Partial Differential Equations 3 – Static PDE Coursework

By : Khalid Hersafril (s2016510)

Question 1

a) The boundary condition on the inside and outside of the pipe are given by:

$$\boldsymbol{u(1,\theta) = 0, \quad 0 \leq \theta \leq 2\pi} \qquad \textbf{(1)}$$

$$\boldsymbol{\frac{\partial u}{\partial r}(3,\theta) = \begin{cases} 0, & 0 \leq \theta < \pi \\ 1, & \pi \leq \theta \leq 2\pi \end{cases}} \qquad \textbf{(2)}$$

The first boundary condition where $u = 0$, tells us that the temperature throughout the whole boundary of the inside pipe at radius = 1, is 0.

The second boundary conditions where $\frac{\partial u}{\partial r} = 0$, from $0\ to\ less\ than\ \pi$, informs us that the boundary of the outside wall of the upper part of the pipe is a perfectly insulated wall where there is no heat gained or loss throughout the whole boundary whereas when $\frac{\partial u}{\partial r} = 1$, from $\pi\ to\ 2\pi$ tells us that the bottom part of the outside wall of the pipe is losing heat flux where the direction of the heat flux is normal to the domain.

b) We know that the basic solutions of cylindrical coordinate is given by:

$$\boldsymbol{\mu = 0} \qquad \boldsymbol{u(r,\theta) = (A\theta + B)(C\ln r + D)} \qquad \textbf{(3)}$$

$$\boldsymbol{\mu^2 < 0} \qquad \boldsymbol{u(r,\theta) = (A\cosh(\mu\theta) + B\sinh(\mu\theta))(Cr^{-\mu} + Dr^{\mu})} \qquad \textbf{(4)}$$

$$\boldsymbol{\mu^2 > 0} \qquad \boldsymbol{u(r,\theta) = (A\cos(\mu\theta) + B\cos(\mu\theta))(Cr^{-\mu} + Dr^{\mu})} \qquad \textbf{(5)}$$

Since we are applying methods of seperation variable, we can generally write equation [3], [4] and [5] as

$$\boldsymbol{u(r,\theta) = \emptyset(\theta)R(r)} \qquad \textbf{(6)}$$

Solving for equation [3] by applying the boundary conditions given from equation [1] and [2], we can see that;

$$\boldsymbol{\emptyset(\theta) = A\theta + B} \qquad \textbf{(7)}$$

Since our solution needs to be periodic, and the relationship given from equation [7] will give a linear relationship for increasing θ, hence we have to let $A = 0$, hence we get $\emptyset(\theta) = B$.

$$\boldsymbol{R(1) = C\ln 1 + D = 0} \qquad \textbf{(8)}$$

Applying the [1] boundary conditions, for equation [8], we get that the expression , $C\ln 1 = 0$, and this gives us that $D = 0$, however, we should not simply drop down the C term and make it equal to zero since coincidentally, $C\ln 1$ gives us zero. Therefore, we get $R(r) = C\ln r$. Substituting equation [7] and [8] into equation [6], we finally get that;

$$\boldsymbol{\mu = 0} \qquad \boldsymbol{u(r,\theta) = BC\ln r = B_n \ln r} \tag{9}$$

Since our solution needs to be periodic, solving equation [4] is trivial. Here, we can see that it equation [4] consists of hyperbolic functions which cannot be used to model periodic behaviors, hence we can let $A = B = 0$, hence leaving us with $u_2(r,\theta) = 0$.

Last but not least, again, since our solution is periodic and equation [5] consists of sin and cosine functions, hence, it needs to be in our solution. We will first use equation [5] to solve for the first boundary condition where $u(1,\theta) = 0$. Substituting this boundary condition into equation [5], we get;

$$\boldsymbol{u(1,\theta) = (A\cos(\mu\theta) + B\cos(\mu\theta))(C + D) = 0}$$

$$\boldsymbol{C = -D} \tag{10}$$

Substituting equation [10] into equation [5], and expanding the expression, we can find that;

$$\boldsymbol{u(r,\theta) = (A_k\, cos\,(\mu\theta) + B_k\, cos\,(\mu\theta)\,)(r^{-n} - r^n)} \tag{11}$$

where, $A_k = AC\ and\ B_k = BC$.

Superposing the solution we get from equation [9] and equation [11], we get the following equation;

$$\boldsymbol{u(r,\theta) = B_n \ln r + \sum_{n=1}^{\infty}[(A_k r^{-n} - A_k r^n)\cos(n\theta) + (B_k r^{-n} - B_k r^n)\sin(n\theta)]} \tag{12}$$

Partially differentiating the following equation with respect to r, we obtain;

$$\boldsymbol{\frac{\partial u}{\partial r}(r,\theta) = \frac{B_n}{r} + \sum_{n=1}^{\infty}[\left(-nA_k r^{-n-1} - nA_k r^{n-1}\right)\cos(n\theta) + \left(-nB_k r^{-n-1} - nB_k r^{n-1}\right)\sin(n\theta)]} \tag{13}$$

From the second boundary condition given from equation [2], we obtain;

$$\boldsymbol{\frac{\partial u}{\partial r}(r,\theta) = \frac{B_n}{3} + \sum_{n=1}^{\infty}[(-nA_k 3^{-n-1} - nA_k 3^{n-1})\cos(n\theta) + (-nB_k 3^{-n-1} - nB_k 3^{n-1})\sin(n\theta)]}$$

Applying fourier series coefficient to the following above equation, we can determine the $a_o, a_n\ and\ b_n$ value which is given as below:

$$\boldsymbol{a_o = \frac{2}{2\pi}\int_0^{2\pi}\frac{\partial u}{\partial r}(r,\theta)\,d\theta} \tag{14}$$

$$\boldsymbol{b_n = \frac{2}{2\pi}\int_0^{2\pi}\frac{\partial u}{\partial r}(r,\theta)\cos(n\theta)\,d\theta} \tag{15}$$

$$b_n = \frac{2}{2\pi}\int_0^{2\pi} \frac{\partial u}{\partial r}(r,\theta)\sin(n\theta)\, d\theta \tag{16}$$

Due to the fact that the boundary condition from $\mathbf{0}\ \boldsymbol{to}\ \boldsymbol{\pi}$ given from equation [2] is 0, we can futher simplify the above expression to

$$a_o = \frac{1}{\pi}\int_\pi^{2\pi} \frac{\partial u}{\partial r}(r,\theta)\, d\theta \tag{17}$$

$$a_n = \frac{1}{\pi}\int_\pi^{2\pi} \frac{\partial u}{\partial r}(r,\theta)\cos(n\theta)\, d\theta \tag{18}$$

$$b_n = \frac{1}{\pi}\int_\pi^{2\pi} \frac{\partial u}{\partial r}(r,\theta)\sin(n\theta)\, d\theta \tag{19}$$

Solving the equation [17] should be trivial, and it is just a matter of integrating $\frac{\partial u}{\partial r}$ **from** $\boldsymbol{\pi}\ \boldsymbol{to}\ \mathbf{2}\boldsymbol{\pi}$, which we can simply obtain from the boundary conditions, hence;

$$\frac{B_n}{3} = \frac{1}{2}a_o$$

Solving the above equation should also be trivial and we should obtain;

$$B_n = \frac{3}{2} \tag{20}$$

Solving for equation [18];

$$(-nA_k 3^{-n-1} - nA_k 3^{n-1}) = \frac{\sin(2\pi n)}{\pi n} = 0 \quad for\ all\ n = 1,2,3 \ldots.$$

Therefore;

$$A_k = 0 \tag{21}$$

Solving for equation [19];

$$(-nB_k 3^{-n-1} - nB_k 3^{n-1}) = \begin{cases} \frac{-2}{\pi n}, & n\ odd \\ 0, & n\ even \end{cases}$$

Therefore;

$$B_k = \frac{2}{\pi n^2(3^{-n-1} + 3^{n-1})} \tag{22}$$

Substituting our coefficients from equation [20], [21] and [22] to our solution in equation [12], we should get;

$$\boldsymbol{u}(\boldsymbol{r},\boldsymbol{\theta}) = \frac{3}{2}\ln r + \sum_{k=1}^{\infty}\left[\frac{2}{\pi(2k-1)^2(3^{-2k}+3^{2k-2})}(r^{1-2k}-r^{2k-1})\sin((2k-1)\theta)\right] \tag{23}$$

c) **Analytical Plot of solution**

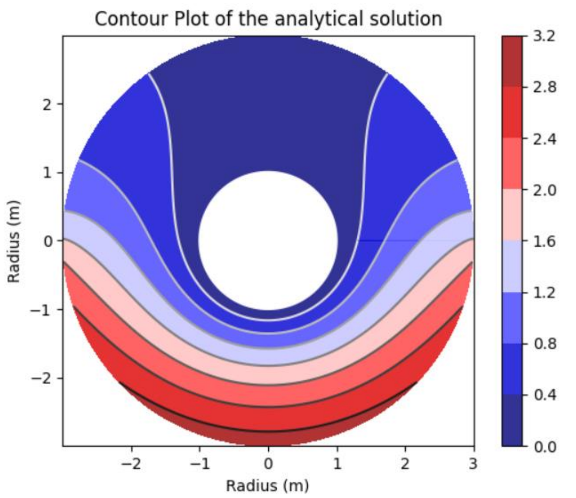

Figure 2. Contour plot of analytical solution

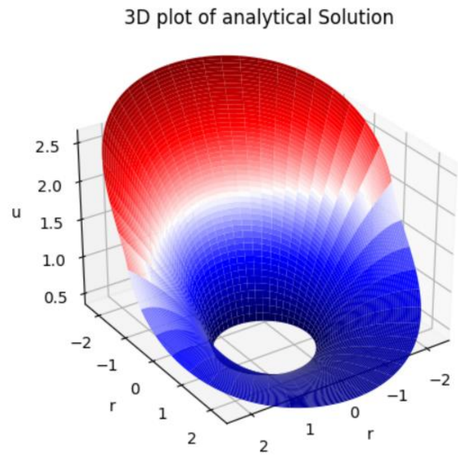

Figure 1a. Surface plot of analytical solution

From the plot of our analytical solution, we can see that the temperature of our pipe ranges from 0 to 3.2 °C. An important thing to first note is that our analytical solution agrees with the boundary condition given from equation [1] and [2]. We can see that the temperature throughout the whole inner diameter of the pipe is 0. Furthermore, it can also be observed that the contour lines are normal to the boundary which corresponds to the Nuemann Boundary Condition given from equation [2].

The fourier series converges very quickly where the rate of convergence depends on the denominator of the second term where it decreases at a rate of $\frac{1}{k^2 3^k}$. This simply means that very few terms are needed us to achieve a good approximation of our solution. In the figure above, I have use k=10, meanwhile k=3 also gives a very similar result.

Question 2 : Numerical Solution

Section A : Concept

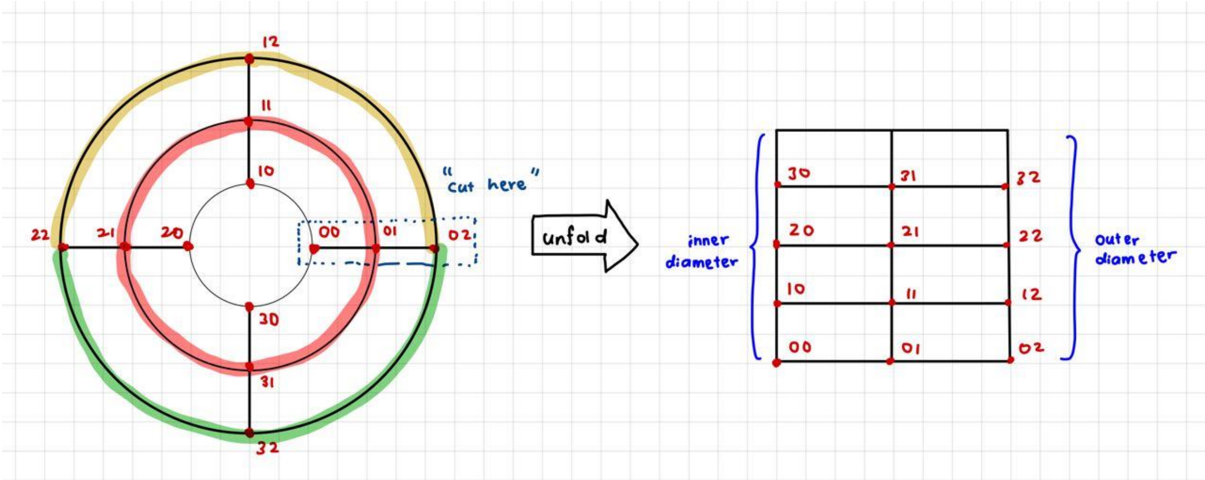

Figure 3. Visualization of nodes in cylindrical coordinate

Before we dive into the implementation of the numerical methods of our problem, it is best for us to first visualize how we should approach this problem. In figure 3, I have drawn an example of how we should look at this problem. The red dot resembles "nodes" of the grids that we will be implenting in our code. The numbers at the side of the nodes gives the coordinates of the nodes in terms of j,i (rows, columns).

Visualizing this problem in terms of circular grids makes it hard to understand, hence to simplify this problem in a way that is more understandable, it is best if we "cut" our circular grid at the first row and "unfolds" it to form a box as shown in the right diagram of figure 3. As labelled in the diagram, now our inner diameter of the pipe is now the left boundary (South boundary) meanwhile the outer diameter of the pipe being the right boundary (North boundary). The top and bottom boundary being the West and East Boundary respectively.

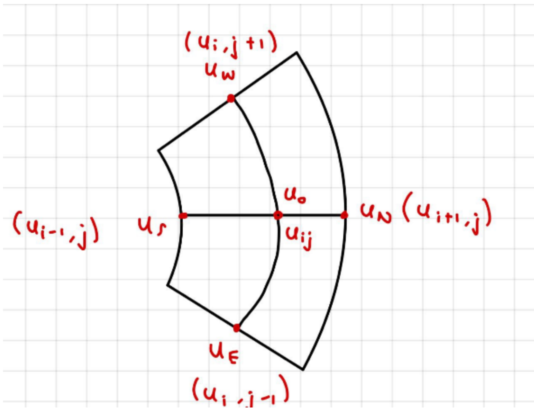

Figure 4. Compass direction of our circular grid

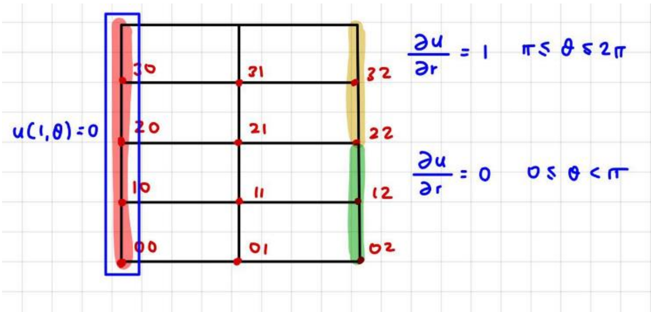

Figure 5. Boundary condition

To apply the boundary condition, since the leftmost boundary is our inner diameter of the pipe where radius = 1, hence from equation [1], we apply the Dirichlet Boundary Condition whereas on the rightmost boundary, from equation [2], we know that Neumann Boundary condition is applied.

Discretization of cylindrical coordinate

With the concepts explained above, it is much easier for us to visualize how we could possibly derive our discretized expression for cylindrical coordinate.

We know that our laplace equation is given by;

$$\frac{\partial^2 u}{\partial r^2} = \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} \tag{24}$$

Using Taylor Series Expansion, to obtain expression for $\frac{\partial u}{\partial r}, \frac{\partial^2 u}{\partial r^2}$ $\boldsymbol{and}$ $\frac{\partial^2 u}{\partial \theta^2}$, we get;

$$\frac{u_N - 2u_o + u_S}{\Delta r^2} + \frac{1}{r}\frac{u_N - u_S}{2\Delta r} + \frac{1}{r^2}\frac{u_E - 2u_o + u_W}{\Delta \theta^2} = 0 \tag{25}$$

where: $u_N = u_{i+1,j}, u_S = u_{i-1,j}, u_E = u_{i,j+1}, u_W = u_{i,j-1}$

Rearranging equation [25] and grouping similar terms, we obtain;

$$-2(\alpha + \beta)\, u_o + (\alpha + \beta)\, u_N + (\alpha - \beta)\, u_S - \gamma(u_E + u_W) = 0 \tag{26}$$

where:

- $\alpha = 2r\Delta\theta^2$
- $\beta = \frac{1}{2}r\Delta r\Delta\theta^2$
- $\gamma = \Delta r^2$

Code implementation

```
 alpha = 2 * r * dtheta**2
beta = 0.5 * r * dr * dtheta**2
gamma = dr**2

R_o = -2 * ( alpha + gamma)
R_n = alpha + beta
R_s = alpha - beta
R_ew = gamma
```

Figure 6. Defining the coefficient of our stencil

This code implementation refers to the BICGStab notebook, and trivial implementation such as how the iteration is implemented will not be discussed in this section.

In figure 6, I have defined the coefficients for each of our stencil, in order to ease readability of our code. The definition being the same as equation [26].

```
            A_mat[k, k] = R_o


            # Left boundary (DIRICHLET)
            # South
            if i > 1:
                A_mat[k, k - 1] = R_s
            else:
                b_vec[k] += - R_s * grid.u[j, i - 1]
```

We will first set our origin coefficient with A_mat[k, k] and set the Dirichlet Boundary condition given from equation [1]. We will iterate through each columns and if the colums are at the leftmost of our grid, we will place our known boundary condition (Dirichlet BC) to the b vector.

```
            # Right boundary (NEUMANN)
            # North
            if i < grid.Ni - 2:
                A_mat[k, k + 1] = R_n
            else:
                A_mat[k, k - 1] += R_n
                if theta >= np.pi:
                    b_vec[k] += - 2 * dr * R_n
```

Here, the same method is applied where we apply the North coefficients to the right of our origin. However, the tricky part here is that at the North boundary, we have Neumann Boundary condition in which we do not have the u value. However, we can possibly approximate the value with the following equation;

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{r}}(\mathbf{3}, \boldsymbol{\theta}) \approx \frac{\boldsymbol{u_s} - \boldsymbol{u_N}}{\mathbf{2}\Delta \boldsymbol{r}} = \begin{cases} \mathbf{0}, & \mathbf{0} \leq \boldsymbol{\theta} < \boldsymbol{\pi} \\ \mathbf{1}, & \boldsymbol{\pi} \leq \boldsymbol{\theta} \leq \mathbf{2}\boldsymbol{\pi} \end{cases} \tag{27}$$

This can further be re-arranged to form;

$$\boldsymbol{u_N} = \boldsymbol{u_S} - \mathbf{2}\Delta \boldsymbol{r}$$

We need not to worry about the boundary condition from $0 \; to \; \pi$ because our matrix assembly has already filled in zero values by default. However, we still have to update our A_matrix nevertheless. The most important part of this implementation is knowing how to effectively include the North Boundary condition (Neumann BC) since u_N is not inside the inner domain. From the equation obtained above, we find that for $\boldsymbol{\pi} \leq \boldsymbol{\theta} \leq \mathbf{2}\boldsymbol{\pi}$**,** we would have to add a term to our **b** vector which is $-2\Delta r R_n$ to apply our boundary condition and this can be proven by substituting the equation above into the discretized expression in order for the expression to only be valid at the boundary.

```
            # Top boundary (PERIODIC)

            if j < grid.Nj - 2:
                A_mat[k, k + (grid.Ni - 2)] = R_ew
```

```
            # Bottom boundary (PERIODIC)
            if j > 1:
                A_mat[k, k -(grid.Ni - 2)] = R_ew
```

This part of the problem is quite tricky to solve, although the matrix assembly is correct, however, the numerical solution does not come out as expected and it is a bit off from our expected solution.

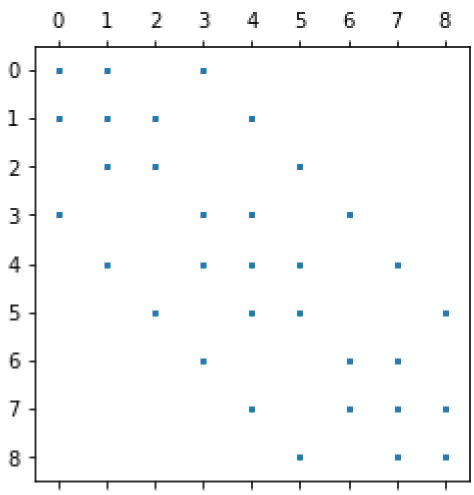

Figure 7. Spy Assembly matrix

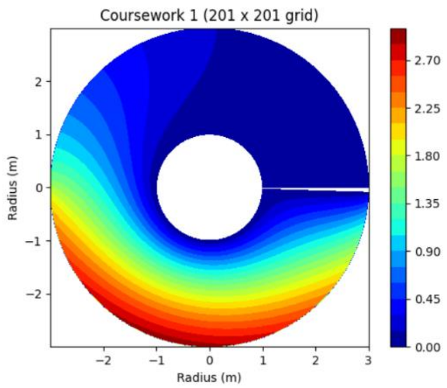

Figure 8. Numerical solution of pipe

Using matplotlib.pyplot.spy() tool, I checked if my assembly matrix is right or wrong. It turns out that the matrix assembly is correct with the coefficients in the right place. Figure 7 is a matrix assembly for a **5 x 5** grid. A quick observation of the following figure informs us that the coefficients are assembled in the right position in our matrix. However, the results of our solution does not come out correct which can be seen in figure 8 (using **201 x 201 grid**) for ease of observation and discussion.

Nevertheless, the contour plot obtained in figure 8, matches our analytical solution but have a slightly lower value with approximately 2.75 °C at peak.

It is also interesting to note that as the grid size increases, the small gap at $\theta = 0$, decreases due to how we have defined our angle with;

```
theta = np.linspace(0.0, 2*np.pi, self.Ni+1)
```

For comparison, I have included figure 9.

Last but not least, as the grid size increases, the numerical solution becomes extremely slow hence we would have to refactor the code to improve efficiency.

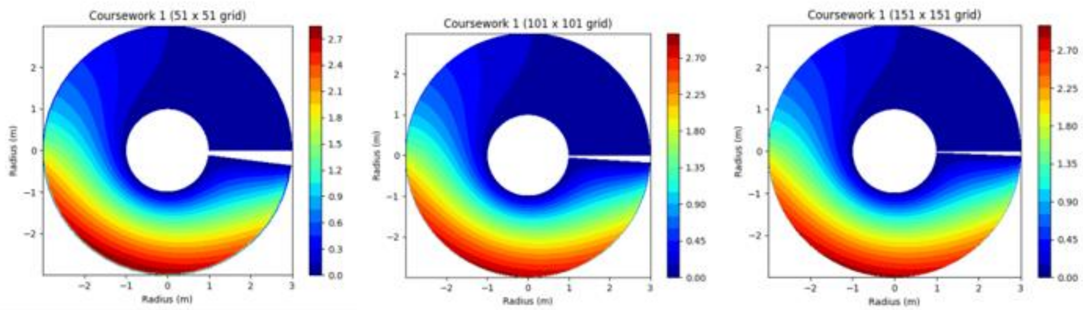

Figure 9. Comparison of different grid sizes