ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

A modified method based on the K-nearest neighbor approach for solving PDE global solutions using the Feynman-Kac formula

Shirui Zheng[1,*], Xin Gui[2]

[1]*Zhizhen College, Beihang University, Beijing, 100191, China*
[2]*School of Civil Engineering, Shandong Jianzhu University, Jinan, 250101, China*
[*]*Corresponding author: APTX4869zsr@outlook.com*

Abstract: *Inspired by the classification idea of the K-nearest neighbors (KNN) algorithm, this paper proposes a new method for numerically solving partial differential equations (PDE) using the Feynman-Kac formula. The method involves establishing a connection between PDE and stochastic differential equations through the Feynman-Kac formula. Random points are selected within the domain of the equation, and the KNN algorithm is used to partition these points into different regions. Then, Monte Carlo simulation is performed on the random points within each region to obtain a series of simulated values. These simulated values are substituted into the corresponding Feynman-Kac formula for each random point to obtain the solution. Finally, within each region, the average of the solutions for all random points belonging to that region is calculated, resulting in the corresponding approximate solution. By selecting an appropriate partitioning approach, a higher-precision global solution to the PDE can be constructed. Through a series of numerical simulations, the results show that the PDE global solution constructed by the new method achieves higher accuracy compared to traditional interpolation methods.*

Keywords: *Monte Carlo Simulation, Feynman-Kac formula, Numerical method of PDE, KNN*

1. Introduction

As early as the 17th century, Newton and others had discovered some simple differential equations and also explored some preliminary solutions. Subsequently, numerical solutions for differential equations emerged. With the development of differential equations, we have obtained various methods for constructing numerical solutions to differential equations, such as the finite difference method, the finite element method, the Runge-Kutta method, and the Von Neumann analysis method. American mathematician Richard Feynman and French mathematician Michelle Kac also proposed the Feynman-Kac formula for constructing numerical solutions to differential equations. It is often used in the field of mathematical finance to solve various mathematical modeling problems, such as issues in Monte Carlo simulations and financial risk management. The finite difference method and the finite element method can achieve good results in solving ordinary differential equations and some general PDE numerically. For example, when solving second-order nonlinear elliptic PDE, the finite difference formulas with sixth-order truncation errors can be employed to obtain solutions [2]; We can utilize the fourth-order unfitted characteristic finite element method to study free-boundary problems of time-dependent partial differential equations, and through numerical experiments, discover some convergence properties [3]. However, when solving more complex PDE, the obtained numerical solutions may have significant deviations, rendering these methods unsuitable.

The Feynman-Kac formula can be used to solve PDE by utilizing stochastic differential equations, which is why it is often employed for numerical solutions of PDE. By utilizing the Feynman-Kac formula, we can explore a new relationship between diffusion and solutions in hyperbolic PDE. Additionally, we can transform the Laplace transform of wave equations with axial symmetry into a simpler form [1]. As for the current methods for numerical solutions of PDE, a common issue is that most of them do not construct global solutions. Alternatively, interpolation and others are used to construct global solutions, but these methods often result in significant deviations in the obtained results. Inspired by the KNN, we propose a new method for constructing global solutions of PDE based on the Feynman-Kac formula, referring to this method as the K-method. In this paper, we will use the heat conduction equation as an example to provide a detailed introduction to the K-method.

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

2. Propaedeutics

2.1 Heat Conduction Equation

The origin of the heat conduction equation lies in the need to solve for the temperature function of a disc. It is used to describe how the temperature in a region changes over time. However, as it has been generalized, this equation has been applied in various fields. The heat conduction equation is not only applicable in the field of thermodynamics but also plays a crucial role in areas such as finance, image processing, and machine learning. To solve a practical problem described by a heat conduction equation, we often incorporate corresponding boundary conditions based on reality. Examples of such boundary conditions include homogeneous boundary conditions and growth conditions. The heat conduction equation is a classic example of PDE. In this paper, we will introduce the K-method using the following heat conduction equation as an example:

$$\begin{cases} \dfrac{\partial u}{\partial t}+\mu(x,t)\dfrac{\partial u}{\partial t}+\dfrac{\sigma^2(x,t)}{2}\dfrac{\partial^2 u(x)}{\partial x^2}+k(x,t)u+f(x,t)=0, x\in R, t\in[0,T] \\ u(x,T)=G(x) \end{cases} \tag{1}$$

2.2 Feynman-Kac Formular

The Feynman-Kac formula is named after Richard Feynman and Mark Kac, who successfully combined stochastic differential equations with parabolic partial differential equations. By expressing the solutions of certain parabolic partial differential equations as conditional expectations of stochastic processes using the Feynman-Kac formula, it is possible to transform the numerical solution of such equations into a probability problem. For example, we can simplify the heat equation driven by time-homogeneous white noise potential using the Feynman-Kac formula [12]. Therefore, the Feynman-Kac formula plays an important role in solving complex mathematical modeling problems. In this paper, we derive the corresponding Feynman-Kac formula using the heat conduction equation (1) as an example:

$$\begin{cases} u(x,t_0)=E[\int_{t_0}^{T} f(X_s,t)\exp(\int_0^t k(X_r,r)dr)ds]+\exp(\int_{t_0}^{T} k(X_s,s)ds)G(X_T) \\ dX_s=\mu(x,t)dx+\sigma(x,t)dB_t \\ X_0=x \end{cases} \tag{2}$$

$u(x,t_0)$ is the point to be determined, T is the termination time of the stochastic differential equation, and X_T is the simulated point at the end of the simulation.

2.3 Monte Carlo Simulation

The Monte Carlo simulation algorithm, also known as the statistical simulation method, is commonly used to solve problems involving approximate calculations. Its fundamental idea is to: By generating random numbers and performing multiple simulations, the Monte Carlo simulation algorithm calculates a more accurate probability of the event occurring. In this way, it provides an approximation of the exact value of the solution to the problem. By using this method, various problems can be transformed into probability calculation problems. The main steps of Monte Carlo simulation are as follows: constructing a random process and generating random numbers; randomly sampling different events and calculating their respective probabilities; using these probabilities to compute the corresponding estimation values. The more simulations conducted, the closer the estimated value gets to the true value. Due to its simplicity in operation, Monte Carlo simulation is commonly used in various fields. For example, it can be used to compute the integral value of complex shapes and handle problems involving multidimensional arrays. However, Monte Carlo simulation also has some limitations: it has a slow convergence rate; its errors have a probabilistic nature rather than the traditional sense of error; in order to make the estimated value approach the true value, a large number of simulations and samples are required, resulting in a significant increase in computational complexity. However, there is a recent method called Multilevel Monte Carlo Simulation that combines Approximate Bayesian Computation to reduce the overall computational cost while maintaining inference accuracy. This approach is particularly useful for complex Bayesian computation

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

problems, especially when simulation and computational costs are high. [5].

2.4 K-Nearest Neighbor

The KNN is a classic and simple machine learning algorithm that helps solve classification and regression prediction problems. Its main steps are as follows: collect samples within a certain range; calculate the Euclidean distance between each point in the sample and the current point; arrange and classify them based on the distance magnitude; calculate the occurrence frequency of each class of points within the range; sort them based on frequency, and select the corresponding number of points with high frequencies as the current estimation. Using KNN for classification is crucial in imbalanced data classification applications. It not only saves the process of handling data but also reduces the cost of misclassification [4]. Inspired by the ability of the K-Nearest Neighbor algorithm to classify data, we can also utilize the KNN to partition random points in stochastic differential equations. We can set the Euclidean distance to be k, randomly select a point x, and classify the points whose Euclidean distance from x is less than k (points within $[x-k, x+k]$) into one class. Then, we can separately calculate the exact solutions of the PDE for each region. Recently, an improved KNN algorithm has been proposed which enhances the classification efficiency while maintaining classification accuracy, demonstrating good classification performance[6-7].

3. The Idea of Constructing the Global Solution to PDE Using the Feynman-Kac Formula

3.1 Construct the Global Solution to a PDE Using Interpolation Methods

When solving partial differential equations, we often utilize traditional methods, like interpolation method, to construct the global solution. For example, when dealing with computationally and analytically challenging PDE, we can employ the approximation method of interpolating polynomials to represent infinite-dimensional PDE as higher-order ordinary differential equations. This approach also incorporates certain characteristics of the PDE, thereby simplifying the handling process of the PDE [8]. By linearizing the nonlinear part, we can solve the Kolmogorov-Petrovskii-Piskunov (KPP) equation and obtain the corresponding results using barycentric rational interpolation basis function [9]. In this paper, we will take the heat conduction equation (1) as an example and use Lagrange interpolation method to construct the global solution on the interval $[a, b]$.

Taking interpolation points on the interval $[a, b]$, denoted as $\{x_1, x_2, \ldots, x_n\}$, we conduct Monte Carlo simulations for these points. With the time step size of Δt, we utilize the following formula:

$$X_{n+1} = X_n + \sqrt{\Delta t} R_n \tag{3}$$

We simulate the discrete path of the stochastic differential equation starting from the point x_i, where $X_0 = x_i$. The increments R_n are generated from a standard normal distribution. If the simulation time reaches T, the simulation is terminated. We denote the time nodes as $\{t_1, t_2, \ldots, t_N\}$, where $t_N = T$. The simulated points are denoted as $\{X_0, X_1, \ldots, X_N\}$, with $X_N = X_T$. This represents one simulation for an interpolation point. Let the number of simulations for each interpolation point be denoted as m. Then we perform m simulations for each interpolation point as described above.

We substitute the simulated points $\{X_0, X_1, \ldots, X_N\}$, time nodes $\{t_0, t_1, \ldots, t_N\}$, and the simulation end time T into equation (2). The specific integration process is as follows:

$$\int_{t_0}^{t} k(X_r, r)dr = \sum_{i=0}^{N} \Delta t k(X_i, t_i) \tag{4}$$

From this, we can obtain:

$$\int_{t_0}^{T} f(X_s, t) \exp\left(\int_{t_0}^{t} k(X_r, r)dr\right)ds = \sum_{j=0}^{N} \Delta t f(X_j) \exp\left(\sum_{i=0}^{N} \Delta t k(X_i, t_i)\right) \tag{5}$$

and:

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

$$\exp(\int_{t_0}^{T} k(X_s, s)ds)G(X_N) = \exp(\sum_{i=0}^{N} \Delta t k(X_i, t_i))G(X_T) \tag{6}$$

Add equation (5) to equation (6):

$$u(x_i) = \sum_{j=0}^{N} \Delta t f(X_j) \exp(\sum_{i=0}^{N} \Delta t k(X_i, t_i)) + \exp(\sum_{i=0}^{N} \Delta t k(X_i, t_i))G(X_T) \tag{7}$$

We can obtain the simulated value $\tilde{u}_l(x_i)$ of x_i in the l-th simulation. Then, taking the average of the simulated values for each point over m simulations give the exact solution of x_i : $\tilde{u}(x_1), \tilde{u}(x_2), \ldots, \tilde{u}(x_n)$

Then, we perform interpolation fitting on these n points to obtain an interpolation polynomial, thereby obtaining the global solution of the aforementioned equation:

$$\tilde{u}(x) = \sum_{i=1}^{n} [\tilde{u}(x_i) \prod_{j=1, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}] \tag{8}$$

From the aforementioned steps, it can be observed that the process of constructing the global solution using interpolation is relatively straightforward. However, it also comes with certain challenges. Firstly, due to the random nature of Brownian motion itself, the simulated values obtained in each simulation exhibit significant randomness. As a result, the average of the simulated values (i.e., the exact solution) also carries some level of randomness and may deviate from the actual values. Secondly, when adding new interpolation points, all polynomial coefficients need to be recalculated, leading to computational inefficiency. Lastly, when dealing with high-dimensional PDE, in order to approximate the actual solution accurately, it becomes necessary to select a larger number of interpolation points. However, this increases the computational workload and may potentially lead to the curse of dimensionality[10-11].

3.2 K-method for Constructing a Global Solution

Inspired by the KNN, this article proposes a novel approach to construct global solutions for PDE. To illustrate the methodology, this paper will focus on solving the heat conduction equation (1) for the interval $[a, b]$, and provide a detailed description of the relevant steps.

Uniformly select n initial points within the defined interval $[a, b]$, denoted as $\{x_1, x_2, \ldots, x_n\}$. Set the threshold (Euclidean distance) as k, and perform Monte Carlo simulations for these points within the range $[a - k, b + k]$. Assuming a time step size of Δt, according to the equation (3), we can simulate a random differential equation path starting from each point, where $X_0 = x_i$ and R_n is generated from a standard normal distribution. If the simulation time reaches T, the simulation ends and the simulation point at the end, X_N, is recorded. Subsequently, the simulated points $\{X_0, X_1, \ldots, X_N\}$ after each simulation, the termination time T, and the termination simulation point X_N are substituted into equation (2), which is consistent with equations (5) and (6) in the interpolation method. This process will yield an estimated value for each initial point: $\tilde{u}(x_1), \tilde{u}(x_2), \ldots, \tilde{u}(x_n)$

Randomly select a number x_i within the interval $[a, b]$, and define the corresponding interval as $[x_i - k, x_i + k]$. For l initial points $\{x_{i_1}, x_{i_2}, \ldots, x_{i_l}\}$ within this interval, calculate the average of the estimated values:, denoted as $\tilde{u}_i$. Thus, $\tilde{u}_i$ represents the exact solution of $u(x)$ at x_i. To construct a global solution, we select multiple points within the region $[a, b]$ and utilize the exact solutions of these points to form the global solution.

$$\{\tilde{u}(x_{i_1}), \tilde{u}(x_{i_2}), \ldots, \tilde{u}(x_{i_l})\}$$

4. The Numerical Experimental Results

4.1 The Model Settings

In this section, we will conduct experiments using a specific heat conduction equation as an example. The heat conduction equation is given by:

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

$$\begin{cases} u_t + \frac{\partial^2 u}{2\partial x^2} + \frac{\sin(x+t)}{2} - \cos(x+t) = 0, x \in R, t \in [0,1] \\ u(x,T) = \frac{\sin(x+1)}{2} - \cos(x+1) \end{cases} \tag{9}$$

The Feynman-Kac formula corresponding to equation (9) is:

$$\begin{cases} u(x,t_0) = \int_{t_0}^{T} [\frac{\sin(x+t)}{2} - \cos(x+t)]ds + \sin(X_N + T) \\ dX_s = dB_t \\ X_0 = x \end{cases} \tag{10}$$

4.2 Interpolation Methods

In this paper, we will utilize interpolation methods to solve the heat conduction equation (9): Taking five points within the domain [0,1] as {0.1,0.2,0.5,0.8,0.9}, we can obtain the corresponding function values using Monte Carlo simulation.

Table 1: Function Value

x	**Function Value** $u(x)$	**Estimated Value** $\tilde{u}(x)$
$x = 0.1$	0.1076	0.0998
$x = 0.2$	0.1963	0.1987
$x = 0.5$	0.4845	0.4794
$x = 0.8$	0.7100	0.7174
$x = 0.9$	0.7962	0.7833

According to the estimated values of Table 1, we can perform interpolation fitting to obtain the interpolation polynomial:

$$\tilde{u}(x) = 2.065x^4 - 4.063x^3 + 2.464x^2 + 0.4018x + 0.04661 \tag{11}$$

The graph of the interpolation polynomial function and the actual function is shown in Figure 1:

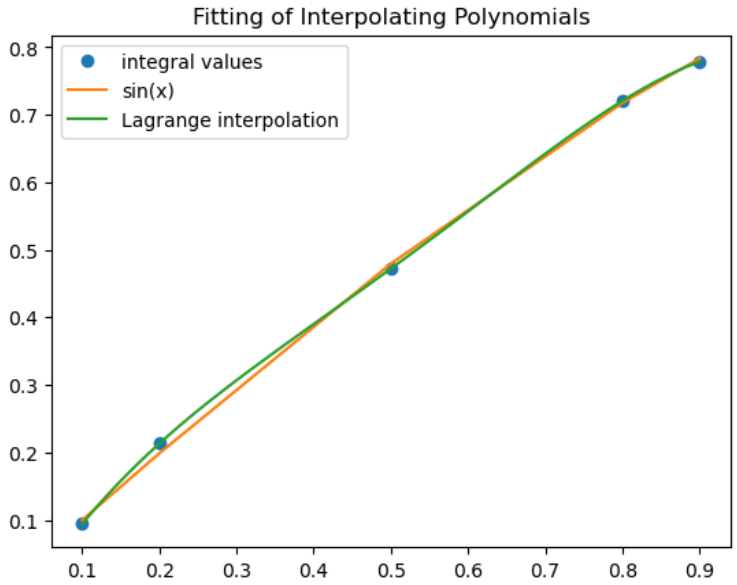

Figure 1: Lagrange interpolation fitting

4.3 K-method

25000 points are uniformly taken within the domain [0,1]. Random differential equation discrete

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

paths are simulated for these points. The time step size, Δt, is set to 0.001, and each point is simulated only once. The termination time, T, is set to 1, and the random number X_T at the end of each simulation is recorded. These values are then substituted into equation (10) to obtain the estimated values for each point. Taking the threshold k as 0.05, a random number x is sampled from the interval [0,1], and the corresponding interval is defined as $[x-k, x+k]$. The average value $\tilde{u}_i$ of the estimated values at all starting points within this interval is considered as the exact solution for x. Now, 20 points are uniformly taken within [0,1], and the exact solutions for these twenty points are obtained using the K-method. The fitting effect of the K-method is shown in Figure 2:

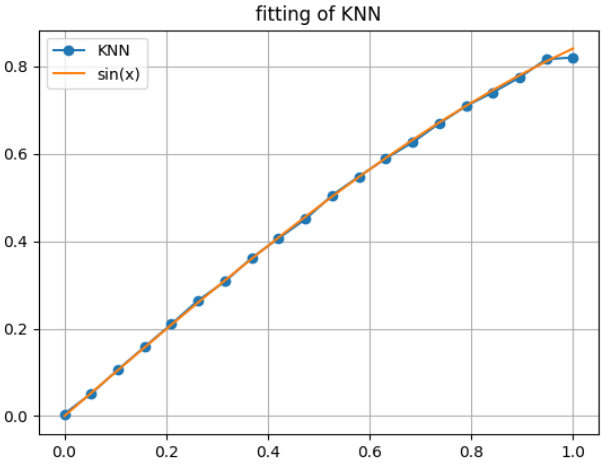

Figure 2: K-method fitting

4.4 Error Analysis

After a series of calculations, we can obtain the error values between the results of the interpolation method and the K-method as follows:

Table 2: Error of two Methods

Fitting Method	Absolute Error Value
Interpolation Method	0.0073
K-Method	0.0021

Based on the error of Table 2, it can be observed that the errors corresponding to the K-method are significantly smaller than the errors corresponding to the interpolation method. Therefore, when constructing a global solution for PDE, the K-method is expected to yield better results.

5. The Convergence Behavior of the K-method under Different Parameter Settings

Based on the above numerical experimental process, it can be observed that the error in constructing the global solution for PDEs using the K-method is mainly influenced by the time step size Δt, the number of simulations m, and the threshold value k. This section will discuss the impact of these three parameters on the error.

For the sake of convenience, let's consider the above heat conduction equation (9) as the object of study. We will also set the sampling points for error analysis as follows:

$$\{0.1,0.2,0.5,0.6,0.8,0.9\}$$

5.1 The Convergence Situation under Different Time Step Sizes

The time step size affects the stochastic process. A smaller time step size results in a less-dispersed stochastic process, allowing the estimated values to approach the true values more closely. However, this also leads to a significant increase in computational cost. On the other hand, a larger time step size increases the dispersion of the stochastic process, resulting in larger errors in the estimated values. Therefore, in order to ensure sufficient accuracy while controlling the computational cost, it is necessary to find a suitable time step size that strikes a balance between accuracy and efficiency. Let's

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

consider a series of time step sizes $\Delta t \in \{0.1,0.05,0.01,0.005,0.001,0.0005\}$ and perform numerical simulations to obtain the corresponding error values. The convergence behavior is shown in the following graph:

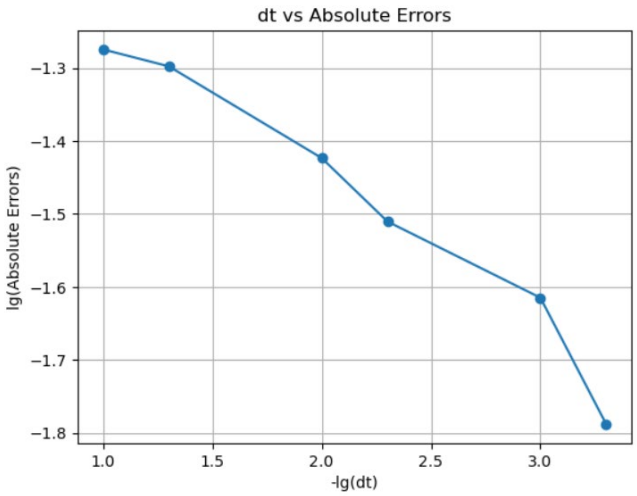

Figure 3: The change of error with respect to the time step size

From the change of error with respect to the time step size in the figure 3, it can be observed that, in general, the error decreases as the time step size Δt decreases, and it shows a tendency to approach zero.

5.2 The Convergence Situation under Different Numbers of Simulations

Monte Carlo simulation is a sampling method, and the sampling error mainly arises from an inappropriate sampling that fails to reflect the characteristics of the population, such as when the number of samples cannot represent the overall characteristics. In such cases, increasing the number of samples or the number of simulations can help alleviate the issue. However, if the number of simulations is too large, it may result in slower convergence. To find an appropriate number of simulations, it is necessary to explore the error values under different simulation counts. In this study, we will investigate different simulation counts to analyze the convergence behavior. Let's consider the following values for the simulation count:

$$m \in \{5,10,20,50,200,1000,2000,5000,10000,15000,20000,25000\}$$

The error values are shown in the following graph:

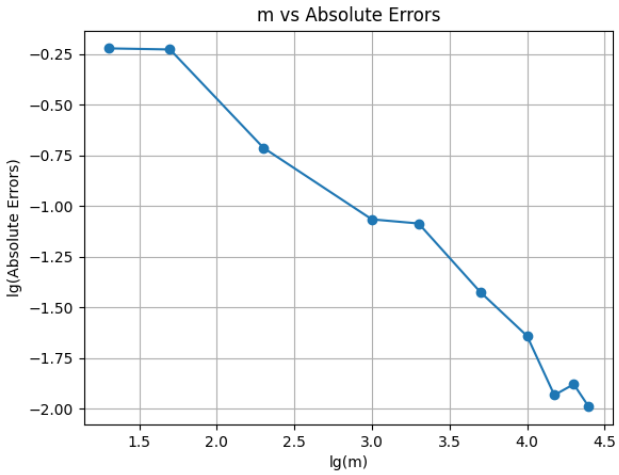

Figure 4: The change of error with respect to the number of simulations

According to the change of error with respect to the number of simulations in the figure 4, it can be observed that as the number of simulations increases, the error generally decreases. However, as the

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

number of simulations continues to increase, the rate of error reduction diminishes.

5.3 The Convergence Situation under Different Thresholds

KNN plays a role in classification during the solving process by assigning simulated points within a certain range to a category and then using mean values to approximate the true solution. When dealing with imbalanced samples, there is a significant prediction bias, and suitable threshold values for k can only be determined through experience and repeated experiments. A larger k value results in more dispersed simulated points, increasing the model's bias and making it less sensitive to noisy data, potentially leading to underfitting. On the other hand, a smaller k value leads to more concentrated simulated points, weakening the model's generalization ability and potentially causing overfitting. To obtain the most accurate solution possible, it is necessary to experiment with different k values. Let's consider the following threshold values for k:

$$k \in \{0.01,0.03,0.05,0.07,0.08,0.10,0.12\}$$

The corresponding error values are shown in the graph below:

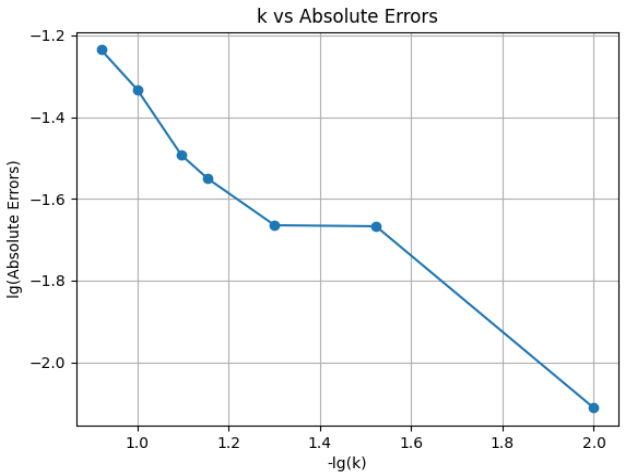

Figure 5: The change of error with respect to the number of threshold value

According to the change of error with respect to the number of threshold value in the figure 5, it can be observed that as the threshold value decreases, the error also gradually decreases, and the reduction amount increases.

6. Conclusion

This article primarily introduces a new method called the K-method for constructing global solutions to partial differential equations (PDE). This method has smaller errors and higher accuracy in computation compared to traditional interpolation methods. Additionally, the article explores the parameters that affect the computational accuracy of the K-method, including time step size, simulation iterations, and threshold values. Setting these parameters too large or too small may make the computation process more complex or reduce the precision of the computed results. Therefore, when using the K-method, it is necessary to perform multiple numerical simulations and make corresponding adjustments to select appropriate parameter values. It is important to note that this article only conducts a simple study on the one-dimensional heat conduction equation and does not extend to higher-dimensional or multivariate equations. However, this does not imply that this method cannot be used to solve complex problems, higher-dimensional equations, or other types of equations. In fact, this method can also be applied to study equations in higher dimensions and holds extensive application prospects in solving other types of PDE as well. In summary, when using this method, it is necessary to make suitable adjustments and solutions based on the specific problem at hand.

References

[1] Jakubowski J, Wiśniewolski M. On matching diffusions, Laplace transforms and partial differential

ISSN 2616-5805 Vol. 4, Issue 4: 36-44, DOI: 10.25236/AJMS.2023.040406

equations [J]. Stochastic Processes and their Applications, 2015, 125(10): 3663-3690.
[2] Gavete L, Ureña F, Benito J J, et al. Solving second order non-linear elliptic partial differential equations using generalized finite difference method [J]. Journal of Computational and Applied Mathematics, 2017, 318: 378-387.
[3] Ma C, Zheng W. A fourth-order unfitted characteristic finite element method for free-boundary problems [J]. Journal of Computational Physics, 2022, 469: 111552.
[4] Zhang S. Cost-sensitive KNN classification [J]. Neurocomputing, 2020, 391: 234-242.
[5] Jasra A, Jo S, Nott D, et al. Multilevel Monte Carlo in approximate Bayesian computation[J]. Stochastic Analysis and Applications, 2019, 37(3): 346-360.
[6] Ferré G, Stoltz G. Error estimates on ergodic properties of discretized Feynman–Kac semigroups [J]. Numerische Mathematik, 2019, 143: 261-313.
[7] Hutzenthaler M, Jentzen A, Kruse T. On multilevel Picard numerical approximations for high-dimensional nonlinear parabolic partial differential equations and high-dimensional nonlinear backward stochastic differential equations [J]. Journal of Scientific Computing, 2019, 79(3): 1534-1571.
[8] Ma T, Cao C. L1 adaptive control for general partial differential equation (PDE) systems [J]. International Journal of General Systems, 2019, 48(6): 656-689.
[9] Li J, Cheng Y. Barycentric rational interpolation method for solving KPP equation [J]. Electronic Research Archive, 2023, 31(5): 3014-3029.
[10] Arendt W, Urban K. Partial Differential Equations: An Introduction to Analytical and Numerical Methods [M]. Springer Nature, 2023.
[11] Xing W, Bei Y. Medical health big data classification based on KNN classification algorithm [J]. IEEE Access, 2019, 8: 28808-28819.
[12] Scorolli R. Feynman-Kac formula for the heat equation driven by time-homogeneous white noise potential [J]. arXiv preprint arXiv: 2108.12406, 2021.