On Approximating Solutions of One-Dimensional Boundary Value Problems by Using Finite Difference Method on Nonuniform Mesh

Nguyen Quan Ba Hong*
Doan Tran Nguyen Tung†
Nguyen An Thinh‡
Students at Faculty of Math and Computer Science
Ho Chi Minh University of Science, Vietnam
email. nguyenquanbahong@gmail.com
email. dtrngtung@live.com
email. anthinh297@gmail.com
blog. http://hongnguyenquanba.wordpress.com §

February 4, 2017

Abstract

This assignment aims at solving one-dimensional boundary value problem with Dirichlet boundary conditions numerically by using finite difference method on both uniform and nonuniform grid. We also implement boundary value problem with Dirichlet-Neumann boundary condition by Matlab scripts at the end of this context.

*Student ID. 1411103
†Student ID. 1411352
‡Student ID. 1411289

Contents

List of Figures

1 Problems

Problem 1.1. *Let* $\Omega = (0,1) \subset \mathbb{R}$ *and* $f \in L^2(\Omega)$.

$$u_{xx} = f(x) \text{ in } \Omega \tag{1.1}$$

subject to a Dirichlet boundary condition

$$u(0) = u(1) = 0 \tag{1.2}$$

1. *How to discretize the previous system equation by finite difference method with non-uniform mesh* $\{x_i\}_{i=0}^N$ *with*

$$0 = x_0 < x_1 < \cdots < x_{N-1} < x_N = 1 \tag{1.3}$$

2. *Demonstrate the convergence of the discrete solution to exact solution.*

Remark 1.2. We note that the sign of the left-hand side of (1.1) is not important. We can take $-f(x)$ in the right-hand side instead. Thus, we can assume that there is no minus in both sides and focus to solve (1.1) numerically.

With a given non-uniform mesh $\{x_i\}_{i=0}^N$, we attempt to compute a grid function consisting of values $U_0, U_1, \ldots, U_N, U_{N+1}$, where U_i is an approximation to the solution $u(x_i)$. We put

$$h_i = x_i - x_{i-1}, \;\; i = 1, 2, \ldots, N+1 \tag{1.4}$$

are the distances between consecutive points in the mesh.

We also define the mesh width as

$$H = \max_{1 \le i \le N+1} h_i \tag{1.5}$$

to analyze the error later.

From the boundary conditions (1.2) we know that

$$U_0 = 0, U_{N+1} = 0 \tag{1.6}$$

and so we have m unknown values $U_1, \ldots, U_N$ to compute.

2 First Approach

We emphasize that there is an alternative approach to (1.1) with different finite difference formula at the section 4. We consider the following approach first because it is simply and the grid mapping function used right now is a nice idea to discover in this topic.

2.1 Deriving A Finite Difference Approximations

Firstly, we emphasize that we are dealing with *nonuniform mesh* (1.3). We must specify (1.3) in some particular ways so that our work on this way gets easier, especially the convergence part later.

One way to specify nonuniform grid points is to start with a uniform grid. We describe this specification by the following process.

NONUNIFORM GRIDS. At the beginning, we choose an arbitrary monotonically increasing $X \in C([-\varepsilon, 1+\varepsilon])$, for a chosen real number $\varepsilon > 0$, for which $X(0) = 0, X(1) = 1$. Then we fix X from here to the end of this context. It is easily to realize that under these conditions, X is a bijection on $[0, 1]$. We call X the *grid mapping* function. As you see right now, these properties of this grid mapping function X help us transform an uniform grid to nonuniform grid.

Remark 2.1. We consider the grid mapping function X on the interval $[-\varepsilon, 1+\varepsilon]$ instead of $[0, 1]$ so that $X'(0), X'(1)$ are still exist in (2.5).

For each positive integer N, we have the uniform grid $\{z_i\}_{i=0}^{N+1}$ associated with N, which is defined by

$$z_i = \frac{i}{N+1}, i = 0, 1, \ldots, N+1 \tag{2.1}$$

We use the fixed grid mapping function X to define the physical grid points

$$x_i = X(z_i), \quad 0, 1, \ldots, N+1 \tag{2.2}$$

By properties of X, we immediately deduce that (1.3) holds. Thus, we have just associated $\{z_i\}_{i=0}^{N+1}$ with a nonuniform grid $\{x_i\}_{i=0}^{N+1}$.

It can be noticed that we have fixed function X to define a nonuniform grid $\{z_i\}_{i=0}^{N+1}$ associated with each positive integer N in the above process. This is illustrated in Figure 2.1, where z is plotted on the vertical axis and x is on the horizontal axis. The curve plotted represents a function $X(x)$, although with this choice of axes it is more properly the graph of the inverse function $z = X^{-1}(x)$. The horizontal and vertical lines indicate how the uniform grid points on the z axis are mapped to nonuniform points in x.

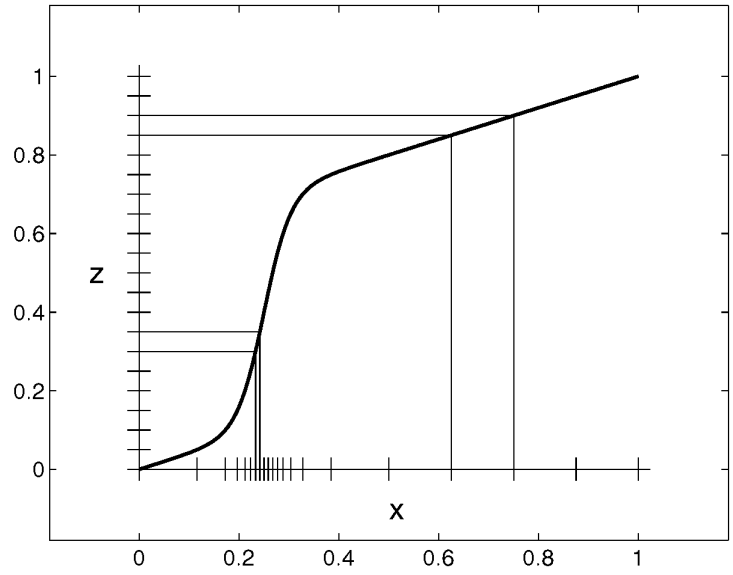

Figure 1: GRID MAPPING FROM A UNIFORM GRID IN $0 \leq z \leq 1$ (VERTICAL AXIS) TO THE NONUNIFORM GRID IN PHYSICAL x-SPACE SHOWN ON THE HORIZONTAL AXIS.

Note that grid points are clustered in regions where the curve is steepest, which means that $X(x)$ varies slowly with z, and spread apart in regions where $X(z)$ varies rapidly with z.

Once a set of grid points x_i is chosen, it is necessary to set up and solve an appropriate system of difference equations on this grid. In general a different set of finite difference coefficients will be required at each grid point, depending on the spacing of the grid points nearby.

Return to our problem, for each $i = 1, 2, \ldots, N+1$, we have

$$\begin{aligned} h_i &= x_i - x_{i-1} && (2.3)\\ &= X(z_i) - X(z_{i-1}) && (2.4)\\ &= hX'(z_{i-1}) + O\left(h^2\right) && (2.5)\\ &= O(h) && (2.6) \end{aligned}$$

as $h \to 0$.

As a consequence, we also have

$$H = \max_{1 \leq i \leq N+1} h_i = O(h) \text{ as } h \to 0 \qquad (2.7)$$

Next, for each $i = 1, 2, \ldots, N$, we have

$$\begin{aligned} h_{i+1} - h_i &= (x_{i+1} - x_i) - (x_i - x_{i-1}) && (2.8)\\ &= x_{i+1} - 2x_i + x_{i-1} && (2.9)\\ &= X(z_{i+1}) - 2X(z_i) + X(z_{i-1}) && (2.10)\\ &= h^2 X''(z_i) + O\left(h^4\right) && (2.11)\\ &= O\left(h^2\right) && (2.12) \end{aligned}$$

as $h \to 0$, where we have used the standard second order centered approximation in the last equality.

We now have enough materials to move forward. Next, we expand $u(x_{i-1})$ and $u(x_{i+1})$ in a Taylor series expansion about the point x_i, e.g.,

$$u(x_{i-1}) = u(x_i) - h_i u'(x_i) + \frac{1}{2}h_i^2 u''(x_i) - \frac{1}{6}h_i^3 u'''(x_i) \tag{2.13}$$

$$+ \frac{1}{24}h_i^4 u^{(4)}(x_i) + O\left(h_i^5\right) \tag{2.14}$$

$$u(x_{i+1}) = u(x_i) + h_{i+1} u'(x_i) + \frac{1}{2}h_{i+1}^2 u''(x_i) \tag{2.15}$$

$$+ \frac{1}{6}h_{i+1}^3 u'''(x_i) + \frac{1}{24}h_{i+1}^4 u^{(4)}(x_i) + O\left(h_{i+1}^5\right) \tag{2.16}$$

Remark 2.2. We can expand $u(x_{i-1})$ and $u(x_i)$ in Taylor series expansion with lower or higher order, for instance, $O\left(h^4\right)$ will works well.

Then we have

$$h_{i+1}u(x_{i-1}) + h_i u(x_{i+1}) \tag{2.17}$$

$$= (h_i + h_{i+1})\, u(x_i) + \frac{1}{2}h_i h_{i+1}(h_i + h_{i+1})\, u''(x_i) \tag{2.18}$$

$$+ \frac{1}{6}h_i h_{i+1}\left(h_{i+1}^2 - h_i^2\right) u'''(x_i) + \frac{1}{24}h_i h_{i+1}\left(h_i^3 + h_{i+1}^3\right) u^{(4)}(x_i) \tag{2.19}$$

$$+ h_{i+1}O\left(h_i^5\right) + h_i O\left(h_{i+1}^5\right) \tag{2.20}$$

equivalently,

$$u''(x_i) = \frac{2\left[h_{i+1}u(x_{i-1}) + h_i u(x_{i+1}) - (h_i + h_{i+1})\, u(x_i)\right]}{h_i h_{i+1}(h_i + h_{i+1})} \tag{2.21}$$

$$- \frac{1}{3}(h_{i+1} - h_i)\, u'''(x_i) - \frac{1}{12}\frac{h_i^3 + h_{i+1}^3}{h_i + h_{i+1}} u^{(4)}(x_i) \tag{2.22}$$

$$+ \frac{1}{h_i(h_i + h_{i+1})}O\left(h_i^5\right) + \frac{1}{h_{i+1}(h_i + h_{i+1})}O\left(h_{i+1}^5\right) \tag{2.23}$$

Put

$$\bar{D}_2 u(x_i) := \frac{2\left[h_{i+1}u(x_{i-1}) + h_i u(x_{i+1}) - (h_i + h_{i+1})\, u(x_i)\right]}{h_i h_{i+1}(h_i + h_{i+1})} \tag{2.24}$$

we obtain

$$\bar{D}_2 u(x_i) - u''(x_i) = \frac{1}{3}(h_{i+1} - h_i)\, u'''(x_i) + \frac{1}{12}\frac{h_i^3 + h_{i+1}^3}{h_i + h_{i+1}} u^{(4)}(x_i) \tag{2.25}$$

$$+ \frac{1}{h_i(h_i + h_{i+1})}O\left(h_i^5\right) + \frac{1}{h_{i+1}(h_i + h_{i+1})}O\left(h_{i+1}^5\right) \tag{2.26}$$

We now need to deal with each term in the right-hand side of (2.25)-(2.26). Using (2.12), we have

$$\left|\frac{1}{3}(h_{i+1} - h_i)\, u'''(x_i)\right| = O\left(h^2\right) \tag{2.27}$$

$$\left|\frac{1}{12}\frac{h_i^3+h_{i+1}^3}{h_i+h_{i+1}}u^{(4)}(x_i)\right| = \frac{u^{(4)}(x_i)}{12}\left(h_i^2+h_ih_{i+1}+h_{i+1}^2\right) \tag{2.28}$$

$$= O\left(h^2\right) \tag{2.29}$$

as $h \to 0$.

$$\frac{1}{h_i(h_i+h_{i+1})}O\left(h_i^5\right) + \frac{1}{h_{i+1}(h_i+h_{i+1})}O\left(h_{i+1}^5\right) \tag{2.30}$$

$$= O\left(\frac{h_i^4}{h_i+h_{i+1}}\right) + O\left(\frac{h_{i+1}^4}{h_i+h_{i+1}}\right) \tag{2.31}$$

$$= O\left(h_i^3\right) + O\left(h_{i+1}^3\right) \tag{2.32}$$

$$= O\left(h^3\right) \tag{2.33}$$

as $h \to 0$.

Combining (2.25)-(2.33), we deduce that $D^2u(x_i)$ is a second order accurate approximation for $u''(x_i)$.

2.2 Derivation of a System of Equations

If we replace $u''(x)$ in (1.1) by (2.24)

$$\bar{D}_2U_i = \frac{2U_{i-1}}{h_i(h_i+h_{i+1})} - \frac{2U_i}{h_ih_{i+1}} + \frac{2U_{i+1}}{h_{i+1}(h_i+h_{i+1})} \tag{2.34}$$

for $i = 1, 2, \ldots, N$, then we obtain a set of algebraic equations

$$\frac{2U_{i-1}}{h_i(h_i+h_{i+1})} - \frac{2U_i}{h_ih_{i+1}} + \frac{2U_{i+1}}{h_{i+1}(h_i+h_{i+1})} = f(x_i) \tag{2.35}$$

for $i = 1, 2, \ldots, N$.

Due to (1.2), the first equation ($i = 1$) becomes

$$\bar{D}_2U_1 = -\frac{2U_1}{h_1h_2} + \frac{2U_2}{h_2(h_1+h_2)} \tag{2.36}$$

and the last equation ($j = N$) becomes

$$\bar{D}_2U_N = \frac{2U_{N-1}}{h_N(h_N+h_{N+1})} - \frac{2U_N}{h_Nh_{N+1}} \tag{2.37}$$

We have a linear system of N equations for the N unknowns, which can be written is the matrix form

$$AU = F \tag{2.38}$$

where U is the vector of unknowns

$$U = [U_1, U_2, \ldots, U_N]^T \tag{2.39}$$

$$A = 2\begin{bmatrix} -\frac{1}{h_1h_2} & \frac{1}{h_2(h_1+h_2)} & & & & \\ \frac{1}{h_2(h_2+h_3)} & -\frac{1}{h_2h_3} & \frac{1}{h_3(h_2+h_3)} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \frac{1}{h_{N-1}(h_{N-1}+h_N)} & -\frac{1}{h_{N-1}h_N} & \frac{1}{h_N(h_{N-1}+h_N)} & \\ & & & \frac{1}{h_N(h_N+h_{N+1})} & -\frac{1}{h_Nh_{N+1}} & \end{bmatrix} \tag{2.40}$$

and

$$F = \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \\ \vdots \\ f(x_{N-1}) \\ f(x_N) \end{bmatrix} \tag{2.41}$$

This tridiagonal linear system is nonsingular. Indeed, we now attempt to compute the determinant of A.

For a moment, we need to ignore the condition

$$\sum_{k=1}^{N} h_k = 1 \tag{2.42}$$

for fixed number N and put

$$D_N = \begin{vmatrix} -\frac{1}{h_1 h_2} & \frac{1}{h_2(h_1+h_2)} & & & \\ \frac{1}{h_2(h_2+h_3)} & -\frac{1}{h_2 h_3} & \frac{1}{h_3(h_2+h_3)} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{h_{N-1}(h_{N-1}+h_N)} & -\frac{1}{h_{N-1}h_N} & \frac{1}{h_N(h_{N-1}+h_N)} \\ & & & \frac{1}{h_N(h_N+h_{N+1})} & -\frac{1}{h_N h_{N+1}} \end{vmatrix} \tag{2.43}$$

and prove by induction that

$$D_N = \frac{(-1)^N \sum_{k=1}^{N+1} h_k}{\prod_{k=1}^{N+1} h_k \prod_{k=1}^{N} (h_k + h_{k+1})}, \forall N \in \mathbb{Z}_+ \tag{2.44}$$

It is easy to check that (2.44) holds for $N = 1, 2$. Suppose that (2.44) holds for N, we now prove (2.44) for $N+1$. We need to compute

$$D_{N+1} \tag{2.45}$$

$$= \begin{vmatrix} -\frac{1}{h_1 h_2} & \frac{1}{h_2(h_1+h_2)} & & & \\ \frac{1}{h_2(h_2+h_3)} & -\frac{1}{h_2 h_3} & \frac{1}{h_3(h_2+h_3)} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{h_N(h_N+h_{N+1})} & -\frac{1}{h_N h_{N+1}} & \frac{1}{h_{N+1}(h_N+h_{N+1})} \\ & & & \frac{1}{h_{N+1}(h_{N+1}+h_{N+2})} & -\frac{1}{h_{N+1}h_{N+2}} \end{vmatrix} \tag{2.46}$$

Expanding respect to the last row of D_{N+1}, we obtain

$$D_{N+1} = -\frac{1}{h_{N+1}h_{N+2}}D_N - \frac{1}{h_{N+1}(h_{N+1}+h_{N+2})}\times \\ \times \begin{vmatrix} -\frac{1}{h_1h_2} & \frac{1}{h_2(h_1+h_2)} & & & \\ \frac{1}{h_2(h_2+h_3)} & -\frac{1}{h_2h_3} & \frac{1}{h_3(h_2+h_3)} & & \\ & & \ddots & & \\ & & \frac{1}{h_{N-1}(h_{N-1}+h_N)} & -\frac{1}{h_{N-1}h_N} & \\ & & & \frac{1}{h_N(h_N+h_{N+1})} & \frac{1}{h_{N+1}(h_N+h_{N+1})} \end{vmatrix} \tag{2.47}$$

Expanding respect to the last column of the remaining determinant again, we obtain

$$D_{N+1} = -\frac{1}{h_{N+1}h_{N+2}}D_N - \frac{1}{h_{N+1}^2\left(h_N+h_{N+1}\right)\left(h_{N+1}+h_{N+2}\right)}D_{N-1} \tag{2.48}$$

Using the induction hypotheses, we get

$$D_{N+1} \tag{2.49}$$

$$= -\frac{1}{h_{N+1}h_{N+2}}D_N - \frac{1}{h_{N+1}^2\left(h_N+h_{N+1}\right)\left(h_{N+1}+h_{N+2}\right)}D_{N-1} \tag{2.50}$$

$$= -\frac{(-1)^N\sum_{k=1}^{N+1}h_k}{h_{N+1}h_{N+2}\prod_{k=1}^{N+1}h_k\prod_{k=1}^{N}\left(h_k+h_{k+1}\right)} \tag{2.51}$$

$$-\frac{(-1)^{N-1}\sum_{k=1}^{N}h_k}{h_{N+1}^2\left(h_N+h_{N+1}\right)\left(h_{N+1}+h_{N+2}\right)\prod_{k=1}^{N}h_k\prod_{k=1}^{N-1}\left(h_k+h_{k+1}\right)} \tag{2.52}$$

$$= \frac{(-1)^{N+1}\sum_{k=1}^{N+1}h_k}{h_{N+1}\prod_{k=1}^{N+2}h_k\prod_{k=1}^{N}\left(h_k+h_{k+1}\right)} - \frac{(-1)^{N+1}\sum_{k=1}^{N}h_k}{h_{N+1}\prod_{k=1}^{N+1}h_k\prod_{k=1}^{N+1}\left(h_k+h_{k+1}\right)} \tag{2.53}$$

$$= \frac{(-1)^{N+1}\left(\left(h_{N+1}+h_{N+2}\right)\sum_{k=1}^{N+1}h_k - h_{N+2}\sum_{k=1}^{N}h_k\right)}{h_{N+1}\prod_{k=1}^{N+2}h_k\prod_{k=1}^{N+1}\left(h_k+h_{k+1}\right)} \tag{2.54}$$

$$= \frac{(-1)^{N+1}\left(h_{N+1}\sum_{k=1}^{N+1}h_k + h_{N+1}h_{N+2}\right)}{h_{N+1}\prod_{k=1}^{N+2}h_k\prod_{k=1}^{N+1}\left(h_k+h_{k+1}\right)} \tag{2.55}$$

$$= \frac{(-1)^{N+1}\left(\sum_{k=1}^{N+2}h_k\right)}{\prod_{k=1}^{N+2}h_k\prod_{k=1}^{N+1}\left(h_k+h_{k+1}\right)} \tag{2.56}$$

Hence, (2.44) holds for $N+1$. By induction principle (2.44) holds for all positive integer N.

Thus, for a fixed number N and under (2.42), we deduce that

$$\det(A) = \frac{(-2)^N}{\prod_{i=1}^{N+1} h_i \prod_{i=1}^{N} (h_i + h_{i+1})} \neq 0 \tag{2.57}$$

i.e., A is nonsingular.

Thus (2.38) can be solved for U from any right-hand side F.

Next, if we let $\hat{U}$ be the vector of the values

$$\hat{U} = \begin{bmatrix} u(x_1) \\ u(x_2) \\ \vdots \\ u(x_N) \end{bmatrix} \tag{2.58}$$

then the error vector E defined by

$$E = U - \hat{U} \tag{2.59}$$

contains the errors at each grid point.

We next use the basic technique to the analysis of finite difference methods in general. It involves two keys steps. We first compute the *local truncation error* (LTE) of the method and then use some form of *stability* to show that the *global error* can be bounded in terms of the LTE.

2.3 Local Truncation Error

The LTE is defined by replacing U_i with the true solution $u(x_i)$ in the finite difference formula (2.34). In general the true solution $u(x_i)$ won't satisfy this equation exactly and the discrepancy is the LTE, which we denote by τ_i

$$\tau_i = \frac{2u(x_{i-1})}{h_i(h_i + h_{i+1})} - \frac{2u(x_i)}{h_i h_{i+1}} + \frac{2u(x_{i+1})}{h_{i+1}(h_i + h_{i+1})} - f(x_i) \tag{2.60}$$

for $i = 1, 2, \ldots, N$. Of course in practice we don't know what the true solution $u(x)$ is, but if we assume it is smooth, then by the Taylor series expansion, we know that

$$\tau_i = u''(x_i) + \frac{1}{3}(h_{i+1} - h_i) u'''(x_i) + \frac{1}{12}\frac{h_i^3 + h_{i+1}^3}{h_i + h_{i+1}} u^{(4)}(x_i) \tag{2.61}$$

$$+ \frac{1}{h_i(h_i + h_{i+1})} O\left(h_i^5\right) + \frac{1}{h_{i+1}(h_i + h_{i+1})} O\left(h_{i+1}^5\right) - f(x_i) \tag{2.62}$$

Using our original differential equation (1.1) this becomes

$$\tau_i = \frac{1}{3}(h_{i+1} - h_i) u'''(x_i) + \frac{1}{12}\frac{h_i^3 + h_{i+1}^3}{h_i + h_{i+1}} u^{(4)}(x_i) \tag{2.63}$$

$$+ \frac{1}{h_i(h_i + h_{i+1})} O\left(h_i^5\right) + \frac{1}{h_{i+1}(h_i + h_{i+1})} O\left(h_{i+1}^5\right) \tag{2.64}$$

We have proved

$$\tau_i = O\left(h^2\right) \text{ as } h \to 0 \tag{2.65}$$

If we define τ to be the vector with components τ_i, then

$$\tau = A\hat{U} - F \tag{2.66}$$

and so

$$A\hat{U} = F + tau \tag{2.67}$$

2.4 Global Error

To obtain a relation between the local error τ and the global error $E = U - \hat{U}$, we subtract (2.67) from (2.38), obtaining

$$AE = -\tau \tag{2.68}$$

This is simply the matrix form of the system of equations

$$\frac{2U_{i-1}}{h_i\left(h_i + h_{i+1}\right)} - \frac{2U_i}{h_i h_{i+1}} + \frac{2U_{i+1}}{h_{i+1}\left(h_i + h_{i+1}\right)} = -\tau\left(x_i\right) \tag{2.69}$$

for $i = 1, 2, \ldots, N$ with the boundary conditions

$$E_0 = E_{N+1} = 0 \tag{2.70}$$

since we are using the exact boundary data $U_0 = U_{N+1} = 0$. We see that the global error satisfies a set of finite difference equations that has exactly the same form as our original difference equations for U except that the right-hand side is given by $-\tau$ rather than F.

From this it should be clear why we expect the global error to be roughly the same magnitude as the local error τ. We can interpret the system (2.68) as a discretization of the ODE

$$e''(x) = -\tau(x), 0 < x < 1 \tag{2.71}$$

$$e(0) = 0, e(1) = 0 \tag{2.72}$$

Since

$$\tau_i = \frac{1}{3}h^2 u'''\left(x_i\right) X''\left(z_i\right) + O\left(h^3\right) \tag{2.73}$$

$$+ \frac{1}{12}h^2 u^{(4)}\left(x_i\right)\left[\left(X'\left(z_{i-1}\right)\right)^2 + X'\left(z_{i-1}\right) X'\left(z_i\right) + \left(X'\left(z_i\right)\right)^2\right] \tag{2.74}$$

integrating twice shows that the global error should be roughly

$$e(x) \tag{2.75}$$

$$= -h^2 \iint \left\{ \begin{array}{l} \frac{1}{3}u'''\left(x_i\right) X''\left(z_i\right) \\ +\frac{1}{12}u^{(4)}\left(x_i\right)\left[\left(X'\left(z_{i-1}\right)\right)^2 + X'\left(z_{i-1}\right) X'\left(z_i\right) + \left(X'\left(z_i\right)\right)^2\right] \end{array} \right\} \tag{2.76}$$

$$+ O\left(h^3\right) \tag{2.77}$$

$$\tag{2.78}$$

and hence the error should be $O\left(h^2\right)$ as $h \to 0$.

2.5 Stability

We rewrite (2.68) in the form

$$A^h E^h = -\tau^h \tag{2.79}$$

where the superscript h indicates that we are on a nonuniform grid associated with a uniform grid with mesh spacing h. This serves as a reminder that these quantities change as we refine the grid. In particular, the matrix A^h is an $N \times N$ matrix with $h = \frac{1}{N+1}$ so that its dimension is growing as $h \to 0$.

Let $\left(A^h\right)^{-1}$ be the inverse of this matrix. The solving the system (2.79) gives

$$E^h = \left(A^h\right)^{-1} \tau^h \tag{2.80}$$

and taking norms gives

$$\left\|E^h\right\| = \left\|\left(A^h\right)^{-1} \tau^h\right\| \tag{2.81}$$

$$\leq \left\|\left(A^h\right)^{-1}\right\| \left\|\tau^h\right\| \tag{2.82}$$

We know that $\left\|\tau^h\right\| = O\left(h^2\right)$ and we are hoping the same will be true of $\left\|E^h\right\|$. It is clear what we need for this to be true: we need $\left\|\left(A^h\right)^{-1}\right\|$ to be bounded by some constant independent of h as $h \to 0$:

$$\left\|\left(A^h\right)^{-1}\right\| \leq C \text{for all } h \text{ sufficiently small} \tag{2.83}$$

Then we will have

$$\left\|E^h\right\| \leq C \left\|\tau^h\right\| \tag{2.84}$$

and so $\left\|E^h\right\|$ goes to zero at least as fast as $\left\|\tau^h\right\|$. This motivates the following definition of *stability* for linear BVPs.

Definition 2.3. *Suppose a finite difference method for a linear BVP gives a sequence of matrix equations of the form $A^h U^h = F^h$, where h is the mesh width of associated uniform grid. We say that the method is stable if $\left(A^h\right)^{-1}$ exists for all h sufficiently small (for $h < h_0$, say) and if there is a constant C, independent of h, such that*

$$\left\|\left(A^h\right)^{-1}\right\| \leq C, \forall h < h_0 \tag{2.85}$$

2.6 Consistency

Definition 2.4. *We say that a method is consistent with the differential equation and boundary conditions if*

$$\left\|\tau^h\right\| \to 0 \text{ as } h \to 0 \tag{2.86}$$

This simply says that we have a sensible discretization of the problem. Typically $\left\|\tau^h\right\| = O\left(h^p\right)$ for some integer $p > 0$, and then the method is certainly consistent.

Return to our problem, we have proved $\left\|\tau^h\right\| = O\left(h^2\right)$. Thus, our method is consistent with (1.1) and (1.2).

2.7 Convergence

Definition 2.5. *A method is said to be convergent if*

$$\left\|E^h\right\| \to 0 \text{ as } h \to 0 \tag{2.87}$$

Combining the ideas introduced above we arrive at the conclusion that

$$\text{consistency} + \text{stability} \Rightarrow \text{convergence} \tag{2.88}$$

This is easily proved by using (2.88) and (2.86) to obtain the bound

$$\left\|E^h\right\| \leq \left\|\left(A^h\right)^{-1}\right\| \left\|\tau^h\right\| \leq C \left\|\tau^h\right\| \to 0 \text{ as } h \to 0 \tag{2.89}$$

Although this has been demonstrated only for the linear BVP, in fact most analyzes of finite difference methods for differential equations follow this same two-tier approach, and the statement (2.88) is sometimes called the *fundamental theorem of finite difference methods*. In fact, as our above analysis indicates, this can generally be strengthened to say that

$$O\left(h^p\right) \text{ local truncation error} + \text{stability} \Rightarrow O\left(h^p\right) \text{ global error} \tag{2.90}$$

2.8 Inverse of The Tridiagonal Matrix A

There is a explicit formula for inverse of general diagonal matrices described in [2], [3] and [4]. See Appendices.

Notations 2.6.

$$b_k = -\frac{2}{h_k h_{k+1}}, k = 1 \sim N \tag{2.91}$$

$$a_k = \frac{2}{h_k \left(h_k + h_{k+1}\right)}, k = 2 \sim N \tag{2.92}$$

$$c_k = \frac{2}{h_{k+1} \left(h_k + h_{k+1}\right)}, k = 1 \sim N - 1 \tag{2.93}$$

Define the second-order linear recurrences

$$z_i = -\frac{2}{h_i h_{i+1}} z_{i-1} - \frac{4}{h_i^2 \left(h_i + h_{i+1}\right)\left(h_{i-1} + h_i\right)} z_{i-2} \tag{2.94}$$

for $i = 2, 3, \ldots, N$, where $z_0 = 1, z_1 = -\frac{2}{h_1 h_2}$ and

$$y_j = -\frac{2}{h_j h_{j+1}} y_{j+1} - \frac{4}{h_{j+1}^2 \left(h_j + h_{j+1}\right)\left(h_{j+1} + h_{j+2}\right)} y_{j+2} \tag{2.95}$$

for $j = N-1, N-2, \ldots, 1$, where $y_{N+1} = 1, y_N = -\frac{2}{h_N h_{N+1}}$.

It is not difficult to find the explicit formulas of z_i's as

$$z_n = 2^n D_n \tag{2.96}$$

$$= \frac{(-1)^n \sum_{k=1}^{n+1} h_k}{\prod_{k=1}^{n+1} h_k \prod_{k=1}^{n} (h_k + h_{k+1})}, n = 1 \sim N \tag{2.97}$$

and

$$y_n = \frac{(-2)^{N-n+1} \sum_{k=n}^{N+1} h_k}{\prod_{k=n}^{N+1} h_k \prod_{k=n}^{N} (h_k + h_{k+1})}, n = 1 \sim N \tag{2.98}$$

The inverse matrix $B = A^{-1}$ can be expressed as

$$B_{ii} = \frac{z_{i-1} y_{i+1}}{z_N} \tag{2.99}$$

$$= -\frac{\sum_{k=1}^{i} h_k \sum_{k=i+1}^{N+1} h_k \prod_{k=1}^{N+1} h_k \prod_{k=1}^{N} (h_k + h_{k+1})}{2 \sum_{k=1}^{N+1} h_k \prod_{k=1}^{i} h_k \prod_{k=1}^{i-1} (h_k + h_{k+1}) \prod_{k=i+1}^{N+1} h_k \prod_{k=i+1}^{N} (h_k + h_{k+1})} \tag{2.100}$$

$$= -\frac{(h_i + h_{i+1}) \sum_{k=1}^{i} h_k \sum_{k=i+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k} \tag{2.101}$$

For $i < j$,

$$B_{ij} = (-1)^{i+j} \left(\prod_{k=i}^{j-1} c_k \right) y_{j+1} \frac{z_{i-1}}{z_N} \tag{2.102}$$

$$= -\frac{\prod_{k=1}^{N} (h_k + h_{k+1}) \sum_{k=1}^{i} h_k \sum_{k=j+1}^{N+1} h_k}{2 \prod_{k=1}^{i-1} (h_k + h_{k+1}) \prod_{k=i}^{j-1} (h_k + h_{k+1}) \prod_{k=j+1}^{N} (h_k + h_{k+1}) \sum_{k=1}^{N+1} h_k} \tag{2.103}$$

$$= -\frac{(h_j + h_{j+1}) \sum_{k=1}^{i} h_k \sum_{k=j+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k} \tag{2.104}$$

Similarly, for $i > j$,

$$B_{ij} = (-1)^{i+j} \left(\prod_{k=j+1}^{i} a_k \right) y_{i+1} \frac{z_{j-1}}{z_N} \tag{2.105}$$

$$= -\frac{\prod_{k=1}^{N} (h_k + h_{k+1}) \sum_{k=1}^{j} h_k \sum_{k=i+1}^{N+1} h_k}{2 \prod_{k=1}^{j-1} (h_k + h_{k+1}) \prod_{k=j+1}^{i} (h_k + h_{k+1}) \prod_{k=i+1}^{N} (h_k + h_{k+1}) \sum_{k=1}^{N+1} h_k} \tag{2.106}$$

$$= -\frac{(h_j + h_{j+1}) \sum_{k=1}^{j} h_k \sum_{k=i+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k} \tag{2.107}$$

Therefore, the inverse matrix $B = A^{-1}$ is given by

$$B_{ij} = \begin{cases} -\frac{(h_j + h_{j+1}) \sum_{k=1}^{i} h_k \sum_{k=j+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k}, i < j \\ -\frac{(h_i + h_{i+1}) \sum_{k=1}^{i} h_k \sum_{k=i+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k}, i = j \\ -\frac{(h_j + h_{j+1}) \sum_{k=1}^{j} h_k \sum_{k=i+1}^{N+1} h_k}{2 \sum_{k=1}^{N+1} h_k}, i > j \end{cases} \tag{2.108}$$

or compactly,

$$B_{ij} = -\frac{(h_j + h_{j+1}) \sum_{k=1}^{\min(i,j)} h_k \sum_{k=\max(i,j)+1}^{N+1} h_k}{2}, \forall 1 \leq i, j \leq N \tag{2.109}$$

2.9 Stability in 1-Norm

As the uniform grid case in the next section, we will calculate some necessary information about the inverse of A and then show that $\left\|A^{-1}\right\|_1 = O(1)$, and hence

$$\|E\|_1 \leq \left\|A^{-1}\right\|_1 \|\tau\|_1 = O\left(h^2\right) \tag{2.110}$$

since $\|\tau\|_1 = O\left(h^2\right)$.

We study the 1-norm stability of the finite difference method, which will be based on explicitly determining the inverse matrix of the matrix A defined by (2.40).

Using (2.109), we find out

$$\sum_{i=1}^{N} |B_{ij}| = \frac{h_j + h_{j+1}}{2} \sum_{m=1}^{j} \sum_{n=j+1}^{N+1} (n - m) h_m h_n \tag{2.111}$$

Hence,

$$\|B\|_1 = \max_{1\le j\le N} \sum_{i=1}^{N} |B_{ij}| \tag{2.112}$$

$$= \max_{1\le j\le N} \frac{h_j + h_{j+1}}{2} \sum_{m=1}^{j} \sum_{n=j+1}^{N+1} (n-m)\, h_m h_n \tag{2.113}$$

$$= \max_{1\le j\le N} O(h) \sum_{m=1}^{j} \sum_{n=j+1}^{N+1} (n-m)\, O\left(h^2\right) \tag{2.114}$$

$$= \left(\max_{1\le j\le N} \sum_{m=1}^{j} \sum_{n=j+1}^{N+1} (n-m) \right) O\left(h^3\right) \tag{2.115}$$

$$= \left(\max_{1\le j\le N} \frac{j\,(N+1)\,(N-j+1)}{2} \right) O\left(h^3\right) \tag{2.116}$$

$$\le \frac{(N+1)^3}{8} O\left(h^3\right) \tag{2.117}$$

$$= \frac{1}{8h^3} O\left(h^3\right) \tag{2.118}$$

$$= O(1) \tag{2.119}$$

where we have used Cauchy inequality to estimate

$$j\,(N-j+1) \le \frac{(N+1)^2}{4}, \;\; 1 \le j \le N \tag{2.120}$$

We have proved that $\|B\|_1 = O(1)$. Hence our difference method is stable in 1-norm.

Furthermore, we have $\left\|E^h\right\|_1 = O\left(h^2\right)$ as $h \to 0$. □

2.10 Stability in Infinity-Norm

Similarly, using (2.109), we find out

$$\sum_{j=1}^{N} |B_{ij}| = \frac{1}{2} \left(\sum_{k=1}^{i} h_k \right) \left(\sum_{k=i+1}^{N+1} h_k \right) \tag{2.121}$$

Hence

$$\|B\|_\infty = \max_{1\le i\le N}\sum_{j=1}^{N}|B_{ij}| \tag{2.122}$$

$$= \max_{1\le i\le N}\frac{1}{2}\left(\sum_{k=1}^{i}h_k\right)\left(\sum_{k=i+1}^{N+1}h_k\right) \tag{2.123}$$

$$= \left(\max_{1\le i\le N}\frac{1}{2}\left(\sum_{k=1}^{i}1\right)\left(\sum_{k=i+1}^{N+1}1\right)\right)O\left(h^2\right) \tag{2.124}$$

$$= \left(\max_{1\le i\le N}\frac{i\left(N-i+1\right)}{2}\right)O\left(h^2\right) \tag{2.125}$$

$$\le \frac{\left(N+1\right)^2}{4}O\left(h^2\right) \tag{2.126}$$

$$= \frac{1}{4h^2}O\left(h^2\right) \tag{2.127}$$

$$= O\left(1\right) \tag{2.128}$$

where we have used Cauchy inequality to estimate

$$i\left(N-i+1\right) \le \frac{\left(N+1\right)^2}{4},\ \ 1\le i\le N \tag{2.129}$$

We have proved that $\|B\|_\infty = O\left(1\right)$. Hence our difference method is stable in infinity-norm.

Furthermore, $\left\|E^h\right\|_\infty = O\left(h^2\right)$ as $h\to 0$. □

2.11 Stability in Frobenius Norm

Problem 2.7 *Prove our finite method* (2.35) *is stable in Frobenius norm.*

We have two solutions for stability of our method in Frobenius norm.

Solution 1. By computation and a little counting combination, we have the following explicit formula for Frobenius norm of A^{-1}.

$$\|B\|_F = \left(\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N}\frac{1}{4}\left(\sum_{k=1}^{\alpha}h_k\right)^2\left(\sum_{k=\beta}^{\gamma}h_k\right)^2\left(\sum_{k=\delta}^{N+1}h_k\right)^2\right)^{\frac{1}{2}} \tag{2.130}$$

where indices $(\alpha,\beta,\gamma,\delta)$ belong to a set Fr_N described as

$$\alpha\le\beta<\beta+1=\gamma\le\delta \tag{2.131}$$

and belong to one of following cases.

1. **Case** $\alpha=1,\delta=N+1$**.** Then

(a) **Subcase** $\beta = 2$**.** We must have $\gamma = 3$. And the term appearing in (2.130) in this subcase is

$$\alpha = 1 \tag{2.132}$$
$$\beta = 2 \tag{2.133}$$
$$\gamma = 3 \tag{2.134}$$
$$\delta = N + 1 \tag{2.135}$$

(b) **Subcase** $\beta = N$**.** Then $\gamma = N + 1$. And the term appearing in (2.130) in this subcase is

$$\alpha = 1 \tag{2.136}$$
$$\beta = N \tag{2.137}$$
$$\gamma = N + 1 \tag{2.138}$$
$$\delta = N + 1 \tag{2.139}$$

Thus, there are 2 terms appearing in (2.130) in this case.

2. **Case** $\alpha = 1, \delta \leq N$**.** Then $\gamma = \beta + 1$. We have two following subcases.

(a) **Subcase** $\beta = 1$**.** Then $\gamma = 2$ and we obtain some terms appearing in (2.130) with the following form.

$$\alpha = 1 \tag{2.140}$$
$$\beta = 1 \tag{2.141}$$
$$\gamma = 2 \tag{2.142}$$
$$\delta = 2, 3, \ldots N \tag{2.143}$$

Counting choices of δ, we have $N - 1$ terms appearing in (2.130) in this subcase.

(b) **Subcase** $\beta \geq 2$**.** Then $\delta = \gamma$ and we obtain some terms appearing in (2.130) with the following form

$$\alpha = 1 \tag{2.144}$$
$$\beta = 2, \ldots N - 1 \tag{2.145}$$
$$\gamma = \beta + 1 \tag{2.146}$$
$$\delta = \gamma \tag{2.147}$$

Counting choices of β, we have $N - 2$ terms appearing in (2.130) in this subcase.

Thus, there are $2N - 3$ terms appearing in (2.130) in this case.

3. **Case** $\alpha \geq 2, \delta = N + 1$**.** Then $\gamma = \beta + 1$. We have two following subcases.

(a) **Subcase** $\beta = \alpha$**.** We obtain some terms appearing in (2.130) with the following form.

$$\alpha = 2, 3, \ldots, N - 1 \tag{2.148}$$
$$\beta = \alpha \tag{2.149}$$
$$\gamma = \beta + 1 \tag{2.150}$$
$$\delta = N + 1 \tag{2.151}$$

Counting choices of α, we have $N-2$ terms appearing in (2.130) in this subcase.

(b) **Subcase $\beta = N$.** We obtain some terms appearing in (2.130) with the following form.

$$\alpha = 2, 3, \ldots, N \tag{2.152}$$
$$\beta = N \tag{2.153}$$
$$\gamma = N+1 \tag{2.154}$$
$$\delta = N+1 \tag{2.155}$$

Counting choices of α, we have $N-1$ terms appearing in (2.130) in this subcase.

Thus, there are $2N-3$ terms appearing in (2.130) in this case.

4. **Case $\alpha \geq 2, \delta \leq N$.** We also have two following subcases.

(a) **Subcase $\alpha = \beta$.** We obtain some terms appearing in (2.130) with the following form.

$$2 \leq \alpha < \delta \leq N \tag{2.156}$$
$$\beta = \alpha \tag{2.157}$$
$$\gamma = \beta + 1 \tag{2.158}$$

Counting choice of (α, δ), we have $\begin{pmatrix} N-1 \\ 2 \end{pmatrix}$ terms appearing in (2.130) in this subcase.

(b) **Subcase $\alpha \neq \beta$.** We obtain some terms appearing in (2.130) with the following form.

$$2 \leq \alpha < \beta \leq N-1 \tag{2.159}$$
$$\gamma = \beta + 1 \tag{2.160}$$
$$\delta = \gamma \tag{2.161}$$

Counting choice of (α, β), we have $\begin{pmatrix} N-2 \\ 2 \end{pmatrix}$ terms appearing in (2.130) in this subcase.

Thus, there are

$$\begin{pmatrix} N-1 \\ 2 \end{pmatrix} + \begin{pmatrix} N-2 \\ 2 \end{pmatrix} = \frac{(N-1)(N-2)}{2} + \frac{(N-2)(N-3)}{2} \tag{2.162}$$
$$= (N-2)\left(\frac{N-1}{2} + \frac{N-3}{2}\right) \tag{2.163}$$
$$= (N-2)^2 \tag{2.164}$$

terms appearing in (2.130) in this case.

In summary, we have

$$2 + (2N-3) + (2N-3) + (N-2)^2 = N^2 \tag{2.165}$$

terms appearing in (2.130) which belong to one of four described cases. This number is exactly the number of terms of a $N \times N$ matrix.

After describing (2.130) so hard, we are now ready to estimate Frobenius norm of A^{-1}. We have

$$\|B\|_F^2 = \sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \frac{1}{4}\left(\sum_{k=1}^{\alpha} h_k\right)^2 \left(\sum_{k=\beta}^{\gamma} h_k\right)^2 \left(\sum_{k=\delta}^{N+1} h_k\right)^2 \tag{2.166}$$

$$= \sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \frac{1}{4}\left(\sum_{k=1}^{\alpha} O(h)\right)^2 \left(\sum_{k=\beta}^{\gamma} O(h)\right)^2 \left(\sum_{k=\delta}^{N+1} O(h)\right)^2 \tag{2.167}$$

$$= \left(\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \frac{1}{4}\left(\sum_{k=1}^{\alpha} 1\right)^2 \left(\sum_{k=\beta}^{\gamma} 1\right)^2 \left(\sum_{k=\delta}^{N+1} 1\right)^2\right) O\left(h^6\right) \tag{2.168}$$

$$= \left(\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \frac{1}{4}\alpha^2(\gamma-\beta+1)^2(N+2-\delta)^2\right) O\left(h^6\right) \tag{2.169}$$

$$= \left(\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \alpha^2(N+2-\delta)^2\right) O\left(h^6\right) \tag{2.170}$$

We now estimate the term $\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \alpha^2(N+2-\delta)^2$ due to the described cases. The terms in the second equality appear in the order described subcases **1a, 1b, 2a, 2b, 3a, 3b, 4a** and **4b**.

$$\sum_{(\alpha,\beta,\gamma,\delta)\in Fr_N} \alpha^2(N+2-\delta)^2 \tag{2.171}$$

$$= 1^2(N+2-(N+1))^2 + 1^2(N+2-(N+1))^2 \tag{2.172}$$

$$+ \sum_{\delta=2}^{N} 1^2(N+2-\delta)^2 + \sum_{\delta=3}^{N} 1^2(N+2-\delta)^2 \tag{2.173}$$

$$+ \sum_{\alpha=2}^{N-1} \alpha^2(N+2-(N+1))^2 + \sum_{\alpha=2}^{N} \alpha^2(N+2-(N+1))^2 \tag{2.174}$$

$$+ \sum_{\delta=3}^{N}\sum_{\alpha=2}^{\delta-1} \alpha^2(N+2-\delta)^2 + \sum_{\delta=4}^{N}\sum_{\alpha=2}^{\delta-2} \alpha^2(N+2-\delta)^2 \tag{2.175}$$

$$= 1 + 1 + \sum_{\delta=2}^{N} (N+2-\delta)^2 + \sum_{\delta=3}^{N} (N+2-\delta)^2 + \sum_{\alpha=2}^{N-1} \alpha^2 \tag{2.176}$$

$$+ \sum_{\alpha=2}^{N} \alpha^2 + \sum_{\delta=3}^{N}\sum_{\alpha=2}^{\delta-1} \alpha^2(N+2-\delta)^2 + \sum_{\delta=4}^{N}\sum_{\alpha=2}^{\delta-2} \alpha^2(N+2-\delta)^2 \tag{2.177}$$

$$= 2 + \sum_{k=2}^{N} k^2 + \sum_{k=2}^{N-1} k^2 + \sum_{\alpha=2}^{N-1} \alpha^2 + \sum_{\alpha=2}^{N} \alpha^2 \tag{2.178}$$

$$+ \sum_{\delta=3}^{N}\sum_{\alpha=2}^{\delta-1} \alpha^2(N+2-\delta)^2 + \sum_{\delta=4}^{N}\sum_{\alpha=2}^{\delta-2} \alpha^2(N+2-\delta)^2 \tag{2.179}$$

$$= 2 + \frac{N\,(N+1)\,(2N+1)}{6} - 1 + \frac{N\,(N-1)\,(2N-1)}{6} - 1 \tag{2.180}$$

$$+ \frac{N\,(N-1)\,(2N-1)}{6} - 1 + \frac{N\,(N+1)\,(2N+1)}{6} - 1 \tag{2.181}$$

$$+ \frac{(N-1)\,(N-2)\left(2N^4+24N^3+133N^2+231N+180\right)}{360} \tag{2.182}$$

$$+ \frac{(N-2)\,(N-3)\left(2N^4+16N^3+73N^2+29N+60\right)}{360} \tag{2.183}$$

$$= \frac{N(N+1)^2\,(N+2)\left(2N^2+4N+9\right)}{180} \tag{2.184}$$

$$= \frac{-7h^4+5h^2+2}{180h^6} \tag{2.185}$$

We have immediately the following estimate

$$\left|\frac{-7h^4+5h^2+2}{180h^6}\right| \leq \frac{7h^4+5h^2+2}{180h^6} \leq \frac{14}{180h^6} = O\left(\frac{1}{h^6}\right) \tag{2.186}$$

Thus,

$$\|B\|_F = O\,(1) \tag{2.187}$$

Therefore, our method is stable in Frobenius norm.

Furthermore, $\left\|E^h\right\|_F = O\left(h^2\right)$ as $h \to 0$. □

Solution 2. This solution does not need counting combination as Solution 1. Using (2.109), we have

$$\|B\|_F^2 = \sum_{1\le i\le j\le N} |B_{ij}|^2 \tag{2.188}$$

$$= \sum_{1\le i,j\le N} \frac{1}{4}(h_j + h_{j+1})^2 \left(\sum_{k=1}^{\min(i,j)} h_k\right)^2 \left(\sum_{k=\max(i,j)+1}^{N+1} h_k\right)^2 \tag{2.189}$$

$$= \frac{1}{4} \sum_{1\le i<j\le N} (h_j + h_{j+1})^2 \left(\sum_{k=1}^{\min(i,j)} h_k\right)^2 \left(\sum_{k=\max(i,j)+1}^{N+1} h_k\right)^2 \tag{2.190}$$

$$+ \frac{1}{4} \sum_{1\le i=j\le N} (h_j + h_{j+1})^2 \left(\sum_{k=1}^{\min(i,j)} h_k\right)^2 \left(\sum_{k=\max(i,j)+1}^{N+1} h_k\right)^2 \tag{2.191}$$

$$+ \frac{1}{4} \sum_{1\le j<i\le N} (h_j + h_{j+1})^2 \left(\sum_{k=1}^{\min(i,j)} h_k\right)^2 \left(\sum_{k=\max(i,j)+1}^{N+1} h_k\right)^2 \tag{2.192}$$

$$= \frac{1}{4} \sum_{1\le i<j\le N} (O(h) + O(h))^2 \left(\sum_{k=1}^{i} O(h)\right)^2 \left(\sum_{k=j+1}^{N+1} O(h)\right)^2 \tag{2.193}$$

$$+ \frac{1}{4} \sum_{1\le i=j\le N} (O(h) + O(h))^2 \left(\sum_{k=1}^{j} O(h)\right)^2 \left(\sum_{k=j+1}^{N+1} O(h)\right)^2 \tag{2.194}$$

$$+ \frac{1}{4} \sum_{1\le j<i\le N} (O(h) + O(h))^2 \left(\sum_{k=1}^{j} O(h)\right)^2 \left(\sum_{k=i+1}^{N+1} O(h)\right)^2 \tag{2.195}$$

$$= \sum_{1\le i<j\le N} \left(\sum_{k=1}^{i} 1\right)^2 \left(\sum_{k=j+1}^{N+1} 1\right)^2 O\left(h^6\right) \tag{2.196}$$

$$+ \sum_{1\le i=j\le N} \left(\sum_{k=1}^{j} 1\right)^2 \left(\sum_{k=j+1}^{N+1} 1\right)^2 O\left(h^6\right) \tag{2.197}$$

$$+ \sum_{1\le j<i\le N} \left(\sum_{k=1}^{j} 1\right)^2 \left(\sum_{k=i+1}^{N+1} 1\right)^2 O\left(h^6\right) \tag{2.198}$$

$$= \Sigma O\left(h^6\right) \tag{2.199}$$

where

$$\Sigma := \sum_{1\le i<j\le N} \left(\sum_{k=1}^{i} 1\right)^2 \left(\sum_{k=j+1}^{N+1} 1\right)^2 + \sum_{1\le i=j\le N} \left(\sum_{k=1}^{j} 1\right)^2 \left(\sum_{k=j+1}^{N+1} 1\right)^2 \tag{2.200}$$

$$+ \sum_{1\le j<i\le N} \left(\sum_{k=1}^{j} 1\right)^2 \left(\sum_{k=i+1}^{N+1} 1\right)^2 \tag{2.201}$$

We now estimate Σ.

$$\Sigma = \sum_{1\le i<j\le N} i^2(N-j+1)^2 + \sum_{j=1}^{N} j^2(N-j+1)^2 + \sum_{1\le j<i\le N} j^2(N-i+1)^2 \tag{2.202}$$

$$= \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} i^2(N-j+1)^2 + \sum_{j=1}^{N} j^2(N-j+1)^2 + \sum_{j=1}^{N-1}\sum_{i=j+1}^{N} j^2(N-i+1)^2 \tag{2.203}$$

$$= \frac{N(N-1)(N+1)(N+2)\left(2N^2+2N+3\right)}{360} \tag{2.204}$$

$$+ \frac{N(N+1)(N+2)\left(N^2+2N+2\right)}{30} \tag{2.205}$$

$$+ \frac{N(N-1)(N+1)(N+2)\left(2N^2+2N+3\right)}{360} \tag{2.206}$$

$$= \frac{N(N+1)^2(N+2)\left(2N^2+4N+9\right)}{180} \tag{2.207}$$

$$= \frac{-7h^4+5h^2+2}{180h^6} \tag{2.208}$$

We also have the following estimate

$$\left|\frac{-7h^4+5h^2+2}{180h^6}\right| \le \frac{7h^4+5h^2+2}{180h^6} \le \frac{14}{180h^6} = O\left(\frac{1}{h^6}\right) \tag{2.209}$$

Thus,

$$\|B\|_F = O(1) \tag{2.210}$$

Therefore, our method is stable in Frobenius norm.

Furthermore, $\left\|E^h\right\|_\infty = O\left(h^2\right)$ as $h \to 0$. □

2.12 Stability in Other Norms

We have proved that our difference method is stable in 1-norm, infinity-norm and Frobenius norm, i.e.,

$$\|B\|_1 = O(1) \tag{2.211}$$

$$\|B\|_\infty = O(1) \tag{2.212}$$

$$\|B\|_F = O(1) \tag{2.213}$$

We now use some popular matrix norm inequalities listed in Appendix or [6] to deduce stability of our method in other matrix norms.

2.12.1 Stability in 2-Norm

Problem 2.8. *Prove the difference method* (2.35) *is stable in* ***2-norm****.*

Solution 1. Using (6.1) yields

$$\|B\|_2 \le \|B\|_F \le \sqrt{N}\|B\|_2 \tag{2.214}$$

Since $\|B\|_F = O(1)$, we deduce that $\|B\|_2 = O(1)$. Therefore our difference method is stable in 2-norm.

Furthermore, $\left\|E^h\right\|_2 = O\left(h^2\right)$ as $h \to 0$. □

SOLUTION 2. Using (5.6) yields

$$\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty} = O(1) \tag{2.215}$$

Therefore our difference method is stable in 2-norm.

Furthermore, $\left\|E^h\right\|_2 = O\left(h^2\right)$ as $h \to 0$. □

Remark 2.9. Using (5.3), (5.4) and (5.5) yields weaker estimate.

2.12.2 Error in Nuclear Norm

Problem 2.10. *Survey* (2.35) *in* ***nuclear norm*** *(also called* ***trace norm****, or the* ***Ky Fan*** n***-norm****.*

SOLUTION. Using (5.2) yields

$$\|B\|_* \leq \sqrt{N}\|B\|_F \leq \sqrt{N+1}\|B\|_F = h^{-\frac{1}{2}} O(1) = O\left(h^{-\frac{1}{2}}\right) \tag{2.216}$$

Hence

$$\left\|E^h\right\|_* \leq \left\|B^h\right\|_* \left\|\tau^h\right\|_* = O\left(h^{\frac{3}{2}}\right) \text{ as } h \to 0 \tag{2.217}$$

Although our finite difference method is not stable in nuclear norm, but the error equals to $O\left(h^{\frac{3}{2}}\right)$ as $h \to 0$. □

2.12.3 Other Norms

See [6] for other norms and use matrix norm inequality to survey our finite element method in other matrix norms.

3 Uniform Grid Case

The author also retype the uniform grid case in [1] here for completeness. Notice the differences between nonuniform grid we deal with and uniform one.

We now using finite difference method to discretize (1.1). To do this, we replace u_{xx} in (1.1) by the centered difference approximation

$$D^2 U_j = \frac{1}{h^2}\left(U_{j-1} - 2U_j + U_{j+1}\right) \tag{3.1}$$

then we obtain a set of algebraic equations

$$\frac{1}{h^2}\left(U_{j-1} - 2U_j + U_{j+1}\right) = f(x_j), \quad j = 1, 2 \ldots, N \tag{3.2}$$

Due to (1.6), the first equation ($j = 1$) becomes

$$\frac{1}{h^2}\left(-2U_1 + U_2\right) = f(x_1) \tag{3.3}$$

and the last equation ($j = N$) becomes

$$\frac{1}{h^2}\left(U_{N-1} - 2U_N\right) = f\left(x_N\right) \tag{3.4}$$

We have a linear system of N equations for the N unknowns, which can be written in the form

$$AU = F \tag{3.5}$$

where U is the vector of unknowns

$$U = \left[U_1, U_2, \ldots, U_N\right]^T \tag{3.6}$$

$$A = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & \end{bmatrix} \tag{3.7}$$

and

$$F = \begin{bmatrix} f\left(x_1\right) \\ f\left(x_2\right) \\ f\left(x_3\right) \\ \vdots \\ f\left(x_{N-1}\right) \\ f\left(x_N\right) \end{bmatrix} \tag{3.8}$$

This tridiagonal linear system is nonsingular. Indeed, we now compute the determinant of A. To this end, we denote

$$D_n = \begin{vmatrix} -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & \end{vmatrix} \tag{3.9}$$

is the determinant of the $n \times n$ matrix given by (3.9).

Expanding to the first column, we obtain

$$D_n = -2D_{n-1} - \begin{vmatrix} 1 & & & & & \\ 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & -2 & 1 & \\ & & & 1 & -2 & \end{vmatrix} \tag{3.10}$$

Again, expanding to the first column, we obtain the difference equation.

$$D_n + 2D_{n-1} + D_{n-2} = 0 \tag{3.11}$$

Solving this difference equation, we easily obtain the explicit formula for D_n.

$$D_n = (-1)^n (1+n), \quad n = 1, 2, \ldots \tag{3.12}$$

In particular, the determinant of A is

$$\det(A) = \left(-\frac{1}{h^2}\right)^N (1+N) \neq 0 \tag{3.13}$$

That means A is nonsingular, (3.5) has a unique solution from any right-hand side F.

Moreover, (3.5) can be easily solved for U from any right-hand side F.

3.1 Local Truncation Error

The LTE is defined by replacing U_i with the true solution $u(u_i)$ in the finite difference formula (3.2). In general the true solution $u(x_i)$ won't satisfy this equation exactly and the discrepancy is the LTE, which we denote by τ_i

$$\tau_i = \frac{1}{h^2}\left[u(x_{i-1}) - 2u(x_i) + u(x_{i+1})\right] - f(x_i) \tag{3.14}$$

for $i = 1, 2, \ldots, N$. Of course in practice we don't know what the true solution $u(x)$ is, but if we assume it is smooth, then by the Taylor series expansion, we know that

$$\tau_i = \left[u''(x_i) + \frac{1}{12}h^2 u^{(4)}(x_i) + O\left(h^4\right)\right] - f(x_i) \tag{3.15}$$

Using (1.1), this becomes

$$\tau_i = \frac{1}{12}h^2 u^{(4)}(x_i) + O\left(h^4\right) \tag{3.16}$$

Although $u^{(4)}$ is in general unknown, it is some fixed function independent of h, and so

$$\tau_i = O\left(h^2\right) \text{ as } h \to 0 \tag{3.17}$$

If we define τ to be the vector with components τ_i, then

$$\tau = A\hat{U} - F \tag{3.18}$$

where $\hat{U}$ is the vector of true solution values (2.58), and so

$$A\hat{U} = F + \tau \tag{3.19}$$

3.2 Global Error

To obtain a relation between the local error τ and the global error $E = U - \hat{U}$, we subtract (3.19) from (3.5) that defines U, obtaining

$$AE = -\tau \tag{3.20}$$

This is simply the matrix form of the system of equations

$$\frac{1}{h^2}\left(E_{i-1} - 2E_i + E_{i+1}\right) = -\tau\left(x_i\right) \tag{3.21}$$

for $i = 1, 2, \ldots, N$ with the boundary conditions

$$E_0 = E_{N+1} = 0 \tag{3.22}$$

since we are using the exact boundary data $U_0 = 0, U_{N+1} = 0$. We see that the global error satisfies a set of finite difference equations that has exactly the same form as our original difference equations for U except that the right-hand side is given by $-\tau$ rather than F.

From this it should be clear why we expect the global error to be roughly the same magnitude as the local error τ. We can interpret the system (3.20) as a discretization of the ODE

$$e''\left(x\right) = -\tau\left(x\right), 0 < x < 1 \tag{3.23}$$

with boundary condition

$$e\left(0\right) = 0, e\left(1\right) = 0 \tag{3.24}$$

Since

$$\tau\left(x\right) \approx \frac{1}{12}h^2 u^{(4)}\left(x\right) \tag{3.25}$$

integrating twice shows that the global error should be roughly

$$e\left(x\right) = \frac{-1}{12}h^2 u''\left(x\right) + \frac{1}{12}h^2\left(u''\left(0\right) + x\left(u''\left(1\right) - u''\left(0\right)\right)\right) \tag{3.26}$$

and hence the error should be $O\left(h^2\right)$.

3.3 Stability in The 2-Norm

Since the matrix A from (3.7) is symmetric, the 2-norm of A is equal to its spectral radius

$$\left[\|A\|_2 = \rho\left(A\right) = \max_{1 \le p \le m} |\lambda_p| \right. \tag{3.27}$$

The matrix A^{-1} is also symmetric, and the eigenvalues of A^{-1} are simply the inverses of the eigenvalues of A, so

$$\left\|A^{-1}\right\|_2 = \rho\left(A^{-1}\right) = \max_{1 \le p \le m} \left|\left(\lambda_p\right)^{-1}\right| = \left(\min_{1 \le p \le m} |\lambda_p|\right)^{-1} \tag{3.28}$$

So all we need to do is compute the eigenvalues of A and show that they are bounded away from zero as $h \to 0$. Of course, we have an infinite set of matrices A^h to consider as h varies, but since the structure of these matrices is so simple, we can obtain a general expression for the eigenvalues of each A^h. For more complicated problems we might not be able to do this.

We will now focus on one particular value of $h = \frac{1}{N+1}$ and drop the superscript h to simplify the notation. Then the N eigenvalues of A are given by

$$\lambda_p = \frac{2}{h^2}\left(\cos\left(p\pi h\right) - 1\right), \quad p = 1, 2, \ldots, N \tag{3.29}$$

The eigenvector u^p corresponding to λ_p has components u_i^p for $i = 1, 2, \ldots, N$ given by

$$u_i^p = \sin\left(p\pi i h\right) \tag{3.30}$$

This can be verified by checking that $Au^p = \lambda_p u^p$. The ith component of the vector Au^p is

$$\begin{aligned}
\left(Au^p\right)_i &= \frac{1}{h^2}\left(u_{i-1}^p - 2u_i^p + u_{i+1}^p\right) && (3.31)\\
&= \frac{1}{h^2}\left(\sin\left(p\pi\left(i-1\right)h\right) - 2\sin\left(p\pi i h\right) + \sin\left(p\pi\left(i+1\right)h\right)\right) && (3.32)\\
&= \frac{1}{h^2}\left(\sin\left(p\pi i h\right)\cos\left(p\pi h\right) - 2\sin\left(p\pi i h\right) + \sin\left(p\pi i h\right)\cos\left(p\pi h\right)\right) && (3.33)\\
&= \lambda_p u_i^p && (3.34)
\end{aligned}$$

Note that for $i = 1$ and $i = m$ the ith component of Au^p looks slightly different the u_{i-1}^p or u_{i+1}^p term is missing) but the above form and trigonometric manipulations are still valid provided that we define

$$u_0^p = u_{N+1}^p = 0 \tag{3.35}$$

as is consistent with (3.30). From (3.29) we see that the smallest eigenvalue of A (in magnitude) is

$$\begin{aligned}
\lambda_1 &= \frac{2}{h^2}\left(\cos\left(\pi h\right) - 1\right) && (3.36)\\
&= \frac{2}{h^2}\left(-\frac{1}{2}\pi^2 h^2 + \frac{1}{24}\pi^4 h^4 + O\left(h^6\right)\right) && (3.37)\\
&= -\pi^2 + O\left(h^2\right) && (3.38)
\end{aligned}$$

This is clearly bounded away from zero as $h \to 0$, and so we see the method is stable in 2-norm. Moreover we get an error bounded from this

$$\left\|E^h\right\|_2 \leq \left\|\left(A^h\right)^{-1}\right\|_2 \left\|\tau^h\right\|_2 \approx \frac{1}{\pi^2}\left\|\tau^h\right\|_2 \tag{3.39}$$

Since

$$\tau_i^h \approx \frac{1}{12}h^2 u^{(4)}\left(x_i\right) \tag{3.40}$$

we expect

$$\left\|\tau_i^h\right\| \approx \frac{1}{12}h^2\left\|u^{(4)}\right\|_2 = \frac{1}{12}h^2\left\|f''\right\|_2 \tag{3.41}$$

The 2-norm of the function f'' here means the grid-function norm of this function evaluated at the discrete points x_i, although this is approximately equal to the function space norm of f'' defined by

$$\left\|f\right\|_2 = \left(\int_a^b \left|f\left(x\right)\right|^2 dx\right)^{\frac{1}{2}} \tag{3.42}$$

3.4 Green's Functions and infinity-norm Stability

We demonstrated that A from (3.7) is stable in the 2-norm, and hence $\|E\|_2 = O\left(h^2\right)$. Suppose, however, that we want a bound on the maximum error over the interval, i.e., a bound on $\|E\|_\infty = \max_{1\le i\le N} |E_i|$. We can obtain one such bound directly from the bound we have for the 2-norm. That is

$$\|E\|_\infty \le \frac{1}{\sqrt{h}}\|E\|_2 = O\left(h^{\frac{3}{2}}\right) \text{ as } h \to 0 \tag{3.43}$$

However, this does not show the second order accurate that we hope to have. To show that $\|E\|_\infty = O\left(h^2\right)$ we will explicitly calculate the inverse of A and then show that $\left\|A^{-1}\right\|_\infty = O(1)$, and hence

$$\|E\|_\infty \le \left\|A^{-1}\right\|_\infty \|\tau\|_\infty = O\left(h^2\right) \tag{3.44}$$

since $\|\tau\|_\infty = O\left(h^2\right)$. As in the computation of the eigenvalues in the last subsection, we can do this only because our model problem (1.1) is so simple. In general it would be impossible to obtain closed form expressions for the inverse of the matrices A^h as h varies.

Each column of the inverse matrix can be interpreted as the solution of a particular BVP. The columns are discrete approximations to the *Green's function*.

For any fixed point $\bar{x} \in [0,1]$, the Green's function $G(x;\bar{x})$ to the BVP

$$u''(x) = f(x), 0 < x < 1 \tag{3.45}$$

with Dirichlet boundary conditions

$$u(0) = 0, u(1) = 0 \tag{3.46}$$

is the function of x that solves the particular BVP of the above form with $f(x) = \delta(x - \bar{x})$. Here $\delta(x - \bar{x})$ is the delta function centered at $\bar{x}$.

We now find the explicit formula of $G(x;\bar{x})$. We obviously start with the definition of $G(x;\bar{x})$

$$G''(x;\bar{x}) = \delta(x - \bar{x}) \tag{3.47}$$
$$G(0;\bar{x}) = 0 \tag{3.48}$$
$$G(1;\bar{x}) = 0 \tag{3.49}$$

Integrating (3.47), we obtain

$$G'(x;\bar{x}) = \begin{cases} C_1 - 1, 0 \le x \le \bar{x} \\ C_1, \bar{x} \le x \le 1 \end{cases} \tag{3.50}$$

where C_1 is independent of x.

Integrating again, we obtain

$$G(x;\bar{x}) = \begin{cases} (C_1 - 1)x + C_2, 0 \le x \le \bar{x} \\ C_1 x + C_3, \bar{x} \le x \le 1 \end{cases} \tag{3.51}$$

where C_2, C_3 are independent of x.

Consider G at $x = 0$, at $x = \bar{x}$ and at $x = 1$ for (3.51), we get

$$C_2 = 0 \tag{3.52}$$
$$(C_1 - 1)\,\bar{x} + C_2 = C_1\bar{x} + C_3 \tag{3.53}$$
$$C_1 + C_3 = 0 \tag{3.54}$$

hence

$$C_1 = \bar{x} \tag{3.55}$$
$$C_2 = 0 \tag{3.56}$$
$$C_3 = -\bar{x} \tag{3.57}$$

Thus, we obtain the piecewise linear function $G(x;\bar{x})$ is given by

$$G(x;\bar{x}) = \begin{cases} (\bar{x}-1)\,x, 0 \le x \le \bar{x} \\ \bar{x}\,(x-1)\,, \bar{x} \le x \le 1 \end{cases} \tag{3.58}$$

Note that by linearity, if we replaced $f(x)$ with $c\delta(x-\bar{x})$ for any constant c, the solution to the BVP would be $cG(x;\bar{x})$. Moreover, any linear combination of Green's functions at different points $\bar{x}$ is a solution to the BVP with the corresponding linear combination of delta functions on the right-hand side. If the right-hand side is a sum of weighted delta functions at any number of points

$$f(x) = \sum_{k=1}^{n} c_k \delta(x - x_k) \tag{3.59}$$

then the solution to the BVP is

$$u(x) = \sum_{k=1}^{n} c_k G(x; x_k) \tag{3.60}$$

Now consider a general source $f(x)$ that is not a discrete sum of delta functions. We can view this as a continuous distribution of point sources, with $f(\bar{x})$ being a density function for the weight assigned to the delta function at $\bar{x}$, i.e.,

$$f(x) = \int_0^1 f(\bar{x})\,\delta(x-\bar{x})\,d\bar{x} \tag{3.61}$$

This suggests that the solution to $u''(x) = f(x)$ (still with $u(0) = u(1) = 1$) is

$$u(x) = \int_0^1 f(\bar{x})\,G(x;\bar{x})\,d\bar{x} \tag{3.62}$$

and indeed it is.

We are ready to return to the study of the infinity-norm stability of the finite difference method, which will be based on explicitly determining the inverse matrix for the matrix arising in this discretization. We will work with a slightly different formulation of the linear algebra problem in which we view U_0 and U_{N+1} as additional "unknowns" in the problem and introduce two new equations

in the system that simply state that $U_0 = 0$ and $U_{N+1} = 0$. The modified system has the form $AU = F$, where now

$$A = \frac{1}{h^2}\begin{bmatrix} h^2 & 0 & & & & & & \\ 1 & -2 & 1 & & & & & \\ & 1 & -2 & 1 & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & 1 & -2 & 1 & & \\ & & & & 1 & -2 & 1 & \\ & & & & & 0 & h^2 & \end{bmatrix} \tag{3.63}$$

$$U = \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ \vdots \\ U_{N-1} \\ U_N \\ U_{N+1} \end{bmatrix} \tag{3.64}$$

$$F = \begin{bmatrix} 0 \\ f(x_1) \\ f(x_1) \\ \vdots \\ f(x_1) \\ f(x_1) \\ 0 \end{bmatrix} \tag{3.65}$$

While we could work directly with the matrix A in (3.7), this reformulation has two advantages.

1. It separates the algebraic equations corresponding to the boundary conditions from the algebraic equations corresponding to the ODE $u''(x) = f(x)$. In the system (3.5), the first and last equations contains a mixture of ODE and boundary conditions. Separating these terms will make it clearer how the inverse of A relates to the Green's function representation of the true solution found above.

2. With Neumann boundary conditions $u'(0) = \sigma$ in place of $u(0) = \alpha$, the value U_0 really is unknown and our new formulation is easily extended to this case by replacing the first row of A with a discretization of this boundary condition.

Let B denote the $(N+2) \times (N+2)$ inverse of A from (3.63), $B = A^{-1}$. We will index the elements of B by B_{00} through $B_{N+1,N+1}$ in the obvious manner. The B_j denote the jth column of B for $j = 0, 1, \ldots, N+1$. Then

$$AB_j = e_j \tag{3.66}$$

where e_j is the jth column of the identity matrix. We can view this as a linear system to be solved for B_j. Note that this linear system is simply the

discretization of the BVP for a special choice of right-hand side F in which only one element of this vector is nonzero. This is exactly analogous to the manner in which the Green's function for the ODE is defined. The column B_0 to be a discrete approximation the function $G_0(0)$. In fact, the first ($j = 0$) column of B has elements obtained by simply evaluating G_0 at the grid points,

$$B_{i0} = G_0(x_i) = 1 - x_i \tag{3.67}$$

Since this is a linear function, the second difference operator applied at any point yields zero. Similarly, the last ($j = N + 1$) column of B has elements

$$B_{i,m+1} = G_1(x_i) = x_i \tag{3.68}$$

The interior columns ($1 \leq j \leq m$) correspond to the Green's function for zero boundary conditions and the source concentrated at a single point, since $F_j = 1$ and $F_i = 0$ for $i \neq j$. Note that this is a discrete version of $h\delta(x - x_j)$ since as a grid function F is nonzero over an interval of length h but has value 1 there, and hence total mash h. Thus we expect the column B_j will be a discrete approximation to the function $hG(x; x_j)$. In fact, it is easy to check that

$$B_{ij} = hG(x_i; x_j) = \begin{cases} h(x_j - 1)\, x_i, i = 1, 2, \ldots, j \\ h(x_i - 1)\, x_j, i = j, j + 1, \ldots, N \end{cases} \tag{3.69}$$

An arbitrary right-hand side F for the linear system can be written as

$$F = \sum_{j=1}^{N} f_j e_j \tag{3.70}$$

and the solution $U = BF$ is

$$U = \sum_{j=1}^{N} f_j B_j \tag{3.71}$$

with elements

$$U_i = h \sum_{j=1}^{N} f_j G(x_i; x_j) \tag{3.72}$$

In fact, something more is true: suppose we define a function $v(x)$ by

$$v(x) = h \sum_{j=1}^{N} f_j G(x; x_j) \tag{3.73}$$

Then $U_i = v(x_i)$ and $v(x)$ is the piecewise linear function that interpolates the numerical solution. This function $v(x)$ is the exact solution to the BVP

$$v''(x) = h \sum_{j=1}^{N} f(x_j)\, \delta(x - x_j) \tag{3.74}$$

$$v(0) = 0 \tag{3.75}$$

$$v(1) = 0 \tag{3.76}$$

Thus we can interpret the discrete solution as the exact solution to a modified problem in which the right-hand side $f(x)$ has been replaced by a finite sum of delta functions at the grid points x_j, with weights

$$hf(x_j) \approx \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} f(x)\, dx \tag{3.77}$$

To verify infinity-norm stability of the numerical method, we must show that $\|B\|_\infty$ is uniformly bounded as $h \to 0$. The infinity norm of the matrix is given by

$$\|B\|_\infty = \max_{0 \le j \le N+1} \sum_{j=0}^{N+1} |B_{ij}| \tag{3.78}$$

the maximum row sum of elements in the matrix. Note that the first row of B has $B_{00} = 1$ and B_{0j}=0 for $j > 0$, and hence row sum 1. Similarly, the last row contains all zeros except for $B_{N+1,N+1} = 1$. The intermediate rows are dense and the first and last elements (from columns B_0 and B_{N+1}) are bounded by 1. The other m elements of each of these rows are all bounded by h from (3.69), and hence

$$\sum_{j=0}^{N+1} |B_{ij}| \le 1 + 1 + Nh = 1 + 1 + \frac{N}{N+1} < 3 \tag{3.79}$$

Every row sum is bounded by 3 at most, and so

$$\left\|A^{-1}\right\|_\infty < 3, \forall h > 0 \tag{3.80}$$

and stability is proved.

The Green's function representation also clearly shows the effect that each local truncation error has on the global error. Recall that the global error E is related to the local truncation error by $AE = -\tau$. This continues to hold for our reformulation of the problem, where we now defined τ_0 and τ_{N+1} as the errors in the imposed boundary conditions, which are typically zero for the Dirichlet problem. Solving this system gives $E = -B\tau$. If we did make an error in one of the boundary conditions, setting F_0 to $\alpha + \tau_0$, the effect on the global error would be $\tau_0 B_0$. The effect of this error is thus nonzero across the interval, decreasing linearly from the boundary where the error is made at the other end. Each truncation error τ_i for $1 \le 1 \le N$ in the difference approximation $u''(x_i) = f(x_i)$ likewise has an effect on the global error everywhere, although the effect is largest at the grid point x_i, where it is $hG(x_i; x_i)\tau_i$, and decays linearly toward each end. Note that since $\tau_i = O\left(h^2\right)$, the contribution of this error to the global error each each point is only $O\left(h^3\right)$. However, since all N local errors contribute to the global error at each point, the total effect is $O\left(Nh^3\right) = O\left(h^2\right)$.

Finally, observe that we have also worked out the inverse of the original matrix A defined in (3.7). Because the first row of B consists of zeros beyond the first element, and the last row consists of zeros, except for the last element, it is easy to check that the inverse of the $N \times N$ matrix from (3.7) is the $N \times N$ central block of B consisting of B_{11} through B_{NN}. The infinity-norm of this matrix is bounded by 1 for all h, so our original formulation is stable as well.

4 Alternative Approach

This addition section contains an alternative approach. The finite difference formula represented here is more complicated than (2.35). Therefore, the author chooses it as a additional approach for further purposes. Although this approach is more complicated to deal with, but it is very prospective, especially combining with the grid mapping function.

We now describe this approach without the powerful grid mapping function. First of all, we need an appropriate finite difference approximations for $u(x_i)$. To do this, we have to use Taylor series expansion.

For each $i = 1, 2, \ldots, N+1$, we expand $u(x_{i-1})$ and $u(x_{i+1})$ in a Taylor series about the point x_i, e.g.,

$$u(x_{i-2}) = u(x_i) - (h_{i-1} + h_i)\, u'(x_i) + \frac{1}{2}(h_{i-1} + h_i)^2 u''(x_i) \tag{4.1}$$

$$- \frac{1}{6}(h_{i-1} + h_i)^3 u'''(x_i) + \frac{1}{24}(h_{i-1} + h_i)^4 u^{(4)}(x_i) \tag{4.2}$$

$$- \frac{1}{120}(h_{i-1} + h_i)^5 u^{(5)}(x_i) + O\left(h^6\right) \tag{4.3}$$

$$u(x_{i-1}) = u(x_i) - h_i u'(x_i) + \frac{1}{2}h_i^2 u''(x_i) - \frac{1}{6}h_i^3 u'''(x_i) \tag{4.4}$$

$$+ \frac{1}{24}h_i^4 u^{(4)}(x_i) - \frac{1}{120}h_i^5 u^{(5)}(x_i) + O\left(h^6\right) \tag{4.5}$$

$$u(x_{i+1}) = u(x_i) + h_{i+1} u'(x_i) + \frac{1}{2}h_{i+1}^2 u''(x_i) + \frac{1}{6}h_{i+1}^3 u'''(x_i) \tag{4.6}$$

$$+ \frac{1}{24}h_{i+1}^4 u^{(4)}(x_i) + \frac{1}{120}h_{i+1}^5 u^{(5)}(x_i) + O\left(h^6\right) \tag{4.7}$$

$$u(x_{i+2}) = u(x_i) + (h_{i+1} + h_{i+2})\, u'(x_i) + \frac{1}{2}(h_{i+1} + h_{i+2})^2 u''(x_i) \tag{4.8}$$

$$+ \frac{1}{6}(h_{i+1} + h_{i+2})^3 u'''(x_i) + \frac{1}{24}(h_{i+1} + h_{i+2})^4 u^{(4)}(x_i) \tag{4.9}$$

$$+ \frac{1}{120}(h_{i+1} + h_{i+2})^5 u^{(5)}(x_i) + O\left(h^6\right) \tag{4.10}$$

We now consider the following quantity

$$\alpha u(x_{i-2}) + \beta u(x_{i-1}) + \gamma u(x_{i+1}) + \delta u(x_{i+2}) \tag{4.11}$$

where $\alpha, \beta, \gamma, \delta$ are appropriate coefficients which we find now.

Clearly, we want the coefficients of $u'(x_i), u'''(x_i), u^{(5)}(x_i)$ in (4.11) equal to zero. This leads us to the following system of equations respect to variables $\alpha, \beta, \gamma, \delta$.

$$-(h_{i-1} + h_i)\,\alpha - h_i\beta + h_{i+1}\gamma + (h_{i+1} + h_{i+2})\,\delta = 0 \tag{4.12}$$

$$-(h_{i-1} + h_i)^3\alpha - h_i^3\beta + h_{i+1}^3\gamma + (h_{i+1} + h_{i+2})^3\delta = 0 \tag{4.13}$$

$$-(h_{i-1} + h_i)^5\alpha - h_i^5\beta + h_{i+1}^5\gamma + (h_{i+1} + h_{i+2})^5\delta = 0 \tag{4.14}$$

A solution of (4.12)-(4.14), which we consider the best one, is

$$\alpha = \frac{\left(h_i^2 - h_{i+1}^2\right)\left(h_{i+1}^2 - (h_i + h_{i+1})^2\right)\left((h_i + h_{i+1})^2 - h_i^2\right)}{h_{i-1} + h_i} \tag{4.15}$$

$$\beta = -\frac{\left((h_{i-1} + h_i)^2 - h_{i+1}^2\right)\left(h_{i+1}^2 - (h_i + h_{i+1})^2\right)\left((h_i + h_{i+1})^2 - (h_{i-1} + h_i)^2\right)}{h_i} \tag{4.16}$$

$$\gamma = -\frac{\left((h_{i-1} + h_i)^2 - h_i^2\right)\left(h_i^2 - (h_i + h_{i+1})^2\right)\left((h_i + h_{i+1})^2 - (h_{i-1} + h_i)^2\right)}{h_{i+1}} \tag{4.17}$$

$$\delta = \frac{\left(h_i^2 - h_{i+1}^2\right)\left(h_{i+1}^2 - (h_{i-1} + h_i)^2\right)\left((h_{i-1} + h_i)^2 - h_i^2\right)}{h_i + h_{i+1}} \tag{4.18}$$

Then (4.11) becomes

$$\alpha u\left(x_{i-2}\right) + \beta u\left(x_{i-1}\right) + \gamma u\left(x_{i+1}\right) + \delta u\left(x_{i+2}\right) \tag{4.19}$$

$$= (\alpha + \beta + \gamma + \delta)\, u\left(x_i\right) \tag{4.20}$$

$$+ \frac{1}{2}\left[\alpha(h_{i-1} + h_i)^2 + \beta h_i^2 + \gamma h_{i+1}^2 + \delta(h_{i+1} + h_{i+2})^2\right] u''\left(x_i\right) \tag{4.21}$$

$$+ \frac{1}{24}\left[\alpha(h_{i-1} + h_i)^4 + \beta h_i^4 + \gamma h_{i+1}^4 + \delta(h_{i+1} + h_{i+2})^4\right] u^{(4)}\left(x_i\right) + O\left(h^6\right) \tag{4.22}$$

equivalently,

$$u''\left(x_i\right) = \frac{2\left[\alpha u\left(x_{i-2}\right) + \beta u\left(x_{i-1}\right) + \gamma u\left(x_{i+1}\right) + \delta u\left(x_{i+2}\right) - (\alpha + \beta + \gamma + \delta)\, u\left(x_i\right)\right]}{\alpha(h_{i-1} + h_i)^2 + \beta h_i^2 + \gamma h_{i+1}^2 + \delta(h_{i+1} + h_{i+2})^2} \tag{4.23}$$

$$- \frac{1}{12}\frac{\alpha(h_{i-1} + h_i)^4 + \beta h_i^4 + \gamma h_{i+1}^4 + \delta(h_{i+1} + h_{i+2})^4}{\alpha(h_{i-1} + h_i)^2 + \beta h_i^2 + \gamma h_{i+1}^2 + \delta(h_{i+1} + h_{i+2})^2} u^{(4)}\left(x_i\right) \tag{4.24}$$

$$+ \frac{1}{\alpha(h_{i-1} + h_i)^2 + \beta h_i^2 + \gamma h_{i+1}^2 + \delta(h_{i+1} + h_{i+2})^2} O\left(h^6\right) \tag{4.25}$$

Taking

$$\bar{D}_2 u\left(x\right) := \frac{2\left[\alpha u\left(x_{i-2}\right) + \beta u\left(x_{i-1}\right) + \gamma u\left(x_{i+1}\right) + \delta u\left(x_{i+2}\right) - (\alpha + \beta + \gamma + \delta)\, u\left(x_i\right)\right]}{\alpha(h_{i-1} + h_i)^2 + \beta h_i^2 + \gamma h_{i+1}^2 + \delta(h_{i+1} + h_{i+2})^2} \tag{4.26}$$

Remark 4.1. If $u^{(4)} = 0$ or very small at the grid points x_i. We may obtain a sixth-order accuracy approximation for the discrete second derivative u''. We can even combine this idea with grid mapping function in the first approach to obtain many powerful features. Because its formula is so complicated, this topic will be discussed later.

5 Appendices

5.1 Some Matrix Norm Inequalities

See [6].

For matrix $A \in \mathbb{R}^{m\times n}$ of rank r, the following inequalities hold

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{r}\|A\|_2 \tag{5.1}$$

$$\|A\|_F \leq \|A\|_* \leq \sqrt{r}\|A\|_F \tag{5.2}$$

$$\|A\|_{\max} \leq \|A\|_2 \leq \sqrt{mn}\|A\|_{\max} \tag{5.3}$$

$$\frac{1}{\sqrt{n}}\|A\|_\infty \leq \|A\|_2 \leq \sqrt{m}\|A\|_\infty \tag{5.4}$$

$$\frac{1}{\sqrt{n}}\|A\|_1 \leq \|A\|_2 \leq \sqrt{n}\|A\|_1 \tag{5.5}$$

$$\|A\|_2 \leq \sqrt{\|A\|_1\|A\|_\infty} \tag{5.6}$$

5.2 Inversion of a Tridiagonal Matrices

See [2], [3], [4] and [5] for references.

6 Matlab Implementation

6.1 Problems

Let $\Omega = (0,1) \subset \mathbb{R}$ and $f \in L^2(\Omega)$.

$$-u_{xx} = f(x) \text{ in } \Omega \tag{6.1}$$

Problem 6.1 (Dirichlet boundary condition).

1. *Solve equation* (6.1) *with uniform mesh subject to a Dirichlet boundary condition*

$$u(0) = 1, u(1) = 2 \tag{6.2}$$

2. *Solve equation* (6.1) *subject to a Dirichlet boundary condition*

$$u(0) = 0, u(1) = 0 \tag{6.3}$$

with nonuniform mesh $x_i = 1 - \cos\frac{\pi i}{2N}, i = 0, 1, \ldots, N$.

Problem 6.2 (Dirichlet-Neumann boundary condition). *Solve equation* (6.1) *with uniform mesh subject to a Dirichlet Neumann boundary condition*

$$u'(0) = 0, u(1) = 3 \tag{6.4}$$

Problem 6.3 (Neumann boundary condition). *Solve equation* (6.1) *with nonuniform mesh subject to a Neumann boundary condition*

$$u'(0) = 0, u'(1) = 0 \tag{6.5}$$

with condition $\int_0^1 f(x)\,dx = 0$.

6.2 Matlab Implementation for Problem 6.1.a

6.2.1 Matlab Scripts

exact_solution.m

```
function u_ex=exact_solution(x,k);
if (k==1)
    u_ex=x^2+1;
end
if(k==2)
    u_ex=(x+1)^3*(x-1)+2;
end
```

functionf.m

```
function f=functionf(x,k);
if (k==1)
    f=-2;
end
if (k==2)
    f=- 6*(x + 1)^2 - 3*(2*x + 2)*(x - 1);
end
```

main.m

```
% Solve equation -u''(x)=f(x)
% with the Dirichlet boundary condition
clear all
close all
format long
clc
%% Initial informations
ax=0.0;
bx=1.0;
alpha=1;
beta=2;
cases=2;
N=2;% number of mesh points of first mesh
number_mesh=8;
number_mesh_point=zeros(number_mesh,1);
norm_max=zeros(number_mesh,1);
norm_l2=zeros(number_mesh,1);
norm_maxh1=zeros(number_mesh,1);
norm_h1=zeros(number_mesh,1);
%% Solve discrite solution and refine mesh
for inumber_mesh=1:number_mesh
    fprintf('N=%d\n',N);
    number_mesh_point(inumber_mesh)=N;
    delta_x=(bx-ax)/N;
%% Create mesh point
    x=zeros(N+1,1);
    for i=1:N+1
```

```
        x(i)=(i-1)*delta_x;
    end
 %% Create matrix A
    A=sparse(N-1,N-1);
    for i=1:N-1
        if (i==1)
            A(i,i)=2;
            A(i,i+1)=-1;
        elseif(i==N-1)
            A(i,i-1)=-1;
            A(i,i)=2;
        else
            A(i,i-1)=-1;
            A(i,i+1)=-1;
            A(i,i)=2;
        end
    end
    A=A/((delta_x)^2);
%% Create vector b
    b=zeros(N-1,1);
    b(1)=functionf(delta_x,cases)+alpha/(delta_x^2);
    for i=2:N-2
        b(i)=functionf(i*delta_x,cases);
    end
    b(N-1)=functionf((N-1)*delta_x,cases)+beta/(delta_x^2);
%% Solve discrete solution
    u=A\b;
%% Get exact solution
    u_ex=zeros(N+1,1);
    for i=1:N+1
        u_ex(i)=exact_solution((i-1)*delta_x,cases);
    end
%% Create discrete solution with boundary
    u_dis=zeros(N+1,1);
    for i=1:N+1
        if (i==1)
            u_dis(i)=alpha;
        elseif(i==N+1)
            u_dis(i)=beta;
        else
            u_dis(i)=u(i-1,1);
        end
    end
%% Calculate the error on L^infinity
    norm_max(inumber_mesh)=0.0;
    for i=1:N+1
        if (abs(u_dis(i)-u_ex(i)) > norm_max(inumber_mesh))
            norm_max(inumber_mesh)=abs(u_dis(i)-u_ex(i));
        end
    end
```

```
    fprintf('error on L^infinity: %df\n',norm_max(inumber_mesh));
%%  Calculate the error on L^2

    norm_l2(inumber_mesh)=0;
    for i=1:N+1
        norm_l2(inumber_mesh)= norm_l2(inumber_mesh) ...
                              +(u_dis(i)-u_ex(i))^2*delta_x;
    end
    norm_l2(inumber_mesh)=(norm_l2(inumber_mesh))^(1/2);
    fprintf('error on L^2: %df\n',norm_l2(inumber_mesh));
%% Calculate the error on maxH1
    norm_maxh1(inumber_mesh)=0;
    for i=1:N
        if (abs(((u_dis(i+1)-u_ex(i+1)) ...
        -(u_dis(i)-u_ex(i)))/delta_x) > norm_maxh1(inumber_mesh))
            norm_maxh1(inumber_mesh)=abs(((u_dis(i+1) ...
            -u_ex(i+1))-(u_dis(i)-u_ex(i)))/delta_x);
        end
    end
    fprintf('error on maxH1: %df\n',norm_maxh1(inumber_mesh));
%% Calculate the error on H1
    norm_h1(inumber_mesh)=0;
    for i=1:N
        norm_h1(inumber_mesh)=norm_h1(inumber_mesh) ...
        +(((u_dis(i+1)-u_ex(i+1)) ...
        -(u_dis(i)-u_ex(i)))/delta_x)^2*delta_x;
    end
    norm_h1(inumber_mesh)=(norm_h1(inumber_mesh))^(1/2);
    fprintf('error on H1: %df\n',norm_h1(inumber_mesh));
%% Figure exact and dicrete solutions
    figure
    plot(x,u_ex,'blue', x,u_dis,'red');
    xlabel('x');ylabel('value');
    str=sprintf('Comparison between exact and discrete ...
        solutions with N=%d',N);
    title(str);
    %ylim([-0.02 0.12]);
    legend('Exact solution','Discrete solution');
    print('-r300','-djpeg');
%% Refine mesh (increse mesh point)
    N=2*N;
%% Seperation
    disp(' ');
end
%% Figure for errors respect to number of mesh point
figure
plot(log(number_mesh_point), -log(norm_max),'blue', ...
     log(number_mesh_point), -log(norm_l2), 'red',...
     log(number_mesh_point), -log(norm_maxh1), 'cyan', ...
     log(number_mesh_point), -log(norm_h1), 'magenta',...
```

```
    log(number_mesh_point), 2*log(number_mesh_point),'black');
xlabel('Log(MeshPoint)');ylabel('-Log(Error)');
title('Errors');
legend('norm_max','norm_l2','norm_maxh1','norm_h1','2x', ...
       'Location','NorthEastOutside');
print('-r300','-djpeg');
```

6.2.2 Results

Case 1.

$$u_{ex} = x^2 + 1 \tag{6.6}$$

$$f = -2 \tag{6.7}$$

Run these scripts for `cases=1`, MATLAB returns

```
N=2
error on L^infinity: 5.000000e-01f
error on L^2: 3.535534e-01f
error on maxH1: 1f
error on H1: 1f

N=4
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=4
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=8
error on L^infinity: 2.220446e-16f
error on L^2: 1.110223e-16f
error on maxH1: 1.776357e-15f
error on H1: 8.881784e-16f

N=16
error on L^infinity: 2.220446e-16f
error on L^2: 1.468687e-16f
error on maxH1: 3.552714e-15f
error on H1: 1.776357e-15f

N=32
error on L^infinity: 4.884981e-15f
error on L^2: 3.275863e-15f
error on maxH1: 2.842171e-14f
error on H1: 1.164835e-14f
```

```
N=64
error on L^infinity: 9.992007e-15f
error on L^2: 6.245560e-15f
error on maxH1: 4.263256e-14f
error on H1: 2.537145e-14f

N=128
error on L^infinity: 1.731948e-14f
error on L^2: 1.365385e-14f
error on maxH1: 1.136868e-13f
error on H1: 5.281489e-14f

N=256
error on L^infinity: 5.462297e-14f
error on L^2: 3.285722e-14f
error on maxH1: 3.979039e-13f
error on H1: 1.512305e-13f
```

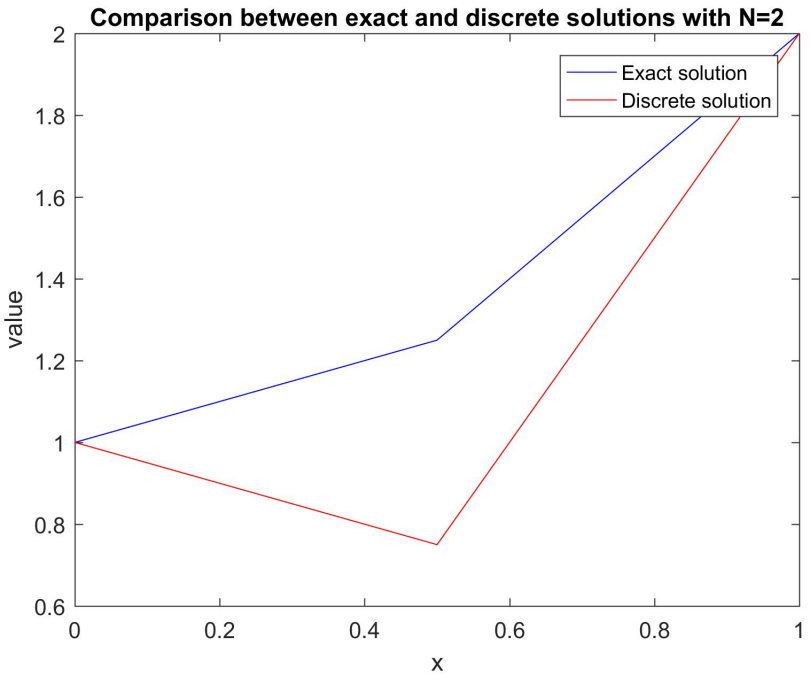

Figure 2: Numerical Solutions: Problem 6.1.a, Test 1, $N = 2$.

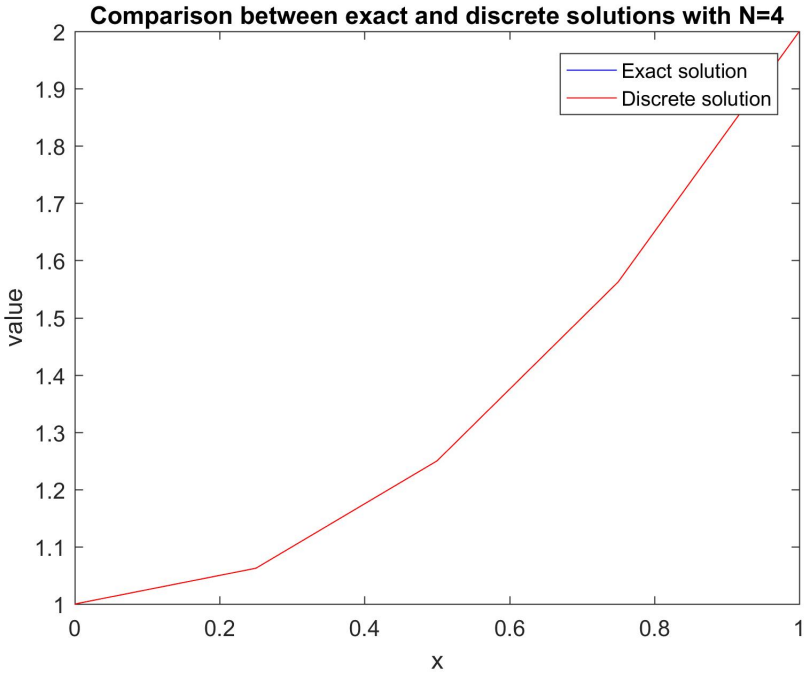

Figure 3: Numerical Solutions: Problem 6.1.a, Test 1, $N = 4$.

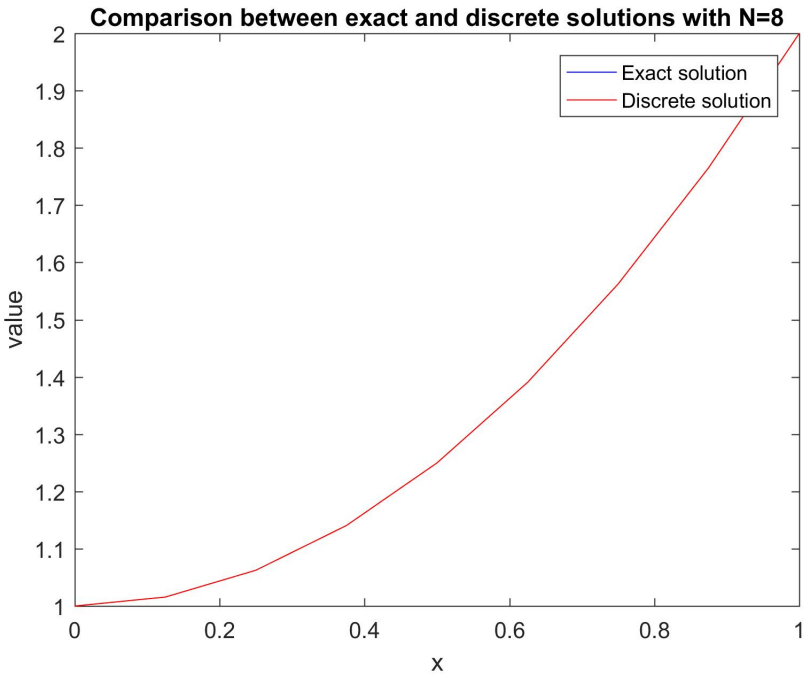

Figure 4: Numerical Solutions: Problem 6.1.a, Test 1, $N = 8$.

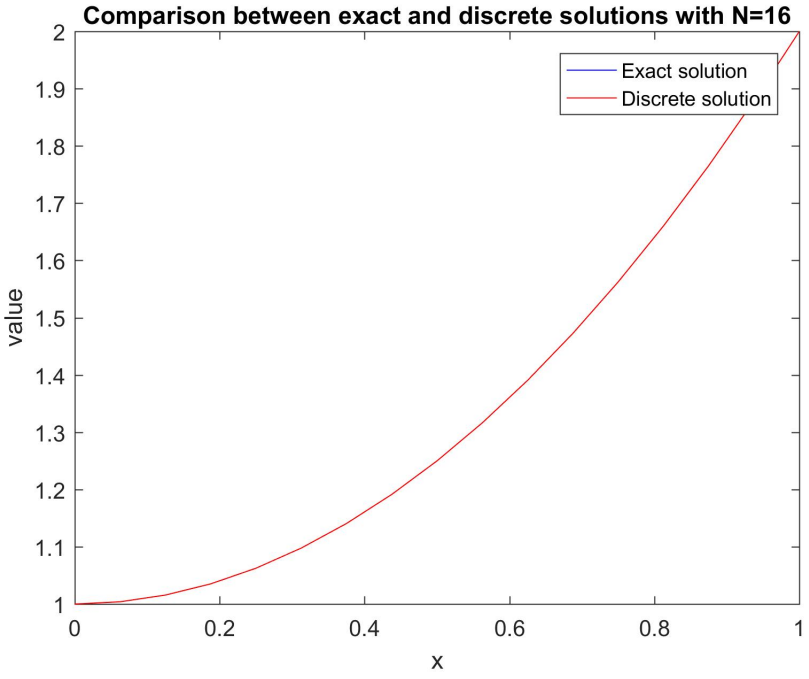

Figure 5: Numerical Solutions: Problem 6.1.a, Test 1, $N = 16$.

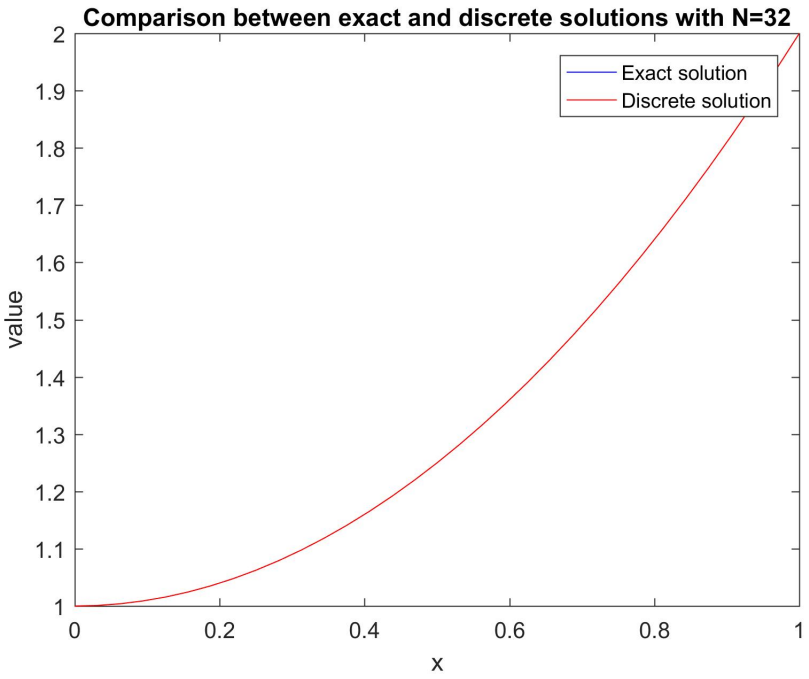

Figure 6: Numerical Solutions: Problem 6.1.a, Test 1, $N = 32$.

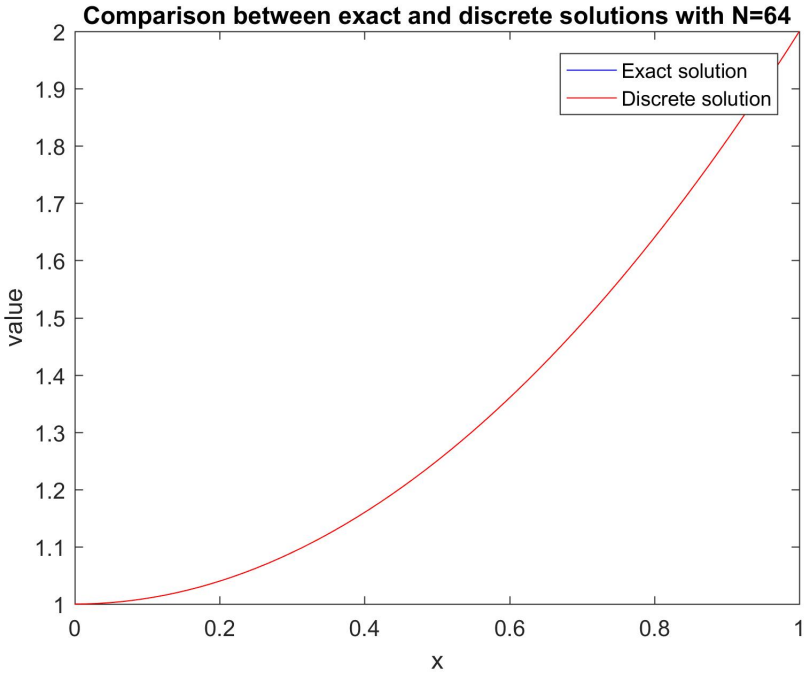

Figure 7: Numerical Solutions: Problem 6.1.a, Test 1, $N = 64$.

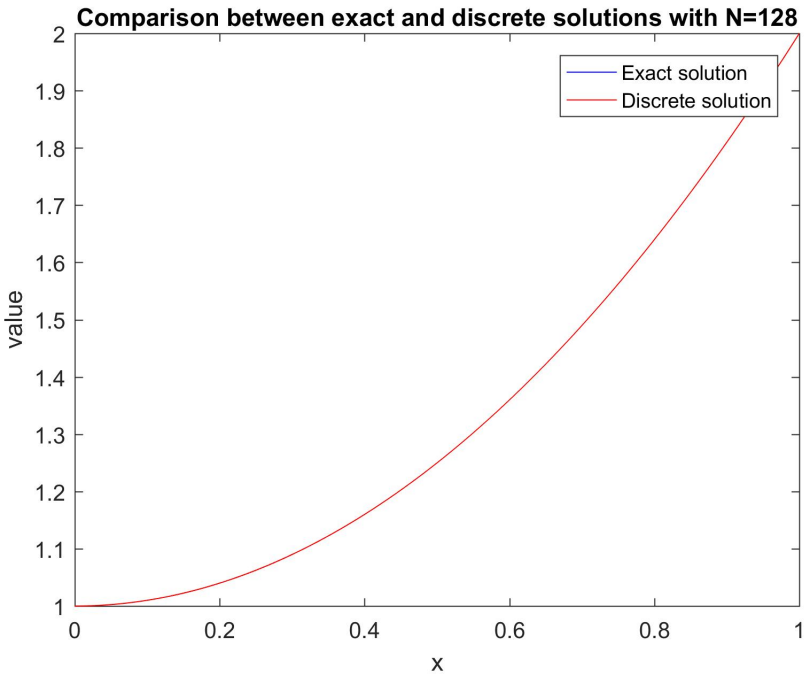

Figure 8: Numerical Solutions: Problem 6.1.a, Test 1, $N = 128$.

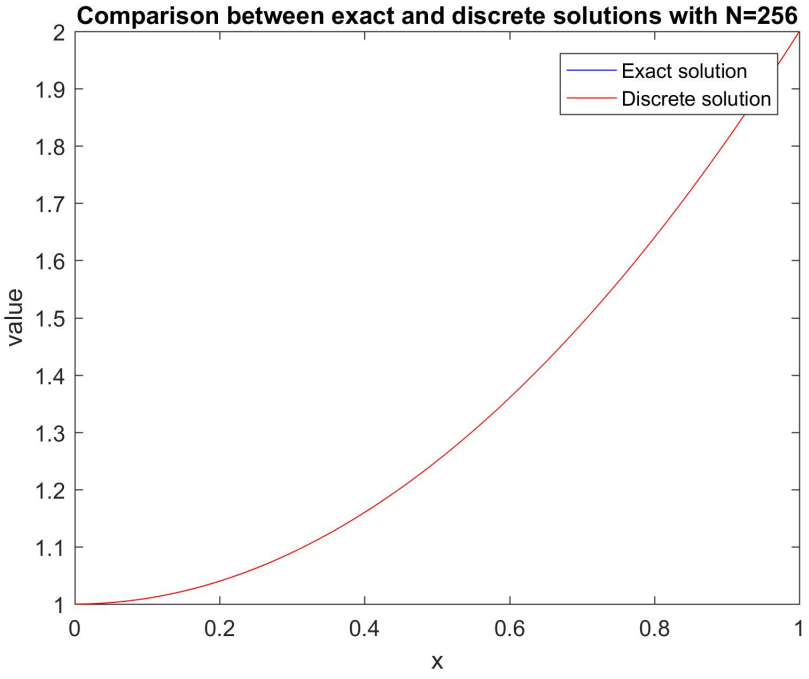

Figure 9: Numerical Solutions: Problem 6.1.a, Test 1, $N = 256$.

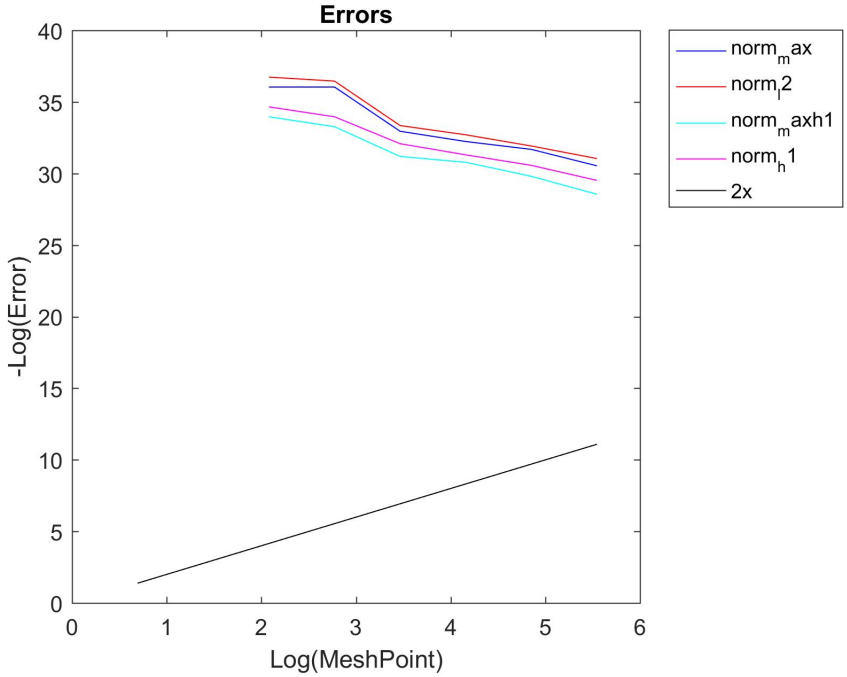

Figure 10: Errors on a log-log scale: Problem 6.1.a, Test 1.

Remark 6.4. We not that the errors in is 0 with $N = 4$. Although the errors with larger N are very low, it appears that the errors increase as N increases and the convergence rate is obviously not $O\left(h^2\right)$.

Case 2.

$$u_{ex} = (x+1)^3 (x-1) + 2 \tag{6.8}$$

$$f = -6(x+1)^2 - 3\,(2x+2)\,(x-1) \tag{6.9}$$

Run these scripts for `cases=2`, MATLAB returns

```
N=2
error on L^infinity: 4.375000e-01f
error on L^2: 3.093592e-01f
error on maxH1: 8.750000e-01f
error on H1: 8.750000e-01f

N=4
error on L^infinity: 1.562500e-02f
error on L^2: 1.138858e-02f
error on maxH1: 4.687500e-02f
error on H1: 3.493856e-02f

N=8
error on L^infinity: 3.906250e-03f
```

```
error on L^2: 2.852373e-03f
error on maxH1: 1.367188e-02f
error on H1: 8.950343e-03f

N=16
error on L^infinity: 9.765625e-04f
error on L^2: 7.131750e-04f
error on maxH1: 3.662109e-03f
error on H1: 2.250865e-03f

N=32
error on L^infinity: 2.441406e-04f
error on L^2: 1.782950e-04f
error on maxH1: 9.460449e-04f
error on H1: 5.635433e-04f

N=64
error on L^infinity: 6.103516e-05f
error on L^2: 4.457377e-05f
error on maxH1: 2.403259e-04f
error on H1: 1.409374e-04f

N=128
error on L^infinity: 1.525879e-05f
error on L^2: 1.114344e-05f
error on maxH1: 6.055832e-05f
error on H1: 3.523759e-05f

N=256
error on L^infinity: 3.814697e-06f
error on L^2: 2.785861e-06f
error on maxH1: 1.519918e-05f
error on H1: 8.809599e-06f
```

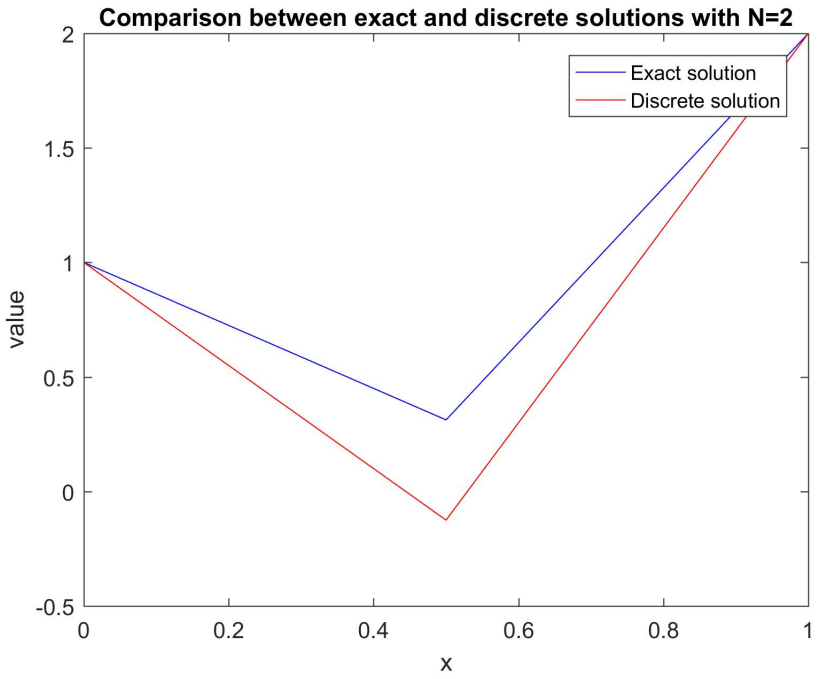

Figure 11: Numerical Solutions: Problem 6.1.a, Test 2, $N = 2$.

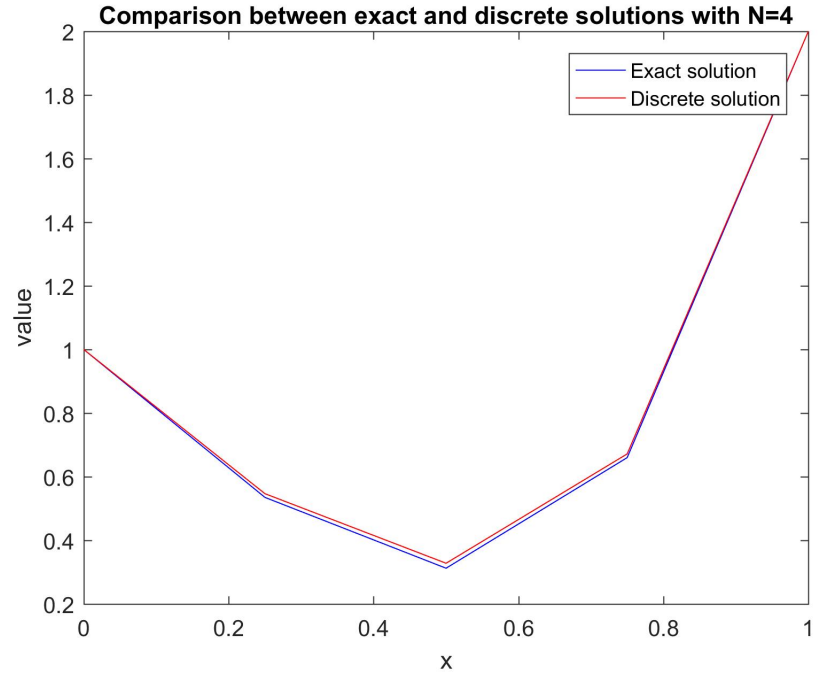

Figure 12: Numerical Solutions: Problem 6.1.a, Test 2, $N = 4$.

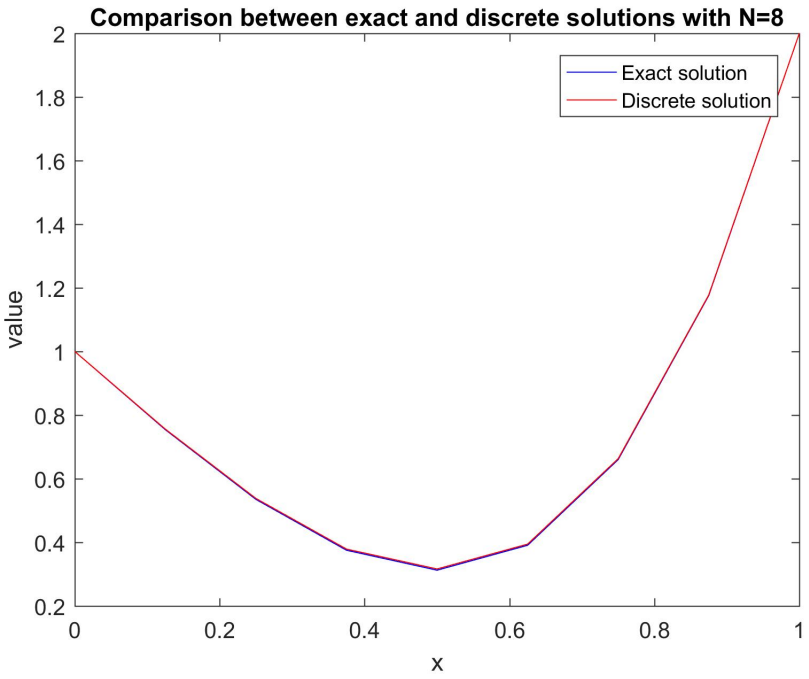

Figure 13: Numerical Solutions: Problem 6.1.a, Test 2, $N = 8$.

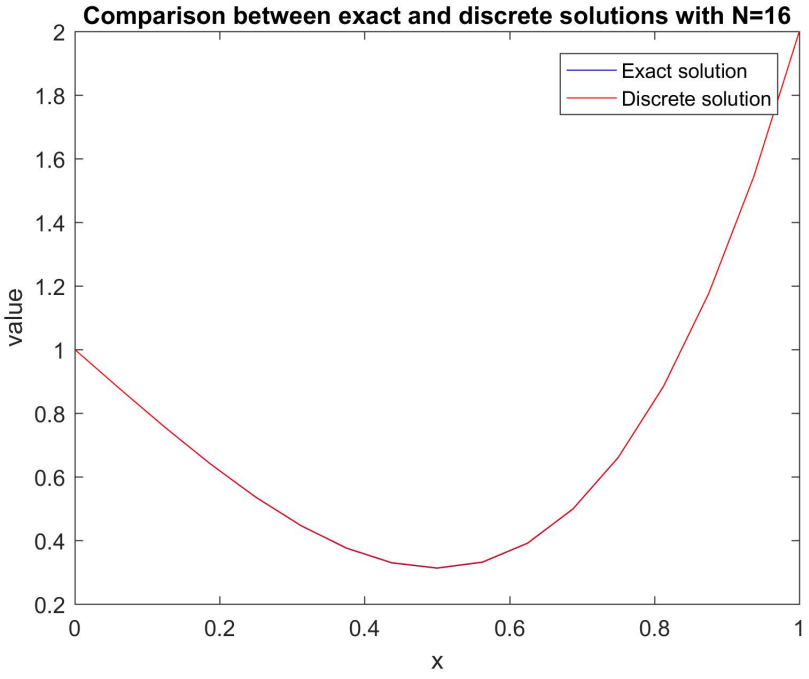

Figure 14: Numerical Solutions: Problem 6.1.a, Test 2, $N = 16$.

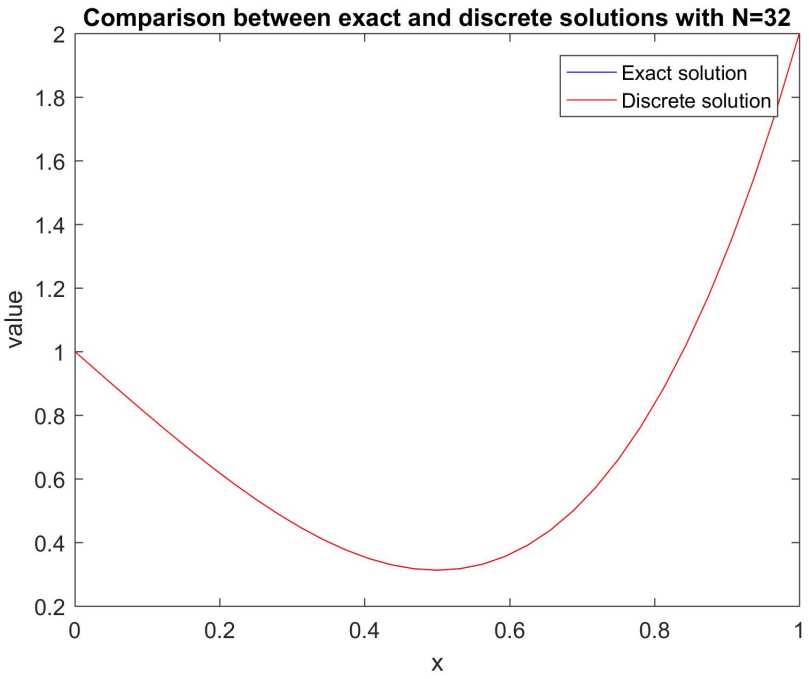

Figure 15: Numerical Solutions: Problem 6.1.a, Test 2, $N = 32$.

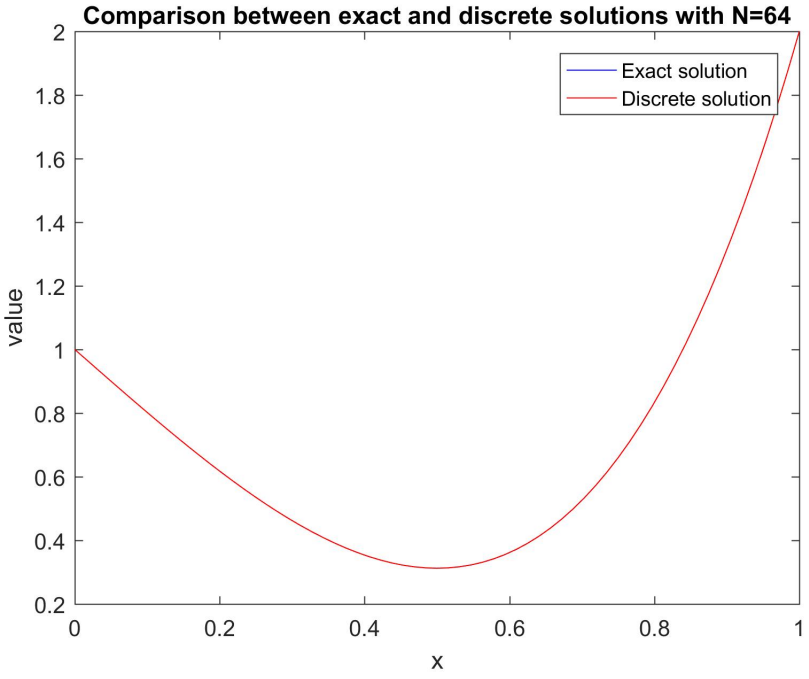

Figure 16: Numerical Solutions: Problem 6.1.a, Test 2, $N = 64$.

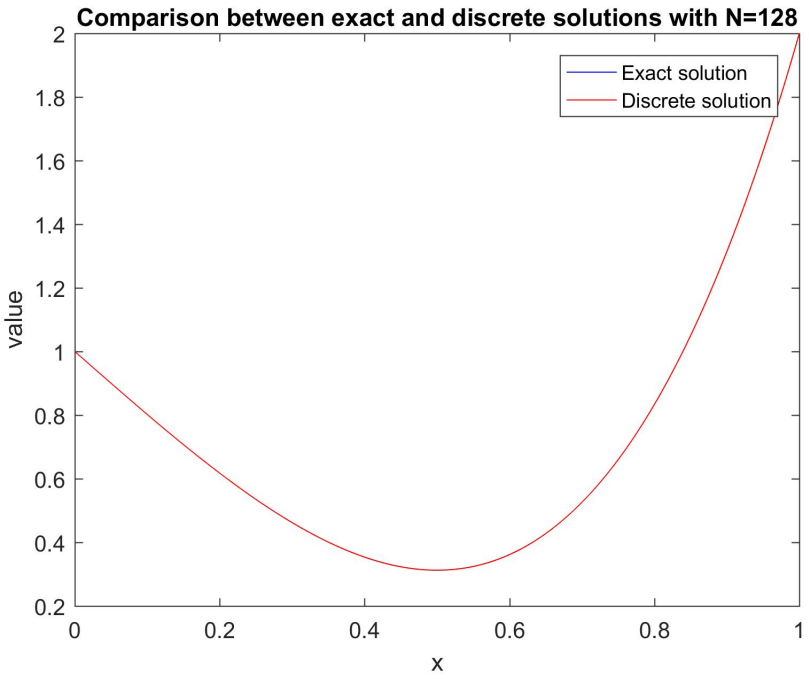

Figure 17: Numerical Solutions: Problem 6.1.a, Test 2, $N = 128$.

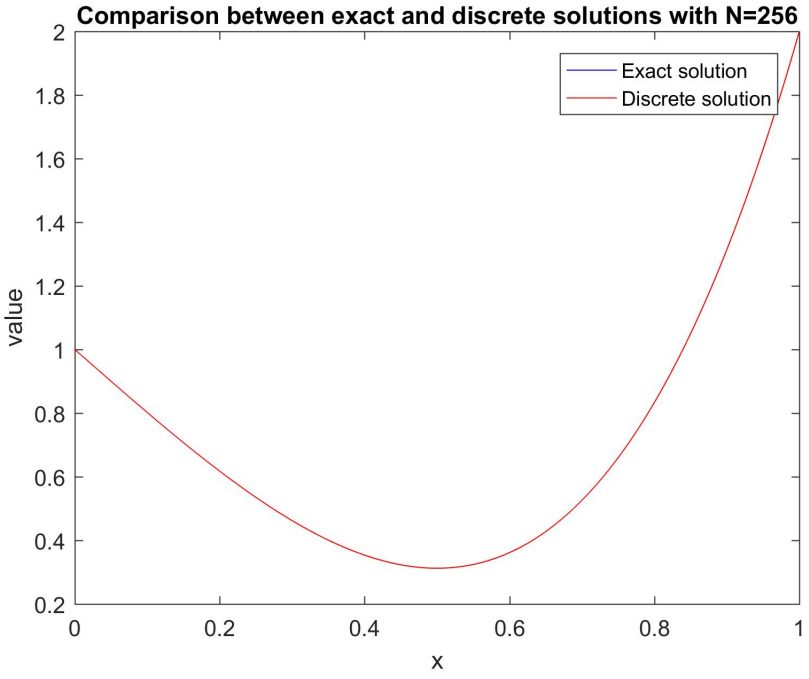

Figure 18: Numerical Solutions: Problem 6.1.a, Test 2, $N = 256$.

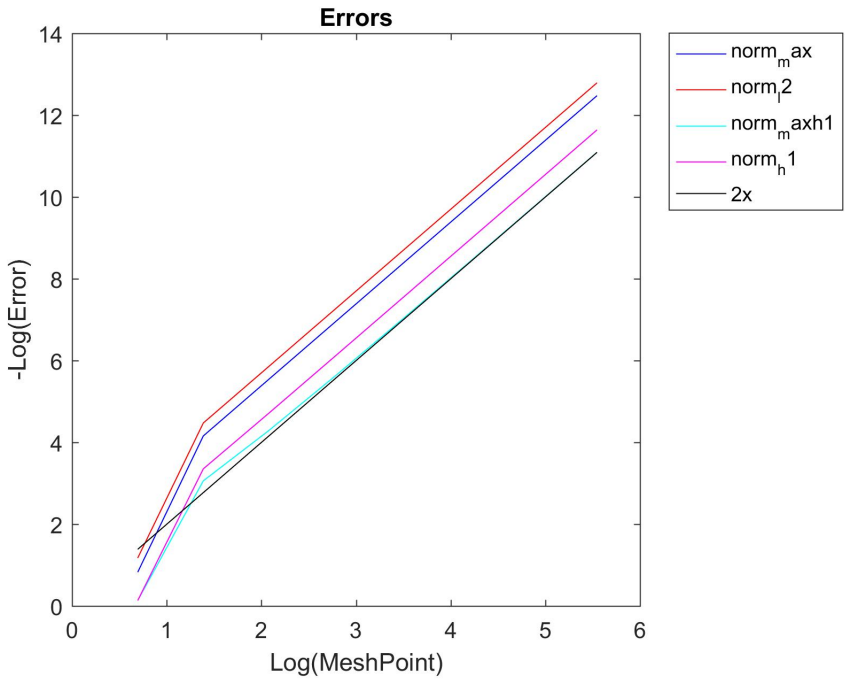

Figure 19: Errors on a log-log scale: Problem 6.1.a, Test 2.

Remark 6.5. We see that the errors converge to 0 as N change from 2 to 8 with rate faster than $O\left(h^2\right)$, but after that, the errors converge with quadratic rate $O\left(h^2\right)$.

6.3 Matlab Implementation for Problem 6.1.b

6.3.1 Matlab Scripts

alpha.m

```
function a=alpha(i,N)
i=i+1;
a=-2/(d1(i,N)^2+d2(i,N)*d1(i,N));
```

alpha.m

```
function b=beta(i,N)
i=i+1;
b=2/(d2(i,N)*d1(i,N));
```

d1.m

```
function d=d1(i,N)
d=cos((pi*(i-2))/(2*N))-cos((pi*(i-1))/(2*N));
% d=cos((pi*(i-1))/(2*N))-cos((pi*(i))/(2*N));
```

d2.m

```
function d=d2(i,N)
d=cos((pi*(i-1))/(2*N))-cos((pi*(i))/(2*N));
% d=cos((pi*(i))/(2*N))-cos((pi*(i+1))/(2*N));
```

exact_solution.m

```
function u_ex=exact_solution(x,k);
if (k==1)
    u_ex=x*(1-x);
end
if(k==2)
    u_ex=x^3*(1-x);
end
```

functionf.m

```
function f=functionf(x,k);
if (k==1)
    f=2;
end
if (k==2)
    f=-6*x+12*x^2;
end
```

gam.m

```
function g=gam(i,N)
i=i+1;
g=-2/(d2(i,N)^2+d2(i,N)*d1(i,N));
```

main.m

```
% Solve equation -u''(x)=f(x) with the
% Dirichlet boundary condition
clear all
close all
clc
format long
%% Initial informations
ax=0.0;
bx=1.0;
cases=2;
N=2;% number of mesh points of first mesh
number_mesh=8;
number_mesh_point=zeros(number_mesh,1);
norm_max=zeros(number_mesh,1);
norm_l2=zeros(number_mesh,1);
norm_maxh1=zeros(number_mesh,1);
norm_h1=zeros(number_mesh,1);
%% Solve discrite solution and refine mesh

for inumber_mesh=1:number_mesh
    fprintf('N=%d\n',N);
```

```
    number_mesh_point(inumber_mesh)=N;
%     delta_x=(bx-ax)/N;
%% Create mesh point
    x=zeros(N+1,1);
    for i=1:N+1
        x(i)=1-cos((pi*(i-1))/(2*N));
    end
%     N=length(x)
 %% Create matrix A
    A=sparse(N-1,N-1);
    for i=1:N-1
        if (i==1)
            A(i,i)=beta(i,N);
            A(i,i+1)=gam(i,N);
        elseif(i==N-1)
            A(i,i-1)=alpha(i,N);
            A(i,i)=beta(i,N);
        else
            A(i,i-1)=alpha(i,N);
            A(i,i+1)=gam(i,N);
            A(i,i)=beta(i,N);
        end
    end
%% Create vector b
    b=zeros(N-1,1);
    for i=1:N-1
        b(i)=functionf(x(i+1),cases);
    end
%% Solve discrete solution
    u=A\b;
%% Get exact solution
    u_ex=zeros(N+1,1);
    for i=1:N+1
        u_ex(i)=exact_solution(x(i),cases);
    end
%% Create discrete solution with boundary
    u_dis=zeros(N+1,1);
    for i=1:N+1
        if (i==1)
            u_dis(i)=0;
        elseif(i==N+1)
            u_dis(i)=0;
        else
            u_dis(i)=u(i-1,1);
        end
    end
%% Calculate the error on L^infinity
    norm_max(inumber_mesh)=0.0;
    for i=1:N+1
        if (abs(u_dis(i)-u_ex(i)) > norm_max(inumber_mesh))
```

```
            norm_max(inumber_mesh)=abs(u_dis(i)-u_ex(i));
        end
    end
    fprintf('error on L^infinity: %df\n',norm_max(inumber_mesh));
%%  Calculate the error on L^2
    norm_l2(inumber_mesh)=0;
    for i=1:N+1
        norm_l2(inumber_mesh)=norm_l2(inumber_mesh) ...
        +(u_dis(i)-u_ex(i))^2*d2(i,N);
    end
    norm_l2(inumber_mesh)=(norm_l2(inumber_mesh))^(1/2);
    fprintf('error on L^2: %df\n',norm_l2(inumber_mesh));
%% Calculate the error on maxH1
    norm_maxh1(inumber_mesh)=0;
    for i=1:N
        if (abs(((u_dis(i+1)-u_ex(i+1)) ...
        -(u_dis(i)-u_ex(i)))/d2(i,N)) > norm_maxh1(inumber_mesh))
            norm_maxh1(inumber_mesh)=abs(((u_dis(i+1)-u_ex(i+1))- ...
            (u_dis(i)-u_ex(i)))/d2(i,N));
        end
    end
    fprintf('error on maxH1: %df\n',norm_maxh1(inumber_mesh));
%% Calculate the error on H1
    norm_h1(inumber_mesh)=0;
    for i=1:N
        norm_h1(inumber_mesh)=norm_h1(inumber_mesh) ...
        +(((u_dis(i+1)-u_ex(i+1))-(u_dis(i)-u_ex(i)))/...
         d2(i,N))^2*d2(i,N);
    end
    norm_h1(inumber_mesh)=(norm_h1(inumber_mesh))^(1/2);
    fprintf('error on H1: %df\n',norm_h1(inumber_mesh));
%% Figure exact and dicrete solutions
    figure
    plot(x,u_ex,'blue', x,u_dis,'red');
    xlabel('x');ylabel('value');
    str=sprintf('Comparison between exact and discrete ...
                solutions with N=%d',N);
    title(str);
    %ylim([-0.02 0.12]);
    legend('Exact solution','Discrete solution');
%% Refine mesh (increse mesh point)
    N=2*N;
%% Seperation
    disp(' ');
end
%% Figure for errors respect to number of mesh point
figure
plot(log(number_mesh_point), -log(norm_max),'blue', ...
     log(number_mesh_point), -log(norm_l2), 'red', ...
     log(number_mesh_point), -log(norm_maxh1), 'cyan', ...
```

```
    log(number_mesh_point), -log(norm_h1), 'magenta',...
    log(number_mesh_point), 2*log(number_mesh_point),'black');
xlabel('Log(MeshPoint)');ylabel('-Log(Error)');
title('Errors');
legend('norm_max','norm_l2','norm_maxh1','norm_h1','2x', ...
       'Location','NorthEastOutside');
print('-r300','-djpeg');
```

6.3.2 Results

Case 1.

$$u_{ex} = x\,(1-x) \tag{6.10}$$

$$f = 2 \tag{6.11}$$

Run these scripts for `cases=1`, MATLAB returns

```
N=2
error on L^infinity: 1.110223e-16f
error on L^2: 1.043777e-16f
error on maxH1: 1.895269e-16f
error on H1: 1.219784e-16f

N=4
error on L^infinity: 1.110223e-16f
error on L^2: 9.048348e-17f
error on maxH1: 3.200999e-16f
error on H1: 1.875324e-16f

N=8
error on L^infinity: 1.387779e-16f
error on L^2: 1.032771e-16f
error on maxH1: 4.463645e-16f
error on H1: 2.435311e-16f

N=16
error on L^infinity: 1.387779e-16f
error on L^2: 9.719438e-17f
error on maxH1: 1.080763e-15f
error on H1: 4.473998e-16f

N=32
error on L^infinity: 6.383782e-16f
error on L^2: 3.956974e-16f
error on maxH1: 4.328783e-15f
error on H1: 1.923950e-15f

N=64
error on L^infinity: 9.159340e-16f
error on L^2: 5.158811e-16f
error on maxH1: 9.767630e-15f
```

```
error on H1: 4.470624e-15f

N=128
error on L^infinity: 4.440892e-15f
error on L^2: 2.897418e-15f
error on maxH1: 3.015346e-14f
error on H1: 1.145961e-14f

N=256
error on L^infinity: 2.359224e-14f
error on L^2: 1.432204e-14f
error on maxH1: 1.823436e-13f
error on H1: 6.368180e-14f
```

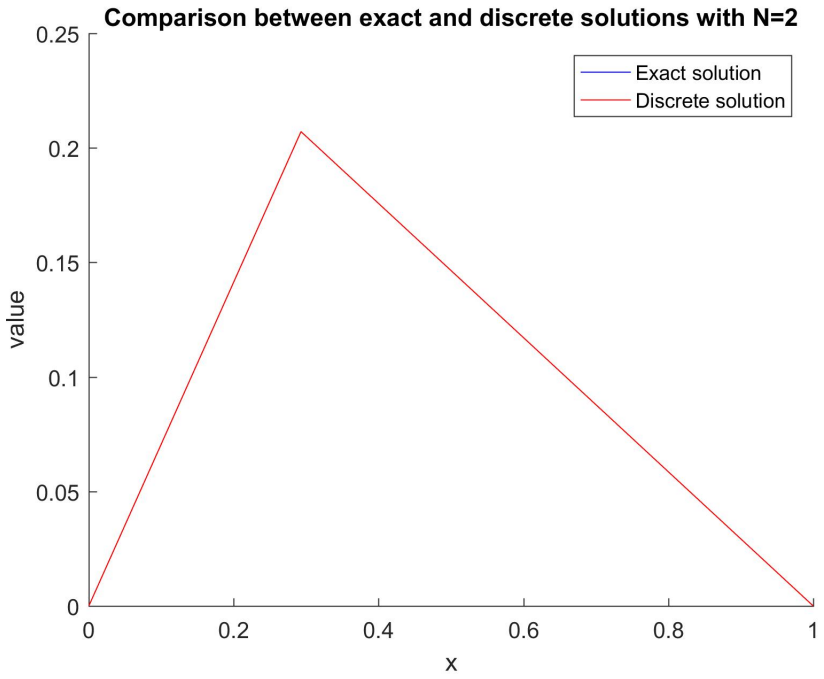

Figure 20: Numerical Solutions: Problem 6.1.b, Test 1, $N = 2$.

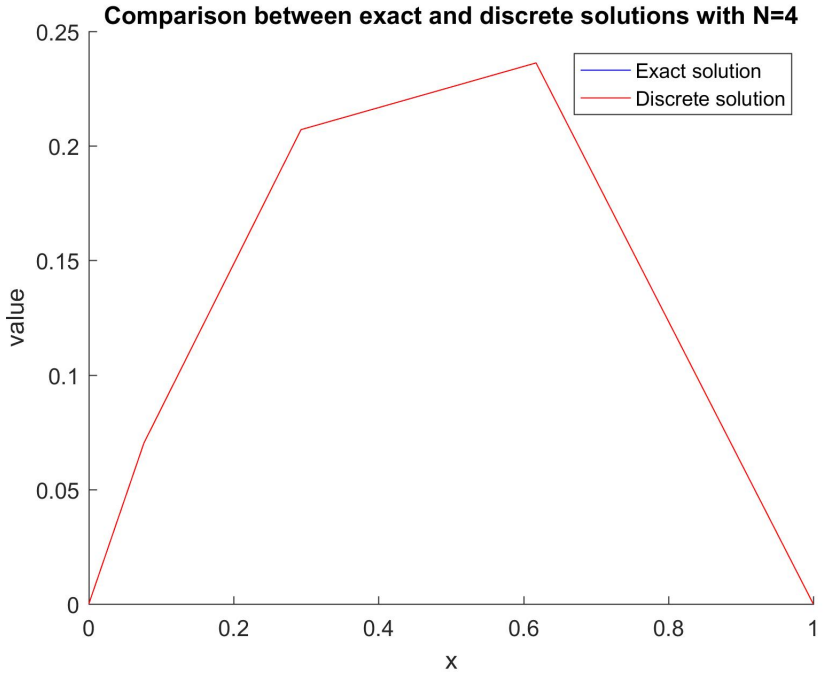

Figure 21: Numerical Solutions: Problem 6.1.b, Test 1, $N = 4$.

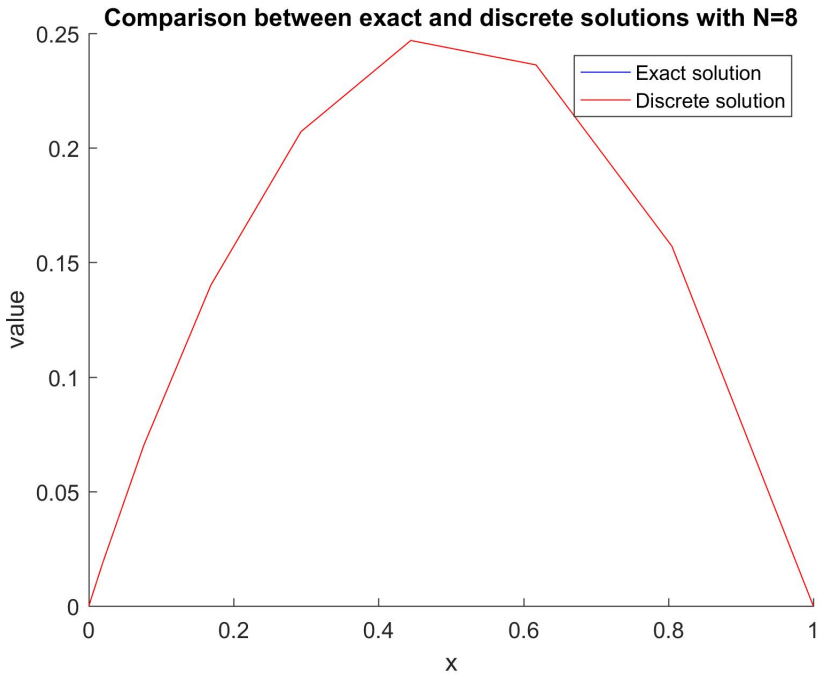

Figure 22: Numerical Solutions: Problem 6.1.b, Test 1, $N = 8$.

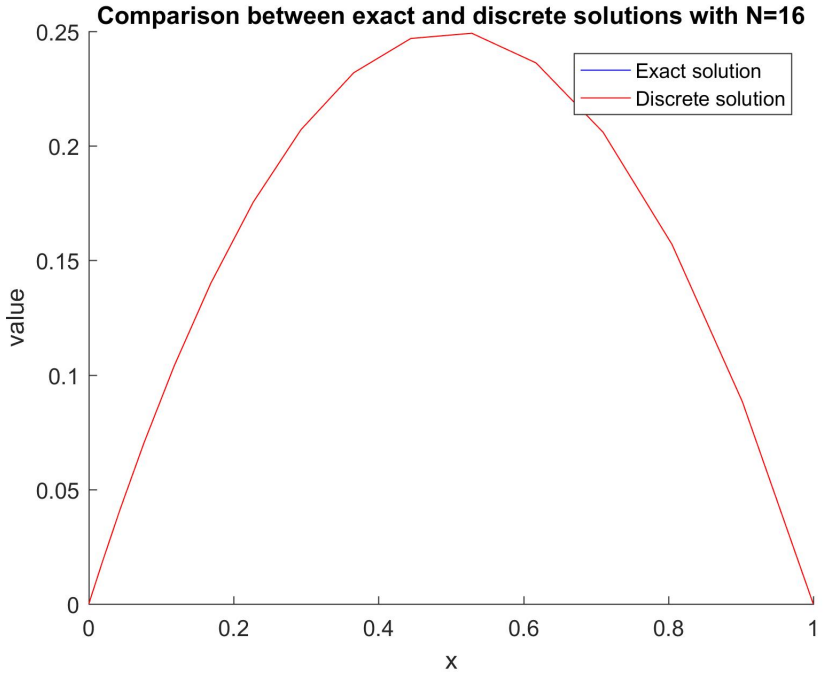

Figure 23: Numerical Solutions: Problem 6.1.b, Test 1, $N = 16$.

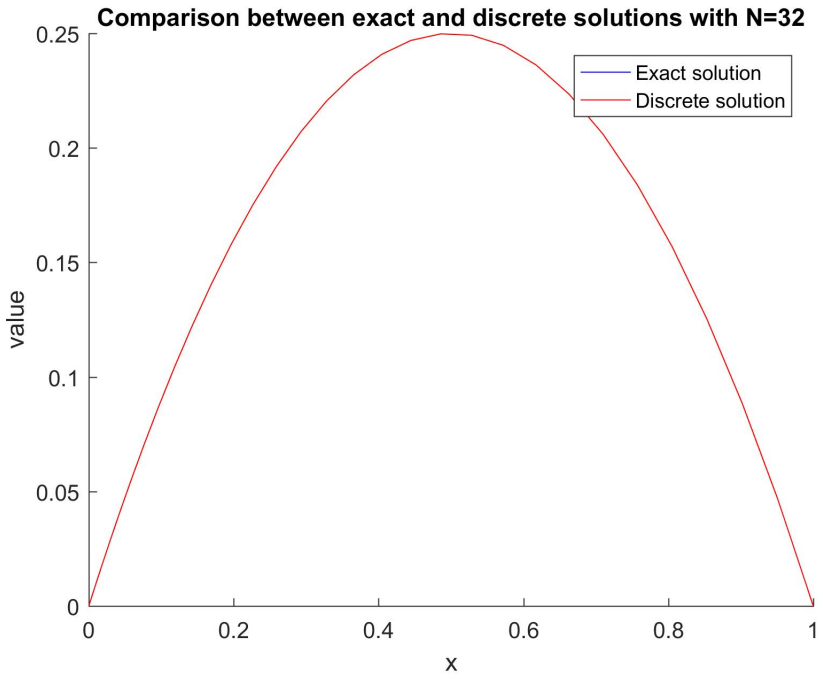

Figure 24: Numerical Solutions: Problem 6.1.b, Test 1, $N = 32$.

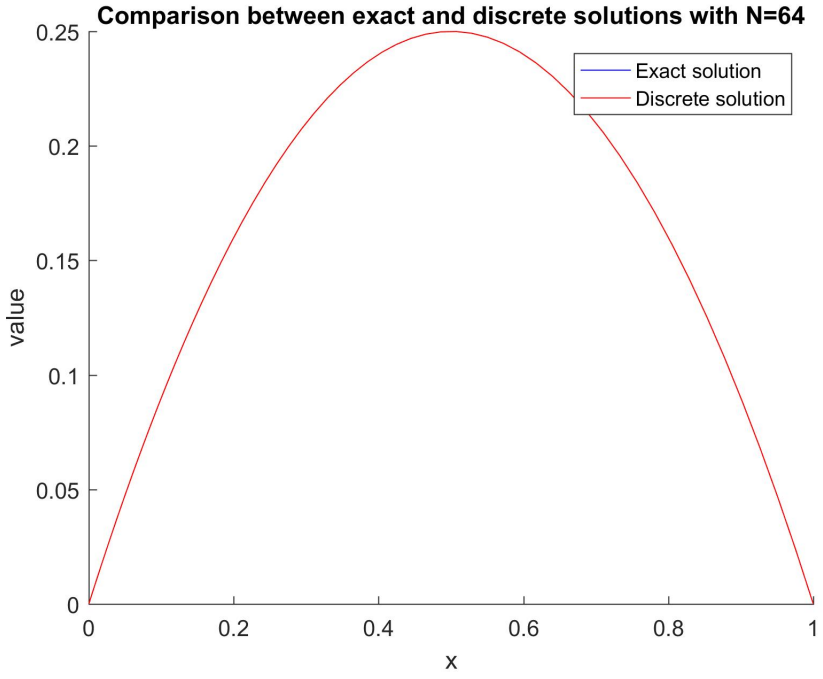

Figure 25: Numerical Solutions: Problem 6.1.b, Test 1, $N = 64$.

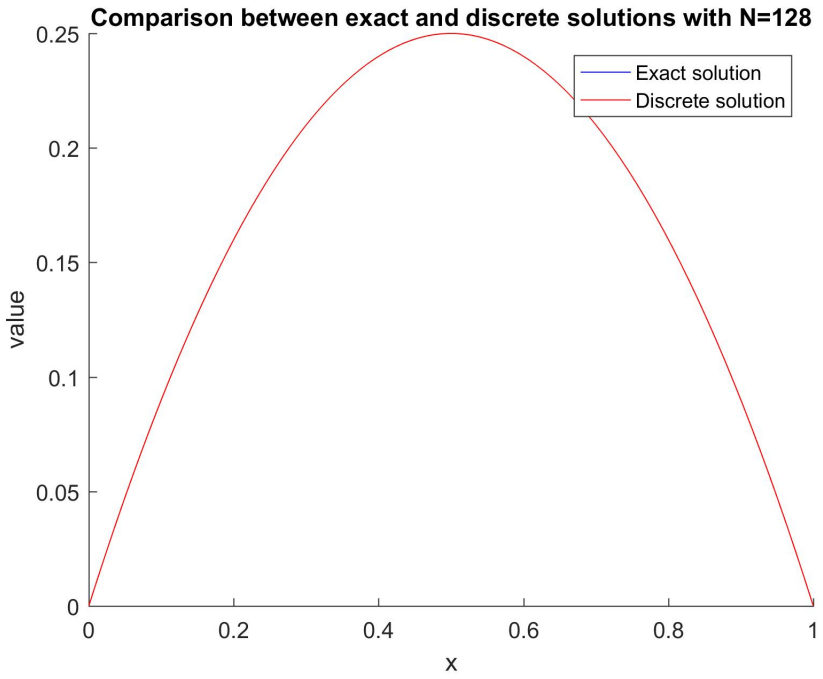

Figure 26: Numerical Solutions: Problem 6.1.b, Test 1, $N = 128$.

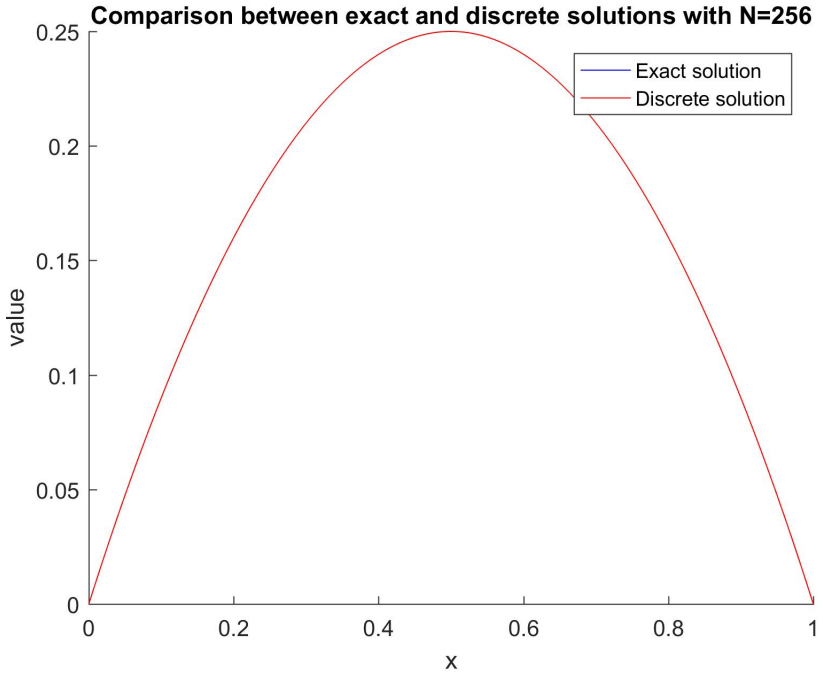

Figure 27: Numerical Solutions: Problem 6.1.b, Test 1, $N = 256$.

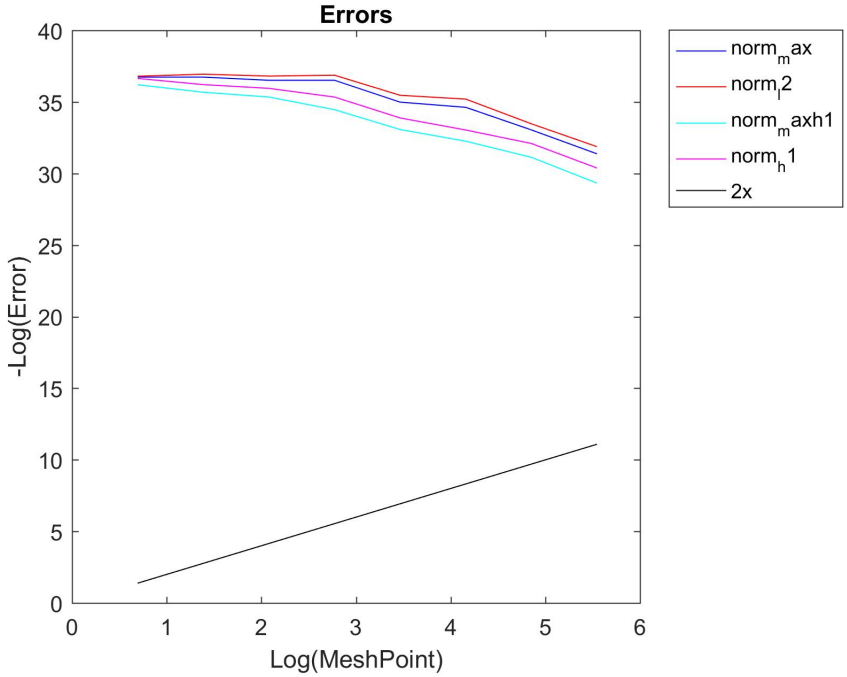

Figure 28: Errors on a log-log scale: Problem 6.1.b, Test 1.

Remark 6.6. We note that the errors decrease as N increases but the convergence rate is not $O\left(h^2\right)$.

Case 2.

$$u_{ex} = x^3 (1 - x) \tag{6.12}$$

$$f = -6x + 12x^2 \tag{6.13}$$

Run these scripts for `cases=1`, Matlab returns

```
N=2
error on L^infinity: 9.314575e-02f
error on L^2: 7.832593e-02f
error on maxH1: 3.180195e-01f
error on H1: 2.046755e-01f

N=4
error on L^infinity: 4.116905e-02f
error on L^2: 2.993502e-02f
error on maxH1: 1.075799e-01f
error on H1: 8.701833e-02f

N=8
error on L^infinity: 1.014554e-02f
error on L^2: 7.370940e-03f
```

```
error on maxH1: 3.650932e-02f
error on H1: 2.281117e-02f

N=16
error on L^infinity: 2.548025e-03f
error on L^2: 1.819903e-03f
error on maxH1: 1.023731e-02f
error on H1: 5.760522e-03f

N=32
error on L^infinity: 6.403917e-04f
error on L^2: 4.519468e-04f
error on maxH1: 2.688919e-03f
error on H1: 1.443629e-03f

N=64
error on L^infinity: 1.600663e-04f
error on L^2: 1.126018e-04f
error on maxH1: 6.877542e-04f
error on H1: 3.611244e-04f

N=128
error on L^infinity: 4.002887e-05f
error on L^2: 2.810203e-05f
error on maxH1: 1.738339e-04f
error on H1: 9.029464e-05f

N=256
error on L^infinity: 1.000710e-05f
error on L^2: 7.019436e-06f
error on maxH1: 4.369246e-05f
error on H1: 2.257450e-05f
```

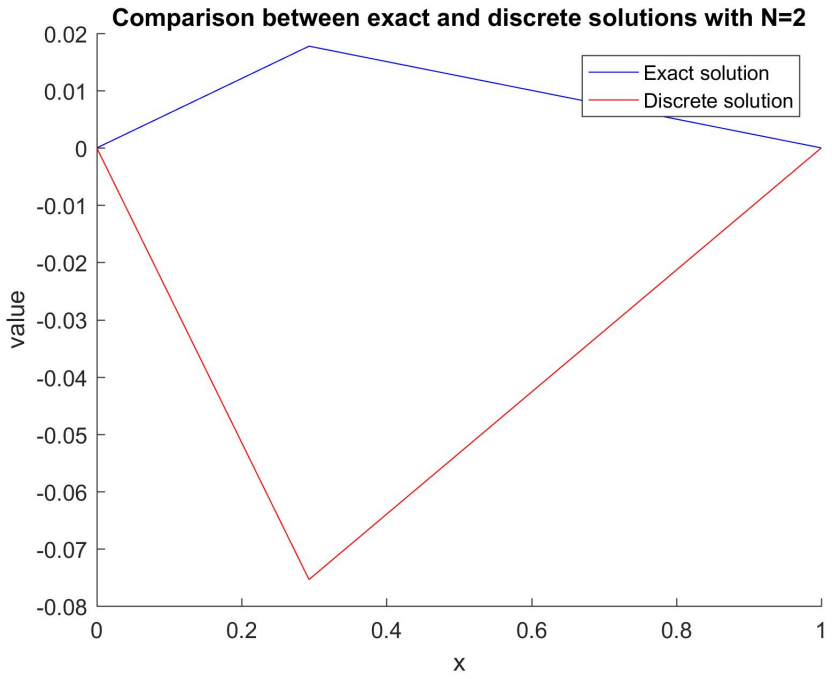

Figure 29: Numerical Solutions: Problem 6.1.b, Test 2, $N = 2$.

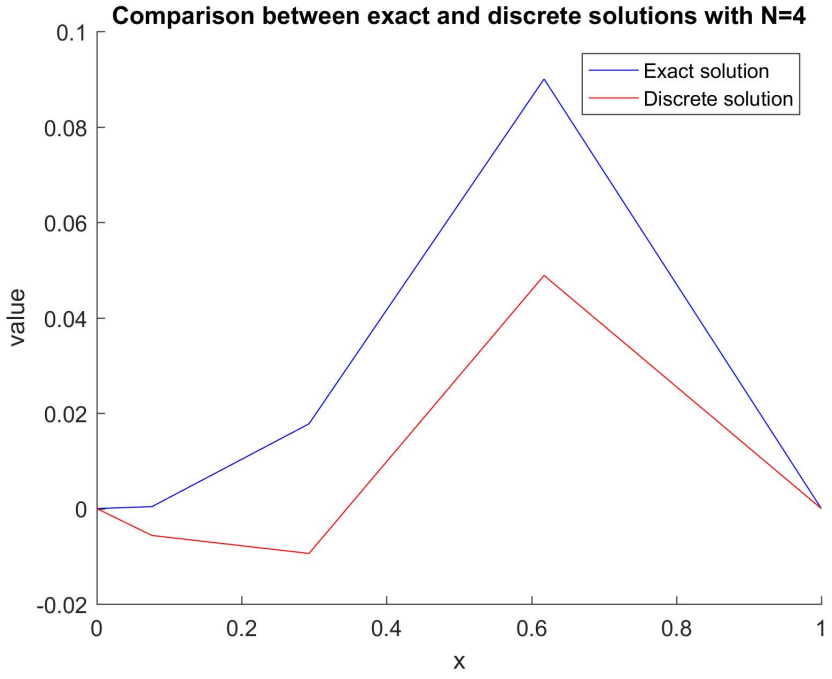

Figure 30: Numerical Solutions: Problem 6.1.b, Test 2, $N = 4$.

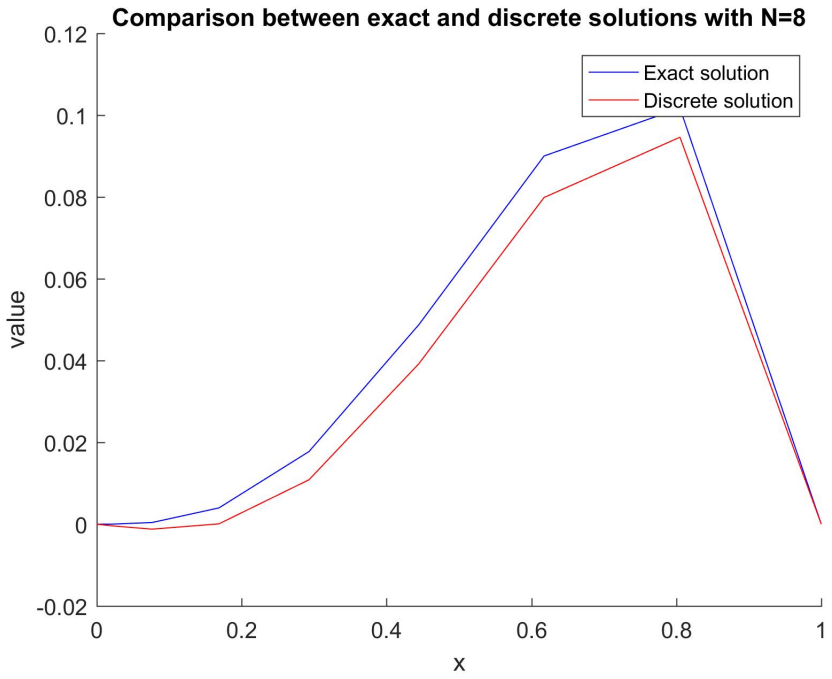

Figure 31: Numerical Solutions: Problem 6.1.b, Test 2, $N = 8$.

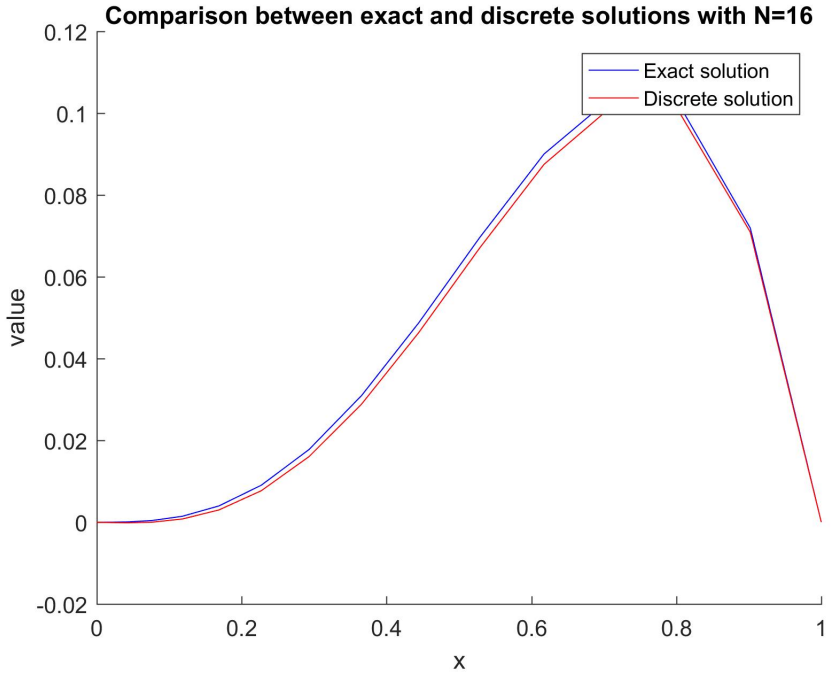

Figure 32: Numerical Solutions: Problem 6.1.b, Test 2, $N = 16$.

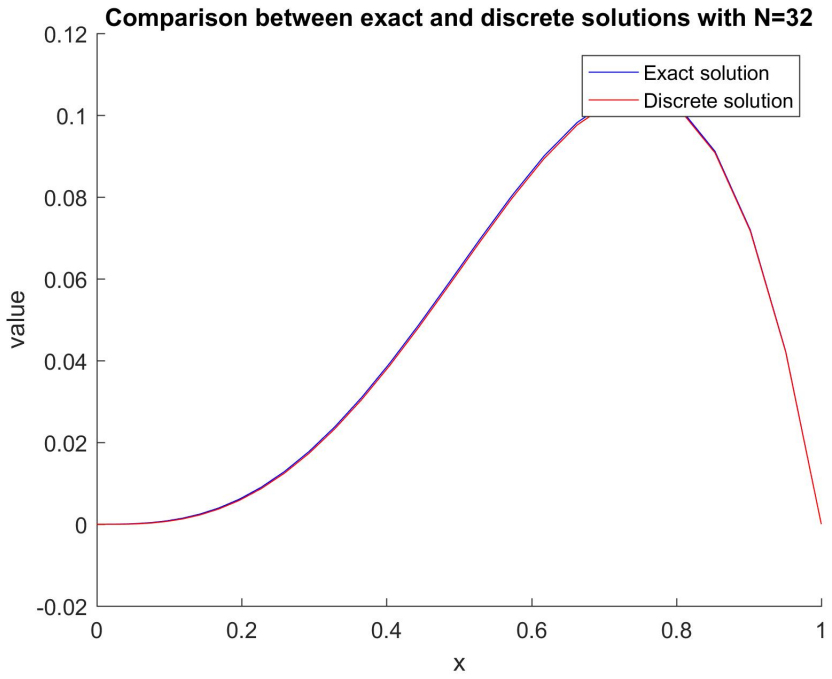

Figure 33: Numerical Solutions: Problem 6.1.b, Test 2, $N = 32$.

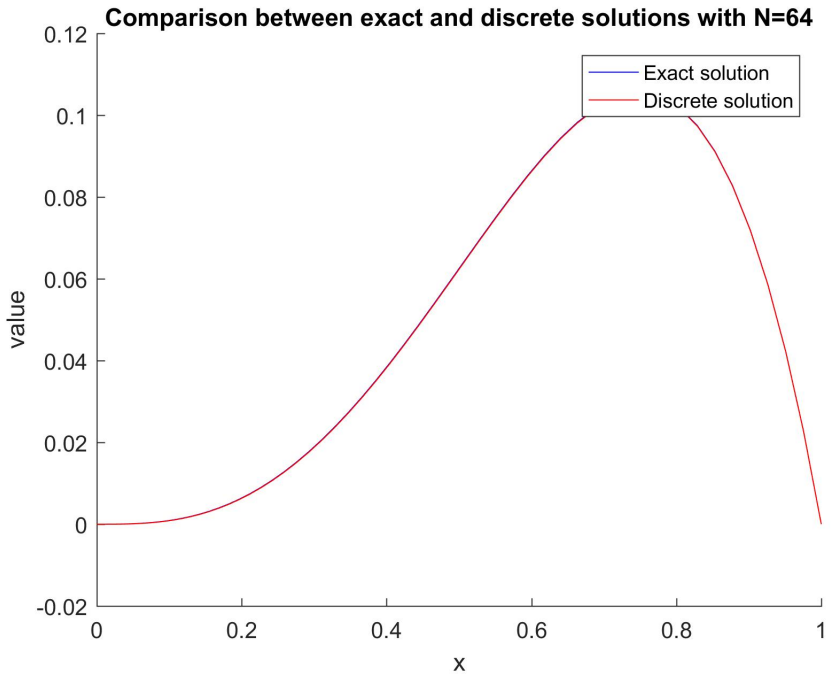

Figure 34: Numerical Solutions: Problem 6.1.b, Test 2, $N = 64$.

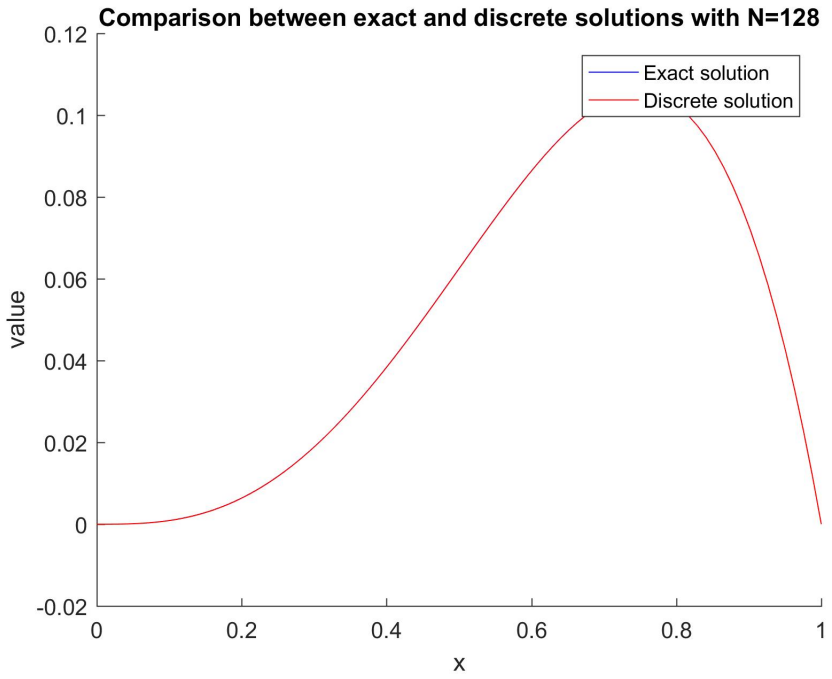

Figure 35: Numerical Solutions: Problem 6.1.b, Test 2, $N = 128$.

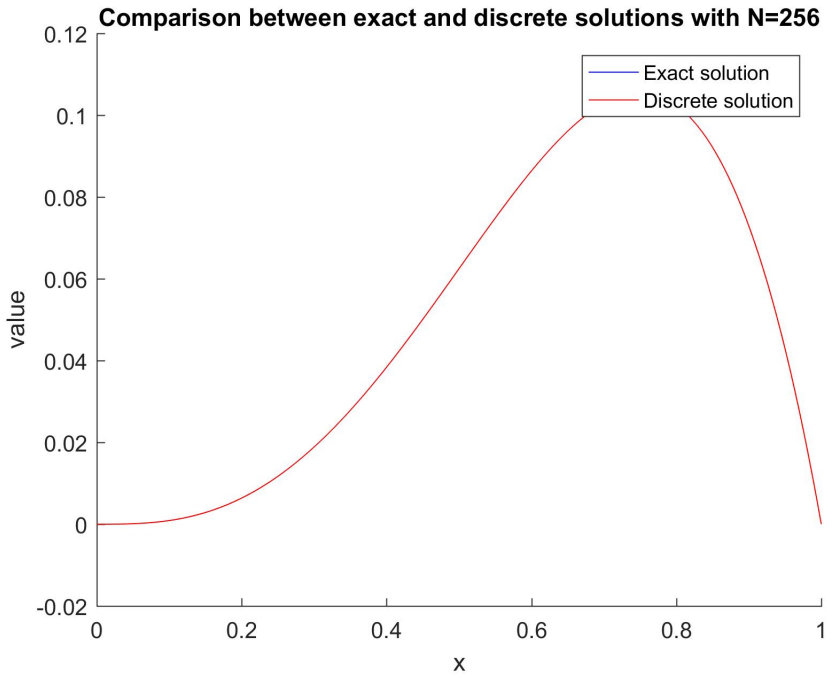

Figure 36: Numerical Solutions: Problem 6.1.b, Test 2, $N = 256$.

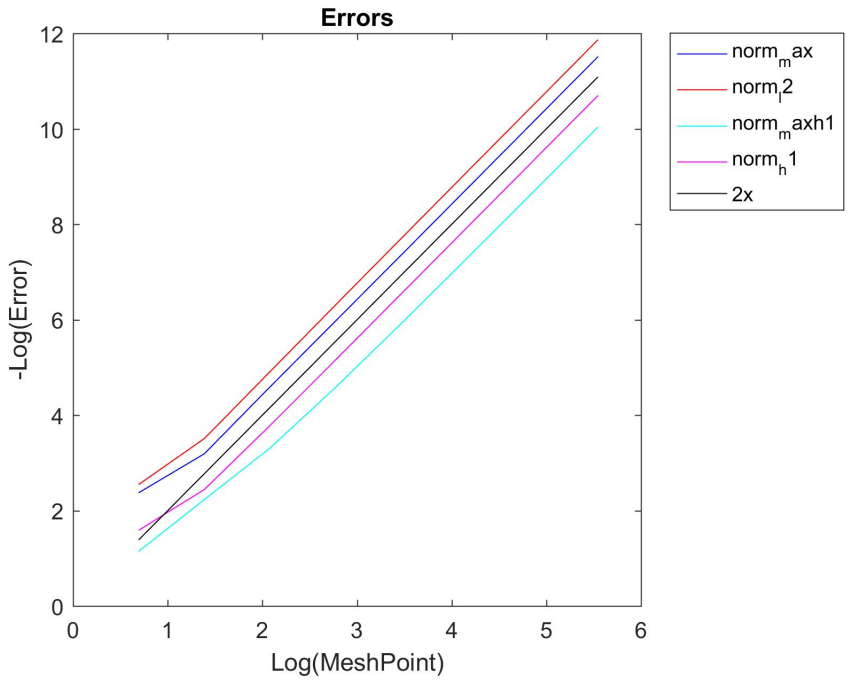

Figure 37: Errors on a log-log scale: Problem 6.1.b, Test 2.

Remark 6.7. We see that the errors converge to 0 as N change from 2 to 8 with rate slightly slower than quadratic rate, but after that, the errors converge with quadratic rate $O\left(h^2\right)$.

Case 3.

$$u_{ex} = x(1-x)^2 \tag{6.14}$$

$$f = 4 - 6x \tag{6.15}$$

Run these scripts for `cases=3`, Matlab returns

```
N=2
error on L^infinity: 8.578644e-02f
error on L^2: 7.213751e-02f
error on maxH1: 2.928932e-01f
error on H1: 1.885044e-01f

N=4
error on L^infinity: 1.890093e-02f
error on L^2: 1.561827e-02f
error on maxH1: 9.502140e-02f
error on H1: 4.602736e-02f

N=8
error on L^infinity: 4.983950e-03f
```

```
error on L^2: 3.699760e-03f
error on maxH1: 2.520887e-02f
error on H1: 1.149505e-02f

N=16
error on L^infinity: 1.236966e-03f
error on L^2: 9.044500e-04f
error on maxH1: 6.394593e-03f
error on H1: 2.873593e-03f

N=32
error on L^infinity: 3.088609e-04f
error on L^2: 2.238231e-04f
error on maxH1: 1.604446e-03f
error on H1: 7.183956e-04f

N=32
error on L^infinity: 3.088609e-04f
error on L^2: 2.238231e-04f
error on maxH1: 1.604446e-03f
error on H1: 7.183956e-04f

N=64
error on L^infinity: 7.729785e-05f
error on L^2: 5.568549e-05f
error on maxH1: 4.014743e-04f
error on H1: 1.795988e-04f

N=128
error on L^infinity: 1.932228e-05f
error on L^2: 1.388855e-05f
error on maxH1: 1.003912e-04f
error on H1: 4.489971e-05f

N=256
error on L^infinity: 4.830434e-06f
error on L^2: 3.468093e-06f
error on maxH1: 2.509923e-05f
error on H1: 1.122493e-05f
```

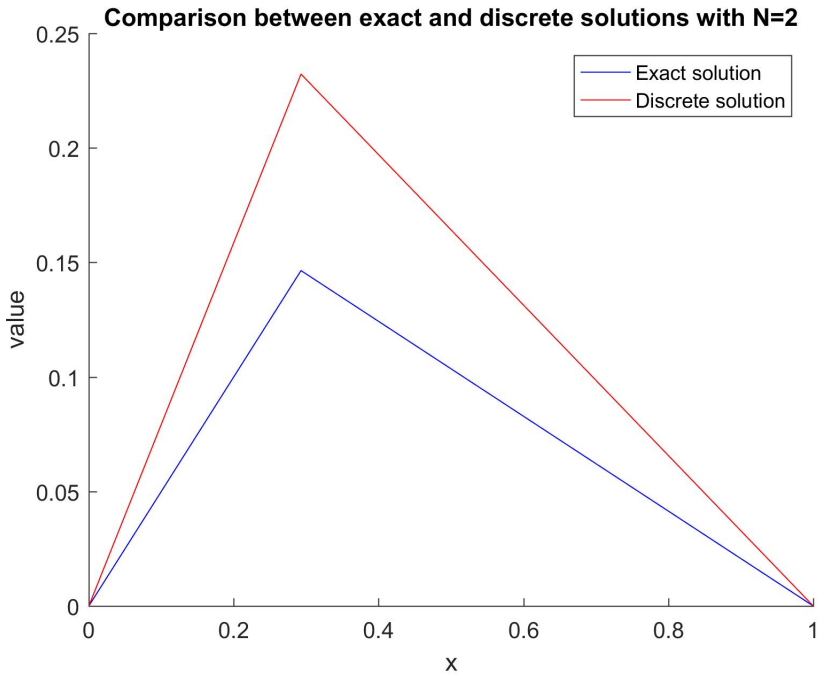

Figure 38: Numerical Solutions: Problem 6.1.b, Test 2, $N = 2$.

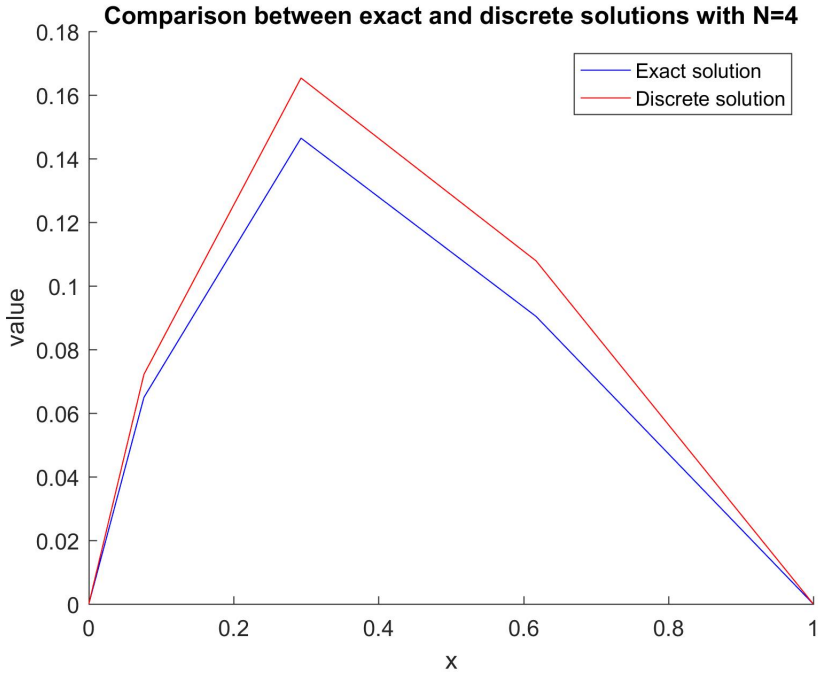

Figure 39: Numerical Solutions: Problem 6.1.b, Test 2, $N = 4$.

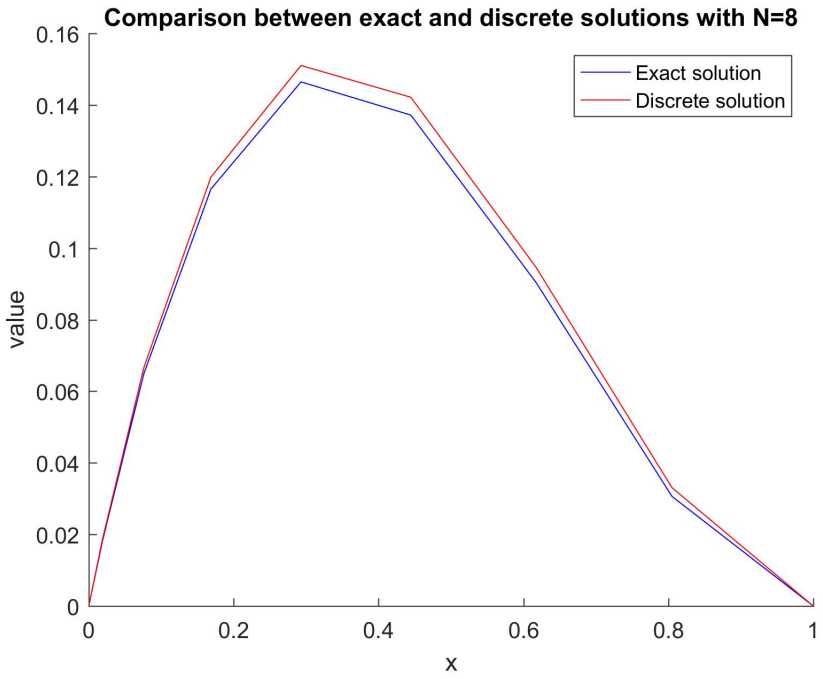

Figure 40: Numerical Solutions: Problem 6.1.b, Test 2, $N = 8$.

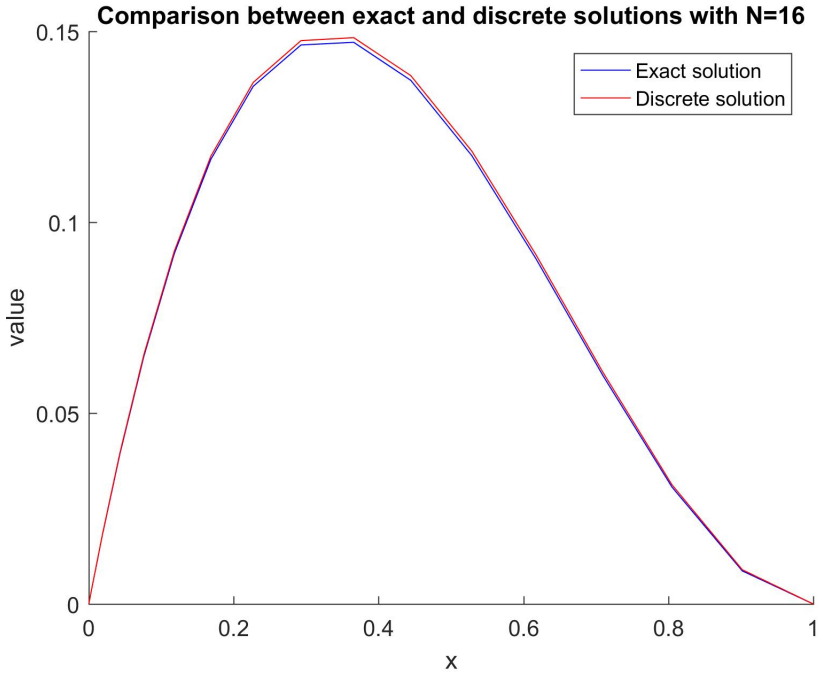

Figure 41: Numerical Solutions: Problem 6.1.b, Test 2, $N = 16$.

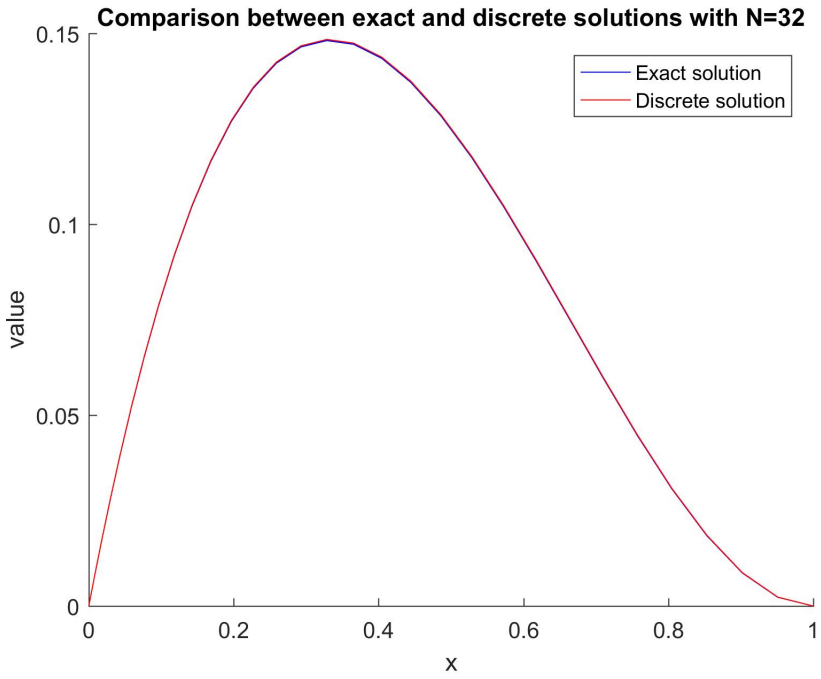

Figure 42: Numerical Solutions: Problem 6.1.b, Test 2, $N = 32$.

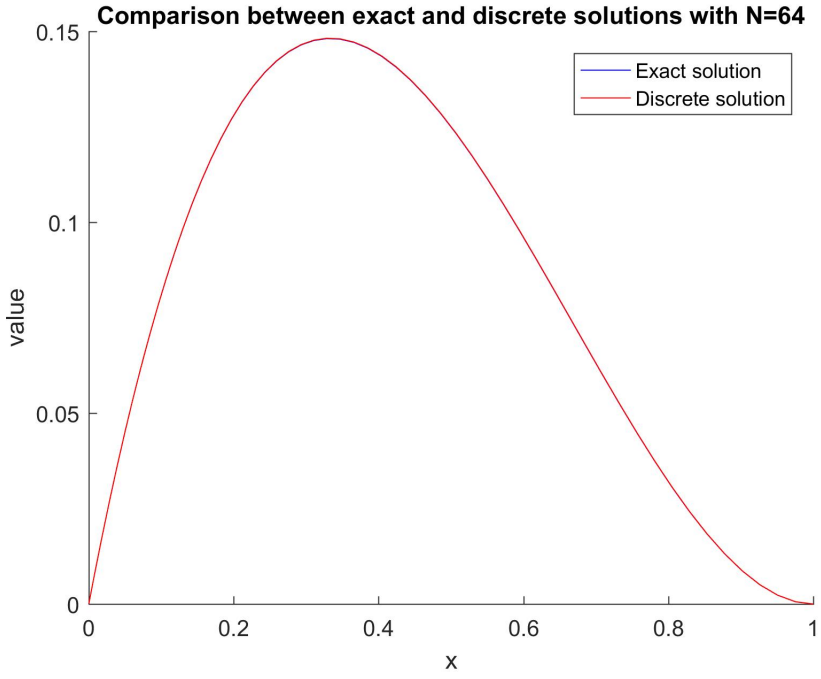

Figure 43: Numerical Solutions: Problem 6.1.b, Test 2, $N = 64$.

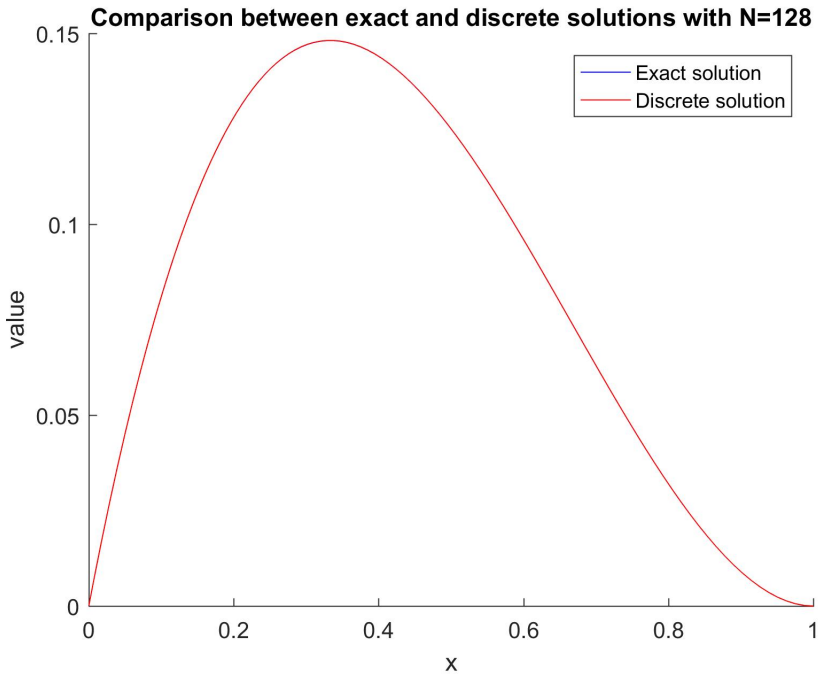

Figure 44: Numerical Solutions: Problem 6.1.b, Test 2, $N = 128$.

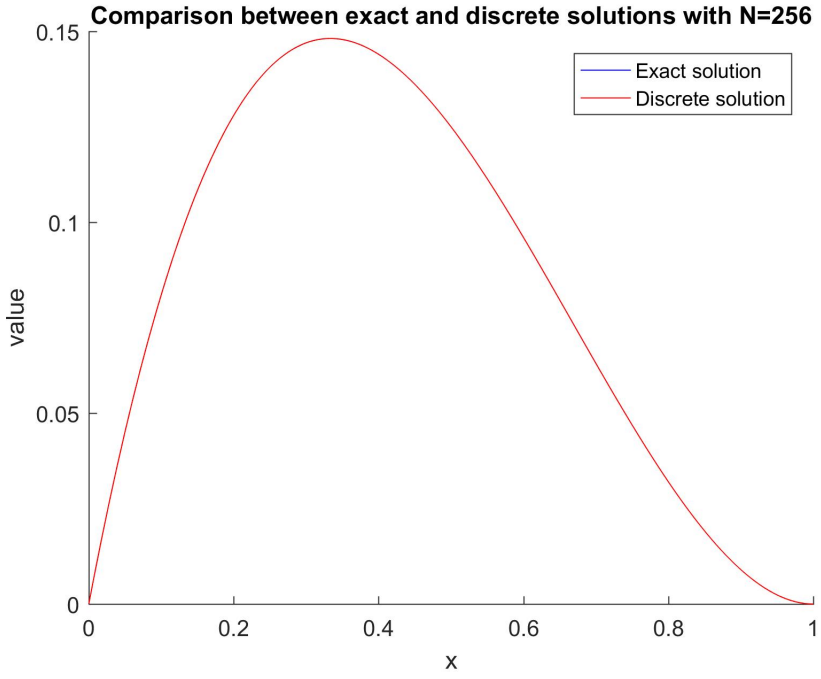

Figure 45: Numerical Solutions: Problem 6.1.b, Test 2, $N = 256$.

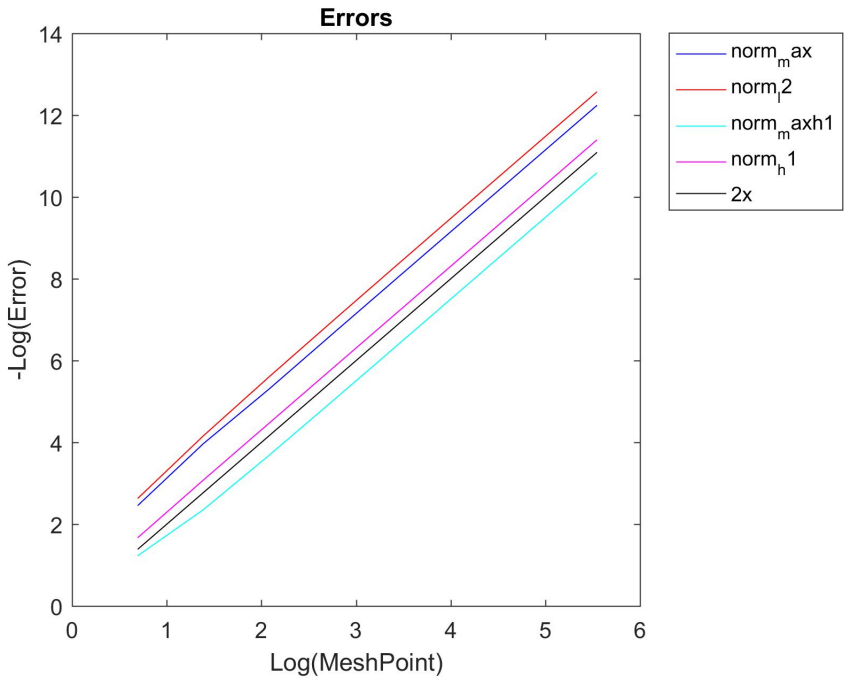

Figure 46: Errors on a log-log scale: Problem 6.1.b, Test 2.

6.4 Matlab Implementation for Problem 6.2

6.4.1 Matlab Scripts

exact_solution.m

```
function u_ex=exact_solution(x,k);
if (k==1)
    u_ex=-x^2+4;
end
if(k==2)
    u_ex=x^3*(1-x)+3;
end
if(k==3)
    u_ex=x^2+2;
end
```

functionf.m

```
function f=functionf(x,k);
if (k==1)
    f=2;
end
if (k==2)
    f=-6*x+12*x^2;
end
```

```
if (k==3)
    f=-2;
end
```

main.m

```
% Solve equation -u''(x)=f(x) with the Dirichlet
% boundary condition
clear all
close all
clc
format long
%% Initial informations
ax=0.0;
bx=1.0;
cases=2;
alpha=0;
beta=3;
N=2;% number of mesh points of first mesh
number_mesh=8;
number_mesh_point=zeros(number_mesh,1);
norm_max=zeros(number_mesh,1);
norm_l2=zeros(number_mesh,1);
norm_maxh1=zeros(number_mesh,1);
norm_h1=zeros(number_mesh,1);
%% Solve discrite solution and refine mesh

for inumber_mesh=1:number_mesh
    fprintf('N=%d\n',N);
    number_mesh_point(inumber_mesh)=N;
    delta_x=(bx-ax)/N;

%% Create mesh point
    x=zeros(N+1,1);
    for i=1:N+1
        x(i)=(i-1)*delta_x;
    end
 %% Create matrix A
    A=sparse(N-1,N-1);
    for i=1:N-1
        if (i==1)
            A(i,i)=2/3;
            A(i,i+1)=-2/3;
        elseif(i==N-1)
            A(i,i-1)=-1;
            A(i,i)=2;
        else
            A(i,i-1)=-1;
            A(i,i+1)=-1;
            A(i,i)=2;
        end
```

```
    end
    A=A/((delta_x)^2);
%% Create vector b
    b=zeros(N-1,1);
    for i=1:N-2
        b(i)=functionf(i*delta_x,cases);
    end
    b(N-1)=functionf((N-1)*delta_x,cases)+beta/(delta_x^2);
%% Solve discrete solution
    u=A\b;
%% Get exact solution
    u_ex=zeros(N+1,1);
    for i=1:N+1
        u_ex(i)=exact_solution((i-1)*delta_x,cases);
    end
%% Create discrete solution with boundary
    u_dis=zeros(N+1,1);
    for i=1:N+1
        if (i==1)
            u_dis(i)=4*u(1)/3-u(2)/3;
        elseif(i==N+1)
            u_dis(i)=beta;
        else
            u_dis(i)=u(i-1,1);
        end
    end
%% Calculate the error on L^infinity
    norm_max(inumber_mesh)=0.0;
    for i=1:N+1
        if (abs(u_dis(i)-u_ex(i)) > norm_max(inumber_mesh))
            norm_max(inumber_mesh)=abs(u_dis(i)-u_ex(i));
        end
    end
    fprintf('error on L^infinity: %df\n',norm_max(inumber_mesh));
%%  Calculate the error on L^2

    norm_l2(inumber_mesh)=0;
    for i=1:N+1
        norm_l2(inumber_mesh)=norm_l2(inumber_mesh) ...
        +(u_dis(i)-u_ex(i))^2*delta_x;
    end
    norm_l2(inumber_mesh)=(norm_l2(inumber_mesh))^(1/2);
    fprintf('error on L^2: %df\n',norm_l2(inumber_mesh));
%% Calculate the error on maxH1

    norm_maxh1(inumber_mesh)=0;
    for i=1:N
        if (abs(((u_dis(i+1)-u_ex(i+1)) ...
        -(u_dis(i)-u_ex(i)))/delta_x) > norm_maxh1(inumber_mesh))
            norm_maxh1(inumber_mesh)=abs(((u_dis(i+1)-u_ex(i+1)) ...
```

```
            -(u_dis(i)-u_ex(i)))/delta_x);
        end
    end
    fprintf('error on maxH1: %df\n',norm_maxh1(inumber_mesh));

%% Calculate the error on H1

    norm_h1(inumber_mesh)=0;
    for i=1:N
        norm_h1(inumber_mesh)=norm_h1(inumber_mesh) ...
        +(((u_dis(i+1)-u_ex(i+1))-(u_dis(i)-u_ex(i)))/...
        delta_x)^2*delta_x;
    end
    norm_h1(inumber_mesh)=(norm_h1(inumber_mesh))^(1/2);
    fprintf('error on H1: %df\n',norm_h1(inumber_mesh));
%% Figure exact and dicrete solutions
    figure
    plot(x,u_ex,'blue', x,u_dis,'red');
    xlabel('x');ylabel('value');
    str=sprintf('Comparison between exact and discrete ...
                solutions with N=%d',N);
    title(str);
%     ylim([-0.02 0.12]);
    legend('Exact solution','Discrete solution');
%     print('-r300','-djpeg');
%% Refine mesh (increse mesh point)
    N=2*N;
%% Seperation
    disp(' ');
end
%% Figure for errors respect to number of mesh point
figure
plot(log(number_mesh_point), -log(norm_max),'blue', ...
     log(number_mesh_point), -log(norm_l2), 'red',...
     log(number_mesh_point), -log(norm_maxh1), 'cyan', ...
     log(number_mesh_point), -log(norm_h1), 'magenta', ...
     log(number_mesh_point), 2*log(number_mesh_point),'black');
xlabel('Log(MeshPoint)');ylabel('-Log(Error)');
title('Errors');
legend('norm_max','norm_l2','norm_maxh1','norm_h1','2x', ...
       'Location','NorthEastOutside');
% print('-r300','-djpeg');
```

6.4.2 Results

Case 1.

$$u_{ex} = -x^2 + 4 \tag{6.16}$$

$$f = 2 \tag{6.17}$$

Run these scripts for `cases=1`, MATLAB returns

```
N=2
error on L^infinity: 3f
error on L^2: 2.371708e+00f
error on maxH1: 3f
error on H1: 3f

N=4
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=8
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=16
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=32
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=64
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=128
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=256
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f
```

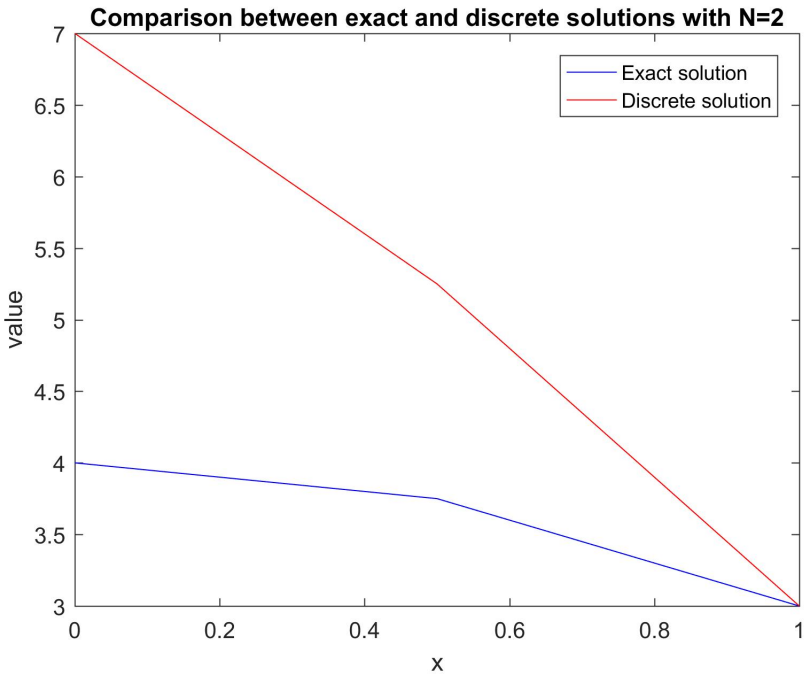

Figure 47: Numerical Solutions: Problem 6.2, Test 1, $N = 2$.

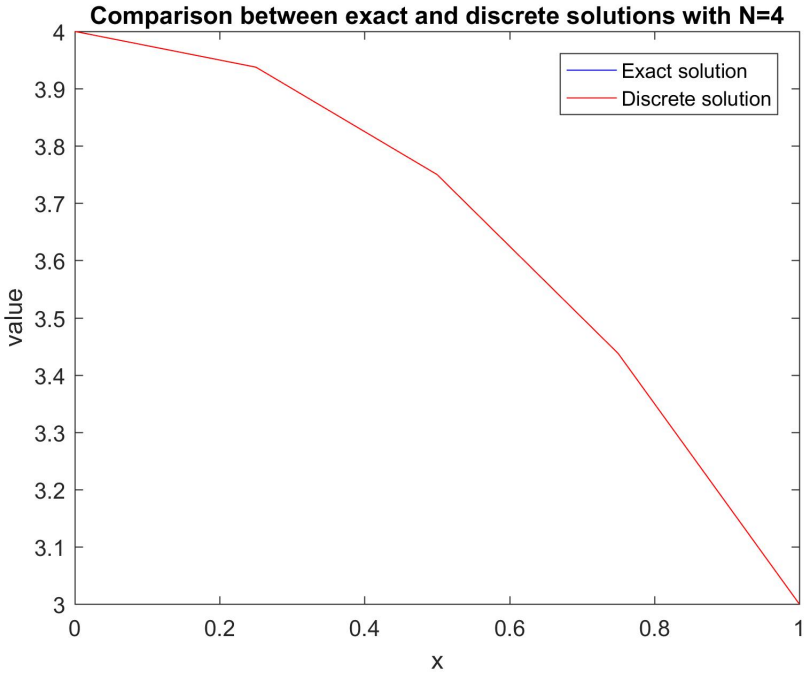

Figure 48: Numerical Solutions: Problem 6.2, Test 1, $N = 4$.

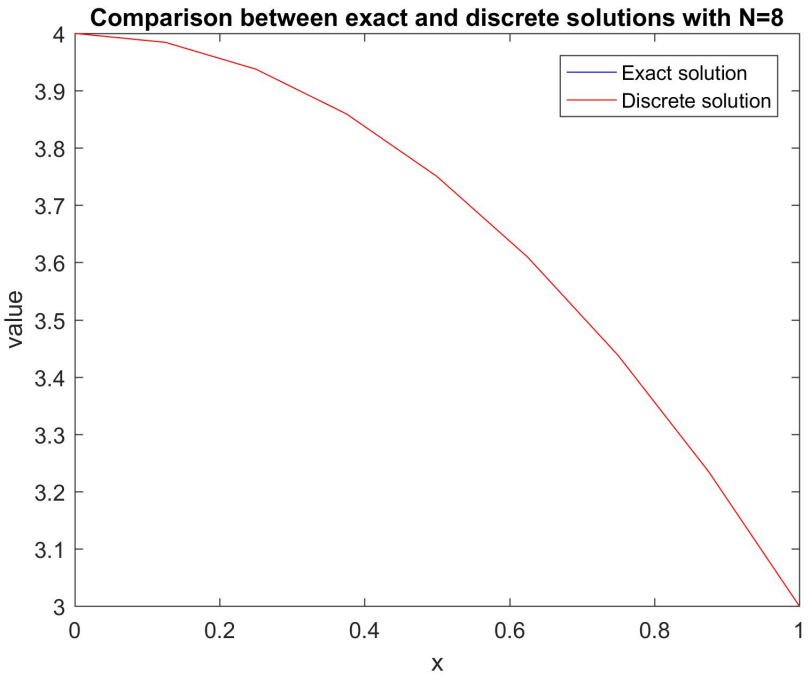

Figure 49: Numerical Solutions: Problem 6.2, Test 1, $N = 8$.

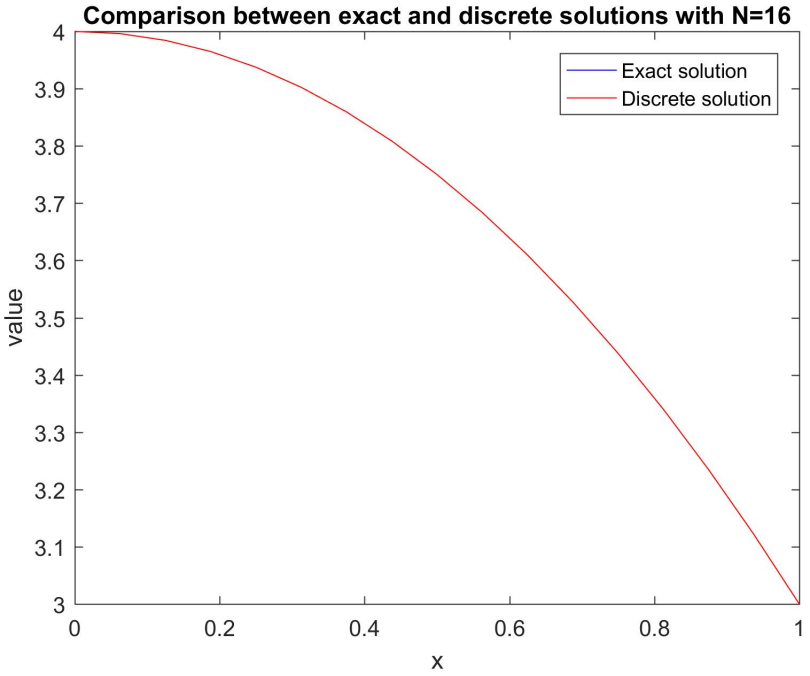

Figure 50: Numerical Solutions: Problem 6.2, Test 1, $N = 16$.

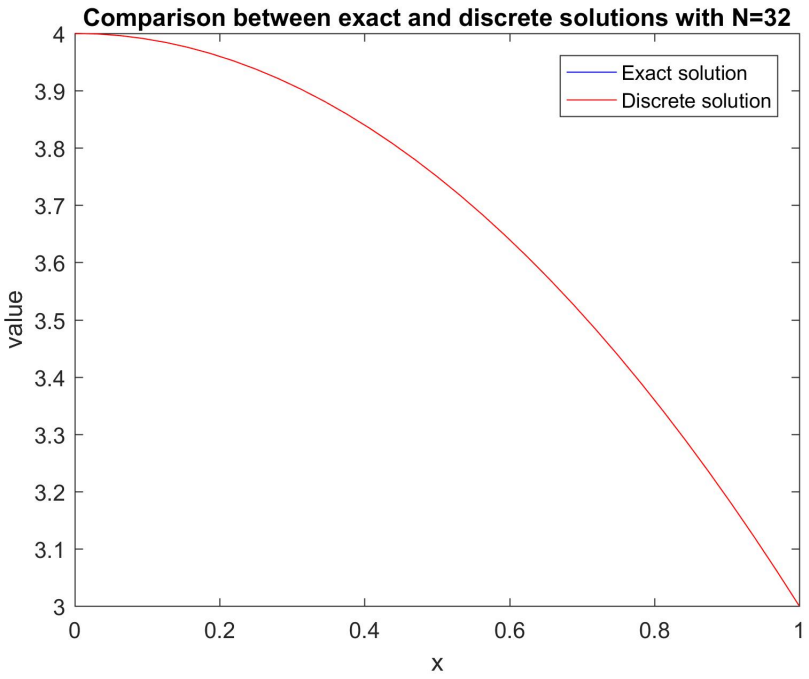

Figure 51: Numerical Solutions: Problem 6.2, Test 1, $N = 32$.

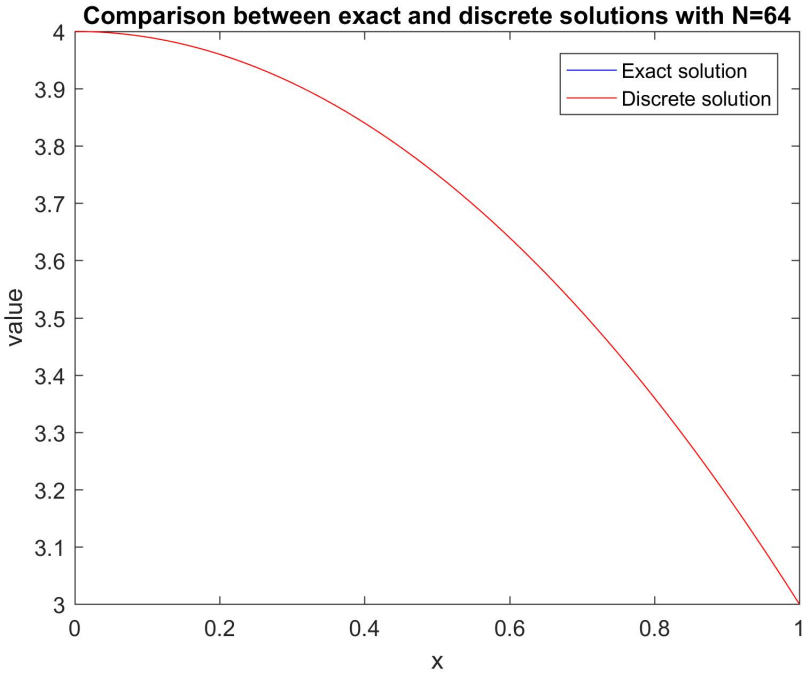

Figure 52: Numerical Solutions: Problem 6.2, Test 1, $N = 64$.

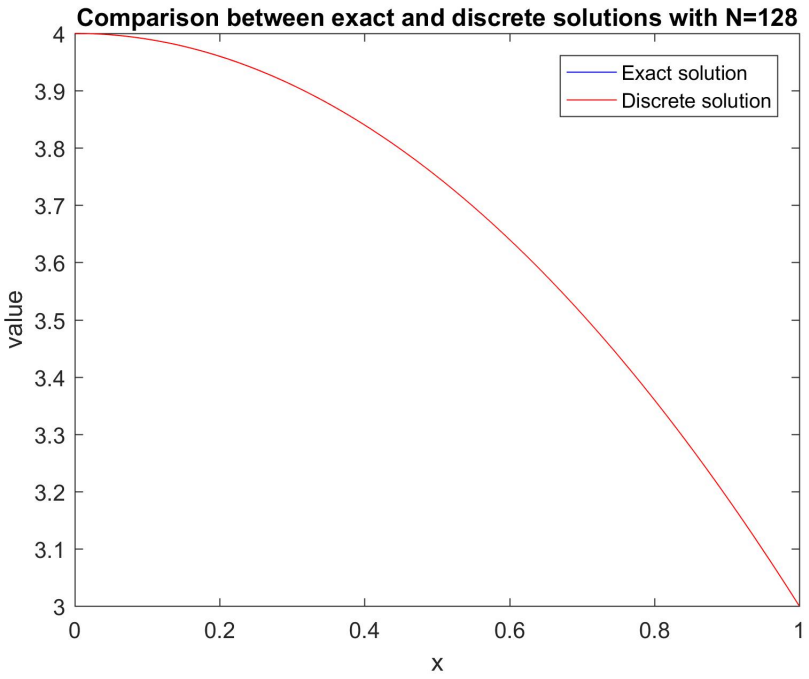

Figure 53: Numerical Solutions: Problem 6.2, Test 1, $N = 128$.

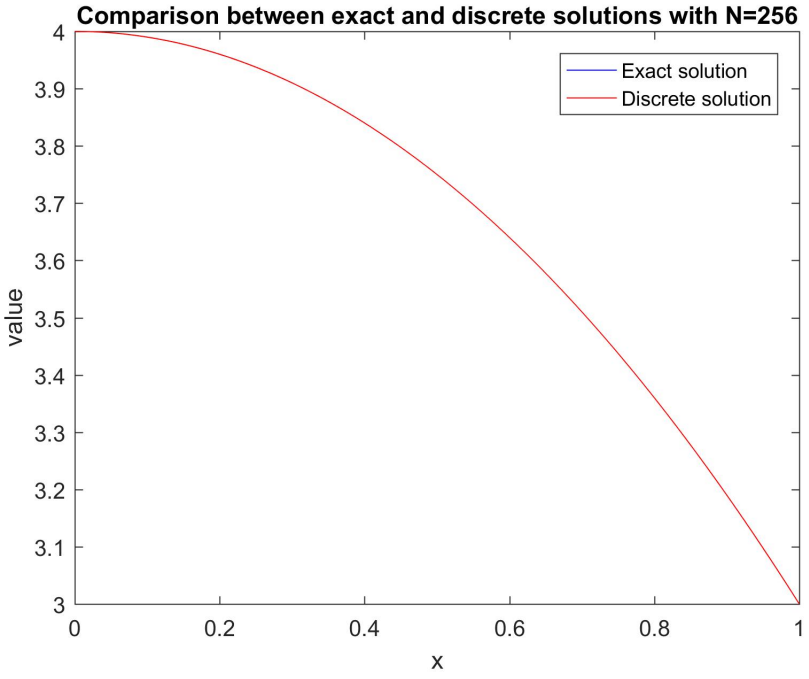

Figure 54: Numerical Solutions: Problem 6.2, Test 1, $N = 256$.

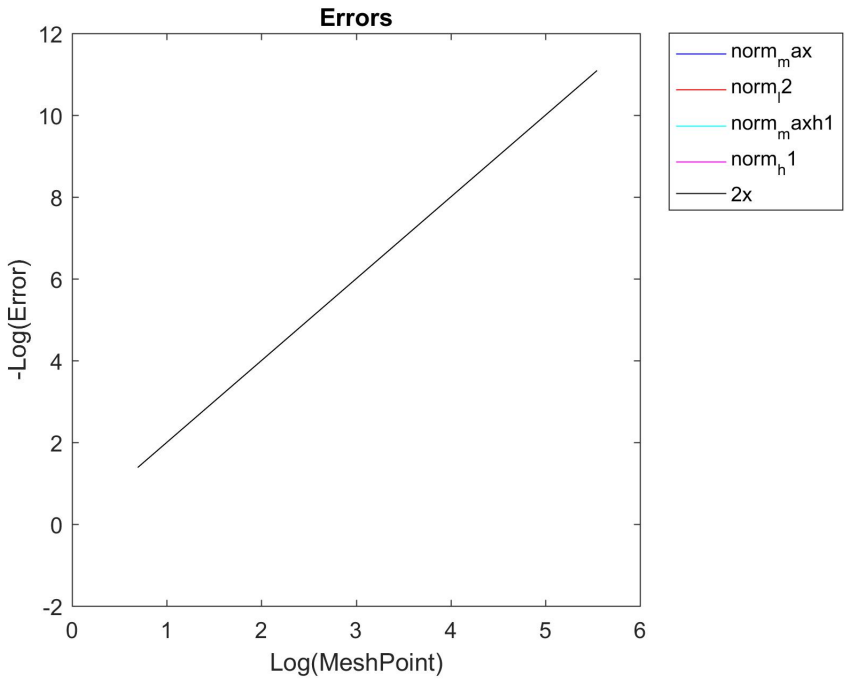

Figure 55: Errors on a log-log scale: Problem 6.2, Test 1.

Case 2.

$$u_{ex} = x^3 (1 - x) + 3 \tag{6.18}$$

$$f = -6x + 12x^2 \tag{6.19}$$

Run these scripts for `cases=2`, MATLAB returns

```
N=2
error on L^infinity: 3f
error on L^2: 2.352276e+00f
error on maxH1: 3.125000e+00f
error on H1: 3.002603e+00f

N=4
error on L^infinity: 9.375000e-02f
error on L^2: 7.186863e-02f
error on maxH1: 1.406250e-01f
error on H1: 1.000488e-01f

N=8
error on L^infinity: 3.515625e-02f
error on L^2: 2.430456e-02f
error on maxH1: 4.882813e-02f
error on H1: 3.627769e-02f

N=16
error on L^infinity: 1.025391e-02f
error on L^2: 6.749811e-03f
error on maxH1: 1.391602e-02f
error on H1: 1.049805e-02f

N=32
error on L^infinity: 2.746582e-03f
error on L^2: 1.764102e-03f
error on maxH1: 3.692627e-03f
error on H1: 2.803800e-03f

N=64
error on L^infinity: 7.095337e-04f
error on L^2: 4.501300e-04f
error on maxH1: 9.498596e-04f
error on H1: 7.233958e-04f

N=128
error on L^infinity: 1.802444e-04f
error on L^2: 1.136406e-04f
error on maxH1: 2.408028e-04f
error on H1: 1.836566e-04f

N=256
```

```
error on L^infinity: 4.541874e-05f
error on L^2: 2.854680e-05f
error on maxH1: 6.061792e-05f
error on H1: 4.626522e-05f
```

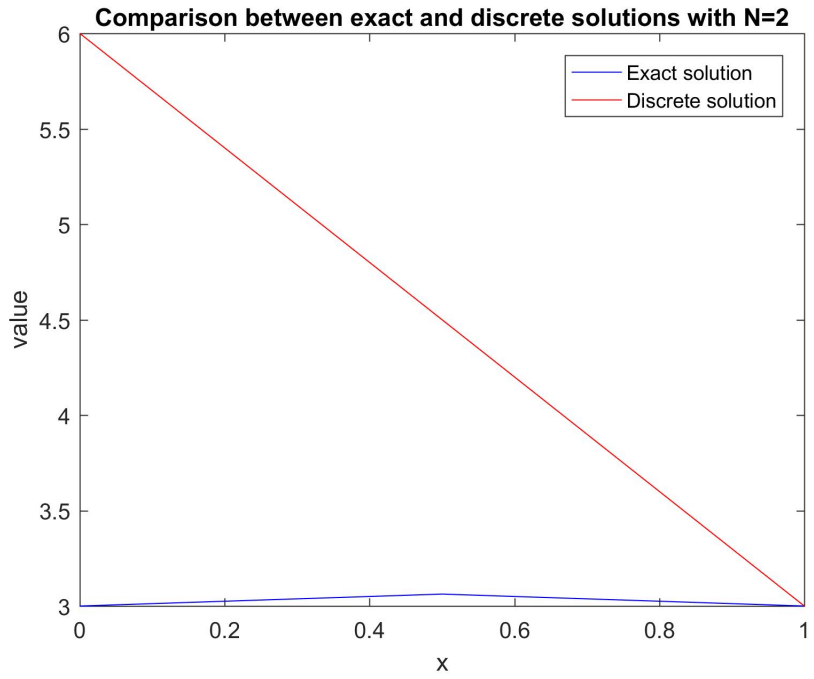

Figure 56: Numerical Solutions: Problem 6.2, Test 2, $N = 2$.

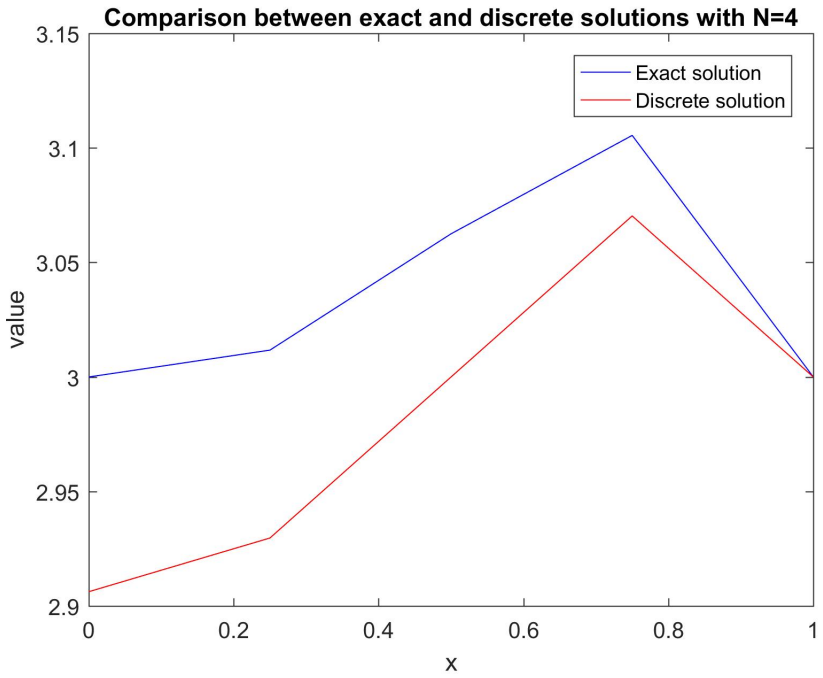

Figure 57: Numerical Solutions: Problem 6.2, Test 2, $N = 4$.

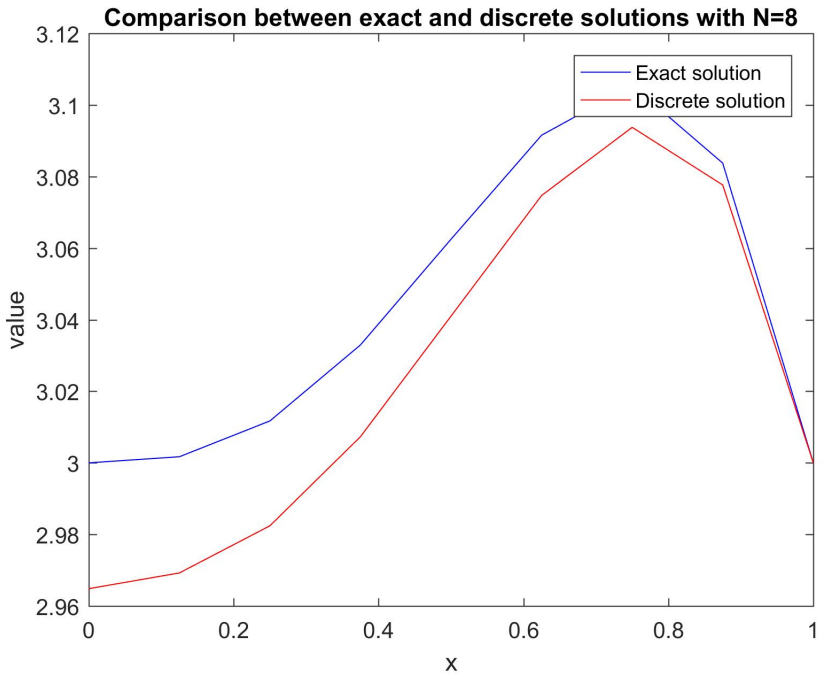

Figure 58: Numerical Solutions: Problem 6.2, Test 2, $N = 8$.

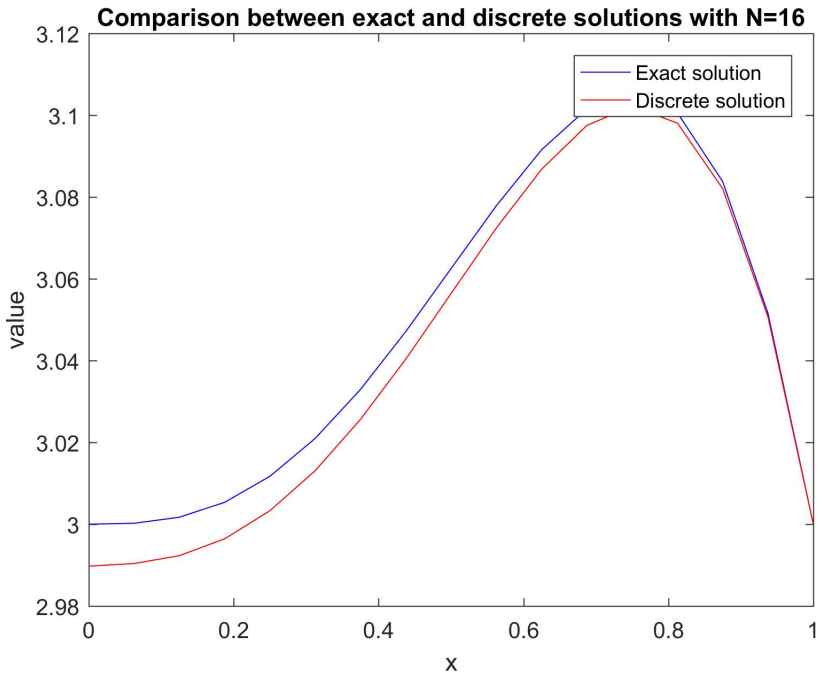

Figure 59: Numerical Solutions: Problem 6.2, Test 2, $N = 16$.

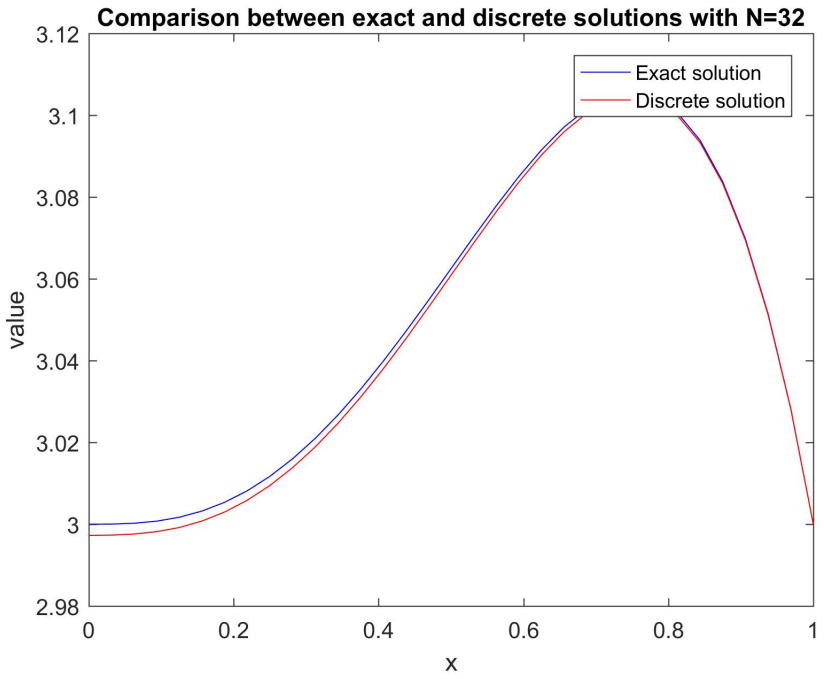

Figure 60: Numerical Solutions: Problem 6.2, Test 2, $N = 32$.

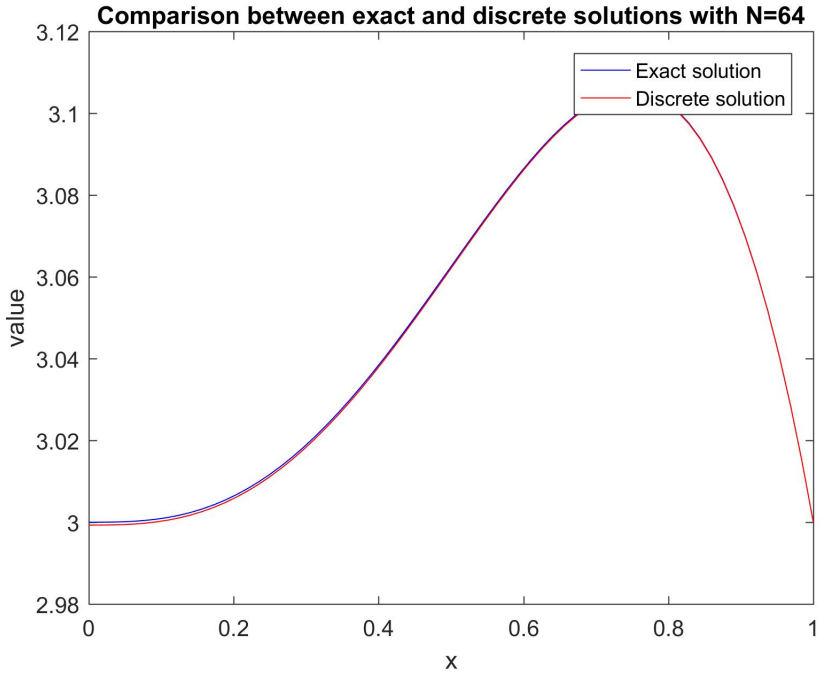

Figure 61: Numerical Solutions: Problem 6.2, Test 2, $N = 64$.

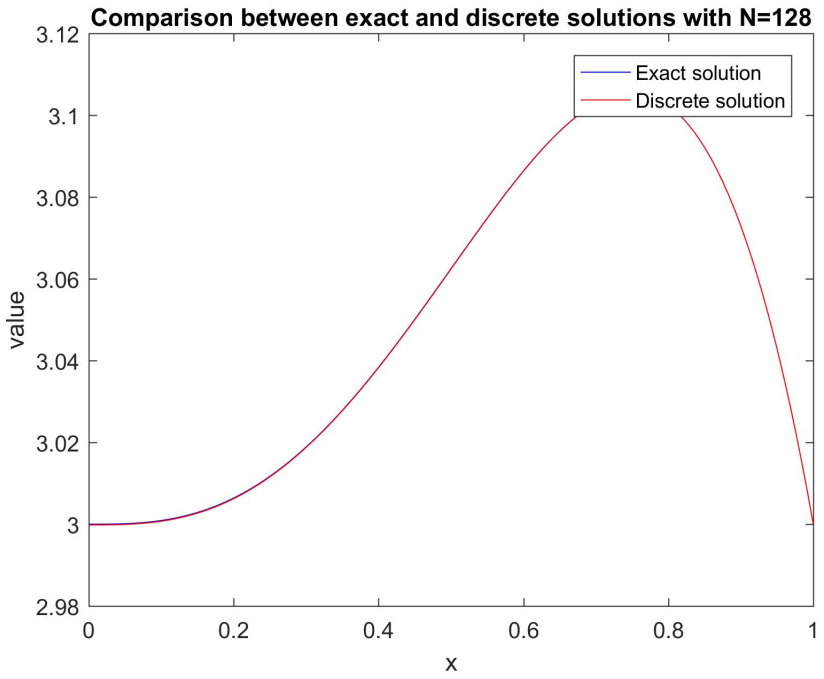

Figure 62: Numerical Solutions: Problem 6.2, Test 2, $N = 128$.

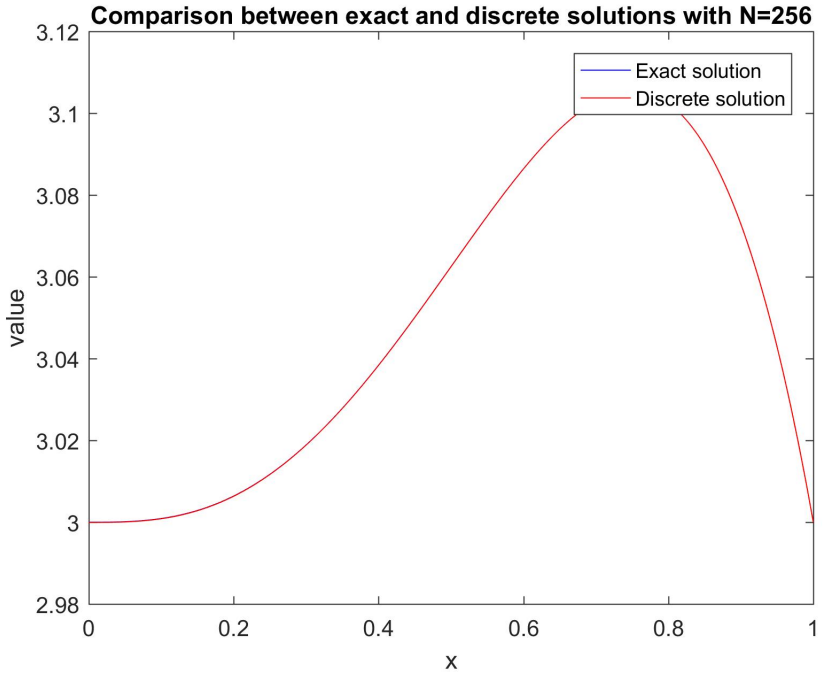

Figure 63: Numerical Solutions: Problem 6.2, Test 2, $N = 256$.

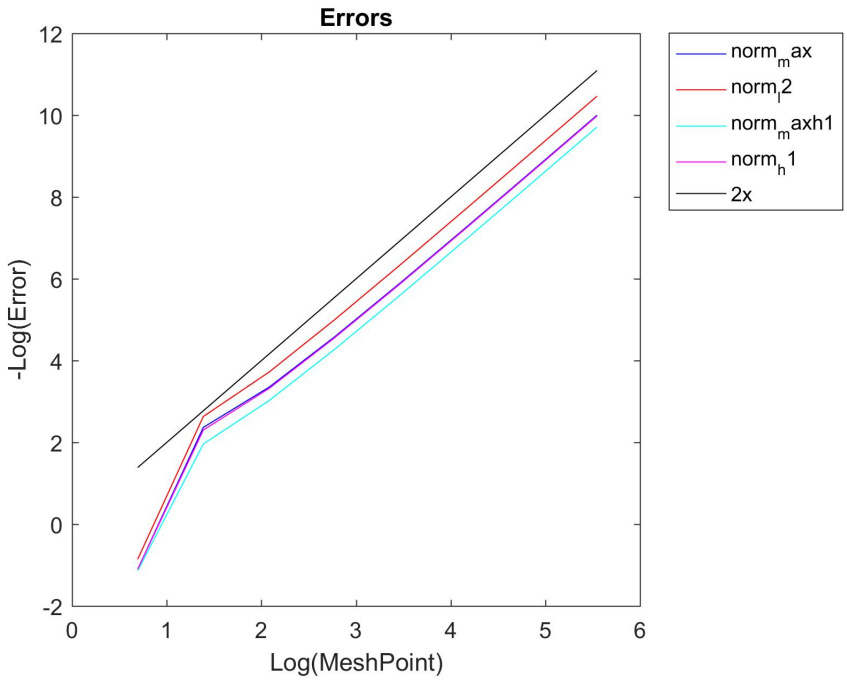

Figure 64: Errors on a log-log scale: Problem 6.2, Test 2.

Remark 6.8. We see that the errors converge to 0 as N change from 2 to 8 with rate slightly faster than quadratic rate, but after that, the errors converge with quadratic rate $O\left(h^2\right)$.

Case 3.

$$u_{ex} = x^2 + 2 \tag{6.20}$$

$$f = -2 \tag{6.21}$$

Run these scripts for `cases=3`, MATLAB returns

```
N=2
error on L^infinity: 3f
error on L^2: 2.371708e+00f
error on maxH1: 3f
error on H1: 3f

N=4
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=8
error on L^infinity: 0f
```

```
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=16
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=32
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=64
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=128
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f

N=256
error on L^infinity: 0f
error on L^2: 0f
error on maxH1: 0f
error on H1: 0f
```

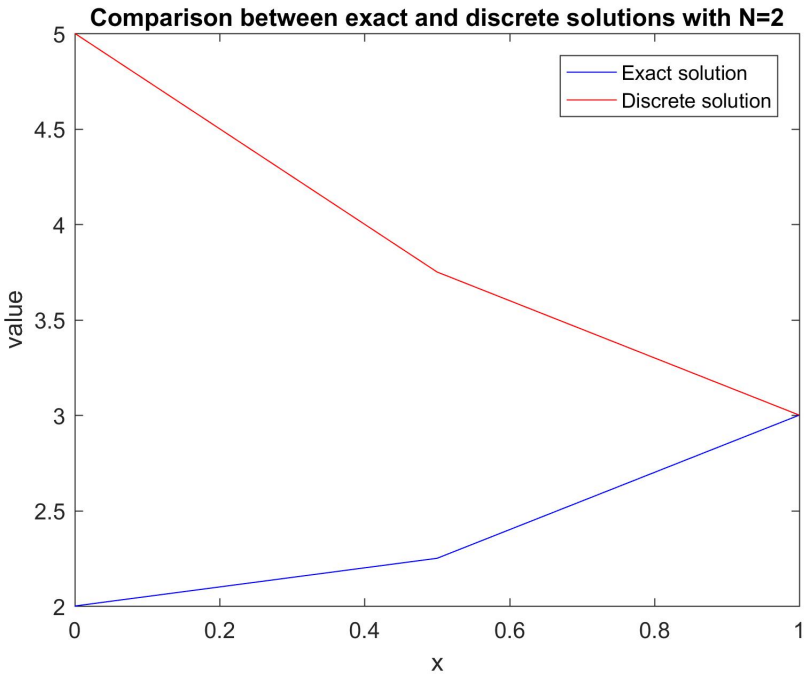

Figure 65: Numerical Solutions: Problem 6.2, Test 3, $N = 2$.

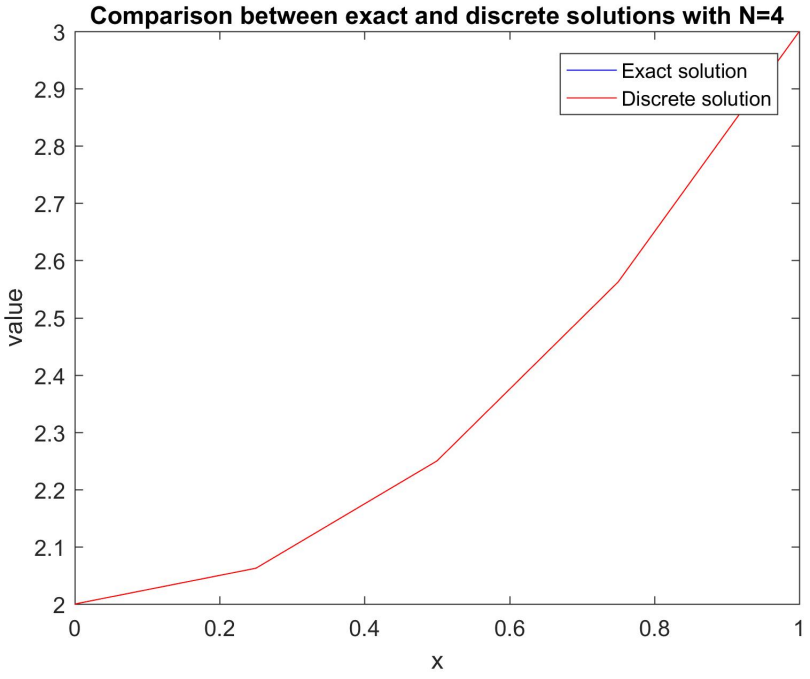

Figure 66: Numerical Solutions: Problem 6.2, Test 3, $N = 4$.

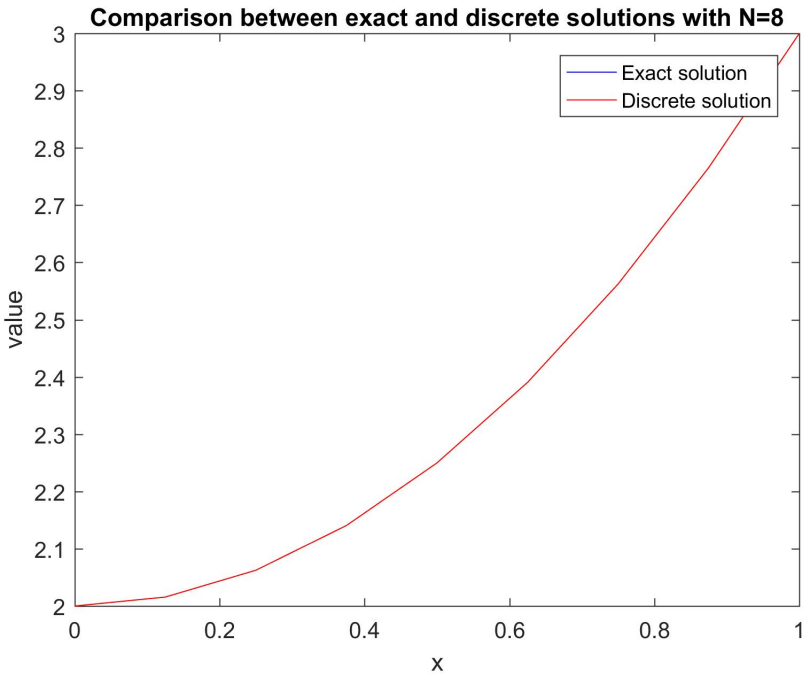

Figure 67: Numerical Solutions: Problem 6.2, Test 3, $N = 8$.

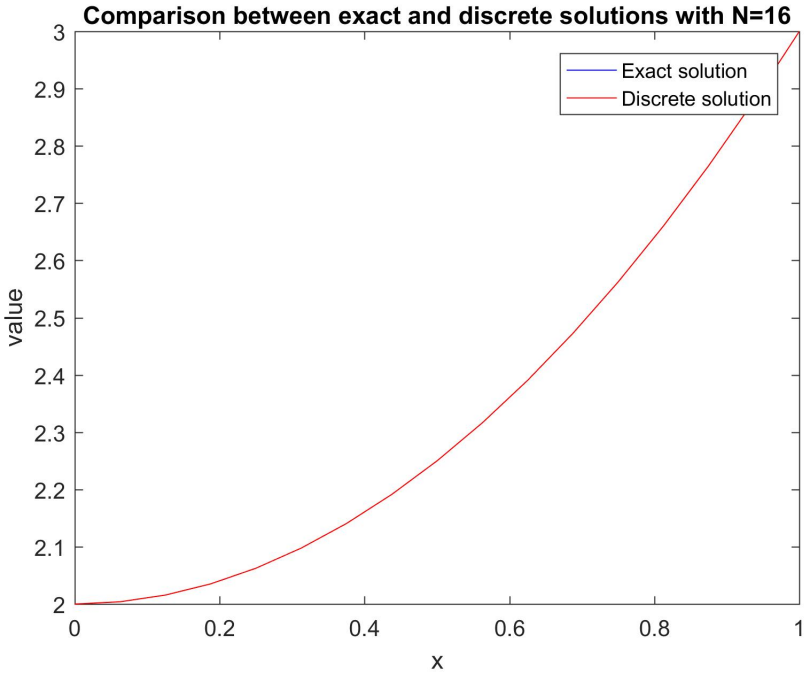

Figure 68: Numerical Solutions: Problem 6.2, Test 3, $N = 16$.

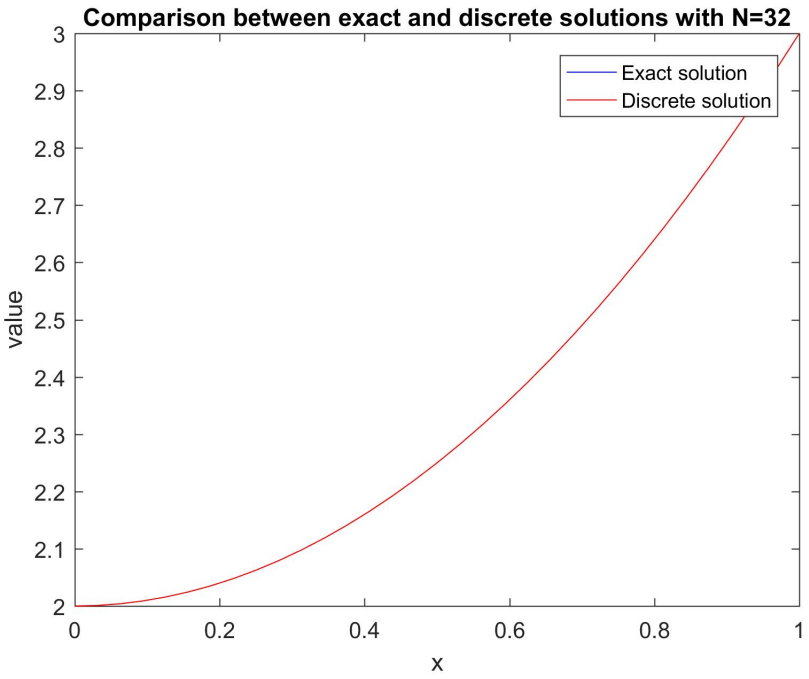

Figure 69: Numerical Solutions: Problem 6.2, Test 3, $N = 32$.

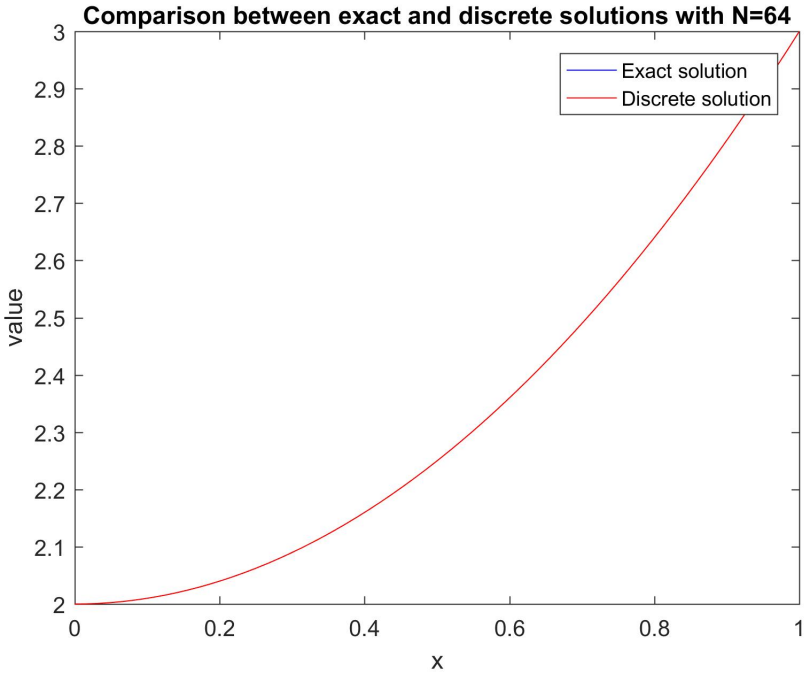

Figure 70: Numerical Solutions: Problem 6.2, Test 3, $N = 64$.

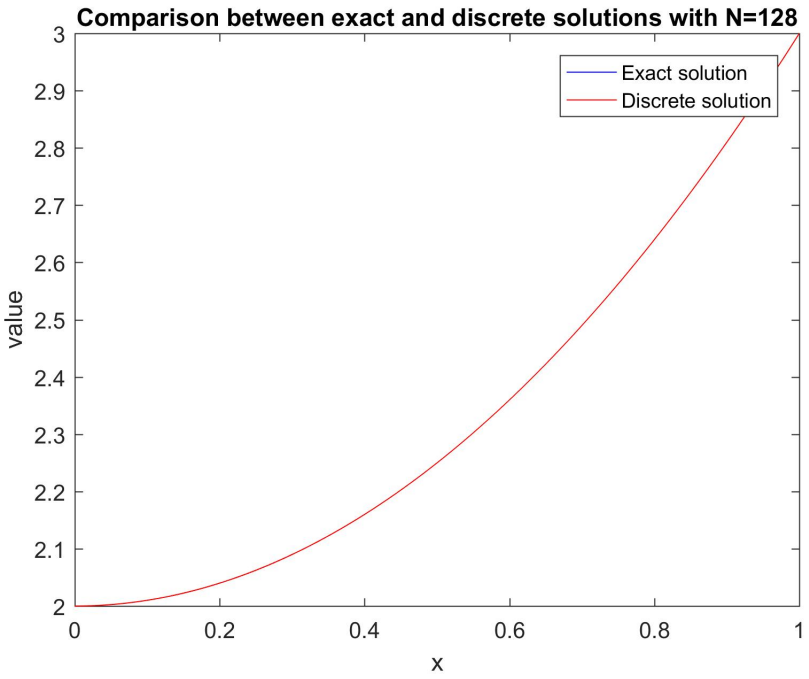

Figure 71: Numerical Solutions: Problem 6.2, Test 3, $N = 128$.

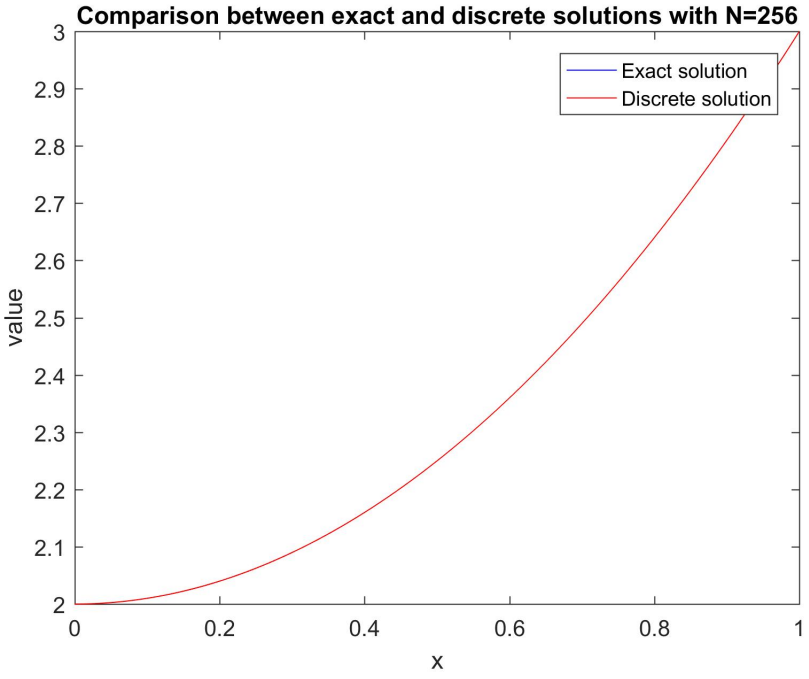

Figure 72: Numerical Solutions: Problem 6.2, Test 3, $N = 256$.

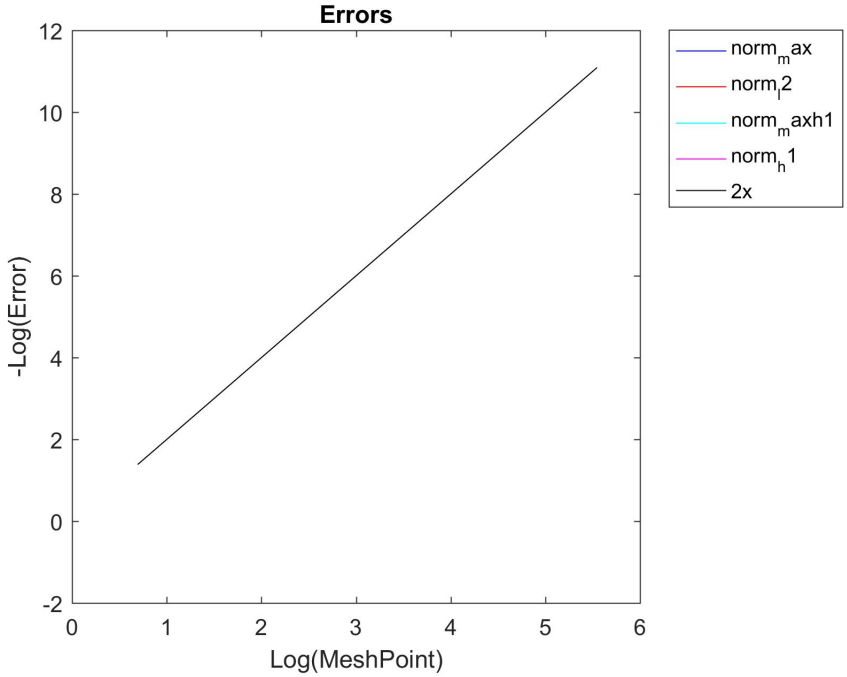

Figure 73: Errors on a log-log scale: Problem 6.2, Test 3.

Remark 6.9. We note that the errors in case 1 and case 3 are the roughly the same. This is not by accident. The key in this situation is the global error formula (3.26). Note that the global error depends only on h and u''. Hence, Case 1 and Case 2 use

$$u_{ex1} = -x^2 + 4 \tag{6.22}$$

$$u_{ex2} = x^2 + 2 \tag{6.23}$$

Thus, ${u_{ex1}}'' = -{u_{ex2}}''$ and we obtain

$$|e_1(x)| = \left| \frac{-1}{12} h^2 {u_{ex1}}''(x) + \frac{1}{12} h^2 \left({u_{ex1}}''(0) + x \left({u_{ex1}}''(1) - {u_{ex1}}''(0) \right) \right) \right| \tag{6.24}$$

$$= \left| \frac{-1}{12} h^2 {u_{ex2}}''(x) + \frac{1}{12} h^2 \left({u_{ex2}}''(0) + x \left({u_{ex2}}''(1) - {u_{ex2}}''(0) \right) \right) \right| \tag{6.25}$$

$$= |e_2(x)| \tag{6.26}$$

6.5 Problem 6.3

We see that if we solve Problem 6.3 analytically by integrating twice and trying to determine the constant of integration from the boundary conditions, there are infinitely many solutions. See [1], p.32-33 for references.

The End

Alphabetical Index

Symbols

A

C

D

E

F

G

I

K

L

M

N

P

S

T

U

References

[1] Randall J. Leveque, *Finite Difference Methods for Ordinary and Partial Differential Equations*, SIAM Society for Industrial and Applied Mathematics, 2007.

[2] Moawwad El-Mikkawy, Abdelrahman Karawia, *Inversion of general tridiagonal matrices*, Department of Mathematics, Faculty of Science, Mansoura University, Mansoura, Egypt, Applied Mathematics Letters 19 (2006) 712-720.

[3] Y Huang, W F McColl, *Analytical inversion of general tridiagonal matrices*, Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, UK, 1997.

[4] Riaz A. Usmani, *Inversion of a Tridiagonal Jacobi Matrix*, Department of Applied Mathematics, University of Manitoba, Winnipeg, Manitoba, Canada R3T 2N2, 1994.

[5] `https://en.wikipedia.org/wiki/Tridiagonal_matrix`

[6] `https://en.wikipedia.org/wiki/Matrix_norm`