Gaussian Process Regression for Pricing Variable Annuities with Stochastic Volatility and Interest Rate

Antonino Zanette

University of Udine (Italy)

22 May 2019-AFIR ERM 19 Florence

From joint works with:

Ludovic Goudenège* and Andrea Molent **

* CNRS, Centrale Supèlec Paris

** University of Udine (Italy)

Introduction

- In this talk we present an efficient approach based on a Machine Learning technique which allows one to quickly evaluate insurance products considering stochastic volatility and interest rate.
- Specifically, following De Spiegeleer et al. 2018, we apply Gaussian Process Regression to compute the no-arbitrage fee and the Greeks of a GMWB Variable Annuity.
- First of all, we price the securities by means of the Hybrid Tree PDE approach.
- Then, we apply the Gaussian Process Regression technique to make fast prediction of the prices.

Introduction

- The regression algorithm consists of two main steps: algorithm training and evaluation.
- The training step is the most time demanding, but it needs to be performed only once.
- The evaluation step is very fast and it requires to be performed only when making predictions.
- We consider three increasing complexity models, namely the Black-Scholes, the Heston and the Heston Hull-White models, which extend the sources of randomness up to consider stochastic volatility and stochastic interest rate together.
- Numerical experiments show that the accuracy of the estimated values is high, while the computational cost is much lower than the one required by a direct calculation with standard approaches.

Outline

The stochastic models

The stochastic models

We focus on the following stochastic models :

- The Heston model

$$\begin{cases} dS_t = rS_t dt + \sqrt{V_t} S_t dZ_t^S & S_0 = \bar{S}_0, \\ dV_t = k\left(\theta - V_t\right) dt + \omega \sqrt{V_t} dZ_t^V & V_0 = \bar{V}_0, \end{cases} \tag{1}$$

  where Z^S and Z^V are Brownian motions, and $d\left\langle Z_t^S, Z_t^V \right\rangle = \rho dt$.

- The Heston Hull-White Model

$$\begin{cases} dS_t & = r_t S_t dt + \sqrt{v_t} S_t dZ_t^S \\ dr_t & = k_r \left(\theta_r\left(t\right) - r_t\right) dt + \omega_r dZ_t^r, \\ dv_t & = k_v \left(\theta_v - v_t\right) dt + \omega_v \sqrt{v_t} dZ_t^v, \end{cases}$$

  θ_t is a deterministic function which is completely determined by the market values of the zero-coupon bonds by calibration.

Variables Annuities

Variables Annuities

- Variable Annuities are unit-linked life insurance contracts with investment guarantees which, in exchange for single or regular premiums, allow the policyholder to benefit from the upside of the market, but be partially or totally protected when the unit loses value.
- To mitigate the policyholder exposure and to offer additional types of benefits, various forms of guarantees, such as guaranteed minimum death benefit, guaranteed minimum income benefit, guaranteed minimum withdrawal benefit, and guaranted lifetime withdrawal benefit are commonly embedded into the contracts.

The GMWB contract

- At time t = 0 the PH pays the premium P to the insurance company. The premium P is invested in a fund.
- The contract guarantees a minimum amount to be withdrawn, even if the account value A declines to 0.
- Immediately after the last event time, the PH receives a final payoff and the contract ends.
- The contract state parameters at a given time t are the account value A_t and the base benefit B_t.
- Let $V(A, B, v, t)$ be the fair value of the considered GMWB contract at time t (under risk neutral probability), considering the Heston model, and $V(A, B, v, r, t)$ considering the Heston-Hull-White model.

Evolution of the Contract between Event Times.

The account value A_t - the risky account - is determined according to the following relation:

$$dA_t = \frac{A_t}{S_t} dS_t - \alpha A_t dt. \tag{2}$$

Event Times

- During the event times, the PH is entitled to withdraw a minimum guaranteed amount G (usually $G = P/N$) from his account.
- Let W_i represent the amount withdrawn at time t_i, which is required to be non-negative and smaller than the base benefit $B_{t_i^{(-)}}$.
- If the amount withdrawn W_i is lower than G, then there is no penalty imposed.
- If W_i is higher than G, a proportional penalty charge $\kappa_i\left(W_i - G\right)$ is imposed, which reduces the amount actually received by the PH.

Final Payoff

- The contract state variables just after the withdrawal updated as follows:

$$\left(A_{t_i^{(+)}}, B_{t_i^{(+)}}, u_{t_i^{(+)}}, t_i\right) = \left(\left(A_{t_i^{(-)}} - W_i\right)_+, B_{t_i^{(-)}} - W_i, u_{t_i^{(-)}}, t_i\right). \tag{3}$$

- After the last event time t_N has occurred, the PH receives the final payoff, which is worth

$$FP = \max\left(A_T, (1-\kappa_N)\, B_T\right), \tag{4}$$

and the contract terminates.

Withdrawals

As far as the withdrawal strategy is concerned, we consider dynamic withdrawals. The PH chooses W_i in order to maximize the total wealth. The following dynamic control problem has to be solve:

$$W_i = \underset{w_i \in \left[0, B_{t_i^{(-)}}\right]}{\operatorname{argmax}} \mathcal{V}\left(\max\left(A_{t_i^{(-)}} - w_i, 0\right), B_{t_i^{(-)}} - w_i, u_{t_i}, t_i^+\right) + f_i\left(w_i\right). \qquad (5)$$

where

$$f_i\left(W_i\right) = \begin{cases} W_i & \text{if } W_i \leq G \\ W_i - \kappa_i\left(W_i - G\right) & \text{if } W_i > G. \end{cases}$$

Hybrid Tree-Finite Difference method

Hybrid Tree-Finite Difference

In order to price the GMWB contracts, we employ the Hybrid Tree-Finite Difference method, introduced by:

Briani, M., Caramellino, L., Zanette, A. (2018): "A hybrid tree/finite-difference approach for Heston-Hull-White type models". *The Journal of Computational Finance*

The hybrid procedure :

- a 2-dimensional binomial tree for the the pair volatility (V, r);
- a finite difference approach in the S-direction.

L.Goudenege A.Molent A.Zanette (2016) *Pricing and Hedging GLWB in the Heston and in the Black-Scholes with Stochastic Interest Rate Models. Insurance: Mathematics and Economics*
L.Goudenege A.Molent A.Zanette (2019) Pricing and Hedging GMWB in the Heston and in the Black-Scholes with Stochastic Interest Rate Models.Computational Management Science

Gaussian Process Regression

Gaussian Process Regression

- Gaussian Process Regression (also known as GPR) is a class of non-parametric kernel-based probabilistic models which represents the input data as the random observations of a Gaussian stochastic process.
- The most important advantage of this approach is that it is possible to effectively exploit a complex dataset which may consist of points sampled randomly in a multidimensional space.

Gaussian process

- A Gaussian process $\mathcal{G}$ is a collection of random variables defined on a common probability space, any finite number of which have consistent joint Gaussian distributions.
- If $\mathbf{x}_i \in \mathbb{R}^D$ for $i = 1, \ldots, n$ then $\left(\mathcal{G}\left(\mathbf{x}_1\right), \ldots, \mathcal{G}\left(\mathbf{x}_n\right)\right)^\top$ is a random Gaussian vector.
- A Gaussian process is fully specified by its mean function $\mu\left(\mathbf{x}\right)$ and by its covariance function $k\left(\mathbf{x}, \mathbf{x}'\right)$.

Training set

- Now, let us consider a training set $\mathcal{D}$ of n observations $\mathcal{D} = \{(\mathbf{x}_i, y_i) \,|\, i = 1, \ldots, n\}$ where $X = \{\mathbf{x}_i | i = 1, \ldots, n\} \subset \mathbb{R}^D$ denotes the set of input vectors and $Y = \{y_i | i = 1, \ldots, n\} \subset \mathbb{R}$ denotes the set of scalar outputs.
- These observations are modeled as the realization of the sum of a Gaussian process and a noise source. Specifically,

$$y_i = f_i + \varepsilon_i \tag{6}$$

where $\{f_i = \mathcal{G}(\mathbf{x}_i) \,|\, i = 1, \ldots, n\}$ is a Gaussian process and $\{\varepsilon_i | i = 1, \ldots, n\}$ are i.i.d. random variables.
- Moreover, the distribution of $\mathbf{f} = (f_1 \ldots f_n)$ is assumed to be given by

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, K(X, X)), \tag{7}$$

where $K(X, X)$ is a $n \times n$ matrix with $K(X, X)_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$.

Test set

- Now, in addition, let us consider a test set $\tilde{X}$ of m points $\{\tilde{\mathbf{x}}_j | j = 1, \ldots, m\}$. The realizations $\tilde{f}_j = \mathcal{G}(\tilde{\mathbf{x}}_j)$ are not known but rather we want to estimate them considering the observed realizations of $\mathcal{G}$ in $\mathcal{D}$.
- The *a priori* joint distribution of $\mathbf{y}$ and $\tilde{\mathbf{f}} = \left(\tilde{f}_1 \ldots \tilde{f}_m\right)$ is given by

$$\begin{bmatrix} \mathbf{y} \\ \tilde{\mathbf{f}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0}_n \\ \mathbf{0}_m \end{bmatrix}, \begin{bmatrix} K(X,X) + \sigma_n^2 I_n & K\left(X, \tilde{X}\right) \\ K\left(\tilde{X}, X\right) & K\left(\tilde{X}, \tilde{X}\right) \end{bmatrix} \right) \quad (8)$$

Test set

- Since we know the values for the training set, we can consider the conditional distribution of $\tilde{\mathbf{f}}$ given $\mathbf{y}$. It is possible to prove that the conditional distribution of $\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X$ is Gaussian and given by

$$\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X \sim \mathcal{N}\left(\mathbb{E}\left[\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X\right], Cov\left[\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X\right]\right), \tag{9}$$

where

$$\mathbb{E}\left[\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X\right] = K\left(\tilde{X}, X\right)\left[K(X,X) + \sigma_n^2 I_n\right]^{-1}\mathbf{y} \tag{10}$$

and

$$Cov\left[\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X\right] = K\left(\tilde{X}, \tilde{X}\right) - K\left(\tilde{X}, X\right)\left[K(X,X) + \sigma_n^2 I_n\right]^{-1} K\left(X, \tilde{X}\right). \tag{11}$$

- Therefore, a natural choice consists in predicting the values $\tilde{\mathbf{f}}$ through $\mathbb{E}\left[\tilde{\mathbf{f}}|\tilde{X}, \mathbf{y}, X\right]$. Moreover, also confidence intervals for such a prediction can be computed.

Covariance function

- The covariance matrix $K\left(\tilde{X}, \tilde{X}\right)$ is given by

$$K\left(\tilde{X}, \tilde{X}\right)_{i,j} = k\left(\mathbf{x}_i, \mathbf{x}_i\right)$$

where k is called the kernel (covariance) function .

- We employ the Automatic Relevance Determination Squared Exponential (ARD SE) kernel, which is given by

$$k\left(\mathbf{x}, \mathbf{x}'\right) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{k=1}^{D}\frac{\left(\mathbf{x}_k - \mathbf{x}_k'\right)^2}{l_k^2}\right), \tag{12}$$

- The parameters σ_f^2, $l_1, \ldots, l_D$ of the kernel function and σ_n^2 of the noise are called hyperparameters and need to be estimated.

Maximum likelihood estimates

- A common approach is to consider the maximum likelihood estimates which can be obtained by maximizing the log-likelihood function of the training data, that is by maximizing the following function:

$$-\frac{1}{2}\log\left(\det\left(K\left(X,X\right)+\sigma_n^2 I_n\right)\right)-\frac{1}{2}\mathbf{y}^\top\left[K\left(X,X\right)+\sigma_n^2 I_n\right]^{-1}\mathbf{y}. \quad (13)$$

Basis functions

- The mean function μ can be modeled using a set of basis functions, such as polynomials. In particular, when we consider a linear basis, the function μ is approximated via $\mathbf{h}(\mathbf{x})\beta$, where $\mathbf{h}(\mathbf{x}) = (1, x_1, \ldots, x_D)$ and $\beta \in \mathbb{R}^{n+1}$ is a vector of coefficients estimated via the observed data.

Testing step and training step

- The development of the GPR model can be divided in the training step and the evaluation step (also called testing step).
- The training step only requires the knowledge of the training set $\mathcal{D}$ and it consists in estimating the vector β, the hyperparameters and the matrix product $A = \left[K\left(X, X\right) + \sigma_n^2 I_n\right]^{-1}\left(\mathbf{y} - \mu\left(X\right)\right)$.
- The evaluation step can be computed only after the training step has been accomplished and it consists in obtaining the predictions via the computation of $\mu\left(\tilde{X}\right) + K\left(\tilde{X}, X\right) A$.
- We stress out that the training step is independent of the test set $\tilde{X}$. Thus one can store the values computed during the training step and perform the evaluation step many times with a small computational cost, which is $\mathcal{O}\left(n \cdot m\right)$.

Applying the GPR to the GMWB Contract

GPR to the GMWB Contract : The training step

- We aim to to apply the GPR method to GMWB products to speed up the computation of the price and of the Greeks.
- The modeling process starts by computing a training set $\mathcal{D}$.
- The predictor set X consists of n combinations of the product, market and model parameters.
- For each parameter combination, we compute the price and then, observed data are passed to the GPR algorithm.

The evaluation step

- Once the training step is finished, the model is ready to estimate prices.
- In order to asses the performances of the algorithm, we compare the exact price with the GPR predicted price for the testing set consisting *m* random combinations of the parameters.

Numerical results

Numerical results

The performance of the GPR algorithm is measured in terms of the following indicators:

- RMSE (Root Mean Squared Error)
  $RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\mathcal{Y}\left(\mathbf{x}_i\right) - \mathcal{Y}_{GPR}\left(\mathbf{x}_i\right)\right)^2}$,
- RMSRE (Root Mean Squared Relative Error)
  $RMSRE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\frac{\mathcal{Y}(\mathbf{x}_i) - \mathcal{Y}_{GPR}(\mathbf{x}_i)}{\mathcal{Y}(\mathbf{x}_i)}\right)^2}$,
- MaxAE (Maximum Absolute Error)
  $MaxAE = \max_i \left|\mathcal{Y}\left(\mathbf{x}_i\right) - \mathcal{Y}_{GPR}\left(\mathbf{x}_i\right)\right|$
- MaxRE (Maximum Relative Error) $MaxRE = \max_i \left|\frac{\mathcal{Y}(\mathbf{x}_i) - \mathcal{Y}_{GPR}(\mathbf{x}_i)}{\mathcal{Y}(\mathbf{x}_i)}\right|$.

GMWB

The range of the parameters for a GMWB contract when considering the Heston Hull-White model. (dimension $d = 11$)

Name	Symbol	Range	Name	Symbol	Range
Premium	P	100	Initial vol.	v_0	$[0.02, 0.10]$
Maturity	T	10 years	Rate of m.r.	k_v	$[1.40, 2.60]$
Initial i.r.	r_0	$[0.01, 0.03]$	Long run vol.	θ_v	$[0.02, 0.10]$
Rate of m.r.	k_r	$[0.05, 0.75]$	Volatility of vol.	ω_v	$[0.45, 0.75]$
Volatility of i.r	ω_r	$[0.005, 0.10]$	Correlation	ρ_v	$[-0.70, -0.40]$
Correlation	ρ_r	$[0.05, 0.35]$	Fees	α	$[0.00, 0.10]$
			Penalty	κ	$[0.00, 0.20]$

GMWB pricing results (Heston Hull-White model)

Size of training set		1250	2500	5000	10000	20000
Out-of-sample prediction	RMSE	7.49e-02	5.91e-02	4.67e-02	4.31e-02	3.78e-02
	RMSRE	7.33e-04	5.79e-04	4.63e-04	4.23e-04	3.73e-04
	MaxAE	8.10e-01	6.28e-01	4.68e-01	3.06e-01	2.28e-01
	MaxRE	7.69e-03	5.96e-03	4.51e-03	2.94e-03	2.23e-03
Speed-up		$\times$420000	$\times$210000	$\times$110000	$\times$52000	$\times$26000

Delta Computation (Heston Hull-White model)						
Out-of-sample prediction	RMSE	2.38e-03	1.86e-03	1.47e-03	1.19e-03	1.04e-03
	RMSRE	1.08e-02	7.86e-03	7.02e-03	5.63e-03	4.17e-03
	MaxAE	2.72e-02	1.83e-02	1.92e-02	1.27e-02	1.15e-02
	MaxRE	4.13e-01	3.41e-01	3.22e-01	2.33e-01	2.15e-01
Speed-up		$\times$410000	$\times$200000	$\times$100000	$\times$52000	$\times$26000

No-arbitrage fee Computation (Heston Hull-White model)					
Size of training set	1250	2500	5000	10000	20000
RMSE	1.02e-03	8.33e-04	7.19e-04	6.15e-04	5.62e-04
RMSRE	2.32e-02	1.91e-02	1.69e-02	1.47e-02	1.42e-02
MaxAE	1.52e-02	1.64e-02	1.51e-02	1.58e-02	1.12e-02
MaxRE	3.31e-01	2.72e-01	2.39e-01	2.33e-01	1.84e-01
Speed-up	$\times$410000	$\times$200000	$\times$100000	$\times$52000	$\times$26000

A last comparison

Heston Hull-White model (with Hybrid PDE)				
Time and space steps	125 × 125	250 × 250	500 × 500	1000 × 1000
Price	102.31 (1.2e+0)	102.33 (1.6e+1)	102.34 (2.0e+2)	102.33 (3.0e+3)
Delta	0.3225 (1.2e+0)	0.3245 (1.6e+1)	0.3252 (2.0e+2)	0.3255 (3.0e+3)
No-arbitrage fee	701.73 (1.1e+1)	703.59 (9.8e+2)	703.68 (1.2e+3)	703.80 (1.8e+4)

Training time (Heston Hull-White model)					
Size of training set	1250	2500	5000	10000	20000
Price	49	154	185	188	3219
Delta	32	110	128	142	3144

GPR prediction (Heston Hull-White model)					
Size of training set	1250	2500	5000	10000	20000
Price	102.34 (8.5e−5)	102.33 (2.2e−4)	102.33 (4.6e−4)	102.33 (6.8e−4)	102.33 (1.3e−3)
Delta	0.3242 (8.4e−5)	0.3245 (1.9e−4)	0.3245 (4.2e−4)	0.3245 (7.4e−4)	0.3246 (1.4e−3)
No-arbitrage fee	702.34 (1.8e−2)	704.15 (1.9e−2)	704.05 (2.5e−2)	704.29 (2.8e−2)	704.71 (3.4e−2)

Conclusion

Conclusion

- In this talk we have presented how Gaussian Process Regression can be applied in the insurance field to address the problem of pricing with stochastic volatility and stochastic interest rate.
- The computational time is considerably reduced as the greater computational effort is carried out during the training phase, which must be performed only once.
- Computing a single prediction is very fast and has a linear cost in the number of input observations.
- The same approach may be applied to all types of insurance contracts for different models.
- Machine Learning seems to be a very promising and interesting tool for insurance risk management.

Literature

- C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. the MIT Press, 2006.
- J. De Spiegeleer, D. B. Madan, S. Reyners, and W. Schoutens. *Machine learning for quantitative finance: fast derivative pricing, hedging and fitting*. Quantitative Finance 2018.
- L. Goudenège, A. Molent, and A. Zanette. *Gaussian Process Regression for Pricing Variable Annuities with Stochastic Volatility and Interest Rate*. IN: ARXIV:1903.00369
- **L.Goudenege A.Molent A.Zanette 2019 Machine Learning for Pricing American Options in High Dimension.** IN: ARXIV:1903.11275

- We present an efficient method to compute the price of American basket options, based on Machine Learning and Monte Carlo simulations.
- Specifically, the options we consider are written on a basket of assets, each of them following a Black-Scholes dynamics.
- The method we propose is a backward dynamic programming algorithm which considers a finite number of uniformly distributed exercise dates.

- The value of the option is computed as the maximum between the exercise value and the continuation value, which is approximated via Gaussian Process Regression.
- Specifically, we consider a finite number of points, each of them representing the values reached by the underlying at a certain time.
- First of all, we compute the continuation value only for these points by means of Monte Carlo simulations and then we employ Gaussian Process Regression to approximate the whole continuation value function.
- Numerical tests show that the algorithm is fast and reliable and it can handle also American options on very large baskets of assets, overcoming the problem of the curse of dimensionality.

American options in the multi-dimensional BS model

The underlying process

Let $\mathbf{S} = (\mathbf{S}_t)_{t \in [0,T]}$ denote the d-dimensional underlying process. Under the risk neutral measure, such a model is given by the following equation

$$dS_t^i = r\, S_t^i\, dt + \sigma_i\, S_t^i\, dW_t^i, \qquad i = 1, \ldots, d,$$

with $\mathbf{S}_0 = (s_{0,1}, \ldots, s_{0,d}) \in \mathbb{R}_+^d$ the spot price, r the (constant) spot rate, $\sigma = (\sigma_1, \ldots, \sigma_d)$ the vector of volatilities, $\mathbf{W}$ a d-dimensional correlated Brownian motion and ρ_{ij} the instantaneous correlation coefficient between W_t^i and W_t^j.

Payoff

Moreover, let $\Psi(\mathbf{S}_T)$ denote the cash-flow associated with the option.

Price

Thus, the price at time t of an American option having maturity T and payoff function $\Psi : \mathbb{R}^d_+ \to \mathbb{R}$ is then

$$v(t, \mathbf{x}) = \sup_{\tau \in \mathcal{T}_{t,T}} \mathbb{E}_{t,\mathbf{x}} \left[e^{-r(\tau - t)} \Psi(\mathbf{S}_\tau) \right], \tag{15}$$

where $\mathcal{T}_{t,T}$ stands for the set of all the stopping times taking values on $[t, T]$ and $\mathbb{E}_{t,\mathbf{x}}[\cdot]$ is the expectation given all the information at time t and assuming $\mathbf{S}_t = \mathbf{x}$.

Machine Learning Monte Carlo algorithm for American options

Approximating the price

We approximate the price of an American option with the price of a Bermudan option on the same basket. Specifically, let N be the number of time steps and $\Delta t = T/N$ the time increment. The discrete exercise dates are $t_n = n\,\Delta t$, as $n = 0, 1, \ldots, N$. If $\mathbf{x}$ represents the vector of the underlying prices at the exercise date t_n, then the value of the option is given by

$$v\left(t_n, \mathbf{x}\right) = \max\left(\Psi\left(\mathbf{x}\right), E_{t_n,\mathbf{x}}\left[e^{-r\Delta t} v\left(t_{n+1}, \mathbf{S}_{t_{n+1}}\right)\right]\right). \tag{16}$$

The set X

In order to compute the expectation in formula (16), we consider a set X of P points whose coordinates represent certain possible values for the underlyings:

$$X = \{\mathbf{x}^p = (x_1^p, \ldots, x_d^p), p = 1, \ldots, P\} \subset \mathbb{R}^d. \tag{17}$$

The sets $\tilde{X}^p$

First of all, we compute $v(t_n, \mathbf{x})$ only for $\mathbf{x} \in X$. This is done by a Monte Carlo simulation. In particular, for each $\mathbf{x}^p \in X$, we simulate a set $\tilde{X}_p$

$$\tilde{X}^p = \{\tilde{\mathbf{x}}^{p,m} = (\tilde{x}_1^{p,m}, \ldots, \tilde{x}_d^{p,m}), m = 1, \ldots, M\} \subset \mathbb{R}^d \tag{18}$$

of M possible values for $\mathbf{S}_{t_{n+1}}$ according to $\mathbf{S}_{t_n} = \mathbf{x}$.

Exploiting GPR

Let $\tilde{v}\left(t_{n+1}, \mathbf{x}\right)$ be the GPR prediction of $v\left(t_{n+1}, \mathbf{x}\right)$.
Then we have

$$\hat{v}\left(t_n, \mathbf{x}^p\right) = \max\left(\Psi\left(\mathbf{x}^p\right), \frac{e^{-r\Delta t}}{M}\sum_{m=1}^{M}\tilde{v}\left(t_{n+1}, \tilde{\mathbf{x}}^{p,m}\right)\right). \tag{19}$$

The sets X

The choice of the set X is a sensitive question. Let H^p be the p-th point of the Halton quasi-random sequence in $\mathbb{R}^d$ and Φ^{-1} the inverse cumulative distribution of a standard normal distribution. Let us define $h(i,p,t)$ as follows:

$$h(i,p,t) = e^{\left(r-\frac{1}{2}\sigma_i^2\right)t+\sigma_i\sqrt{t}\sum_{j=1}^{d}\Sigma_{d,j}\Phi^{-1}\left(H_j^p\right)}, \tag{20}$$

In order to define X, we set

$$x_i^1 = s_{0,i}, \tag{21}$$

and

$$x_i^p = s_{0,i}\cdot h(i,p-1,T). \tag{22}$$

```
Preprocessing:  compute $\mathbf{x}^p$ and $\tilde{\mathbf{x}}^{p,m}$

Step $N-1$:  shaping of $\tilde{v}\left(t_{N-1},\cdot\right)$:
             $\hookrightarrow$ For $p=1,\ldots,P$ compute
                  $\hat{v}\left(t_{N-1},\mathbf{x}^p\right)=\max\left(\Psi\left(\mathbf{x}^p\right),\frac{e^{-r\Delta t}}{M}\sum_{m=1}^M\Psi\left(\tilde{\mathbf{x}}^{p,m}\right)\right)$
             $\hookrightarrow$ Define the training set $\mathcal{D}=\left\{\left(\mathbf{x}^p,\hat{v}\left(t_{N-1},\mathbf{x}^p\right)\right),p=1,\ldots,P\right\}$
             $\hookrightarrow$ Apply GPR on $\mathcal{D}$ to obtain $\tilde{v}\left(t_{N-1},\cdot\right)$

Step $N-2$:  shaping of $\tilde{v}\left(t_{N-2},\cdot\right)$:
             $\hookrightarrow$ For $p=1,\ldots,P$ compute
                  $\hat{v}\left(t_{N-2},\mathbf{x}^p\right)=\max\left(\Psi\left(\mathbf{x}^p\right),\frac{e^{-r\Delta t}}{M}\sum_{m=1}^M\tilde{v}\left(t_{N-1},\tilde{\mathbf{x}}^{p,m}\right)\right)$
             $\hookrightarrow$ Define the training set $\mathcal{D}=\left\{\left(\mathbf{x}^p,\hat{v}\left(t_{N-2},\mathbf{x}^p\right)\right),p=1,\ldots,P\right\}$
             $\hookrightarrow$ Apply GPR on $\mathcal{D}$ to obtain $\tilde{v}\left(t_{N-2},\cdot\right)$

  $\vdots$  $\leftarrow$ Steps $n=N-3,\ldots,1$ [ replace $N-2$ with $n$ and $N-1$ with $n+1$; ]

 Step 0:  computation of the price:
```

$$\hat{v}\left(0,\mathbf{s}_0\right)=\max\left(\Psi\left(\mathbf{x}^1\right),\frac{e^{-r\Delta t}}{M}\sum_{m=1}^{M}\tilde{v}\left(t_{N-1},\tilde{\mathbf{x}}^{1,m}\right)\right)$$

Numerical Results

In this Section we report some numerical results in order to investigate the effectiveness of the proposed Machine Learning algorithm for pricing American options in the multi-dimensional Black-Scholes model.

We vary the dimension d, considering $d = 2, 5, 10, 20, 40$ and 100.

We consider the following parameters $T = 1$, $S_i = 100$, $K = 100$, $r = 0.05$, constant volatilities $\sigma_i = 0.2$, constant correlations $\rho_{ij} = 0.2$ and $N = 10$ exercise dates.

Moreover, we consider $P = 250, 500$ or 1000 points and $M = 10^3, 10^4$ or 10^5 Monte Carlo simulations.

Numerical Results

In particular, we consider the following payoff examples:

- Arithmetic basket Put

$$\Psi(\mathbf{S}_T) = \left(K - \frac{1}{d}\sum_{i=1}^{d} S_T^i\right)_+ ,$$

- Geometric basket Put

$$\Psi(\mathbf{S}_T) = \left(K - \left(\prod_{i=1}^{d} S_T^i\right)^{\frac{1}{d}}\right)_+ ,$$

- Call on the Maximum on d-assets

$$\Psi(\mathbf{S}_T) = \left(\max_{i=1\ldots d} S_T^i - K\right)_+ .$$

Geometric basket Put

- In this case, it is possible to reduce the problem of pricing in the d-dimensional model to a one dimensional American Put option in the Black-Scholes model with opportune parameters
- A fully reliable benchmark can be computed by using the CRR algorithm with 1000 steps.

Geometric basket Put

	P	250			500			1000				
d	M	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	Tree	Bm
2		4.59 (8)	4.58 (27)	4.57 (236)	4.59 (24)	4.57 (115)	4.57 (808)	4.59 (53)	4.57 (515)	4.57 (3199)	4.62	4.62
5		3.46 (8)	3.43 (27)	3.43 (244)	3.45 (21)	3.42 (116)	3.42 (903)	3.44 (61)	3.41 (380)	3.41 (3025)	3.44	3.45
10		2.96 (8)	2.93 (28)	2.93 (256)	2.94 (17)	2.92 (103)	2.92 (1121)	2.93 (65)	2.90 (384)	2.90 (2982)		2.97
20		2.74 (9)	2.69 (52)	2.68 (276)	2.74 (24)	2.69 (104)	2.69 (870)	2.76 (59)	2.70 (388)	2.70 (3577)		2.70
40		2.60 (12)	2.50 (53)	2.50 (336)	2.63 (25)	2.53 (123)	2.53 (873)	2.67 (61)	2.57 (409)	2.57 (4681)		2.56
100		2.49 (14)	2.33 (80)	2.32 (507)	2.52 (25)	2.35 (151)	2.34 (1246)	2.58 (68)	2.40 (432)	2.40 (4288)		2.47

Table: Results for a Geometric basket Put option (and cp times in s).

Geometric basket Put

- The proposed method is accurate and stable and the computational time can be very small.
- The computer processing time increase little with the size of the problem and this makes the method particularly effective when the dimension of the problem is high
- This is due to the fact that the dimension affects significantly only the computational time of the Monte Carlo step while the GPR step is only minimally distressed

Arithmetic basket Put option

- In this case we have no method to obtain a fully reliable benchmark.
- However, for small values of d, a reference price can be obtained by means of a multidimensional tree method
- As in the case of the Geometric basket Put, the values do not change much with respect to the number P of points and to the number M of Monte Carlo simulations.

Arithmetic basket Put

P	250			500			1000			
d M	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	Tree
2	4.39 (8)	4.37 (27)	4.37 (235)	4.39 (16)	4.38 (100)	4.37 (881)	4.39 (46)	4.37 (446)	4.37 (3263)	4.42
5	3.15 (8)	3.13 (27)	3.12 (245)	3.12 (23)	3.10 (101)	3.10 (823)	3.12 (54)	3.09 (377)	3.09 (4491)	3.15
10	2.64 (9)	2.62 (28)	2.62 (260)	2.62 (16)	2.59 (105)	2.59 (921)	2.61 (47)	2.58 (385)	2.58 (3486)	
20	2.34 (9)	2.28 (52)	2.28 (281)	2.37 (23)	2.32 (105)	2.32 (975)	2.44 (67)	2.38 (525)	2.38 (3548)	
40	2.09 (10)	1.99 (54)	1.99 (341)	2.20 (29)	2.09 (126)	2.09 (966)	2.28 (80)	2.18 (410)	2.17 (3774)	
100	1.96 (14)	1.80 (79)	1.80 (504)	2.03 (20)	1.85 (134)	1.84 (1457)	2.12 (59)	1.93 (439)	1.92 (4113)	

Table: Results for a Arithmetic basket Put option (and cp times in s).

Call on the Maximum option

- Similar to the case of the Arithmetic basket Put
- In this case the payoff is an unbounded function

Call on the Maximum option

	P	250			500			1000			
d	M	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	Tree
2		16.82 (8)	16.86 (28)	16.87 (242)	16.81 (23)	16.85 (139)	16.86 (882)	16.81 (47)	16.85 (377)	16.86 (3443)	16.86
5		27.05 (8)	27.12 (28)	27.13 (242)	27.09 (21)	27.16 (141)	27.17 (906)	27.12 (45)	27.19 (385)	27.20 (3426)	27.20
10		35.73 (8)	35.82 (29)	35.82 (260)	35.12 (18)	35.19 (105)	35.19 (917)	35.07 (63)	35.16 (402)	35.17 (3201)	
20		43.13 (9)	43.22 (51)	43.22 (276)	42.83 (18)	42.93 (105)	42.93 (939)	42.65 (64)	42.75 (398)	42.76 (3618)	
40		54.92 (10)	55.00 (54)	55.01 (340)	51.04 (8)	51.13 (126)	51.15 (978)	50.66 (63)	50.68 (419)	50.70 (3691)	
100		57.70 (14)	57.71 (78)	57.71 (503)	54.38 (28)	54.41 (157)	54.41 (1293)	59.60 (82)	59.68 (516)	59.69 (4838)	

Table: Results for a Call on the Maximum option (and cp times in s).

Conclusions

- Numerical results show that the method is accurate and fast for baskets including up to 100 assets.
- Furthermore, the computation time is shortly growing with respect to the dimension of the basket.
- Moreover, we stress out that the algorithm is completely parallelizable and therefore the computing time can be significantly reduced.