Assignement 1 (SF2521) - Heat Equation

SF2521

Heat equation in 2D with source

Auteur :
Valentin Duvivier

Encadred by :
Patrick Henning
Johan Wärnegård

February 20th 2020

1 Introduction

The main objective of this assignment is to study and solve the heat equation on the 2D case. The function to be solved and the condition bounded to it are listed below :

$$\begin{cases} q_t - \Delta q = S & (x,y) \in [0,1] * [0,1] \quad t > 0 \\ q(x,y,0) = 0 & (x,y) \in [0,1] * [0,1] \\ \nabla q * \mathbf{n} = \mathbf{0} & (x,y) \in \partial[0,1] * [0,1] \quad t > 0; \end{cases} \tag{1}$$

So we have a 2D heat equation, with a zero initial value, with no flux on the boundaries, and with a source.

For this last point, we will be considering two different sources, both centered on the middle of the mesh :

— S1(x,y) $= exp^{-\frac{((x-x_s)^2+(y-y_s)^2)}{\omega^2}}$

— S2(x,y,t) $= \delta(x - x_s, y - y_s) * g(t)$, with g(t) a step function that we will treat later

We will throughout this report see the different methods computed and the results coming with it when it comes to the numerical resolution of the PDE of heat.

2 Analytical preamble

In this part, we begin by a study aiming to be a recall of the basics of the course and a general understanding of the important calculations to use in heat equation resolution.

2.1

The flux vector, noted F_{ij}, for the heat equation, noted q_{ij}, is defined as follows :

$$F_{ij} = -\nabla q_{ij} \tag{2}$$

The flux vector is of the most importance when it comes to discretiation as it will allow us to both get back to an expression over q, and thus apply the boundary conditions, and to get the fully discretize problem.

We will thus use this equality at a few moments throughout this assignment.

2.2

We are in this question defining the function Q(t) as equal to the integral of the heat equation, a relation which is summarize in the following equation :

$$Q(t) = \int_0^1 \int_0^1 g(x,y,t)dxdy \tag{3}$$

We note that the integrals over the space are between 0 and 1 because we decide to get the mean heat on an array of 1-by-1. This integration will ensue in the fact that the result will only be time-dependent. We thus give the function Q(t) as a function depending only on time, and we will use the equation that is to be solved (eq 1) and the expression of the flux vector as declared on the previous response (eq 2).

In fact, as we have on the first equation indications on the derivatives of the function q and that we have the condition of a zero flux on the boundaries, we will rather pass from the equation 3 to an equation where appear the element we just listed. We proceed on the following manner :

$$\frac{dQ}{dt} = \frac{d}{dt}\int_0^1\int_0^1 g(x,y,t)dxdy \longrightarrow \int_0^1\int_0^1 \partial_t g(x,y,t)dxdy$$
$$\longrightarrow \int_0^1\int_0^1 \Delta g(x,y,t)dxdy + \int_0^1\int_0^1 S(x,y,t)dxdy$$

Using the divergence's theorem and the boundary condition stipulating no flux across the boundaries, we obtain the following equality :

$$\frac{dQ}{dt} = \int_0^1\int_0^1 \Delta g(x,y,t)dxdy + \int_0^1\int_0^1 S(x,y,t)dxdy$$

$$\longrightarrow -\int_\Omega \nabla q(x,y,t) * \mathbf{n} d\nu + \int_0^1\int_0^1 S(x,y,t)dxdy$$

Yet, the integral over the mesh of $\nabla q(x,y,t) * \mathbf{n}$ is equal to the integral of zero, which is equal to zero. We end up with this relation :

$$\frac{dQ}{dt} = \int_0^1\int_0^1 S(x,y,t)dxdy$$

By integrating on both side over the time, we get an expression of Q as a function depending only on time :

$Q(t) = t * \int_0^1\int_0^1 S(x,y,t)dxdy + Q(0)$

Yet, $Q(t) = \int_0^1\int_0^1 g(x,y,t)dxdy$, so

$Q(0) = \int_0^1\int_0^1 g(x,y,0)dxdy = \int_0^1\int_0^1 0dxdy = 0$

We finally get an expression for Q, which is depending only on the time as, by integration, the x and y components will appear with finite value :

$$Q(t) = t * \int_0^1\int_0^1 S(x,y,t)dxdy$$

Now that we obtained an other form to define the function Q(t), we will make the calculation of the integral above, considering first the source S1 and then the source S2 :

— S1(x,y) $= exp^{-\frac{((x-x_s)^2+(y-y_s)^2)}{\omega^2}}$, with $x_s = 1/2$ and $y_s = 1/2$, $\omega = 0.2$.

As (x,y) $\in [0,1] * [0,1]$, we have indeed a source on the middle of our mesh, even if this source isn't zero on the other considered points.

To calculate the integral, we will go through several variable changes and integral "redefinition". First, we try to simplify the function S1(x,y) :
With X $= \frac{x-x_s}{\omega}$ and Y $= \frac{y-y_s}{\omega}$, we have :

$$\text{S1(X,Y)} = exp^{-(X^2+Y^2)}$$

Nevertheless, when doing the integration we will have to reconsider the range over which we are integrating. Thus, with (x,y) $\in [0,1] * [0,1]$, we have X $\in [-\frac{x_s}{\omega}, \frac{1-x_s}{\omega}]$ and Y $\in [\frac{-y_s}{\omega}, \frac{1-y_s}{\omega}]$.
Before making any more calculation, we will make an other variable change in order to make appear the derivative of our exponential : we thus put X = rcos(θ) and Y = rsin(θ).

Once again, we redefine our range for the integrals : as $-\frac{x_s}{\omega} < X < \frac{1-x_s}{\omega}$ and $\frac{-y_s}{\omega} < Y < \frac{1-y_s}{\omega}$, with r $= \sqrt{X^2+Y^2}$ we obtain :

$$\sqrt{\frac{x_s^2+y_s^2}{\omega^2}} < r < \sqrt{\frac{1-x_s}{\omega}^2 + \frac{1-y_s}{\omega}^2}$$

We ultimately get this form for the integral of S1(x,y) over the space :

$$Q(t) = t * \int_{\sqrt{\frac{x_s^2+y_s^2}{\omega^2}}}^{\sqrt{\frac{1-x_s}{\omega}^2+\frac{1-y_s}{\omega}^2}} \int_0^{2\pi} r * exp^{-r^2} d\theta dr$$

To conclude, we have an expression where we made appear a function and her derivative. By a simple integration and because we know x_s, y_s and ω, we obtain :

$$Q1(t) = \pi * (exp^{-2.5^2} - exp^{-4*2.5^4}) = 6.10^{-3} * t \quad (4)$$

We have a function that represents the mean temperature over a defined square mesh, and that's increasing over time. We indeed have a continuous source on the center that is spreading over time to every position of the mesh. We will display the behavior of the source as well as the one of S2 later on this document.

— S2(x,y,t) $= \delta(x - x_s, y - y_s) * g(t)$. We have also have g(t) defined as follows :

$$g(t) = \begin{cases} 2, & if \quad t < 1/4 \\ 0, & if \quad t \geq 1/4 \end{cases} \tag{5}$$

This source can be summarize by saying that it is a source fixed on (x_s,y_s) as the Dirac function is only different from zero on these coordinates. Moreover, we here have a step function g(t) indicating that the source is only active from t = 0s to t = 0.25s.

If we reconsider the integral over space of S2(x,y,t), we easily obtain the following results :

$$Q_2(t) = t * \int_0^1 \int_0^1 S2(x, y, t)dxdy$$

$\longrightarrow$ Q$_2(t) = t * \int_0^1 \int_0^1 \delta(x - x_s, y - y_s) * g(t)dxdy = t * g(t)dxdy,$ for (x,y) = (x_s,y_s)

$$Q_2(t) = \begin{cases} 2 * t, & if \quad t < 1/4 \\ 0, & if \quad t \geq 1/4 \end{cases} \tag{6}$$

We can conclude that with this source the heat is increasing a llot faster, but only during a considered range of time.

3 Discretization and implementation

In this part we will make the link between the physical problem of heat and the ways to study it through discretization, and generally speaking, numerical approaches.

3.1

We recall the PDE that is to be numerically studied and solved :

$$q_t - \Delta q = S \tag{7}$$

With the idea of discretization comes the idea that we will approximate our initial function by simplifying some of its elements, which will lead in some errors.

In this case, we will approximate each element above by a numerical approximated function which is the mean value of the element over the mesh space. We thus have the following equality :

$$\frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} \partial_t q(x, y, t)dxdy = \frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} \Delta q(x, y, t)dxdy + \frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} S(x, y, t)dxdy \tag{8}$$

We get integrals over a mesh which is here a cell called C_{ij} and which is a cell of 1-by-1. This expression of the PDE will lead us into declaring function being approximation of those current equations.

Here, we mainly base the method using the course SF2521. We won't make the whole process to not waste place into mathematic methods but the main idea is this one :

— $\frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} \partial_t q(x,y,t) dxdy = \frac{1}{\Delta_x \Delta_y} \frac{d}{dt} \int_{C_{ij}} q(x,y,t) dxdy$, as the integration isn't over the time variable.

$\longrightarrow \frac{1}{\Delta_x \Delta_y} \frac{d}{dt} \int_{C_{ij}}$ q(x,y,t)dxdy $= \frac{d}{dt} Q_{ij}$

— $\frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} S(x,y,t) dxdy = S_{ij}$

Using the heat flux equation (2), a physical approach of the flux across the boundary (flux direction, flux sens, etc) and the cell discretization discussed above, we have this last approximation :

$$\frac{1}{\Delta_x \Delta_y} \int_{C_{ij}} \Delta q(x,y,t) dxdy = \Delta_5 Q_{ij} \tag{9}$$

with $\Delta_5 Q_{ij} = Q_{i+1,j} + Q_{i-1,j} + Q_{i,j+1} + Q_{i,j-1} - 4Q_{i,j}$ the five point Laplacian stencil from finite difference method.

We eventually obtain the equality asked :

$$\frac{d}{dt} Q_{ij} = \Delta_5 Q_{ij} + S_{ij} \tag{10}$$

Furthermore, even if we are in 2D, the stencils on the edges are defined using the finite difference of first order, because we have on the boundary a condition over the flux, which we recall is defined by :

$$\nabla q * \mathbf{n} = \mathbf{0} (x,y) \in \partial[0,1] * [0,1] t > 0$$

Hence, the condition upper can be decompose into two sub-conditions which are :

$$\partial_x Q_x(x,y,t) = \frac{Q_{i,j} - Q_{i-1,j}}{\Delta_x} = 0 \text{ and } \partial_y Q_y(x,y,t) = \frac{Q_{i,j} - Q_{i,j-1}}{\Delta_y} = 0.$$

These relation, on the boundary (considered at (i,j) $\in [1,1] * [N,N]$) give the stencils at the boundaries :

— $\frac{Q_{1,j} - Q_{0,j}}{\Delta_x} = 0 \longrightarrow$ Q$_{1,j} = Q_{0,j}$ and $\frac{Q_{N,j} - Q_{N-1,j}}{\Delta_x} = 0 \longrightarrow$ Q$_{N,j} = Q_{N-1,j}$

— $\frac{Q_{i,1} - Q_{i,0}}{\Delta_x} = 0 \longrightarrow$ Q$_{i,1} = Q_{i,0}$ and $\frac{Q_{i,N} - Q_{i,N-1}}{\Delta_x} = 0 \longrightarrow$ Q$_{i,N} = Q_{i,N-1}$

To sum-up, the flux is zero throughout the entire boundary and we will now see how it translates to matrix form.

3.2

We will now discretize our equation again, using the first order implicit Euler scheme. We will thus linearize the partial element along the time to get the fully discrete problem :

$$\frac{d}{dt}Q(x,y,t) = \frac{Q^{n+1} - Q^n}{\Delta_t} = f(Q^{n+1}, t^{n+1}) \tag{11}$$

By implementing this scheme into the equation (10), we get the following fully discrete problem :

$$Q^{n+1} = (\Delta_5 Q^{n+1} + S^{n+1}) * \Delta_t + Q^n \tag{12}$$

To use this implicit scheme has several positives aspects. The main one is that the heat solution solving our system will be unconditionally stable. Thus, there is no condition over the time-step to get a stable result. Though, the precision of this solution is still time-dependant.

3.3

The last step to go from discretization to numerical analysis will be to get a linear equation and to expilictly express each member of equation (12). Thus, we introduce a new expression for the Laplacian $\Delta_5 Q(x,y,t)$:

$$\Delta_5 Q = T_x.Q + Q.T_y$$

From the definition of T, which represent the second derivative difference operator in 1D, we get an other expression for Tx.Q, Q.Ty and thus $\Delta_5 Q$:

— $Tx.Q = \frac{Q^{n+1}_{i+1,j} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i-1,j}}{\Delta_x^2}$

— $Q.Ty = \frac{Q^{n+1}_{i,j+1} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i,j-1}}{\Delta_y^2}$

$\longrightarrow \Delta_5 Q = \frac{Q^{n+1}_{i+1,j} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i-1,j}}{\Delta_x^2} + \frac{Q^{n+1}_{i,j+1} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i,j-1}}{\Delta_y^2}$

Finally, we include this new expression in the equation (12), and we get the fully discrete problem in matrix form :

$$(I - \Delta_t(\frac{D.Q}{\Delta_x^2} + \frac{(D.Q^T)^T}{\Delta_y^2}))Q^{n+1} = Q^n + \Delta_t.S^{n+1} \tag{13}$$

With D the matrix associated to the second derivative difference operator, defined by :

$$D = \begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -1 \end{pmatrix} \tag{14}$$

The values in (1,1) and (M,N) come from the boundary condition defined in the part 3.1. For instance, if we consider the point (1,1) :

- a) $T_x.Q(1,1) = \frac{Q_{1+1,1} - 2*Q_{1,1} + Q_{0,1}}{\Delta_x}$
- b) $Q_{1,1} = Q_{0,1}$ because of the boundary condition

$\longrightarrow$T$_x.Q(1,1) = \frac{Q_{2,1} \;\; \textbf{-1}*Q_{1,1}}{\Delta_x}$

3.4

Before solving numerically the equation (13), we discretize the second source S2 :
$S2(x,y,t) = \delta(x - x_s, y - y_s) * g(t) \longrightarrow S2(r,t) =\delta_e(r) * g(t)$

with $\delta_e(r) = \frac{\pi}{\epsilon^2(\pi^2-4)}(1 + cos(\frac{\pi}{\epsilon}r)$ if r $<\epsilon$ and 0 otherwise ;
and with $r = \sqrt{(x - x_s)^2 + (y - y_s)^2}$ and $\epsilon = \sqrt{max(\Delta x, \Delta y)}$

3.5

This part consist in the Matlab implementation of the equation discussed and detailed above (eq. (13)).

This part is the source of many of our own researches and the advice from Johan Wärnegård.

The idea was to retranscribe what we have been talking about throughout the previous question. That is to say that we will make the implementation using mostly the equations and expressions (13) and (14). The code won't be more explained here as it is simply the heat equation's resolution as defined upper.

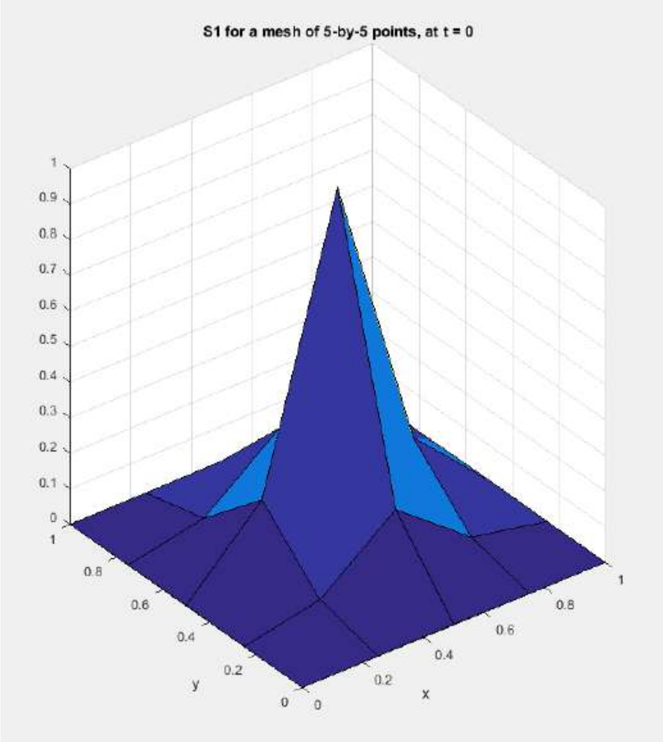
For any other information, we encourage you to take a look at the code as this part is fully described in it.

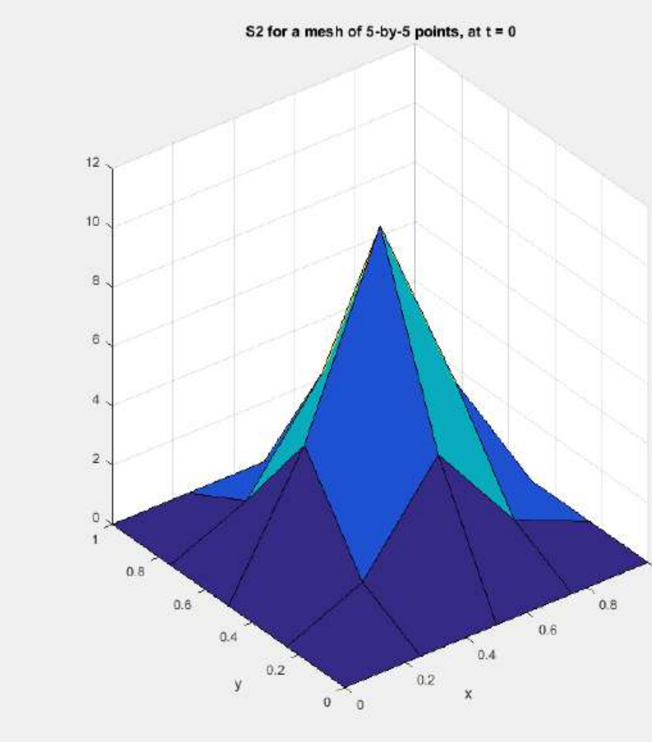
4 Numerical results

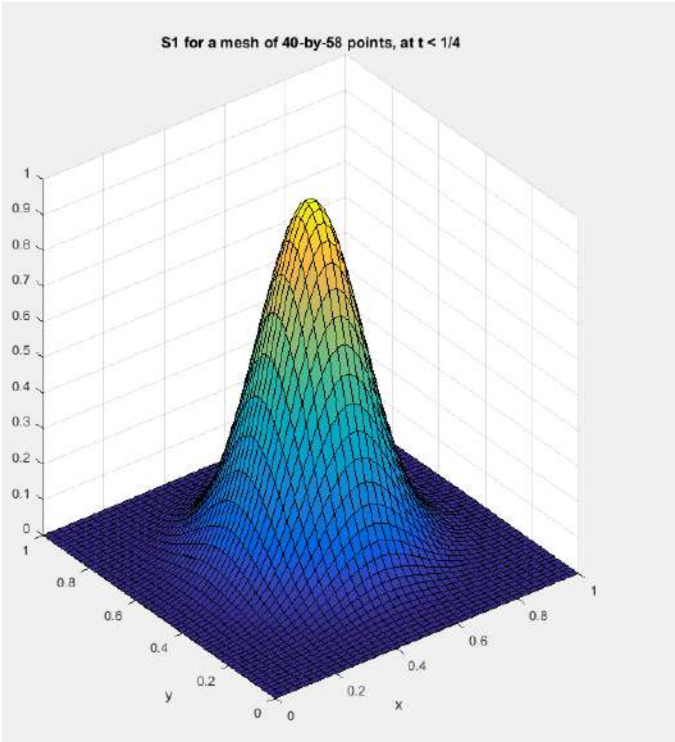
4.1

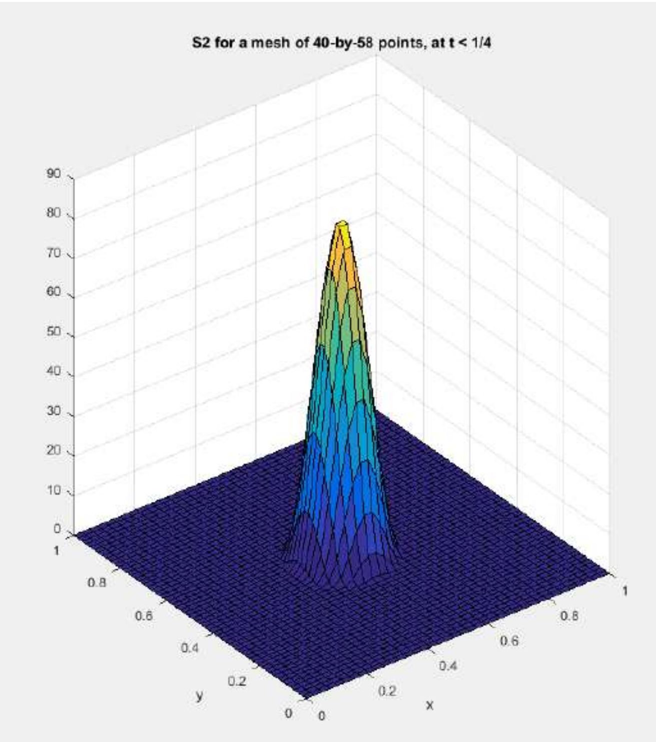
We are now willing to show the program working for both sources and at different time levels.

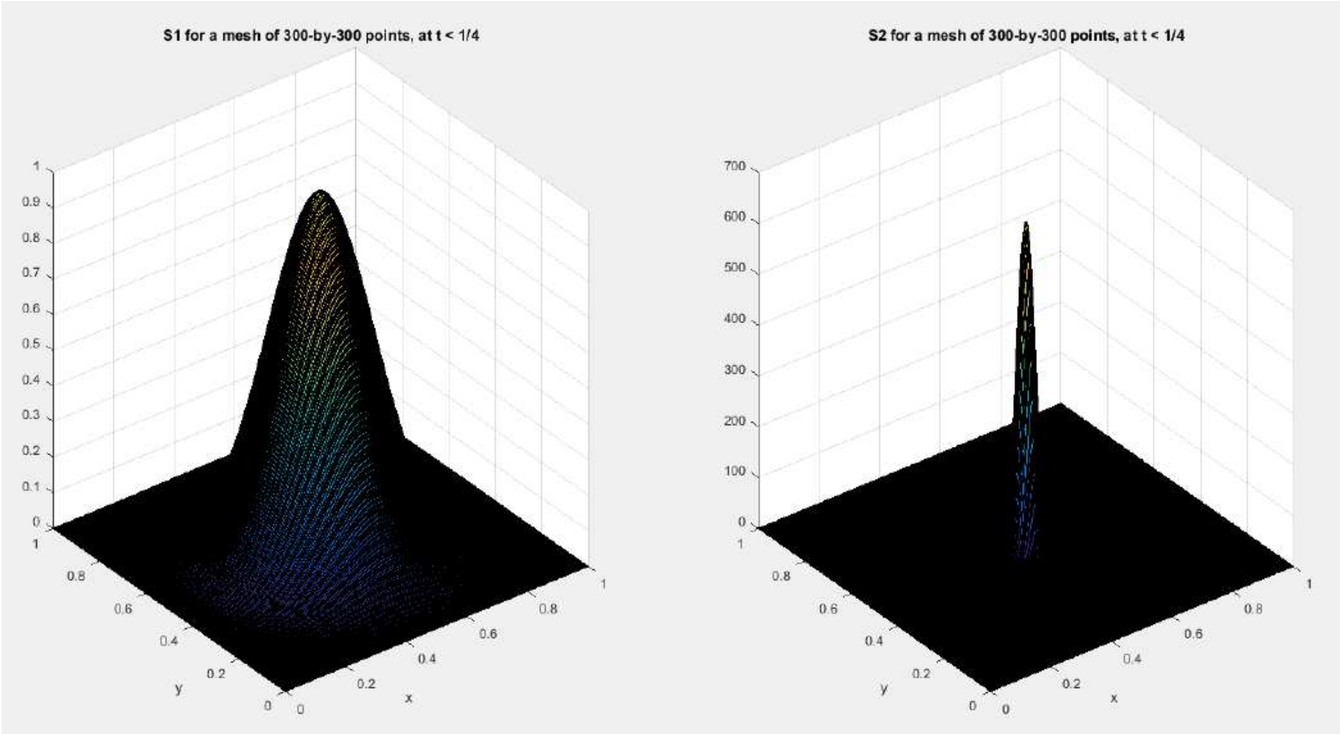
First, here is a sight of both sources, considering different length for the mesh :


S1 for a mesh of 5-by-5 points, at t = 0
y
x


S2 for a mesh of 5-by-5 points, at t = 0
y
x


S1 for a mesh of 40-by-58 points, at t < 1/4
y
x


S2 for a mesh of 40-by-58 points, at t < 1/4
y
x

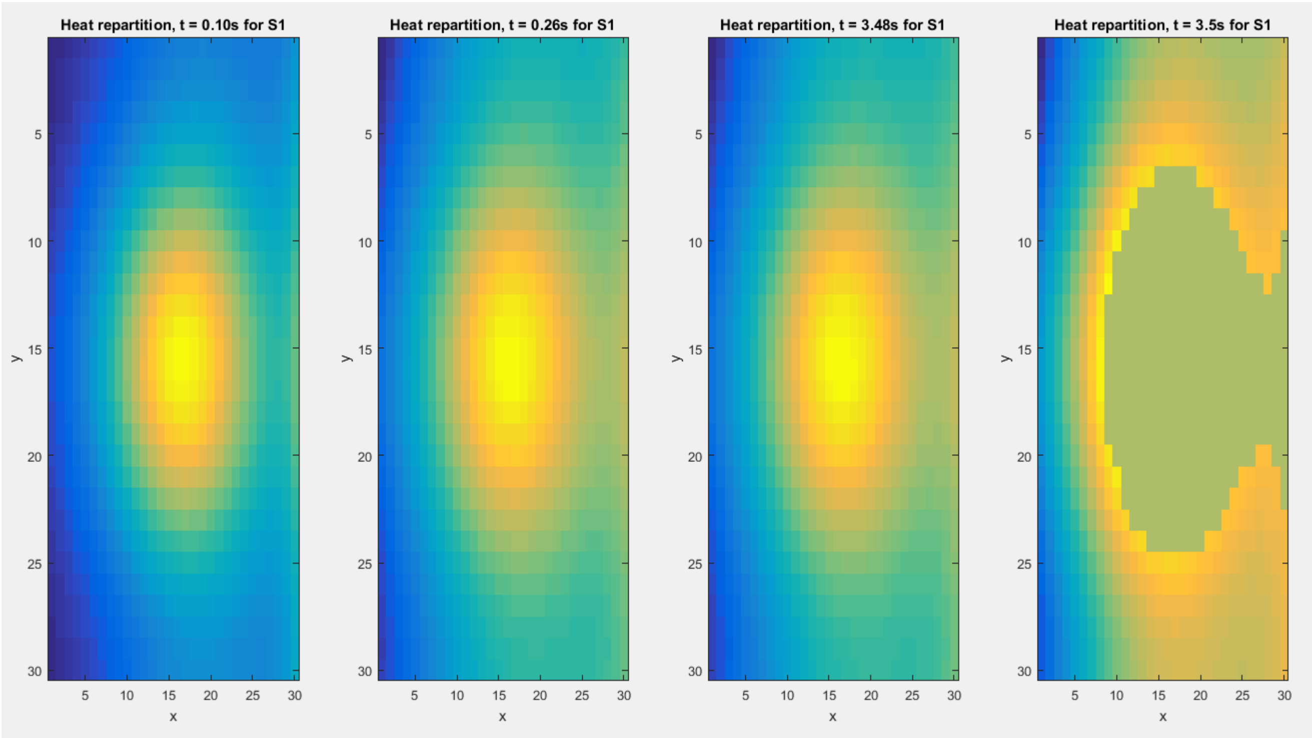


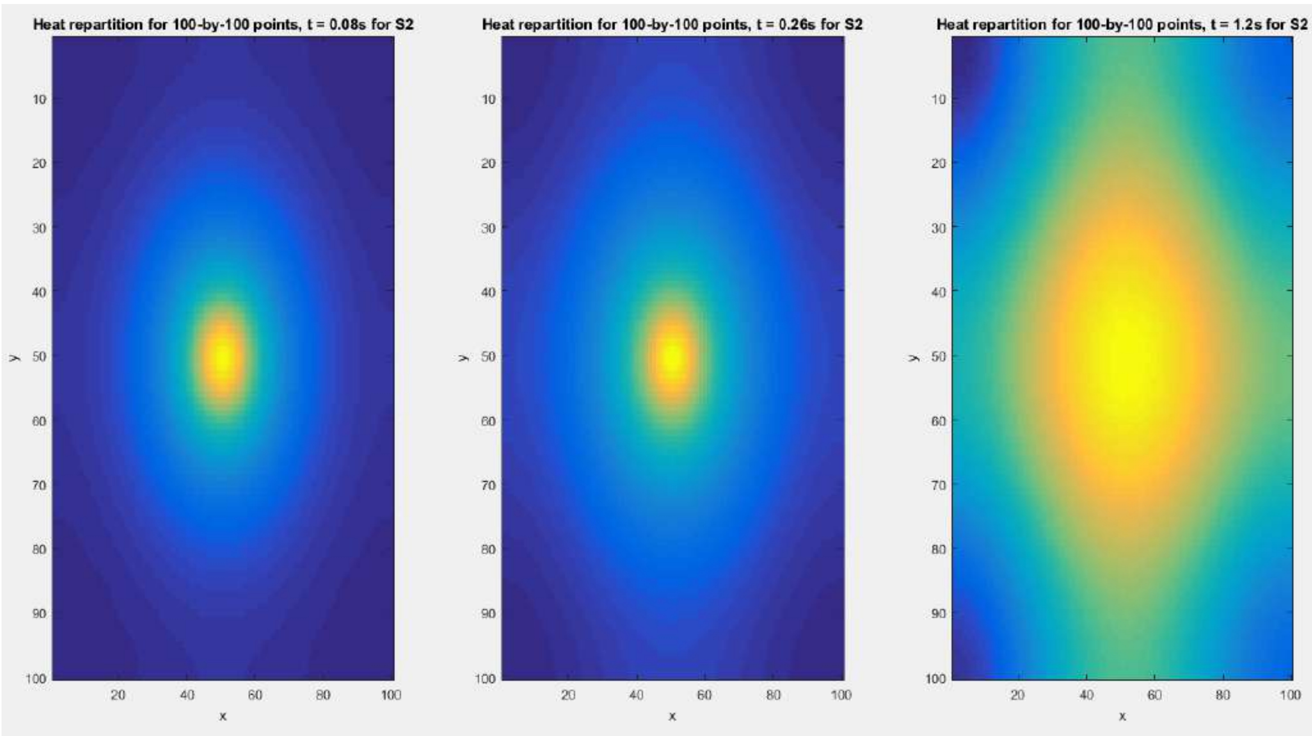
We can see that for the source S1, the maximum of the function is the one of the length of the grid considered (1 here) and that the source isn't different from zero just at enter. The source will diffuse its heat to the surrounding particle, and we will here get a final state where the heat is the same in every point of the mesh, assuming we wait enough for the system to get to the heat equilibrium. Indeed, even if we have a continuous source on the middle, a physical approach push us to consider that if the heat source isn't increasing, the heat on the rest of the source won't increase neither.

For instance, to sum 60°C with 60°C leads to 60°C in our code and obviously not to 120°C.

When considering the source S2, more points are considered, more the function acts as the dirac function. In this case, after 0.25s, we should have a source that stops and thus the heat should be the same along the entire mesh after a certain time, and the final heat will be depending on the number of points and the time considered.

We will now take a look at the behavior of the heat equation to see if it behaves as assumed :

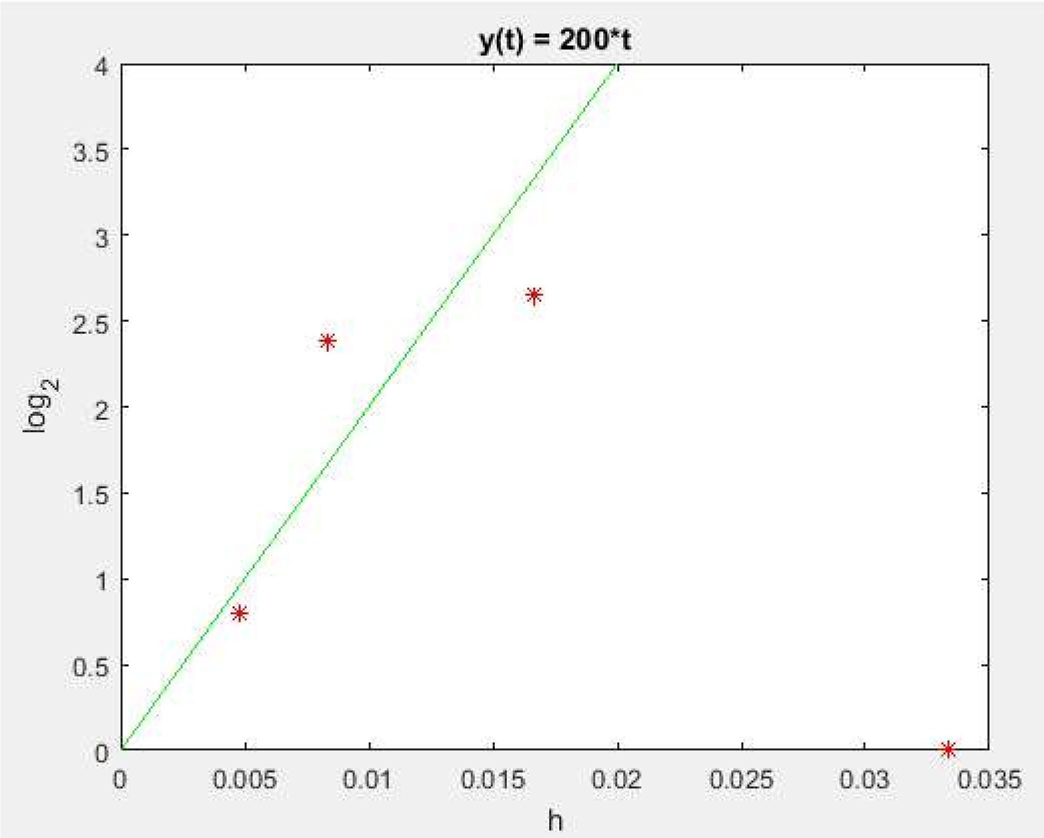

Heat repartition, t = 0.10s for S1
Heat repartition, t = 0.26s for S1
Heat repartition, t = 3.48s for S1
Heat repartition, t = 3.5s for S1
x
y


Heat repartition for 100-by-100 points, t = 0.08s for S2
Heat repartition for 100-by-100 points, t = 0.26s for S2
Heat repartition for 100-by-100 points, t = 1.2s for S2
x
y

Note : the color tending to red/orange doesn't imply that the heat is increasing.

For instance, in the graphics gathering information at different time-step for the source 2 (fig 2), the third graphic show that the heat is spreading to the entire mesh and that almost all of the mesh is at the maximum temperature.

We plot the error for a fixed time-step, for several space delta, for S1. We then get that, for a fixed time, we have a slope of 200 for h. Thus, we might have r = 200 :



This result looks to be an error since the real value we need to find, the one from the theoretical case, is 2. Indeed, we have, writting the previous equations by making appearing the errors due to discretization, the following errors :

$$Q(x+h) = Q(x) + h * Q'(x) + \frac{h^2}{2} * Q''(x) + o(h^2)$$

$$Q(x-h) = Q(x) - h * Q'(x) + \frac{h^2}{2} * Q''(x) + o(h^2)$$

$$\longrightarrow \text{Q(x+h) + Q(x-h) = Q(x) + h}^2 * Q''(x) + o(h^2)$$

$$\longrightarrow \text{Tx.Q = Q(x+h) - 2*Q(x) + Q(x-h) + o(h}^2\text{)}$$

Using the derivative difference operator, we also find that the error associated to Δ_t has to be 1, so p = 1.

The error on these results probably comes from a wrong application of the method "Verifying Numerical Convergence Rates". We see that we have a factor 100 that makes our slope not being the one predicted, but we couldn't figure it out where the error i coming from.

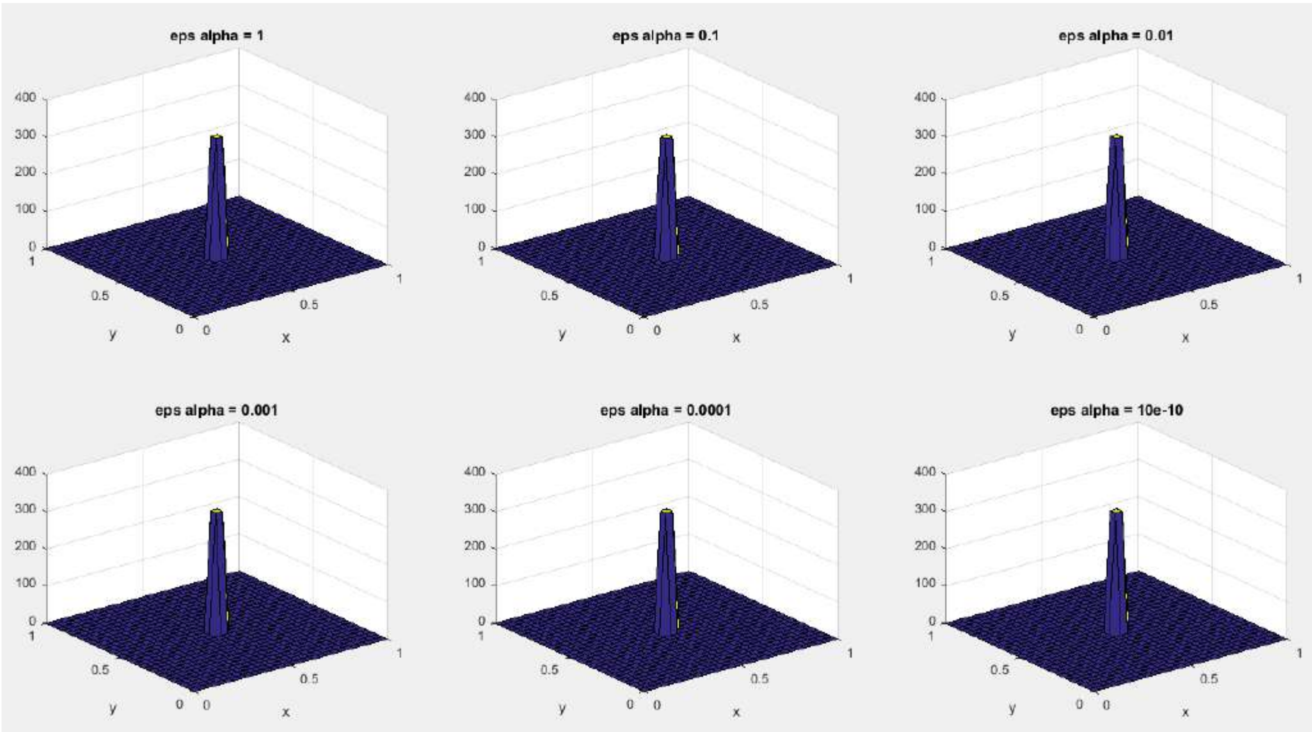
Nevertheless, we can still stipulate on the asked question as we did the theoretical analysis and that we have now enough hindsight to get an idea of the behavior of our function, depending on the source.

For instance, we do know that we can't apply the L_2-norm because it is not relevant when looking to the matrices conditioning. We might then encounter long calculus and non-convergent solution when also working on minus infinite.

We now on want to observe the influence of ϵ over the dirac approximation.

To do so, we compute our program for several $\epsilon = \alpha * \sqrt{h}$ with α = (1, 0.1, 0.01, 0.001, 0.0001, and 10e-10).

We then show the drawings obtained for the source 2 and we report here the results for any t, with a mesh of 30-by-30 :



We observe that the source acts as the dirac function when we are considering very low value for epsilon but without increasing its value. Thereby, while we could have wanted to improve the method precision by getting epsilon as little as possible, we see here that the numerical method used is limited. Thus, using this discretized function, we can only reach a certain precision, that doesn't increase with epsilon decreasing.

4.2

To finish the third part, we are gonna check numerically that the method is conservative. To do so, we will try to show the following equality for $0 < t < 2$:

$$\int q dx dy = \Delta x \Delta y \sum Q_{ij} \tag{15}$$

By a simple calculation using Matlab, we have, for t = 4s (200 iterations), and a mesh of 30-by-30 :

— $\int q dx dy = \frac{Q(N,N)+Q(1,1)}{2} = 0.4487$

— $\Delta x \Delta y \sum Q_{ij} = 0.4487$

You can see this result looking to the Matlab code, as well as all the other plot showed for now.
For the error between the two results, and which is approximately of 10 percent, we conclued that it is coming from our delta x and y that are maybe to high. In fact, we decided to keep the same Δ_x and Δ_y on the entire assignment rather than comparing method and numerical approximation that are of a different precision.

4.2.1 (b)

The space discretization has made the object of an entire part on the Matlab code to define the coordinates of a point. We were then taking the example of the point (x,y) = (1/4, 1/4) and the objective was, depending on the size of the mesh, to define where was located this point.

More specifically, our objective there was to define the heat considering the said point.

Even if the method for the point (x,y) = (1/4,1/4) is a bit more complex, the functioning is the same as for the point (1/2,1/2).

We firmly encourage you to take a look to the code in order to get an idea of the numerical approach beyond this.

5 Refinements

This part aims at developing deeper our analysis of the heat transfer function.

5.1 Variable coefficients

We consider from now on that the equation that is to be solve is the following one :

$$q_t = a(y)q_{xx} + b(x)q_{yy} + S \tag{16}$$

In order to keep a well posed equation, we have to get a(y) > 0 $\forall y \in [0,1] and b(x) \forall x \in [0,1]$.

We choose arbitrarly a(y) = y^2 and b(x) = x^2.

5.1.1

We are not developping anymore the method to get to the fully discrete problem. We will only show the raw step to get to it in order to get directly to the numeria approach part :

$$q_t = y^2 * q_{xx} + x^2 * q_{yy} + S$$

$$\frac{d}{dt}Q = y^2 * q_{xx} + x^2 * q_y + S$$

$$\frac{dQ}{dt} = y^2 * \frac{(Q^{n+1}_{i+1,j} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i-1,j})}{\Delta_x^2} + x^2 * \frac{(Q^{n+1}_{i,j+1} - 2*Q^{n+1}_{i,j} + Q^{n+1}_{i,j-1})}{\Delta_y^2} + S$$

In matrix form, we use once again the Kronecker product and we then obtain :

$$\frac{Q^{n+1} - Q^n}{\Delta_t} = (Y.D.Q + X.D.Q)^{n+1} + S^{n+1} \tag{17}$$

$$\frac{Q^{n+1} - Q^n}{\Delta_t} = (Y.(D \otimes I) + X.(I \otimes D))Q^{n+1} + S^{n+1} \tag{18}$$

If we think in term of the size of each matrix, we have :

— $dimension(D \otimes I) = [M, M] \otimes [N, N] = [M * N, M * N]$

— $dimension(I \otimes D) = [M, M] \otimes [N, N] = [M * N, M * N]$

— $\longrightarrow Y.(D \otimes I) \Rightarrow dim(Y) = [1, M * N]$. We have 1 line because we have value only for the number of point in the mesh, which is M*N. This is also why we have M*N column : to scale our vector Y for the entire mesh. Furthermore, we have mandatory the same number of element as

— $\longrightarrow X.(I \otimes D) \Rightarrow dim(X) = [1, M * N]$. We have 1 line for the same reasons.

We are unfortunately missing time to achieve the total resolution of the assignement, and thus we have no more results to show.

6 Conclusion

We have been able, throughout this document, to make the link between the mathematical expression of the heat flux and the numerical ways to solve it.

We have now a better knowledge of the PDE's solving methods and more precisely of the numerical discretization to study and solve the parabolic equation, considering smooth and non-smooth sources.