

## Modeling Document

### • Governing Equations

$$\begin{aligned} -k\nabla^2 T(x, y) &= q(x, y) \text{ in } \Omega \\ T(x, y) &= T_{masa}(x, y) \text{ on } \partial\Omega \end{aligned} \tag{1}$$

where,

$\Omega$  is a 2D bounded domain

$\partial\Omega$  is the boundary of the domain

$T$  is the material's temperature field

$q$  is the heat source term

$k$  is the thermal conductivity

We have the Dirichlet boundary conditions.

### • Assumptions

- The thermal conductivity is assumed to be constant.
- We assume a square domain  $\Omega = \{ (x, y) : x \in [0, L], y \in [0, L] \}$  for the 2D case. For 1D, it will obviously be a line  $\Omega = \{ (x, y) : x \in [0, L] \}$
- Dirichlet boundary condition is assumed at the boundaries
- For the fourth order scheme, we assume that the values at the points adjacent to the boundary points are known from the MASA solution. This is to reduce the cumbersome effort required to come up with different schemes at the boundary.
- For the 2D case we assume symmetrical discretization i.e. the number of grid points in both directions are the same and  $\Delta x = \Delta y = h$ .

### • Nomenclature for discretization

Our numerical methods are all node based (as will be reiterated later).

- 1D We have  $(N + 1)$  points  $\{x_0, x_1, x_2, \dots x_N\}$  in the x-direction with  $x_i = i\Delta x$ , where  $\Delta x = L/N$
- 2D We have  $(N + 1)$  points  $\{x_0, x_1, x_2, \dots x_N\}$  in the x-direction with  $x_i = i\Delta x$ , where  $\Delta x = L/N = h$  and  $(N + 1)$  points  $\{y_0, y_1, y_2, \dots y_N\}$  in the y-direction with  $y_j = j\Delta y$ , where  $\Delta y = L/N = h$ . Hence, we have a  $(N + 1) \times (N + 1)$  grid.
- $i$  is always associated with the indexing in x-direction and  $j$  is always associated with the indexing in y-direction.

- $T(x_i, y_j)$  is given the shorthand notation  $T(i, j)$  and  $q(x_i, y_j)$  is given the shorthand notation  $q(i, j)$
- The composite index for 2D grid is given by  $k = i + (N + 1)j$ .

## • Numerical Method

Our numerical methods are all node based (as will be reiterated later).

### – 2<sup>nd</sup> order finite difference approximation

#### Definition

$$\frac{\partial^2 T}{\partial x^2} \approx \frac{T(x + \Delta x) - 2T(x) + T(x - \Delta x)}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

#### Discretized heat equation

##### 1. 1D

$$\begin{cases} -k \left( \frac{T(i+1) - 2T(i) + T(i-1))}{\Delta x^2} \right) + \mathcal{O}(\Delta x^2) = q(i), & i \in \{1, 2, 3, \dots, N-1\} \\ T(0) = T_{masa}(0) \text{ and } T(N) = T_{masa}(L) \end{cases} \quad (2)$$

##### 2. 2D

$$\begin{cases} -k \left( \frac{T(i+1, j) - 2T(i, j) + T(i-1, j))}{h^2} \right) - k \left( \frac{T(i, j+1) - 2T(i, j) + T(i, j-1))}{h^2} \right) \\ + \mathcal{O}(h^2) = q(i, j), & i \in \{1, 2, 3, \dots, N-1\} \text{ } j \in \{1, 2, 3, \dots, N-1\} \\ T(i, j) = T_{masa}(x_i, y_j) \text{ for } i \in \{0, N\}, j \in \{0, 1, 2, \dots, N\} \\ T(i, j) = T_{masa}(x_i, y_j) \text{ for } i \in \{0, 1, 2, \dots, N\}, j \in \{0, N\} \end{cases} \quad (3)$$

### – 4<sup>th</sup> order finite difference approximation

#### Definition

$$\frac{\partial^2 T}{\partial x^2} \approx \frac{-T(x - 2\Delta x) + 16T(x - \Delta x) - 30T(x) + 16T(x + \Delta x) - T(x + 2\Delta x)}{12\Delta x^2} + \mathcal{O}(\Delta x^4)$$

#### Discretized heat equation

We have Dirichlet boundary conditions at both the boundary points and adjacent to boundary points.

##### 1. 1D

$$\begin{cases} -k \left( \frac{-T(i-2) + 16T(i-1) - 30T(i) + 16T(i+1) - T(i+2))}{12\Delta x^2} \right) + \mathcal{O}(\Delta x^4) = q(i), & i \in \{2, 3, \dots, N-2\} \\ T(i) = T_{masa}(x_i), & i \in \{0, 1, N-1, N\} \end{cases} \quad (4)$$

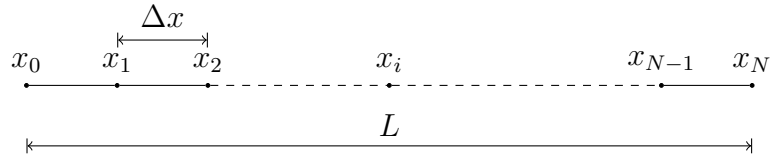
## 2. 2D

$$\left\{ \begin{array}{l} -k \left( \frac{-T(i-2,j)+16T(i-1,j)-30T(i,j)+16T(i+1,j)-T(i+2,j)}{12h^2} \right) \\ -k \left( \frac{-T(i,j-2)+16T(i,j-1)-30T(i,j)+16T(i,j+1)-T(i,j+2)}{12h^2} \right) + \mathcal{O}(h^4) = q(i,j), \\ i \in \{2, \dots, N-2\}, j \in \{2, \dots, N-2\} \\ \\ T(i,j) = T_{masa}(x_i, y_j) \text{ for } i \in \{0, 1, N-1, N\}, j \in \{0, 1, 2, \dots, N\} \\ T(i,j) = T_{masa}(x_i, y_j) \text{ for } i \in \{0, 1, 2, \dots, N\}, j \in \{0, 1, N-1, N\} \end{array} \right. \quad (5)$$

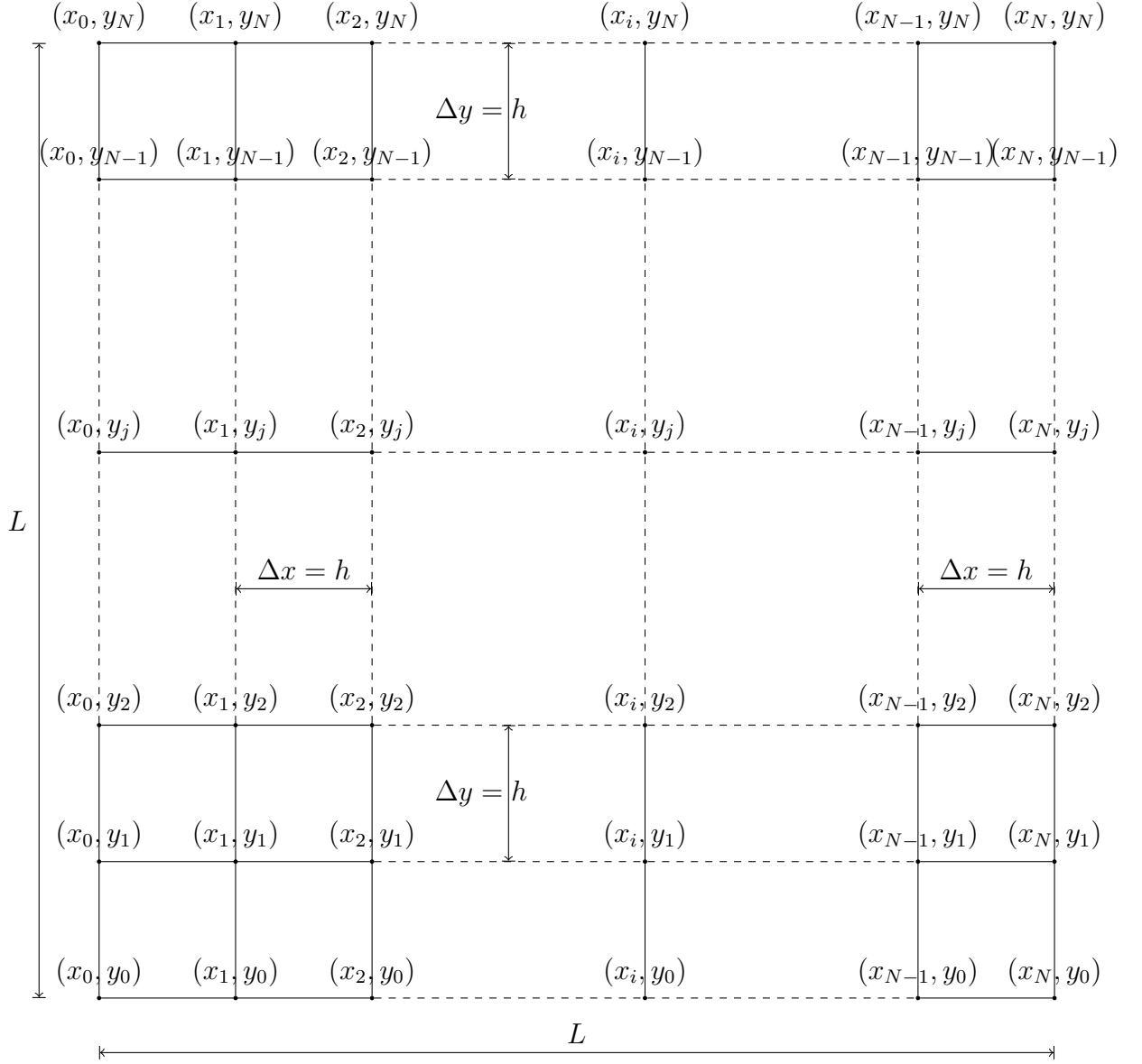
### • Mesh diagrams

Our schemes are node based.

#### 1. 1D



#### 2. 2D



- **Linear system of Equations**

- **2<sup>nd</sup> order finite difference approximation**

1. 1D

We first define the following vectors

$$\mathbf{q} = \left[ T_{masa}(0), \frac{\Delta x^2}{k} q(1), \dots, \frac{\Delta x^2}{k} q(N-1), T_{masa}(L) \right]^T$$

$$\mathbf{T} = [T(0), \dots, T(N)]^T$$

We now define a  $(N + 1) \times (N + 1)$  tridiagonal matrix  $\mathbf{A}$  such that,

$$\mathbf{A} = \begin{bmatrix} 1 & & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & & 1 \end{bmatrix}$$

Now (??) can be written as,

$$\mathbf{A}\mathbf{T} = \mathbf{q}$$

The first and last rows are different because they account for boundary conditions. The sparsity pattern of  $\mathbf{A}$  can be given by,

$$\mathbf{A} = \begin{bmatrix} \times & & & & & & \\ \times & \times & \times & & & & \\ & \times & \times & \times & & & \\ & & \times & \times & \times & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \times & \times & \times \\ & & & & & \times & \times & \times \\ & & & & & & & \times \end{bmatrix}$$

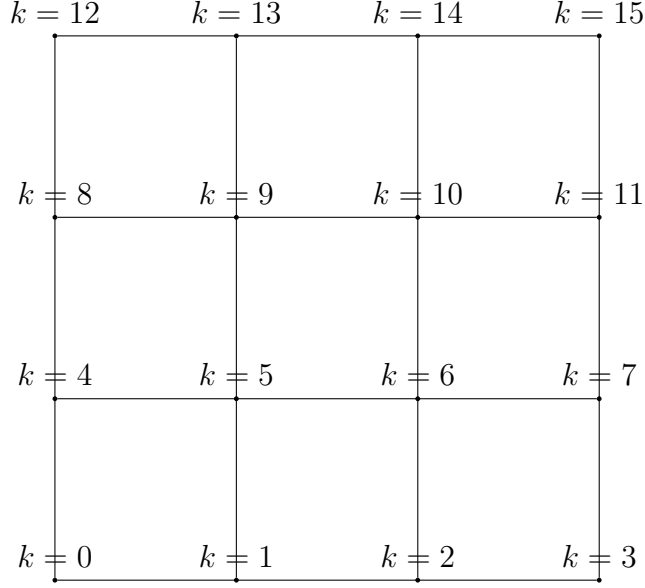
Number of non-zero elements on an interior row of the matrix = 3

## 2. 2D

The elements will be numbered in the following fashion

$$k = i + (N + 1)j, \quad i \in [0, \dots, N], \quad j \in [0, \dots, N]$$

Here is an illustration for a  $4 \times 4$  grid i.e.  $N = 3$ ,



So we can imagine how the matrix is going to look: the terms of second derivative in x-direction will be adjacent to each other, but they terms of the second derivative in y-direction will be offset by  $N$  points. This will become clear in the visual representation below. We first define the following  $(N + 1)^2 \times 1$  vectors

$$\mathbf{q}(k) = \begin{cases} \frac{h^2}{k} q(i, j) & \text{if } i \in \{1, 2, \dots, N - 1\}, j \in \{1, 2, \dots, J - 1\} \\ T_{masa}(x_i, y_j) & \text{otherwise i.e. at boundaries} \end{cases}$$

$$\mathbf{T} = [T(0), T(1), \dots, T((N + 1)^2)]^T$$

We now define  $(N + 1)^2 \times (N + 1)^2$  matrices  $\mathbf{A}_x$  (interior x-direction derivatives),  $\mathbf{A}_y$  (interior y-direction derivatives) and  $\mathbf{A}_b$  (dirichlet nodes) such that,

$$\mathbf{A}_x = \begin{cases} \mathbf{A}_x(i, i) = \begin{cases} 2 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i - 1) = \begin{cases} -1 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i + 1) = \begin{cases} -1 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

$$\mathbf{A}_y = \begin{cases} \mathbf{A}_y(j, j) = \begin{cases} 2 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j - N) = \begin{cases} -1 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j + N) = \begin{cases} -1 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

$$\mathbf{A}_b = \begin{cases} \mathbf{A}_b(i, i) = \begin{cases} 1 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

Now, the net matrix is  $\mathbf{A} = \mathbf{A}_x + \mathbf{A}_y + \mathbf{A}_b$ . The sparsity pattern of  $\mathbf{A}$  is given by, (illustration for a  $5 \times 5$  grid)

$$\mathbf{A} = \begin{bmatrix} \times & & & & \\ & \times & & & \\ & & \times & & \\ & & & \times & \\ & & & & \times \\ & \times & & & \\ & & \times & & \\ & & & \times & \\ & & & & \times \\ & & & & & \times \\ & & & & & & \times \\ & & & & & & & \times \\ & & & & & & & & \times \\ & & & & & & & & & \times \\ & & & & & & & & & & \times \\ & & & & & & & & & & & \times \\ & & & & & & & & & & & & \times \\ & & & & & & & & & & & & & \times \\ & & & & & & & & & & & & & & \times \\ & & & & & & & & & & & & & & & \times \\ & & & & & & & & & & & & & & & & \times \end{bmatrix}$$

Number of non-zero elements on an interior row of the matrix = 5.

The repeating interior block of  $\mathbf{A}$  is given by,

$$\mathbf{A} = \begin{bmatrix} & & 1 & & & & \\ -1 & \dots & -1 & 4 & -1 & \dots & -1 \\ & -1 & \dots & -1 & 4 & -1 & \dots & -1 \\ & & -1 & \dots & -1 & 4 & -1 & \dots & -1 \\ & & & & & & 1 & & \end{bmatrix}$$

Now (??) can be written as,

$$\mathbf{AT} = \mathbf{q}$$

– 4<sup>th</sup> order finite difference approximation

1. 1D

We first define the following vectors

$$\mathbf{q} = \left[ T_{masa}(0), T_{masa}(x_1), \frac{12\Delta x^2}{k}q(3) \dots, \frac{12\Delta x^2}{k}q(N-2), T_{masa}(x_{N-1}), T_{masa}(L) \right]^T$$

$$\mathbf{T} = [T(0), \dots, T(N)]^T$$

We now define a  $(N+1) \times (N+1)$  pentadiagonal matrix  $\mathbf{A}$  such that (blank elements are 0),

$$\mathbf{A} = \begin{bmatrix} 1 & & & & & & & & & & \\ & 1 & & & & & & & & & \\ 1 & -16 & 30 & -16 & 1 & & & & & & \\ & 1 & -16 & 30 & -16 & 1 & & & & & \\ & & 1 & -16 & 30 & -16 & 1 & & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & & 1 & -16 & 30 & -16 & 1 & & \\ & & & & & 1 & -16 & 30 & -16 & 1 & \\ & & & & & & & 1 & & & \\ & & & & & & & & 1 & & \\ & & & & & & & & & 1 & \\ & & & & & & & & & & 1 \end{bmatrix}$$

Now (??) can be written as,

$$\mathbf{AT} = \mathbf{q}$$

Note that the first and last two rows in  $\mathbf{A}$  looks different because we accounted for the Dirichlet conditions. The sparsity pattern of the matrix is given by,

$$\mathbf{A} = \begin{bmatrix} \times & & & & & & & & & & \\ & \times & & & & & & & & & \\ \times & \times & \times & \times & \times & & & & & & \\ & \times & \times & \times & \times & \times & & & & & \\ & & \times & \times & \times & \times & \times & & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & & \times & \times & \times & \times & \times & & \\ & & & & & \times & \times & \times & \times & \times & \\ & & & & & & & & \times & & \\ & & & & & & & & & \times & \end{bmatrix}$$

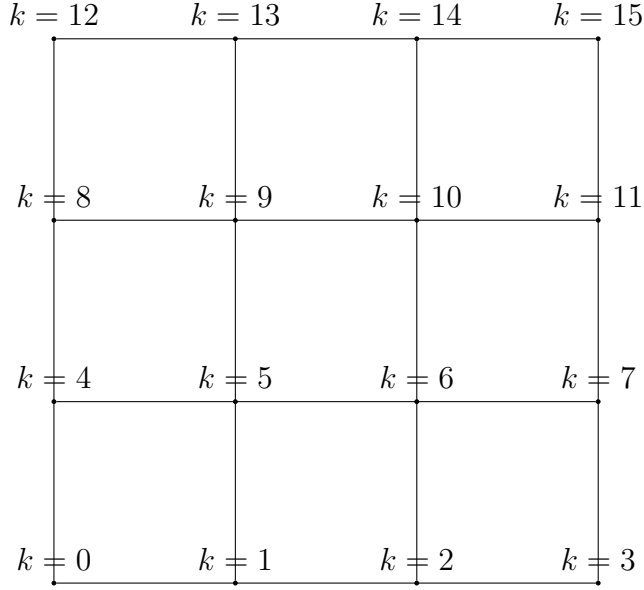
Number of non-zero elements on an interior row of the matrix = 5.

2. 2D The elements will be numbered in the following fashion

$$k = i + (N+1)j, \quad i \in [0, \dots, N], \quad j \in [0, \dots, N]$$



Here is an illustration for a  $4 \times 4$  grid i.e.  $N = 3$ ,



So we can imagine how the matrix is going to look: the terms of second derivative in x-direction will be adjacent to each other, but they terms of the second derivative in y-direction will be offset by  $N$  points. This will become clear in the visual representation below. We first define the following  $(N + 1)^2 \times 1$  vectors

$$\mathbf{q}(k) = \begin{cases} \frac{12h^2}{k}q(i, j) & \text{if } i \in \{1, 2, \dots, N - 1\}, j \in \{1, 2, \dots, N - 1\} \\ T_{masa}(x_i, y_j) & \text{otherwise} \end{cases}$$

$$\mathbf{T} = [T(0), T(1), \dots, T((N + 1)^2)]^T$$

We now define  $(N + 1)^2 \times (N + 1)^2$  matrices  $\mathbf{A}_x$  (interior x-direction derivatives),  $\mathbf{A}_y$  (interior y-direction derivatives) and  $\mathbf{A}_b$  (boundary and close to boundary elements) such that,

$$\mathbf{A}_x = \begin{cases} \mathbf{A}_x(i, i) = \begin{cases} 30 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i-1) = \begin{cases} -16 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i+1) = \begin{cases} -16 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i+2) = \begin{cases} 1 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_x(i, i-2) = \begin{cases} 1 & \text{if } i \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

$$\mathbf{A}_y = \begin{cases} \mathbf{A}_y(j, j) = \begin{cases} 30 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j-N) = \begin{cases} -16 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j+N) = \begin{cases} -16 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j+2N) = \begin{cases} 1 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \\ \mathbf{A}_y(j, j-2N) = \begin{cases} 1 & \text{if } j \text{ corresponds to an interior point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

$$\mathbf{A}_b = \begin{cases} \mathbf{A}_b(j, j) = \begin{cases} 1 & \text{if } j \text{ is a boundary point or an adjacent to boundary point} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

Now, the net matrix is  $\mathbf{A} = \mathbf{A}_x + \mathbf{A}_y + \mathbf{A}_b$ . The sparsity pattern of  $\mathbf{A}$  is given

by, (illustration for a  $7 \times 7$  grid, which means the interior matrix is  $3 \times 3$ )

$$\mathbf{A} = \begin{bmatrix} & & & & & & \\ & & & & & & \\ & & \ddots & & & & \\ & & & \times & & & \\ & & & & \times & & \\ & \times & & & & \times & \\ & & \times & & & & \times \\ & & & \times & \times & \times & \times \\ & & & & \times & \times & \times \\ & & & & & \times & \times \\ & & & & & & \times \\ & & & & & & & \times \\ & & & & & & & & \times \end{bmatrix}$$

Number of non-zero elements on an interior row of the matrix = 9.

The repeating interior block of  $\mathbf{A}$  is given by,

$$\mathbf{A} = \begin{bmatrix} & & & 1 & & & & & & & & & \\ & & & & 1 & & & & & & & & \\ 1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots & -16 & \dots & 1 \\ & & 1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots & -16 \\ & & & 1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & 1 & & \\ & & & & & & & & & & & 1 & \end{bmatrix}$$

Now (??) can be written as,

$$\mathbf{AT} = \mathbf{q}$$

## • Iterative solvers

Here is the pseudo-code for dense matrix solvers (the actual implementation might be for sparse matrices or not, this is not decided yet).

### 1. Jacobi

```
subroutine jacobi (A,q, TOL, max_iter)
!!! Find T = inv(A)*q using Jacobi iteration

K = size(q)
iters = 0 !!! Number of iterations
error = 0 !!! Compare to tolerance
T(1:K) = 0 !!! Initialize entire T array to zero

do while (iters <= max_iter)

    T_old(:) = T(:)

    do i = 1,...,K
```

```

        !!! We use only the old values and none of the ←
        updated values for the update.

        T(i) = 1/A(i,i) * (q(i)- $\sum_{j \neq i}^K A(i,j)T_{old}(j)$ )
    end do

    error =  $\frac{\|T_{old}-T\|}{\|T\|}$ 

    iters = iters + 1

    if (error <= TOL)
        break
    end if

end do

return T
end subroutine

```

## 2. Gauss-Seidel

```

subroutine gauss_seidel (A,q, TOL, max_iter)
    !!! Find T = inv(A)*q using Gauss Seidel iteration

    K = size(q)
    iters = 0 !!! Number of iterations
    error = 0 !!! Compare to tolerance
    T(1:K) = 0 !!! Initialize entire T array to zero

    do while (iters <= max_iter)

        T_old(:) = T(:)

        do i = 1,...,K
            !!! T_old is not used at all, we use older values ←
            for >i and new values for <i.

                T(i) = 1/A(i,i) * (q(i)- $\sum_{j \neq i}^K A(i,j)T(j)$ )
            end do

            error =  $\frac{\|T_{old}-T\|}{\|T\|}$ 
            iters = iters + 1
            if (error <= TOL)
                break
            end if

        end do

    end do

    return T
end subroutine

```

- **Estimate of memory requirements**

We assume that our matrices are stored as normal, dense matrices (assuming that our iterative solvers have generic, dense implementations).

– Second order approximation

\* 1D

Variable	Dimension	Memory
T	$N+1$	$8 \times (N+1)$
q	$N+1$	$8 \times (N+1)$
A	$(N+1) \times (N+1)$	$8 \times (N+1)^2$
N	1	4
L	1	8
$\Delta x$	1	8
k	1	8
	Total	$8(N+2)^2 + 20$

\* 2D

Variable	Dimension	Memory
T	$(N+1)(N+1)$	$8 \times (N+1)^2$
q	$(N+1)(N+1)$	$8 \times (N+1)^2$
A	$(N+1)(N+1) \times (N+1)(N+1)$	$8 \times (N+1)^4$
N	1	4
L	1	8
$\Delta x$	1	8
$\Delta y$	1	8
k	1	8
	Total	$8(N+1)^4 + 16(N+1)^2 + 36$

– Fourth order approximation

\* 1D

Variable	Dimension	Memory
T	$N+1$	$8 \times (N+1)$
q	$N+1$	$8 \times (N+1)$
A	$(N+1) \times (N+1)$	$8 \times (N+1)^2$
N	1	4
L	1	8
$\Delta x$	1	8
k	1	8
	Total	$8(N+2)^2+20$

\* 2D

Variable	Dimension	Memory
T	$(N+1)(N+1)$	$8 \times (N+1)^2$
q	$(N+1)(N+1)$	$8 \times (N+1)^2$
A	$(N+1)(N+1) \times (N+1)(N+1)$	$8 \times (N+1)^4$
N	1	4
L	1	8
$\Delta x$	1	8
$\Delta y$	1	8
k	1	8
	Total	$8(N+1)^4+16(N+1)^2+36$