Transactions on Computational and Applied Mathematics (2024)
Clausius Scientific Press, Canada

DOI: 10.23977/tracam.2024.040114
ISSN 2616-1826 Vol. 4 Num. 1

Optimum Scheme of Feynman-Kac Formula Based on Regression and Monte Carlo Methods

Yidan Feng

College of Science, Northeastern University, Boston, 02120, USA

Keywords: Partial differential equation, Feynman-Kac formula, Brownian Motion, regression

Abstract: This research paper introduces a new computational scheme that uses regression techniques to solve high-dimensional partial differential equations (PDEs) through the Feynman-Kac formula and Monte Carlo Method (MCM) simulations. The conventional approach using interpolation faces significant challenges in high-dimensional spaces, which will cause more complex computations and less accuracy. The proposed method integrates stochastic differential equations (SDEs) and regression analysis to formulate a more efficient and accurate algorithm for computing global solutions. Comparing the traditional interpolation method with the new regression-based approach demonstrates significant improvements in computational efficiency and a reduction in error. The paper meticulously analyzes the error differences, convergence properties, and the practical implications of the regression method in overcoming the limitations faced by interpolation in high-dimensional problem spaces. The results underscore the potential of this new scheme to enhance the accuracy and speed of solving PDEs using the Feynman-Kac and MCM, offering a substantial contribution to the field of numerical analysis and computational mathematics.

1. Introduction

A PDE is an equation that formulates an unknown function when there are two or more independent variables, and it can handle boundary condition problems [1]. It is widely used in math-based fields and can be used to analyze the rate of changes, such as temperature changes in physics, the sensitivity of financial derivatives, etc. However, PDE lacks exact solutions and can only be solved by acquiring numerical solutions [1]. Conventional numerical methods will fail when in high-dimensional problems, which will cause the Curse of Dimensionality [2]. Curse of Dimensionality is a term used to describe phenomena or problems that only occur in high-dimensional problems instead of low-dimensional issues [3]. For example, when dimensionality increases, the volume of the math space will expand so fast that the required data numbers will grow exponentially to get valid results. Therefore, the MCM and Feynman-Kac formula can help to solve high-dimensional problems [4].

In section 2, I will talk about the basis of the modeling, which is how the Feynman-Kac formula will be used in the model. Feynman-Kac formula is a standard method to use when meeting high-dimensional problems, and it is an applicable tool to achieve stochastic solutions from partial differential equations [5]. Based on the Feynman-Kac formula, MCM can transfer numerical problems into statistical problems [6][7]. However, many people will use interpolation methods to construct global solutions [8]. Take [7] and [8] as examples. Several points will be picked out and simulated tens of thousands of times using interpolation methods. The disadvantage of this algorithm is that it will take longer since it will have more points to be computed during the simulation. In parts 3 and 4, I will discuss the error difference between two interpolation and regression, which are the new methods I suggested, and the numerical result of errors. Part 5 will present the convergence of using the new method. Because of the disadvantage of interpolation methods, I suggest a new algorithm using SDE and regression inspired by [9]. In the new algorithm, I will use tens of thousands of points and simulate each once using regression instead of interpolation to build global solutions. This new algorithm can help decrease time-consuming and increase accuracy.

2. Feynman-Kac Formula and Process of Simulation

To elucidate the process of simulating Feynman-Kac formula, let's consider a parabolic PDE expressed by equation (1) that has dependent variable time t, and it can be reduced to Laplace's equation in steady states when $\Delta u = 0$, where Δ is the Laplace operator. This operator is central to Laplace's equation, a widely applied mathematical statement in physics, mainly when the system is in a steady state with no net change—commonly denoted as $\Delta u(x) = 0$. In (1), $u(x,t)$ is continues, and $V(x,t)$ is a potential function.

$$\frac{\partial u(x,t)}{\partial t} + \mu(x,t)\frac{\partial u(x,t)}{\partial x} + \frac{1}{2}\sigma^2(x,t)\frac{\partial^2 u(x,t)}{\partial x^2} - V(x,t)u(x,t) + f(x,t) = 0 \tag{1}$$

Based on equation (1), let's consider a simple PDE with Dirichlet boundary conditions within domain D as listed below in (2)(3).

$$-Au = f \tag{2}$$

$$with\, u = g \tag{3}$$

For every x in D, the solution for u can be acquired as (4).

$$u(x) = \mathbb{E}[g(X_{\tau D}) + \int_0^{\tau D} f(X_s)ds] \tag{4}$$

To get (4), the MCM and the Feynman-Kac formula will form the expected value for stochastic processes, such as the Wiener processes or Brownian motion [10]. The corresponding Stochastic Differential Equation (SDE) for (1) will be (5), which is Ito's lemma. With (5), when the higher order goes infinitesimal to zero, Ito's lemma can be derived. Ito's lemma is helpful in generating many formulas. Here, we can use Ito's lemma to derive the Feynman-Kac formula.

$$dX_t = \mu(X,t)dt + \sigma(X,t)dB_t \tag{5}$$

Where dB_t is the increment of Brownian Motion when there is a Δt as time interval. Firstly, the values of the drift (μ) and the diffusion (σ) terms are needed to be calculated. Then, dB_t can be generated by Δt. Usually, dB_t can be simulated by using $\sqrt{\Delta t}Z$ where Z is s random sample of the standard normal distribution. Then, the process can be renewed from X_{t_i} to $X_{t_{i+1}}$ by (6), while t increased by 1 Δt.

$$X_{t_{i+1}} = X_{t_i} + \mu_i(X,t)dt + \sigma_i(X,t)dB_{t_i} \tag{6}$$

For each t_i, this process will be repeated starting with X_0 until the end of time limit. To obtain the statistical properties of a random process, it is usually necessary to repeat the above steps several times to generate multiple paths. Considering the Dirichlet boundary value problem, the PDE can be converted to the expected value of the corresponding stochastic process, which represents the dynamics of a continuous-time stochastic process and will help acquire the solution $u(x,t)$ (4). In (4), τ_D is the first exit time from domain D in Brownian motion, and $(X_t)_{t\geq 0}$ here will be a stochastic process solution.

After deriving Feynman-Kac solutions, interpolation is commonly applied to simulate those solutions and get a global solution. However, I claim using regression instead inspired by [11] and [12]. This study validates the efficiency of regression-based Monte Carlo simulations in solving PDEs, highlighting their optimal convergence rate under Lipschitz conditions and their lessened susceptibility to the curse of dimensionality. Based on [11] and [9], I think using regression is more efficient and accurate simultaneously than using interpolation.

3. Traditional Interpolation Method and Regression Algorithm to Solve Feynman-Kac Formula

The primary interpolation method will need a couple of steps. Firstly, it needs to pick n points in a range D. These points will be estimated using the Feynman-Kac formula by N times and get the mean values of the approximations of all N times. Finally, the global solution can be found after using interpolation to connect those mean values.

However, the two biggest problems of using interpolation are that it will give rise to a considerable loss of time due to a large number of simulations and more complex computation in a higher dimensional problem; it will also have a bigger error than the new algorithm. If we use this method in a 2-dimension condition, we will need at least n points for the x-axis and n points for the y-axis, making it n^2 points, and the number of simulations will be $N \times n^2$ times. The process will take a long time to compute because of the number of processes and more complex computation for 2-dimensional problems. Instead, the new algorithm will use $N \times n^2$ points between 0 and 1. Every point will be put into simulation around once, making it $N \times n^2$ times of simulations in total as well. With the same number of simulations, the time consuming of interpolation will be a lot longer than regression.

Will making computation faster decrease the accuracy of high-dimensional problems? The answer is no. To understand the reason, firstly, we need to know where the error comes from for the Feynman-Kac formula (δ_{Fk}) and each algorithm. We must admit that the Feynman-Kac formula will cause a more significant error compared to those from traditional solutions (δ_{tradi}), such as finite element method (FEM), finite volume methods (FVM), and finite difference methods (FDM). Interpolation is a mathematical technique to estimate values between two points. In other words, we need those two points to be highly accurate to find the more trustworthy estimated values in between. Since the Feynman-Kac formula already increases the error, our points are not accurate enough to find accurate estimated points, which will cause interpolation errors (δ_{inter}). Thus, the error will add up as ($\delta_{Fk} + \delta_{inter}$). A schematic diagram is shown in Figure.

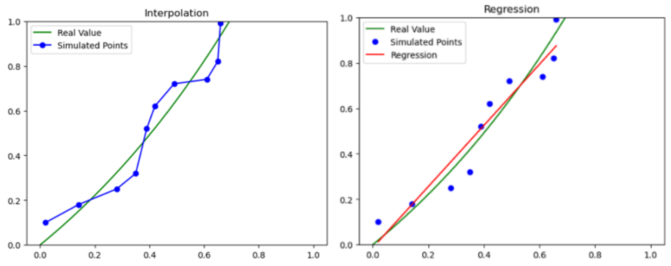

Figure 1: Approximate Algorithm Simulation Diagram for Interpolation and Regression

Take the left plot in Figure 1 as a specific example; the blue line will be the approximation line after interpolation. Each point and overall global solution have quite a big error compared to the real solution. Meanwhile, for the new algorithm, as shown on the right side of Figure 1, even though we only take simulation once for each point, which might cause a larger error based on MCM properties, the regression will help compensate for this problem. When we take one simulation for each point, there will be some points with very large errors and some points with extremely small errors, and regression can help neutralize the points of large and small errors and help us get a more accurate global solution. See the right plot above; the red line is the global solution after regression. Compared to the blue line in the left plot, the red line in the right plot is significantly closer to the real solution, while the green lines have the same points.

Comparing the two graphs above, the plot at left shows another issue: the global solution cannot be adjusted easily, leading to error rather than regression. However, for the plot at right, regression can dynamically adjust the global solution depending on the position and number of estimated points, while more points are given and put into the simulation, which can reduce the error of global solutions.

4. Numerical Experiment

Let's consider a Poisson equation as an example. For a multi-index m, sum the solutions to get the final approximation $u = (x, y)$. An elliptical PDE is shown in equation (7) below:

$$-\frac{1}{2}\Delta u = f \tag{7}$$

where Δ is the Laplacian operator, $u = g$, and g and f are (8) and (9) below:

$$g(x, y) = \exp\left((1+i)(x+y)\right) \tag{8}$$

$$f(x, y) = (1+i)^2 \exp\left((1+i)(x+y)\right) \tag{9}$$

When in two-dimensional problems with domain $D = (0,1)^2$, equation (7) will turn into equation (10):

$$\Delta u = -\exp(x+y) \tag{10}$$

with choosing Dirichlet boundary conditions. The Euler Scheme in (11)(12) discretizes the SDE for numerical simulation [7], where B_t refers to the Brownian Motion.

$$B_{n+1} = B_n + \sqrt{\Delta t}Y_n \ (7-1) \tag{11}$$

$$B_0 = x \ (7-2) \tag{12}$$

where Δt is the discretization parameter for time and Y_n is the sequence for identity normal random variables. If there is a stop between n and n+1, the approximation of Brownian Motion for the point jumping out of the domain can be calculated based on the value of B_n and B_{n+1}. Therefore, the $\int_0^{\tau D} f(X_s)ds$ part in the square of equation (4) can be calculated. After applying (11)(12) in (5), $u_m = (x, y)$ can be solved in conjunction with the discretized paths generated from the Euler scheme, based on equation (10), and X_t can be used to approximate the solutions. Based on the Dirichlet boundary value problem with $x \in D$ we mentioned above and the Euler Scheme, we can get a numerical result for (4) written as (13) below [7]. Therefore, the solved PDE will be (13).

$$u(x, y) = \exp(2(x + y)) \tag{13}$$

To get the solution, the points with position (x, y) will be picked out randomly within a uniform distribution ranging from 0 to 1. To get the real value, we only need to compute (13) with x and y values for each point. Then, we can use (x, y) points to compute the global solution using the interpolation method and regression, respectively, and compare the error.

4.1 Parameter Settings

4.1.1 Interpolation

A two-dimensional graph with domains 0 to 1 is set to generate 10 points x and y, respectively, meaning there will be 100 points. Each point will be simulated 10,000 times; therefore, the total number of simulations will be 1,000,000 times. The time interval will be set as 0.001. Each given point (x, y) will be put into the Feynman-Kac formula as illustrated above 10,000 times, and we will use the mean of the result after 10,000 times of simulation as the estimated solution by using the Feynman-Kac formula. Finally, there will be 100 estimated points after using the Feynman-Kac formula. These 100 points will be used to find the estimated points between each two points so that the global solution will be found. Then, we will repeat the same steps to other totals the number of simulations, changing to 100,000, 10,000, and 1,000 times.

4.1.2 Regression

To make sure the other conditions are the same as the one in the interpolation method, the two-dimensional graph will be set as same, and there will be 1,000,000 points generated in total, and each point will be simulated once, so the total number of simulations is 1,000,000 times as well. The time interval will also be set as 0.001 seconds. After 1,000,000 points were put into the Feynman-Kac formula to get 1,000,000 estimated points, these approximations will be used in regression with a polynomial degree of 3 to get the global solution after the regression line fits all estimated points. Then, we will repeat the same steps to other totals the number of simulations, changing to 100,000, 10,000, and 1,000 times. Then, the table will be set to compare the error differences between the two methods when the number of simulations increases.

4.2 Table Result of Using Interpolation and Regression

Table 1: Comparison between Interpolation and Regression

Algorithm	Number of Points Picked	Total Number of Simulations	Absolute Error Mean	Relative Error Mean
Interpolation	100	1,000,000	7.803	1.080
	100	100,000	7.813	1.088
	100	10,000	7.927	1.094
	100	1,000	8.397	1.148
Regression	1,000,000	1,000,000	0.197	0.039
	100,000	100,000	0.205	0.045
	10,000	10,000	0.274	0.046
	1,000	1,000	1.011	0.162

From Table 1, the average absolute and relative errors of interpolation when the total number of simulations changes are overall larger than the errors of regression. Since MCM is used to help solve the Feynman-Kac formula, one of the most important properties of MCM is that as the number of simulations increases, the error will decrease. From Table 1, even when the total number of simulations increases, the errors slightly decrease; compared to using the regression method, the overall error is still larger. When the total number of simulations is smaller, the error difference between the two methods will be smaller, and vice versa. When the total number is 1,000 times, both absolute and relative errors of using the interpolation are 7.5 ± 0.5 times larger than using the regression method. When the total number is 1 million times, both errors for interpolation are 33.1 ± 6 times larger.

Table 2 below shows how the multiple difference changes as the total number of simulations increases. After errors for interpolation are divided by errors for regression, the multiple difference of errors increases when the total number of multiplication increases, which means the increase in the total number of simulations will cause a quicker decrease in regression error while there will be a slower decrease in interpolation error. Therefore, it can easily be concluded that regression is a better way to approach a global solution than the traditional interpolation method because regression is more accurate.

Table 2: The multiple differences between errors by using two methods.

Total Number of Simulations	Multiple difference for Absolute Error	Multiple difference for Relative Error
1,000,000	39.6091371	27.6923077
100,000	38.1121951	24.1777778
10,000	28.9306569	23.7826087
1,000	8.30563798	7.08641975

4.3 Graph Result of Using Regression

To graph points and errors out easily, I chose fewer points to show the conjoint ratio. Since $D = (0,1)^2$, there will be only 999 points instead of 1,000 points. To make 999 simulations, 999 points will be needed for regression derived from 0 to 1 with a 0.001 difference between each point for x values, and y values are equal to x values. Specifically, the x list will be [0.001, 0.002, 0.003, 0.004, 0.005, …, 0.998, 0.999]. After simulating each point once through the Feynman-Kac formula, we can get all 999 approximations shown as yellow points in plot 2. The real values for each point will be shown as the green line, and the regression based on the Feynman-Kac expected value will be presented as the red line below.

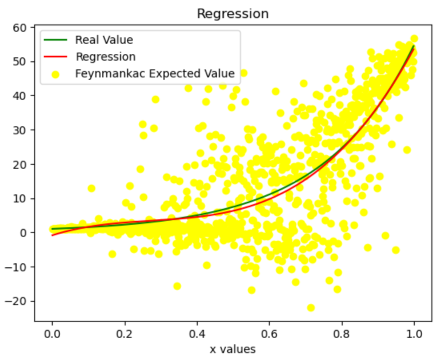

Figure 2: Regression results, real value, and the Feynman-Kac formula expected value when x = y, and x is number s between 0 to 1

From Figure 2 above, it is obvious that the error between the real value and the approximation after regression is quite small. Some parts of the two lines overlap each other, and even the parts that are not overlapping, the error between them is extremely small. To present the error difference between using interpolation and using regression to simulate clearly, the total number of simulations, time intervals, and ranges of points are chosen to be the same. Both absolute and relative errors are computed to prove that the regression method will be more accurate than the interpolation method.

5. Convergence Analysis

Three types of main variables will affect the result in this new algorithm: time interval, number of simulations, and polynomial degree of regression. The time interval will affect the result because the smaller the time interval, the better the continuity of the simulation path. Several simulations are also one effect of convergence in the Monte Carlo simulation of the Feynman-Kac formula because it relies on the law of large numbers, which suggests that when the number of simulations is infinite, the average of the result will converge to the expected value. Also, with increasing simulations, there will be a reduction of variance. If an error has been reduced to half, the number of simulations must be increased to four times the original size. Last of all, the polynomial degree is also one of the effects. Theoretically, as the number of polynomial degrees increases, the regression will fit the real solution more, reducing the error. Thus, it should converge as the polynomial degree increases. To ensure it will not be affected by other effects, numerical examples and control variable methods will help present the convergence situation under each effect.

5.1 Effect on Relative Error When Time Interval Decreases

For time intervals, 10,000 given points are used to find estimated points using the Feynman-Kac formula and MCM, meaning there will be 10,000 times simulations, and the domain will always be 0 to 1. When using regression to find the global solution, the polynomial degree is set to 3 while the time interval changes.

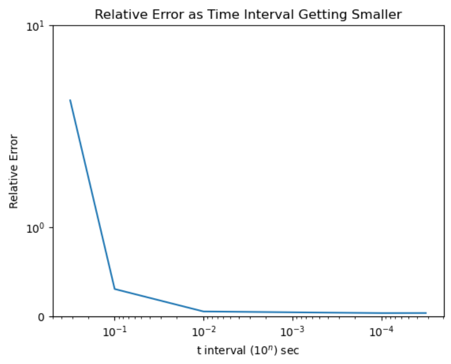

Figure 3: Relative Error vs. t interval

From Figure 3, it is obvious that the relative error decreases exponentially when the time interval gets smaller. Especially when the time interval changes from 1 to 0.1 seconds, the relative error gets 55 times smaller. It is obvious from the diagram that the time interval converges.

5.2 Effect on Relative Error When Number of Simulations Increases

To see the changes in relative error as the number of simulations increases, the time interval is set to 0.001 seconds, and the degree of the polynomial is set to 3. As shown in the x-axis in the plot below, the number of simulations changed from 1 to 10^6. After the condition has been set, the global solution will be solved after the regression, and the relative error can be approached after comparing it to the actual value.

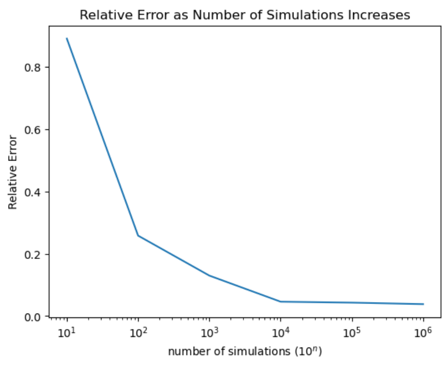

Figure 4: Relative Error as the number of simulations changes

From Figure 4, the relative error will get smaller as the number of simulations increases. Imagine squared charts with domains 0 to 1. When the number of simulations increases, the number of points in this square chart increases, and the regression can adjust closer to real solutions because the average error from each point to real value will be smaller. Therefore, the overall error will decrease as well. It proves that the relative error converges while the number of simulations increases.

5.3 Effect on Relative Error When Degree of Polynomial Regression Increases

To ensure only the polynomial degree changes, the number of simulations is set to 10,000, and the time interval is set to 0.001. At the same time, the polynomial degree increases from 1 to 6.

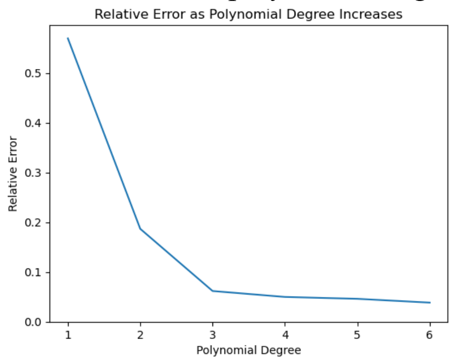

Figure 5: Relative Error as the polynomial degree in regression changes

From Figure 5, we can clearly see that the data we have proven the theoretical assumption as the number of polynomial degrees of the regression increases. The relative error gets smaller. Thus, the relative error will converge if the number of polynomial degrees for regression increases.

6. Conclusion

This study presents a more efficient and accurate algorithm for using regression instead of traditional interpolation. The Feynman-Kac formula and MCM are commonly known to solve the computational challenges caused by high-dimensional problems. Typically, when simulating Feynman-Kac expected values, interpolation will be used, but the interpolation method will need more complex computation, which means it will be longer and time-consuming and less accurate. In our new algorithm, we suggest a new scheme combining stochastic processes and regression. The advantages of this scheme are that it demonstrates a significantly faster computation while having better accuracy. From the convergence conditions shown for the three effects and the error compared to using the interpolation method, the empirical results underscore the ability of this new algorithm in high-dimensional problems, which can contribute to the application of finding PDEs' global solutions. One concern about this method is overfitting, especially when more complex regression models are used, which might lead to poor generalization of unseen data. Overall, the graphs and data prove the accuracy and fitness of using regression than interpolation.

References

[1] DiBenedetto, Emmanuele, and Ugo Gianazza. Partial Differential Equations. Third edition., Birkhäuser, 2023.
[2] Cheng Y, Reich S .Assimilating data into scientific models: An optimal coupling perspective[J].Springer International Publishing, 2015.DOI:10.1007/978-3-319-18347-3_2.
[3] Darbon, Jérôme, et al. "Overcoming the Curse of Dimensionality for Some Hamilton–Jacobi Partial Differential Equations via Neural Network Architectures." Research in the Mathematical Sciences, vol. 7, no. 3, 2020.
[4] Bertoli F, Bishop A .Monte Carlo Methods for Controller Approximation and Stabilization in Nonlinear Stochastic Optimal Control[J].Ifac Papersonline, 2015, 48(28):811-816.DOI:10.1016/j.ifacol.2015.12.229.
[5] Zhou Y , Cai W .A Path Integral Monte Carlo Method based on Feynman-Kac Formula for Electrical Impedance Tomography[J]. 2019.DOI:10.48550/arXiv.1907.13147.
[6] Khodadadian, Amirreza, et al. "An Adaptive Multilevel Monte Carlo Algorithm for the Stochastic Drift–Diffusion–Poisson System." Computer Methods in Applied Mechanics and Engineering, vol. 368, pp. 113163, 2020
[7] Maire, S. & Tanré, E. Some new simulation schemes for the evaluation of Feynman–Kac representations. Monte Carlo Methods and Applications, 14(1), 29-51, 2008.
[8] Chengyu, C. & Tong, M. L1 adaptive control for general partial differential equation (PDE) systems [J]. International Journal of General Systems, 48(6): 656-689, 2019.
[9] Anker, F., et al. SDE BASED REGRESSION FOR LINEAR RANDOM PDEs, SIAM J. SCI. COMPUT, Vol. 39, No. 3, pp. A1168-A1200, 2017.
[10] Shen, Guangjun, et al. "Averaging Principle for Distribution Dependent Stochastic Differential Equations Driven by Fractional Brownian Motion and Standard Brownian Motion." Journal of Differential Equations, vol. 321, 2022, pp. 381–414
[11] Pham, H. Feynman-Kac Representation of Fully Nonlinear PDEs and Applications. Acta Mathematica Vietnamica, 40(2), 255–269, 2015.
[12] Mertz, Laurent, et al. "A Feynman–Kac Formula Approach for Computing Expectations and Threshold Crossing Probabilities of Non-Smooth Stochastic Dynamical Systems." Physica. D, vol. 397, pp. 25–38, 2019.