UNIVERSITY OF OSLO

Project 5. Diffusion of neurotransmitters in the synaptic cleft in two dimensions

by

Candidate number 70

in the

Faculty of Mathematics and Natural Sciences

Department of Physics

December 2014

UNIVERSITY OF OSLO

Abstract

Faculty of Mathematics and Natural Sciences
Department of Physics

by Candidate number 70

Background. In this project the partial differential equation describing diffusion of neurotransmitters in the synaptic cleft is solved by different methods. The equation is written for the one-dimensional case and for 2+1-dimensional case, the boundary and initial conditions are given. Firstly, in the one dimensional problem the finite difference methods are used, viz. explicit forward Euler and the implicit backward Euler algorithms, based on approximations for the derivatives and Crank-Nicolson scheme, combining the two former. For the explicit scheme the problem is limited to a matrix-vector multiplication, while the implicit one uses the form of Gaussian elimination for tridiagonal matrix of coefficients. Both explicit and implicit methods are combined into one Crank-Nicolson method, which produces quite smaller truncation error for t because the method uses midpoint approximation for it. The solution is meant as existing and unique by default. The second used set of methods are Monte Carlo and random walks methods. In comparison to the first methods, these do not require any initial conditions and based only on calculating the number of moves by random walker to any interval. The step of the walks is constant or defined by random number chosen from a normal PDF. The most important part here are sampling rules.

The described methods are also used for solving 2+1-dimensional diffusion equation. The boundary and initial conditions are increased correspondingly. The analytical solutions are found.

Results. The results of numerical calculations are presented in the Chapter 3. The closed-form solutions are found in order to control and measure the results of the numerical experiment. There are plots of the results for the chosen methods as well as the closed-form results. The calculations are made for the cases when $t\_final << 1$ and $t\_final = 1$. The figures are compared and the results are discussed.

Conclusions. It is clear form the results that in the case of 1+1-dimensional diffusion equation FDMs perform results of high preciseness which repeats exactly the analytical solution for the same boundary and initial conditions. But one should always be aware of the relation between the step lengths for x and t when using the explicit scheme. Random walk simulations return not that smooth plot, however the shape of the solution is close to the closed-form one.

In the 2+1-dimensional problem FDMs (explicit and implicit) present the result of the same preciseness, however the limitations for explicit scheme are still should be taken into account. Monte Carlo and random walks simulations gives results in form of histograms, thus the plots are not smooth. However, the method requires less predefined parameters in order to present the solution. Moreover, the method describes the behaviour of the system in a real life.

The codes of the solvers and the PDF version of the report could be found under =https://github.com/daria-titova/Fys-4150-Project-5.git

Contents

Abstract i

1 Introduction 1
1.1 The task description . . . 2

2 Theoretical part 6
2.1 Finite difference method. Explicit Scheme . . . 6
2.1.1 Explicit Scheme for 1+1 dimensional partial differential equation . 6
2.1.2 Explicit Scheme for the diffusion equation in two dimensions . . . 8
2.2 Implicit Scheme . . . 9
2.2.1 Implicit Scheme for the diffusion equation in one dimension 9
2.2.2 Implicit Scheme for the diffusion equation in two dimensions with Jacobi's method as the iterative method . . . 10
2.3 Crank-Nicolson Scheme . . . 12
2.4 Truncation errors and stability properties of the finite difference methods 14
2.4.1 One dimensional problems . . . 14
2.4.2 Two dimensional problems . . . 15
2.5 Monte Carlo and random walks methods for solving diffusion equation . . 18
2.5.1 One dimensional problems . . . 18
2.5.2 Two dimensional problems . . . 20

3 Experimental Setup 23
3.1 Closed-form solution . . . 23
3.1.1 One dimensional problem . . . 23
3.1.2 Two dimensional problem . . . 25
3.2 Numerical experiment . . . 29
3.2.1 1+1-dimensional diffusion equation . . . 29
3.2.2 2+1-dimensional diffusion equation . . . 36

Bibliography 47

Chapter 1

Introduction

In mathematics, a partial differential equation (PDE) is a differential equation that contains unknown multivariable functions and their partial derivatives. PDEs are used to formulate problems involving functions of several variables. They can be used to describe a wide variety of phenomena such as sound, heat, electrostatics, electrodynamics, fluid flow, elasticity, or quantum mechanics. These seemingly distinct physical phenomena can be formalised similarly in terms of PDEs. [1] Just as ordinary differential equations often model one-dimensional dynamical systems, partial differential equations often model multidimensional systems. The diffusion equation is an example of PDE which describes the evolution in time of the density of a quantity like concentration or energy density.[2]

The three most widely used numerical methods to solve PDEs are the finite element method (FEM), finite volume methods (FVM) and finite difference methods (FDM). Among the FDMs, using finite difference equations to approximate derivatives, there are explicit scheme, implicit scheme and Cranc-Nicolson scheme.

Furthermore, one of the widely used algorithms, Monte Carlo methods, is also used in the project. The methods are used in Science, from the integration of multi-dimensional integrals to solving problems in chemistry, physics, medicine, biology, or even Dow-Jones forecasting or forecasting of the results of any event. Thus Monte Carlo methods (or Monte Carlo experiments) are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results; typically one runs simulations many times over in order to obtain the distribution of an unknown probabilistic entity.

In the project we work with the diffusion equation describing the behavior of neurotransmitters in the synapse. The FDMs are used. The issue of existence and uniqueness of solutions of ordinary differential equations are not discussed.

1.1 The task description

The dominant way of transporting signals between neurons (nerve cells) in the brain is by means of diffusion of particular signal molecules called *neurotransmitters* across the synaptic cleft separating the cell membranes of the two cells. A drawing of a synapse is given in Fig. 1.1. [2]

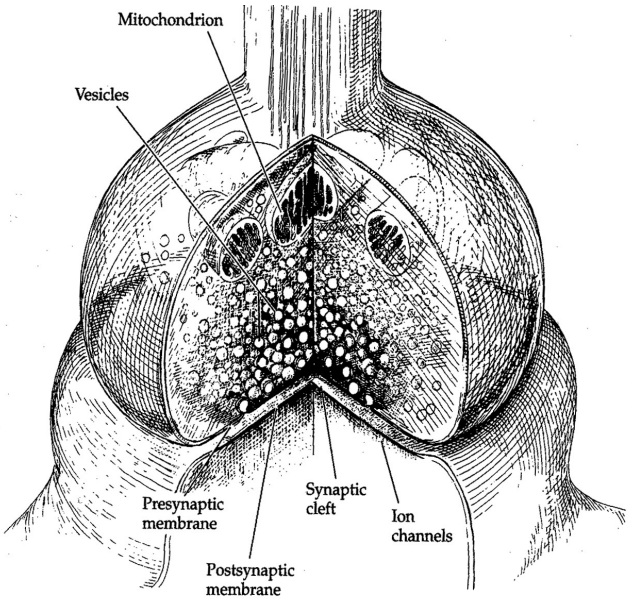

FIGURE 1.1: Drawing of a synapse. The axon terminal is the knoblike structure and the spine of the receiving neuron is the bottom one. The synaptic cleft is the small space between the presynaptic (axon) and postsynaptic (dendritic spine) membrane. (From Thompson: "The Brain", Worth Publ., 2000)

Following the arrival of an action potential in the axon terminal a process is initiated in which (i) vesicles inside the axon terminal (filled with neurotransmitter molecules) merge with the presynaptic (axon) membrane and (ii) release neurotransmitters into the synaptic cleft. These neurotransmitters diffuse across the synaptic cleft to receptors on the postsynaptic side which "receives" the signal. A schematic illustration of this process is shown in Fig. 1.2(left). Since the transport process in the synaptic cleft is governed by diffusion, we can describe it mathematically by

$$\frac{\partial u}{\partial t} = D\nabla^2 u, \tag{1.1}$$

where u is the concentration of the particular neurotransmitter, and D is the diffusion coefficient of the neurotransmitter in this particular environment (solvent in synaptic cleft).

If we assume (i) that the neurotransmitter is released roughly equally on the "presynaptic" side of the synaptic cleft, and (ii) that the synaptic cleft is roughly equally

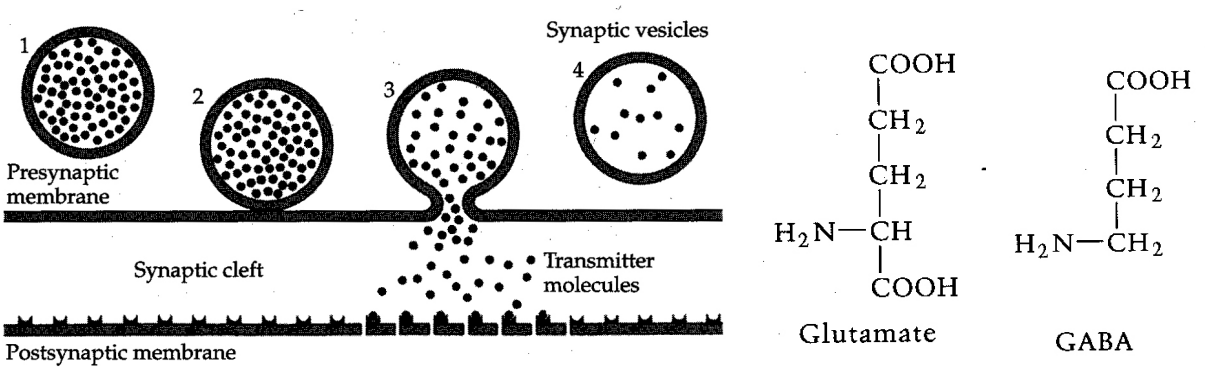

FIGURE 1.2: Left: Schematic drawing of the process of vesicle release from the axon terminal and release of transmitter molecules into the synaptic cleft. (From Thompson: "The Brain", Worth Publ., 2000). Right: Molecular structure of the two important neurotransmitters *glutamate* and *GABA*.

wide across the whole synaptic terminal, we can, given the large area of the synaptic cleft compared to its width, assume that the neurotransmitter concentration only varies in the direction across the synaptic cleft (from presynaptic to postsynaptic side). We choose this direction to be the x-direction (see Fig. 1.3). In this case $u(\mathbf{r}) = u(x)$, the diffusion equation reduces to

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}. \tag{1.2}$$

Immediately after the release of a neurotransmitter into the synaptic cleft ($t = 0$) the concentration profile in the x-direction is given by

$$u(x, t = 0) = N\,\delta(x), \tag{1.3}$$

where N is the number of particle released into the synaptic cleft per area of membrane.

In this project we impose the following boundary and initial conditions with $x \in [0, d]$

$$u(x = 0, t > 0) = u_0, \quad u(x = d, \text{all } t) = 0, \quad u(0 < x < d, t < 0) = 0 \ . \tag{1.4}$$

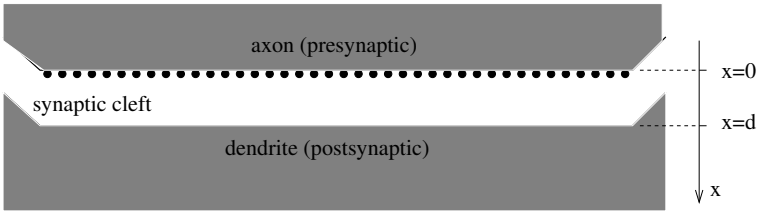

FIGURE 1.3: Schematic drawing of the synaptic cleft in our model. The black dots represent neurotransmitter molecules, and the situation shown corresponds to the situation immediately after neurotransmitter release into the synaptic cleft.

The full solution of the diffusion equation with boundary/initial conditions in Eq. (1.4) can be found in a closed form. We will use this solution to test our numerical calculations.

We are thus looking at a one-dimensional problem [2]

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{\partial u(x,t)}{\partial t}, t > 0, x \in [0,d]$$

or

$$u_{xx} = u_t,$$

with initial conditions, i.e., the conditions at $t = 0$,

$$u(x,0) = 0 \quad 0 < x < d$$

with $d = 1$ the length of the x-region of interest. The boundary conditions are

$$u(0,t) = 1 \quad t > 0,$$

and

$$u(d,t) = 0 \quad t > 0.$$

If we now extend this task to two dimensions, the 2+1 dimensional problem read the following:

$$\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} = \frac{\partial u(x,y,t)}{\partial t}, t > 0, x, y \in [0,d]$$

or

$$u_{xx} + u_{yy} = u_t.$$

The initial conditions ($t = 0$) in this case are

$$u(x,y,0) = 0 \quad 0 < x, y < d$$

The boundary conditions are extended the following way. We assume the same concentration of neurotransmitters on x-scale, while the concentration of neurotransmitter on y-scale is uniform. Therefore the boundary conditions are

$$u(0,y,t) = 1 \quad t > 0, y \in [0,d]$$

and

$$u(d,y,t) = 0 \quad t > 0, y \in [0,d].$$

In this project we want to study the numerical stability of Monte-Carlo methods for 1D and 2D partial differential equations (PDEs) in comparison with explicit and implicit schemes.

Chapter 2

Theoretical part

2.1 Finite difference method. Explicit Scheme

2.1.1 Explicit Scheme for 1+1 dimensional partial differential equation

In mathematics, finite-difference methods (FDM) are numerical methods for approximating the solutions to differential equations using finite difference equations to approximate derivatives. Lets look at diffusion equation, which is nothing but a partial differential equation. [3], [2]

$$\frac{\partial \varphi(r,t)}{\partial t} = D\nabla^2\varphi(r,t), \tag{2.1}$$

In one dimension (as it is described in the task) we have

$$\frac{\partial u(x,t)}{\partial t} = \frac{\partial^2 u(x,t)}{\partial t^2}, \tag{2.2}$$

with the given initial conditions, i.e., the conditions at $t = 0$,

$$u(x,0) = 0 \quad 0 < x < d$$

with $d = 1$ the length of the x-region of interest and boundary conditions:

$$u(0,t) = 1 \quad t > 0,$$

and

$$u(d,t) = u(1,t) = 0 \quad t > 0.$$

If we now introduce different step lengths for the space- and time- variables, x and t, the position after i steps and time at j^{th} time-step are given by

$$t_j = j\Delta t,$$

$$x_i = i\Delta x,$$

where $j \geq 0$, $0 \leq i \geq n+1$. Using the standard approximation for the derivatives we obtain the following:

$$u_t \approx \frac{u(x, t+\Delta t) - u(x,t)}{\Delta t} = \frac{u(x_i, t_j + \Delta t) - u(x_i, t_j)}{\Delta t}$$

and

$$u_{xx} \approx \frac{u(x+\Delta x, t) - 2u(x,t) + u(x-\Delta x, t)}{\Delta x^2},$$

or

$$u_{xx} \approx \frac{u(x_i+\Delta x, t_j) - 2u(x_i,t_j) + u(x_i-\Delta x, t_j)}{\Delta x^2}.$$

These equations can be further simplified as

$$u_t \approx \frac{u_{i,j+1} - u_{i,j}}{\Delta t},$$

and

$$u_{xx} \approx \frac{u_{i+1,j} - 2u_{i,j}) + u_{i-1,j}}{\Delta x^2}.$$

Defining

$$\alpha = \frac{\Delta t}{\Delta x^2},$$

we get the explicit Euler's scheme for our task:

$$u_{i,j+1} = \alpha u_{i-1,j} + (1-2\alpha)u_{i,j} + u_{i+1,j}.$$

We will implement this algorithm for the task by the following way:

```
//create array U
  mat U(n,m);

//boundary conditions
  U.col(0).ones();
  U.col(m-1).zeros();
```

```
//initial conditions
  U.row(0).zeros();

  vec V(m);
  V=U.row(0);   //initial values, t=0;

  int j=1;
  vec V_new(m);
  while (j<n){
  for (int i=0; i<m; i++)
  { V_new(i)=alpha*V(i-1) + (1-2*alpha)*V(i) + alpha*V(i+1);}
   V=V_new;   //reassign vector V
  j++;}       //increment time step
```

2.1.2 Explicit Scheme for the diffusion equation in two dimensions

The 2+1-dimensional diffusion equation is given by

$$\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} = \frac{\partial u(x,y,t)}{\partial t}.$$

We assume that we have a square lattice of length L with equally many mesh points in the x and y directions. Thus we can disceitize position and time using the standard approximation for the derivatives:

$$u_{xx} \approx \frac{u(x+\Delta x,y,t) - 2u(x,y,t) + u(x-\Delta x,y,t)}{\Delta x^2}.$$

Whith $\Delta x = h$ we rewrite it as

$$u_{xx} \approx \frac{u^l_{i+1,j} - 2u^l_{i,j} + u^l_{i-1,j}}{h^2},$$

where $x_i = x_0 + ih$, $y_j = y_0 + jh$ and $t_l = t_0 + l\Delta t$, with $h = \frac{L}{n+1}$ and Δt the time step. We have defined our domain to start $x(y) = 0$ and end at $x(y) = L$. The second derivative with the respect to y reads

$$u_{yy} \approx \frac{u(x,y+\Delta y,t) - 2u(x,y,t) + u(x,y-\Delta y,t)}{\Delta y^2}$$

or

$$u_{yy} \approx \frac{u^l_{i,j+1} - 2u^l_{i,j} + u^l_{i,j-1}}{h^2}.$$

We can use again the so-called forward-going Euler formula for the first derivative in time:

$$u_{tt} \approx \frac{u^{l+1}_{i,j} - u^{l+l}_{i,j}}{\Delta t}.$$

These approximations relut in

$$u_{i,j}^{l+1} = u_{i,j}^{l} + \alpha[u_{i+1,j}^{l} + u_{i-1,j}^{l} + u_{i,j+1}^{l} + u_{i,j-1}^{l} - 4u_{i,j}^{l}],$$

where the left hand side, with the solution at the new time step, is the only unknown term, since starting with $t = t0$, the right hand side is entirely determined by the boundary and initial conditions. We have

$$\alpha = \frac{\Delta t}{h^2}.$$

This scheme can be implemented using essentially the same approach as we used in 1+1 PDE. The constraints on Δt and h are discussed in the next section.

2.2 Implicit Scheme

2.2.1 Implicit Scheme for the diffusion equation in one dimension

In the previous part we use forward formula for the first derivative. But if we use the so-called backward formula for the derivative, which is [2], [**?**]

$$u_t \approx \frac{u(x,t) - u(x,t-\Delta t)}{\Delta t} = \frac{u(x_i,t_j) - u(x_i,t_j-\Delta t)}{\Delta t}$$

or

$$u_t \approx \frac{u_{i,j} - u_{i,j-1}}{\Delta t},$$

and use the second derivative for x and the same coefficient α, as we did earlier, we obtain the implicit Euler's formula:

$$u_{i,j-1} = -\alpha u_{i-1,j} + (1-2\alpha)u_{i,j} - \alpha u_{i+1,j}.$$

Here we can define matrix A as:

$$\mathrm{A} = \begin{bmatrix} 1+2\alpha & -\alpha & 0 & 0 & \dots \\ -\alpha & 1+2\alpha & -\alpha & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & -\alpha \\ 0 & 0 & \dots & - & 1+2\alpha \end{bmatrix}$$

what gives us a possibility to rewrite the problem in a matrix-vector form as

$$AV_j = V_{j-1}.$$

Therefore, we should solve just a set of linear equations in order to get the answer. As far as the matrix A is tridiagonal, we will use the solver for the tridiagonal matrices, namely the Thomas algorithm, that is a simplified form of Gaussian elimination. Thus, the algorithm for the implicit scheme looks like:

```
//create array U
  mat U(n,m);

//boundary conditions
  U.col(0).ones();
  U.col(m-1).zeros();
//initial conditions
  U.row(0).zeros();

  vec V(m);
  V=U.row(0);   //initial values, t=0;

//create the tridiagonal matrix A
  mat A(m,m);
  A.zeros();
  A.diag().fill(1+2*alpha);
  A.diag(1).fill(-alpha);
  A.diag(-1).fill(-alpha);

  int j=1;
  while (j<n){
      tridiag solver;   //initialize solver
      solver.tridiag_solver(A, V, m);
    //the solver finds the solution
    //and assign it to the V vector
      V(0)=1.0;    //boundary condition for x=0;
      V(m-1)=0.0;  //boundary condition for x=L=1;
  j++;}       //increment time step
```

2.2.2 Implicit Scheme for the diffusion equation in two dimensions with Jacobi's method as the iterative method

Earlier we discussed the extension of the diffusion equation to two dimensions, using an explicit scheme. Let us know implememt the implicit scheme and extend the algorithm for solving Laplace's or Poisson's equations to the diffusion equation. Let us first set up the diffusion in two spatial dimensions, with boundary and initial conditions. The 2+1-dimensional diffusion equation (with dimensionless variables) reads for a function $u = u(x, y, t)$

$$D(\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2}) = \frac{\partial u(x,y,t)}{\partial t}.$$

We assume that we have a square lattice of length L with equally many mesh points in the x and y directions. Setting the diffusion constant $D = 1$ and using shorthand notation we have, with a given set of boundary and initial conditions,

$$u_t = u_{xx} + u_{yy}, \quad t > 0, x \in (0, L)$$

$$u(x, 0)u(0, y, t) = u(L, y, t) = u(x, 0, t) = u(L, 0, t), \quad t > 0.$$

We discretize again position and time, and use the following approximation for the second derivatives

$$u_{xx} \approx \frac{u(x + h, y, t) - 2u(x, y, t) + u(x - h, y, t)}{h^2},$$

which we rewrite as, in its discretized version,

$$u_{xx} \approx \frac{u^l_{i+1,j} - 2u^l_{i,j} + u^l_{i-1,j}}{h^2},$$

where $x_i = x_0 + ih$, $y_j = y_0 + jh$ and $t_l = t_0 + l\Delta t$, with $h = \frac{L}{n+1}$ and Δt the time step. We have defined our domain to start $x(y) = 0$ and end at $x(y) = L$. The second derivative with the respect to y reads

$$u_{yy} \approx \frac{u(x, y + \Delta y, t) - 2u(x, y, t) + u(x, y - \Delta y, t)}{\Delta y^2}$$

or

$$u_{yy} \approx \frac{u^l_{i,j+1} - 2u^l_{i,j} + u^l_{i,j-1}}{h^2}.$$

We can use again the so-called backward Euler formula for the first derivative in time, that in its discretized form read

$$u_t \approx \frac{u^l_{i,j} - u^{l-l}_{i,j}}{\Delta t},$$

resluting in

$$u^l_{i,j} + 4\alpha u^l_{i,j} - \alpha[u^l_{i+1,j} + u^l_{i-1,j} + u^l_{i,j+1} + u^l_{i,j-1}] = u^{l-1}_{i,j},$$

where the right hand side is the only known term, since starting with $t = t_0$, the right hand side is entirely determined by the boundary and initial conditions. $\alpha = \frac{\Delta t}{h^2}$.

Now we rewrite the previous equation as

$$u^l_{i,j} = \frac{1}{1 + 4\alpha}[\alpha(u^l_{i+1,j} + u^l_{i-1,j} + u^l_{i,j+1} + u^l_{i,j-1}) + u^{l-1}_{i,j}],$$

or in a more compact form as

$$u_{i,j}^{l} = \frac{1}{1+4\alpha}[\alpha\Delta_{i+1,j}^{l} + u_{i,j}^{l-1}],$$

with

$$\Delta_{i+1,j}^{l} = u_{i+1,j}^{l} + u_{i-1,j}^{l} + u_{i,j+1}^{l} + u_{i,j-1}^{l},$$

This equation has essentially the same structure as discretized (with derivatives) Poisson's equation.Thus the scheme for solving Poisson's equation can be implemented here. Furthermore, the diagonal matrix elements are now given by $1+4\alpha$, while the nonzero non-diagonal matrix elements equal α. This matrix is also positive definite, meaning in turn that iterative schemes like the Jacobi or the Gauss-Seidel methods will converge to the desired solution after a given number of iterations.

Let us implement the Jacobi algorithm. In that case we know the values of $u(x,y,t)$ at $i=0$ or $i=n+1$ and at $j=0$ or $j=n+1$ but we cannot start at one of the boundaries and work our way into and across the system since the method requires the knowledge of u at all of the neighbouring points in order to calculate the value of u at any given point.

Therefore we should make a guess about the values of u for all i and j. We start with an initial guess for $u_{i,j}^{(0)}$ where all values are known. To obtain a new solution $u_{i,j}^{(1)}$ we solve the equation

$$u_{i,j}^{l} = \frac{1}{1+4\alpha}[\alpha(u_{i+1,j}^{l} + u_{i-1,j}^{l} + u_{i,j+1}^{l} + u_{i,j-1}^{l}) + u_{i,j}^{l-1}],$$

This solution is in turn used to obtain a new and improved $u_{i,j}^{(2)}$. We continue this process till we obtain a result which satisfies some specific convergence criterion. While iterating we can test the difference between the temporary vector and the solution got and to compare this difference with the chosen convergence tolerance. We can iterate through the given number of steps or until the condition on the convergence tolerance is met.

2.3 Crank-Nicolson Scheme

If we now combine the implicit and explicit methods in a slightly more general approach, we can set up an equation [2], [3]

$$\frac{\theta}{\Delta x^2}(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) + \frac{1-\theta}{\Delta x^2}(u_{i-1,j-1} - 2u_{i,j-1} + u_{i+1,j-1}) = \frac{1}{\Delta t}(u_{i,j} - u_{i,j-1}),$$

which for $\theta = 0$ yields the forward formula for the first derivative and the explicit scheme, while $\theta = 1$ yields the backward formula and the implicit scheme. These two schemes are called the backward and forward Euler schemes, respectively. For $\theta = 1/2$ we obtain a new scheme after its inventors, Crank and Nicolson.

Using our previous definition of α we can rewrite this equation for the Crank-Nicolson method as

$$-\alpha u_{i-1,j} + (2 + 2\alpha)u_{i,j} - \alpha u_{i+1,j} = \alpha u_{i-1,j-1} + (2 + 2\alpha)u_{i,j-1} + \alpha u_{i+1,j-1},$$

or in a matrix-vector form:

$$(2I + \alpha B)V_j = (2I - \alpha B)V_{j-1}$$

with the identity matrix I and the matrix B defined as

$$\mathrm{B} = \begin{bmatrix} 2 & -1a & 0 & 0 & \dots \\ -1 & 2a & -1 & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & -1 \\ 0 & 0 & \dots & -1 & 2 \end{bmatrix}$$

Therefore the solution can be divided into two parts. Firstly, we calculate the right-hand part of the equation above, which is actually the right-hand of a set of linear equations. Secondly, we solve the set of linear equations using Gaussian elimination for tridiagonal matrices (Thomas algorithm).

The implementation of the Crank-Nicolson algorithm can be described as

```
  //create array U
    mat U(n,m);

  //boundary conditions
    U.col(0).ones();
    U.col(m-1).zeros();
  //initial conditions
    U.row(0).zeros();

    vec V(m);
    V=U.row(0);   //initial values, t=0;

 //create the tridiagonal matrix A
    mat A(m,m);
    A.zeros();
    A.diag().fill(2+2*alpha);
    A.diag(1).fill(-alpha);
```

```
A.diag(-1).fill(-alpha);

vec V_new(m);

int j=1;
while (j<n){

for (int i=0; i<m; i++)
{ V_new(i)=alpha*V(i-1) + (2-2*alpha)*V(i) + alpha*V(i+1);}
 V=V_new;   //reassign vector V

tridiag solver;   //initialize solver
    solver.tridiag_solver(A, V, m);
  //the solver finds the solution
  //and assign it to the V vector
    V(0)=1.0;    //boundary condition for x=0;
    V(m-1)=0.0;  //boundary condition for x=L=1;

j++;}       //increment time step
```

Truncation errors of the methods are discussed later.

2.4 Truncation errors and stability properties of the finite difference methods

2.4.1 One dimensional problems

In the explicit scheme the approximation formulas for the first and second derivatives are used: [2]

$$u_t \approx \frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = \frac{u(x_i,t_j+\Delta t)-u(x_i,t_j)}{\Delta t}$$

and

$$u_{xx} \approx \frac{u(x_i+\Delta x,t_j)-2u(x_i,t_j)+u(x_i-\Delta x,t_j)}{\Delta x^2},$$

what gives a local approximation error $O(\Delta x^2)$ and $O(\Delta t)$ for the second and the first derivatives correspondingly. Thus the numerical errors of the method are proportional to the time step and the square of the space step:

$\Delta u = O(k) + O(h^2)$

Moreover, the explicit scheme is known to have a very weak stability condition, written as [2]

$$\frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}. \tag{2.3}$$

This means that if, for example, $\Delta x = 0.01$, then $\Delta t = \frac{\Delta x^2}{2} = 5 \times 10^{-5}$.

As for the implicit scheme, the approximation for the derivatives is also used here. This means that in that case the truncation erroe is the same. However, there are no conditions on Δx and Δt, thus the implicit scheme is always numerically stable and convergent but usually more numerically intensive than the explicit method as it requires solving a system of numerical equations on each time step [3]. The errors are linear over the time step and quadratic over the space step:

$\Delta u = O(k) + O(h^2)$.

In the Crank-Nicolson scheme we use the Taylor expansions the approximations for the derivative for $\frac{\partial u(x,t)}{\partial x^2}$, which gives the truncation error $O(\Delta x^2)$. But for the scheme we also require the Taylor expansion of $u(x + \Delta x, t + \Delta t)$ and $u(x - \Delta x, t + \Delta t)$ around $t = t + \Delta t/2$, what results in a truncation error $O(\Delta t^2)$. But the method is also not depend on relation between Δx and Δt. The scheme is always numerically stable and convergent but usually more numerically intensive as it requires solving a system of numerical equations on each time step [3]. The errors are quadratic over both the time step and the space step:

$\Delta u = O(k^2) + O(h^2)$.

Usually the Crank–Nicolson scheme is the most accurate scheme for small time steps. The explicit scheme is the least accurate and can be unstable, but is also the easiest to implement and the least numerically intensive. The implicit scheme works the best for large time steps [3].

2.4.2 Two dimensional problems

The explicit scheme is know to have a very weak stability, that depends on constraints on Δt and h. In one dimensional problem the stability condition is

$$\frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}. \tag{2.4}$$

.

Let us now obtain correlation between Δt and $\Delta x = \Delta y = h$ for the explicit scheme in two dimensions that provides a stable solution. In order to derive this relation we need

some results from studies of iterative schemes. If we require that our solution approaches a definite value after a certain amount of time steps we need to require that the so-called spectral radius $\rho(A)$ of our matrix A satisfies the condition

$$\rho(A) < 1, \tag{2.5}$$

The spectral radius is defined as

$$\rho(A) = max|\lambda| : det(A - \lambda I) = 0, \tag{2.6}$$

which is interpreted as the smallest number such that a circle with radius centered at zero in the complex plane contains all eigenvalues of A. If the matrix is positive definite, the condition in $\rho(A) < 1$ is always satisfied.

We can obtain closed-form expressions for the eigenvalues of A. To achieve this it is convenient to rewrite the calculation molecule for the 2+1 dimensional explicit scheme as

$$u_{i,j}^{l+1} = u_{i,j}^{l} + \alpha[u_{i+1,j}^{l} + u_{i-1,j}^{l} + u_{i,j+1}^{l} + u_{i,j-1}^{l} - 4u_{i,j}^{l}],$$

$$u_{i,j}^{l+1} = (\alpha[u_{i+1,j}^{l} + u_{i-1,j}^{l}] + u_{i,j}^{l}) + (\alpha[u_{i,j+1}^{l} + u_{i,j-1}^{l}] + (-4\alpha)u_{i,j}^{l}]),$$

$$u_{i,j}^{l+1} = AV_j + BV_i,$$

where V_j is the j^{th} column of the matrix $u(i,j)l$ and V_i is the i^{th} row of the matrix $u(i,j)l$. The matrices A and B read as

$$\mathrm{A} = \begin{bmatrix} 1 & \alpha & 0 & 0 & \dots \\ \alpha & 1 & \alpha & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \alpha \\ 0 & 0 & \dots & \alpha & 1 \end{bmatrix}$$

$$\mathrm{B} = \begin{bmatrix} -4\alpha & \alpha & 0 & 0 & \dots \\ \alpha & -4\alpha & \alpha & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \alpha \\ 0 & 0 & \dots & \alpha & -4\alpha \end{bmatrix}$$

Let us now obtain closed-form expressions for the eigenvalues of B. First we rewrite it as

$$B = I - \alpha D;$$

with I as a unity matrix and D read as

$$\mathrm{D} = \begin{bmatrix} 4 & -1 & 0 & 0 & \dots \\ -1 & 4 & -1 & 0 & \dots \\ 0 & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & -1 \\ 0 & 0 & \dots & -1 & 4 \end{bmatrix}.$$

The eigenvalues of B are $\lambda_i = 1 - \alpha\mu_i$, with μ_i being the eigenvalues of D. To find μ_i we note that the matrix elements of D are

$$d_{ij} = 4\delta_{ij} - \delta_{i+1j} - \delta_{i-1j},$$

meaning that we have the following set of eigenequations for component i

$$(Dx)_i = \mu_i x_i,$$

resulting in

$$(Dx)_i = \sum_{j=1}^{n} (4\delta_{ij} - \delta_{i+1j} - \delta_{i-1j})x_j = 4x_i - x_{i+1} - x_{i-1} = \mu_i x_i.$$

If we assume that x can be expanded in a basis of $x = (sin(\theta), sin(2\theta), ..., sin(n\theta))$ with $\theta = l\pi/n+1$, where we have the endpoints given by $x_0 = 0$ and $x_{n+1} = 0$, we can rewrite the last equation as

$$4sin(i\theta) - sin((i+1)\theta) - sin((i-1)\theta) = \mu_i sin(i\theta),$$

or

$$4(1 - cos(i\theta))sin(i\theta) = \mu_i sin(i\theta),$$

which is nothing but

$$4(1 - cos(i\theta))x_i = \mu_i x_i,$$

with eigenvalues $\mu_i = 4 - 4cos(i\theta)$. Therefore the requirement $\rho(B) < 1$ result in

$$-1 < (1 - 4\alpha(1 - cos(i\theta)) < 1,$$

which is satisfied only if

$$\alpha < 4(1 - cos(i\theta))^{-1} < 1,$$

resulting in

$$\alpha \leq \frac{1}{4}$$

or

$$\frac{\Delta t}{\Delta x^2} \leq \frac{1}{4}. \tag{2.7}$$

.

The equal steps implemented for the matrix A from

$$u_{i,j}^{l+1} = AV_j + BV_i,$$

result in the relation

$$\frac{\Delta t}{\Delta x^2} \leq 1, \tag{2.8}$$

that is already met by $\frac{\Delta t}{\Delta x^2} \leq \frac{1}{4}$.

This means that if, for example, $\Delta x = 0.01$, then $\Delta t = \frac{\Delta x^2}{4} = 2.5 \times 10^{-5} = 0.000025$. Thus an explicit scheme for two dimensions has even weaker stability than the same scheme for one-dimensional PDE.

2.5 Monte Carlo and random walks methods for solving diffusion equation

2.5.1 One dimensional problems

A random walk is a mathematical formalization of a path that consists of a succession of random steps. For example, the path traced by a molecule as it travels in a liquid or a gas, the search path of a foraging animal, the price of a fluctuating stock and the financial status of a gambler can all be modeled as random walks, although they may not be truly random in reality. Random walks have been used in many fields: ecology, economics, psychology, computer science, physics, chemistry, and biology. Random walks explain the observed behaviors of many processes in these fields, and thus serve as a fundamental model for the recorded stochastic activity.

FIGURE 2.1: All possible random walk outcomes after 5 flips of a fair coin.

Various different types of random walks are of interest. Often, random walks are assumed to be Markov chains or Markov processes, but other, more complicated walks are also of interest. Some random walks are on graphs, others on the line, in the plane, in higher dimensions, or even curved surfaces, while some random walks are on groups.

All Monte Carlo schemes used are based on Markov processes in order to generate new random states. A Markov process is a random walk with a selected probability for making a move. The new move is independent of the previous history of the system. The Markov process is used repeatedly in Monte Carlo simulations in order to generate new random states. The reason for choosing a Markov process is that when it is run for a long enough time starting with a random state, we will eventually reach the most likely state of the system. In thermodynamics, this means that after a certain number of Markov processes we reach an equilibrium distribution. This mimicks the way a real system reaches its most likely state at a given temperature of the surroundings. To reach this distribution, the Markov process needs to obey two important conditions, that of ergodicity and detailed balance. These conditions impose constraints on our algorithms for accepting or rejecting new random states (sampling rules).

An elementary example of a random walk is the random walk on the integer number line, Z, which starts at 0 and at each step moves +1 or 1 with equal probability. This walk can be illustrated as follows. A marker is placed at zero on the number line and a fair coin is flipped. If it lands on heads, the marker is moved one unit to the right. If it lands on tails, the marker is moved one unit to the left. After five flips, the marker could now be on 1, 1, 3, 3, 5, or 5. With five flips, three heads and two tails, in any order, will land on 1. There are 10 ways of landing on 1 (by flipping three heads and two tails), 10 ways of landing on 1 (by flipping three tails and two heads), 5 ways of landing on 3 (by flipping four heads and one tail), 5 ways of landing on 3 (by flipping four tails and one head), 1 way of landing on 5 (by flipping five heads), and 1 way of landing on 5 (by flipping five tails). See the figure below for an illustration of the possible outcomes of 5 flips.

To define this walk formally, take independent random variables Z_1, $Z_2, \ldots$, where each variable is either 1 or 1, with a 50% probability for either value, and set $S_0 = 0$ and $S_n = \sum_{j=1}^{n} Z_j$. The series $\{S_n\}$ is called the simple random walk on $\mathbb{Z}$. This series (the sum of the sequence of 1s and 1s) gives the distance walked, if each part of the walk is of length one. The expectation $E(S_n)$ of S_n is zero. That is, the mean of all coin flips approaches zero as the number of flips increases. This follows by the finite additivity property of expectation:

$\mathrm{E}(\mathrm{S}_n) = \sum_{j=1}^{n} E(Z_j) = 0.$

In our case in order to solve the given one-dimensional equation with the particular boundary and initial conditions, we will use the following algorithm and sampling.

1. We choose a fixed number of particles (walkers) at the point ($x = 0$). Choose also the final time and the time step, define a step length for the walkers (l_o).

2. Create a vector, that contains the position of every particle (at the beginning there are only walkers at the position $x = 0$).

3. Set up a loop over the time steps. In each time step there is a loop over all the particles (their positions). For each particle we assume the probabilities of jumping back and forward are equal. Therefore for each particle we draw a random number ϵ

$$\epsilon \in [0; 1],$$

and make the particle move in accordance with the following rule

$$x_{new} = x_{old} - l_o \quad if \quad \epsilon \leq 0.5,$$

$$x_{new} = x_{old} + l_o \quad if \quad \epsilon > 0.5.$$

This is the sampling rule in our case.

4. When the move is done one should apply the boundary conditions. Therefore if the new move moves particle x_i beyond $x = 1$ or at $x = 1$, we remove this particle from the vector (according to the boundary condition $x(1) = 0$). If the old position (before the move) of the particle is $x = 0$, we add a new particle at $x = 0$ after the move due to conservation at $x = 0$ (this meets the condition $x(0) = 1$). For the same reason we delete the particle, if the new move is back to zero. Finally, if the new move moves the particle beyond $x = 0$ we remove that new particle ($x \in [0; 1]$).

5. We repeat steps 3, 4 till the final time. Then we build a histogram by calculating the number of particles in each small box of length l_o.

2.5.2 Two dimensional problems

If we now have a two dimensional problem it is easy to increase the algorithm for solving the diffusion equation from the previous section to two dimensions. But in that case we create two vectors of positions (for x and y) and define ϵ for each of the x_i and y_i. Moreover, we should take into account the boundary conditions for the two dimension problem, that, according to the task, suggests us to preserve particles at $x = 0$ for any y

and sets infinite boundary conditions on y. Thus, the algorithm for solving the diffusion equation is read as

1. Choose a fixed number of particles (walkers) at the point $(x = 0; y = 0)$. Choose also the final time and the time step, define a step length for the walkers (l_o).

2. Create two vectors (for x and y positions), that contain the position of every particle (at the beginning there are only walkers at the position $(x = 0; y = 0)$).

3. Set up a loop over the time steps. In each time step there is a loop over all the particles (their positions). For each particle we assume the probabilities of jumping back and forward are equal. Therefore for each particle we draw two random numbers ϵ_x and ϵ_y for each coordinate correspondingly

$$\epsilon\_x, \epsilon\_y \in [0; 1],$$

and make the particle move in accordance with the following rule

$$x_{new} = x_{old} - l_o \quad if \quad \epsilon\_x \leq 0.5,$$

$$x_{new} = x_{old} + l_o \quad if \quad \epsilon\_x > 0.5.$$

$$y_{new} = y_{old} - l_o \quad if \quad \epsilon\_y \leq 0.5,$$

$$y_{new} = y_{old} + l_o \quad if \quad \epsilon\_y > 0.5.$$

This is the sampling rule for two dimensions.

4. When the move is done, one should apply the boundary conditions. Therefore if the new move moves particle x_i or y_i beyond $x = 1$ or $y = 1$ or at $x = 1$ or $y = 1$, we remove this particle from the vector (according to the boundary condition $x(1, y) = 0$). If the old position (before the move) of the particle is $(x = 0, y_i)$, we add a new particle at $(x = 0, y_i)$ after the move due to conservation at $x = 0$ (this meets the condition $x(0, y) = 1$). For the same reason we delete the particle, if the new move is back to zero ($x_{new} = 0$). Finally, if the new move moves the particle beyond $x = 0$ or $y = 0$ we remove that new particle ($x \in [0; 1]$, $y \in [0; 1]$).

Moreover, in order to meet the conditions for y, we should preserve the number of particles at $y = 0$ and $y = 1$. Therefore if we move from the $y = 0$ and $y = 1$ we insert the particle

$$(x_{new}; y_{new} - 1) \quad if \quad y > 1.0,$$

$$(x_{new}; y_{new} + 1) \quad if \quad y < 0.0.$$

5. We repeat steps 3, 4 till the final time. Then we build a histogram by calculating the number of particles in each small box $l_o \times l_o$.

Chapter 3

Experimental Setup

3.1 Closed-form solution

3.1.1 One dimensional problem

In order to study the numerical accuracy of the results got by the described methods we will need to find the closed form solution to the problem. To find the closed-form solution, we will need the stationary solution (steady-state solution) [2]. The solution to the steady-state problem is on the form $u(x) = Ax + b$. The solution for the steady-state case u_s that obeys the described boundary conditions is

$$u_s(x) = 1 - x.$$

with

$$u_s(x = 0) = 1,$$

$$u_s(x = d = 1) = 0.$$

Therefore we define a new function $v(x) = u(x) - u_s(x)$ with boundary conditions $v(0) = v(d) = 0$. Let us now rewrite the problem as

$$\frac{\partial v}{\partial t} + \frac{\partial (1-x)}{\partial t} = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 (1-x)}{\partial x^2}, \tag{3.1}$$

$$\frac{\partial v}{\partial t} = \frac{\partial^2 v}{\partial x^2}, \tag{3.2}$$

and to find an analytical solution for it. In order to do it we use the method of separation variables [4]. Assume

$$v(x,t) = X(x)T(t), \tag{3.3}$$

therefore

$$\frac{1}{T}\frac{\partial T}{\partial t} = \frac{1}{X}\frac{\partial^2 X}{\partial x^2}. \tag{3.4}$$

As far as both part of the equation depend on different variables, it is obvious that both of them are equal to a particular constant, let's say $-\lambda^2$. Using this constant, we can rewrite the problem as a couple of equations:

$$\frac{1}{T}\frac{\partial T}{\partial t} = \frac{1}{X}\frac{\partial^2 X}{\partial x^2} = -\lambda^2, \tag{3.5}$$

$$X'' + X\lambda^2 = 0, \tag{3.6}$$

$$T' + T\lambda^2 = 0, \tag{3.7}$$

Suppose the case that ¿ 0. Then there exist real numbers A, B, C such that

$$T(t) = Ce^{-\lambda^2 t}, \tag{3.8}$$

and

$$X(x) = A\sin(\lambda\, x) + B\cos(\lambda\, x). \tag{3.9}$$

Assuming now that A, B, C are real and taking into account the boundary conditions, we get:

$$X(0) = 0 \tag{3.10}$$

with

$$B = 0, \tag{3.11}$$

and

$$X(1) = A\sin(\lambda\, x) = 0; \tag{3.12}$$

where $\lambda = n\pi$, $n = \pm 1, \pm 2, \ldots$ (in order to make sin equal to zero as it is said in the boundary conditions).

For every λ (for every n) we can define $T(t)$ the following way:

$$T(t) = Ce^{-\lambda^2 t} = Ce^{-n^2\pi^2 t}. \tag{3.13}$$

General solution for the method of separation of variables (the Fourier method, [4]) can be read as

$$v(x,t) = \sum_{n=1}^{\infty} A_n \sin(n\pi x) \exp(-n^2\pi^2 t). \tag{3.14}$$

Using the initial condition for the function $v(x,t)$ we get

$$v(x,0) = u(x,0) - 1 + x = x - 1, \tag{3.15}$$

$$x - 1 = \sum_{n=1}^{\infty} A_n \sin(n\pi x). \tag{3.16}$$

If we multiply the both parts of the above equation by $\sin(m\pi x)$ and integrate them, the result will be

$$\int_0^{d=1} (x-1)\sin(m\pi x)\mathrm{d}x = \sum_{n=1}^{\infty} \int_0^{d=1} \sin(m\pi x)\sin(n\pi x)\mathrm{d}x, \tag{3.17}$$

thus

$$A_n = 2\int_0^{d=1} (x-1)\sin(n\pi x)\mathrm{d}x = -\frac{2}{n\pi}. \tag{3.18}$$

Finally, the closed-form solution for the function $u(x,t)$ will have the following form:

$$u(x,t) = 1 - x - \sum_{n=1}^{\infty} \frac{2}{n\pi} \sin(n\pi x) \exp(-n^2\pi^2 t). \tag{3.19}$$

3.1.2 Two dimensional problem

Let us now find an analytical solution for the 2+1-dimensional diffusion problem. For that purpose we have already increased the given boundary and initial conditions for the 1+1-dimensional problem to two dimensions. Now the conditions are read:

$$u(x, y, t = 0) = 0 \quad 0 < x, y < d \quad (initialconditions)$$

and the boundary conditions

$$u(0, y, t) = 1 \quad t > 0, y \in [0, d]$$

and

$$u(d, y, t) = 0 \quad t > 0, y \in [0, d].$$

To find the closed-form solution, we need the stationary solution (steady-state solution) [2]. The solution to the steady-state problem in two dimensions is on the form $u(x, y) = Ax + b$. The solution for the steady-state case u_s that obeys the described boundary conditions is

$$u_s(x, y) = 1 - x.$$

with

$$u_s(x = 0, y) = 1,$$

$$u_s(x = d = 1, y) = 0.$$

Therefore we define a new function $v(x) = u(x) - u_s(x)$ with boundary conditions $v(0, y) = v(d, y) = 0$ and initial conditions $v(x, y, t = 0) = f(x)$. Let us now rewrite the problem as

$$\frac{\partial v}{\partial t} + \frac{\partial (1 - x)}{\partial t} = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 (1 - x)}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 (1 - x)}{\partial y^2}, \tag{3.20}$$

$$\frac{\partial v}{\partial t} = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}, \tag{3.21}$$

and to find an analytical solution for it.

Here we use again a method of separation of variables. Our first step is to make the ansatz

$$v(x, y, t) = F(x, y)T(t), \tag{3.22}$$

resulting in the equation

$$FT = F_{xx}T + F_{yy}T. \tag{3.23}$$

Therefore

$$\frac{1}{T}\frac{\partial T}{\partial t} = \frac{1}{F}(\frac{\partial^2 F}{\partial x^2} + \frac{\partial^2 F}{\partial y^2}). \tag{3.24}$$

As far as both part of the equation depend on different variables, it is obvious that both of them are equal to a particular constant, let us say $-\nu^2$. Using this constant, we can rewrite the problem as a couple of equations:

$$\frac{1}{T}\frac{\partial T}{\partial t} = \frac{1}{F}(\frac{\partial^2 F}{\partial x^2} + \frac{\partial^2 F}{\partial y^2}) = -\nu^2, \tag{3.25}$$

$$F_{xx} + F_{yy} + F\nu^2 = 0, \tag{3.26}$$

$$T' + T\nu^2 = 0, \tag{3.27}$$

Now we can in turn make the following ansatz for the x and y dependent part

$$F(x, y) = X(x)Y(y), \tag{3.28}$$

which results in

$$X_{xx}Y + Y_{yy}X + \nu^2 XY = 0, \tag{3.29}$$

$$\frac{1}{X}\frac{\partial^2 X}{\partial x^2} + \frac{1}{Y}\frac{\partial^2 Y}{\partial y^2} + \nu^2 = 0, \tag{3.30}$$

As earlier, the parts of this equation depend on different variables, thus we rewrite it as

$$\frac{1}{X}\frac{\partial^2 X}{\partial x^2} = -\frac{1}{Y}\frac{\partial^2 Y}{\partial y^2} - \nu^2 = -k^2, \tag{3.31}$$

Since the left-hand side and the right-hand side are again independent of each other, we separate the latter equation into two independent equations, one for x and one for y, namely

$$X_{xx} + k^2 X = 0, \tag{3.32}$$

$$Y_{yy} + \nu^2 Y - k^2 Y = 0, \tag{3.33}$$

$$Y_{yy} + \rho^2 Y = 0, \tag{3.34}$$

with $\rho^2 = \nu^2 - k^2$.

The second step is to solve these differential equations, with X and Y having trigonometric functions as solutions and T having exponent as a solution, viz.

$$T(t) = Ee^{-\nu^2 t}, \tag{3.35}$$

and

$$X(x) = A\sin(kx) + B\cos(kx), \tag{3.36}$$

$$Y(y) = C\sin(\rho y) + D\cos(\rho y). \tag{3.37}$$

The boundary conditions require that $F(x, y) = X(x)Y(y)$ are zero at the boundaries, meaning that $X(0) = X(1) = Y(0) = Y(1) = 0$. This yields the solutions

$$X_m(x) = \sin(m\pi x), \tag{3.38}$$

$$Y_n(y) = \sin(n\pi y), \tag{3.39}$$

with $k = m\pi$ and $\rho = n\pi$, or

$$F_{mn}(x, y) = \sin(m\pi x)\sin(n\pi y), \tag{3.40}$$

With $\rho^2 = \nu^2 - k^2$ we have an eigenspectrum $\nu = \sqrt{\rho^2 + k^2}$ or $\nu = \pi\sqrt{n^2 + m^2}$

Thus the solution for $T(t)$ is

$$T_{mn}(t) = E_{mn}e^{-\nu^2 t} = E_{mn}e^{-\pi^2(n^2+m^2)t}, \tag{3.41}$$

with the general solution of the form

$$v(x, y, t) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} E_{mn}\sin(m\pi x)\sin(n\pi y)\exp(-\pi^2(n^2 + m^2)t). \tag{3.42}$$

Using the initial condition for the function $v(x, y, t)$ we can write the following

$$v(x, y, 0) = u(x, y, 0) - 1 + x = x - 1, \tag{3.43}$$

$$x - 1 = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} E_{mn} \sin(m\pi x) \sin(n\pi y). \tag{3.44}$$

If we multiply the both parts of the above equation by $\sin(w\pi x)$ and integrate them, the result will be

$$\int_0^{d=1} (x-1) \sin(w\pi x) \mathrm{d}x = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} E_{mn} \sin(n\pi y) \int_0^{d=1} \sin(m\pi x) \sin(w\pi x) \mathrm{d}x, \tag{3.45}$$

thus

$$\frac{-1}{w\pi} = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} E_{mn} \sin(n\pi y) \frac{1}{2} \delta_{wm}, \tag{3.46}$$

$$E_{mn} = -\frac{2}{m\pi sin(n\pi y)}. \tag{3.47}$$

Finally, the closed-form solution for the function $u(x, y, t)$ will have the following form:

$$u(x, y, t) = 1 - x - \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{2}{m\pi} \sin(m\pi x) \exp(-\pi^2(n^2 + m^2)t). \tag{3.48}$$

3.2 Numerical experiment

3.2.1 1+1-dimensional diffusion equation

In this part we will implement the described above algorithms in order to solve the diffusion equation from the chapter one. We test our solvers for $\Delta x = 0.1$, $\Delta x = 0.01$. Thus, according to the condition for the stability of the explicit scheme, we get $\Delta t \leq 0.005$ and $\Delta t \leq 0.00005$ correspondingly.

First we look at the case $t = 0.1$, $\Delta x = 0.1$ ($m = 11$), resulting for Δt in $\Delta t = 0.005$ ($n = 21$): Fig. 3.1 is for the final time, Fig. 3.2 is for the first quarter of the whole period of time.

The next interesting case is for the $\Delta x = 0.01$. Thus we have $t = 0.1$, $\Delta x = 0.01$ ($m = 101$), resulting in $\Delta t = 0.00005$ ($n = 2001$): Fig. 3.3 is for the final time, Fig. 3.4 is for the one fifth of the whole time.

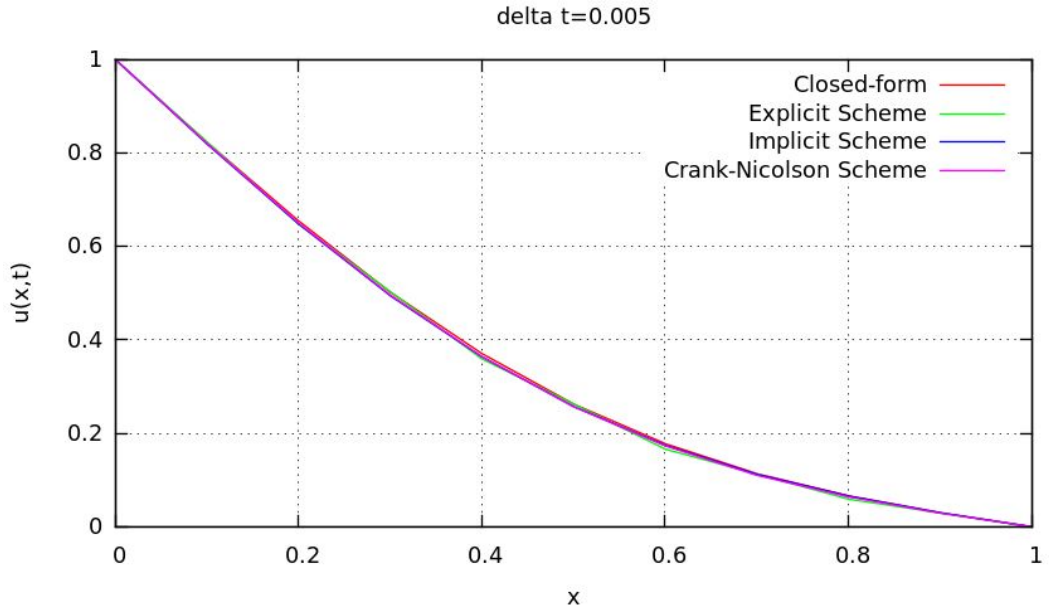

FIGURE 3.1: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.1$, number of points for x $m = 11$, $t\_final = 0.1$, $\Delta t = 0.005$, number of points for t $n = 21$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = t\_final$.

From the figures above we can observe that for the moment $t = t\_final/4$ and $\Delta x = 0.1$ the solution is significantly curved, but all of the three methods give results close to the closed-form solution except small error. For the final time $t = t\_final = 0.1$, $t \ll 1$ the plots for the three methods and for the closed-form solution are equal.

But if we increase the number of points for the coordinate x to $m = 101$ in order to get $\Delta x = 0.01$, the explicit, implicit and Crank-Nicolson schemes present precise results for both $t = t\_final = 0.1$, almost linear solution, closed to the stationary state $1 - x$, and $t = t\_final/5$, smooth but still significantly curved solution.

Therefore one can conclude that in order to get high precisely results by any of the described schemes, it is better to use large ($n \geq 100$) number of points for both the position and time. But we have to take into account the condition for Δx and Δt in the explicit scheme.

Now we will look at the case $t = 1$. Thus we get $\Delta x = 0.1$ ($m = 11$), $\Delta t = 0.005$ ($n = 201$): Fig. 3.5 is for the final time, Fig. 3.6 is for the one twentieth of the whole time.

In the case for $t = 1$ and $\Delta x = 0.01$ ($m = 101$) the other parameters are $\Delta t = 0.00005$ ($n = 20001$): Fig. 3.7 is for the final time, Fig. 3.8 is for the one twentieth of the whole time.

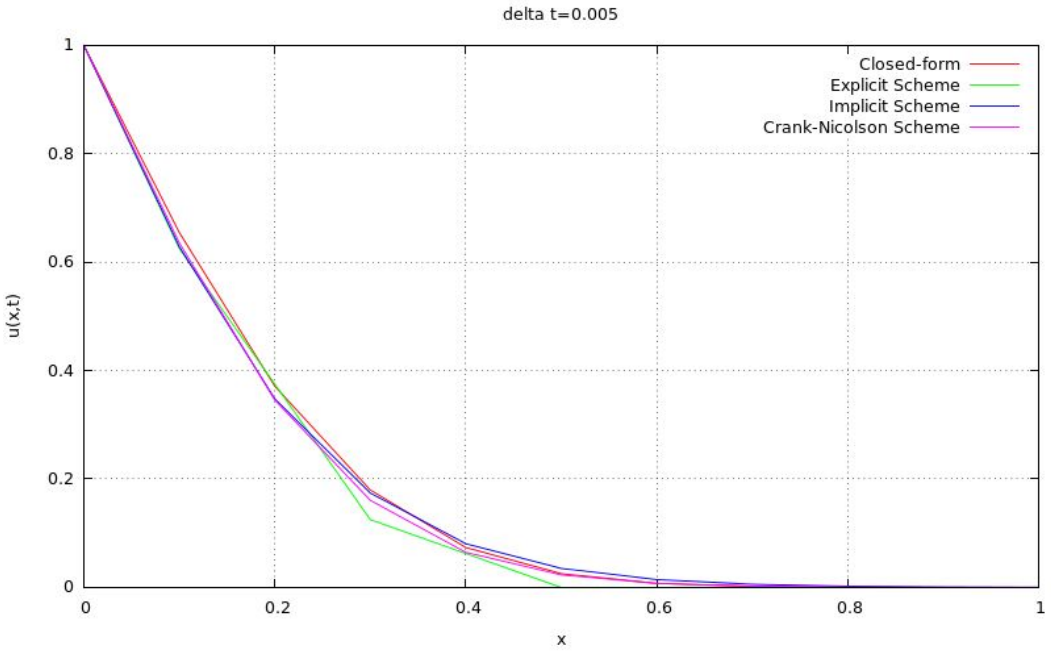

FIGURE 3.2: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.1$, number of points for x $m = 11$, $t\_final = 0.1$, $\Delta t = 0.005$, number of points for t $n = 21$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = \frac{t\_final}{4}$.

From the figures for $t\_final = 1$ we can observe that for the moment $t = t\_final/4$ and $\Delta x = 0.1$ that the solution is significantly curved, all of the three methods give results close to the closed-form solution but with some error. For the final time the plots for the three methods and for the closed-form solution are completely equal to the steady-state solution $1 - x$.

With the increase of the number of points for the coordinate x to $m = 101$ in order to get $\Delta x = 0.01$, the explicit, implicit and Crank-Nicolson schemes present precise results for both $t = t\_final = 1$, stationary state $1 - x$, and $t = t\_final/20$, significantly curved solution.

Therefore, as in the case for $t\_final = 0.1$ we see that for high precisely results we should use large ($n \geq 100$) number of points for both the position and time. All three schemes perform equally precise results, but if the implicit and Crank-Nicolson schemes do it for every Δx and Δt, the explicit method requires Δx and Δt to satisfy a particular condition. The latter make us use the larger number of points for time then for the position, which results in larger number of iterations. Therefore, if one want to avoid it, it is better to choose between Crank-Nicolson and implicit scheme.

It is also interesting to study the behaviour of the explicit scheme around $\frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}$: Fig. 3.9, Fig. 3.10. Thus we can ensure that the explicit method is very sensitive to the relation between Δx and Δt.

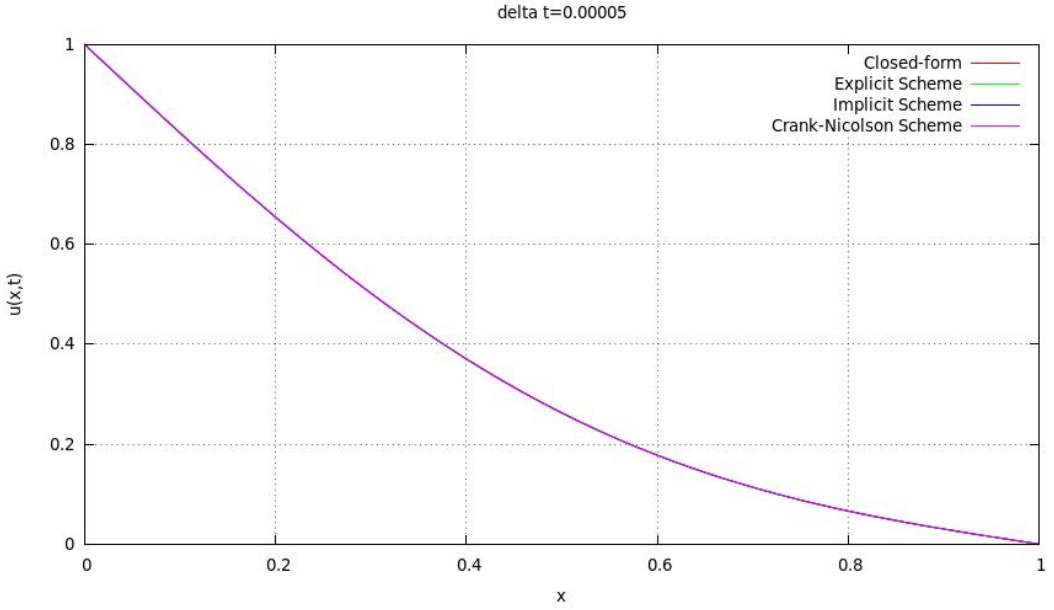

FIGURE 3.3: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 0.1$, $\Delta t = 0.00005$, number of points for t $n = 2001$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = t\_final$.

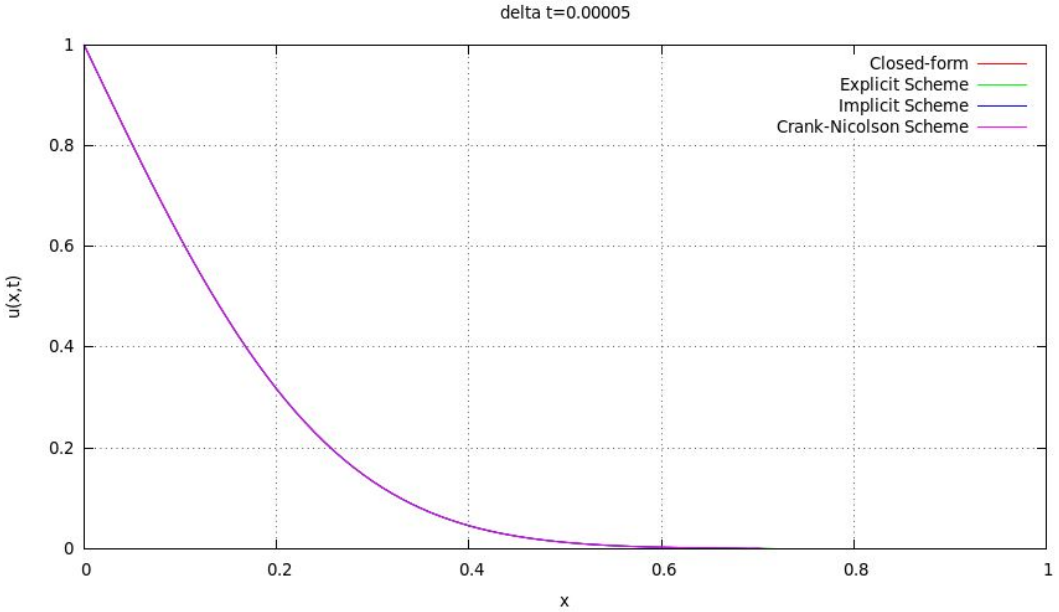

FIGURE 3.4: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 0.1$, $\Delta t = 0.00005$, number of points for t $n = 2001$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = \frac{t\_final}{5}$.

Let us now test the Monte Carlo solver for 1+1-dimensional diffusion equation. We test it for different t_{final} and different length of walk step l_o. We compare the results with the closed-form solution.

From the plots Fig. 3.11, Fig. 3.12 it can be observed that Monte Carlo method performs the solution in the form closed to the steady-state solution $x - 1$ when $t\_final = 1$. But

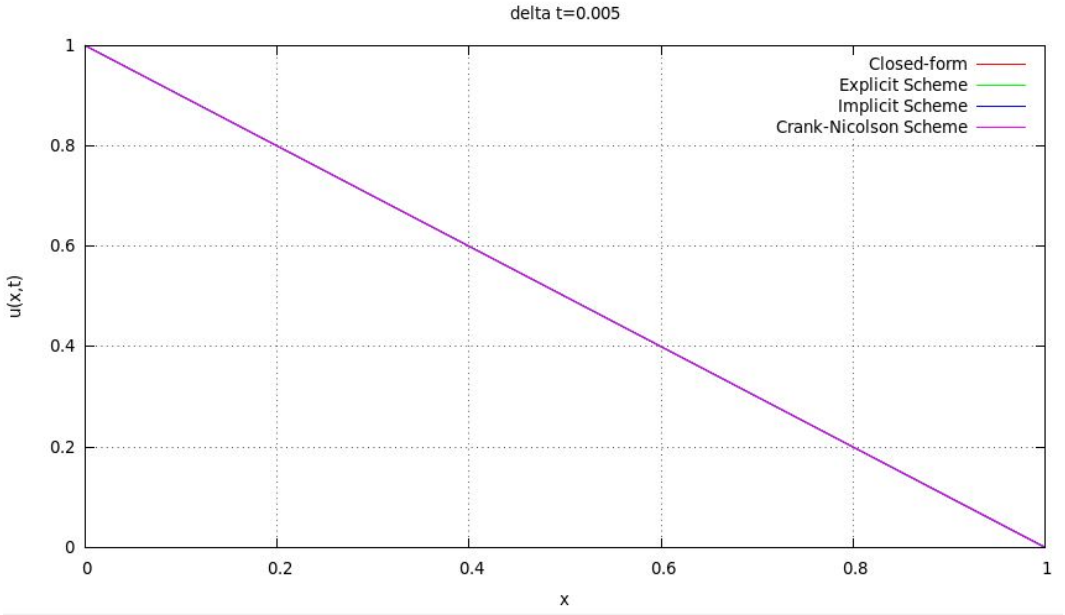

FIGURE 3.5: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.1$, number of points for x $m = 11$, $t\_final = 1$, $\Delta t = 0.005$, number of points for t $n = 201$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = t\_final$.

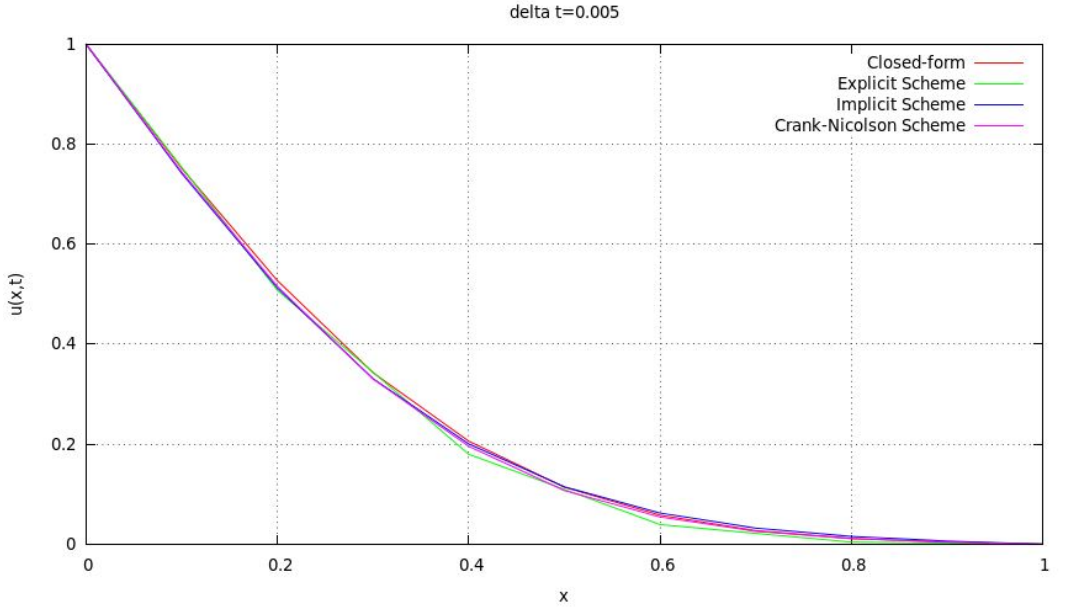

FIGURE 3.6: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.1$, number of points for x $m = 11$, $t\_final = 1$, $\Delta t = 0.005$, number of points for t $n = 201$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = \frac{t\_final}{20}$.

in contrast to finite difference methods the random walks lead us to the steady-state without any initial conditions. Thus, the latter simulate more the real processes in nature. However, the method produces the plot in form of histogram, not a smooth function. Thus the result is not as good as the one by the finite difference methods, Fig. 3.5.

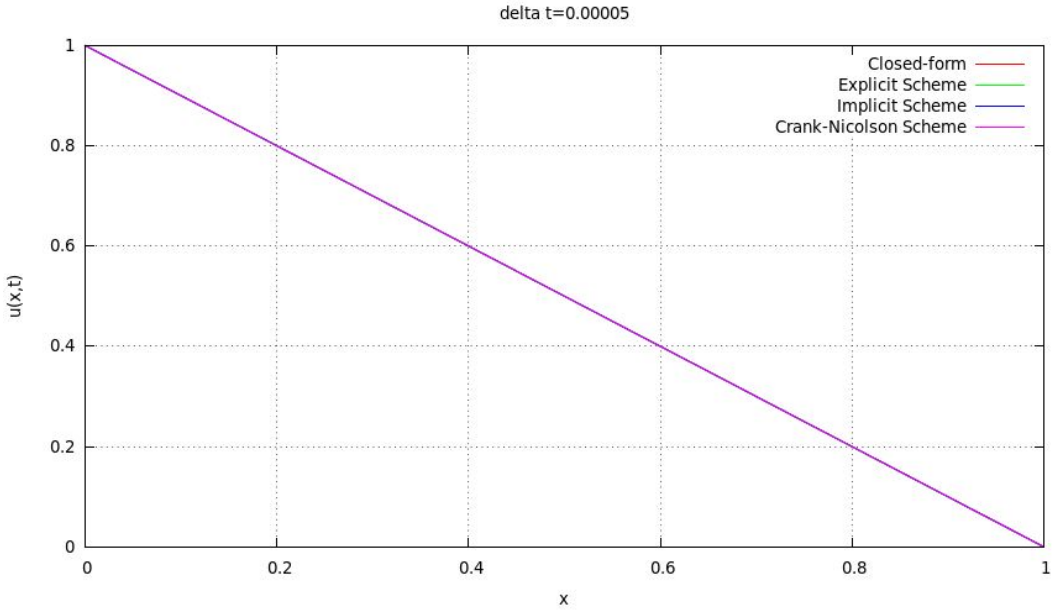

FIGURE 3.7: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 1$, $\Delta t = 0.00005$, number of points for t $n = 20001$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = t\_final$.

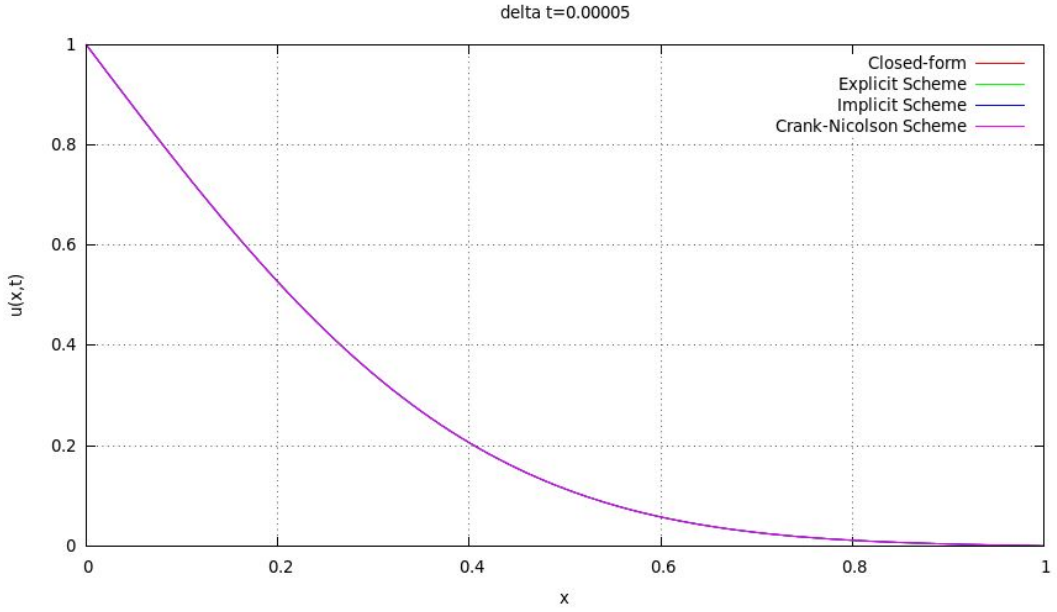

FIGURE 3.8: The solutions for the three approaches for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 1$, $\Delta t = 0.00005$, number of points for t $n = 20001$, $\alpha = 0.5$. The plot corresponds to the moment of time $t = \frac{t\_final}{20}$.

Figures Fig. 3.13, Fig. 3.14 illustrate the solution for the time $t = 0.1$. One can notice that the result by the random walks with constant step l_o, Fig. 3.13, are much closer to the closed-form solution and the solution by the FMDs, Fig. 3.3, than the solution with the varied step length $l = l_o\xi$, with ξ chosen randomly from Gaussian distribution. It results from the fact that diffusion is strongly linked with random walks, and a random

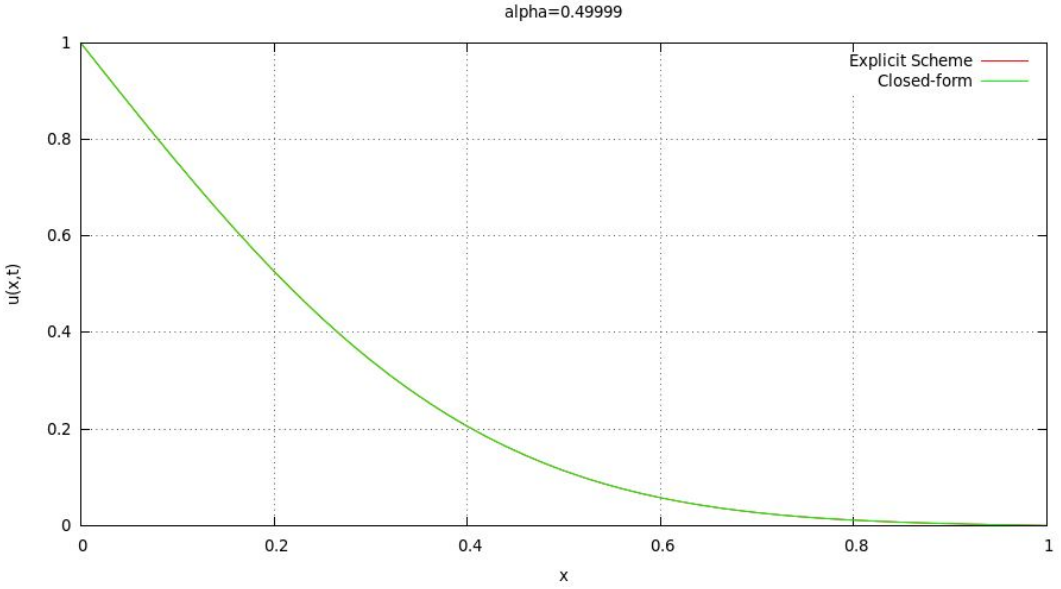

FIGURE 3.9: The solutions got by explicit scheme for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 0.049999$, $\Delta t = 4.(9)e - 5$, number of points for t $n = 1001$, $\alpha = 0.49999$. The plot corresponds to the moment of time $t = t\_final$.

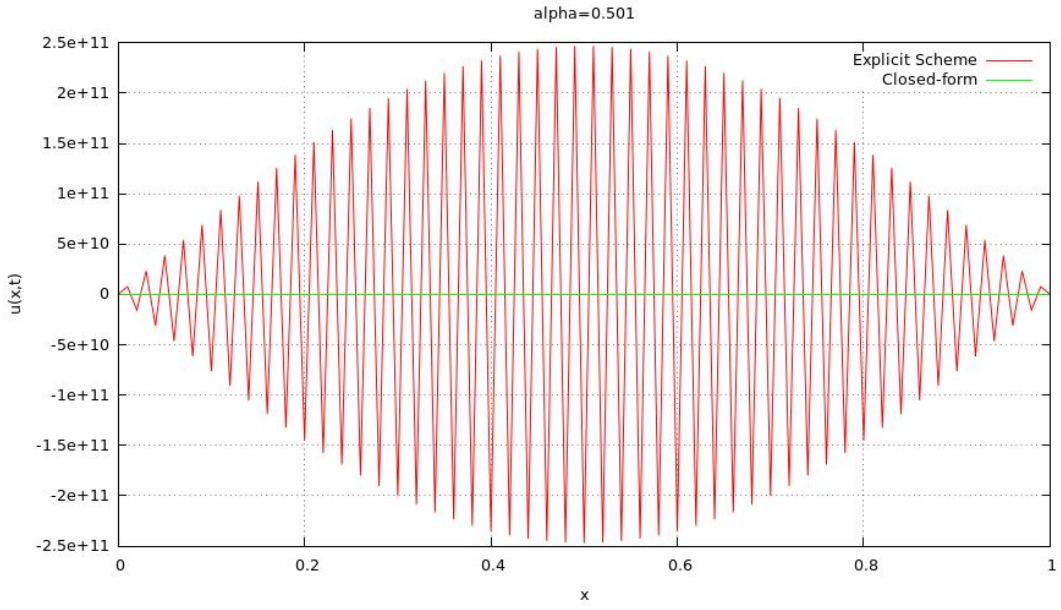

FIGURE 3.10: The solutions got by explicit scheme for $x\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 0.501$, $\Delta t = 5.01e - 5$, number of points for t $n = 10001$, $\alpha = 0.501$. The plot corresponds to the moment of time $t = t\_final$.

walker escapes much more slowly from the starting point than would a free particle. Therefore normal distribution is often used as a model for real-world time series data such as financial markets or cream in a cup of coffee.

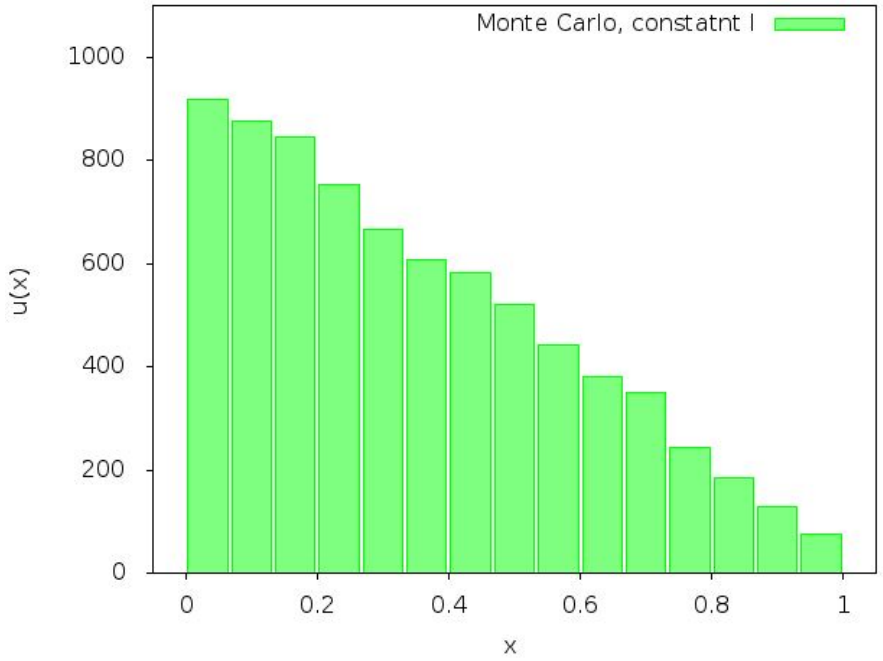

FIGURE 3.11: The solutions by Monte Carlo method with the constant lengths of walk $l_o = 0.06324$, number of walkers at the beginning if the interval $N = 1000$, number of boxes for x $m = 15$, $t\_final = 1.0$, number of points in time t $n = 501$, $\Delta t = 0.002$. The plot corresponds to the moment of time $t = t\_final$.

3.2.2 2+1-dimensional diffusion equation

In this part we will implement the described above algorithms in order to solve the 2+1 diffusion equation from the chapter one. First we look at the solutions by FDMs. The closed-form solution and the plots of the results are presented below.

As it is can be observed from the plots Fig. 3.15, Fig. 3.16, Fig. 3.17, both the explicit and implicit schemes perform precise results for the case $t\_final = 1.0$. The algorithms can easily be calculated and produce valuable results even for small number of steps in time and position (e.g. Fig. 3.18 for implicit scheme). But the implicit scheme could be more preferable because, as it has been mentioned in the previous section, the explicit scheme requires an appropriate relation between Δt and Δx. That relation results in a huge number of steps in time needed for the relatively small number of steps in position. However, in that particular case this drawback did not influence the elapsed time and perform presice solution.

Plots Fig. 3.19, Fig. 3.20, Fig. 3.21 represent the results for closed-form solution and explicit schemes for the final time $t\_final = 0.1$, viz. we do not reach the steady-state in that case. One can see that the result for the explicit scheme repeats the closed-form solution. But due to special features of the implicit scheme, it can lead us only to a steady-state solution.

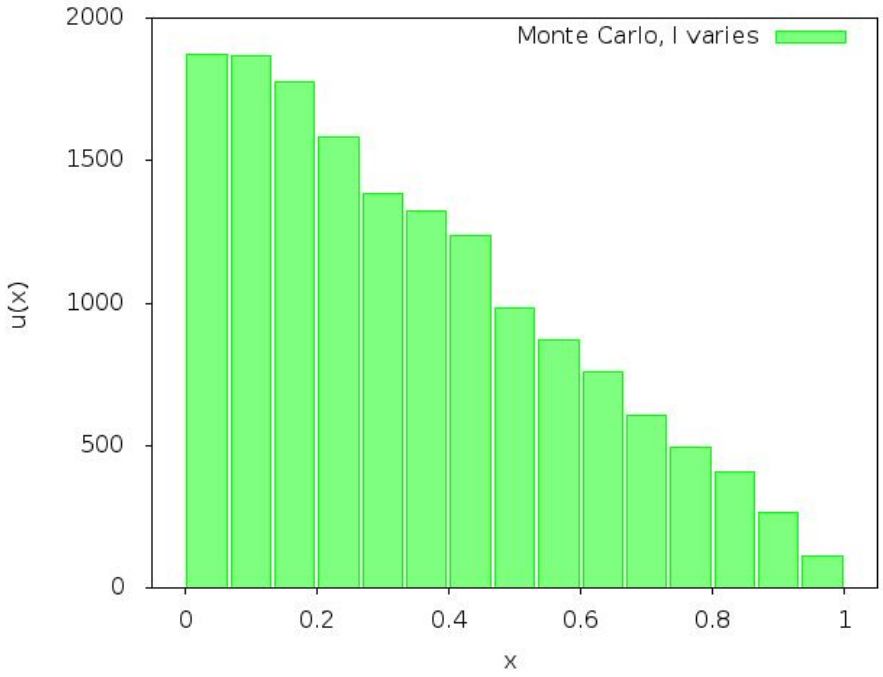

FIGURE 3.12: The solutions by Monte Carlo method with varying lengths of walk l, $l_o = 0.06324$, number of walkers at the beginning if the interval $N = 1000$, number of boxes for x $m = 15$, $t\_final = 1.0$, number of points in time t $n = 501$, $\Delta t = 0.002$. The plot corresponds to the moment of time $t = t\_final$.

On the figures Fig. 3.22, Fig. 3.23 the results of the test of the stability of the explicit scheme in two dimensions are performed. It is clear that if the condition for $\frac{\Delta t}{\Delta x^2} \leq 0.25$ is met the results is close to the analytical solution. But if the condition is not met the solution does not converge.

Both the explicit and implicit schemes gives precise results. But in order to get these results we should know at list the initial conditions. Moreover, there are sometimes limits and conditions for step lengths, like for explicit scheme. And finally, the schemes gives bare results, without showing how the system works in real life. All this details can be met by Monte Carlo and random walk methods. The results of the simulations are presented on the figures below.

The figure Fig. 3.24 shows the histogram for the random walks with the constant length of the step. In that case the number of particles in every bin was calculated by the plotter. But there is also a plot (Fig. 3.25) for the case, when the calculating of the number of particles is organised inside the solver, what helped to avoid some mistakes and get more smooth solution. The latter calculation is also made for the case $t\_final = 0.1$, that is not a steady-state.

The solution of the 2+1-dimensional problem by random walks with the variable step length could be observed from the Fig. 3.27 and Fig. 3.28. This solution, in comparison

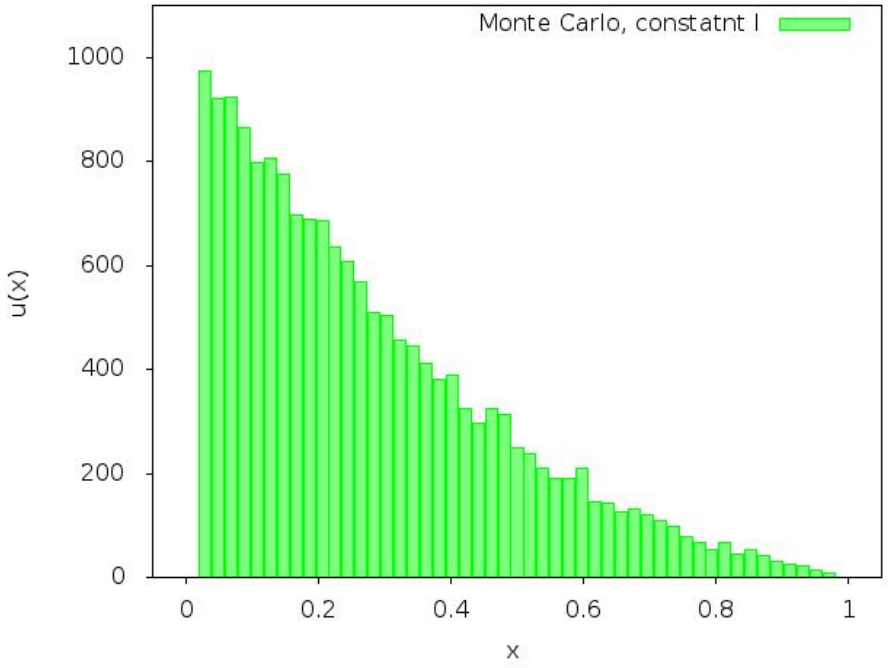

FIGURE 3.13: The solutions by Monte Carlo method with the constant lengths of walk $l_o = 0.02$, number of walkers at the beginning if the interval $N = 1000$, number of boxes for x $m = 50$, $t\_final = 0.1$, number of points in time t $n = 501$, $\Delta t = 0.0002$. The plot corresponds to the moment of time $t = 0.1$.

to that with the stable step length, is closer to the steady state solution, although the shape of the solution is not so smooth.

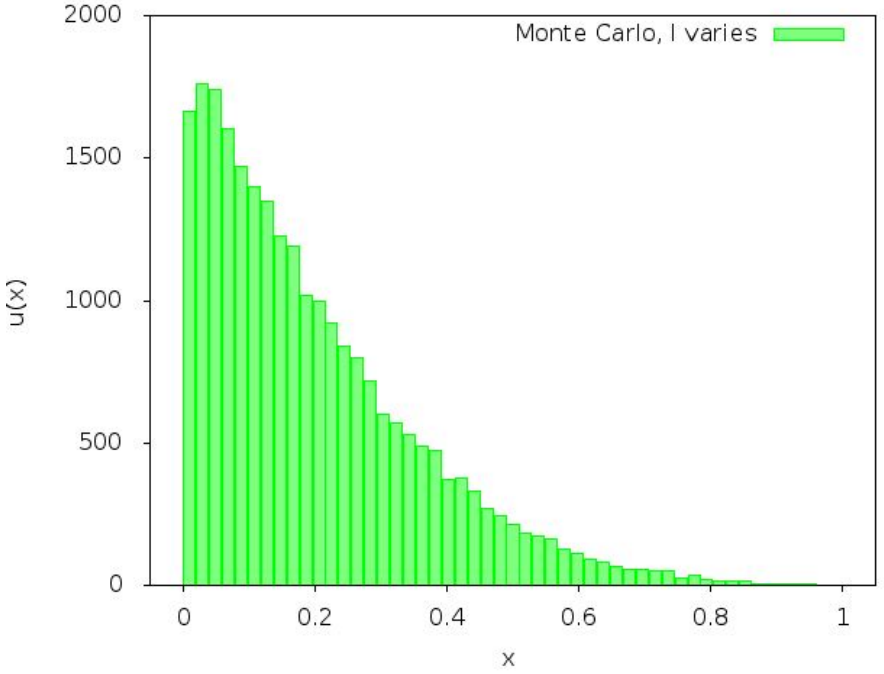

FIGURE 3.14: The solutions by Monte Carlo method with varying lengths of walk l, $l_o = 0.02$, number of walkers at the beginning if the interval $N = 1000$, number of boxes for x $m = 50$, $t\_final = 0.1$, number of points in time t $n = 501$, $\Delta t = 0.0002$. The plot corresponds to the moment of time $t = 0.1$.

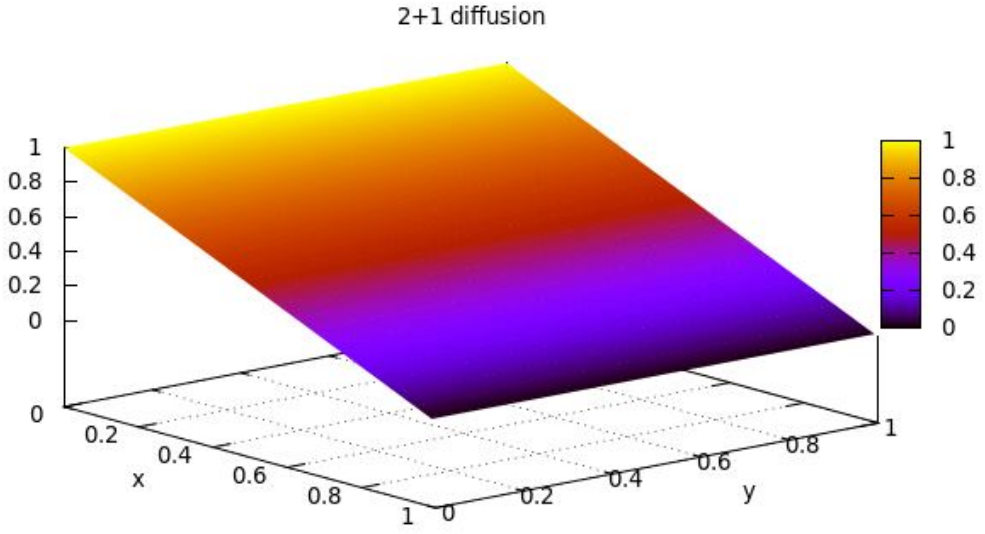

FIGURE 3.15: The closed-form solutions for $t\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 1$, $\Delta t = 0.01$, number of points for t $n = 101$, $\alpha = 100$. The plot corresponds to the moment of time $t = t\_final$.

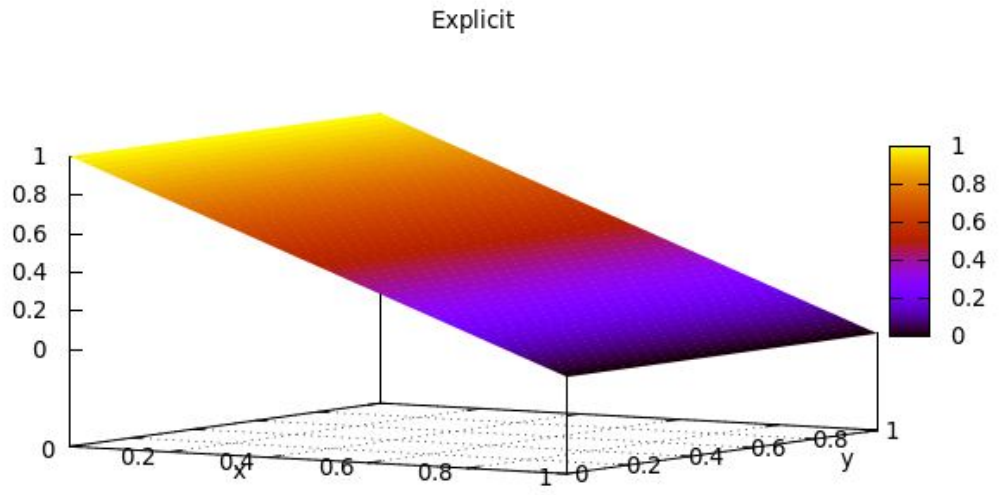

FIGURE 3.16: The solutions for the explicit scheme for $t\_final = 1$, $\Delta x = 0.02$, number of points for x $m = 51$, $t\_final = 1$, $\Delta t = 0.0001$, number of points for t $n = 10001$, $\alpha = 0.25$. The plot corresponds to the moment of time $t = t\_final$.

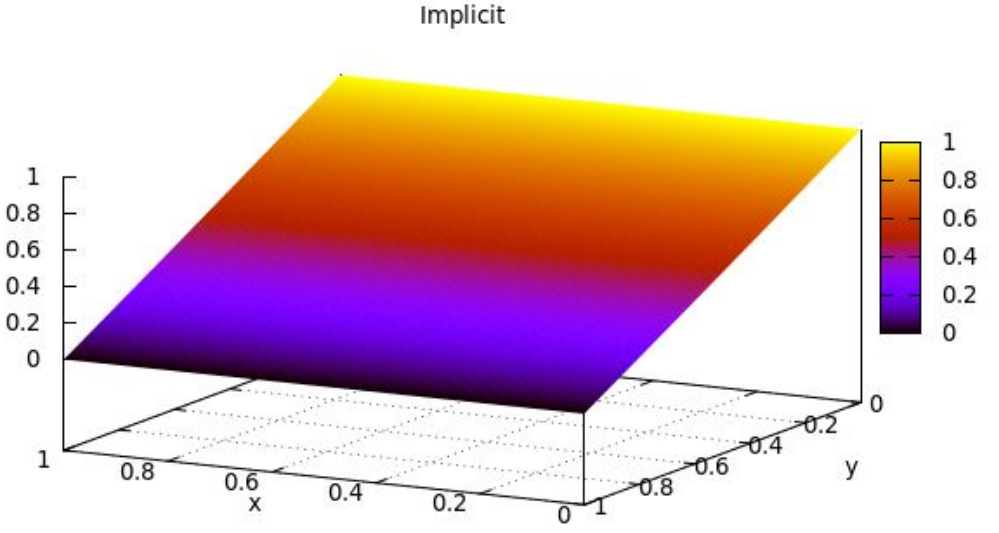

FIGURE 3.17: The solutions for the implicit scheme for $t\_final = 1$, $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 1$, $\Delta t = 0.01$, number of points for t $n = 101$, $\alpha = 100$. The plot corresponds to the moment of time $t = t\_final$.

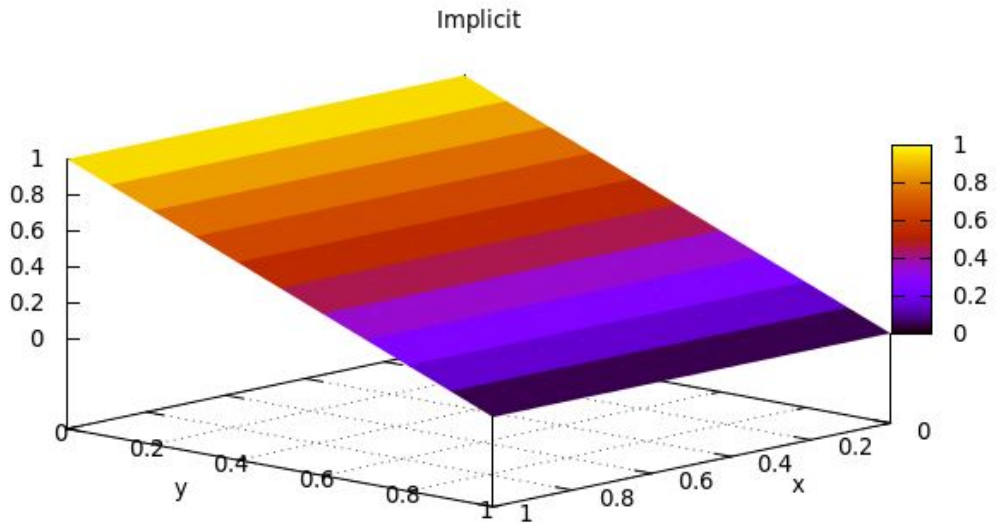

FIGURE 3.18: The solutions for the implicit scheme for $t\_final = 1$, $\Delta x = 0.1$, number of points for x $m = 11$, $t\_final = 1$, $\Delta t = 0.1$, number of points for t $n = 11$, $\alpha = 10$. The plot corresponds to the moment of time $t = t\_final$.

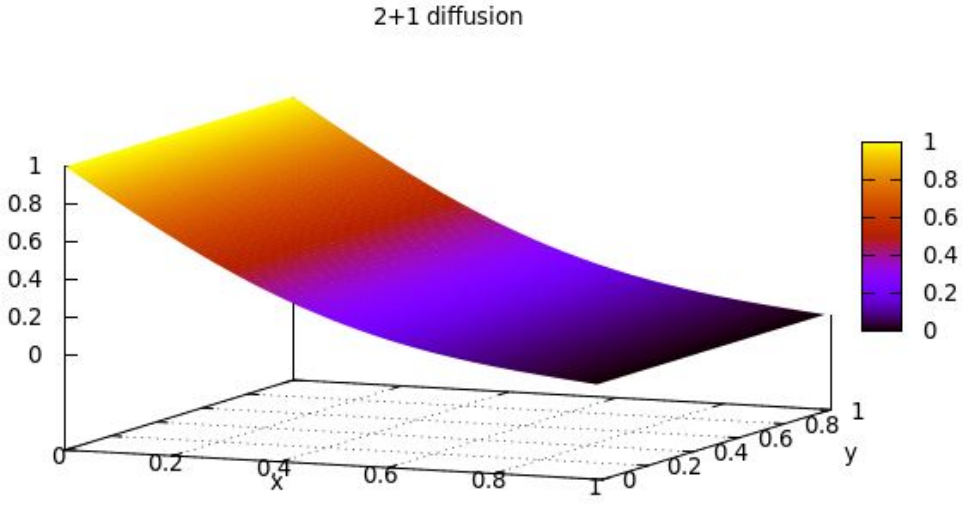

FIGURE 3.19: The close-form solution for $\Delta x = 0.01$, number of points for x $m = 101$, $t\_final = 0.1$, $\Delta t = 0.001$, number of points for t $n = 101$, $\alpha = 10$. The plot corresponds to the moment of time $t = 0.1$.

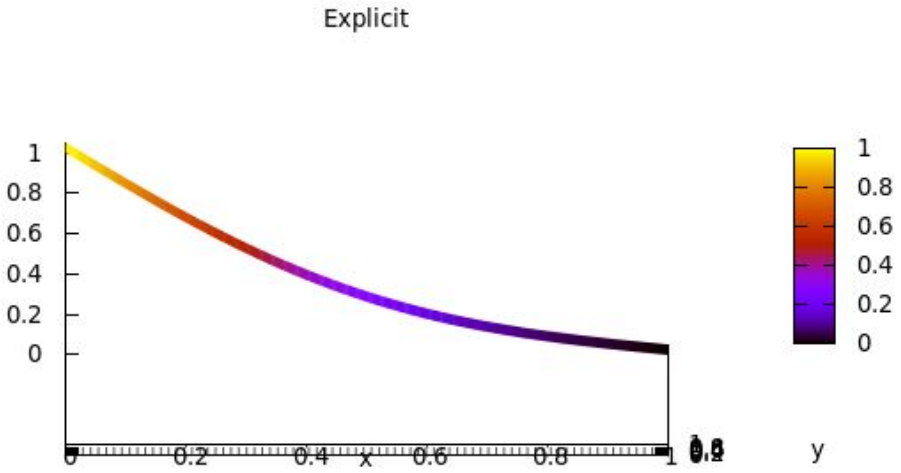

FIGURE 3.20: The solution for explicit scheme for $\Delta x = 0.01$, number of points for x $m = 51$, $t\_final = 0.1$, $\Delta t = 0.00001$, number of points for t $n = 10001$, $\alpha = 0.025$. The plot corresponds to the moment of time $t = 0.1$.

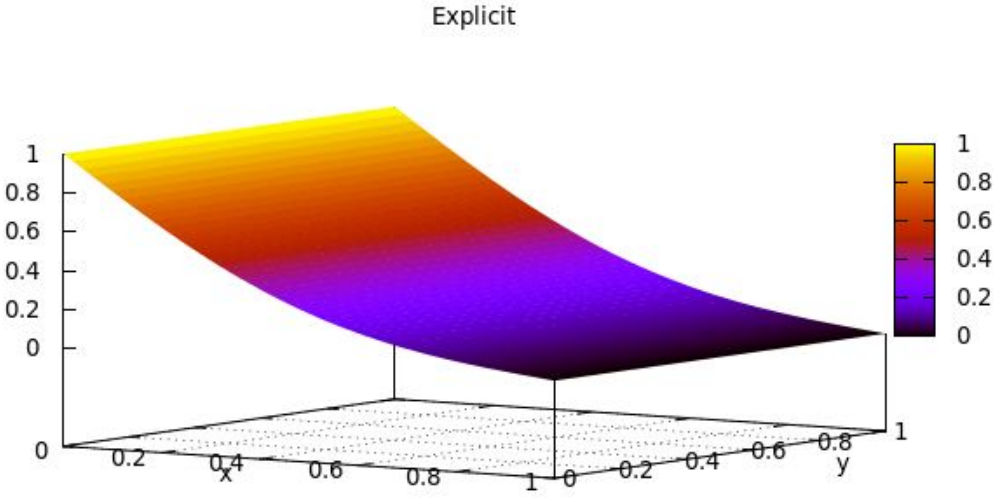

FIGURE 3.21: The solution for explicit scheme (another view) for $\Delta x = 0.01$, number of points for x $m = 51$, $t\_final = 0.1$, $\Delta t = 0.00001$, number of points for t $n = 10001$, $\alpha = 0.025$. The plot corresponds to the moment of time $t = 0.1$.

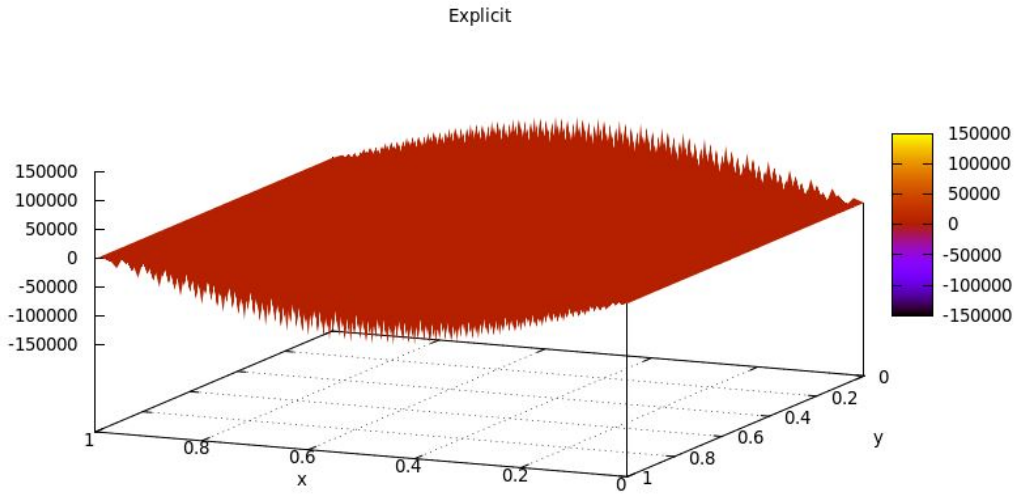

FIGURE 3.22: The solution for explicit scheme for $\Delta x = 0.02$, number of points for x $m = 51$, $t\_final = 1$, $\Delta t = 0.0001003$, number of points for t $n = 9970$, $\alpha = 0.2507$. The plot corresponds to the moment of time $t = 1$.

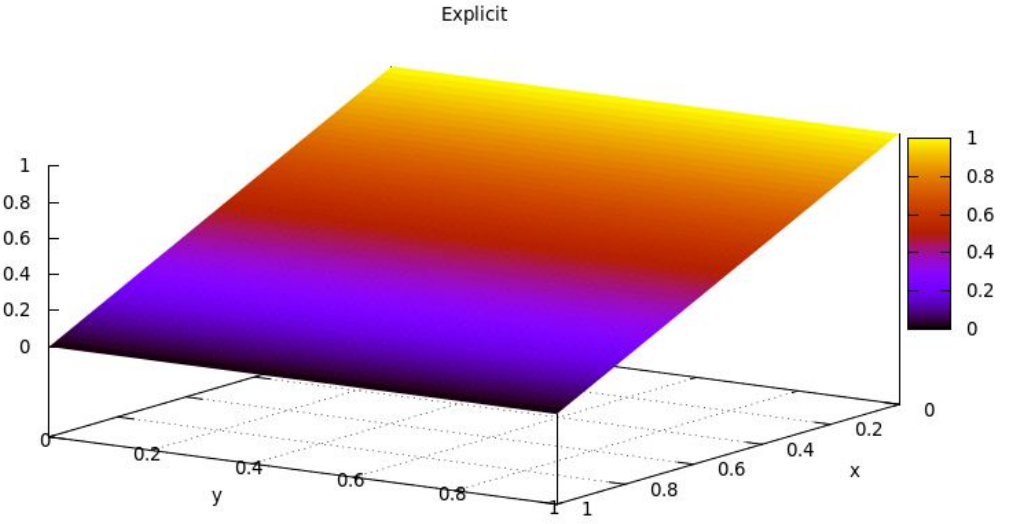

FIGURE 3.23: The solution for explicit scheme for $\Delta x = 0.02$, number of points for x $m = 51$, $t\_final = 1$, $\Delta t = 0.000099$, number of points for t $n = 10002$, $\alpha = 0.24997$. The plot corresponds to the moment of time $t = 1$.

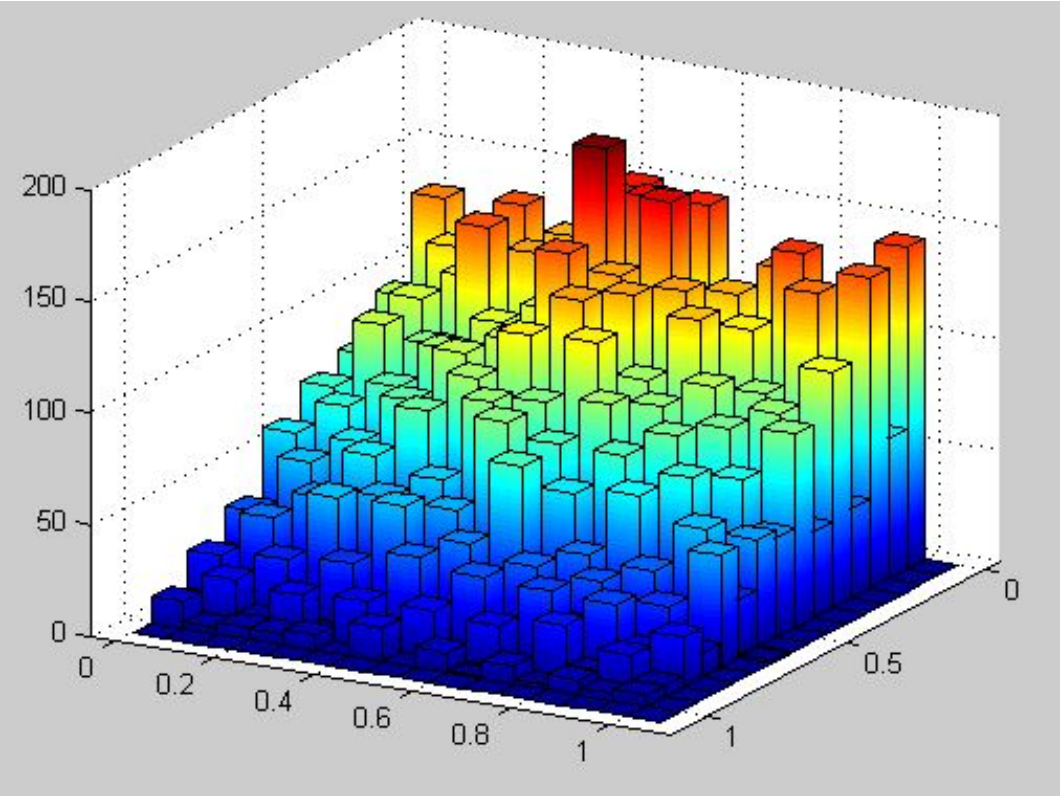

FIGURE 3.24: The solution for random walks for $t\_final = 1$, $\Delta t = 0.002$, number of points for t $n = 501$, step length $l_o = 0.063$, the number of bins $m = 15$, the number of particles at the beginning $N = 100$. The plotter calculates itself the number of walkers in a bin.

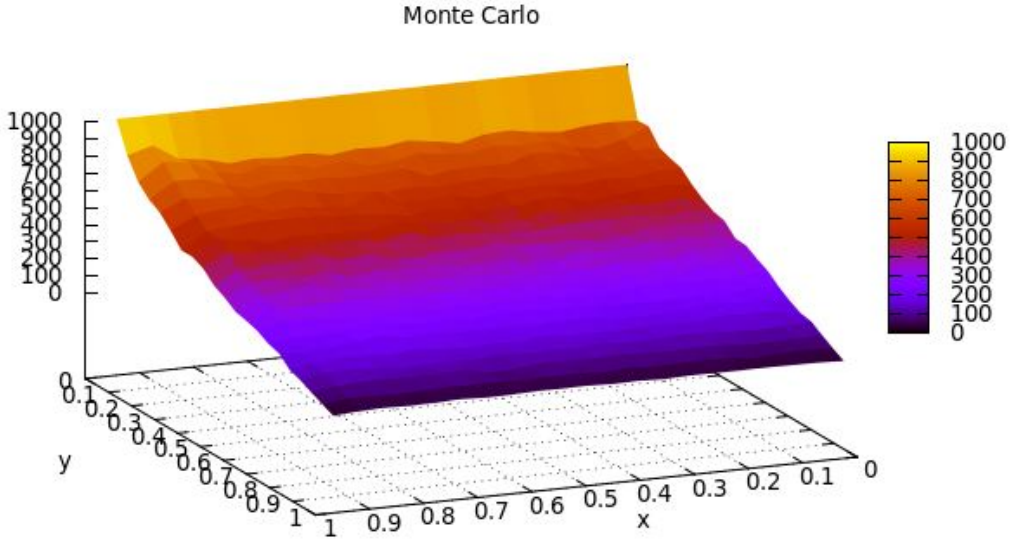

FIGURE 3.25: The solution for random walks for $t\_final = 1$, $\Delta t = 0.00125$, number of points for t $n = 801$, step length $l_o = 0.05$, the number of bins $m = 19$, the number of particles at the beginning $N = 1000$. The number of particles in each box is calculated inside the solver.

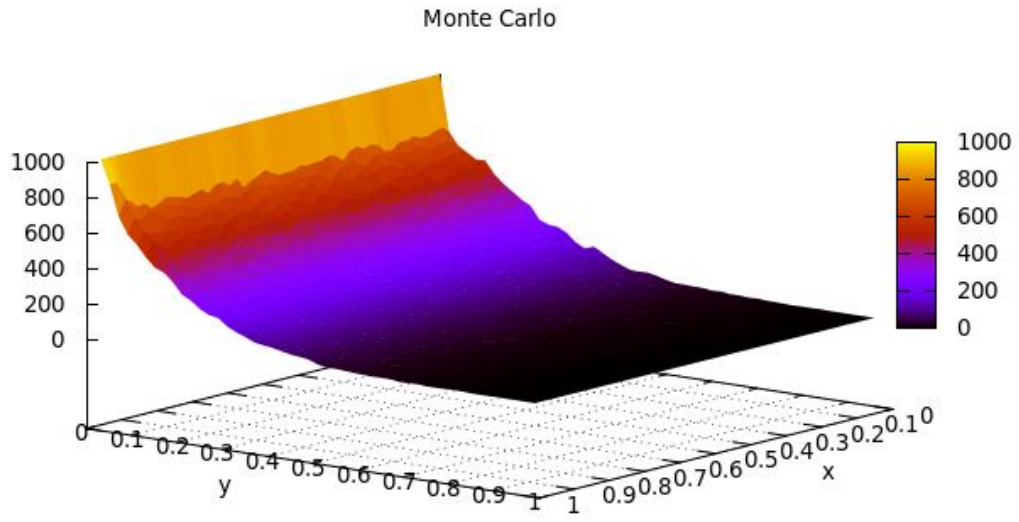

FIGURE 3.26: The solution for random walks for $t\_final = 0.1$, $\Delta t = 0.00125$, number of points for t $n = 801$, step length $l_o = 0.05$, the number of bins $m = 19$, the number of particles at the beginning $N = 1000$. The number of particles in each box is calculated inside the solver.

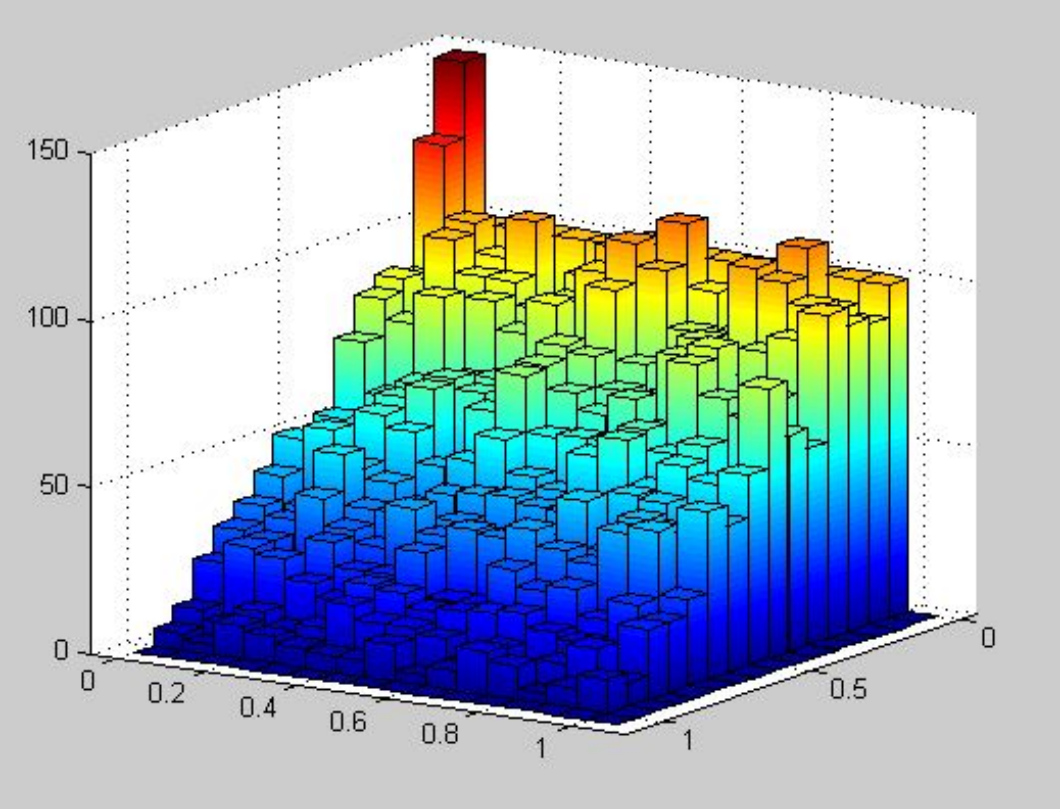

FIGURE 3.27: The solution for random walks for $t\_final = 1$, $\Delta t = 0.002$, number of points for t $n = 501$, step length $l_o = 0.063$, the number of bins $m = 15$, the number of particles at the beginning $N = 100$. The number of particles in each box is calculated inside the solver.

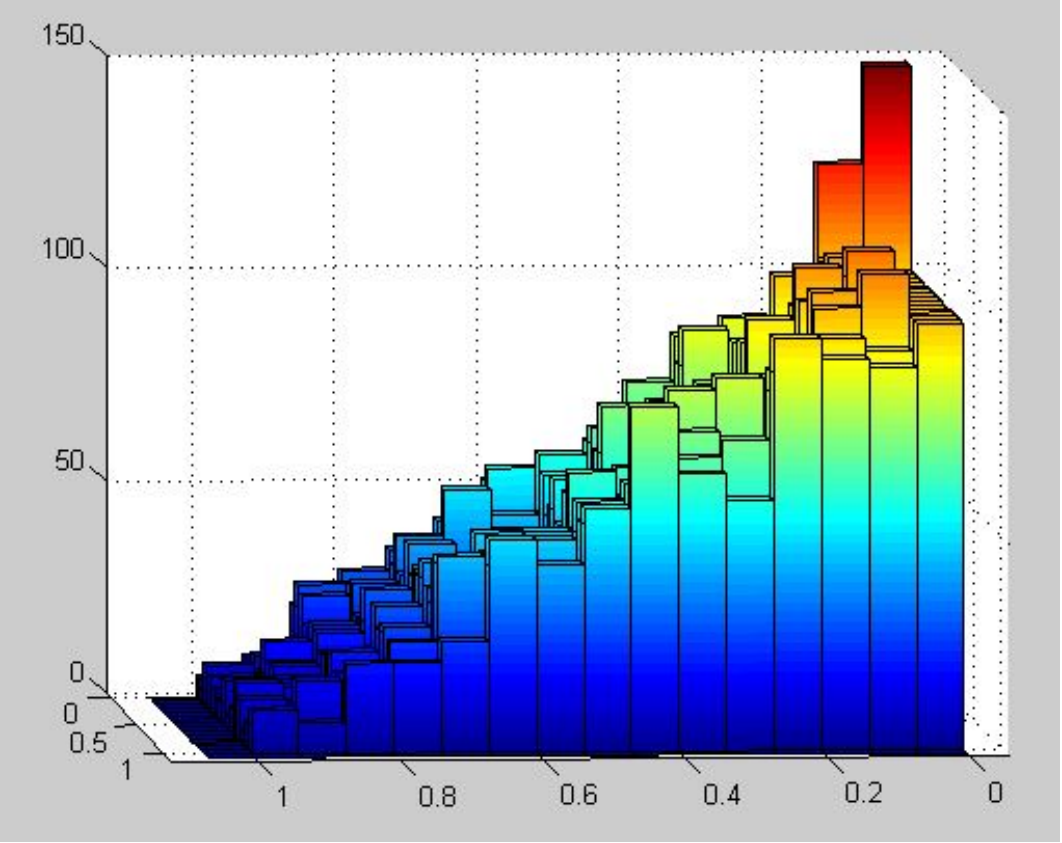

FIGURE 3.28: The solution for random walks (another view) for $t\_final = 1$, $\Delta t = 0.002$, number of points for t $n = 501$, step length $l_o = 0.063$, the number of bins $m = 15$, the number of particles at the beginning $N = 100$. The number of particles in each box is calculated inside the solver.

Bibliography

[1] Partial differential equation, Nov 2014. URL http://en.wikipedia.org/wiki/Partial_differential_equation.

[2] Morten Hjorth-Jensen. *Computational Physics*. University of Oslo, 2014.

[3] Finite difference method, Nov 2014. URL http://en.wikipedia.org/wiki/Finite_difference_method.

[4] Separation of variables, Nov 2014. URL http://en.wikipedia.org/wiki/Separation_of_variables.

[5] Chapter 7. the diffusion equation, Nov 2014. URL http://pauli.uni-muenster.de/tp/fileadmin/lehre/NumMethoden/WS0910/ScriptPDE/Heat.pdf.

[6] WILLIAM J. MOROKOFF and RUSSEL E. CAFLISCH. A quasi-monte carlo approach to particle simulation of the heat equation. *SIAM J. NUMER. ANAL.*, Vol. 30(No. 6):1558–1573, 1993.

[7] Diffusion via monte carlo, Dec 2014. URL http://msg.byu.edu/430/mc_lab.pdf.

[8] Steven R. Cranmer. Monte carlo solutions to diffusion-like equations: A practical application of the ito calculus. *Harvard-Smithsonian Center for Astrophysics, Cambridge, MA 02138 . . . . . Draft notes*, 2003.

[9] Nam Zin Cho. Particle transport monte carlo method for heat conduction problems. *Heat Conduction – Basic Research*, 2003.

[10] Gregory F. Lawler. *Random Walk and the Heat Equation*. Department of Mathematics, University of Chicago, Chicago, IL 60637, 2014.

[11] Chris H. Rycroft (and Martin Z. Bazant). *Lecture 1: Introduction to Random Walks and Diffusion*. Department of Mathematics, MIT, 2005.