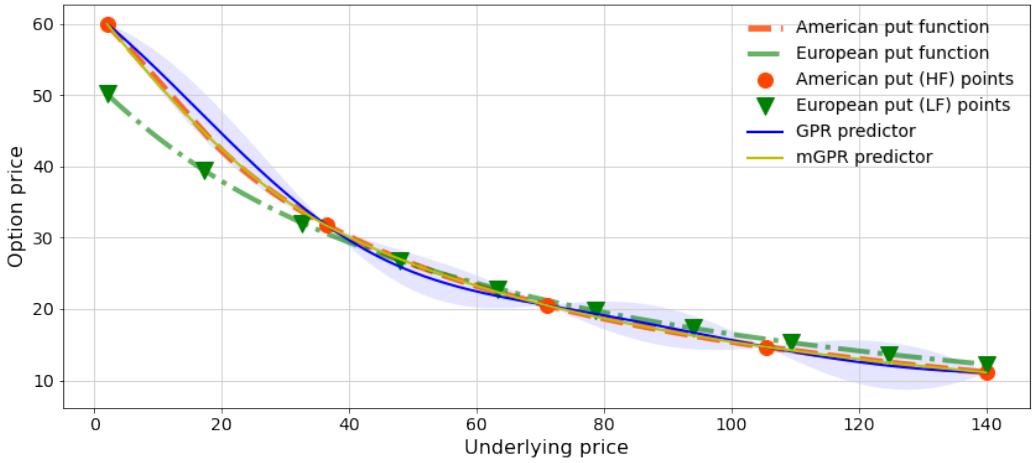

Figure 2: American call prices predicted by GPRs learned from 5 points. Red circles represent training points in GPR and high-fidelity points in mGPR, and green inverted triangles are low-fidelity points in mGPR. The light blue and light yellow (very close to yellow curve) regions depict the uncertainty of the GPR and mGPR predictions respectively.

Remark 4.1. Barone-Adesi and Whaley [1987] propose an analytic approximation of an American option which is equal to the value of its European counterpart plus an add-on or their intrinsic values under with a switch condition detailed in Appendix A. Hence, we can use this approximation as a competitor for the mGPR estimate which will model the American option price as the sum of the European option price and a sophisticated Gaussian process term.

Figure 2 demonstrates the numerical test of the American option pricing, which also helps to illustrate the use of multi-fidelity Gaussian processes regression. The configuration of this experiment is detailed in Section 5 and 6.1. In particular, the GPR is trained with only 5 American put prices, while the mGPR integrates additionally 10 European prices as low-fidelity data. All training points are drawn from a regular grid of the underlying asset within an interval[3] $[1, 140]$, and test points are uniformly sampled within the same interval. We notice a significant mismatch of the GPR estimates (cf. blue curve) and a satisfactory fit of the mGPR estimates (cf. yellow curve) to the American put price computed by binomial tree of 100 time steps. Additionally, mGPR provides smaller prediction uncertainty represented by the light yellow interval, which closely aligns with the yellow curve (visually indistinguishable) in Figure 2. For comparison, we also implement a second standard GPR, learning the discrepancy between American and European prices, and the approximation introduced in Barone-Adesi and Whaley [1987] (see Appendix A). Table 2 shows that mGPR estimates are the most accurate among them.

[3]This interval is larger than the one we use in Section 6.1.

Model	GPR	GPR with control variate	BW approximation	mGPR
MAE	0.6848	0.3199	0.2859	0.1367

Table 2: Out-of-sample mean absolute error (MAE) of the predictors against the American put price computed by binomial tree of 100 time steps. The GPR with control variate learn the discrepancy between American price and its European counterpart. The BW approximation refers to the Barone-Adesi and Whaley [1987] approach.

4.3 Conditional Distribution of the Estimate

We describe now the posterior mean and variance of the prediction once the model is completely learned. Following the demonstration in [Forrester et al., 2007, Appendix A], for a new point $\mathbf{x}^* \in \mathcal{X}$, the prediction of the high-fidelity model follows normal distribution:

$$f_h(\mathbf{x}^*) \mid \mathbf{f}_l = \mathbf{y}_l, \mathbf{f}_h = \mathbf{y}_h, (\beta_l, \sigma_l, \boldsymbol{\theta}_l), (\rho, \beta_h, \sigma_h, \boldsymbol{\theta}_h) \sim \mathcal{N}(m_h(\mathbf{x}^*), s_h^2(\mathbf{x}^*)), \tag{21}$$

with mean and variance functions:

$$\begin{aligned} m_h(\mathbf{x}^*) &= \beta + K(\mathbf{x}^*)^\top \mathbf{K}^{-1}(\mathbf{y} - \beta), \\ s_h^2(\mathbf{x}^*) &= \rho^2\sigma_l^2 + \sigma_h^2 - K(\mathbf{x}^*)^\top \mathbf{K}^{-1} K(\mathbf{x}^*), \end{aligned} \tag{22}$$

where

$$\beta = \frac{\mathbf{1}^\top \mathbf{K}^{-1}\mathbf{y}}{\mathbf{1}^\top \mathbf{K}^{-1}\mathbf{1}}, \qquad \text{with} \qquad \mathbf{y} = \begin{bmatrix} \mathbf{y}_l \\ \vdots \\ \mathbf{y}_h \end{bmatrix}.$$

In addition, $K(\mathbf{x}^*)^\top$ is the covariance function between $f_h(\mathbf{x}^*)$ and $[f_h(\mathbf{x}_1) \dots f_h(\mathbf{x}_{N_l}), f_h(\mathbf{X}_h)]$

$$K(\mathbf{x}^*)^\top = \left[\rho\sigma_l^2 k_{\boldsymbol{\theta}_l}(\mathbf{x}^*, \mathbf{X}_l) \quad \rho^2\sigma_l^2 k_{\boldsymbol{\theta}_l}(\mathbf{x}^*, \mathbf{X}_h) + \sigma_h^2 k_{\boldsymbol{\theta}_h}(\mathbf{x}^*, \mathbf{X}_h)\right],$$

and the covariance matrix $\mathbf{K}$ of vector $(\mathbf{f}_l, \mathbf{f}_h)$ reads

$$\mathbf{K} = \begin{pmatrix} \sigma_l^2 K_{\boldsymbol{\theta}_l}(\mathbf{X}_l, \mathbf{X}_l) & \rho\sigma_l^2 K_{\boldsymbol{\theta}_l}(\mathbf{X}_l, \mathbf{X}_h) \\ \rho\sigma_l^2 K_{\boldsymbol{\theta}_l}(\mathbf{X}_h, \mathbf{X}_l) & \rho^2\sigma_l^2 K_{\boldsymbol{\theta}_l}(X_h, X_h) + \sigma_h^2 K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h) \end{pmatrix}. \tag{23}$$

We can notice that the computation of (22) involves reversing the massive covariance matrix $\mathbf{K}$ of size $(N_l + N_h) \times (N_l + N_h)$. To reduce the computational complexity, one can apply the Schur complement of block matrix $\mathbf{K}$ which is thus presented by Proposition 3.1 in Le Gratiet [2013].

Proposition 4.2. *By sorting the input data such that* $\mathbf{X}_l = (\mathbf{X}_l \backslash \mathbf{X}_h, \mathbf{X}_h)$ *(cf.* (17)*) and with all above notations, the inverse matrix of* $\mathbf{K}$ *in* (23) *has the form:*

$$\mathbf{K}^{-1} = \begin{pmatrix} \sigma_l^{-2} K_{\boldsymbol{\theta}_l}(\mathbf{X}_l, \mathbf{X}_l)^{-1} + \begin{pmatrix} 0 & 0 \\ 0 & \rho^2\sigma_h^{-2} K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1} \end{pmatrix} & -\begin{pmatrix} 0 \\ \rho\sigma_h^{-2} K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1} \end{pmatrix} \\ -\begin{pmatrix} 0 & \rho\sigma_h^{-2} K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1} \end{pmatrix} & \sigma_h^{-2} K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1} \end{pmatrix}. \tag{24}$$

Instead of performing the inversion matrix in (23) which has a complexity of $\mathcal{O}((N_l+N_h)^3)$, the computation of $\mathbf{K}^{-1}$ in (24), involving the inverse of $K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}$ and $K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}$, has a much lower complexity of $\mathcal{O}(N_l^3+N_h^3)$. By substituting (24) into (22) and doing some calculus, one can simplify the above formulas as follows

$$\beta = \frac{\sigma_l^{-2}\mathbf{1}^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}\mathbf{y}_l + (1-\rho)\sigma_h^{-2}K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}(\mathbf{y}_h - \rho[y_{l_{N_l-N_h+1}}\cdots y_{l_{N_l}}]^T)}{\sigma_l^{-2}\mathbf{1}^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}\mathbf{1} + (1-\rho)^2\sigma_h^{-2}\mathbf{1}^\top K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}\mathbf{1}}, \quad (25)$$

yielding the posterior mean and variance

$$\begin{aligned} m_h(\mathbf{x}^*) &= \beta + \rho K_{\boldsymbol{\theta}_l}(\mathbf{x}^*,\mathbf{X}_l)K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}(\mathbf{y}_l-\beta) + K_{\boldsymbol{\theta}_h}(\mathbf{x}^*,\mathbf{X}_h)K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}(\mathbf{y}_h-\beta), \\ s_h(\mathbf{x}^*) &= \sigma_h^2 + \rho\sigma_l^2 - \sigma_h^2 K_{\boldsymbol{\theta}_h}(\mathbf{x}^*,\mathbf{X}_h)K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{x}^*) \\ &\quad - \rho^2\sigma_l^2 K_{\boldsymbol{\theta}_l}(\mathbf{x}^*,\mathbf{X}_l)K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{x}^*). \end{aligned} \quad (26)$$

4.4 Bayesian Estimation of Model Parameters

Due to the conditional independence assumption in the second line of (18), the first set $(\beta_l,\sigma_l,\boldsymbol{\theta}_l)$ can be separately learnt by, for example, the maximum likelihood method using data set $\mathcal{D}_l$. While the parameters of the second Gaussian processes $(\beta_h,\sigma_h,\boldsymbol{\theta}_h)$ and the correlation parameter ρ need to be jointly estimated to carry out the idea of formulation (18) [Le Gratiet, 2013, Section 3.3.2]. Similarly to (13), the parameters of the low-fidelity model are estimated by maximizing the following log-likelihood

$$-\frac{1}{2}\left[N_l\log(\sigma_l^2) + \log(\det(K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)) + \frac{(\mathbf{y}_l-\beta_l)^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}(\mathbf{y}_l-\beta_l)}{\sigma_l^2}\right] \quad (27)$$

Differentiating the above equation w.r.t. β_l and σ_l^2 and solving it at 0 yield

$$\hat{\beta}_l = \frac{\mathbf{1}^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}\mathbf{y}_l}{\mathbf{1}^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}\mathbf{1}}, \quad (28)$$

and

$$\hat{\sigma}_l^2 = \frac{1}{N_l}(\mathbf{y}_l-\hat{\beta}_l)^\top K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)^{-1}(\mathbf{y}_l-\hat{\beta}_l). \quad (29)$$

By substituting (28) and (29) into (27), we have the following optimization problem

$$\hat{\boldsymbol{\theta}}_l = \arg\min_{\boldsymbol{\theta}_l}\left[N_l\log(\hat{\sigma}_l^2) + \log(\det(K_{\boldsymbol{\theta}_l}(\mathbf{X}_l,\mathbf{X}_l)))\right]. \quad (30)$$

To learn the second Gaussian process model, we define

$$\mathbf{d} = \mathbf{y}_h - \rho\begin{pmatrix} y_{l_{N_l-N_h+1}} \\ \vdots \\ y_{l_{N_l}} \end{pmatrix}. \quad (31)$$

The log-likelihood of the model reads

$$-\frac{1}{2}\left[N_h\log(\sigma_h^2) + \log(\det(K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)) + \frac{(\mathbf{d}-\beta_h)^\top K_{\boldsymbol{\theta}_h}(\mathbf{X}_h,\mathbf{X}_h)^{-1}(\mathbf{d}-\beta_h)}{\sigma_h^2}\right], \quad (32)$$

yieling the MLEs of

$$\hat{\beta}_h = \frac{\mathbf{1}^\top K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1}\mathbf{d}}{\mathbf{1}^\top K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1}\mathbf{1}}, \tag{33}$$

and

$$\hat{\sigma}_h^2 = \frac{1}{N_l}(\mathbf{d} - \hat{\beta}_h)^\top K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h)^{-1}(\mathbf{d} - \hat{\beta}_h). \tag{34}$$

To compute the above parameters, one needs to estimate $\hat{\boldsymbol{\theta}}_h, \rho$ by solving the following optimization

$$\hat{\boldsymbol{\theta}}_h, \rho = \arg\min_{\boldsymbol{\theta}_h, \rho} \left[N_h \log(\hat{\sigma}_h^2) + \log(\det(K_{\boldsymbol{\theta}_h}(\mathbf{X}_h, \mathbf{X}_h))) \right]. \tag{35}$$

Note that (35) can be directly solved by numerical methods mentioned in Section 3.2 since the inversion of the small matrix of size $N_h \times N_h$ is not problematic. While the same procedure may not be applied for system (30) when the number of low-fidelity N_l is large. We then present in the next subsection the sparse Gaussian processes model for low-fidelity model.

5 Experiment Design and Model Specification

In this section, we present numerical tests dedicated to evaluating the accuracy and performance of the algorithms studied earlier in this paper. More specifically, these experiments are conducted on two trading portfolios (cf. Section 6.1 and 6.3), interspersed by an examination of individual multi-asset calls (cf. Section 6.2). The first one concerns a portfolio of single underlying derivatives, including plain vanilla (call and put), barrier and American options. The second one consists of multi-asset options, such as geometric, basket, best-of- and worst-of- options. The comparison is based on benchmark methods and scores explained below. We conclude this section by specifying the model of the implemented algorithms.

First, we analyze the case of the single-asset options portfolio described in our preprint Lehdili et al. [2019]. This portfolio, with a time maturity of ten years, consists of 100 distinct calls and puts issued on each of the four considered underlying assets. In particular, each asset is used to write 10 vanilla, 10 barrier and 5 American options with their strikes randomly generated from 60 to 140. More details about market and product parameters in the portfolio are presented in Tables 1 and A.1 of the preprint. We then evaluate the price of each multi-asset option, as indicated in Table 1. Each option expires after two years and is based on the same five assets. The actual values of geometric call options can be calculated using the Black-Scholes formula and serve as a reference. In the case of high dimensionality, the considered portfolio comprises 500 derivatives, depending on the performance of 20 stocks, generating a total of 20 market risk factors. Each derivative is characterized by various parameters, including a random selection of underlying assets from 2 to 20, a strike price ranging from 80 to 160, and a number of options from 100 to 500. It is noteworthy that each derivative can be either a call or a put option among the four types introduced in Section 2.3. The portfolio maturity is set at 5 years, meaning that all its derivatives have a maturity ranging from 6 months to 5 years. In the Black-Scholes model framework, we choose a risk-free rate of 1%. Initial asset prices are arbitrarily chosen within a range of 100 to 120, and the volatility of these assets is set to vary between 0.2 and 0.5 with a random correlation matrix, refer to Table

S_0^1	103.79	S_0^{11}	115.33
S_0^2	100.41	S_0^{12}	102.28
S_0^3	109.73	S_0^{13}	118.77
S_0^4	115.22	S_0^{14}	118
S_0^5	108.82	S_0^{15}	103.47
S_0^6	110.19	S_0^{16}	113.28
S_0^7	100.27	S_0^{17}	100.43
S_0^8	110.23	S_0^{18}	102.45
S_0^9	119.78	S_0^{19}	106.64
S_0^{10}	117.87	S_0^{20}	103.32

Table 3: Initial stock prices.

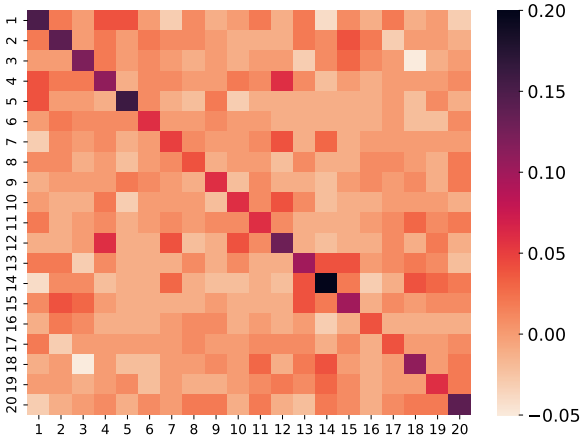

Figure 3: Heatmap of the stock covariance matrix.

3 and Figure 3. Finally, we examine the pricing and risk calculation problem for different time horizons, namely one day, ten days, and one month. At longer horizons, the magnitude of shocks and their volatilities increase, leading to a significant challenge in adapting the number of interpolation points. Financially, the price function becomes less linear, and the tail of the potential loss distribution becomes heavier.

5.1 Benchmarking

For the numerical implementation of market risk metrics calculation, the Monte-Carlo approach with $N = 100,000$ paths is utilized for any case. This level is chosen to ensure a satisfactory level of convergence of high quantiles estimation. However, a limited number (N_{train}) of these prices is chosen to train the machine learning models.

Regarding the exact pricing engine, the value of vanilla and barrier options in the mono-asset derivatives portfolio are calculated by analytical formulas, while the binomial tree with 100 time steps is used to evaluate American options. For multi-asset options, the exact pricing engine is defined by a Monte-Carlo simulation of $100,000$ paths, except for geometric options whose the price can be computed by Black-Scholes formula.

We implement the sensitivity-based pricing approximation (SxS) in Appendix B. To get sensitivities data, we shock relatively 0.1% of each risk factor and use the central finite difference, yielding totally 40 portfolio evaluations for the first order sensitivity-based approach. The second order sensitivities are numerical unstable and lead to a worse approximation of the price. Hence they are not considered in our numerics.

Regarding machine learning methods, we implement a two-hidden-layer neural network[4] (NN) and two Gaussian process models introduced in Section 3 (GPR) and 4 (mGPR). In

[4]see Appendix C.

the latter, the high-fidelity data consists of the $100,000$ Monte-Carlo paths prices, while the low-fidelity comprises weak prices based on 100 MC paths in Section 6.2 and 500 MC paths in Section 6.3. For measuring the pricing mismatch, we use the mean absolute percentage error (MAPE) over $N = 100,000$ scenarios in the VaR and ES computation:

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{p}_0(S_h) - p_0(S_h)|}{p_0(S_0)}, \tag{36}$$

where $p_0(S_h)$ and $\hat{p}_0(S_h)$ are respectively actual price (i.e. analytical or Monte-Carlo price) and predicted price (e.g. GPR approximative price), and $p_0(S_0)$ is the actual price without shock. The MAPE evaluating the average relative error of the estimate and the true observation is a common choice for benchmarking in finance. For risk assessment purpose, we measure the error of value-at-risk and expected shortfall calculated based on surrogate models (i.e. $\widehat{\text{VaR}}(\cdot, h, \alpha)$ and $\widehat{\text{ES}}(\cdot, h, \alpha)$) against the full revaluation ones calculated based on actual prices (i.e. $\text{VaR}(\cdot, h, \alpha)$ and $\text{ES}(\cdot, h, \alpha)$). For the relative measure, the error is then standardised by today price $p_0(S_0)$

$$\text{err. VaR}\,\alpha = \frac{\left|\widehat{\text{VaR}}(\cdot, h, \alpha) - \text{VaR}(\cdot, h, \alpha)\right|}{p_0(S_0)}, \quad \text{err. ES}\,\alpha = \frac{\left|\widehat{\text{ES}}(\cdot, h, \alpha) - \text{ES}(\cdot, h, \alpha)\right|}{p_0(S_0)}. \tag{37}$$

Finally, the computation time is also reported to compare the resource efficiency.

5.2 Model Specification

For neural network model, we train a two-hidden-layer Relu network of 50 hidden units (cf. Appendix C). We observe numerically that more complex architectures do not significantly improve the approximation performance. Network parameters are calibrated by the stochastic gradient descent algorithm, i.e. Adam optimizer [Kingma and Ba, 2015], using 2000 epochs. The learning rate is set at 0.01 and each batch takes one fifth of the training data size.

The standard Gaussian processes regression is trained using the scikit-learn Python package. We set a zero mean prior distribution and use Matern kernels with $\nu = 2.5$ (see (8)) to model the covariance matrix. A single length-scale is selected; however, the input is normalized by subtracting its empirical mean and then dividing by its empirical standard error. The optimization is restarted 5 times for the numerical stability. To implement multi-fidelity Gaussian processes regression, we use emukit package [Paleyes et al., 2023] which is built upon the Gpy framework developed by machine learning researchers from the University of Sheffield. The model configuration is either set as the standard Gaussian process or left at its default. The optimization is rerun 5 times.

6 Numerical Results

The main aim of this section is to provide a numerical analysis of algorithms based on Gaussian process regression (GPR) and the multi-fidelity model (mGPR) when applied to the repeated valuation and market risk calculation of an equity trading portfolio. It is worth emphasizing that a key advantage of the proposed new methods is that valuation and risk assessment are

performed at the portfolio level, rather than on a deal-by-deal basis, which is the case with traditional approaches.

In Section 6.1, we examine the accuracy and efficiency of Gaussian process regression (GPR) when applied to the valuation and market risk calculation (VaR and ES) for the aforementioned first portfolio, composed of single-underlying options. Section 6.2 focuses on the quantitative evaluation of both GPR and mGPR approaches in the context of pricing and risk management of a single option where the payoff is either a worst/best option or a basket option, or a geometric arithmetic average introduced in Table 1. In the case of a basket option with a geometric average a closed-form formula is available to value this option, enabling a comparison between the two algorithms, GPR and mGPR, as well as the more explicit classical method. Finally, Section 6.3 address the third application dedicated to pricing and market risk calculation of the previously mentioned second portfolio, consisting of multi-underlying options.

6.1 Mono-asset Options Portfolio Case

Figures 4-5 and Table 4 outline the performance of Gaussian Process Regression (GPR) in the full revaluation and risk calculation of a mono-asset portfolio. We use the sub-portfolio techniques outlined in Section 3.3 to achieve this. The accuracy of the GPR techniques is evident in both pricing and risk calculation. For this mono-asset experiment, we adopt a grid point sampling technique, similar to the method described in the preprint by Lehdili et al. [2019]. However, the training stock prices were constrained within a specific range: the minimum and maximum values are determined by adjusting the (log) initial values of the stocks. Specifically, the range for i-th stock is set as $\left[S_0^i \exp\left((r - \frac{(\sigma^i)^2}{2})h - 3\sqrt{h}\sigma^i\right), S_0^i \exp\left((r - \frac{(\sigma^i)^2}{2})h + 3\sqrt{h}\sigma^i\right)\right]$, where S_0^i, σ^i, r and h denote respectively the initial stock price, volatility, interest rate and time step.

In Figures 4-5, a precise fit of the price function is achieved when employing 5 points or more for training each sub-portfolio. Noteworthy is the convergence of risk measures estimated by the GPR approach to the true measures obtained through the computationally intensive full revaluation approach, which involves Monte Carlo VaR Method with $100,000$ paths. Consequently, GPR expedites the risk assessment process significantly. When learning from 5 points, it accelerates the process by 2000 times, and when learning from 10 points, it speeds up the assessment by 1000 times compared to the full revaluation approach, which takes approximately 2 hours and 25 minutes on our machine. The high runtime of the full revaluation approach in this experiment is primarily due to the American pricing calculations using a binomial tree with 100 time steps.

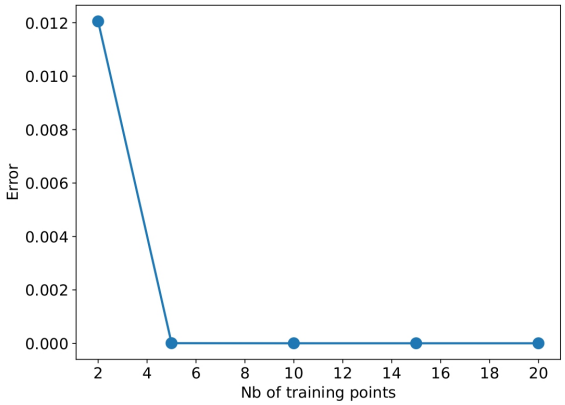

Figure 4: Convergence of MAPE of GPR estimates when increasing the number of training points in mono-asset portfolio case.

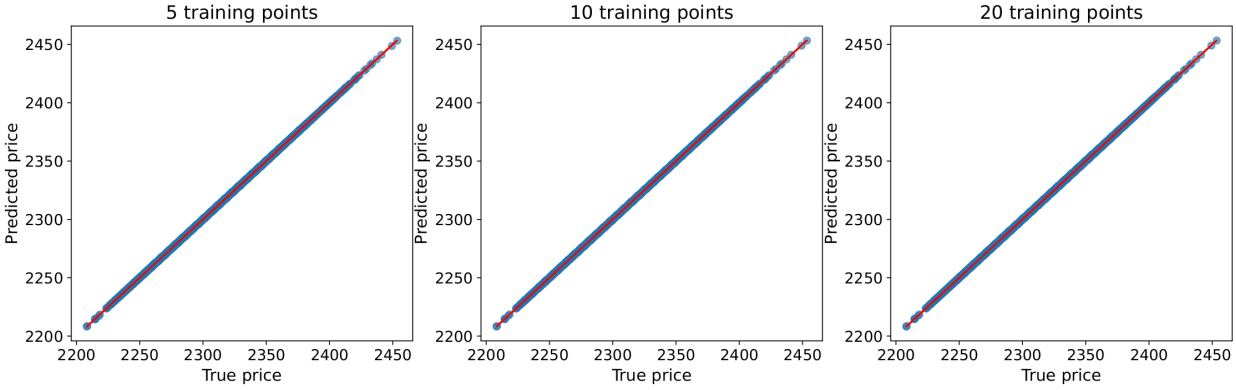

Figure 5: q-q plots of GPR predicted price using 5 (*Left*), 10 (*Center*) and 20 (*Right*) training points against the true price, i.e. analytical and binomial tree prices, in the mono-asset portfolio case.

	Confidence level	True measure	$N_{train} = 5$	$N_{train} = 10$	$N_{train} = 20$
VaR	90%	39.83	39.84	39.83	39.83
	95%	50.90	50.89	50.92	50.91
	97.5%	60.28	60.29	60.29	60.29
	99%	70.53	70.55	70.56	70.54
ES	90%	53.98	53.98	53.99	53.98
	95%	63.02	63.02	63.03	63.02
	97.5%	70.95	70.95	70.97	70.96
	99%	80.15	80.15	80.17	80.15
	Speed-up	2h25 - benchmark	x2000	x1000	x500

Table 4: VaR and ES estimates using GPR against the true measures in the mono-asset portfolio case.

6.2 Multi-asset Options Case

Figure 6 and Table 5 display the results of the pricing and risk calculation for a five-asset geometric average call. Both GPR and mGPR models are trained using Monte-Carlo prices, with the analytical price serving as a benchmark. Additionally, mGPR uses 200 prices obtained through the Monte-Carlo method with 100 paths as low-fidelity data, which is more cost-effective than one training price point used 100,000 paths. The top left plot in Figure 6 shows that two GPR models trained with 50 points can outperform Monte Carlo prices. The error decreases, reaching 10 basis points at 200 training points. Notably, mGPR outperforms GPR at lower numbers of data points before their error curves converge.

The quantile-quantile (q-q) plots in Figure 6 trace the left tail of the simulated price distribution below the 10th percentile, corresponding to the right tail of the loss in the risk calculation in Table 5. Despite the Monte-Carlo price having a higher MAPE error compared to GPR prices, its tail perfectly matches the true price. The tails of GPR and mGPR prices also converge as the number of training points increases. In Table 5, the one-day VaR 99% and ES 97.5% estimated by GPRs using 200 data points have an error of around 7 basis points. Notably, GPR models significantly save time in risk calculation, including sampling and training time with 200 points (13 seconds for GPR compared to 9190 seconds for the brute force Monte-Carlo approach). The mGPR model remains an effective approximation for risk measurement, especially in cases of high pricing engine costs, such as with complex portfolios, and when using minimal price points is preferred.

Figure 7-9 and Table 6-8 show the result of other options, with similar computational times to those in Table 5. Most of the remarks made in the geometric average call case are still valid for these other cases. In the arithmetic average basket call case, the risk measures computed by two GPR models have an error of about 23 basis points, which is less than 10 basis points in the two other cases[5].

[5]These errors can be computed by applying results in Table 6-8 into (37).

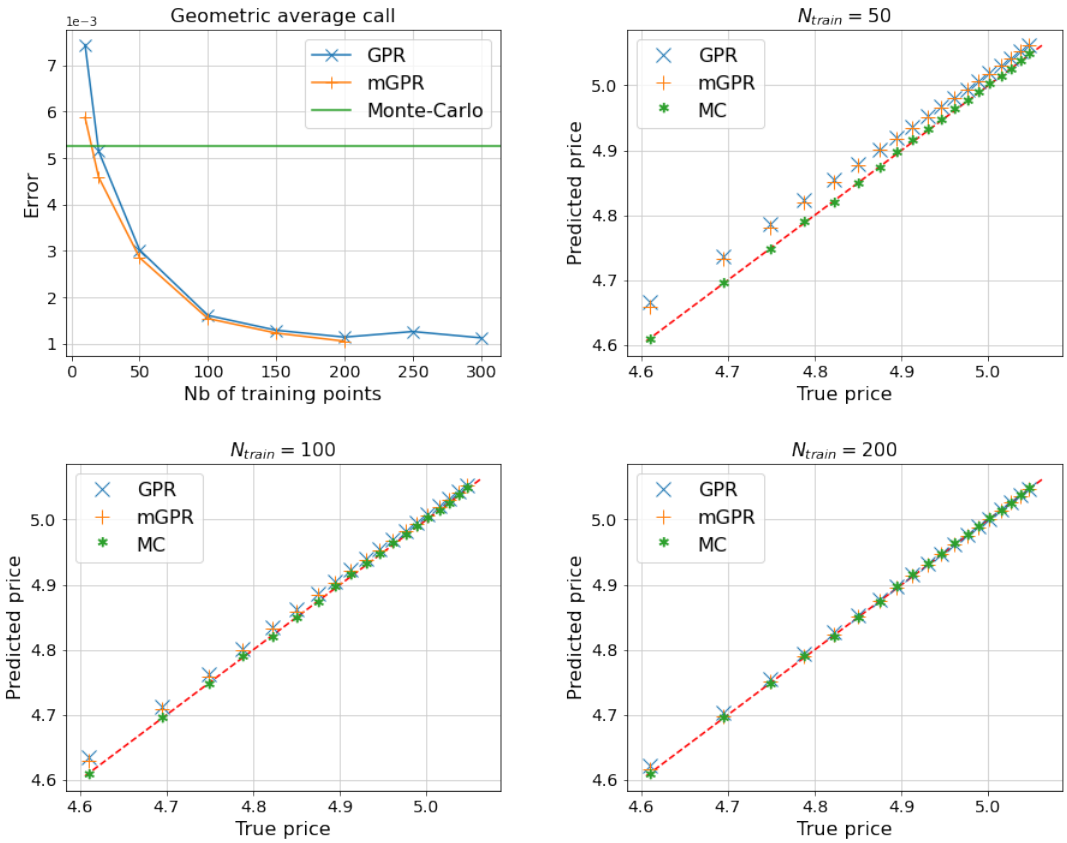

Figure 6: MAPE of estimates of 5-asset geometric average call as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 50 (*Top Right*), 100 (*Bottom Left*) and 200 (*Bottom Right*) points against the true price, i.e. Black-Scholes price, below their 10% quantile levels.
We stop the convergence curve of mGPR at 200 training points (corresponding to high-fidelity data) because we use 200 low-fidelity points.

	Model	Number of training points (N_{train})					
		10	20	50	100	150	200
VaR 99%	GP	0.6900	0.7639	0.7768	0.8007	0.7989	0.8108
	mGP	0.7749	0.8028	0.7815	0.8054	0.8018	0.8155
	MC	0.8220					
	True	0.8190					
ES 97.5%	GP	0.6931	0.7650	0.7757	0.8011	0.8005	0.8108
	mGP	0.7781	0.8060	0.7811	0.8057	0.8033	0.8159
	MC	0.8235					
	True	0.8202					
Computational time (in second)	GP	0	1	3	6	10	13
	mGP	10	10	12	23	30	32
	MC	9190					
True initial price		5.5135					

Table 5: One day VaR 99% and ES 97.5% of 5-asset geometric average call estimated by proxy pricing models.

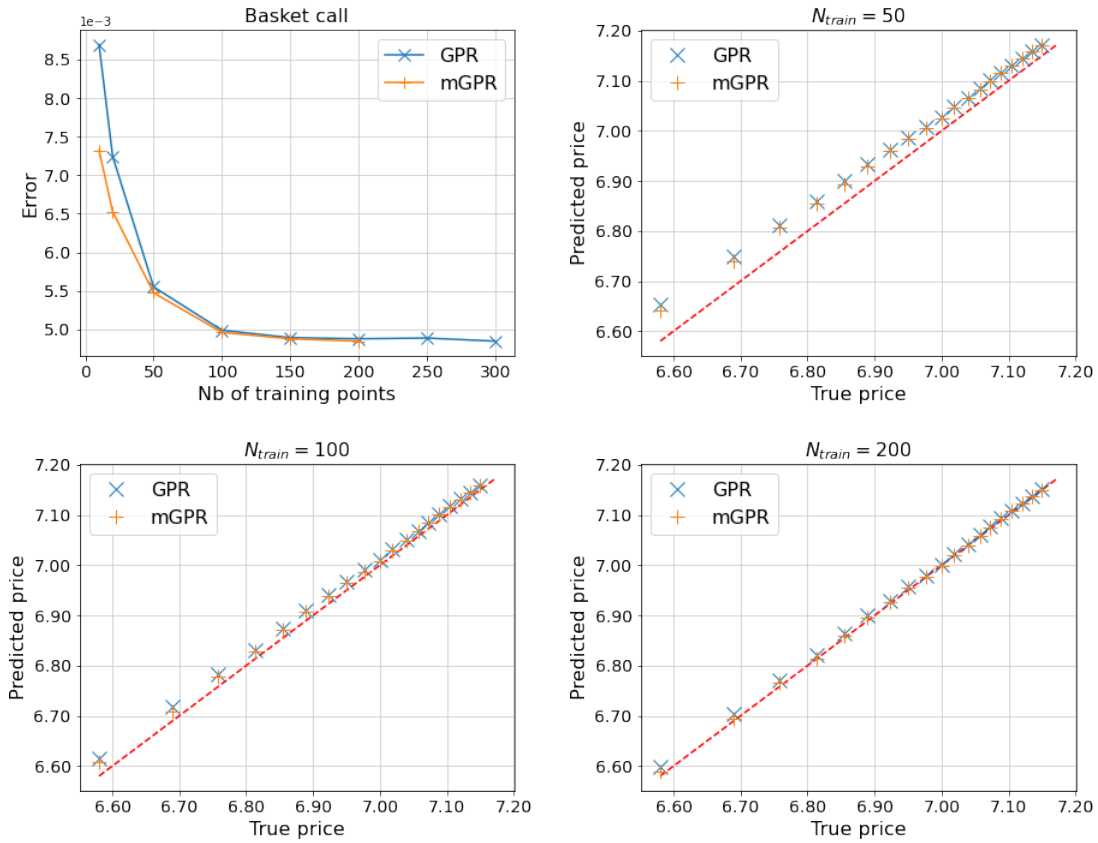

Figure 7: MAPE of estimates of 5-asset basket call as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 50 (*Top Right*), 100 (*Bottom Left*) and 200 (*Bottom Right*) points against the true price, i.e. Monte-Carlo price with $100,000$ paths, below their 10% quantile levels.

	Model	Number of training points (N_{train})					
		10	20	50	100	150	200
VaR 99%	GP	0.9163	1.0067	1.0289	1.0597	1.0588	1.0732
	mGP	1.0123	1.0550	1.0371	1.0673	1.0630	1.0824
	True	1.1013					
ES 97.5%	GP	0.9199	1.0077	1.0280	1.0615	1.0607	1.0748
	mGP	1.0143	1.0583	1.0371	1.0674	1.0643	1.0832
	True	1.1016					
True initial price		7.7770					

Table 6: One day VaR 99% and ES 97.5% of 5-asset basket call estimated by proxy pricing models.

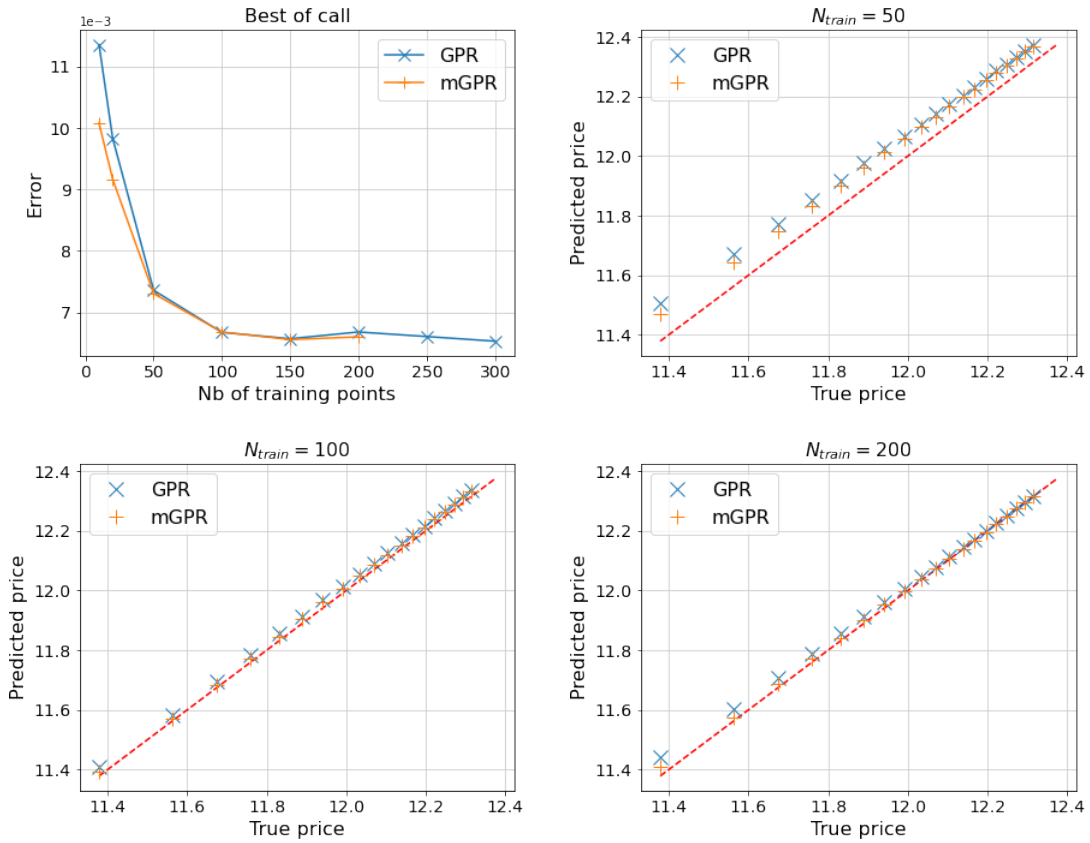

Figure 8: MAPE of estimates of 5-asset best-of-call as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 50 (*Top Right*), 100 (*Bottom Left*) and 200 (*Bottom Right*) points against the true price, i.e. Monte-Carlo price with $100,000$ paths, below their 10% quantile levels.

	Model	Number of training points (N_{train})					
		10	20	50	100	150	200
VaR 99%	GP	1.4704	1.5850	1.6380	1.7258	1.6907	1.7064
	mGP	1.5869	1.6604	1.6682	1.7403	1.6982	1.7346
	True	1.7483					
ES 97.5%	GP	1.4714	1.5878	1.6435	1.7279	1.6928	1.7084
	mGP	1.5906	1.6675	1.6721	1.7438	1.7010	1.7373
	True	1.7517					
True initial price		13.3101					

Table 7: One day VaR 99% and ES 97.5% of 5-asset best-of-call estimated by proxy pricing models.

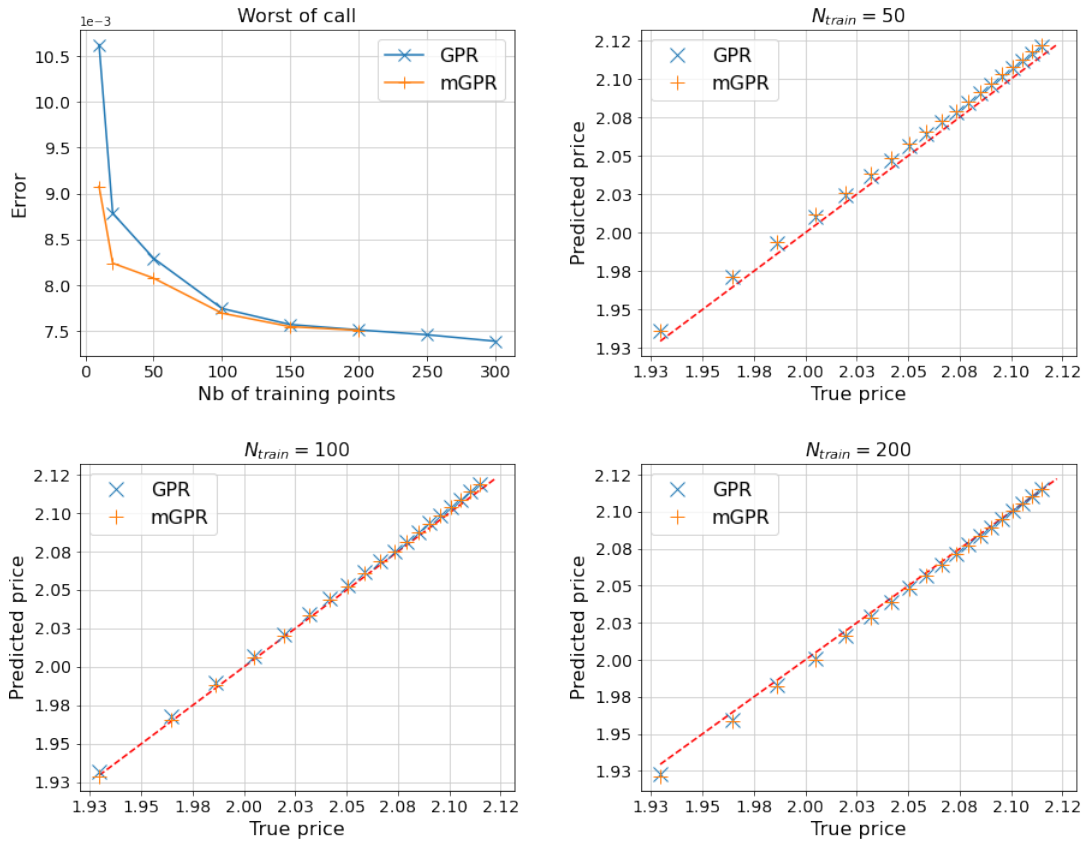

Figure 9: MAPE of estimates of 5-asset worst-of call as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 50 (*Top Right*), 100 (*Bottom Left*) and 200 (*Bottom Right*) points against the true price, i.e. Monte-Carlo price with 100, 000 paths, below their 10% quantile levels.

	Model	Number of training points (N_{train})					
		10	20	50	100	150	200
VaR 99%	GP	0.3153	0.3447	0.3584	0.3619	0.3661	0.3704
	mGP	0.3625	0.3673	0.3584	0.3640	0.3677	0.3711
	True	0.3651					
ES 97.5%	GP	0.3625	0.3673	0.3584	0.3640	0.3677	0.3711
	mGP	0.3624	0.3670	0.3587	0.3645	0.3681	0.3715
	True	0.3657					
True initial price		2.3296					

Table 8: One day VaR 99% and ES 97.5% of 5-asset worst-of call estimated by proxy pricing models.

6.3 Multi-asset Options Portfolio Case

Table 9 and Figure 10 illustrate the numerical results of a multi-asset options portfolio with a one-day. The horizon of the value-at-risk and expected shortfall is set to be one day. The

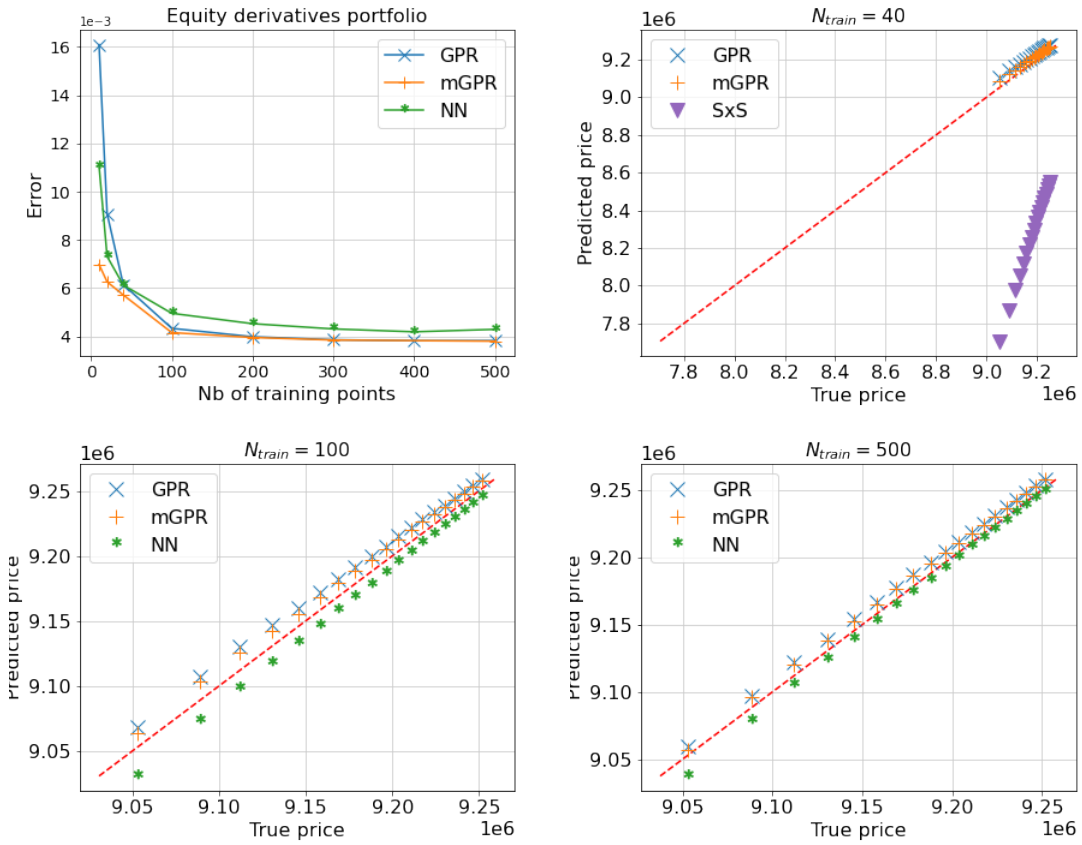

Figure 10: MAPE of estimates of equity option portfolio with one day horizon as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 40 (*Top Right*), 100 (*Bottom Left*) and 500 (*Bottom Right*) points against the true price, i.e. Monte-Carlo price with 100, 000 paths, below their 10% quantile levels. The MAPE of SxS prediction is not included in figures due to its high value but can be found in Table 9.

second-order sensitivity-based approach (SxS) and the neural network learning from 40 points are excluded due to their low precision. In contrast, mGPR incorporates 500 less precise prices from 500 Monte-Carlo paths as low-fidelity data. Despite the portfolio complexity, both GPR models (GPR and mGPR) achieve impressive results with only 40 points. The GPR model has a MAPE of 0.61%, and the mGPR model slightly outperforms the latter with 0.57%. In comparison, SxS performs about 10 times worse with a score of 6.13%. Figure 10 visually represents this difference in which the predicted left tail of SxS prices diverges significantly from the true price represented by the red diagonal line. While the predicted left tails of GPR and mGPR prices are not perfect, but they still concentrate around the red line, i.e. the tail of the true price.

In Table 9, mGPR predictions offer more accurate estimates of the VaR 99% and ES 97.5% compared to GPR predictions, with errors of 34bps and 32bps, respectively, versus 54bps for both errors in GPR. As more data becomes available, the performance of all models improves. Generally, two GPR models outperform NN, with a MAPE of 0.42% (0.38%) compared to the one of NN's 0.5% (0.43%) when learning with $N_{train} = 100$ ($N_{train} = 500$). The difference is more pronounced in the prediction of the left tail, as seen in the two bottom plots in Figure

N_{train}	Full	40			100			500		
model	pricing	SxS	GPR	mGPR	NN	GPR	mGPR	NN	GPR	mGPR
MAPE	9,447,616	6.13%	0.61%	**0.57%**	0.48%	0.43%	**0.42%**	0.43%	**0.38%**	**0.38%**
VaR 99%	358,862	1,575,039	307,983	325,145	374,986	340,394	343,926	368,887	350,397	351,493
ES 97.5%	359,972	1,584,907	309,301	328,193	377,288	342,130	347,312	370,475	351,211	353,116
Err. VaR 99%	-	12.87%	0.54%	**0.36%**	0.17%	0.20%	**0.16%**	0.11%	0.09%	**0.08%**
Err. ES 97.5%	-	12.97%	0.54%	**0.34%**	0.18%	0.19%	**0.13%**	0.11%	0.09%	**0.07%**

Table 9: Pricing approximation and risk calculation of the portfolio with one day horizon.

10. The left tails predicted by GPR and mGPR closely align with the true one, while NN underestimates the left tail. When trained with $N_{train} = 100$, mGPR's errors in risk measures, around 30bps, are only half of GPR's errors. However, this difference disappears when training with $N_{train} = 500$.

N_{train}	Full	40			100			500		
model	valorisation	SxS	GPR	mGPR	NN	GPR	mGPR	NN	GPR	mGPR
Sampling time	600	0.2			0.6			3		
Learning time	0	0	1	41	25	2	56	30	13	135

Table 10: Computation time comparison of VaR and ES calculation by alternative methods.

Table 10 displays the runtime for various risk calculation methods on our server with an Intel(R) Xeon(R) Gold 5217 central processing unit (CPU) and a Nvidia Tesla V100 graphics processing unit (GPU). The full repricing approach involves simulating $100,000$ scenarios of shock, with each scenario being a Monte-Carlo simulation of $100,000$ paths computed over 500 derivatives. Without advanced programming on the GPU, the sampling procedure in banks may take up to one day. Alternative approaches show significant computational performance gains. For instance, standard Gaussian processes regression (GPR) achieves a speedup of 230 times compared to the full repricing approach when learning with $N_{train} = 100$ points, i.e. a total of 2.6 comparing to 600 seconds. Despite mGPR's longer learning times (i.e. 56 when learning with $N_{train} = 100$), its computational efficiency compared to full repricing remains remarkable. It is important to note that, under realistic constraints of the more time-consuming pricing engine in banks, the mGPR approach may take a bit longer than the GPR approach for the learning procedure, but it provides better precision in risk calculation, especially when limiting the use of the pricing engine. Table 11-12 and Figure 11-12 correspond to the risk assessment results for a ten-day and one-month horizon, and observations from the one-day horizon case hold true for other time horizons.

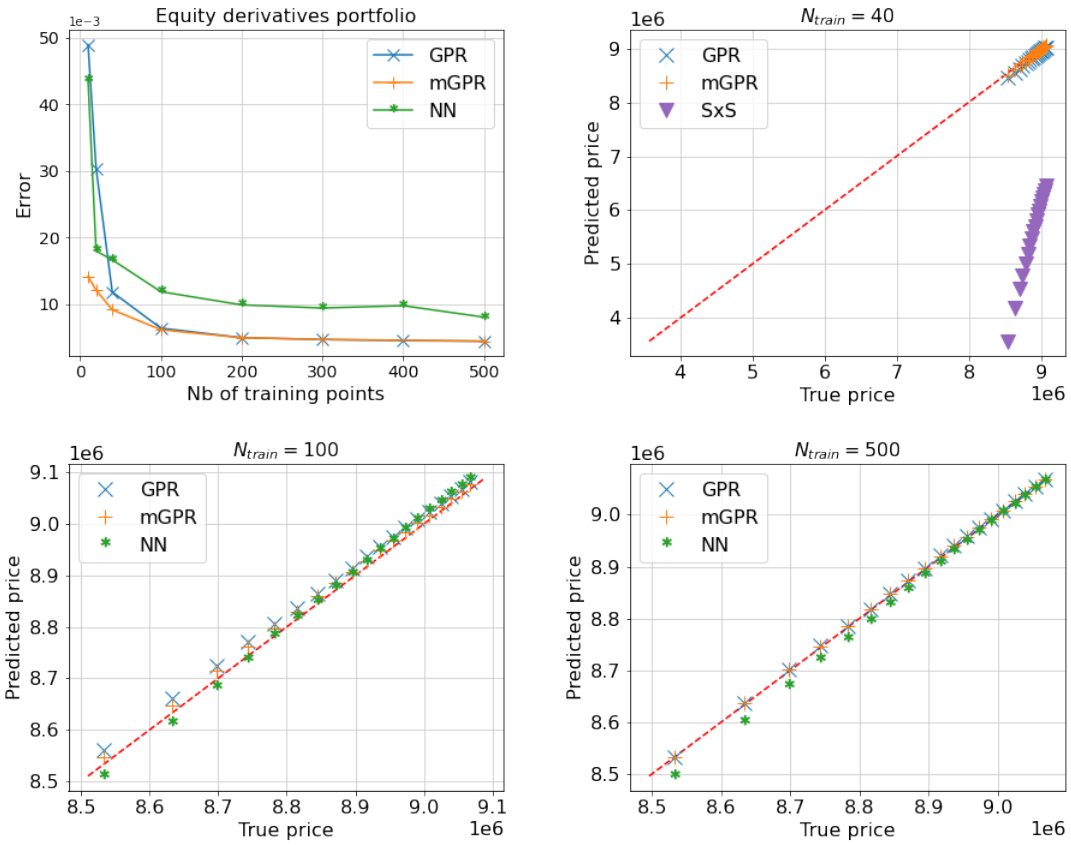

Figure 11: MAPE of estimates of equity option portfolio with ten days horizon as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 40 (*Top Right*), 100 (*Bottom Left*) and 500 (*Bottom Right*) points against the true price, below their 10% quantile levels. The MAPE of SxS prediction is not included in figures due to its high value but can be found in Table 11.

N_{train}	Full	40			100			500		
model	pricing	SxS	GPR	mGPR	NN	GPR	mGPR	NN	GPR	mGPR
MAPE	9,447,616	20.27%	1.18%	**0.92%**	1.19%	0.64%	**0.61%**	0.8%	**0.44%**	**0.44%**
VaR 99%	814,166	5,252,412	896,191	843,186	835,971	785,850	799,501	847,257	810,857	811,540
ES 97.5%	814,604	5,320,871	895,574	843,831	836,584	787,204	800,163	848,514	812,423	812,947
Err. VaR 99%	-	46.98%	0.87%	**0.31%**	0.23%	0.30%	**0.16%**	0.35%	**0.04%**	**0.03%**
Err. ES 97.5%	-	47.70%	0.86%	**0.31%**	0.23%	0.29%	**0.15%**	0.36%	**0.02%**	**0.02%**

Table 11: Pricing approximation and risk calculation of the portfolio with ten days horizon.

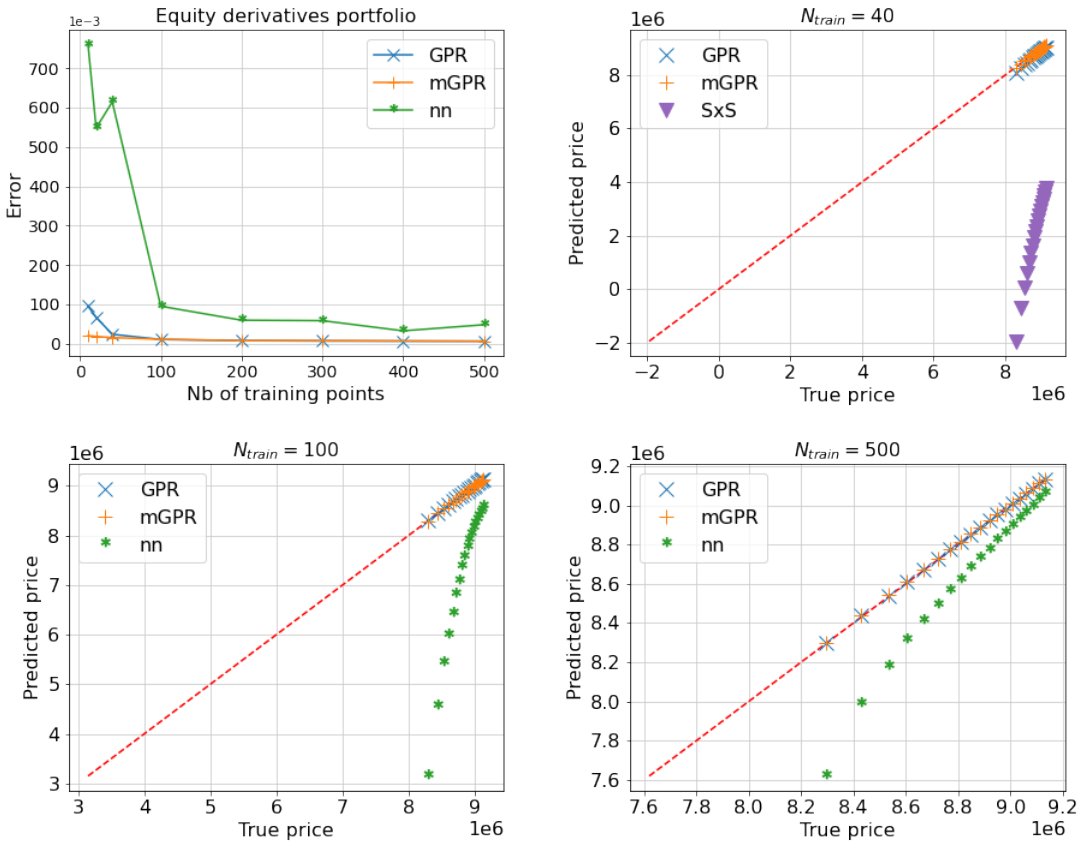

Figure 12: MAPE of estimates of equity option portfolio with one month horizon as a function of the number of training points (*Top Left*) and q-q plots comparing predictors learned from 40 (*Top Right*), 100 (*Bottom Left*) and 500 (*Bottom Right*) points against the true price, i.e. Monte-Carlo price with 100, 000 paths, below their 10% quantile levels. The MAPE of SxS prediction is not included in figures due to its high value but can be found in Table 12.

N_{train}	Full	40			100			500		
model	pricing	SxS	GPR	mGPR	NN	GPR	mGPR	NN	GPR	mGPR
MAPE	9,447,616	38.44%	2.35%	**1.50%**	9.48%	**1.02%**	1.09%	1.72%	**0.60%**	**0.60%**
VaR 99%	1,016,862	10,137,189	1,175,837	1,046,029	1,092,622	995,794	1,006,735	1,103,964	1,009,737	1,010,166
ES 97.5%	1,011,890	10,267,028	1,177,443	1,042,407	1,104,343	991,431	1,004,123	1,119,376	1,005,141	1,005,065
Err. VaR 99%		96.54%	1.68%	**0.31%**	0.8%	0.22%	**0.11%**	0.92%	**0.08%**	**0.07%**
Err. ES 97.5%		97.96%	1.75%	**0.32%**	0.98%	0.22%	**0.08%**	1.14%	**0.07%**	**0.07%**

Table 12: Pricing approximation and risk calculation of the portfolio with one month horizon.

7 Conclusion

In the aftermath of the last financial crises, banking and financial regulations have undergone strengthening, prompting regulators to inspect the internal models of banks. The demand for upgrading risk models and implementing new quantitative risk tools has created enormous challenges. As an initial response, banks have turned to Monte Carlo simulations for implementing time-consuming risk models. The primary limitations arise from the frequent

real-time revaluation of large portfolios containing both linear and non-linear derivatives. To address these constraints, our focus is on the effective implementation and performance issues of risk models. Consequently, we have introduced statistical and machine learning tools, specifically a powerful algorithm for both regression and classification, to handle the repeated valuation of extensive portfolios comprising linear and non-linear derivatives. To be more precise, we applied a Bayesian non-parametric technique, known as Gaussian process regression (GPR). To evaluate the accuracy and performance of these algorithms, we have considered a multi-asset options portfolio and a portfolio of non-linear derivatives, including vanilla and barrier/American options on an equity asset. The numerical tests demonstrated that the GPR algorithm outperforms pricing models in efficiently conducting repeated valuations of large portfolios. It is noteworthy that the GPR algorithm eliminates the need to separately revalue each derivative security within the portfolio, leading to a significant speed-up in calculating value-at-risk (VaR) and expected shortfall (ES). This independence from the size and composition of the trading portfolio is remarkable. Consequently, we argue that it is more advantageous for banks to construct their risk models using the power of GPR techniques in terms of calculation accuracy and speed-up.

Moreover, we have investigated multi-fidelity modeling technique and have explored its applications in pricing and risk calculation, a relatively novel concept in quantitative finance. The multi-fidelity modeling approach leverages limited access to the price engine while incorporating information from other related, more affordable resources to enhance the approximation. Trained regression models offer price estimates at the portfolio level, making them more favorable for risk calculation compared to classical pricing approaches that evaluate products individually. Our numerical findings indicate that machine learning models such as neural networks and Gaussian process regression provide more accurate results and significant time savings compared to traditional sensitivity-based approaches, maintaining precision in risk calculation akin to the full repricing approach. With the limitation of training price data, GPR models yield more impressive results than neural network. Despite the longer learning time required for mGPR, especially when low-fidelity data is available, it yields better or at least equal precision compared to the standard GPR model. Notably, as multi-fidelity modeling is not yet well-known in quantitative finance, it presents an intriguing direction for research, allowing scientists to explore traditional relationships in finance to define low-fidelity models and enhance the learning process.

Future research delves into other topics and applications of algorithms such that Gaussian process regression and multi-fidelity modeling, with a particular focus on the fixed income portfolio. We also envisage the use of the replicating portfolio value, assumed to be accessible, as low-fidelity data to leverage the learning of the target portfolio value. Additionally, the price of European basket options can serve as low-fidelity data to approximate the price of their American/Bermudian counterparts. The Black-Scholes price of derivatives can be used to improve the approximation of their Heston price. We leave these interesting applications there for future study.

A Barone-Adesi and Whaley Approximation of American Option Values

In contrast to European options, American ones grant the holder the right to exercise the contract at any time up to the expiration date. This leads to a more complicated mathematical problem when pricing American options, for which we need to use some numerical methods, such as tree models with high time steps as mentioned in Section 2.3 and 6.1. For this reason, pioneers in finance quantitative have looked for a way to approximate the American option values. Among them, Barone-Adesi and Whaley [1987] propose an analytical approximation of American option value based on the price its European counterparts, as presented below.

For a European call (put) of strike K, maturity T and underlying asset S_t following log normal dynamic (cf. (3)), its value c_{BS} (p_{BS}) is computed by the well-known Black-Scholes formula:

$$\begin{aligned} c_{BS}\left(S_t,t\right) &= \Phi\left(d_1(S_t)\right)S_t - \Phi\left(d_2(S_t)\right)Ke^{-r(T-t)}, \text{ and} \\ p_{BS}\left(S_t,t\right) &= -\Phi\left(-d_1(S_t)\right)S_t + \Phi\left(-d_2(S_t)\right)Ke^{-r(T-t)}, \end{aligned} \tag{38}$$

where

$$\begin{aligned} d_1(\cdot) &= \frac{1}{\sigma\sqrt{T-t}}\left[\ln\left(\frac{\cdot}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)\right], \\ d_2(\cdot) &= d_1(\cdot) - \sigma\sqrt{T-t}, \end{aligned}$$

with r is risk-free rate, σ is the asset volatility and Φ is cumulative density function of the standard normal variable.

Barone-Adesi and Whaley [1987] approximation of American option values reads

$$C_{BW}(S_t,t) = \begin{cases} c_{BS}(S_t,t) + A_2\left(\dfrac{S_t}{S^*}\right)^{q_2}, & \text{when } S_t < S^* \text{ and} \\ S_t - K, & \text{when } S_t \geq S^*, \end{cases}$$

$$P_{BW}(S_t,t) = \begin{cases} p_{BS}(S_t,t) + A_1\left(\dfrac{S_t}{S^{**}}\right)^{q_1}, & \text{when } S_t > S^{**} \text{ and} \\ K - S_t, & \text{when } S_t \leq S^{**}, \end{cases}$$

where

$$\begin{aligned} q_1 &= \frac{1}{2}\left[-\left(\frac{2r}{\sigma^2}-1\right) - \sqrt{\left(\frac{2r}{\sigma^2}-1\right)^2 + \frac{8r}{\sigma^2(1-e^{-rT})}}\right], \\ q_2 &= \frac{1}{2}\left[-\left(\frac{2r}{\sigma^2}-1\right) + \sqrt{\left(\frac{2r}{\sigma^2}-1\right)^2 + \frac{8r}{\sigma^2(1-e^{-rT})}}\right], \\ A_1 &= -\frac{S^{**}}{q_1}\left[1 - \Phi(-d_1(S^{**}))\right],\ A_2 = \frac{S^*}{q_2}\left[1 - \Phi(d_1(S^*))\right], \end{aligned}$$

with S^* and S^{**} satisfy

$$S^* - K = c_{BS}(S^*, t) + \frac{S^*}{q_2}[1 - \Phi(d_1(S^*))], \text{ and}$$
$$K - S^{**} = p_{BS}(S^{**}, t) + \frac{S^{**}}{q_1}[1 - \Phi(-d_1(S^{**}))].$$

B Sensitivity-based pricing approximation

In order to measure the market risk by Monte-Carlo without recalling expensive pricing engine, many banks use Taylor approximation for portfolio evaluation. This technique is also known as delta-gamma approximation method in literature [Britten-Jones and Schaefer, 1999; Broadie et al., 2015]. With the same notation used in Section 2.1, for each scenario of underlying assets RF_{t+h}, the corresponding value of the portfolio $V_{t+h} = p_{t+h}(RF_{t+h})$ is approximated by

$$p_{t+h}(RF_{t+h}) \approx p_t(RF_t) + (\delta RF_h)^\top \Delta + \frac{1}{2}(\delta RF_h)^\top \Gamma \, \delta RF_h, \tag{39}$$

where Δ and Γ are, respectively, the gradient (i.e. delta) and Hessian matrix (i.e. gamma) of the portfolio value V_t with respect to risk factor RF, and $\delta RF_h = RF_{t+h} - RF_t$. Because of the nature of (39), pioneers in banks usually call this pricing approximation by sensitivities times shocks, abbreviated by SxS. In practice, Δ and Γ are not analytically available in closed form and they are usually estimated by finite difference (or bump and revalue) in practice. If there are d risk factors affecting the portfolio, then one needs to recall the price engine $2d$ times for estimating Δ by central finite difference [Crépey, 2013, Chapter 8] and additionally $d(d-1)$ times for estimating Γ. This approximation pricing approach provides a significant computational improvement in risk calculation for small and medium numbers of risk factors d, i.e. less than 100. However, its precision can reach an acceptable level once several conditions are met: the portfolio value is well-approximated by a linear or quadratic functions of risk factors; and numerical estimates of Δ and Γ are accurate.

C Neural Network Regression

Alongside Gaussian processes, the neural network is also a powerful non-linear model for interpolation. Different to GP regression, the neural network can be learned quickly and is scalable with the number of training points, however the model requires a sufficient training data set to learn. In this section, we provide the main notation and concept of neural networks referring the reader to LeCun et al. [2015] for a detailed presentation.

We denote by $\mathcal{NN}_{d,o,h,u,\sigma}$ neural network family of h hidden layers, u hidden units and activation function σ, furthermore, this family maps input vectors in $\mathcal{X} \in \mathbb{R}^d$ to output vectors in $\mathcal{Y} \in \mathbb{R}^o$. Functions in this family are represented by the following sequential mapping (see also Figure 13),

$$\begin{aligned} \mathbf{a}^0 &= \mathbf{z}^0 = \mathbf{x} \\ \mathbf{a}^l &= \sigma(\mathbf{z}^l) = \sigma\left(\mathbf{W}^l \mathbf{a}^{l-1} + \mathbf{b}^l\right) \quad , l = 1, \ldots, h \\ z^{h+1} &= (\mathbf{w}^{h+1})^\top \mathbf{a}^h + b^{h+1}. \end{aligned} \tag{40}$$

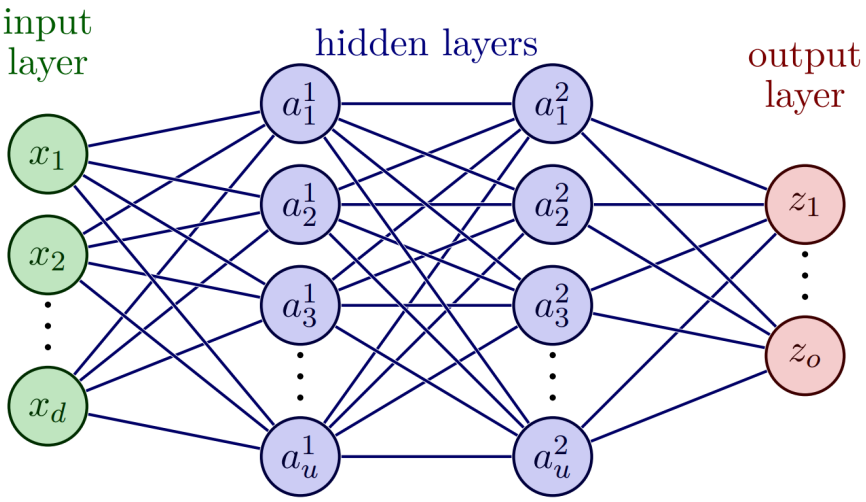

Figure 13: Two-hidden-layer neural network architecture.

This family of functions is parameterized by weights $\mathbf{W}^1 \in \mathbb{R}^{u\times d}, \ldots, \mathbf{W}^l \in \mathbb{R}^{u\times u}, \ldots, \mathbf{w}^{h+1} \in \mathbb{R}^u$ and biases $\mathbf{b}^1 \in \mathbb{R}^u, \ldots \mathbf{b}^l \in \mathbb{R}^u, \ldots, b^{h+1} \in \mathbb{R}^o$. We denote the set of trainable parameters as $\theta = \left(\mathbf{W}^1, \ldots, \mathbf{W}^{h+1}, \mathbf{b}^1, \ldots, \mathbf{b}^h, b^{h+1}\right)$. Additionally, nonlinear activation function σ, element-wisely applied, plays a crucial role in establishing the complex mapping between inputs and outputs of the network. The rectified linear unit function (Relu), i.e. $\text{Relu}(x) = \max(x, 0)$ is usually chosen, that is also our case, because of its generalisation property [LeCun et al., 2015, Section 6.3.1].

Given a set of training data $\{(\mathbf{x}_i, y_i)_{i=1\ldots N_{train}}\}$, the neural network regression looks for minimizing the following mean square error

$$\hat{f} \in \underset{f\in\mathcal{NN}_{d,o,h,u,\sigma}}{\arg\min} \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} \|\mathbf{x}_i) - y_i\|_2^2 , \tag{41}$$

where $\|\cdot\|_2$ is the Euclidean norm notation. Equation (41) is solved by the combination of backpropagation technique and (stochastic) gradient descent. The detail of these algorithms can be found in [LeCun et al., 2015, Chapter 8].

References

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

Barone-Adesi, G. and R. E. Whaley (1987). Efficient analytic approximation of american option values. *the Journal of Finance 42*(2), 301–320.

Basel Committee on Banking Supervision (2013). Fundamental review of the trading book: A revised market risk framework. *Available at https://www.bis.org/publ/bcbs265.pdf*.

Basel Committee on Banking Supervision (2019). Minimum capital requirements for market risk. *Available at https://www.bis.org/bcbs/publ/d457.pdf*.

Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association 112*(518), 859–877.

Brevault, L., M. Balesdent, and A. Hebbal (2020). Overview of gaussian process based multi-fidelity techniques with variable relationship between fidelities, application to aerospace systems. *Aerospace Science and Technology 107*, 106339.

Britten-Jones, M. and S. M. Schaefer (1999). Non-linear value-at-risk. *Review of Finance 2*(2), 161–187.

Broadie, M., Y. Du, and C. C. Moallemi (2015). Risk estimation via regression. *Operations Research 63*(5), 1077–1097.

Buehler, H., L. Gonon, J. Teichmann, and B. Wood (2019). Deep hedging. *Quantitative Finance 19*(8), 1271–1291.

Crépey, S. (2013). *Financial Modeling, A Backward Stochastic Differential Equations Perspective.* Springer Finance Textbook Series.

Crépey, S. and M. Dixon (2019). Gaussian process regression for derivative portfolio modeling and application to CVA computations. *Journal of Computational Finance 24*(1), 1–35.

Culkin, R. and S. R. Das (2017). Machine learning in finance: the case of deep learning for option pricing. *Journal of Investment Management 15*(4), 92–100.

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

Dowd, K. (2003). *An introduction to market risk measurement.* John Wiley & Sons.

Dowd, K. (2007). *Measuring market risk.* John Wiley & Sons.

Ferguson, R. and A. Green (2018). Deeply learning derivatives. *CompSciRN: Artificial Intelligence (Topic).*

Fernández-Godino, M. G., C. Park, N.-H. Kim, and R. T. Haftka (2016). Review of multi-fidelity models. *Available at arXiv:1609.07196*.

Financial Stability Board (2017). Artificial intelligence and machine learning in financial services: Market developments and financial stability implications. *Available at https://www.fsb.org/wp-content/uploads/P011117.pdf*.

Forrester, A. I. J., A. Sóbester, and A. J. Keane (2007). Multi-fidelity optimization via surrogate modelling. *the Royal Society A: Mathematical, Physical and Engineering Sciences 463*, 3251 – 3269.

Gardner, J., G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson (2018). Gpytorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. *Advances in neural information processing systems 31*, 7576–7586.

Gardner, J., G. Pleiss, R. Wu, K. Weinberger, and A. Wilson (2018). Product kernel interpolation for scalable Gaussian processes. *International Conference on Artificial Intelligence and Statistics 84*, 1407–1416.

Goudenège, L., A. Molent, and A. Zanette (2021). Variance reduction applied to machine learning for pricing Bermudan/American options in high dimension. Oleg Kudryavtsev, Antonino Zanette. *Applications of Lévy Processes*.

Hoffman, M. D., D. M. Blei, C. Wang, and J. Paisley (2013). Stochastic variational inference. *Journal of Machine Learning Research 14*, 1303–1347.

Hong, L. J., Z. Hu, and G. Liu (2014). Monte carlo methods for value-at-risk and conditional value-at-risk: a review. *ACM Transactions on Modeling and Computer Simulation (TOMACS) 24*(4), 1–37.

Horvath, B., A. Muguruza, and M. Tomas (2021). Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. *Quantitative Finance 21*(1), 11–27.

Huang, G.-B., Q.-Y. Zhu, and C.-K. Siew (2006). Extreme learning machine: theory and applications. *Neurocomputing 70*(1-3), 489–501.

Huré, C., H. Pham, and X. Warin (2020). Deep backward schemes for high-dimensional nonlinear PDEs. *Mathematics of Computation 89*(324), 1547–1579.

Hutchinson, J. M., A. W. Lo, and T. Poggio (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *The journal of Finance 49*(3), 851–889.

Kennedy, M. C. and A. O'Hagan (2000). Predicting the output from a complex computer code when fast approximations are available. *Biometrika 87*(1), 1–13.

Kingma, D. P. and J. Ba (2015). Adam: A method for stochastic optimization. *The 3rd International Conference on Learning Representations*.

Le Gratiet, L. (2013). *Multi-fidelity Gaussian process regression for computer experiments*. Ph. D. thesis, Université Paris-Diderot-Paris VII.

LeCun, Y., Y. Bengio, and G. Hinton (2015). Deep learning. *Nature 521*(7553), 436–444.

Lehdili, N., P. Oswald, and H. Gueneau (2019). Market risk assessment of a trading book using statistical and machine learning. *Available at DOI: 10.13140/RG.2.2.23796.71047*.

Li, S., W. Xing, R. Kirby, and S. Zhe (2020). Multi-fidelity bayesian optimization via deep neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Eds.), *Advances in Neural Information Processing Systems*, Volume 33, pp. 8521–8531. Curran Associates, Inc.

Liu, S., C. W. Oosterlee, and S. M. Bohte (2019). Pricing options and computing implied volatilities using neural networks. *Risks 7*(1), 16.

Longstaff, F. A. and E. S. Schwartz (2001). Valuing American options by simulation: a simple least-squares approach. *The review of financial studies 14*(1), 113–147.

Ludkovski, M. (2018). Kriging metamodels and experimental design for Bermudan option pricing. *Journal of Computational Finance 22*(1), 37–77.

Meng, X. and G. E. Karniadakis (2020). A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems. *Journal of Computational Physics 401*, 109020.

Mu, G., T. Godina, A. Maffia, and Y. C. Sun (2018). Supervised machine learning with control variates for american option pricing. *Foundations of Computing and Decision Sciences 43*(3), 207–217.

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. The MIT Press.

Nocedal, J. and S. J. Wright (1999). *Numerical optimization*. Springer.

Paleyes, A., M. Mahsereci, and N. D. Lawrence (2023). Emukit: A python toolkit for decision making under uncertainty. *Proceedings of Python in Science Conference 22*, 68–75.

Rasmussen, C. E. and C. K. Williams (2006). *Gaussian Processes for Machine learning*. The MIT Press.

Roncalli, T. (2020). *Handbook of Financial Risk Management*. Chapman and Hall/CRC.

Ruf, J. and W. Wang (2019). Neural networks for option pricing and hedging: A literature review. *Journal of Computational Finance 24*(1), 1–46.

Ruiz, I. and M. Zeron (2021). *Machine Learning for Risk Calculations: A Practitioner's View*. John Wiley & Sons.

Saunders, A., M. M. Cornett, and O. Erhemjamts (2021). *Financial institutions management: A risk management approach*. McGraw-Hill.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. *Proceedings of Machine Learning Research 5*, 567–574.

Zhang, H.-G., C.-W. Su, Y. Song, S. Qiu, R. Xiao, and F. Su (2017). Calculating value-at-risk for high-dimensional time series using a nonlinear random mapping model. *Economic Modelling 67*, 355–367.