Report for Exercise 4&5

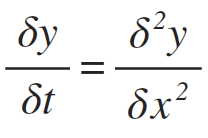
1. Background for the method which is called the finite differences for the PDE

The definition for the finite difference:

In mathematics, finite-difference methods (FDM) are numerical methods for solving differential equations by approximating them with difference equations, in which finite differences approximate the derivatives. FDMs are thus discretization methods. FDMs convert a linear (non-linear) ODE/PDE into a system of linear (non-linear) equations, which can then be solved by matrix algebra techniques. The reduction of the differential equation to a system of algebraic equations makes the problem of finding the solution to a given ODE ideally suited to modern computers, hence the widespread use of FDMs in modern numerical analysis.

In general, the finite difference is the method which using the approximation to get the result of derivatives. Due the method is an approximation, the error from the approximation to the actual results depends on the interval of the arguments,

1. The problem of Exercise 4:

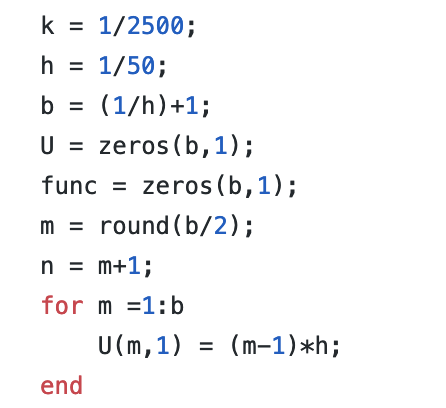
In this exercise we are asked to solve the 1-D heated equation which could be written as the  which the condition give us is that zero boundary conditions y(0, t) = y(1, t) = 0, and initial condition y(x, 0) = y0(x)

//the mathematical proof for the Finite difference

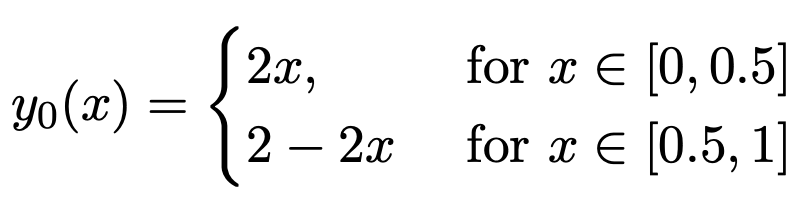
1. The implement for the finite difference by code:

First of all，we set the value of the intervals for two different arguments. In the code we set N to be the segments of the equal length of x, so h=1/N. Similarly, we set the k to the number of segments of the equal length of t, which 1/k would be the small increasement for the argument t.

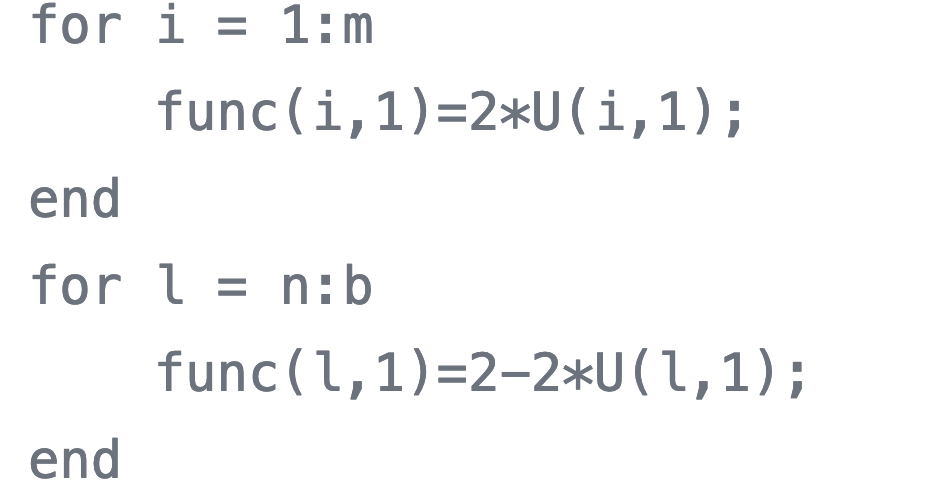
We should create the vector which is also the (N+1 x 1) matrix of the argument of x and we applied the test function to set the initial condition for the y0 which the value at x is=0. The vector called U is the set of the x



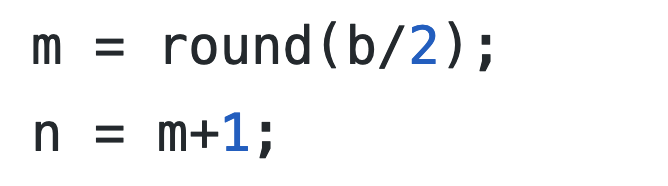
The vector called func is the set of y; which we set the value of y according to the test function, for instance when the test function is



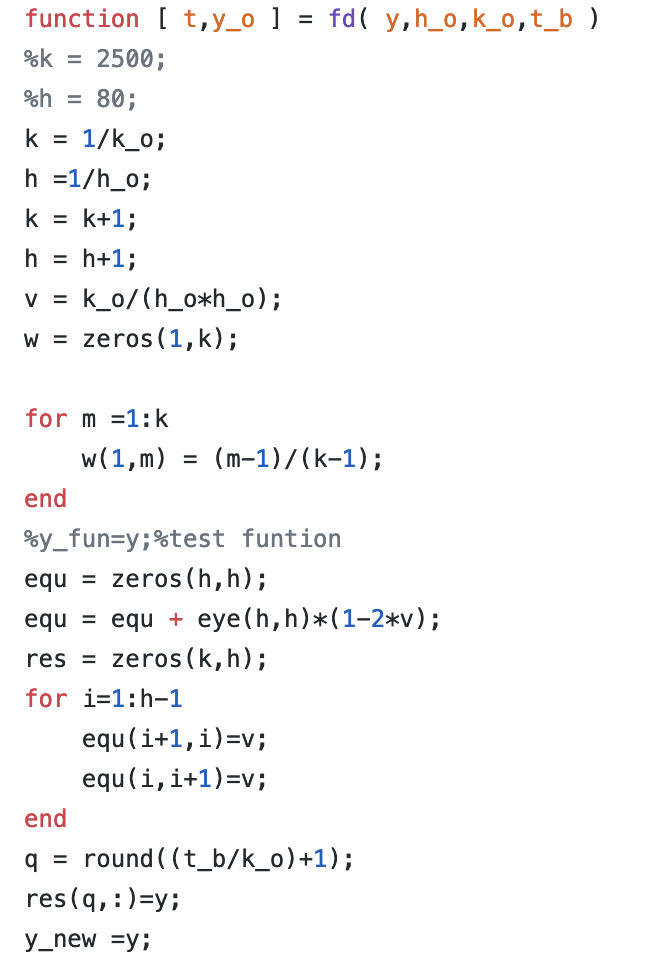
We could the for loop to set the value of y:



Which m and n are:

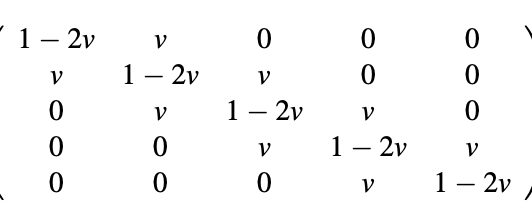


Then we apply the method we have proved pervious:



In the finite difference method, we get the value of y for each x, and each y is depending on the previous y. In order to get the value of they, we also need the matrix to multiply the previous vector y, in the code we call the matrix equ.

The matrix has a dimensional of (h x h). the main goals of setting the matrix to be in this form:

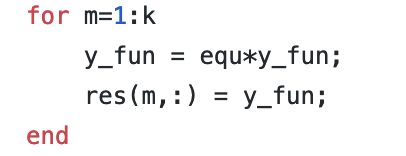


First, we set the matrix to be the diagonal matrix and then we multiply the matrix by the value of 1-2v which v is equal to k/h\*h. One of the most significant things is that we need to mention here is that the value of v should be less than 0.5 which could keep the tool matrix converge, if the tool matrix diverges, the multiplication for the y value would become diverge, which means the elements in the matrix would become infinite.The next step is to set the elements next to diagonal to be v; the for loop is used to set each corresponding position.

The equ matrix remains constant during the calculating process, and we could use the for loop to calculate the value of y with the corresponding x. We now define the equ function as the tool function.

For each iteration what we do is put the vector results into a new matrix which stores the corresponding results for the calculation and the first column would be the initial condition for y.

The code is shown below:

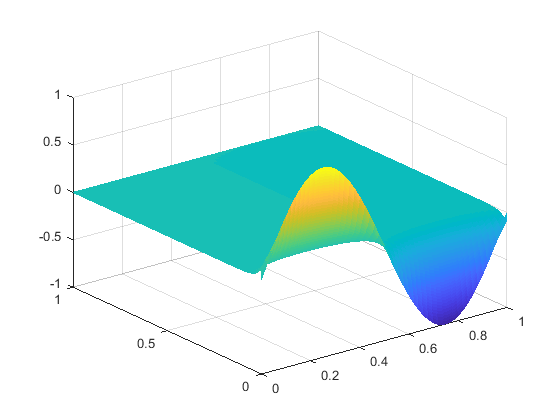


And eventually, we get the results for the Y and we plot the 3d graph for the y.

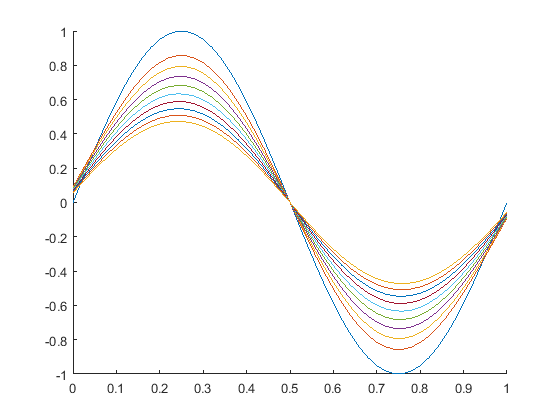
4.The results of testing:

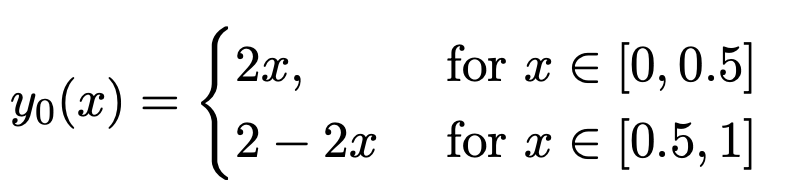
As required, we set different test function to test cases

first y=sin(2\*pi\*x), for k=1/10000, h=1/50 and the 3Dgraph is shown:

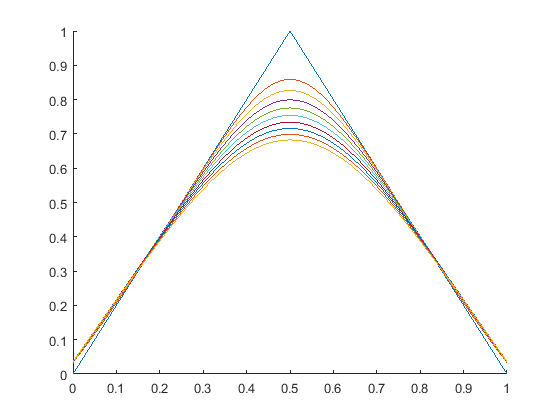
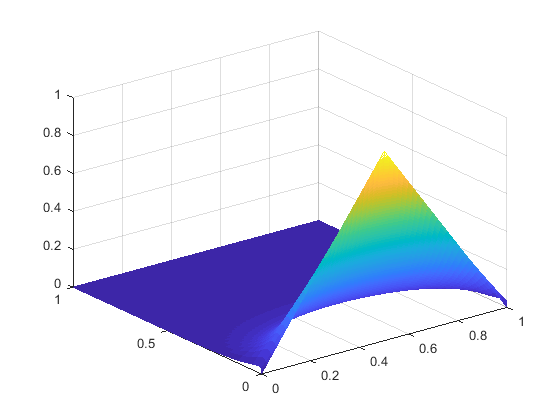


and then we take the cross-section of the graph:

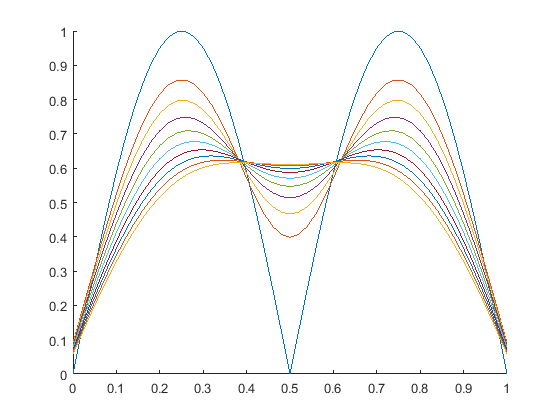
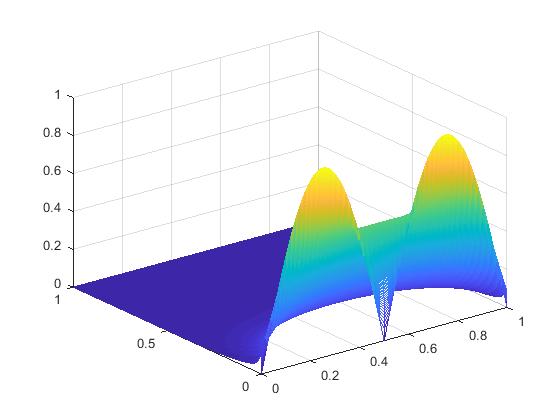


second the test function has been set with 

here are the 3D graph and cross-section graph:



the last test function is the absolute value for sine function:

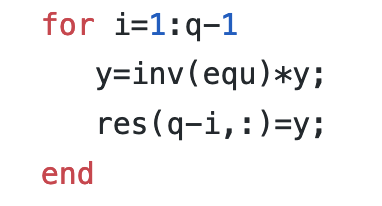


1. The optimization for the code(ex5)

In the exercise5, we have to transfer the code into more general case, which means that we need to set other value instead of 0 to the zeros condition, however, we have already set the vector of the input function to be the initial value of the function, which has the code here:



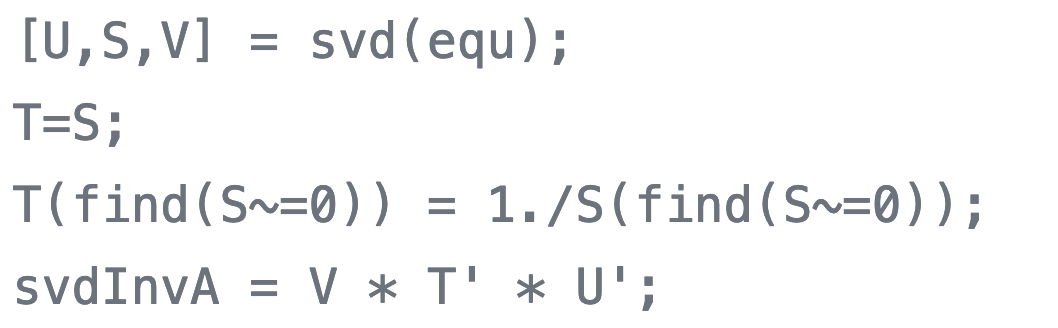
The second requirement of the exercise is changing the time varies, which means we need to change the index of the x in a different value.

In order to implement the requirement, we need to use the inverse matrix to calculate the previous value of y, the code has been shown, which we use inv function in MATLAB to get the inverse matrix of the tool matrix:

There are some other ways to get the inverse matrix of tools matrix, one common way is to use the SVD method:

The SVD method is commonly used in machine learning nowadays. The singular value decomposition as known as SVD, in linear algebra, the singular-value decomposition (SVD) is a factorization of a real or complex matrix. It is the generalization of the eigendecomposition of a positive semidefinite normal matrix

And the code has become this:



In the code, svdInvA become the inverse of the tool matrix, which could increase the stability of the algorithm to calculate the inverse matrix. However, in results, it did not make any difference.

However when we were trying to using a large t\_b in the program the code could not generate the graph with the correct form, which was caused by the inverse of the tools matrix, when we were doing the matrix calculation of the backpropagation of the previous value of y , the value of y grows to become larger and larger and eventually to be the infinite. Cause that the multiplication between inverse tool matrix and the y vector become diverge .As the results, we need to keep a small v which means we need to take as many intervals of argument x as possible.