CSE 380 Project 2: Steady State Heat Equation Model Document

Sam Potter - STP663

December 13, 2018

Contents

1 Abstract

This document outlines the problem setup (governing equations and boundary conditions) as well as numerical solution techniques for solving 1D and 2D steady state heat equation problems using 2nd and 4th order finite difference.

2 Executive Summary

As of 12/13/18, there are still issues with convergence analysis. Due to this, while most tests can be performed, it is dubious to draw any conclusions because solver correctness has not been verified.

Some lessons learned from this setback include

- Get autotools working at the very beginning of the project so it's not a roadblock at the end
- Test out compiling the code more frequently and start earlier. This makes it easier to address compiler issues and finds potential problems as the code unfolds. It also makes it easier to incrementally verify functionality.
- Don't let my C skills go unused so long. I am rusty!

3 Usage

3.1 Build Procedure

3.1.1 Autotools

Autotools is used to compile the source code. The files `configure.ac` and `Makefile.am` are both setup to link to `MASA`, `GRVY`, and `HDF5`. The `Makefile.am` in `src` is also modified with lines

```
AM_CFLAGS = $(GRVY_CFLAGS)
AM_CFLAGS += $(MASA_CXXFLAGS)
AM_CFLAGS += $(HDF5_CFLAGS)
LIBS = $(GRVY_LIBS)
LIBS += $(MASA_LIBS)
LIBS += $(HDF5_LIBS)
LIBS += -lm
```

A `test` directory was created to perform regression tests using shell scripts and the `make check` target, but convergence analysis issues (see below) have limited the number of regression tests to just working cases (as of 12/13/18, this is the 1d 2nd order case).

To facilitate troubleshooting, a shell script `bootstrap.sh` can be sourced (very important to `source` the script so environment variables are appropriately set in the parent process) to load modules for `autotools` and `hdf5`, to set necessary environment variables (`PKGPATH`), then run `autoreconf -f -i` then `./configure` with appropriate command line options. After that, user can run `make` and `make check`.

3.2 Input Options

Input options available to the user are summarized in Table 3.2.

3.3 Verification Procedure

To run the application in verification mode, the user can either set the value in the input file to 1 or pass the argument `verification 1` on the command line. The user must also specify reasonable numbers for MASA parameters `Ax` and `By` in the input file based on domain size (assumed to be the unit interval in either $\mathbb{R}$ or $\mathbb{R}^2$). Output to `stdout` includes problem details and L2 norm error with MASA exact solution. **Sample output to be include when troubleshooting complete**.

Table 1: Application input options

option	description	acceptable range
method	iterative solver method	(Jacob,Gauss-Seidel)
iter_max	iterative solver max number of iterations	$>= 1$
tol	iterative solver convergence tolerance	positive real double
order	finite difference order	(2,4)
dimension	problem dimension	(1,2)
kappa	material thermal conductivity	positive real double
n	number of nodes along a side	> 2
Txmin	temperature at lower bound of x	real double
Txmax	temperature at upper bound of x	real double
Tymin	temperature at lower bound of y	real double
Tymax	temperature at upper bound of y	real double
Ax	MASA parameter Ax	real double
By	MASA parameter By	real double
verification	Bool switch: runn verification	(0=no, 1=yes)
out_mode	switch for output verbosity	(0=silent, 1=standard, 2=debug)
timer	Bool switch: timing to stdout	(0=no, 1=yes)
out_file	file name to save results to	string

3.4 Verification Exercise

Outstanding issues with convergence results in all cases except 1D 2nd order. See Fig 3.4. through Fig. for current results. Note, using Gauss-Seidal iterative solver because Jacobi diverges for 4th order finite difference. This divergence stems from the fact that the system matrix A is no longer strictly diagonally dominate, thus the Jacobi method won't converge with an arbitrary initial guess.

3.5 Performance Measures

Output to be include when troubleshooting complete. As mentioned above, Gauss-Seidal method used. One dimensional results in Table 3.5. Two dimensional results in Table 3.5.

Table 2: Timing results for 1D convergence testing

n	input	assemble	solve	total

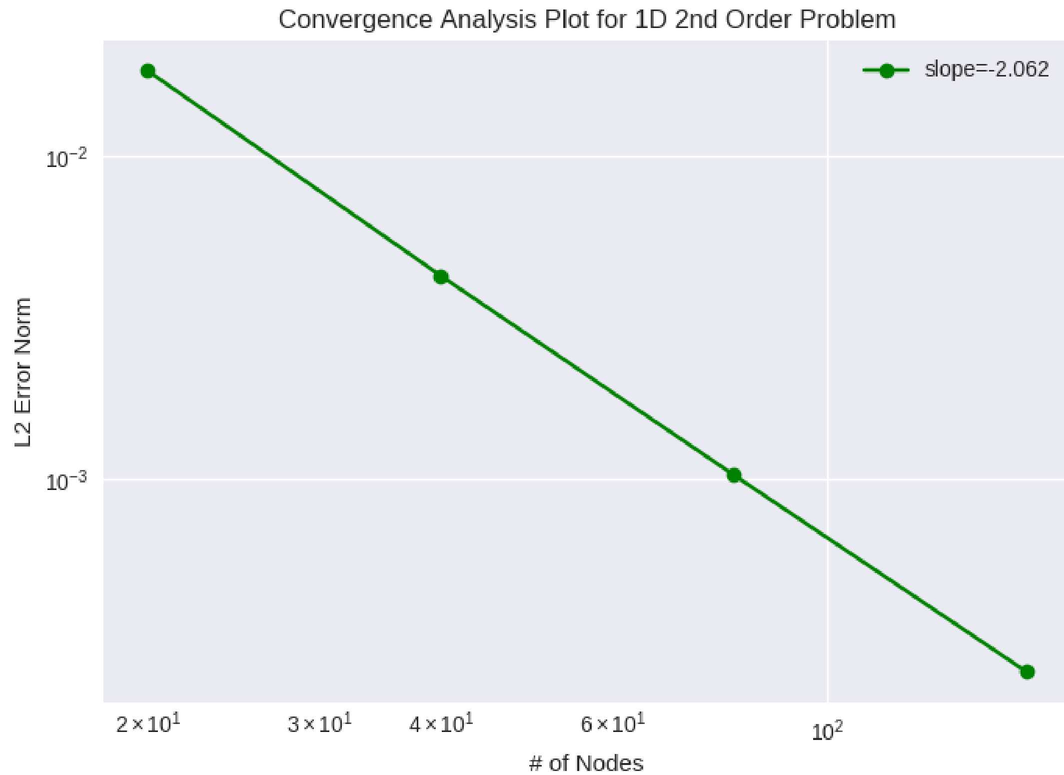

Figure 1: Convergence results for 1D 2nd Order. Working as expected.

Table 3: Timing results for 2D convergence testing

n	input	assemble	solve	total

4 Problem Setup

4.1 Governing Equations

4.1.1 One Dimensional Case

In one dimension, the governing equation is

$$-k\frac{\partial^2 T(x)}{\partial x^2} = q(x) \tag{1}$$

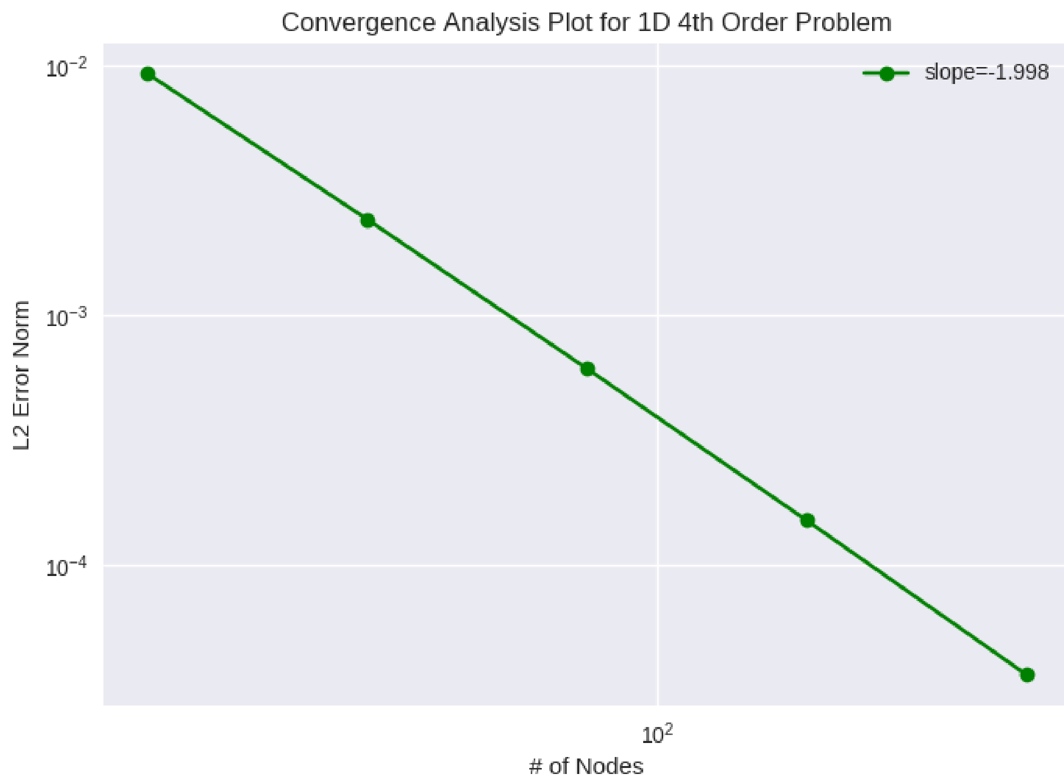

Figure 2: Convergence results for 1D 4th Order. Not working as expected. Convergence rate is half that of theoretical expectation.

4.1.2 Two Dimensional Case

In two dimensions, the governing equation is

$$-k\nabla^2 T(x,y) = q(x,y) \tag{2}$$

4.2 Nomenclature

4.2.1 One Dimensional Case

- $q(x)$: Heat flux at spatial coordinate location x
- x: Spatial coordinate in 1D
- $T(x)$: Temperature of material at spatial coordinate location x
- k: Material thermal conductivity

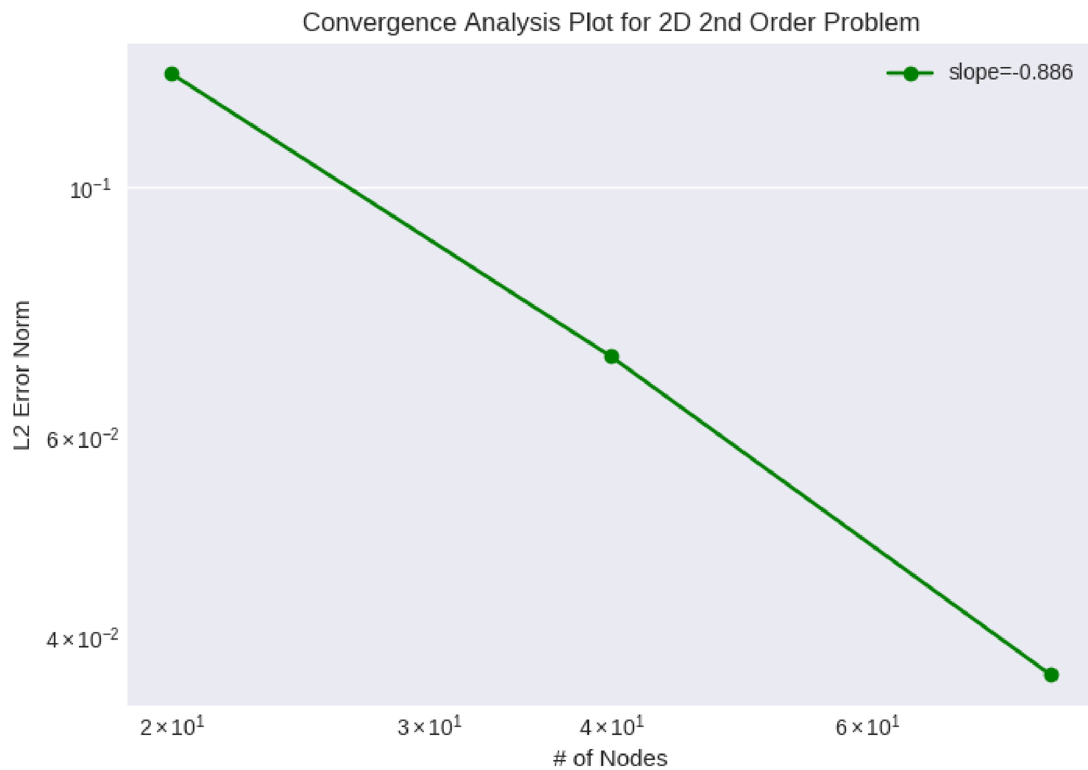

Figure 3: Convergence results for 2D 2nd Order. Not working as expected. Convergence rate is half that of theoretical expectation.

4.2.2 Two Dimensional Case

- $q(x, y)$: Heat flux at spatial coordinate location (x, y)
- x: Spatial coordinate in first dimension.
- y: Spatial coordinate in second dimension
- $T(x, y)$: Temperature of material at spatial coordinate location (x, y)
- k: Material thermal conductivity
- ∇: Gradient operator in 2D Cartesian coordinates

4.3 Boundary Conditions

4.3.1 One Dimensional Case

- $T(x = 0)$: **Required:** Temperature at spatial location $x = 0$.

- $T(x=1)$: **Required:** Temperature at spatial location $x=1$.
- $q(x)$: **Optional:** Heat flux at spatial coordinate location x

4.3.2 Two Dimensional Case

- $T(x=0,y)$: **Required:** Temperature at spatial boundary $(x=0,y)$.
- $T(x=1,y)$: **Required:** Temperature at spatial boundary $(x=1,y)$.
- $T(x,y=0)$: **Required:** Temperature at spatial boundary $(x,y=0)$.
- $T(x,y=1)$: **Required:** Temperature at spatial boundary $(x,y=1)$.
- $q(x,y)$: **Optional:** Heat flux at spatial coordinate location (x,y)

4.4 Constitutive Inputs

4.4.1 One Dimensional Case

- k: **Required:** Material thermal conductivity. Assumed homogeneous across the material domain

4.4.2 Two Dimensional Case

- k **Required:** Material thermal conductivity. Assumed homogeneous across the material domain

5 Assumptions

5.1 One Dimensional Case

- The problem domain is on the interval $\Omega = x \in [0,1]$.
- The boundary conditions are restricted to Dirichlet types, i.e. only temperature can be specified at the domain boundaries, not heat flux.
- Further, these Dirichlet boundary conditions are given, i.e. temperature at mesh nodes on the boundary are specified
- The material thermal conductivity is constant over the domain
- Mesh step size ($\Delta x = h$) is uniform, i.e. no refinement near boundaries, source terms, etc.

5.2 Two Dimensional Case

- The problem domain is the unit square $\Omega = (x, y) \in [0, 1] \times [0, 1]$
- The boundary conditions are restricted to Dirichlet types, i.e. only temperature can be specified at the domain boundaries, not heat flux.
- Further, these Dirichlet boundary conditions are given, i.e. temperature at mesh nodes on the boundary are specified
- The material thermal conductivity is constant over the domain
- Mesh step size ($\Delta x = \Delta y = h$) is uniform, i.e. no refinement near boundaries, source terms, etc.

6 Numerics

6.1 Finite Difference

6.1.1 One Dimensional Case

2nd Order Finite Difference in One Dimension

$$\frac{\partial^2 T(x)}{\partial x^2} = \frac{T(x_{i+1}) - 2T(x_i) + T(x_{i-1})}{\Delta x^2} - \frac{\Delta x^2}{12}\frac{\partial^4 T(x)}{\partial x^4}|_{x_i} + \ldots \tag{3}$$

Expansion of 1D Heat Equation with 2nd Order Finite Difference

$$-k\frac{T(x_{i+1}) - 2T(x_i) + T(x_{i-1})}{\Delta x^2} = q(x_i) \tag{4}$$

4th Order Finite Difference in One Dimension

$$\frac{\partial^2 T(x)}{\partial x^2} = \frac{-T(x_{i+2}) + 16T(x_{i+1}) - 30T(x_i) + 16T(x_{i-1}) - T(x_{i-2})}{12\Delta x^2} - \frac{\Delta x^4}{90}\frac{\partial^6 T(x)}{\partial x^6}|_{x_i} + \ldots \tag{5}$$

Expansion of 1D Heat Equation with 4th Order Finite Difference

$$-k\frac{-T(x_{i+2}) + 16T(x_{i+1}) - 30T(x_i) + 16T(x_{i-1}) - T(x_{i-2})}{12\Delta x^2} = q(x_i) \tag{6}$$

6.1.2 Two Dimensional Case

Finite difference formulas for the second order partial derivatives are the same as in the 1D case. The difference is that there are now two of the terms (one for each spatial direction) in the discrete form of the governing equation

2D Heat Equation with 2nd Order Finite Difference

$$-k\Big(\frac{T(x_{i+1},y_j)-2T(x_i,y_j)+T(x_{i-1},y_j)}{\Delta x^2}+ \frac{T(x_i,y_{j+1})-2T(x_i,y_j)+T(x_i,y_{j-1})}{\Delta y^2}\Big)=q(x_i,y_i) \quad (7)$$

2D Heat Equation with 4th Order Finite Difference

$$-k\Big(\frac{-T(x_{i+2},y_j)+16T(x_{i+1},y_j)-30T(x_i,y_j)+16T(x_{i-1},y_j)-T(x_{i-2},y_j)}{12\Delta x^2}+ \frac{-T(x_i,y_{j+2})+16T(x_i,y_{j+1})-30T(x_i,y_j)+16T(x_i,y_{j-1})-T(x_i,y_{j-2})}{12\Delta y^2}\Big)=q(x_i,y_j) \quad (8)$$

6.2 Meshing

6.2.1 General Notes

- Mesh is grid based

6.2.2 1D Problem

Mesh and finite difference stencil details can be found in Fig. 3

6.2.3 2D Problem

Mesh details can be found in Fig. 4

Finite difference stencil details can be found in Fig. 5

6.3 Linear System

6.3.1 1D Problem

2nd Order Finite Difference The resulting system of equations has the form

$$2T(x_1)-T(x_2)=\frac{h^2}{k}(q(x_1))+T(x_0) \quad (9)$$

$$-T(x_1)+2T(x_2)-T(x_3)=\frac{h^2}{k}q(x_2) \quad (10)$$

$$\vdots \quad (11)$$

$$2T(x_{n_x-1})-T(x_{n_x-2})=\frac{h^2}{k}(q(x_{n_x-1}))+T(x_{n_x}) \quad (12)$$

$$(13)$$

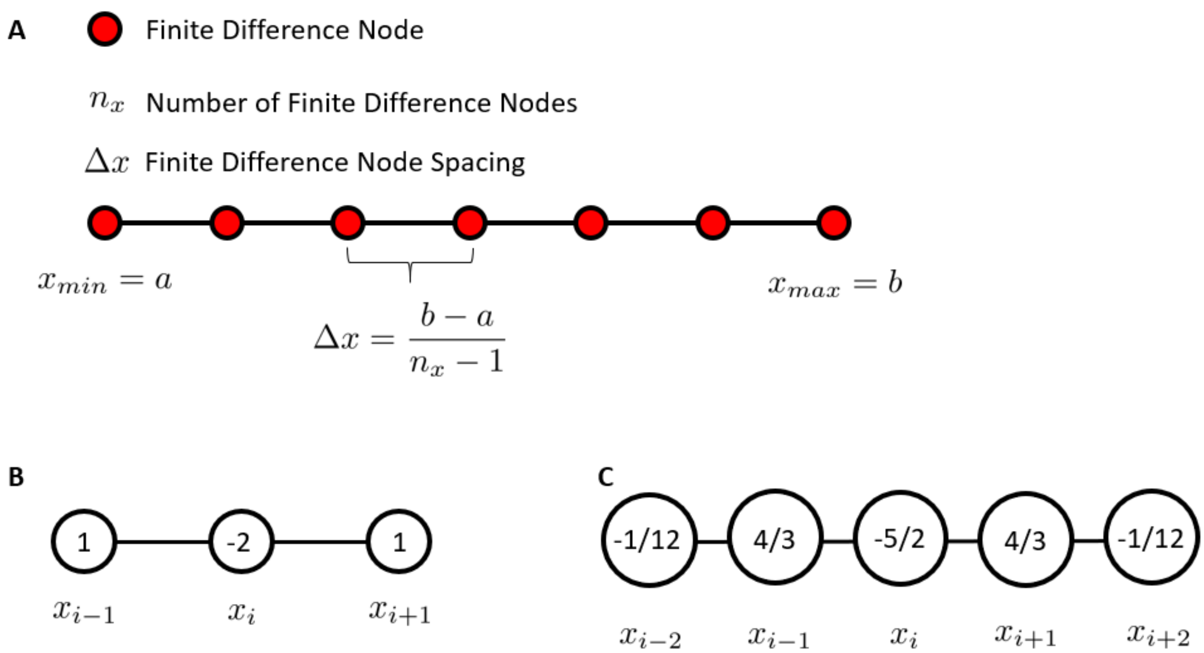

Figure 4: Finite difference mesh and stencil details for the 1D Heat Equation problem. A) Mesh details. B) Stencil for 2nd order finite difference. C) Stencil for 4th order finite difference.

The resulting matrix equation is $AT = Q$ with $A \in \mathbb{R}^{n_x n_y \times n_x n_y}$ given by the matrix

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 & \dots & & 0 \\ -1 & 2 & -1 & 0 & \dots & & 0 \\ 0 & -1 & 2 & -1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & 0 \\ & & & -1 & 2 & -1 & 0 \\ 0 & \dots & \dots & 0 & -1 & 2 & -1 \\ 0 & \dots & \dots & \dots & 0 & -1 & 2 \end{bmatrix}$$

and the RHS vector, $Q \in \mathbb{R}^{n_x}$ by

$$Q = \begin{bmatrix} \frac{h^2}{k} q(x_1) + T(x_0) \\ \frac{h^2}{k} q(x_2) \\ \vdots \\ \frac{h^2}{k} q(x_{n_x - 2}) \\ \frac{h^2}{k} q(x_{n_x - 1}) + T(x_{n_x}) \end{bmatrix}$$

The sparsity of the matrix A is visualized in Fig. 6. There are typically 3 non-zero entries for a interior element row

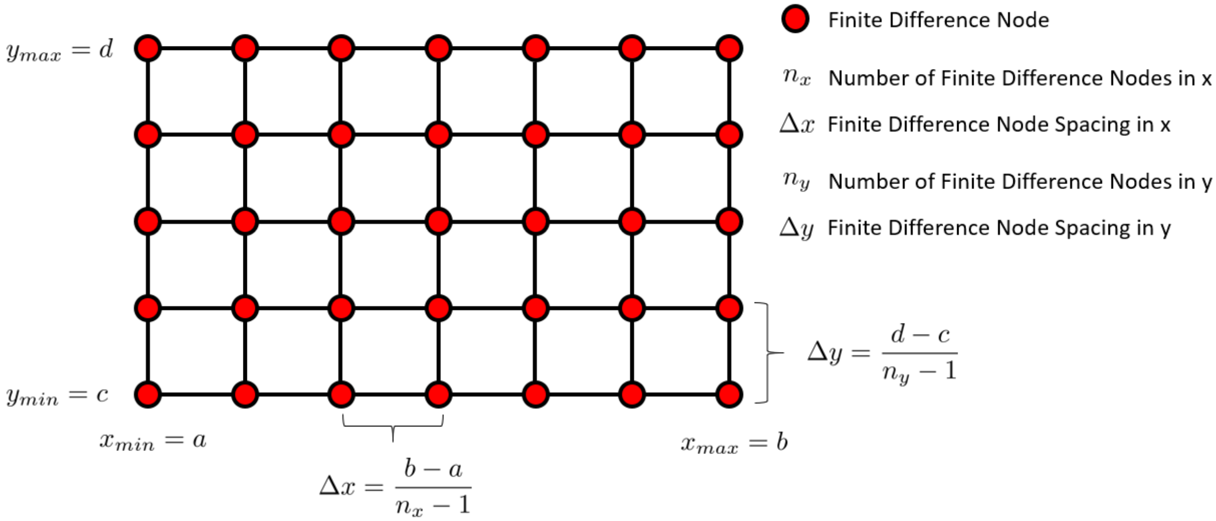

Figure 5: Finite difference mesh details for the 2D Heat Equation problem.

4th Order Finite Difference The resulting system of equations has the form

$$
\begin{aligned}
2T(x_1) - T(x_2) &= \frac{h^2}{k}(q(x_1)) + T(x_0) \\
-\frac{4}{3}T(x_1) + \frac{5}{2}T(x_2) - \frac{4}{3}T(x_3) + \frac{1}{12}T(x_4) &= \frac{h^2}{k}(q(x_2)) - \frac{1}{12}T(x_0) \\
\frac{1}{12}T(x_1) - \frac{4}{3}T(x_2) + \frac{5}{2}T(x_3) - \frac{4}{3}T(x_4) + \frac{1}{12}T(x_5) &= \frac{h^2}{k}q(x_3) \\
&\vdots \\
\frac{1}{12}T(x_{n_x-5}) - \frac{4}{3}T(x_{n_x-4}) + \frac{5}{2}T(x_{n_x-3}) - \frac{4}{3}T(x_{n_x-2}) + \frac{1}{12}T(x_{n_x-1}) &= \frac{h^2}{k}q(x_{n_x-3}) \\
-\frac{4}{3}T(x_{n_x-3}) + \frac{5}{2}T(x_{n_x-2}) - \frac{4}{3}T(x_{n_x-1}) + \frac{1}{12}T(x_{n_x-4}) &= \frac{h^2}{k}(q(x_{n_x-2})) - \frac{1}{12}T(x_{n_x}) \\
2T(x_{n_x-1}) - T(x_{n_x-2}) &= \frac{h^2}{k}(q(x_{n_x-1})) + T(x_{n_x})
\end{aligned}
\tag{14}
$$

where second order finite difference approximations are used on the first interior boundary nodes to deal with the fact that in those locations the 4th order approximations require information from nodes that do not exist.

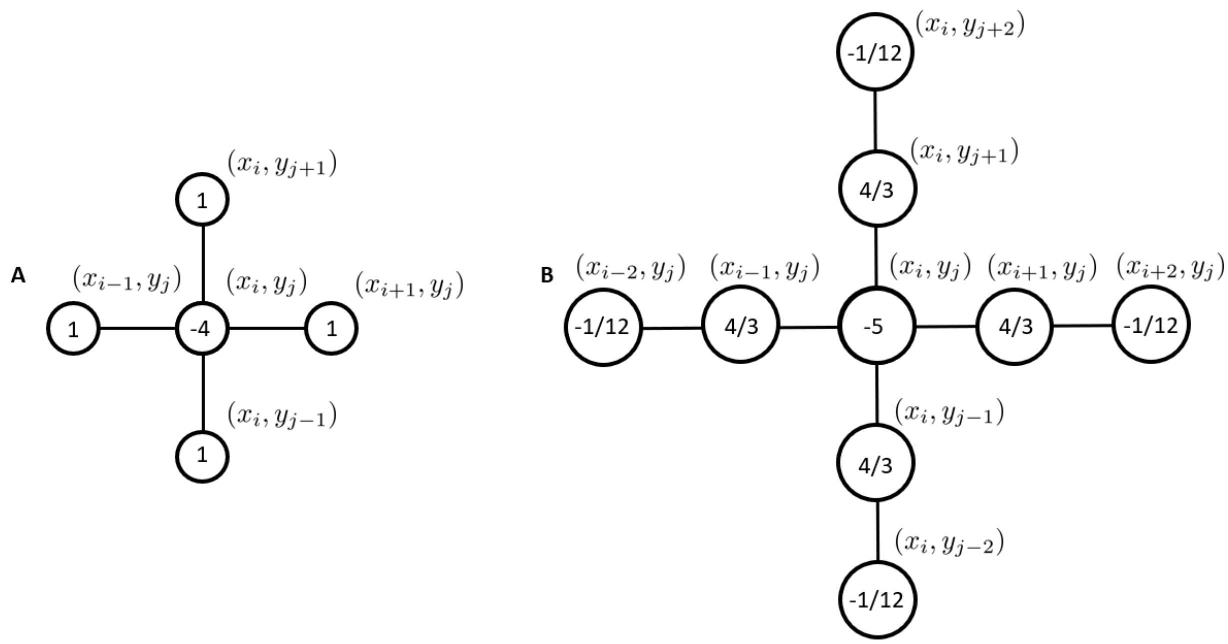

Figure 6: Finite difference mesh details for the 2D Heat Equation problem.

The resulting linear system of equations is $AT = Q$ with $A \in \mathbb{R}^{n_x n_y \times n_x n_y}$ given by the matrix

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 & \dots & & & 0 \\ -\frac{4}{3} & \frac{5}{2} & -\frac{4}{3} & \frac{1}{12} & 0 & \dots & & 0 \\ \frac{1}{12} & -\frac{4}{3} & \frac{5}{2} & -\frac{4}{3} & \frac{1}{12} & 0 & \dots & 0 \\ 0 & \frac{1}{12} & -\frac{4}{3} & \frac{5}{2} & -\frac{4}{3} & \frac{1}{12} & 0 & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & \dots & 0 & \frac{1}{12} & -\frac{4}{3} & \frac{5}{2} & -\frac{4}{3} & \frac{1}{12} \\ 0 & \dots & \dots & 0 & \frac{1}{12} & -\frac{4}{3} & \frac{5}{2} & -\frac{4}{3} \\ 0 & \dots & & & \dots & 0 & -1 & 2 \end{bmatrix}$$

and the RHS vector, $Q \in \mathbb{R}^{n_x}$ by

$$Q = \begin{bmatrix} \frac{h^2}{k} q(x_1) + T(x_0) \\ \frac{h^2}{k} q(x_2) - \frac{1}{12} T(x_0) \\ \frac{h^2}{k} q(x_3) \\ \vdots \\ \frac{h^2}{k} q(x_{n_x-3}) \\ \frac{h^2}{k} q(x_{n_x-2}) - \frac{1}{12} T(x_{n_x}) \\ \frac{h^2}{k} q(x_{n_x-2}) + T(x_{n_x}) \end{bmatrix}$$

The sparsity of the matrix A is visualized in Fig. 7. There are typically 5 non-zero entries for a interior element row

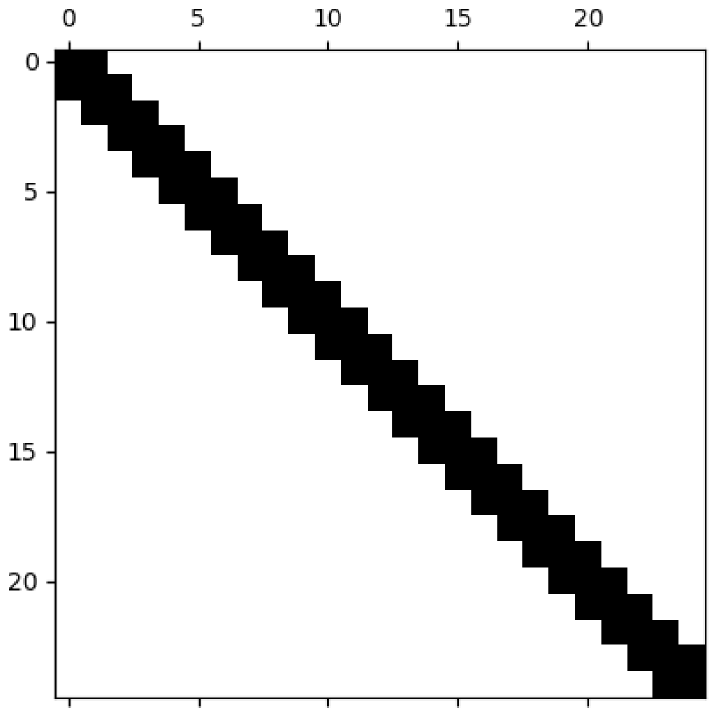

Figure 7: Sparsity of problem system in 1D and 2nd order finite difference for an example 25 degrees of freedom system. Blocks in black indicate non-zero entries

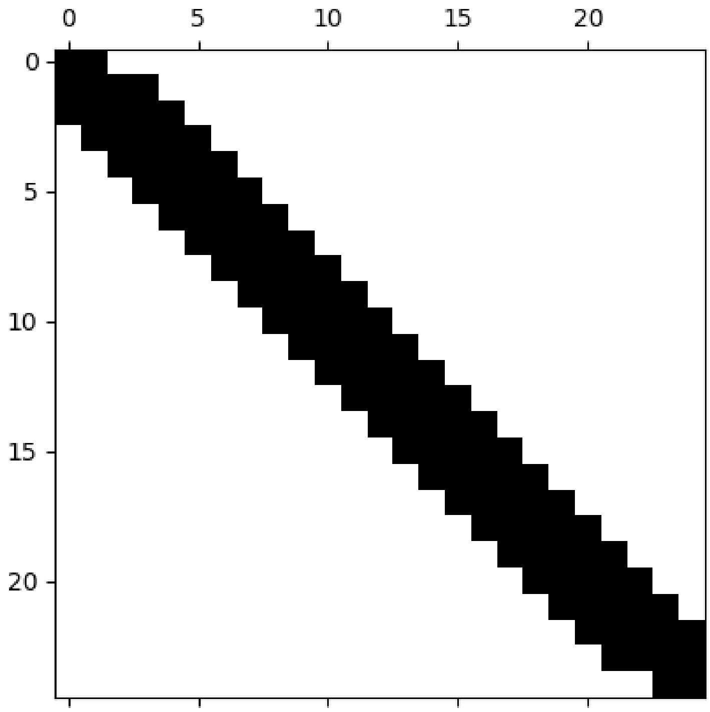

Figure 8: Sparsity of problem system in 1D and 4th order finite difference for an example 25 degrees of freedom system. Blocks in black indicate non-zero entries

6.3.2 2D Problem

2nd Order Finite Difference The resulting system of equations has the form (stepping through x first) and is a simple extension of the 1D problem.

$$\begin{aligned}
4T(x_1,y_1) - T(x_2,y_1) &= \frac{h^2}{k}(q(x_1,y_1)) + T(x_0,y_1) \\
-T(x_1,y_1) + 4T(x_2,y_1) - T(x_3,y_1) &= \frac{h^2}{k}q(x_2,y_1) \\
&\vdots \\
4T(x_{n_x-1},y_1) - T(x_{n_x-2},y_1) &= \frac{h^2}{k}(q(x_{n_x-1},y_1)) + T(x_{n_x},y_1) \\
&\vdots \\
4T(x_1,y_2) - T(x_2,y_2) &= \frac{h^2}{k}(q(x_1,y_2)) + T(x_0,y_2) \\
&\vdots \\
4T(x_{n_x-1},y_2) - T(x_{n_x-2},y_2) &= \frac{h^2}{k}(q(x_{n_x-1}),y_2) + T(x_{n_x},y_2)
\end{aligned} \tag{15}$$

The resulting linear system of equations is $AT = Q$ with $A \in \mathbb{R}^{n_x n_y \times n_x n_y}$ given by the block, tridiagonal matrix

$$A = \begin{bmatrix} S & -I & & & & \\ -I & S & -I & & & \\ & -I & S & -I & & \\ & & \ddots & \ddots & \ddots & \\ & & & -I & S & -I \\ & & & & -I & S \end{bmatrix} S \in \mathbb{R}^{n_x \times n_x} = \begin{bmatrix} 4 & -1 & 0 & 0 & \dots & & 0 \\ -1 & 4 & -1 & 0 & \dots & & 0 \\ 0 & -1 & 4 & -1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & 0 \\ & & & -1 & 4 & -1 & 0 \\ 0 & \dots & \dots & 0 & -1 & 4 & -1 \\ 0 & \dots & \dots & \dots & 0 & -1 & 4 \end{bmatrix}$$

and I is the identity matrix in $\mathbb{R}^{n_x \times n_x}$. The RHS vector, $Q \in \mathbb{R}^{n_x n_y}$ is

$$Q = \begin{bmatrix} \frac{h^2}{k}q(x_1, y_1) + T(x_0, y_1) \\ \frac{h^2}{k}q(x_2, y_1) \\ \vdots \\ \frac{h^2}{k}q(x_{n_x-2}, y_1) \\ \frac{h^2}{k}q(x_{n_x-1}, y_1) + T(x_{n_x}, y_1) \\ \vdots \frac{h^2}{k}q(x_1, y_1) + T(x_0, y_2) \\ \frac{h^2}{k}q(x_2, y_2) \\ \vdots \\ \frac{h^2}{k}q(x_{n_x-2}, y_2) \\ \frac{h^2}{k}q(x_{n_x-1}, y_2) + T(x_{n_x}, y_2) \end{bmatrix}$$

The sparsity of the matrix A is visualized in Fig. 8. There are typically 5 non-zero entries for a interior element row

4th Order Finite Difference As in the 2nd order case, the resulting system of equations has form that is a simple extension of the 1D problem. The resulting linear system of equations is $AT = Q$ with $A \in \mathbb{R}^{n_x n_y \times n_x n_y}$ given by the block, tridiagonal matrix

$$A = \begin{bmatrix} S & -I & & & & \\ -I & S & -I & & & \\ & -I & S & -I & & \\ & & \ddots & \ddots & \ddots & \\ & & & -I & S & -I \\ & & & & -I & S \end{bmatrix} S \in \mathbb{R}^{n_x \times n_x} = \begin{bmatrix} 4 & -1 & 0 & 0 & \dots & & & 0 \\ -\frac{4}{3} & 5 & -\frac{4}{3} & \frac{1}{12} & 0 & \dots & & 0 \\ \frac{1}{12} & -\frac{4}{3} & 5 & -\frac{4}{3} & \frac{1}{12} & 0 & \dots & 0 \\ 0 & \frac{1}{12} & -\frac{4}{3} & 5 & -\frac{4}{3} & \frac{1}{12} & 0 & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & \dots & 0 & \frac{1}{12} & -\frac{4}{3} & 5 & -\frac{4}{3} & \frac{1}{12} \\ 0 & \dots & \dots & 0 & \frac{1}{12} & -\frac{4}{3} & 5 & -\frac{4}{3} \\ 0 & \dots & & & \dots & 0 & -1 & 4 \end{bmatrix}$$

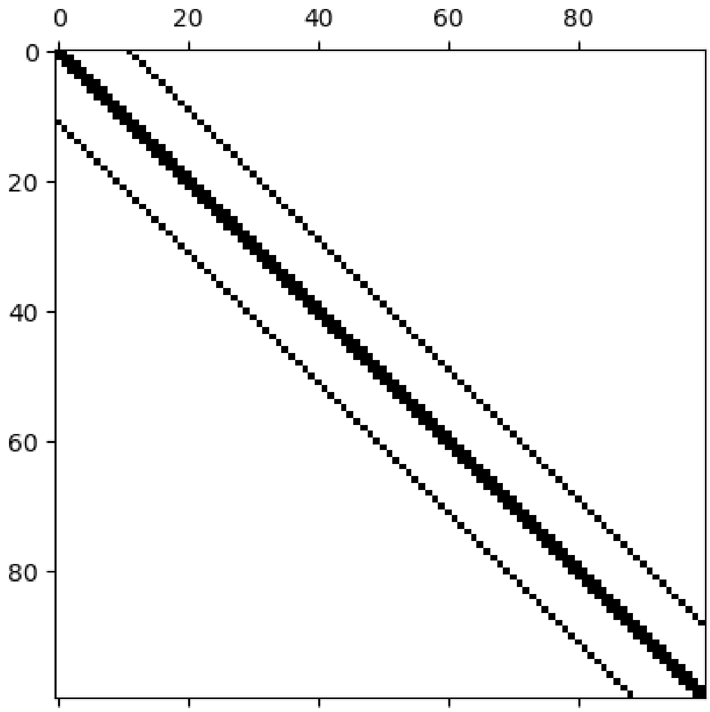

Figure 9: Sparsity of problem system in 2D and 2nd order finite difference for an example 10 x 10 degrees of freedom system. Blocks in black indicate non-zero entries

and I is the identity matrix in $\mathbb{R}^{n_x \times n_x}$. The RHS vector, $Q \in \mathbb{R}^{n_x n_y}$ is

$$Q = \begin{bmatrix} \frac{h^2}{k} q(x_1, y_1) + T(x_0, y_1) \\ \frac{h^2}{k} q(x_2, y_1) - \frac{1}{12} T(x_0, y_1) \\ \frac{h^2}{k} q(x_3, y_1) \\ \vdots \\ \frac{h^2}{k} q(x_{n_x-3}, y_1) \\ \frac{h^2}{k} q(x_{n_x-2}, y_1) - \frac{1}{12} T(x_{n_x}, y_1) \\ \frac{h^2}{k} q(x_{n_x-2}, y_1) + T(x_{n_x}, y_1) \\ \frac{h^2}{k} q(x_1, y_2) + T(x_0, y_2) \\ \frac{h^2}{k} q(x_2, y_2) - \frac{1}{12} T(x_0, y_2) \\ \frac{h^2}{k} q(x_3, y_2) \\ \vdots \\ \frac{h^2}{k} q(x_{n_x-3}, y_2) \\ \frac{h^2}{k} q(x_{n_x-2}, y_2) - \frac{1}{12} T(x_{n_x}, y_2) \\ \frac{h^2}{k} q(x_{n_x-2}, y_2) + T(x_{n_x}, y_2) \end{bmatrix}$$

The sparsity of the matrix A is visualized in Fig. 9. There are typically 7 non-zero entries for a interior element row

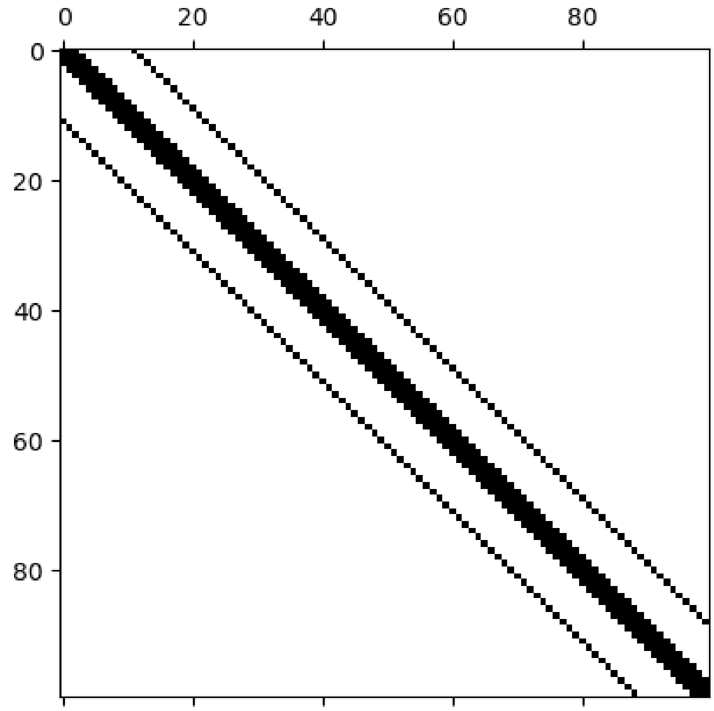

Figure 10: Sparsity of problem system in 2D and 4th order finite difference for an example 10 x 10 degrees of freedom system. Blocks in black indicate non-zero entries

6.4 Solvers

6.4.1 Jacobi

Algorithm Outline Given $Ax = B$, decompose A into $A = D + R$ with D being the diagonal and R the off diagonal terms. Solve for next iterative guess of x via

$$x^{k+1} = D^{-1}(b - Rx^k) \tag{16}$$

and terminate when difference norm term $||x^k - x^{k-1}||$ falls below user specified tolerance ϵ. User must also specify an initial guess x^0.

Psuedo-Code

```
k = 0;
error = ||b − Ax^(k)||_2 ;
while error > epsilon do
    for i = 1 to n do
        σ = 0;
        for j = 1 to n do
            if j ≠ i then
                σ = σ + a_ij x_j^(k);
            end
        end
        x_i^(k+1) = (1/a_ii)(b_i − σ) ;
    end
    k = k + 1 ;
    error = ||b − Ax^(k+1)||_2 ;
end
```

6.4.2 Gauss-Seidel

Algorithm Outline Given $Ax = B$, decompose A into $A = L + R$ with L being lower triangular and U upper triangular. Solve for next iterative guess of x via

$$x^{k+1} = L^{-1}(b - Ux^k) \tag{17}$$

and terminate when difference norm term $||x^k - x^{k-1}||$falls below user specified tolerance ϵ. Note, this method only requires storage of a single vector sized to the number of variables, whereas the Jacobi requires storage of two such vecotrs. User must also specify an initial guess ϕ.

Psuedo-Code

```
error = ||b − Aφ||_2 ;
while error > epsilon do
    for i = 1 to n do
        σ = 0;
        for j = 1 to n do
            if j ≠ i then
                σ = σ + a_ii φ_j;
            end
        end
        φ_i = (1/a_ii)(b_i − σ) ;
    end
    error = ||b − Aφ||_2 ;
end
```

6.5 Memory Considerations

A conservative memory estimate would be to calculate the memory required to store each element of the matrix A and vectors T, Q in double precision floating point as a function of the problem size. Count the T vector twice to accommodate the need to store two of these vectors if using Jacobi solution method. Note, the storage could be vastly improved if matvec storage is used for A.

6.5.1 1D Problem

- A: $n_x \times n_x$ floating point numbers
- Q: n_x floating point numbers
- T: n_x floating point numbers

Total Floating Point Numbers: $n_x^2 + 3n_x$

Memory estimation: 64 bits $*(n_x^2 + 3n_x)$

6.5.2 2D Problem

- A: $n_xn_y \times n_xn_y$ floating point numbers
- Q: n_xn_y floating point numbers
- T: n_xn_y floating point numbers

Total Floating Point Numbers: $(n_xn_y)^2 + 3(n_xn_x)$

Memory estimation: 64 bits $*((n_xn_y)^2 + 3(n_xn_x))$