Modeling Document

- **Governing Equations**

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

where,

Ω 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 $\{x_0, x_1, x_2, \ldots x_N\}$ in the x-direction with $x_i = i\Delta x$, where $\Delta x = L/N$
  - 2D We have $(N+1)$ points $\{x_0, x_1, x_2, \ldots x_N\}$ in the x-direction with $x_i = i\Delta x$, where $\Delta x = L/N = h$ and $(N+1)$ points $\{y_0, y_1, y_2, \ldots y_N\}$ 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).

  - $2^{\textbf{nd}}$ **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

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

    2. 2D

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

  - $4^{\textbf{th}}$ **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

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

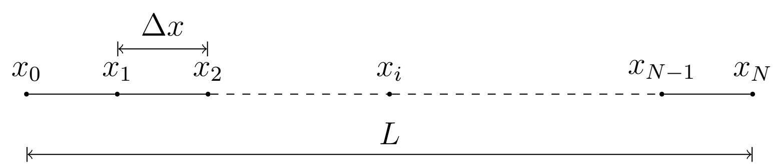
2. 2D

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

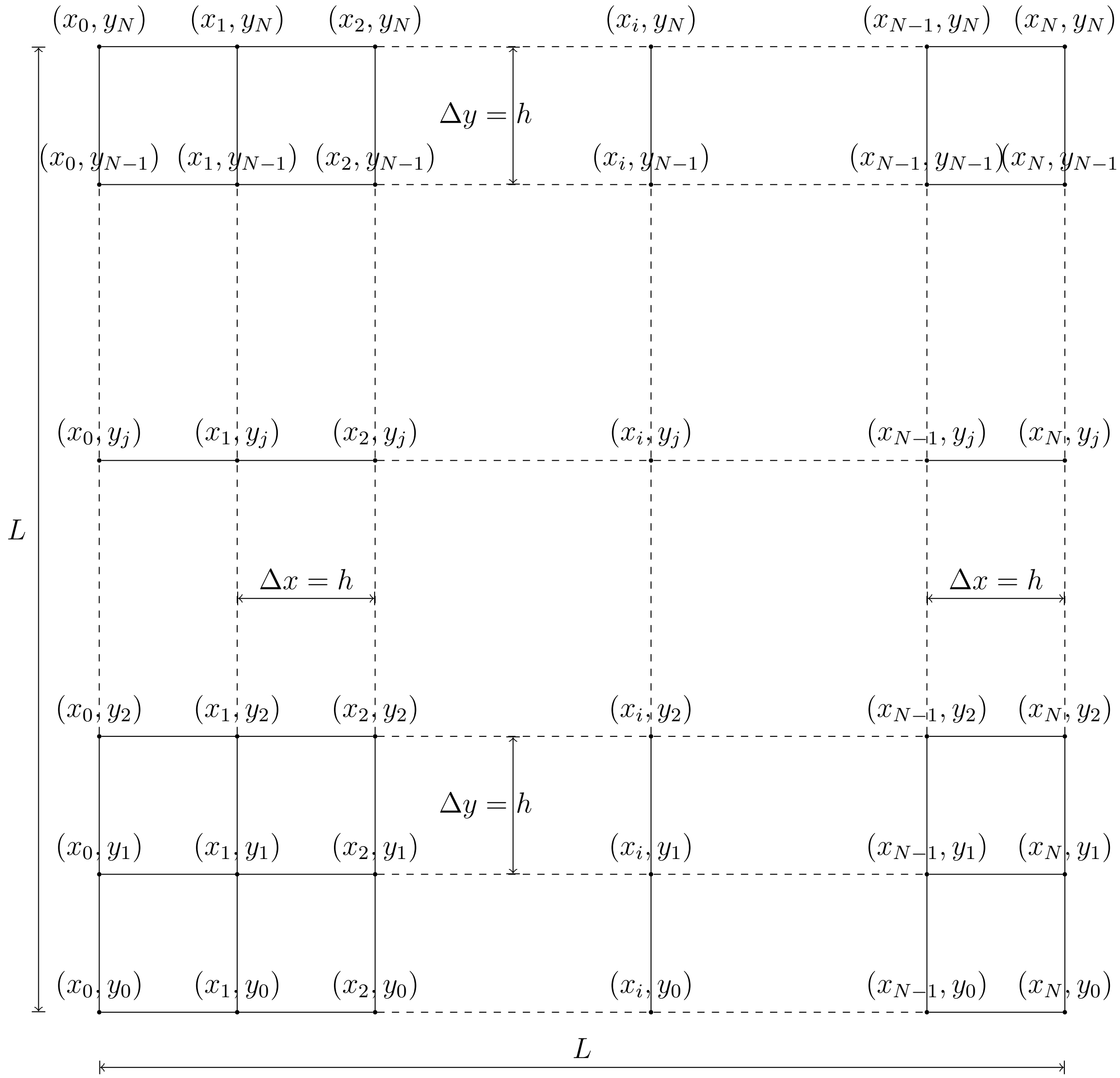
- **Mesh diagrams**

  Our schemes are node based.

  1. 1D



  2. 2D



- **Linear system of Equations**
  - 2^{nd} **order finite difference approximation**
    1. 1D

       We first define the following vectors

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

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 (2) can be written as,

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

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

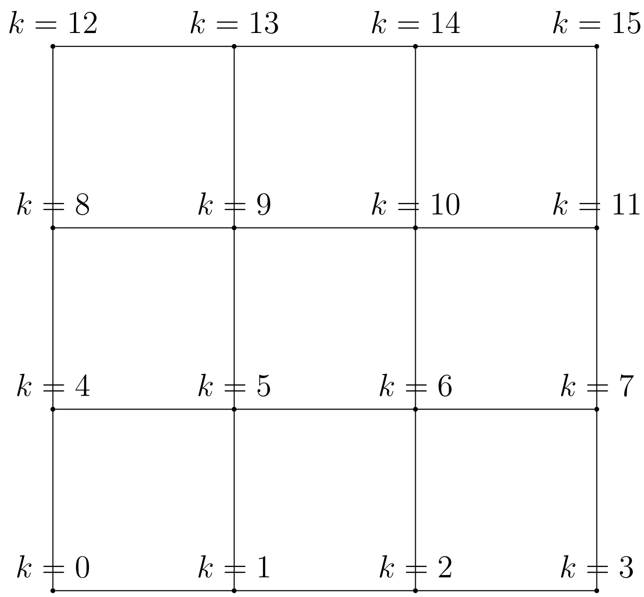
$$\mathbf{A} = \begin{bmatrix} \times & & & & & & \\ \times & \times & \times & & & & \\ & \times & \times & \times & & & \\ & & \times & \times & \times & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \times & \times & \times \\ & & & & & \times & \times & \times \\ & & & & & & & \times \end{bmatrix}$$

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

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

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

   Here is an illustration for a 4×4 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

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

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

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 5×5 grid)

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

`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 (3) can be written as,

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

– 4^{th} **order finite difference approximation**

1. 1D
   We first define the following vectors

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

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

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

   Now (4) 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,

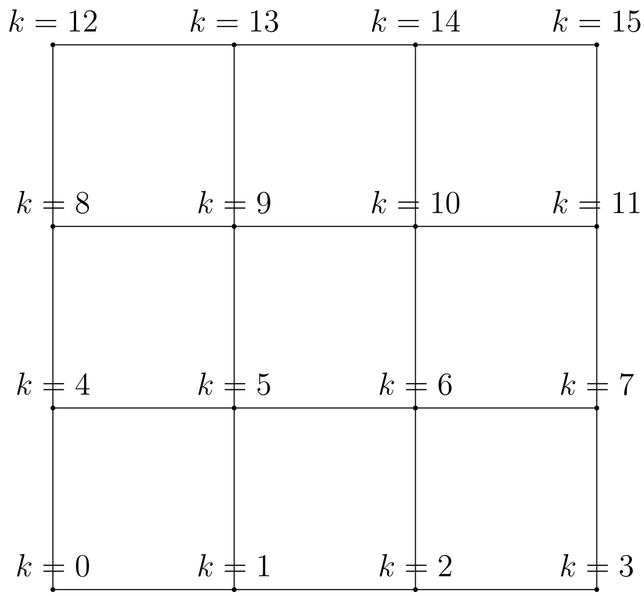
$$\mathbf{A} = \begin{bmatrix} \times & & & & & & & & & \\ & \times & & & & & & & & \\ \times & \times & \times & \times & \times & & & & & \\ & \times & \times & \times & \times & \times & & & & \\ & & \times & \times & \times & \times & \times & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots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

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

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

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

Here is an illustration for a 4×4 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

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

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

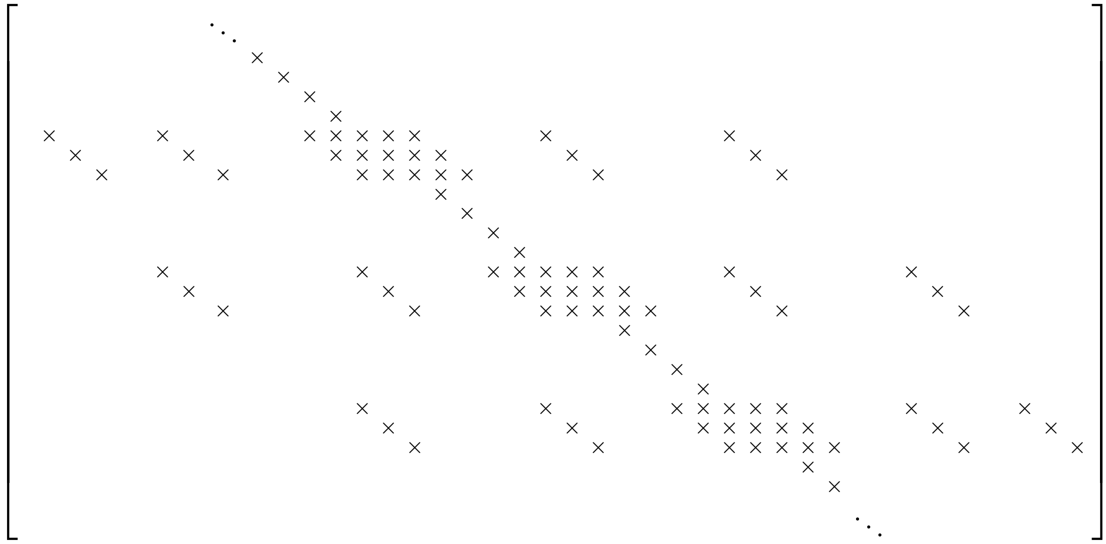
We now define $(N+1)^2 \times (N+1)^2$ matrices $\mathbf{A_x}$ (interior x-direction derivatives), $\mathbf{A_y}$ (interior y-direction derivatives) and $\mathbf{A_b}$ (boundary and close to boundary elements) such that,

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

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

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

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 7×7 grid, which means the interior matrix is 3×3)

$\mathbf{A} =$ 

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

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

$$\mathbf{A} = \left[\begin{array}{ccccccccccccccc}
 & & & & 1 & & & & & & & & & & \\
 & & & & & 1 & & & & & & & & & \\
1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots & -16 & \dots & 1 & & \\
 & 1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots & -16 & \dots & 1 & \\
 & & 1 & \dots & -16 & \dots & 1 & -16 & 60 & -16 & 1 & \dots & -16 & \dots & 1 \\
 & & & & & & & & & 1 & & & & & \\
 & & & & & & & & & & 1 & & & &
\end{array}\right]$$

Now (5) 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)-Σ_{j≠i}^{K} A(i,j)T_old(j))
        end do

        error = ||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)-Σ_{j≠i}^{K} A(i,j)T(j))
        end do

        error = ||T_old−T|| / ||T||
        iters = iters + 1
        if (error <= TOL)
            break
        end if



    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 & T_old	2(N+1)	$16\times$(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
Δx	1	8
k	1	8
	Total	8(N+1)^2+24(N+1)+28

    * 2D

Variable	Dimension	Memory
T & T_old	2(N+1)(N+1)	$16\times$(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
Δx	1	8
Δy	1	8
k	1	8
	Total	8(N+1)^4+24(N+1)^2+36

  - Fourth order approximation
    * 1D

Variable	Dimension	Memory
T & T_old	2(N+1)	$16\times$(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
Δx	1	8
k	1	8
	Total	8(N+1)^2+24(N+1)+28

* 2D

Variable	Dimension	Memory
T & T_old	2(N+1)(N+1)	$16\times$(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
Δx	1	8
Δy	1	8
k	1	8
	Total	8(N+1)^4+24(N+1)^2+36

- **Build Procedures and description of files**

  - Various `C++` files

    `main.cpp` - The driver routine, also has defensive checks. It parses the input file, feeds it to various functions and depending on mode, prints out to `std::out`. It also stores the result to `output.log`.

    `q_assemble.cpp` - Assembles `q` vector based on specifications in the input file

    `T_exact_assemble.cpp` - Assembles the `MASA` solution of the temperature field based on specifications in the input file

    `matrix_assemble.cpp` - Assembles `A` matrix based on specifications in the input file

    `print.cpp` - Contains various functions to print stuff, this is just so that `main.cpp` remains relatively uncluttered.

    `global_variables.h` - Defined objects of various `GRVY` classes as `extern` variables. It is imported in all other cpp files so that the same object is used in all files.

    `solvers.cpp` - Contains the solvers and function to choose solvers based on input file specifications, petsc has also been added here `my_inputfile_parser.cpp` is defining a class to parse the input file. This was done to clean up the rather large `main.cpp` being used earlier

`my_solver_class.cpp` is defining a solver class which makes the flow of the main routine look like a pseudocode in essence. An object of this class can do it all - allocating, deallocating memory, assembling, solving, I/O, etc.

– Running the primary code with petsc enabled

You will find `Makefile.am` in all subdirectories and `configure.ac` in `proj02`.

These are the steps to run the primary code with petsc and without code coverage (assuming you are in the `proj02` subdirectory). All of the code is in the `proj02/src` subdir. The shell script should work but if doesn't - change the permissions using `chmod 777 build_autotools.sh`.

```
$ module load petsc
$ module load phdf5
$ rm src/*.gcno src/*.gcda ### Just to be safe against gcov files ↩
    causing issues
### Go to the ./build_autotools.sh file and uncomment the option 1 ↩
    configure line and comment out option 2.
$ ./build_autotools.sh ### Runs autoreconf and ./configure with ↩
    appropriate options to link to MASA and GRVY
$ make clean; make ### Creates an executable in proj02/src subdir ↩
    named heat_solve
$ cd src/
$ ./heat_solve
```

– Running the primary code with petsc disabled and make coverage

You will find `Makefile.am` in all subdirectories and `configure.ac` in `proj02`.

These are the steps to run the primary code without petsc but with code coverage (assuming you are in the `proj02` subdirectory). All of the code is in the `proj02/src` subdir. The shell script should work but if doesn't - change the permissions using `chmod 777 build_autotools.sh`.

```
$ module load hdf5
$ module unload petsc
$ export CLASSPATH=/work/00161/karl/stampede2/public
$ export MODULEPATH=$CLASSPATH/ohpc/pub/modulefiles/:$MODULEPATH
$ module swap intel gnu7 ## or module load gnu7
$ which gcc ## check if the output is - /work/00161/karl/stampede2/↩
    public/ohpc/pub/compiler/gcc/7.1.0/bin/gcc

### Go to the ./build_autotools.sh file and comment out the option 1 ↩
    configure line and uncomment the option 2 configure line.
$ ./build_autotools.sh ### Runs autoreconf and ./configure with ↩
    appropriate options to link to MASA and GRVY
$ make clean; make ### Creates an executable in proj02/src subdir ↩
    named heat_solve
$ cd src/
$ ./heat_solve
```

To run `make coverage`,

```
$ module load hdf5
$ module unload petsc
$ export  PATH=/work/00161/karl/stampede2/public/bats/bin/:$PATH ### ↩
    Add bats to path to run regression tests
$ export CLASSPATH=/work/00161/karl/stampede2/public
$ export MODULEPATH=$CLASSPATH/ohpc/pub/modulefiles/:$MODULEPATH
$ module swap intel gnu7 ## or module load gnu7
$ which gcc ## check if the output is - /work/00161/karl/stampede2/↩
    public/ohpc/pub/compiler/gcc/7.1.0/bin/gcc

### Go to the ./build_autotools.sh file and comment out the option 1 ↩
    configure line and uncomment the option 2 configure line.
$ ./build_autotools.sh ### Runs autoreconf and ./configure with ↩
    appropriate options to link to MASA and GRVY
$ make clean; make coverage### Creates an executable in proj02/src ↩
    subdir named heat_solve
$ tar cvfz cover.tar.gz coverage/lcov

### Do this on local machine
local_device$ scp shrey911@stampede2.tacc.utexas.edu:~/cse380-2020-↩
    student-Shreyas911/proj02/cover.tar.gz ~/Desktop
local_device$ tar xfz cover.tar.gz
```

You should have a new `coverage/proj02` directory after executing these commands. You can tar it and `scp` to local device.

- Running regression tests

  You can find these tests in the `proj02/tests` subdir in `test.sh`.

```
#### Do the prior steps for either of the two sections above and then↩
     run this shell script
$ ./build_autotools.sh ### Runs autoreconf and ./configure with ↩
    appropriate options to link to MASA and GRVY
$ export  PATH=/work/00161/karl/stampede2/public/bats/bin/:$PATH ### ↩
    Add bats to path to run regression tests
$ make check
```

- Running the various post-processing scripts

  The shell script should work but if they don't - change the permissions using `chmod 777 shell_script_name`.

  `mesh_size_change.sh` is a script to run all configurations you want in a for loop. All you have to do is change the array variables inside. The plotting is now done by `plot_convergence.py`.

  `curve_1D_plotter.script` and `surface_2D_plotter.script` create 1D curves (`plot.png`) and 2D surfaces (`surface.png`) for the temperature fields that get stored in `output.log` or `output_100x100.log`. These are gnuplot scripts.

  `plot_convergence.script` (again gnuplot) was used in `proj01` for getting norm vs n and time vs n graphs. But we have a python script to do this now.

h5py_surface_plotter.py is used to create beautiful surface plots and is a new addtion.

plot_convergence.py is also a new addition. It does all of the error vs n and time vs n plotting.

– Tex files and report The tex files and figures can be found in the tex_report subdir. One can use pdflatex proj02.tex to build the report proj02.pdf.

– Various output files

reference_sol_*.log contain reference solutions created using mesh_size_change.sh for regression testing using TOL = 1e-17 and MAX_ITERS= 1000000.

reference_sol_*.h5 contain reference solutions created using mesh_size_change.sh for regression testing using TOL = 1e-17 and MAX_ITERS= 1000000. They are hdf5 files. Sample hdf5 files can be found further down in the report.

convergence_*.png contains images of convergence analysis, created by mesh_size_change.sh. This was a proj01 feature.

output.log contains the position vs Temperature data, created by main.cpp. It gets updated every time the executable is run. This was a proj01 feature.

output_100x100.log and output_100x100.h5 is a snapshot solution of a 100x100 grid with gauss-seidel 4th order to plot surface plots. This was a proj01 feature. We now use a .h5 file instead with the hdf5 format.

l2norm_petsc_gauss_order*.png contains comparisons in error norms vs n for petsc and gauss solvers.

data.h5 is the output of main.cpp. It is in the hdf5 format.

hdf5*.png contains surface and contourf plots created using output_100x100.h5. The plots are illustrated below.

output_*.dat is created by mesh_size_change.sh and contains data on n, L2 error and time taken.

time_all_solvers_order*.png is the time comparison for all solvers.

time_gauss*.png and time_jacobi*.png are the time plots for the individual solvers. They come from proj01.

plot.png has the 1D temperature vs x plot

surface_T_computed.png and surface_T_exact.png have the surface plot for temperature over a 2D domain.

– Other files

.travis.yml is a file which is responsible for CI with Travis. You will find it at the top level of the repo. You will notice that the configure commands inside the container are much easier to use than on Stampede2. (you can see the sequence of commands in this file).

`README.md` This file is linked to the travis badge and it shows us if the CI tests ran properly for the latest version or not.

`proj02/cover.tar.gz` has the coverage results also shown below.

`/proj02/lcov/` directory is just for the code coverage section. You do not need to worry too much about this one.

- **Input Options**

  Here is an example `input.dat` file.

```
verification = 1              #Comparison with MASA solution? 1 for yes; no otherwise; Always keep as 1. Otherwise ↩
    a couple of regression tests might fail.
mode = debug                  #To enable debug mode, use 'debug', anything else is normal mode.

[grid]

length        = 1.0           # Length of domain in each direction
dimension = 1                 # dimension of domain
grid_points = 200             # Number of points in one direction


[solver]

thermal_conductivity    = 1.0              # Thermal conductivity k_0
solver_name = jacobi                       # Use either jacobi or gauss or petsc (first ensure you build with petsc)
order = 2                                  # Order of accuracy of stencil, use 2 or 4
error_TOL       = 0.00000000000000001      # Tolerance
max_iters       = 1000000                  # Maximum number of iterations
```

  - Modes

    The debug mode can be activated by using `mode = debug`. Anything else is assumed to mean not in debug mode. It gives out a verbose output, an example can be found below.

    There is also a verification mode which is recommended to always be turned on, since post-processing scripts grep for certain strings to aggregate the data for convergence plots.

  - Grid options

    The options are domain length `length`, dimension `dimension` (which can be 1D or 2D) and number of grid points in one direction `grid_points`.

  - Solver options

    The solver options are essentially specifications of physical parameters such as `thermal_conductivity` and other specifications such as error tolerance `error_TOL`, order of accuracy desired `order`, name of the solver `solver_name` and maximum iterations allowed `max_iters`. DO NOT change the `error_TOL = 1e-17` and `max_iters = 1000000` if you want to compare to reference solutions.

- **Verification procedures** By default, the verification mode is always on. You can tweak that in `input.dat` if you don't want to keep it on.

It is recommended to always be turned on, since post-processing scripts grep for certain strings to aggregate the data for convergence plots.

Here is a sample output for verification mode on , debug mode off, 1D gauss 2nd order solution -

```
--> verification_mode = 1
--> mode          = no_debug
--> n             = 10
--> dimension     = 1
--> length        = 1.000000
--> order         = 2
--> error_TOL     = 1.0000000000000001e-17
--> thermal_conductivity = 1.000000
--> max_iters     = 1000000
--> solver        = gauss

VERIFICATION MODE -
L2 norm of error for n 10    is    0.048363774712789243

------------------------------------------------------------------------------------------------------------------------
GRVY Performance timing - Performance Timings:                         |      Mean        Variance         Count
--> q_order2_dim1                      : 1.72210e-03 secs ( 30.1486 %) | [1.72210e-03  0.00000e+00             1]
--> main                               : 1.38593e-03 secs ( 24.2633 %) | [1.38593e-03  0.00000e+00             1]
--> write_results_output_file          : 1.26815e-03 secs ( 22.2014 %) | [1.26815e-03  0.00000e+00             1]
--> T_exact_order2_dim1                : 1.14799e-03 secs ( 20.0977 %) | [1.14799e-03  0.00000e+00             1]
--> l2_norm                            : 8.03471e-05 secs (  1.4066 %) | [2.94312e-07  2.31745e-13           273]
--> gauss                              : 5.81741e-05 secs (  1.0184 %) | [5.81741e-05  0.00000e+00             1]
--> print_verification_mode            : 2.09808e-05 secs (  0.3673 %) | [2.09808e-05  0.00000e+00             1]
--> assemble_A                         : 5.96046e-06 secs (  0.1043 %) | [5.96046e-06  0.00000e+00             1]
--> print_matrix_A                     : 1.90735e-06 secs (  0.0334 %) | [1.90735e-06  0.00000e+00             1]
--> solve                              : 9.53674e-07 secs (  0.0167 %) | [9.53674e-07  0.00000e+00             1]
--> assemble_q                         : 0.00000e+00 secs (  0.0000 %) | [0.00000e+00  0.00000e+00             1]
--> GRVY_Unassigned                    : 1.57356e-05 secs (  0.2755 %)

                 Total Measured Time = 5.71203e-03 secs ( 99.9332 %)
------------------------------------------------------------------------------------------------------------------------
```

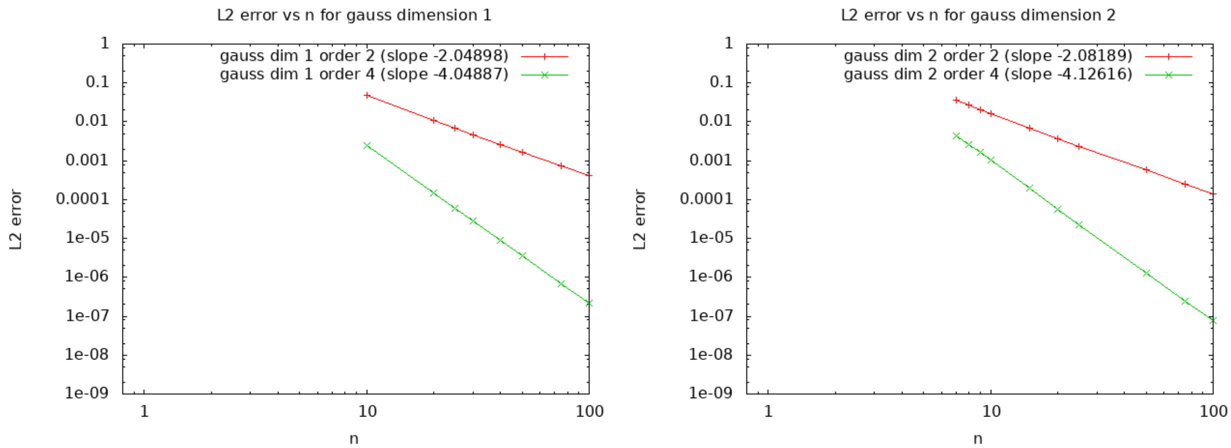
- **Debug mode output** With debug mode on, the output looks as -

```
--> verification_mode = 1
--> mode          = debug
Registering user-supplied default value for grid/grid_points
--> n             = 10
Registering user-supplied default value for grid/dimension
--> dimension     = 1
Registering user-supplied default value for grid/length
--> length        = 1.000000
Registering user-supplied default value for solver/order
--> order         = 2
Registering user-supplied default value for solver/error_TOL
--> error_TOL     = 1.0000000000000001e-17
Registering user-supplied default value for solver/thermal_conductivity
--> thermal_conductivity = 1.000000
Registering user-supplied default value for solver/max_iters
--> max_iters     = 1000000
--> solver        = gauss

VERIFICATION MODE -
L2 norm of error for n 10    is    0.048363774712789243

DEBUG MODE - printing A in MATLAB compatible form
A = [1  0  0  0  0  0  0  0  0  0  ;
-1  2  -1  0  0  0  0  0  0  0  ;
0  -1  2  -1  0  0  0  0  0  0  ;
0  0  -1  2  -1  0  0  0  0  0  ;
0  0  0  -1  2  -1  0  0  0  0  ;
0  0  0  0  -1  2  -1  0  0  0  ;
0  0  0  0  0  -1  2  -1  0  0  ;
0  0  0  0  0  0  -1  2  -1  0  ;
0  0  0  0  0  0  0  -1  2  -1  ;
0  0  0  0  0  0  0  0  0  1  ;
]

DEBUG MODE - printing q in MATLAB compatible form
q = [1
```

```
0.37336077072775858
0.084634015730502873
-0.24369393582936677
-0.45799478645826142
-0.45799478645826142
-0.24369393582936708
0.084634015730502651
0.37336077072775847
1
]

DEBUG MODE - Compare vector values
q              T_exact          T_computed
1.000000       1.000000         1.000000
0.373361       0.766044         0.756306
0.084634       0.173648         0.139251
-0.243694      -0.500000        -0.562437
-0.457995      -0.939693        -1.020432
-0.457995      -0.939693        -1.020432
-0.243694      -0.500000        -0.562437
0.084634       0.173648         0.139251
0.373361       0.766044         0.756306
1.000000       1.000000         1.000000

-----------------------------------------------------------------------------------------------------------------
GRVY Performance timing - Performance Timings:                        |    Mean          Variance        Count
--> q_order2_dim1                   : 1.86706e-03 secs ( 27.8743 %) | [1.86706e-03  0.00000e+00          1]
--> main                            : 1.71304e-03 secs ( 25.5749 %) | [1.71304e-03  0.00000e+00          1]
--> write_results_output_file       : 1.44100e-03 secs ( 21.5135 %) | [1.44100e-03  0.00000e+00          1]
--> T_exact_order2_dim1             : 1.21903e-03 secs ( 18.1996 %) | [1.21903e-03  0.00000e+00          1]
--> print_matrix_A                  : 1.03951e-04 secs (  1.5519 %) | [1.03951e-04  0.00000e+00          1]
--> l2_norm                         : 9.27448e-05 secs (  1.3846 %) | [3.39725e-07  2.79576e-13        273]
--> print_compare_q_Texact_Tcomputed : 7.39098e-05 secs (  1.1034 %) | [7.39098e-05  0.00000e+00          1]
--> print_vector_q                  : 7.20024e-05 secs (  1.0750 %) | [7.20024e-05  0.00000e+00          1]
--> gauss                           : 5.57899e-05 secs (  0.8329 %) | [5.57899e-05  0.00000e+00          1]
--> print_verification_mode         : 3.19481e-05 secs (  0.4770 %) | [3.19481e-05  0.00000e+00          1]
--> assemble_A                      : 8.10623e-06 secs (  0.1210 %) | [8.10623e-06  0.00000e+00          1]
--> solve                           : 1.90735e-06 secs (  0.0285 %) | [1.90735e-06  0.00000e+00          1]
--> matrix_order2_dim1              : 9.53674e-07 secs (  0.0142 %) | [9.53674e-07  0.00000e+00          1]
--> assemble_q                      : 0.00000e+00 secs (  0.0000 %) | [0.00000e+00  0.00000e+00          1]
--> GRVY_Unassigned                 : 1.57356e-05 secs (  0.2349 %)

                  Total Measured Time = 6.69813e-03 secs ( 99.9858 %)
-----------------------------------------------------------------------------------------------------------------
```

- **Verification Exercise**
  - Gauss solver

    These are the grid convergence plots for the gauss-seidel solver. The expected slopes were -2 and -4 for 2nd and 4th order respectively. We have slopes pretty close to these expected values.

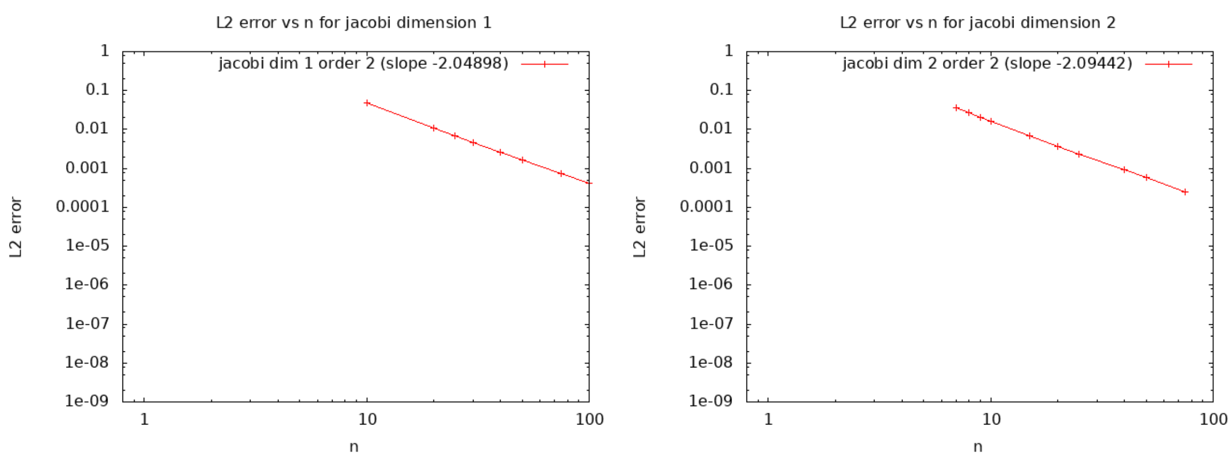


(a) 1D (b) 2D

Figure 1: Convergence analysis of gauss-seidel solver

- <u>Jacobi solver</u>

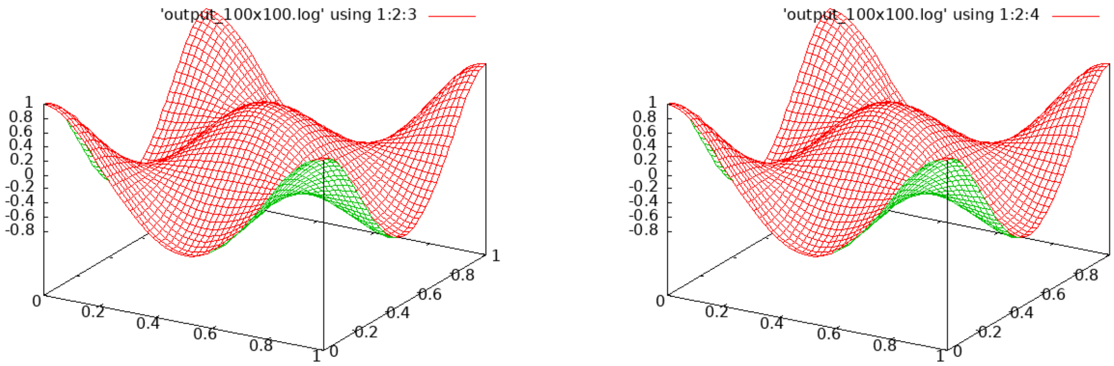
  These are the grid convergence plots for the jacobi solver. We don't consider jacobi with 4th order solver since it is not stable. The expected slope was -2 for 2nd order. We have slopes pretty close to this expected value.



(a) 1D (b) 2D

Figure 2: Convergence analysis of jacobi solver

Just for completeness, here we have two surface plots for a 4th order gauss 2D simulation on a `100x100` grid. Both plots look essentially similar.



(a) T_exact

(b) T_computed Gauss 4th order 100x100

Figure 3: Surface plots

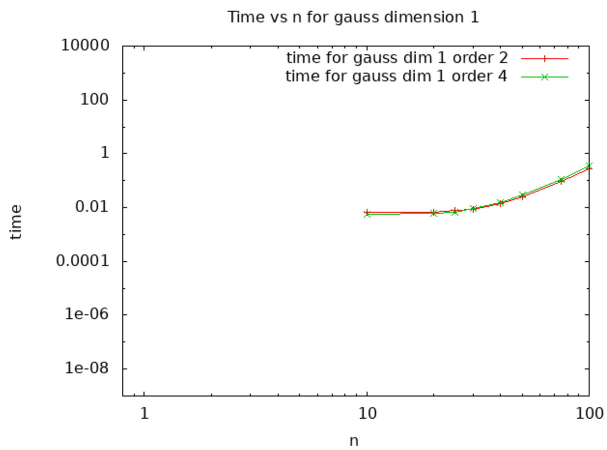
- **Runtime performance**

  A sample of time output for the given input configuration is shown below.

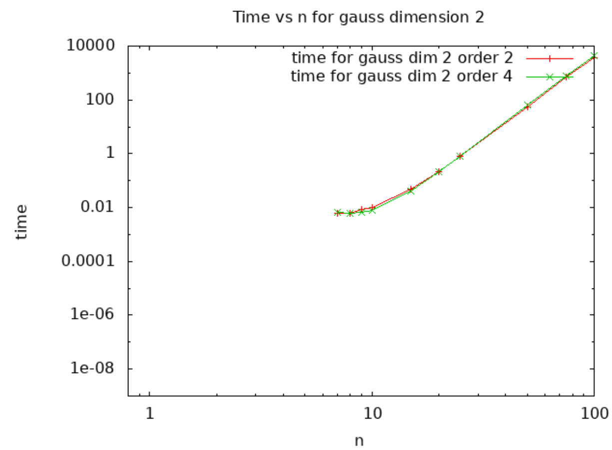
```
--> verification_mode = 1
--> mode          = no_debug
--> n             = 20
--> dimension     = 2
--> length        = 1.000000
--> order         = 4
--> error_TOL     = 1.0000000000000001e-17
--> thermal_conductivity = 1.000000
--> max_iters     = 1000000
--> solver        = gauss

VERIFICATION MODE -
L2 norm of error for n 20    is    5.7410182758840052e-05

---------------------------------------------------------------------------------------------------------
GRVY Performance timing - Performance Timings:                  |      Mean        Variance        Count
--> gauss                             : 2.02494e-01 secs ( 94.6842 %) | [2.02494e-01  0.00000e+00          1]
--> write_results_output_file         : 4.57191e-03 secs (  2.1378 %) | [4.57191e-03  0.00000e+00          1]
--> q_order4_dim2                     : 1.84488e-03 secs (  0.8626 %) | [1.84488e-03  0.00000e+00          1]
--> main                              : 1.34730e-03 secs (  0.6300 %) | [1.34730e-03  0.00000e+00          1]
--> T_exact_order4_dim2               : 1.22809e-03 secs (  0.5742 %) | [1.22809e-03  0.00000e+00          1]
--> l2_norm                           : 9.53197e-04 secs (  0.4457 %) | [7.93670e-07  2.06032e-13       1201]
--> matrix_order4_dim2                : 7.10964e-04 secs (  0.3324 %) | [7.10964e-04  0.00000e+00          1]
--> print_matrix_A                    : 6.26087e-04 secs (  0.2928 %) | [6.26087e-04  0.00000e+00          1]
--> print_verification_mode           : 2.81334e-05 secs (  0.0132 %) | [2.81334e-05  0.00000e+00          1]
--> assemble_A                        : 5.00679e-06 secs (  0.0023 %) | [5.00679e-06  0.00000e+00          1]
--> print_vector_q                    : 3.09944e-06 secs (  0.0014 %) | [3.09944e-06  0.00000e+00          1]
--> print_compare_q_Texact_Tcomputed : 2.86102e-06 secs (  0.0013 %) | [2.86102e-06  0.00000e+00          1]
--> assemble_T_exact                  : 2.14577e-06 secs (  0.0010 %) | [2.14577e-06  0.00000e+00          1]
--> solve                             : 1.90735e-06 secs (  0.0009 %) | [1.90735e-06  0.00000e+00          1]
--> assemble_q                        : 9.53674e-07 secs (  0.0004 %) | [9.53674e-07  0.00000e+00          1]
--> GRVY_Unassigned                   : 4.19617e-05 secs (  0.0196 %)

                  Total Measured Time = 2.13863e-01 secs (100.0000 %)
---------------------------------------------------------------------------------------------------------
```

  The total time for various cases for various solver configurations is plotted and can be found below.

  - Gauss solver



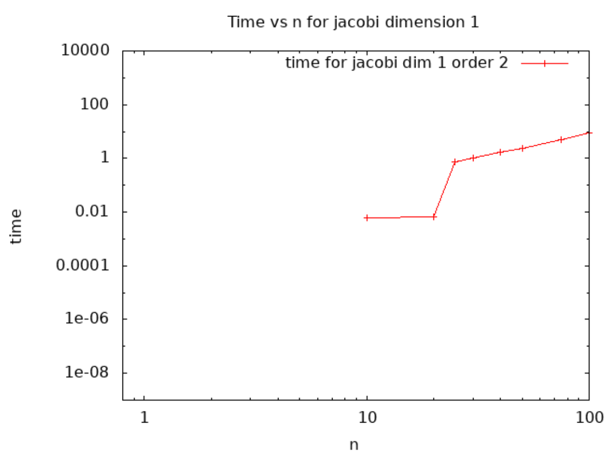
(a) 1D



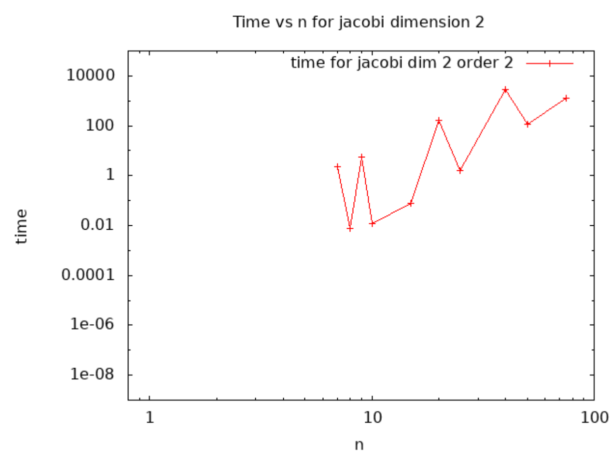
(b) 2D

Figure 4: Runtime analysis of gauss solver

– Jacobi



(a) 1D



(b) 2D

Figure 5: Runtime analysis of jacobi solver

- **Regression testing**

  `make check` runs the regression tests, which are stored in `proj02/tests/test.sh`. The sample output of `make check` is given below. (the colour is red due to LaTeXbut the test has actually passed). It shows only 1 test but technically the `test.sh` script contains 5 tests with the 4th test actually containing 4 tests and the 5th containing 2 tests. This makes a total of 9 tests.

```
Making check in src
make[1]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/src'
make[1]: Nothing to be done for `check'.
make[1]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/src'
```

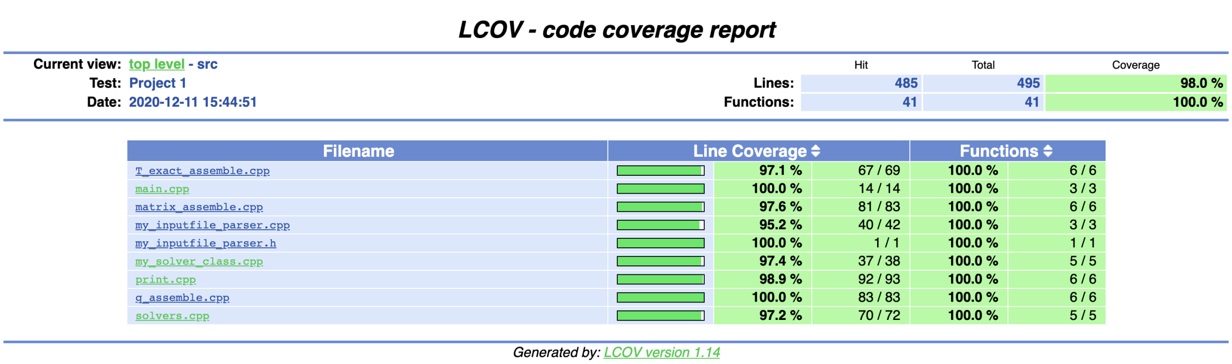
```
Making check in tests
make[1]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make  check-TESTS
make[2]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make[3]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
PASS: test.sh
make[4]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make[4]: Nothing to be done for `all'.
make[4]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
============================================================================
Testsuite summary for FULL-PACKAGE-NAME VERSION
============================================================================
# TOTAL: 1
# PASS:  1
# SKIP:  0
# XFAIL: 0
# FAIL:  0
# XPASS: 0
# ERROR: 0
============================================================================
make[3]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make[2]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make[1]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02/tests'
make[1]: Entering directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02'
make[1]: Leaving directory `/home1/07665/shrey911/temp/cse380-2020-student-Shreyas911/proj02'
```

Alternatively, you can run `proj02/tests/test.sh` directly, which gives the following output to `std::out`-

```
$ ./test.sh
✓ verify that the code is compiling
✓ verify that the verification mode runs fine
✓ verify that the debug mode runs fine
✓ verify all gauss solver outputs match reference outputs
✓ verify all jacobi solver outputs match reference outputs
```

- **Code coverage**

  Running make coverage as described above in the 'Running the primary code with petsc disabled and make coverage' section, we get a `proj02/coverage` directory. Tar it up, scp it to your local device, untar it and go inside to view `coverage/lcov/index.html`. You can find the `proj02/cover.tar.gz` tarball in the repo. You will see an output that looks as follows -



LCOV - code coverage report

Current view: top level - src
Test: Project 1
Date: 2020-12-11 15:44:51

	Hit	Total	Coverage
Lines:	485	495	98.0 %
Functions:	41	41	100.0 %

Filename	Line Coverage		Functions	
T_exact_assemble.cpp	97.1 %	67 / 69	100.0 %	6 / 6
main.cpp	100.0 %	14 / 14	100.0 %	3 / 3
matrix_assemble.cpp	97.6 %	81 / 83	100.0 %	6 / 6
my_inputfile_parser.cpp	95.2 %	40 / 42	100.0 %	3 / 3
my_inputfile_parser.h	100.0 %	1 / 1	100.0 %	1 / 1
my_solver_class.cpp	97.4 %	37 / 38	100.0 %	5 / 5
print.cpp	98.9 %	92 / 93	100.0 %	6 / 6
q_assemble.cpp	100.0 %	83 / 83	100.0 %	6 / 6
solvers.cpp	97.2 %	70 / 72	100.0 %	5 / 5

Generated by: LCOV version 1.14

Figure 6: lcov output

- **HDF5 I/O** Shown below is a sample `.h5` file. We use `h5dump` for this purpose. The solver is jacobi and the rest of the metadata can be seen inside the `.h5` file itself.

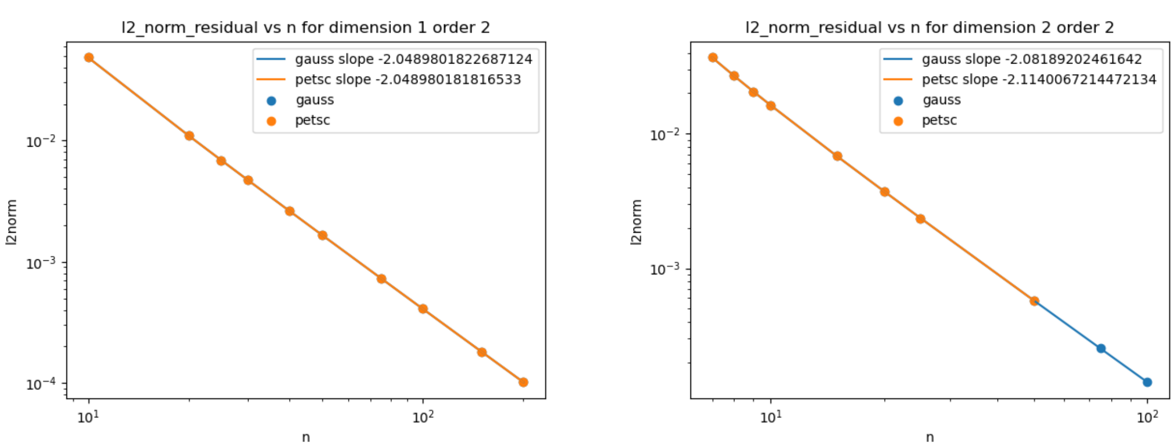
```
$ h5dump data.h5
HDF5 "data.h5" {
GROUP "/" {
   GROUP "coordinates" {
      DATASET "x data" {
         DATATYPE  H5T_IEEE_F64LE
         DATASPACE  SIMPLE { ( 100 ) / ( 100 ) }
         DATA {
         (0): 0, 0.111111, 0.222222, 0.333333, 0.444444, 0.555556, ↩
            0.666667,
         (7): 0.777778, 0.888889, 1, 0, 0.111111, 0.222222, 0.333333,
         (14): 0.444444, 0.555556, 0.666667, 0.777778, 0.888889, 1, 0,
         (21): 0.111111, 0.222222, 0.333333, 0.444444, 0.555556, 0.666667,
         (27): 0.777778, 0.888889, 1, 0, 0.111111, 0.222222, 0.333333,
         (34): 0.444444, 0.555556, 0.666667, 0.777778, 0.888889, 1, 0,
         (41): 0.111111, 0.222222, 0.333333, 0.444444, 0.555556, 0.666667,
         (47): 0.777778, 0.888889, 1, 0, 0.111111, 0.222222, 0.333333,
         (54): 0.444444, 0.555556, 0.666667, 0.777778, 0.888889, 1, 0,
         (61): 0.111111, 0.222222, 0.333333, 0.444444, 0.555556, 0.666667,
         (67): 0.777778, 0.888889, 1, 0, 0.111111, 0.222222, 0.333333,
         (74): 0.444444, 0.555556, 0.666667, 0.777778, 0.888889, 1, 0,
         (81): 0.111111, 0.222222, 0.333333, 0.444444, 0.555556, 0.666667,
         (87): 0.777778, 0.888889, 1, 0, 0.111111, 0.222222, 0.333333,
         (94): 0.444444, 0.555556, 0.666667, 0.777778, 0.888889, 1
         }
      }
      DATASET "y data" {
         DATATYPE  H5T_IEEE_F64LE
         DATASPACE  SIMPLE { ( 100 ) / ( 100 ) }
         DATA {
         (0): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.111111, 0.111111, 0.111111,
         (13): 0.111111, 0.111111, 0.111111, 0.111111, 0.111111, 0.111111,
         (19): 0.111111, 0.222222, 0.222222, 0.222222, 0.222222, 0.222222,
         (25): 0.222222, 0.222222, 0.222222, 0.222222, 0.222222, 0.333333,
         (31): 0.333333, 0.333333, 0.333333, 0.333333, 0.333333, 0.333333,
         (37): 0.333333, 0.333333, 0.333333, 0.444444, 0.444444, 0.444444,
         (43): 0.444444, 0.444444, 0.444444, 0.444444, 0.444444, 0.444444,
         (49): 0.444444, 0.555556, 0.555556, 0.555556, 0.555556, 0.555556,
         (55): 0.555556, 0.555556, 0.555556, 0.555556, 0.555556, 0.666667,
         (61): 0.666667, 0.666667, 0.666667, 0.666667, 0.666667, 0.666667,
         (67): 0.666667, 0.666667, 0.666667, 0.777778, 0.777778, 0.777778,
         (73): 0.777778, 0.777778, 0.777778, 0.777778, 0.777778, 0.777778,
         (79): 0.777778, 0.888889, 0.888889, 0.888889, 0.888889, 0.888889,
         (85): 0.888889, 0.888889, 0.888889, 0.888889, 0.888889, 1, 1, 1, ↩
            1,
         (94): 1, 1, 1, 1, 1, 1
         }
      }
   }
   GROUP "metadata" {
      DATASET "dimensions" {
         DATATYPE  H5T_STD_I32LE
         DATASPACE  SIMPLE { ( 1 ) / ( 1 ) }
```

```
         DATA {
         (0): 2
         }
      }
      DATASET "grid_points" {
         DATATYPE  H5T_STD_I32LE
         DATASPACE  SIMPLE { ( 1 ) / ( 1 ) }
         DATA {
         (0): 10
         }
      }
      DATASET "order" {
         DATATYPE  H5T_STD_I32LE
         DATASPACE  SIMPLE { ( 1 ) / ( 1 ) }
         DATA {
         (0): 2
         }
      }
   }
   GROUP "temperature" {
      DATASET "Analytical Temperature" {
         DATATYPE  H5T_IEEE_F64LE
         DATASPACE  SIMPLE { ( 100 ) / ( 100 ) }
         DATA {
         (0): 1, 0.766044, 0.173648, -0.5, -0.939693, -0.939693, -0.5,
         (7): 0.173648, 0.766044, 1, 0.766044, 0.586824, 0.133022, ↩
            -0.383022,
         (14): -0.719846, -0.719846, -0.383022, 0.133022, 0.586824, ↩
            0.766044,
         (20): 0.173648, 0.133022, 0.0301537, -0.0868241, -0.163176,
         (25): -0.163176, -0.0868241, 0.0301537, 0.133022, 0.173648, -0.5,
         (31): -0.383022, -0.0868241, 0.25, 0.469846, 0.469846, 0.25,
         (37): -0.0868241, -0.383022, -0.5, -0.939693, -0.719846, ↩
            -0.163176,
         (43): 0.469846, 0.883022, 0.883022, 0.469846, -0.163176, ↩
            -0.719846,
         (49): -0.939693, -0.939693, -0.719846, -0.163176, 0.469846,
         (54): 0.883022, 0.883022, 0.469846, -0.163176, -0.719846, ↩
            -0.939693,
         (60): -0.5, -0.383022, -0.0868241, 0.25, 0.469846, 0.469846, ↩
            0.25,
         (67): -0.0868241, -0.383022, -0.5, 0.173648, 0.133022, 0.0301537,
         (73): -0.0868241, -0.163176, -0.163176, -0.0868241, 0.0301537,
         (78): 0.133022, 0.173648, 0.766044, 0.586824, 0.133022, ↩
            -0.383022,
         (84): -0.719846, -0.719846, -0.383022, 0.133022, 0.586824, ↩
            0.766044,
         (90): 1, 0.766044, 0.173648, -0.5, -0.939693, -0.939693, -0.5,
         (97): 0.173648, 0.766044, 1
         }
      }
      DATASET "Numerical Temperature" {
         DATATYPE  H5T_IEEE_F64LE
         DATASPACE  SIMPLE { ( 100 ) / ( 100 ) }
         DATA {
         (0): 1, 0.766044, 0.173648, -0.5, -0.939693, -0.939693, -0.5,
```

```
        (7): 0.173648, 0.766044, 1, 0.766044, 0.594919, 0.137784, -0.384731,
        (14): -0.726125, -0.726125, -0.384731, 0.137784, 0.594919, 0.766044,
        (20): 0.173648, 0.137784, 0.0376302, -0.0772197, -0.152264,
        (25): -0.152264, -0.0772197, 0.0376302, 0.137784, 0.173648, -0.5,
        (31): -0.384731, -0.0772197, 0.27512, 0.505612, 0.505612, 0.27512,
        (37): -0.0772197, -0.384731, -0.5, -0.939693, -0.726125, -0.152264,
        (43): 0.505612, 0.935987, 0.935987, 0.505612, -0.152264, -0.726125,
        (49): -0.939693, -0.939693, -0.726125, -0.152264, 0.505612,
        (54): 0.935987, 0.935987, 0.505612, -0.152264, -0.726125, -0.939693,
        (60): -0.5, -0.384731, -0.0772197, 0.27512, 0.505612, 0.505612,
        (66): 0.27512, -0.0772197, -0.384731, -0.5, 0.173648, 0.137784,
        (72): 0.0376302, -0.0772197, -0.152264, -0.152264, -0.0772197,
        (77): 0.0376302, 0.137784, 0.173648, 0.766044, 0.594919, 0.137784,
        (83): -0.384731, -0.726125, -0.726125, -0.384731, 0.137784,
        (88): 0.594919, 0.766044, 1, 0.766044, 0.173648, -0.5, -0.939693,
        (95): -0.939693, -0.5, 0.173648, 0.766044, 1
        }
      }
    }
}
}
```

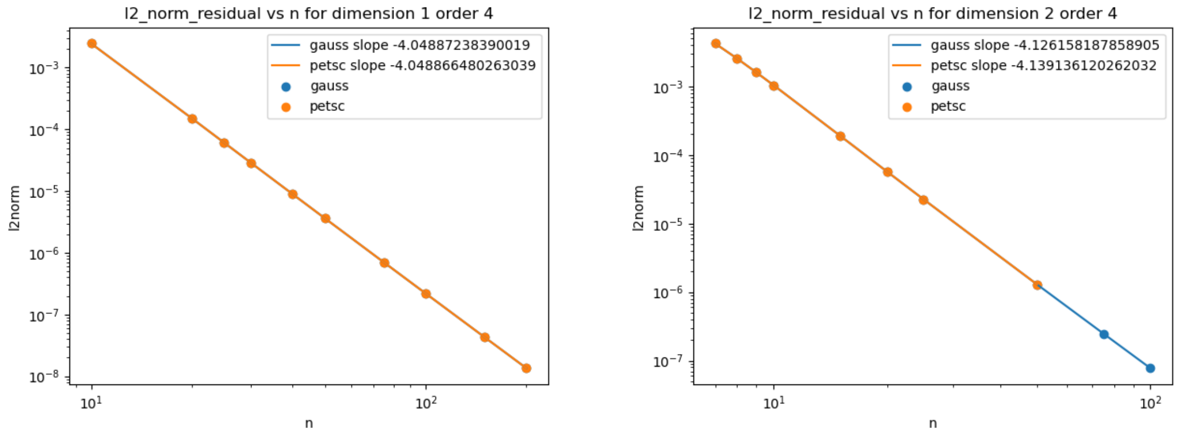
- **Petsc solver**

  Grid convergence as compared to gauss-seidel is shown in the plots below. We see that the error norms are almost identical to the gauss-seidel ones, thus completing our verification for petsc. For 2D, petsc was run up to n=50 due to lack of time.



(a) 1D (b) 2D

Figure 7: Convergence analysis of petsc solver for order 2

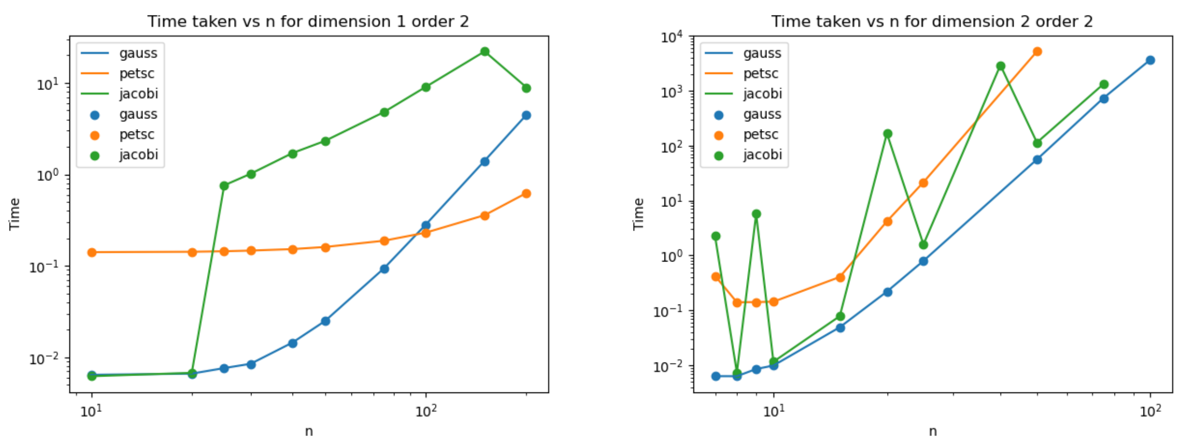


(a) 1D

(b) 2D

Figure 8: Convergence analysis of petsc solver for order 4

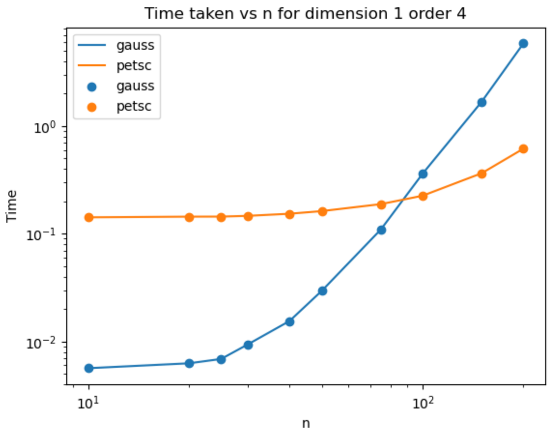
Next we have the timing comparison plots of all three solvers. petsc does well for 1D compared to other solvers, but not so well for 2D (meaning petsc does badly for super large number of points). This might be because the whole pipeline is not set up for petsc, for example assembling was done without inbuilt functions from petsc.



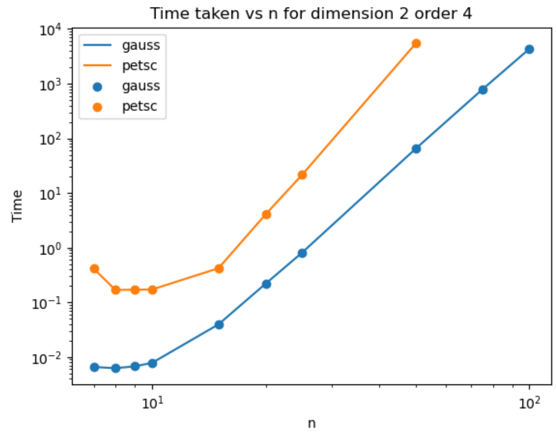
(a) 1D

(b) 2D

Figure 9: Timing comparison of all solvers for order 2



(a) 1D

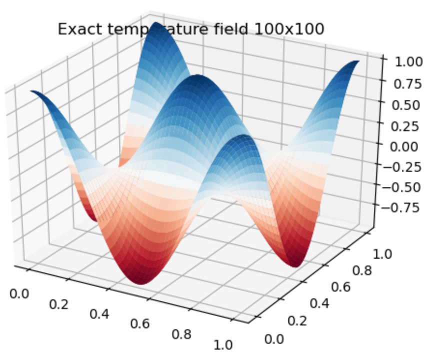


(b) 2D

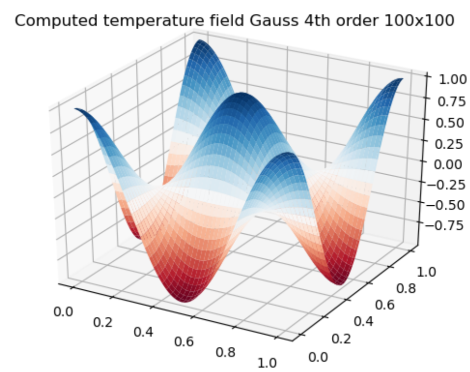
Figure 10: Timing comparison of gauss and petsc solvers for order 4

- **Beautiful plots using hdf5 files**

  We use `h5py` for this purpose. The plotting script is `h5py_surface_plotter.py`.

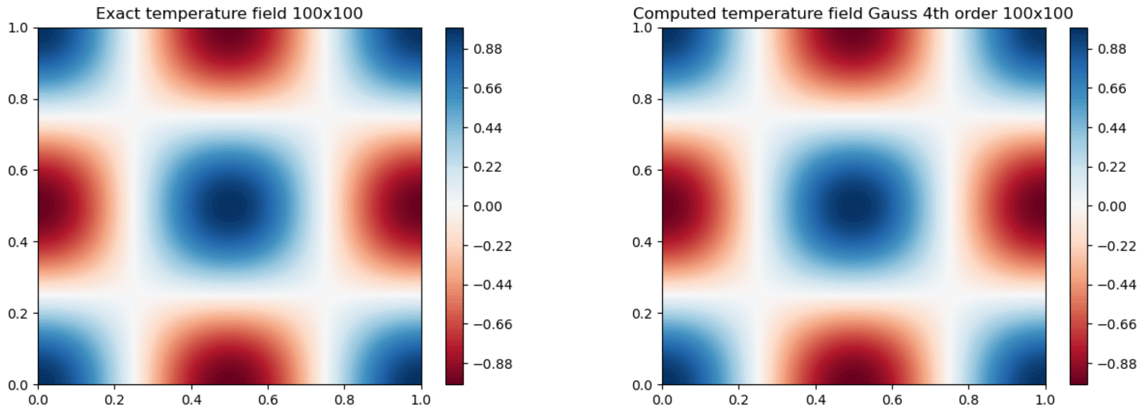


(a) T_exact



(b) T_computed Gauss 4th order 100x100

Figure 11: Surface plots



(a) T_exact (b) T_computed Gauss 4th order 100x100

Figure 12: Contour filled plots

- **Continuous integration**

  The dockerfile used for the container can be found in `proj02/src/`. The only difference between Karl's file and mine is that I have also installed vim. Here is the dockerfile for convenience.

```
FROM centos:7
MAINTAINER Karl W. Schulz <karl@oden.utexas.edu>

# Define a user
RUN useradd -u 2000 -m test

# enable OpenHPC repository
RUN yum install -y https://github.com/openhpc/ohpc/releases/download/v1.3.↩
    GA/ohpc-release-1.3-1.el7.x86_64.rpm

# Add some packages
RUN yum -y install make which git
RUN yum -y install vim
RUN yum -y install ohpc-autotools
RUN yum -y install valgrind-ohpc
RUN yum -y install gnu8-compilers-ohpc
RUN yum -y install gsl-gnu8-ohpc hdf5-gnu8-ohpc
RUN yum -y install openmpi3-gnu8-ohpc boost-gnu8-openmpi3-ohpc petsc-gnu8-↩
    openmpi3-ohpc
RUN yum -y install lmod-defaults-gnu8-openmpi3-ohpc
RUN yum -y install bc wget zlib-devel perl-Digest-MD5
# Add Bats
RUN yum -y install bats
# Add MASA
ENV masa_ver=0.50.0
RUN wget https://github.com/manufactured-solutions/MASA/releases/download/↩
    $masa_ver/masa-$masa_ver.tar.gz -P /tmp
RUN cd /tmp; tar xfz /tmp/masa-$masa_ver.tar.gz
RUN . /etc/profile.d/lmod.sh \
```

```
    && cd /tmp/masa-$masa_ver \
    && ./configure --enable-fortran-interfaces \
    && make \
    && make install
RUN rm /tmp/masa-$masa_ver.tar.gz
# Add GRVY
ENV grvy_ver="0.34.0"
RUN wget https://github.com/hpcsi/grvy/releases/download/$grvy_ver/grvy-↩
    $grvy_ver.tar.gz -P /tmp
RUN cd /tmp; tar xfz /tmp/grvy-$grvy_ver.tar.gz
RUN . /etc/profile.d/lmod.sh \
    && module load boost \
    && cd /tmp/grvy-$grvy_ver \
    && ./configure CXXFLAGS="-std=c++11" LDFLAGS="-Wl,-rpath,$BOOST_LIB" \
    && make \
    && make install
RUN rm /tmp/grvy-$grvy_ver.tar.gz

# Register new libs installed into /usr/local/lib with linker
RUN echo "/usr/local/lib" > /etc/ld.so.conf.d/class.conf
RUN ldconfig

# User to run as
WORKDIR /home/test
USER test
```

Here is the `.travis.yml` file for reference. The configure command is much simpler inside the docker as you can see.

```
sudo: required
language: C

services:
        - docker

before_install:
        - docker pull shreyas911/my_docker_image:v1.0
        - export DOCKER_RUN="docker run -v ${TRAVIS_BUILD_DIR}:/home/test/↩
            shreyas911/my_docker_image:v1.0"

script:
        - ${DOCKER_RUN} /bin/bash -l -c "module load petsc && cd proj02 &&↩
            autoreconf -f -i && ./configure CC=mpicc CXX=mpicxx --enable-↩
           coverage && make && make check"

notifications:
        email: false
```

The Travis CI tests complete successfully, a screenshot of the sample output is given below. The build is #36 on travis.

```
                                                                                  0.07s
Worker information                                                  worker_info
                                                                                  0.01s
Build system information                                            system_info


                                                                                  2.60s
                                          docker_mtu_and_registry_mirrors
                                                                     resolvconf
$ sudo systemctl start docker                                          services   3.01s
Installing SSH key from: default repository key                         ssh_key
Using /home/travis/.netrc to clone repository.

$ git clone --depth=50 --branch=master https://github.com/uthpc/cse380-2020-student-Shreyas911.git   git.checkout   0.97s

$ export TRAVIS_COMPILER=gcc                                                      0.01s
$ export CC=${CC:-gcc}
$ export CC_FOR_BUILD=${CC_FOR_BUILD:-gcc}
$ gcc --version
gcc (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

$ docker pull shreyas911/my_docker_image:v1.0                  before_install.1   65.56s
$ export DOCKER_RUN="docker run -v ${TRAVIS_BUILD_DIR}:/home/test/ shreyas911/my_docker_image:v1.0"   before_install.2   0.00s
$ ${DOCKER_RUN} /bin/bash -l -c "module load petsc && cd proj02 && autoreconf -f -i && ./configure CC=mpicc CXX=mpicxx --enable-   128.64s
coverage && make && make check"
configure.ac:12: installing './compile'
configure.ac:9: installing './install-sh'
```

```
configure.ac:9: installing './missing'
src/Makefile.am: installing './depcomp'
parallel-tests: installing './test-driver'
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether the C++ compiler works... yes
checking for C++ compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C++ compiler... yes
checking whether mpicxx accepts -g... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of mpicxx... gcc3
checking for gcc... mpicc
checking whether we are using the GNU C compiler... yes
checking whether mpicc accepts -g... yes
checking for mpicc option to accept ISO C89... none needed
checking whether mpicc understands -c and -o together... yes
checking dependency style of mpicc... gcc3
checking how to run the C++ preprocessor... mpicxx -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for ANSI C header files... yes
checking for sys/types.h... yes
checking for sys/stat.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for memory.h... yes
checking for strings.h... yes
```

```
checking for inttypes.h... yes
checking for stdint.h... yes
checking for unistd.h... yes
checking masa.h usability... yes
checking masa.h presence... yes
checking for masa.h... yes
checking for masa - version >= 0.30... yes
checking for -lmasa linkage... yes
checking how to run the C preprocessor... mpicc -E
checking grvy.h usability... yes
checking grvy.h presence... yes
checking for grvy.h... yes
checking for grvy - version >= 0.32... yes
checking for -lgrvy linkage... checking for grvy_input_fopen in -lgrvy... yes
checking hdf5.h usability... yes
checking hdf5.h presence... yes
checking for hdf5.h... yes
checking for hdf5 - version >= 1.8.0... yes
checking for H5Fopen in -lhdf5... yes
checking for /opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/lib/petsc/conf/variables... yes
configure: PETSC install path found
checking for gfortran... gfortran
checking whether we are using the GNU Fortran compiler... yes
checking whether gfortran accepts -g... yes
checking for gcov... yes
checking for pow... no
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/Makefile
config.status: creating tests/Makefile
config.status: creating config.h
config.status: executing depfiles commands
make  all-recursive
```

```
make[1]: Entering directory `/home/test/proj02'
Making all in src
make[2]: Entering directory `/home/test/proj02/src'
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
main.o -MD -MP -MF .deps/main.Tpo -c -o main.o main.cpp
mv -f .deps/main.Tpo .deps/main.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
matrix_assemble.o -MD -MP -MF .deps/matrix_assemble.Tpo -c -o matrix_assemble.o matrix_assemble.cpp
mv -f .deps/matrix_assemble.Tpo .deps/matrix_assemble.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
q_assemble.o -MD -MP -MF .deps/q_assemble.Tpo -c -o q_assemble.o q_assemble.cpp
mv -f .deps/q_assemble.Tpo .deps/q_assemble.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
solvers.o -MD -MP -MF .deps/solvers.Tpo -c -o solvers.o solvers.cpp
mv -f .deps/solvers.Tpo .deps/solvers.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
T_exact_assemble.o -MD -MP -MF .deps/T_exact_assemble.Tpo -c -o T_exact_assemble.o T_exact_assemble.cpp
mv -f .deps/T_exact_assemble.Tpo .deps/T_exact_assemble.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
print.o -MD -MP -MF .deps/print.Tpo -c -o print.o print.cpp
mv -f .deps/print.Tpo .deps/print.Po
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
my_inputfile_parser.o -MD -MP -MF .deps/my_inputfile_parser.Tpo -c -o my_inputfile_parser.o my_inputfile_parser.cpp
mv -f .deps/my_inputfile_parser.Tpo .deps/my_inputfile_parser.Po
```

```
mpicxx -DHAVE_CONFIG_H -I. -I..      -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -
I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2 -MT
my_solver_class.o -MD -MP -MF .deps/my_solver_class.Tpo -c -o my_solver_class.o my_solver_class.cpp
mv -f .deps/my_solver_class.Tpo .deps/my_solver_class.Po
mpicxx   -I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include -DINCLUDE_PETSC -I/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/include
-I/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/include --coverage -g -O2   -o heat_solve main.o matrix_assemble.o q_assemble.o
solvers.o T_exact_assemble.o print.o my_inputfile_parser.o my_solver_class.o  -L/lib -lgrvy -Wl,-rpath,/lib -L/lib -lmasa -Wl,-
rpath,/lib -L/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/lib -lhdf5 -Wl,-rpath,/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/lib -
L/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/lib -L/opt/ohpc/pub/libs/gnu8/openmpi3/petsc/3.12.0/lib -
L/opt/ohpc/pub/libs/gnu8/openmpi3/scalapack/2.0.2/lib -L/opt/ohpc/pub/libs/gnu8/openblas/0.3.7/lib -
L/opt/ohpc/pub/libs/gnu8/openmpi3/hdf5/1.10.5/lib -L/opt/ohpc/pub/mpi/openmpi3-gnu8/3.1.4/lib -
L/opt/ohpc/pub/compiler/gcc/8.3.0/lib/gcc/x86_64-pc-linux-gnu/8.3.0 -L/opt/ohpc/pub/compiler/gcc/8.3.0/lib64 -
L/opt/ohpc/pub/compiler/gcc/8.3.0/lib -lpetsc -lscalapack -lopenblas -lhdf5 -lm -lstdc++ -ldl -lmpi_usempif08 -lmpi_usempi_ignore_tkr -
lmpi_mpifh -lmpi -lgfortran -lm -lgfortran -lm -lgcc_s -lquadmath -lpthread -lquadmath -lstdc++ -ldl
make[2]: Leaving directory `/home/test/proj02/src'
Making all in tests
make[2]: Entering directory `/home/test/proj02/tests'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/home/test/proj02/tests'
make[2]: Entering directory `/home/test/proj02'
make[2]: Leaving directory `/home/test/proj02'
make[1]: Leaving directory `/home/test/proj02'
Making check in src
make[1]: Entering directory `/home/test/proj02/src'
make[1]: Nothing to be done for `check'.
make[1]: Leaving directory `/home/test/proj02/src'
Making check in tests
make[1]: Entering directory `/home/test/proj02/tests'
make  check-TESTS
make[2]: Entering directory `/home/test/proj02/tests'
make[3]: Entering directory `/home/test/proj02/tests'
PASS: test.sh
============================================================================
Testsuite summary for FULL-PACKAGE-NAME VERSION
```

```
============================================================================
# TOTAL: 1
# PASS:  1
# SKIP:  0
# XFAIL: 0
# FAIL:  0
# XPASS: 0
# ERROR: 0
============================================================================
make[3]: Leaving directory `/home/test/proj02/tests'
make[2]: Leaving directory `/home/test/proj02/tests'
make[1]: Leaving directory `/home/test/proj02/tests'
make[1]: Entering directory `/home/test/proj02'
make[1]: Leaving directory `/home/test/proj02'
The command "${DOCKER_RUN} /bin/bash -l -c "module load petsc && cd proj02 && autoreconf -f -i && ./configure CC=mpicc CXX=mpicxx --
enable-coverage && make && make check"" exited with 0.


Done. Your build exited with 0.
```

Figure 13: Travis output