Report for Exercise 4&5

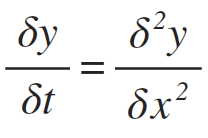
1. Background for the method which is called the finite differences for the PDE

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

Firstly, we set the interval of argument of x to be h, and total N segments which means N = 1/h, as the results we could easily generate the results which

X0 = 0, x1=h, x2= 2h, xn=1;

Alternatively, if we do the same thing with the argument of t, discretize time into equal segments of length k:

T0= x, t1=k ,t2 = 2k

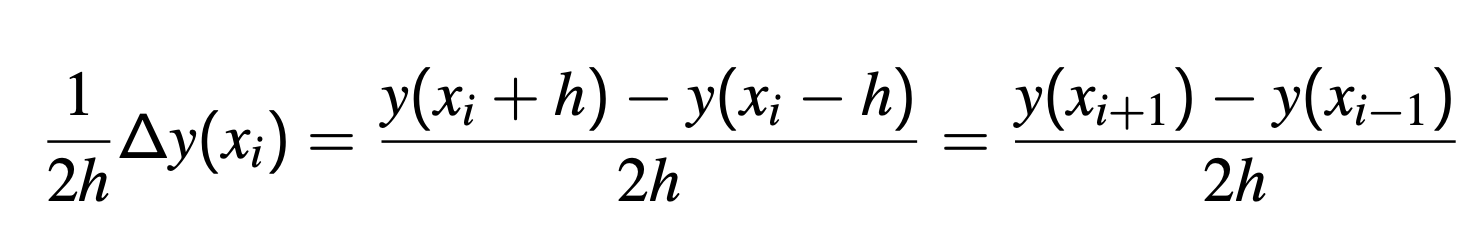
Now we need go back to the pervious knowledge about the finite difference for ODE,

From the first year, as the definition of derivative:

图片包含 物体, 时钟

描述已自动生成

When we drive both sides by 2h, the expression would become:



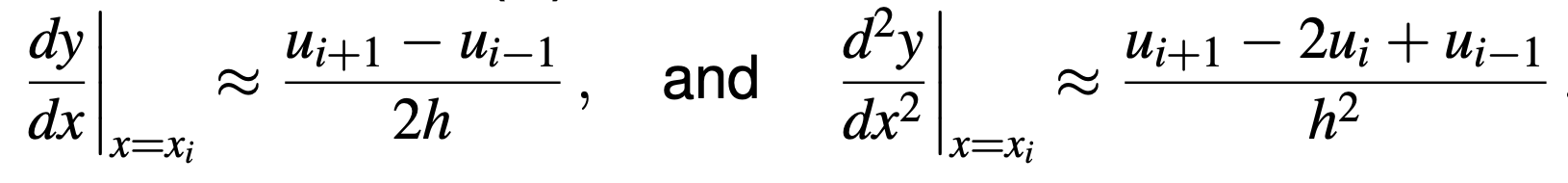
As we defined,

图片包含 物体

描述已自动生成 for j from 0 to N

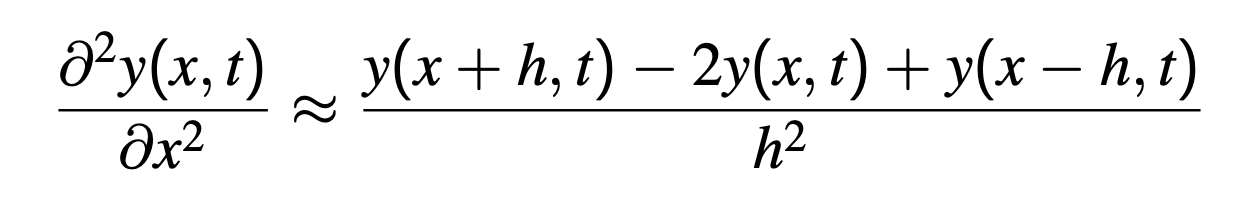
Obviously, 图片包含 物体

描述已自动生成, so the pervious expression would become



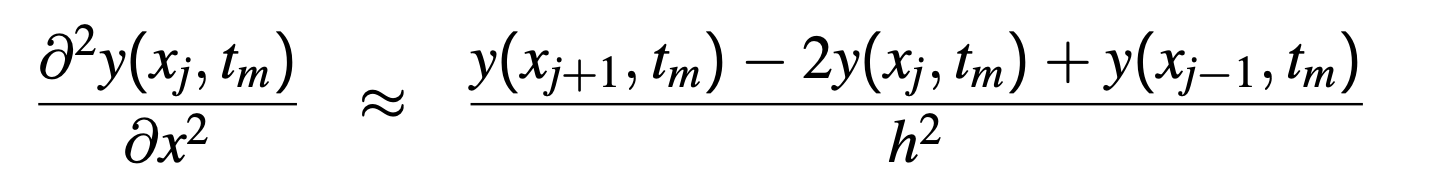
As we set ui equal to yi.

Up to now, we have gotten the expression of the second order derivatives, eventually, we could write the expression for the partial derivatives which is:



As the result, when we set the x = xj, and t= tm, we could drive that x+h = xj+1

x-h = xj-1; the y in expression could be substitute into:



Which is 图片包含 物体

描述已自动生成

Similarly, we could use the same principle to derive the expression of left-hand side of the heat function which is

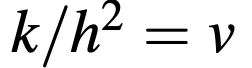
图片包含 物体

描述已自动生成

Finally, when we get this two expression for both left hand side and right hand side, we could easily write the equation ：

图片包含 物体

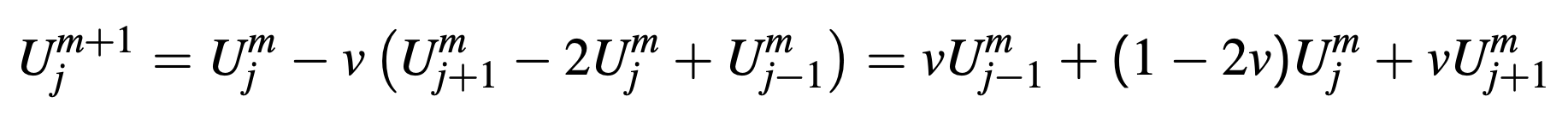
描述已自动生成

Cause the value of h and k are constant, so we could set the ,and the equation could be rearranged into

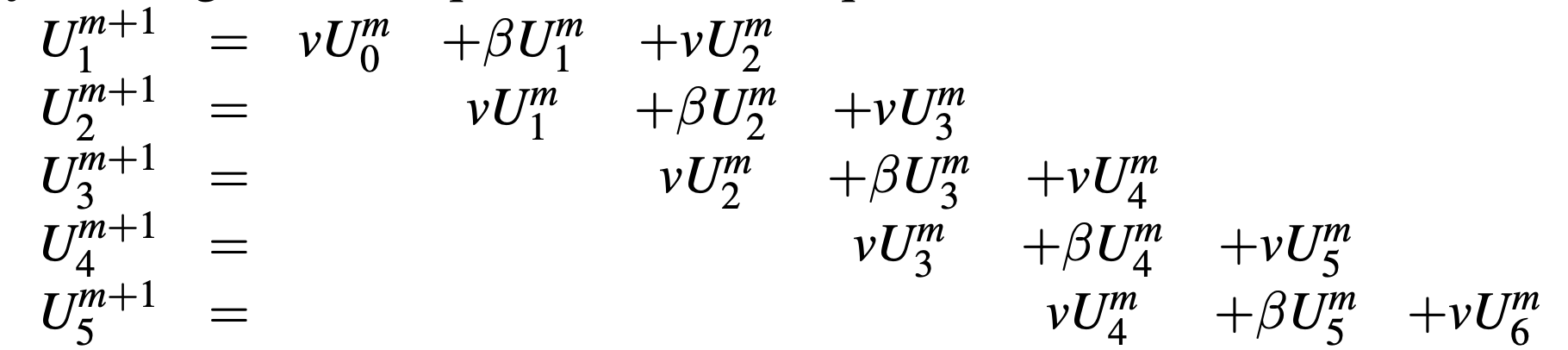
图片包含 物体

描述已自动生成

The equation could be written into another form which is



So we could set the function to be matrix when we set the N=6, the whole set of 5 function could be written into



Where 图片包含 家具

描述已自动生成.

Now the matrix equation could be written into:

图片包含 物体, 时钟, 天空

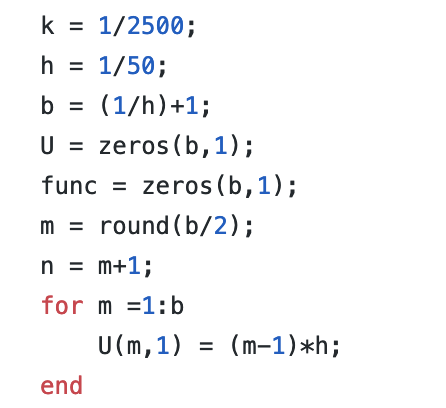
描述已自动生成

Where we could get the next value of y from the previous value

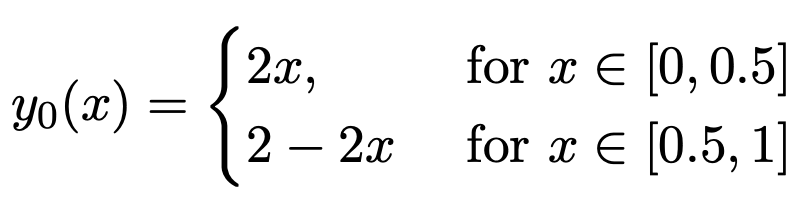
1. The implement for the finite difference by code:

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.

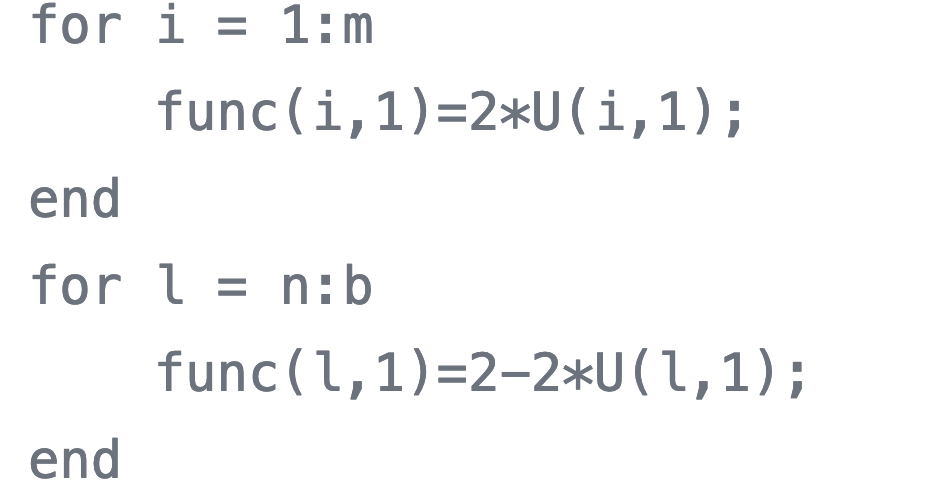
We should create the vector which is also the (N+1 x 1) matrix of the argument of x and we applied the test function to set the initial condition for the y0 which the value at x is=0. The vector called U is the set of the x



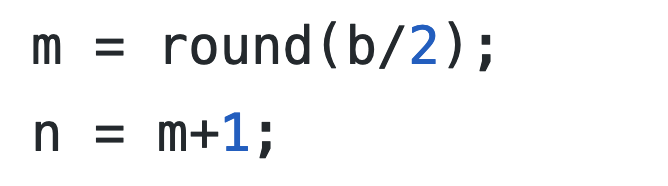
The vector called func is the set of y; which we set the value of y according to the test function, for instance when the test function is



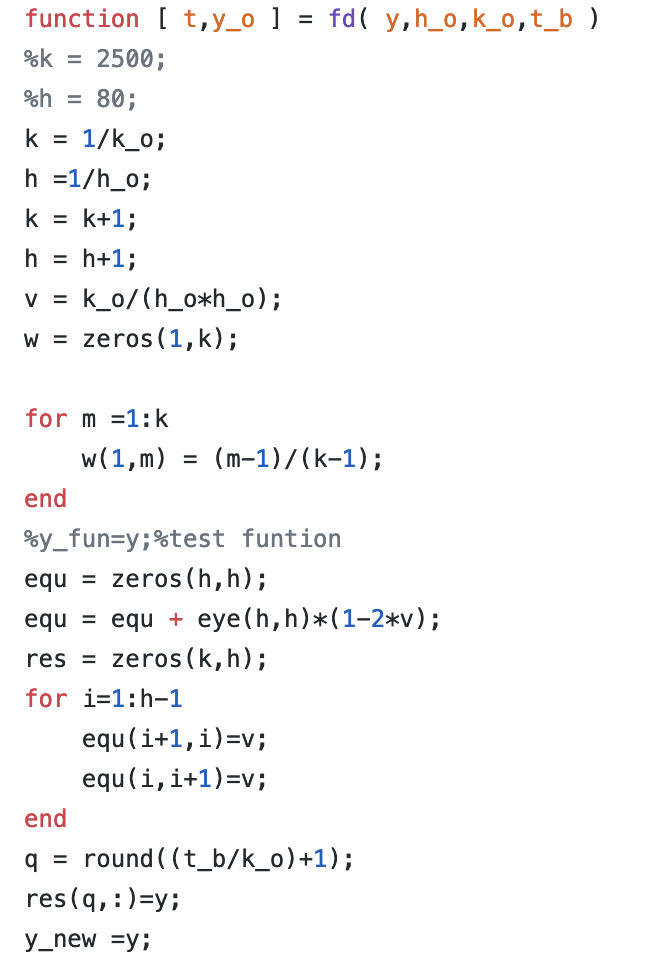
We could the for loop to set the value of y:



Which m and n are:

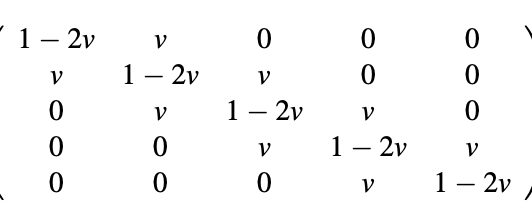


Then we apply the method we have proved pervious:



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 vectory, 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:

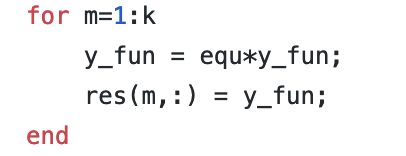


First, we set the matrix to be the diagonal matrix and then we multiply the matrix by the value of 1-2v which v is equal to k/h\*h. One of the most significant things is that we need to mention here is that the value of v should be less than 0.5 which could keep the tool matrix converge, if the tool 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 that the equ matrix 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:

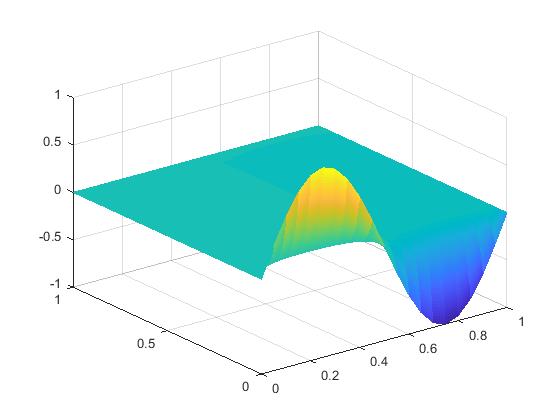


And eventually, we get the results for the Y and we plot the 3d graph for the y.

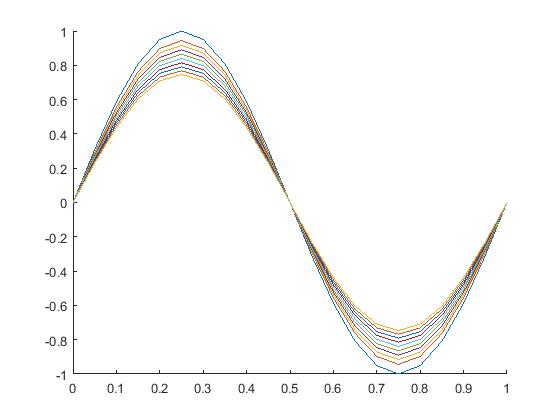
4.The results of testing:

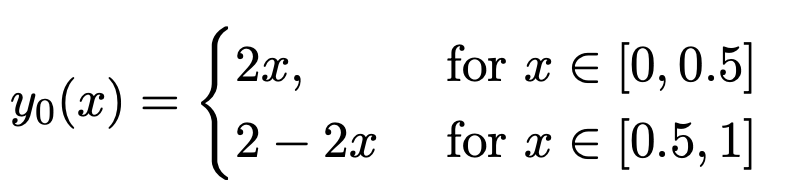
As required, we set different test function to test cases

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

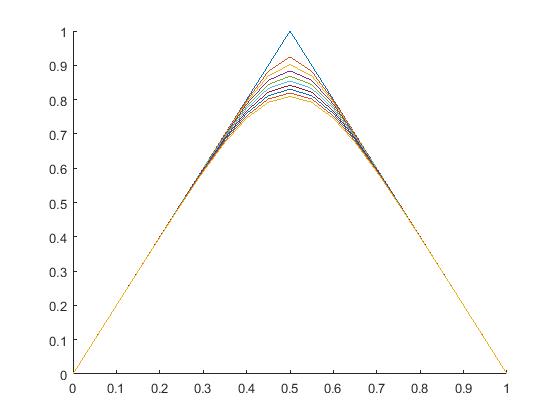
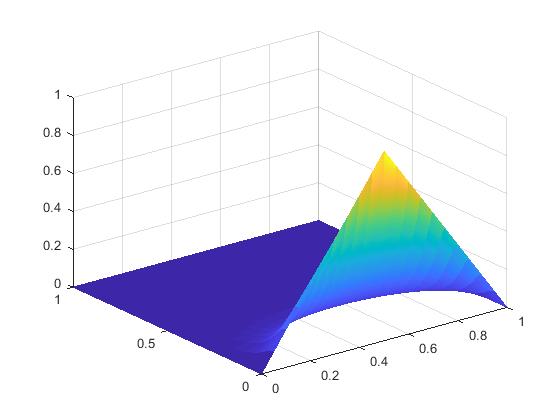


and then we take the cross-section of the graph:

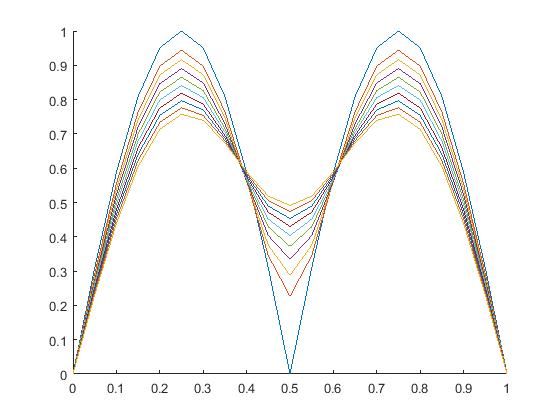
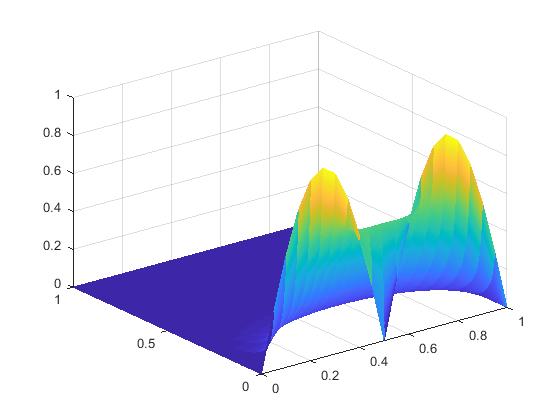


second the test function has been set with 

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

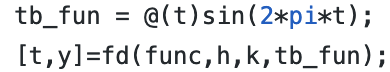


the last test function is the absolute value for sine function:

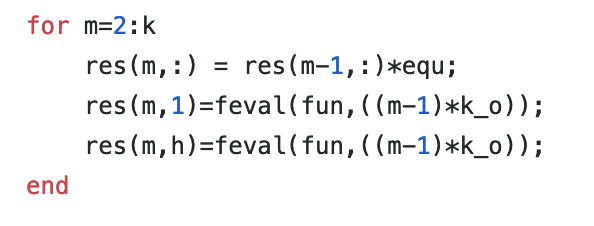


1. The optimization for the code(ex5)

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, one of the important things need to be mentioned is that the two-boundary conditions should keep same. As the results, we changed the code in the MATLAB into below:

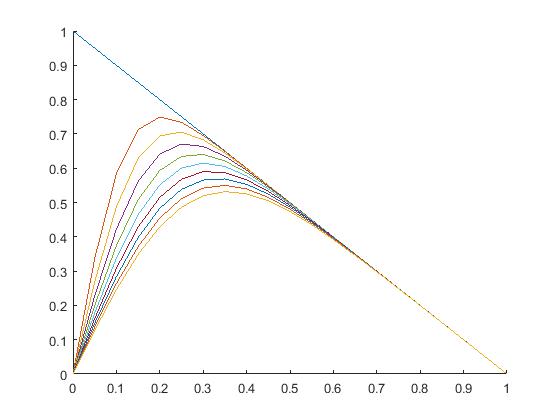
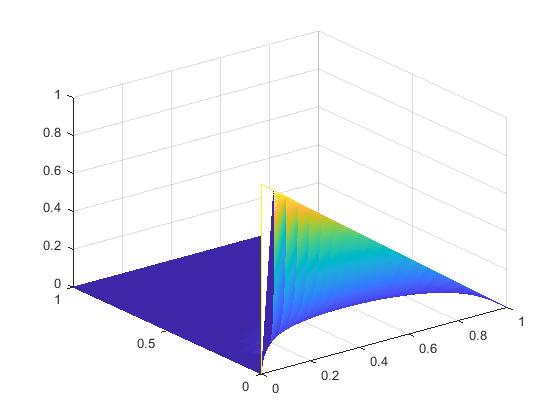


Which tb\_fun is the function for the boundary conditions, and we set as the variable as the variable in the fd function, also in the function file we should change the for loop of setting the value for the first and last elements into the value of the boundary conditions in each column



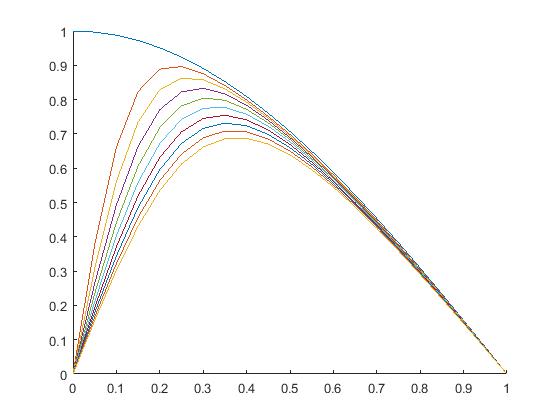
And the first test function in this exercise is 

The results of the 3D plot graph and cross-sectional graph shown:



And the other test function is y0= cos(pi\*x\*0.5), the result has shown: 图片包含 配件, 雨伞

描述已自动生成

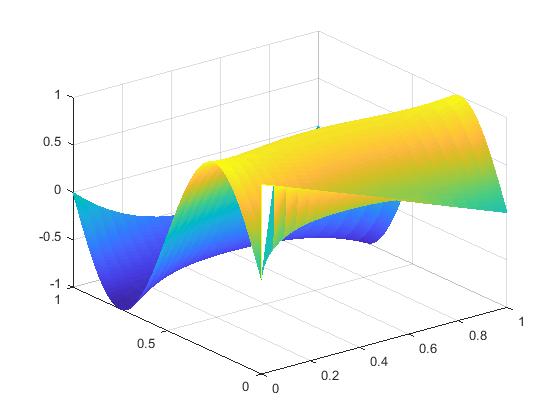


As the conclusion, the boundary conditions are different, however, we set both of them to be zero.

Alternatively, we could also change the boundary conditions into the sine wave function or any other function which ensures the value at x=0 equals to the value at x=1;

For instance, we could set the boundary into the sin(2\*pi\*t), which match the conditions that y=0 at x=1 and x=0.

As the result, the graph shown:



The result is satisfied with the expectation.