Applied Numerical methods, Group Assignment 2

Klara Zimmerman and Ville Wassberg

October 2023

Problem 1

In the first part of this lab, we used the finite difference approximation method in 1D to determine the steady state temperature distibution, $T(z)$, for a moving fluid inside a short cylinder (defined in the region where $0 \leq z \leq 1$ length units) that is being heated in a small section (between z_{in} and z_{out}) such that the heat convects with the fluid. The problem is modelled through the following equation:

$$-\frac{d^2T}{dz^2} + v\frac{dT}{dz} = Q(z), \tag{1}$$

where

$$Q(z) = \begin{cases} 0, & 0 \leq z < 0.1 \\ 7000 * \sin\left(\frac{(z-0.1)\pi}{0.3}\right), & 0.1 \leq z \leq 0.4 \\ 0, & 0.4 < z \leq 1. \end{cases}$$

We are given the following Dirichlet boundary condition at $z = 0$, and Robin boundary condition at $z = 1$, respectively:

$$T(0) = 100$$

$$-\frac{dT(1)}{dz} = \alpha(v)(T(1) - T_{out}),$$

where $T_{out} = 25$ and $\alpha(v) = \sqrt{\frac{v^2}{4} + \alpha_0^2} - \frac{v}{2}$.

a)

We start with the case when the fluid velocity coefficient is set to one, $v = 1$. To solve the equation using finite difference approximation, we first discretize the z-interval. By discretizing into N sub-intervals, we get $N + 1$ points in total, hence $N - 1$ inner grid points.

We rewrite (1), using the central difference approximation of the derivatives in the point z_i:

$$-T''(z_i)+vT'(z_i) \approx \frac{-T(z_i + h) + 2T(z_i) - T(z_i - h)}{h^2} + v\frac{T(z_i + h) - T(z_i - h)}{2h} =$$

$$= \frac{1}{2h^2}((-2 - vh)T(z_i - h) + 4T(z_i) + (-2 + vh)T(z_i + h)) = Q(z_i).$$

Letting $a = \frac{-1}{h^2} - \frac{v}{2h}$, $b = \frac{2}{h^2}$ and $c = -\frac{1}{h^2} + \frac{v}{2h}$, we obtain:

$$aT_{i-1} + bT_i + cT_{i+1} = Q(z_i). \tag{2}$$

We are given the value $T(z_0)$, but want to use equation 2 to approximate the function values $T(z_i)$ when $1 \leq i \leq N$, that is we want to solve a system of N equations.

For the first equation, when $i = 1$, we have $aT_0 + bT_1 + T_2 = Q(z_1)$. We can use our first boundary condition, $T(0) = 100$, to rewrite this as $Q(z_1) - aT_0 = bT_1 + cT_2$.

For the last equation, when $i = N$, we get

$$aT_{N-1} + bT_N + cT_{N+1} = Q(z_N). \tag{3}$$

The point T_{i+1} is outside of our interval, so we use the ghost point method to construct this point using the given Robin condition. This is done by first approximating the given derivative at z_N, again using the central difference approximation, so that the method is still of order 2.

$$\alpha(v)(T(z_N) - T_{out}) = -T'(z_N) \approx -\frac{T(z_N + h) - T(z_N - h)}{2h}$$

$$\iff T_{N+1} = 2 * T_{out}h\alpha(v) - 2h\alpha(v)T_N + T_{N-1}.$$

We label these coefficients $d_{N-1} = 2 * T_{out}h\alpha(v)$, $d_N = -2h\alpha(v)$ and $d_{N+1} = 1$, so we get

$$T_{N+1} = d_{N-1} + d_N T_N + d_{N+1}T_{N+1}. \tag{4}$$

Inserting 4 into 3, we get:

$$Q(z_N) = aT_{N-1} + bT_N + c(d_{N-1} + d_N T_N + d_{N+1}T_{N+1}).$$

Rearranging, this gives us:

$$Q(z_N) = (a + cd_{N-1})T_{N-1} + (b + cd_N)T_N + cd_{N+1}.$$

$$\iff Q(z_N) - cd_{N+1} = (a + cd_{N-1})T_{N-1} + (b + cd_N)T_N.$$

Using the ghost point method, we have hence found a way to express equation 3 without T_{N+1}, while preserving second order accuracy of the method.

We now want to set up matrices A and $\mathbf{f}$, so that $A\mathbf{T} = \mathbf{f}$ solves our system of N equations, that is we want to find the vector $\mathbf{T} = (T_1, T_2, \dots, T_{N-1}, T_N)'$. From our discretization of z and our boundary conditions, we summarize our N equations below.

$$\begin{cases} bT_1 + cT_2 = Q(z_1) - aT_0, & \\ aT_{i-1} + bT_i + cT_{i+1} = Q(z_i), & \text{for } 2 \le i \le N-1, \\ (a + cd_{N-1})T_{N-1} + (b + cd_N)T_N = Q(z_N) - cd_{N+1}. & \end{cases} \tag{5}$$

Equations (5) are combined into matrices A, $\mathbf{f}$ and $\mathbf{T}$, which give us the following setup:

$$\begin{pmatrix} b & c & 0 & 0 & \dots & 0 \\ a & b & c & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & a & b & c \\ 0 & \dots & \dots & 0 & (a + cd_{N-1}) & (b + cd_N) \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ \vdots \\ T_{N-1} \\ T_N \end{pmatrix} = \begin{pmatrix} Q(z_1) - aT_0 \\ Q(z_2) \\ \vdots \\ \vdots \\ Q(z_{N-1}) \\ Q(z_N) - cd_{N+1} \end{pmatrix}$$

Since all elements in A and $\mathbf{f}$ are known, we use MATLAB's bulit in backslash method to solve for $\mathbf{T}$. In figure 1 we plot the solution vector $\mathbf{T}$ for different stepsizes, $h = 1/N$, when $N = 10$, $N = 20$ $N = 40$ and $N = 80$.

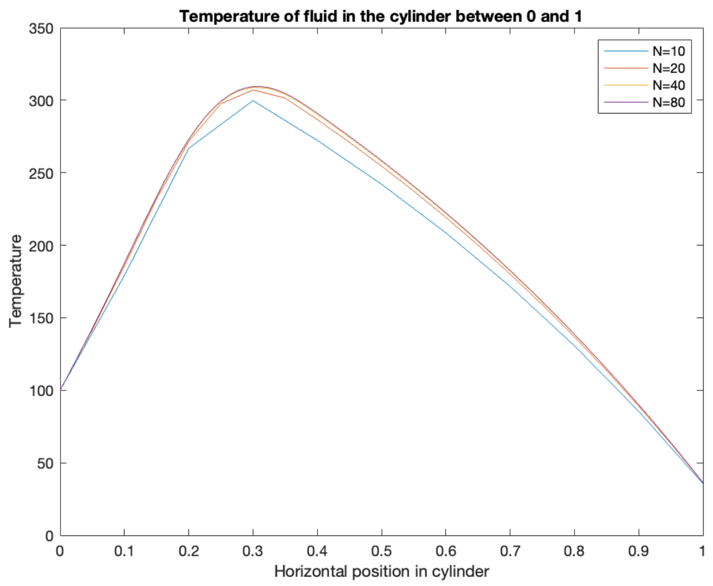

Figure 1: The Temperature T as a function of the position in the cylinder, z, plotted for different stepsizes (ie different values of N), using the finite difference approximation in $1D$.

For larger values of N, we find $T(0.5)$, ie the temperature when the horizontal position in the cylinder equals 0.5. This is presented in figure 2.

To verify that our approximation exhibits second order accuracy, we computed the solution up to when $N = 30000$, for which the solution converged to $T(0.5) = 258.6853$ (for $N \geq 35000$, the array exceeds MATLAB's maximum size preference). Using this as our "true" value, we computed the absolute error when $N = 80$, $N = 160$ and $N = 320$, which is also reported in figure 2. Squaring the error when $N = 80$ gives us approximately the error when $N = 2 * 80 = 160$, and squaring the error when $N = 160$ gives us approximately the error when $N = 2 * 160 = 320$, indicating that our method indeed converges with second order accuracy.

N	T(0.5)	Absolute error
80	258.4308	0.2545
160	258.6217	0.0636
320	258.6794	0.0059

Figure 2: The temperature in the cylinder when $z = 0.5$, approximated using the central difference approximation with different values of N.

b)

In this part we now solve the same equation as in a), however, using four different values of the fluid velocity, $v: 1, 5, 15, 100$. For larger v, we increase N, hence decrease the stepsize, which together with the results are plotted in the same graph as shown in figure 3.

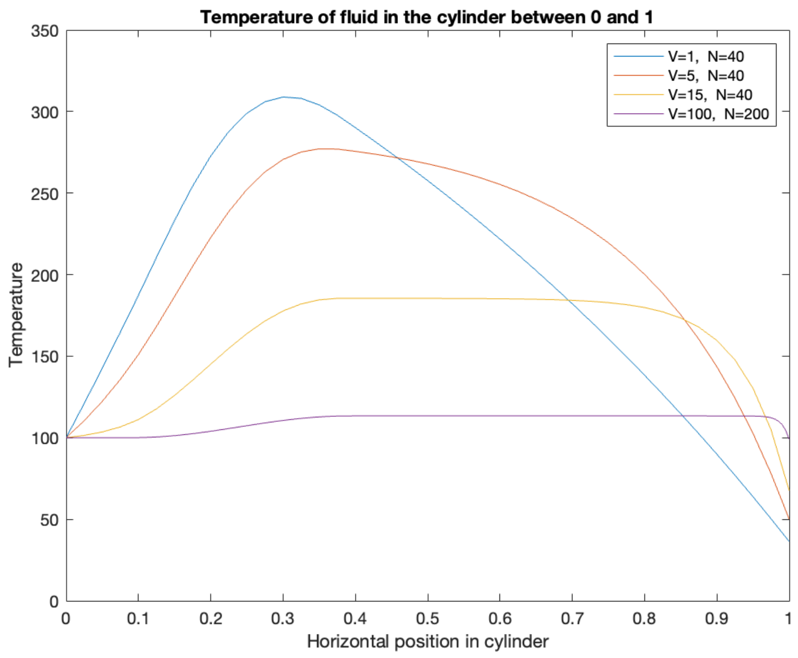

Figure 3: The solution $T(z)$ plotted for different values of v, with fitting stepsizes, $1/N$.

As we see in our differential equation (1), v affects the first derivative of T, that is the fluid velocity in the cylinder. As v increases, our plot shows that the maximum temperature of the fluid obtained inside the cylinder decreases. This seems reasonable, as the fluid spends less time being heated. The second derivative is, however, always negative, hence each curve will have a concave graph. Thus, the results are as expected and the temperature of the fluid will not be as significantly affected by the heat source when the fluid velocity is larger and the exposure to the heat source is shorter.

Problem 2

In this problem, we use the Finite Difference Method in 2D, where we consider a rectangular metal block with heat conducted through it. The block is defined by the region $\Omega = [0 < x < L_x, 0 < y < L_y]$, in $\mathbb{R}^2$. The block is insulated at all sides except at $y = 0$, where the external temperature which is held constantly at 25 sets the temperatur along the boundary. The elliptic problem is formulated as follows:

$$\begin{cases} T(x,0) = T_{ext}, & 0 < x < L_x, \\ \frac{\partial T}{\partial x}(0,y) = 0, & 0 < y < L_y, \\ \frac{\partial T}{\partial x}(L_x,y) = 0, & 0 < y < L_y, \\ \frac{\partial T}{\partial y}(x,L_y) = 0, & 0 < x < L_x, \\ \text{where } L_x = 12, L_y = 5, T_{ext} = 25. \end{cases} \tag{6}$$

We begin by discretizing the region in Matlab by creating two vectors representing the x and y steps, and creating a mesh using matlabs in-built function Meshgrid. We discretized the region with the same stepsize in both directions $h = L_x/N = L_y/M$, where M is the amount of steps in y-direction and N in x-direction, in both directions; where $(N-1)(M-1)$ are the amount of inner gridpoints. We use the Ghost point method to implement the boundary conditions (BC), from where we defined a matrix A by using the central difference method to express the ghost points in terms of the inner grid points, with the aim of obtaining a second order method. This is done in the following equations.

$$\begin{aligned} -\Delta T = -(T_{xx} + T_{yy}) \approx -\frac{1}{h^2}(T(x_i+h,y_i) + T(x_i-h,y_i) + T(x_i,y_i+k) + \\ T(x_i,y_i-k) - 4T(x_i,y_i)) = 2, \text{ for } 1 \leq x \leq L_x - h, 1 \leq y \leq L_y - h, \end{aligned} \tag{7}$$

and the first order derivatives were approximated as

$$\begin{cases} 0 = \frac{\partial T}{\partial x} \approx \frac{T(x_i+h)-T(x_i-h)}{2h} \Rightarrow T(x+h) = T(x-h), \text{ for } T(0,y), T(L_x,y), \\ 0 = \frac{\partial T}{\partial y} \approx \frac{T(y_i+h)-T(y_i-h)}{2h} \Rightarrow T(y+h) = T(y-h), \text{ for } T(x,L_y). \end{cases}$$

Hence, by plugging in the first order derivatives at the boundaries $x = 0$, $x = L_x$ and $y = L_y$ respectively, to the general form (7), we get:

$$\begin{cases} -\Delta T \approx -\frac{1}{h^2}(2T(x+h,y_i)+T(x,y_i+k)+T(x,y_i-k)-4T(x,y_i)) = 2, \\ \text{for } x=0, 1 \le y \le L_y - h, \\ -\Delta T \approx -\frac{1}{h^2}(2T(x-h,y_i)+T(x,y_i+k)+T(x,y_i-k)-4T(x,y_i)) = 2, \\ \text{for } x=L_x, 1 \le y \le L_y - h, \\ -\Delta T \approx -\frac{1}{h^2}(T(x_i+h,y)+T(x_i-h,y)+2T(x_i,y-k)-4T(x_i,y)) = 2, \\ \text{for } 1 \le x \le L_x - h, y = L_y, \end{cases} \tag{8}$$

Since we have defined $N-1$ inner grid points along the x-axis, and then discretized two Neumann BCs at $x=0$ and $x=L_x$ respectively, we get $N+1$ equations to solve for each value of y. For the y-axis we have defined $M-1$ inner grid points. We know the value at $y=0$ through the Dirichlet BC, and we discretize the Neumann BC at $y=L_y$. This generates M equations to solve at each value of x.

Through (7) and (8) we hence obtain a system of $M(N+1)\times M(N+1)$ equations to solve in total. These are represented in a system $A\mathbf{T}=\mathbf{f}$, where A is the matrix of discretized points consisting of M blocks each of size $N+1$. The solution vector, $\mathbf{f}$, accordingly consists of M block vectors each of length $N+1$.
Below is an example of matrix A, for the simple case where $N=2$ and $M=3$.

$$A_{(N,M)=(2,3)} = -\frac{1}{h^2}\begin{pmatrix} -4 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -4 & 2 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 2 & -4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 & 0 & -4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & -4 \end{pmatrix} \tag{9}$$

Hence, the diagonal of A consists of M blocks of size $(N+1)\times(N+1)$. We call these blocks S_N, and define their general form as following:

$$S_N = \begin{pmatrix} -4 & 2 & 0 & 0 & 0 & \dots & 0 \\ 1 & -4 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & -4 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & 1 & -4 & 1 \\ 0 & \dots & \dots & 0 & 0 & 2 & -4 \end{pmatrix}$$

The superdiagonal blocks of A consist of the identity matrix I_{N+1}. As do the subdiagonal blocks of A, with the exception of the last block which instead consists of $2 * I_{N+1}$ in order to account for the ghost points created to derive $T(y)$ at $y = L_y$. As we can see, the dimensions of A are hence $(M(N+1)) \times (M(N+1))$.

Thus, when defining the vector $\mathbf{f}$ we had to add $\frac{1}{h^2}T_{ext}$ to the first $N+1$ elements for the vector corresponding to $y = 0$. The final equation to be solved with Matlab's backslash method, was $A\mathbf{T} = \mathbf{f}$, where T and f are $M \times 1$ block vectors of $(N+1) \times 1$ vectors, each related to their corresponding y-value.

a)

In this part of the problem we plot the numerical solution, using the method described above, when the surrounding function is constant, 2. We discretize the mesh with 60 steps in each direction, and the stepsize is thus 0.2. Furthermore, we extract the temperature evaluated at the point $(x, y) = (6, 2)$, which we just used the convenient indexing for matrices in Matlab writing $T(2/h + 1, 6/h + 1)$ for T being our soulution matrix which is the same as the vector $\mathbf{T}$ earlier described but with each block as a row of the matrix. The extracted value when $N = 60$ at the point $(6, 2)$ is 41.0000.

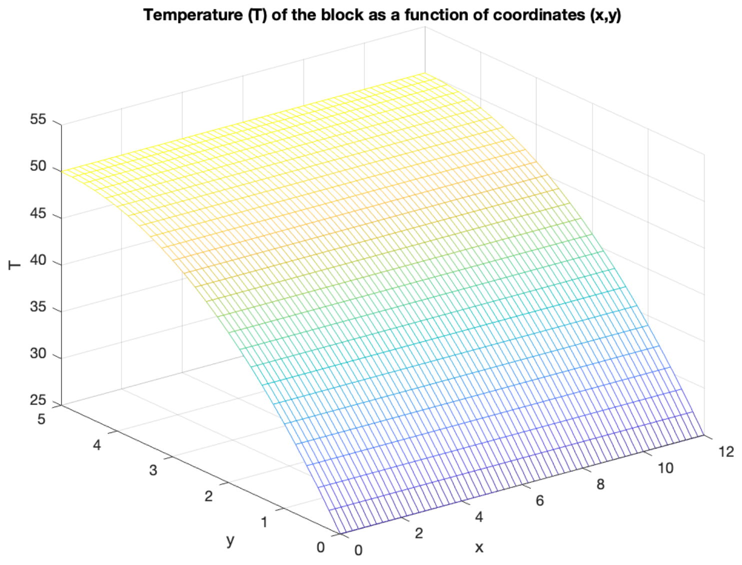

Figure 4: Plot of the numerical solution using Matlab's mesh function, with $N = 60$, step size $h = 0.2$, when the surrounding heat source is defined by $f(x, y) = 2$.

b)

When **f** is constant, as in a), the exact solution can be described as a second order polynomial depending on y, $T(x, y) = c_0 + c_1 y + c_2 y^2$. Thus, when using the boundary conditions given by 6, we get the following system of equations:

$$\begin{cases} T(x,0) = c_0 = T_{ext} = 25, \\ \frac{\partial T}{\partial x}(x, L_y) = c_1 + 2c_2 * 5 = 0, \text{ where } L_y = 5, \\ -\Delta T = -2c_2 = 2. \end{cases}$$

Solving for the coefficients we get the equation

$$T(x, y) = 25 + 10y - y^2. \tag{10}$$

c)

Comparing the results in a) and b) for the point $(x, y) = (6, 2)$ we see that they give the exact same value, since (10) gives $T(6, 2) = 25 + 20 - 4 = 41$. Thus, verifying that the numerical solution in a) is also the exact solution.

In order to find the approximation of the error of the central difference method when $x = c$, where c is a constant we can expand it as follows.

$$T''(c, y) = \frac{T(c, y + h) - 2T(c, y) + T(c, y - h)}{h^2} + R(y) =$$
$$\frac{25 + 10(y + h) - (y + h)^2 - 2(25 + 10y - y^2) + 25 + 10(y - h) - (y - h)^2}{h^2} + R(y) =$$
$$= \frac{-2h^2}{h^2} + R(y) = -2 + R(y).$$

Subtracting the analytical second derivative of (10) we see that the error is 0. Thus, the numerical and the analytical solution are the same when the surrounding function is kept constant!

d)

Now, setting the heat source to $f(x, y) = 100exp(-\frac{1}{2}(x - 4)^2 - 4(y - 1)^2)$, makes the problem more interesting and we cannot describe the solution as a second order function of only y. Here we compute the values of $T(6, 2)$ using three different step sizes: 0.2, 0.1, 0.05. This is shown by following table.

h	**N**	**T(6,2)**
0.2	60	47.2201
0.1	120	47.2240
0.05	240	47.2250

Figure 5: Values at T(6,2) when $f = 100exp(-\frac{1}{2}(x - 4)^2 - 4(y - 1)^2)$

We also plot the solutions when $N = 120$ $(h = 0.1)$, using Matlab's three inbuildt functions: mesh, imagesc and contour. These plots clearly visualises how the heat source is affecting the rectangular metal block's temperature along the surface.

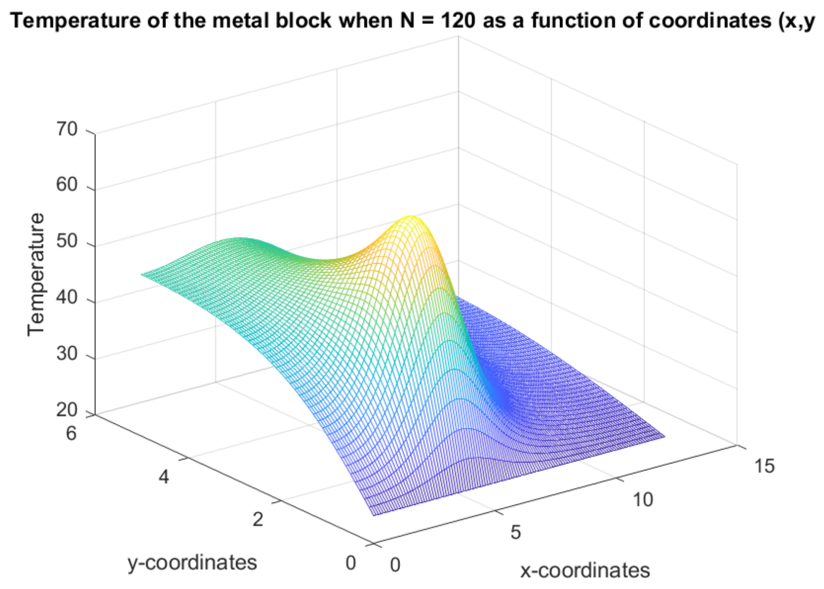

Figure 6: Plot of the numerical solution using Matlab's mesh function, with $N = 120$, $h = 0.1$ when the surrounding heat source is defined by $f(x,y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$.

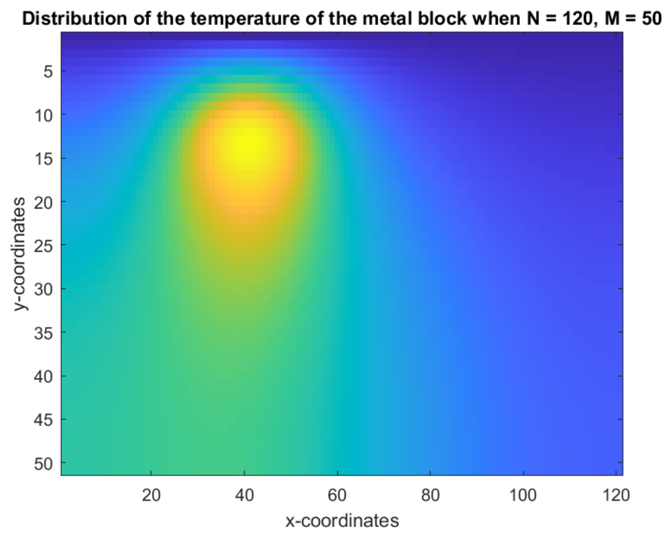

Figure 7: Plot of the numerical solution using Matlab's imagesc function, with $N = 120$, $h = 0.1$ when the surrounding heat source is defined by $f(x, y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$.

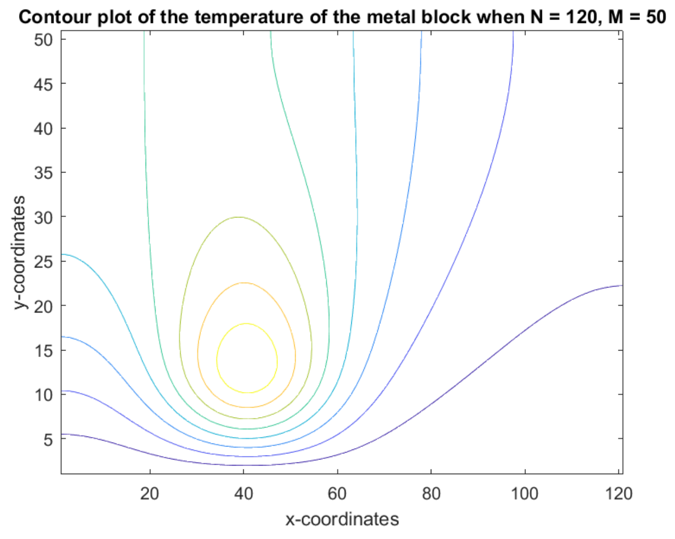

Figure 8: Plot of the numerical solution using Matlab's contour function, with $N = 120$, $h = 0.1$ when the surrounding heat source is defined by $f(x, y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$.

Problem 3

In this problem we ought to solve the same problem as in Problem 2, with some changes to make it more physically interesting, using Comsol. A heads up: When observing the different plots in this problem, keep in mind that different plots might have the same colour even though the temperature is completely different.

a)

In this part we solve the same problem as in Problem 2 part d) using Comsol. We start by creating the domain (the metal block) and let Comsol triangulate it by Comsol's default settings. When that is made, we refine the triangulation of the area to make a more accurate solution, and we stop when the difference is neglectable in order to obtain a solution as precise as possible. At the point $(x, y) = (6, 2)$, Comsol gives us the result shown in figure 9.

T(6,2)	**mesh domain elements**	boundary elements
47.2264	262	44
47.2254	23196	424
47.2253	92396	848
47.2253	389000	1692

Figure 9: Temperature measured in metal block at point $(6, 2)$, approximated by Comsol using different amount of mesh domain elements.

When analysing the different results for different amount of mesh domain elements, we deside to continue as in the third row of the table above, using 92396 mesh domain elements, since it is not too computationally demanding, and as shown, accurate enough.

b)

In this part, we change the conditions for the problem. Now the Neumann and Robin conditions are defined as follows:

$$\begin{cases} \frac{\partial T}{\partial y}(x,0) = 0, & 0 < x < L_x, \\ -\frac{\partial T}{\partial n} = \alpha(T - T_{ext}), & \alpha = 0.06,\ T_{ext} = 25. \end{cases} \tag{11}$$

The second row of (11), the Robin condition, is modeling the block's top edges as being cooled by air flow, while the first is making the block's bottom edge at $y = 0$ being modeled as if it is insulated. We set these conditions using Comsol's inbuilt Flux/Source settings under the Poisson equation alternatives. When computing Comsol's solution we get an average temperature at the top edge, $y = 5$, as 193.9779, when using the built in boundary probe average method. Figure 10 visualises the distribution of the temperature during these conditions.

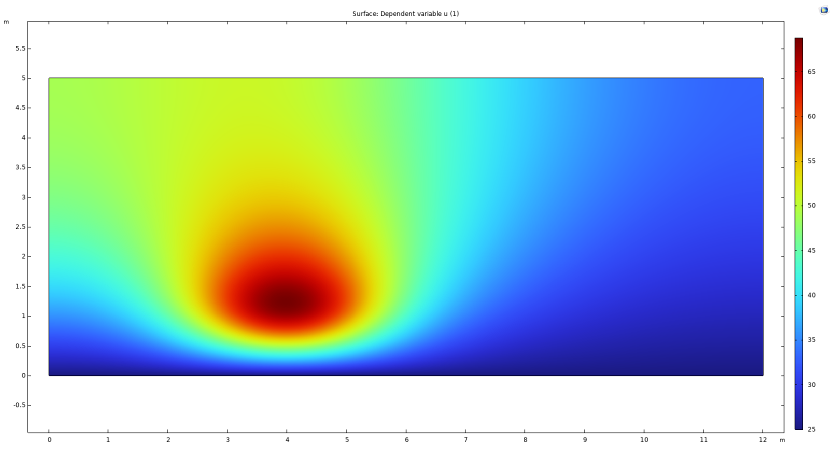

Figure 10: Plot of the temperature of the block when the surrounding heat source is defined by $f(x, y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$

c)

A ventilation is needed! Now we drill a hole in the metal block centred at the point $(x, y) = (6.0, 3.5)$ with radius one unit of length. The same Robin conditions as in b) are used at the boundaries of this hole, too. When computing this, we get the top edge average at $y = 5$ as 147.8523. The result is plotted in figure 11.

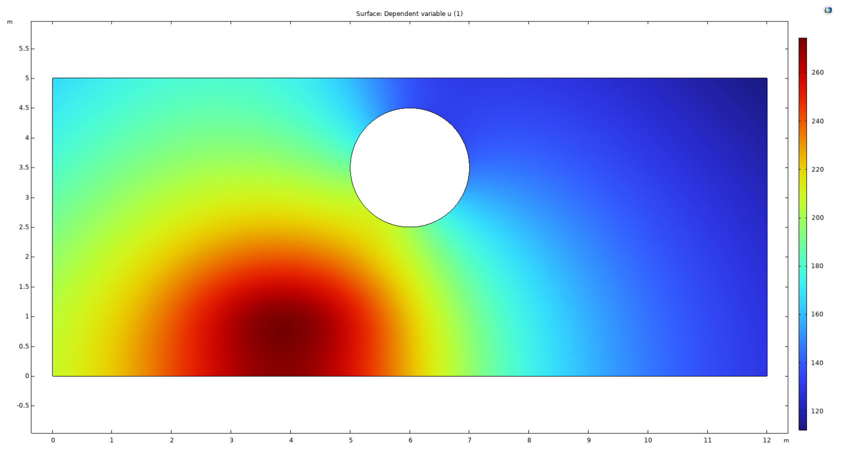

Figure 11: Heat block with a hole with radius 1, computed in 2D using Comsol. The heat source is defined by $f(x, y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$

d)

In this part we mix with the cooling a bit. Now instead of one hole, we have four smaller ones with radius 0.4 and centred at the points $(x, y) = (6.0 \pm 0.5, 3.5 \pm 0.5)$. The top edge average at $y = 5$ in this part is 134.0991, hence cooler than previous experiment. The block is visualised in figure 12.

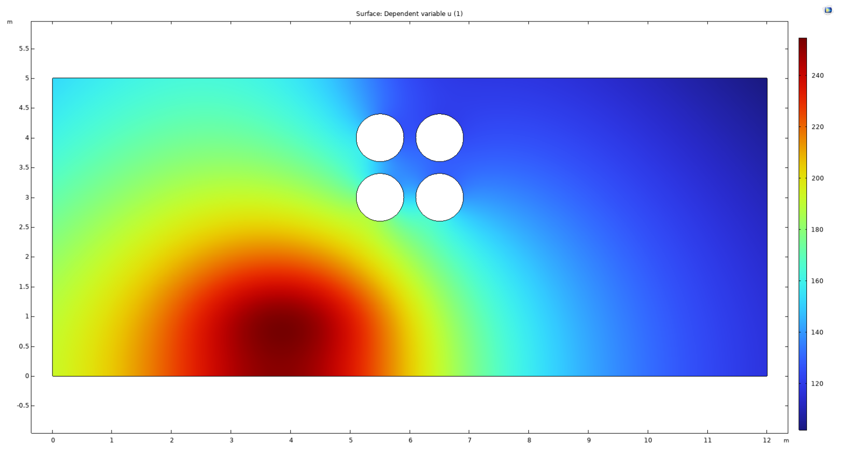

Figure 12: Heat block with 4 holes with radius 0.4, computed in 2D using Comsol. The heat source is defined by $f(x,y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$

In figure 13 we plot how Comsol triangulates the mesh of this problem. As we can see it is making the triangles smaller around the boundaries of the circles cooling the block, which is in accordance with the physical aspects of the problem, since the heat distribution is affected in a larger scale around the ventilation.

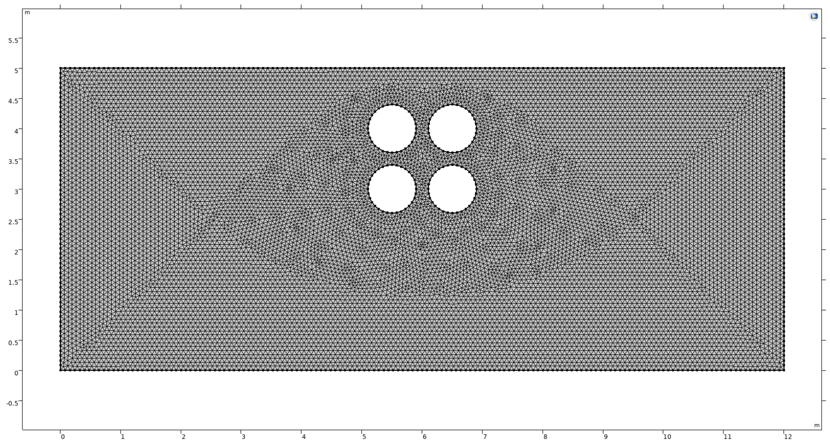

Figure 13: Mesh of the heat block with 4 holes with radius 0.4, computed in 2D using Comsol. Here we see that the triangles are much smaller around the outflow.

e)

In this part we design a cooling area of our own within the domain $(x, y) = (6.0\pm1.5, 3.5\pm1)$. We decided to make a grille made of ten smaller renctangular holes, continuing to explore the effects a large surface area in comparison to the volume. With this grille we obtain a top edge average of 80.5717, thus our theory is corroborated. How the temperature is distributed is visualised in figure 14, as well as the mesh created by Comsol in figure 15, again, in accordance with the physical expectations regarding the lowered average temperature.

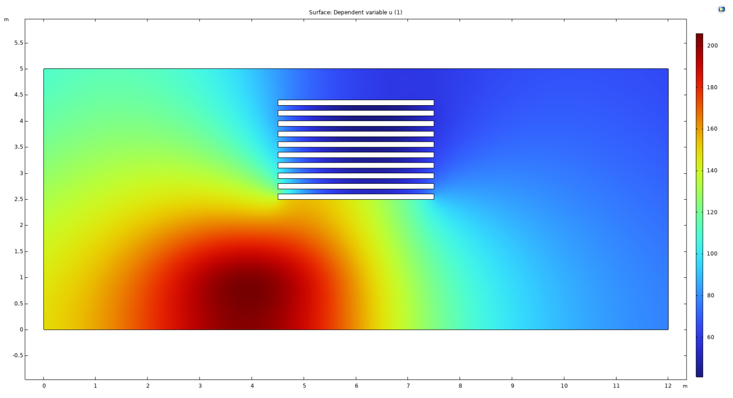

Figure 14: Mesh of the heat block with a grille, computed in 2D using Comsol. The heat source is defined by $f(x, y) = 100exp(-\frac{1}{2}(x-4)^2 - 4(y-1)^2)$

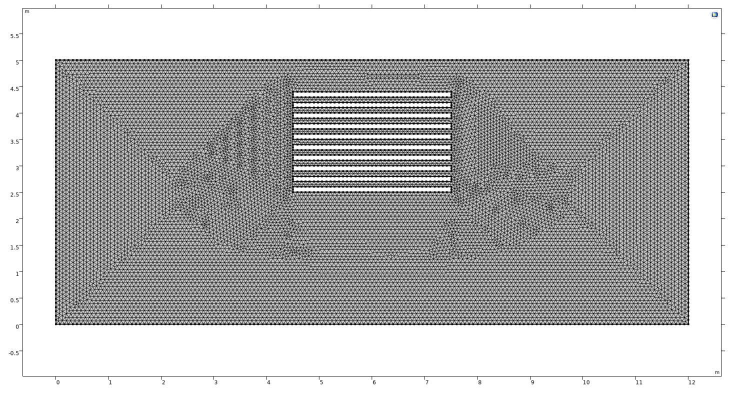

Figure 15: Mesh of the heat block with a grille, computed in 2D using Comsol. Here we see that the triangles are much smaller around the outflow.

However, the grille could most likely be improved by increasing the surface area of it and perhaps rotate it a bit. We reckon the result to be physically relevant, considering the similarity to heatsinks used in many applications. This particular ventilation alternative could probably easily be manufactured, considering the simple and regular shape of the grille. Although, if one continued to increase the surface area exposed to the surounding air, one could end up using millions of fractals as the ventilation, which would plausibly be quite hard to manufacture efficiently.