CEGEN algorithm, application on SABR model

Mohamed Raed Blel*,[1]

[1]Laboratoire de Probabilités Statistiques et Modélisation, LPSM

April 23, 2024

Contents

1 Introduction

In our study, our primary objective is to utilize the CEGEN algorithm [4] for the calibration of various models. The notable advantage of this method lies in its ability to provide theoretical results and exhibit favorable performance in high-dimensional scenarios. In this paper, we present a selection of results obtained from the calibration of two specific models: the Black-Scholes model, which involves one dimension, and the SABR model, which involves two dimensions. In a second work

*Corresponding author: www.linkedin.com/in/mohamed-raed-blel-link

i propose a novel neural network method for the calibration of GARCH parameters by developing first a better neural network for the prediction of the parameters and second a neural network that provides all the parameters with high performance contrarily to the work in [2] third by modifying the loss and adding a penalization term we insure that the condition to obtain an ergodic solution is also satisfied by the predicted parameters. this will be useful for the calibration of several problem such the SABR model and the implied volatility.

In a second study, I introduce an novel neural network approach for calibrating GARCH parameters. This method involves three key advancements: first, the development of an enhanced neural network model for predicting the parameters, second, the creation of a comprehensive neural network capable of accurately providing all the parameters not only one and third, i introduce a modified loss function with a penalization term, thereby ensuring that the predicted parameters satisfy the condition for obtaining an ergodic solution. This approach marks a significant departure from previous works, such as that described in [2]. The proposed method not only demonstrates superior performance but also offers versatility in addressing various problems, including the calibration of models like the SABR model and the estimation of implied volatility.

1.1 Completed work

In this paragraph we summarize the different accomplished points:

- An implementation of the CEGEN algorithm [4] using python language.
- We have developed a robust numerical implementation of the Bures metric ([1], [3]), addressing several challenges to ensure convergence. One of the key issues we encountered was the need for a well-defined and compact partition to avoid singularities. To overcome this, we employed a partition by quantiles approach. Another significant aspect of our implementation involved utilizing the singular value decomposition technique to handle matrix square roots and related conditional problems. By tackling these challenges, we have achieved a reliable and effective implementation of the Bures metric.
- We have programmed entirely a parameterized neural network architecture where for certain parameters, we have utilized the sigmoid activation function, while for others, the cotangent activation function has been applied. This combination of activation functions allows us to achieve stability and optimize the performance of the neural network in our specific context.
- The implemented algorithm was tested on both the Black-Scholes (BS) model and the SABR model. While the main authors of the algorithm did not originally test the SABR model, we encountered additional convergence issues

due to the direct dependency of the forward rate on the volatility process in this model. To address this challenge, we made adaptations to the algorithm by incorporating a partitioning technique that alternates the partitioning and the minimization function depending on the variable. This approach helped to improve the convergence of the algorithm for the SABR model.

Remark

- The current method is particularly useful when dealing with a basket portfolio that involves dependent noises.

Remarks

- Our objective is to calibrate the parameters (nu, beta, and rho) of the SABR model using the temporal series of the forward rate. Notably, we aim to achieve this calibration without relying on the availability of the volatility temporal series, which is not directly observable in practice. By focusing only on the forward rate temporal series, we aim to relate the GARCH volatility to the volatility of the SABR model and use the results obtained in section 2.

- An alternative avenue for calibrating the SABR model involves the utilization of standard methods based on GARCH volatility approximation. An intriguing aspect is the potential comparison between these conventional methods and our proposed approach. Investigating and contrasting both methods could provide valuable insights into the strengths and limitations of each, contributing to a comprehensive understanding of SABR model calibration.

2 The CEGEN Method

The primary objective of this method is to acquire an empirical probability distribution that effectively approximates the data distribution. In pursuit of this goal, we focus our framework on Ito processes due to their ability to provide robust theoretical results, which we will discuss in the subsequent section.

2.1 Context and theoretical results

The context

Let be the following stochastic differential equation:

$$dX_t = b_X(t, X_t)dt + \sigma_X(t, X_t)dW_t, \quad \forall t \in [0, T], \tag{1}$$

where $b_X : \mathbb{R} \times \mathbb{R}^d \longrightarrow \mathbb{R}^d$ is the drift term and $\sigma_X : \mathbb{R} \times \mathbb{R}^d \longrightarrow \mathcal{M}_{d\times d}$ the volatility term and W is a d-dimensional Brownian motion. We suppose that the parameters b_X and σ_X satisfy the usual Lipschitz conditions and thus ensuring existence eand uniqueness of the solution. Let $\{t_i\}_{0\leq i\leq N}$ a discretization of $[0,T]$, then the Euler Maruyama scheme gives,

$$X_{t_i+\Delta t} = X_{t_i} + b_X(t_i, X_{t_i})\Delta t + \sigma_X(t_i, X_{t_i})\Delta W_{t_i}, \tag{2}$$

where $\Delta W_{t_i} \sim \mathcal{N}(0, \Delta t I_d)$ and are iid for all $\{t_i\}_{0\leq i\leq N}$. We aim to replace this procedure by a process defined by,

$$Y^\theta_{t_i+\Delta t} = Y^\theta_{t_i} + b^\theta_Y(t_i, Y^\theta_{t_i})\Delta t + \sigma^\theta_Y(t_i, Y^\theta_{t_i})Z_{t_i}, \tag{3}$$

where $Z_{t_i} \sim \mathcal{N}(0, \Delta t I_d)$ and b^θ_Y and σ^θ_Y are θ-parametrized functions approximated by neural networks, such that,

$$\mathcal{L}(X_{t_i+\Delta t}|_{X_{t_i}}) \simeq \mathcal{L}(Y^\theta_{t_i+\Delta t}|_{Y^\theta_{t_i}}) \quad \text{for all} \quad \{t_i\}_{0\leq i\leq N}.$$

Hence, the objective is to learn b^θ_Y and σ^θ_Y such that the distribution of X and Y^θ are close.

The metric

The loss function used in this problem is based on the Bures metric and is defined by,

$$\mathcal{W}_2^2(\mathcal{L}(X), \mathcal{L}(Y)) = ||\mathbb{E}[X] - \mathbb{E}[Y]||^2 + \mathcal{B}^2(Var(X), Var(Y)), \tag{4}$$

where $\mathcal{B}$ is the Bures metric. This metric is interesting as:

- **If X and Y are gaussian**, the considered metric is the definition of the **Wasserstein 2 distance**.
- We can compute exactly the wasserstein 2 distance from the Bures formulation:
$$\mathcal{B}^2(A, B) = Tr(A) + Tr(B) - 2Tr(\sqrt{A^{\frac{1}{2}}BA^{\frac{1}{2}}}),$$
for positive definite matrices A and B.
- **Whenever $\mathcal{L}(X)$ and $\mathcal{L}(Y)$ coincide in $\mathcal{W}_2$ then the drift and the volatility terms coincide as well.**

Loss function

To build the generator we create at each time t_i a partition $(I_k)_{k\leq N_K}$ of the union of supports of X_{t_i} and $Y^\theta_{t_i}$. For a given batch of samples, $\mathcal{L}(Y^\theta_{t_{i+1}}|Y^\theta_{t_i} \in I_k)$ is then approximated by extracting $Y^\theta_{t_i}$ such that $Y^\theta_{t_i} \in I_k$ and evaluated with the $\mathcal{W}_2$ metric in order to approach $\mathcal{L}(X_{t_{i+1}}|X_{t_i} \in I_k)$ where X_{t_i} is extracted also such that $X_{t_i} \in I_k$. Hence the loss function is defined as,

$$l(X, Y^\theta) = \sum_{i=0}^{N-1} \sum_{k=1}^{N_K} \mathcal{W}_2^2(\mathcal{L}(X_{t_{i+1}}|X_{t_i} \in I_k), \mathcal{L}(Y^\theta_{t_{i+1}}|Y^\theta_{t_i} \in I_k)). \tag{5}$$

Theoretical result

The expression of the metric yields the following results in terms of controlling the error between the predicted drift and the actual drift, as well as the error between the predicted volatility and the actual volatility.

Proposition: Assume that $\sigma^2_X(t_i, .)$ and $\sigma^2_{Y^\theta}(t_i, .)$ are strictly positive and, together with $b_X(t_i, .)$ and $b_{Y^\theta}(t_i, .)$, K-Lipschitz in their second coordinate. For $t_i \in \mathcal{T}$, let $(I_k)_k$ be a regular partition covering $Supp(X_{t_i}) \cup Supp(Y_{t_i})$ with mesh size Δx and let $\epsilon > 0$.
If $\mathcal{W}_2^2(\mathcal{L}(X_{t_i+\Delta t}|X_{t_i} \in I_k), \mathcal{L}(Y^\theta_{t_i+\Delta t}|Y^\theta_{t_i} \in I_k)) \leq \epsilon^2$ for any k, then, for z in the partition we have,

$$||b_X(t_i, z) - b_{Y^\theta}(t_i, z)||_2 \leq \frac{\epsilon + \Delta x}{\Delta t} + 2K\Delta x, \tag{6}$$

Furthermore if d=1,

$$||\sigma_X(t_i, z) - \sigma_{Y^\theta}(t_i, z)||_2 \leq \frac{\epsilon}{\sqrt{\Delta t}} + 2K\Delta x. \tag{7}$$

and, when $d \geq 1$ and $Tr(\sigma^2_X(t_i, z)) = Tr(\sigma^2_{Y^\theta}(t_i, z)) = \alpha$, we have,

$$||\sigma_X(t_i, z) - \sigma_{Y^\theta}(t_i, z)||_2 \leq \sqrt{\frac{2\alpha}{\Delta t}}\epsilon + 2K\Delta x. \tag{8}$$

Key elements of the proof
To obtain these results we only need to observe that

$$X_{t_{i+1}}|(X_{t_i} = z) \sim \mathcal{N}(z + b_X(t_i, z)\Delta t, \sigma^2_X(t_i, z)\Delta t)$$

and,

$$Y^\theta_{t_{i+1}}|(Y^\theta_{t_i} = z) \sim \mathcal{N}(z + b_{Y^\theta}(t_i, z)\Delta t, \sigma^2_{Y^\theta}(t_i, z)\Delta t)$$

then considering the K-Lipschitz condition and applying the condition

$$\mathcal{W}_2^2(\mathcal{L}(X_{t_i+\Delta t}|_{X_{t_i}\in I_k}), \mathcal{L}(Y^\theta_{t_i+\Delta t}|_{Y^\theta_{t_i}\in I_k})) \leq \epsilon^2$$

with the reverse triangular inequality we end up with the required results.

- The proof is checked and the inequalities are correct.
- We need to control the ratio by considering a fine partition, which increases the computational cost. In the same time we are constrained to the Wasserstein error ϵ that needs an enough high sampling number to estimate correctly.
- In order to ensure accurate control over the ratio $\frac{\Delta x}{\Delta t}$, we must employ a fine partition, which inevitably increases the computational cost. However, we are simultaneously faced with the constraint of the Wasserstein error ϵ, which requires a sufficiently large number of samples for accurate estimation. Balancing these considerations becomes crucial as we strive to achieve both precise control over the ratio $\frac{\Delta x}{\Delta t}$ and accurate estimation of the Wasserstein error ϵ, while being mindful of the associated computational demands.

The algorithm

In this paragraph, we introduce the CEGEN algorithm 1. At each time step t_i, we sample m observations of $X_{t_{i+1}}$ and $Y^\theta_{t_{i+1}}$ according to the corresponding equation. Subsequently, a partition function is applied using the quantiles method to create N_K subdivisions, from which the mesh size Δx is determined. For each subdivision I_k, we estimate the Wasserstein distance $\mathcal{W}_2^2(\mathcal{L}(X_{t_i+\Delta t}|X_{t_i} \in I_k), \mathcal{L}(Y^\theta_{t_i+\Delta t}|Y^\theta_{t_i} \in I_k))$. Finally, the backpropagation is performed on the overall loss function given by

$$l(X, Y^\theta) = \sum_{i=0}^{N-1} \sum_{k=1}^{N_K} \mathcal{W}_2^2(\mathcal{L}(X_{t_{i+1}}|X_{t_i} \in I_k), \mathcal{L}(Y^\theta_{t_{i+1}}|Y^\theta_{t_i} \in I_k)).$$

Algorithm 1 Algorithm CEGEN.

Input: $\mathcal{D}$ samples of X, m batch size, K Nb of subdivisions, γ learning rate
Initialize: θ (randomly picked)
while Not converged **do**
  for $t_i = 0...T$ **do**
    Sample m observations (x_{t_i+1}) from of X_{t_i+1}
    Sample $z \sim \mathcal{N}(0, I_D \Delta t)$
    $y_{t_i+1} \leftarrow y_{t_i} + g_\theta^b(t_i, y_{t_i})\Delta t + g_\theta^\Sigma(t_i, y_{t_i})z$
    $I_K \leftarrow K$ subdivisions of $\text{Supp}(X_{t_i}) \cup \text{Supp}(Y_{t_i})$
    for $k = 0...K$ **do**
      $\ell_{t_i+1,k} \leftarrow \mathcal{W}_2^2(\mathcal{L}(x_{t_i+1}|x_{t_i} \in I_k), \mathcal{L}(y_{t_i+1}|y_{t_i} \in I_k))$
    end for
  end for
  $\theta = \theta - \gamma \nabla_\theta \sum_{t_i=1}^{T-1} \sum_{k=1}^{K} \ell_{k,t_i+1}$
end while
Output: y

Figure 1: CEGEN algorithm

Neural Network architecture

We employ a neural network as the generator, which have entirely programmed and which consists of three layers with four times the dimension of the data for each layer. To ensure appropriate output ranges, we apply the sigmoid function at the end of the network for the parameters ν and β, while the cotangent function is utilized for ρ. The Adam optimizer is employed for training the neural network, with a learning rate (l_r) set to 10^{-4}. The factors β_1 and β_2, which correspond to gradients and second moments of gradients, are respectively assigned values of 0.5 and 0.99.

2.2 Application to a fully observed one-dimensional Black Scholes model

2.2.1 The model

The Black Scholes model is described by the following equation,

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{9}$$

with initial values S_0. In this equation, S_t is the spot price, σ is the volatility of the return, μ the drift term of the return and W_t Brownian motion. We are

looking for calibrating the following parameters σ and μ using the temporal series of the spot price.

Predictive model

The predictive model is given by the following discretized Euler Maruyama equation:

$$S_{t_{i+1}} = S_{t_i} + b^\theta(t_i, S_{t_i})\Delta t + \sigma^\theta(t_i, S_{t_i})\Delta W_{t_i}, \tag{10}$$

Such that the predictive drift term $b^\theta(t, S_t)$ tries to approximates the term μS_t and the predictive volatility term $\sigma^\theta(t, S_t)$ tries to approximate the term σS_t for each t.

2.2.2 Numerical Results

Calibration phase for the Black and Scholes model

During the calibration phase, we simulate trajectories over the interval $[0, T] = [0, 1/4]$ using a time step of $\Delta t = 2.2510^{-2}$. We utilize a total of $M = 10000$ trajectories, with an initial condition of $S_0 = 1$. The partitioning of the space is performed by dividing it into four parts using the quantile method.
For the training process, we set a batch size of 300 and conduct 10 epochs.
The obtained results include the Wasserstein error, as well as the estimated values for the volatility σ and the drift term μ. In figure 2, we observe a decrease of the loss function during the training phase to achieve convergence at 10^{-6}.

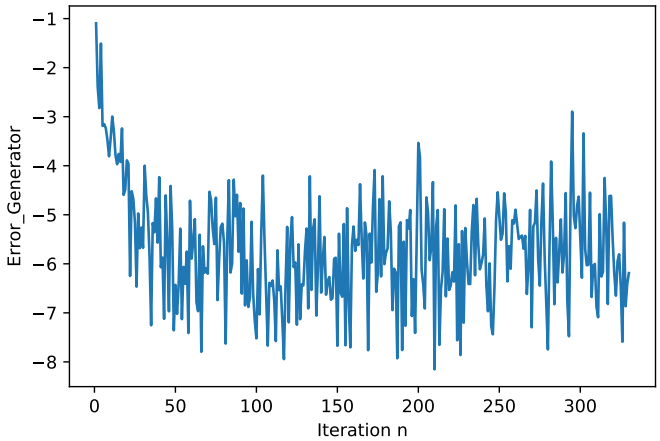

Figure 2: Generator error

In figure 3, we observe that the volatility term converges to a predicted volatility around 0.28 while the actual volatility is equal to 0.3.

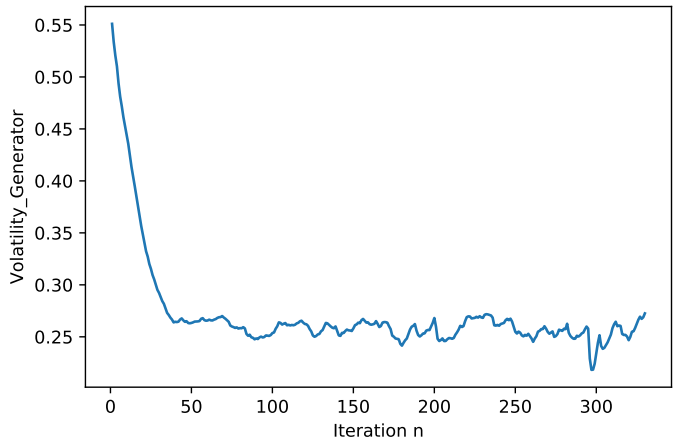

Figure 3: Prediction of the volatility

In figure 4, we observe that the drift term converges to a predicted drift around 0.78 while the actual drift is equal to 0.8.

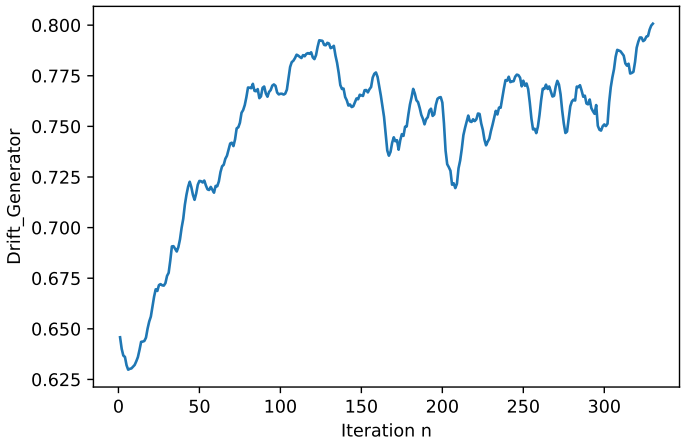

Figure 4: Prediction of the drift

2.3 Application to a fully observed two-dimensional SABR model

This section serves as an intermediate modelization between the one-dimensional model and the partial two-dimensional model where only one dimension is observed while the second dimension remains unknown. However it useful for basket portfolio problems.

2.3.1 The model

The SABR model is described by the following three equations,

$$dF_t = \alpha_t F_t^{\beta} dW_t^1, \tag{11}$$

$$d\alpha_t = \nu \alpha_t dW_t^2, \tag{12}$$

$$\mathbb{E}[dW_t^1 dW_t^2] = \rho dt, \tag{13}$$

with initial values F_0 and α_0. In these equations, F_t is the forward rate, α_t is the volatility, and W_t^1 and W_t^2 are correlated Brownian motions, with correlation ρ. We are looking for calibrating the following parameters,

- ν the volatility of the volatility,
- β the exponent for the forward rate,
- ρ the correlation between the Brownian motions.

Predictive model

The predictive model is given by the following discretized Euler Maruyama equation:

$$F_{t_{i+1}} = F_{t_i} + \sigma_F^{\theta}(t_i, \alpha_{t_i}, F_{t_i}) \Delta W_{t_i}^1 \tag{14}$$

$$\alpha_{t_{i+1}} = \alpha_{t_i} + \sigma_{\alpha}^{\theta}(t_i, \alpha_{t_i}) \Delta W_{t_i}^2 \tag{15}$$

$$\Delta W_{t_i}^2 = \rho^{\theta} \Delta W_{t_i}^1 + \sqrt{1 - (\rho^{\theta})^2} \Delta W_{t_i}^1, \tag{16}$$

such that the predictive volatility term $\sigma_F^{\theta}(t, \alpha_t, F_t)$ approximates $\alpha_t F_t^{\beta}$ and the predictive volatility term of the second equation $\sigma_{\alpha}^{\theta}(t, \alpha_t)$ approximates $\nu \alpha_t$ for each t where the Brownian increment $\Delta W_{t_i}^2$ is taken under the equation (16). Hence we look to calibrate the parameters β, ν and ρ by β^{θ}, ν^{θ} and ρ^{θ} following the method given in algorithm 1 such that,

$$\beta^{\theta} = \frac{ln(\frac{\sigma_F^{\theta}(t_i, \alpha_{t_i}, F_{t_i})}{\alpha_t})}{ln(F_{t_i})},$$

$$\nu^{\theta} = \frac{\sigma_{\alpha}^{\theta}(t_i, \alpha_{t_i})}{\alpha_{t_i}}.$$

2.3.2 Numerical results

Calibration phase for the SABR model

In the calibration phase we run our trajectories on $[0, T] = [0, 1/4]$ with time step Δt equal $\Delta t = 2.2510^{-2}$, the number of trajectories used is equal to $M = 10000$, the parameters $\alpha_0 = 0.36$, and $F_0 = 100$ and the partition space is divided in 4 parts using the quantile method.
We set a number of batch size equal to 300 and a number of epochs equal to 22. We obtain the following results for respectively the parameters β, ν and ρ. In figure 5, we observe that the parameter β^{θ} converges to a predicted β around 0.8.

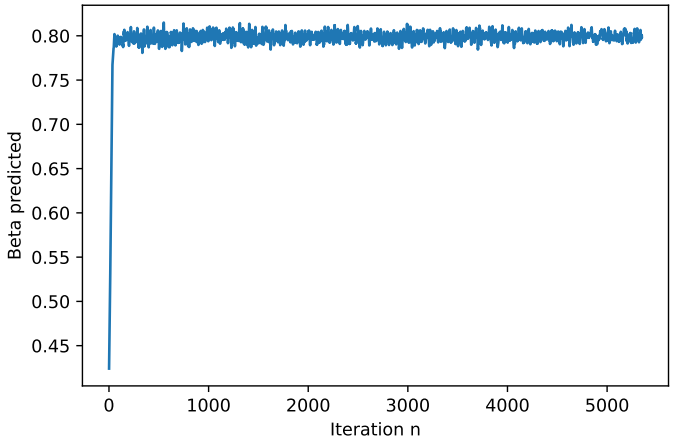

Figure 5: Prediction of Beta

In figure 6, we observe that the parameter ν converges to a predicted ν around 0.21 while the actual ν is equal to 0.2.

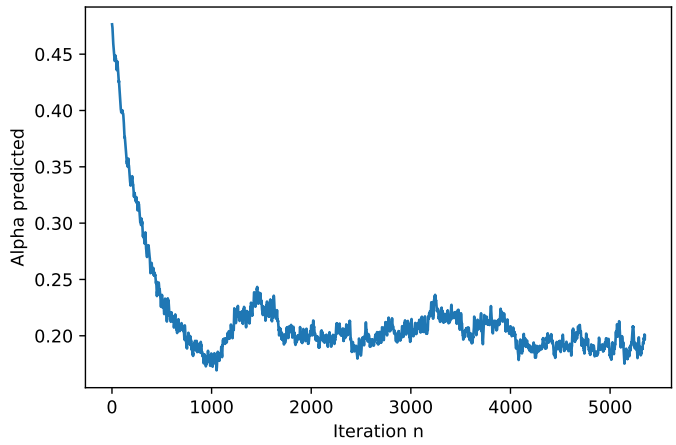

Figure 6: Prediction of ν

In figure 7, we observe that the parameter ρ is quite unstable and converges to a predicted rho around -0.2 while the actual rho parameter is equal to -0.3.

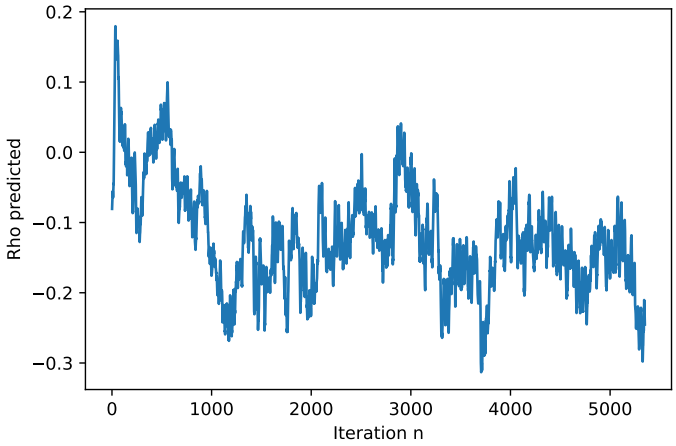

Figure 7: Prediction of Rho

Hence we succeeded to calibrate the SABR model in two dimensions and the Black and Scholes model in one dimension using a neural network method.
Remark:

- The number of trajectories $M = 10000$ is huge and one can cut the main temporal series in several trajectories to get the calibration.

- The code uses some hyper parameters crucial to the convergence of the method, one is the number of partitions used in the quantile method in our case we take it equal to 4.

- Due to the restricted time, we were unable to deeply analyze and compare the sensitivity of convergence when modifying various hyperparameters such as the number of observations or time step.

3 Conclusion and perspectives

Throughout this project, we have demonstrated favorable outcomes in calibrating both the Black-Scholes and SABR models using CEGEN algorithm. The methodology holds the potential for straightforward extension to high-dimensional problems such as basket portfolios involving multiple correlated spot prices. However, our primary focus is to achieve calibration for the SABR model without relying on the temporal series of volatility, which presents a distinct challenge.

To tackle this objective, we decided to approximate the volatility with a GARCH model but using a neural network model to calibrate the parameters of the GARCH model and not standard methods (This is done in the project Garch Neural network calibration).

References

[1] Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the bures-wasserstein distance between positive definite matrices. *arXiv*, 2017.

[2] Luke De Clerk and Sergey Savl'ev. A machine learning search for optimal garch parameters, 2022.

[3] Luigi Malagò, Luigi Montrucchio, and Giovanni Pistone. Wasserstein riemannian geometry of positive definite matrices. *arXiv*, 2018.

[4] Carl Remlinger, Joseph Mikael, and Romuald Elie. Conditional loss and deep euler scheme for time series generation. *arXiv*, 2021.