INVERSE PROBLEM FOR LIGHT SPREADING APPLIED TO MEDICAL TOMOGRAPHY

Project Report Version 0 : Guidelines reformulation

Desmond Roussel Ngueguin

Under de supervision of

Emmanuel Franck, Laurent Navoret, and Vincent Vigon

I. OBJECTIVES

This project aims to find the optical properties of an organ by solving an inverse problem. We send an infrared light beam through the organ and we measure the signal on part of said organ. Knowing the initial conditions and the signal at all times, we can infer the organ's properties. Namely the density, the scattering opacity, and the absorption opacity. As usually done in medical imagery, we can then tell if the organ if tumorous or not.

The project is divided into two equally important parts. First, we must solve the partial differential equation (PDE) for a simplified version of the **radiative transfer equation** (1) in 1D. This is done numerically, knowing the optical properties of some tissues or organs the program will be tested on. The second part of the project will cover machine learning. Knowing the signal's value on the boundaries of our domain at all times, the machine learning algorithm will predict the properties of said domain.

$$\begin{cases} \partial_t E + c\partial_x F = c\sigma_a(aT^4 - E) \\ \partial_t F + \frac{c}{3}\partial_x E = -c\sigma_c F \\ \rho C_v \partial_t T = c\sigma_a(E - aT^4) \end{cases} \tag{1}$$

Where:

- a is the Stefan-Boltzmann constant
- C_v is the thermal capacity of the medium
- c is the speed of light
- $T(t,x) > 0$ is the temperature of the medium
- $E(t,x) \in \mathbb{R}$ is the energy of the photons
- $F(t,x) \in \mathbb{R}$ is the flux of the photons
- $\rho(x) > 0$ is the density of the medium
- $\sigma_a(\rho,T) > 0$ is the absorption opacity
- $\sigma_c(\rho,T) > 0$ is the scattering opacity

We should note that when the scattering opacity σ_c and the absorption opacity σ_a are high enough to be close to the speed of light c (which is far greater than the observed phenomenon's speed), the model in (1) is reduced to the following:

$$\partial_t(aT^4 + \rho C_v T) - \partial_x\left(\frac{c}{3\sigma_c}\partial_x aT^4\right) = O\left(\frac{1}{c}\right) \tag{2}$$

This model is called the **diffusion approximation** (Franck, 2020, p.7)**.** It matches a diffusion equation with a diffusion coefficient of $\frac{c}{3\sigma_c} \approx O(1)$. But more importantly, it gives us a good way of testing our program.

II. ROADMAP AND DEADLINES

This section states the tasks that have already been completed and gives the planning for the remaining ones.

1. Phase 1: Solving the PDE with Finite Volumes

Due to the hyperbolic nature of the PDE (1), the constraints of our domain, and the need to deal with discontinuities, the **finite volumes** method is well suited to solving it. However, the well-known Rusanov scheme is not appropriate, especially when the opacities are very high. Therefore, we will use a semi-implicit Euler Scheme by "splitting".

First, we must discretize our domain to form a mesh. Let a and b be the two real numbers such that $b > a$, and $N > 0$ an integer. We split the domain $[a, b]$ into N cells of equal length to obtain a uniform mesh. At the two edges, we add two "ghost" cells. In total, we have $N + 2$ cells. Let's denote by Δx the length of the intervals (the volume of the cells). For each cell $j \in [\![0, N + 1]\!]$, we write x_j its center, $x_{j-\frac{1}{2}}$ its left edge, and $x_{j+\frac{1}{2}}$ its right edge.

Next, we write the splitting scheme. Two steps are required here.

a. Step 1: The equilibrium part

We consider the "equilibrium" part of (1). That is, the photons are not moving, and we only take into account the relaxation terms in temperature (Franck, 2012, p. 160). This leads to all the terms with ∂_x in (1) to be equal to 0. The equation becomes:

$$\begin{cases} \partial_t E = c\sigma_a(aT^4 - E) \\ \rho C_v \partial_t T = c\sigma_a(E - aT^4) \end{cases} \tag{3}$$

Writing $\Theta = aT^4$, we solve (3) on each independent cell. The numerical scheme is given below.

$$\begin{cases} \dfrac{E_j^{q+1} - E_j^n}{\Delta t} = c\sigma_a\left(\Theta_j^{q+1} - E_j^{q+1}\right) \\ \dfrac{F_j^{q+1} - F_j^n}{\Delta t} = 0 \\ \rho_j C_v \mu_q \dfrac{\Theta_j^{q+1} - \Theta_j^n}{\Delta t} = c\sigma_a(E_j^{q+1} - \Theta_j^{q+1}) \end{cases}$$

Rewritten as:

$$\begin{cases} E_j^{q+1} = \alpha E_j^n + \beta \Theta_j^{q+1} \\ F_j^{q+1} = F_j^n \\ \Theta_j^{q+1} = \gamma \Theta_j^n + \delta E_j^{q+1} \end{cases}$$

Applying Cramer's rule, we get:

$$\begin{cases} E_j^{q+1} = \dfrac{\alpha E_j^n + \gamma \Theta_j^n}{1-\alpha\beta} \\ F_j^{q+1} = F_j^n \\ \Theta_j^{q+1} = \dfrac{\gamma\Theta_j^n + \alpha\delta E_j^n}{1-\alpha\beta} \end{cases} \tag{4}$$

Where

- E_j^n, F_j^n and Θ_j^n are the value values of E_j, F_j and Θ_j on the cell at the beginning of the step.
- $\alpha = \frac{1}{\Delta t}\left(\frac{1}{\Delta t} + c\sigma_a\right)^{-1}$, $\beta = c\sigma_a\left(\frac{1}{\Delta t} + c\sigma_a\right)^{-1}$, $\gamma = \frac{\rho_j C_v \mu_q}{\Delta t}\left(\frac{\rho_j C_v \mu_q}{\Delta t} + c\sigma_a\right)^{-1}$, and $\delta = c\sigma_a\left(\frac{\rho_j C_v \mu_q}{\Delta t} + c\sigma_a\right)^{-1}$
- σ_a written above is a function of ρ_j and T_j^n. Thus, it is actually $\sigma_a(\rho_j, T_j^n)$.
- μ_q is such that $\mu_q = \dfrac{1}{T^{3,n} + T^n T^{2,q} + T^q T^{2,n} + T^{3,q}}$

Since this step is a fixed point method, we iterate on q (on each cell j independently) until we get close enough to the fixed point $(E_j^*, F_j^*, \Theta_j^*)$, or more precisely (E_j^*, F_j^*, T_j^*). We then move to the next step.

b. <u>Step 2: Solving the rest</u>

Once the first step converges, we move to this step with the values of $E, F,$ and T on each cell updated. We write:

$$\begin{cases} \dfrac{E_j^{n+1} - E_j^*}{\Delta t} + c\dfrac{F_{j+\frac{1}{2}} - F_{j-\frac{1}{2}}}{\Delta t} = 0 \\ \dfrac{F_j^{n+1} - F_j^*}{\Delta t} + c\dfrac{E_{j+\frac{1}{2}} - E_{j-\frac{1}{2}}}{\Delta t} = cS_j \\ \rho_j C_v \mu_q \dfrac{T_j^{n+1} - T_j^*}{\Delta t} = 0 \end{cases} \tag{5}$$

With

$$F_{j+\frac{1}{2}} = M_{j+\frac{1}{2}}\left(\frac{F_{j+1}^n + F_j^n}{2} - \frac{E_{j+1}^n - E_j^n}{2}\right), \quad F_{j-\frac{1}{2}} = M_{j-\frac{1}{2}}\left(\frac{F_j^n + F_{j-1}^n}{2} - \frac{E_j^n - E_{j-1}^n}{2}\right)$$

$$E_{j+\frac{1}{2}} = M_{j+\frac{1}{2}}\left(\frac{E_{j+1}^n + E_j^n}{2} - \frac{F_{j+1}^n - F_j^n}{2}\right), \quad E_{j-\frac{1}{2}} = M_{j-\frac{1}{2}}\left(\frac{E + E_{j-1}^n}{2} - \frac{F_j^n - F_{j-1}^n}{2}\right)$$

$$S_j = -\frac{1}{2}\left(M_{j+\frac{1}{2}}\sigma_{j+\frac{1}{2}} + M_{j-\frac{1}{2}}\sigma_{j-\frac{1}{2}}\right)F_j^{n+1}$$

$$M_{j+\frac{1}{2}} = \frac{2}{2 + \Delta x \sigma_{j+\frac{1}{2}}}, \quad M_{j-\frac{1}{2}} = \frac{2}{2 + \Delta x \sigma_{j-\frac{1}{2}}}$$

$$\sigma_{j+\frac{1}{2}} = \frac{1}{2}\left(\sigma_c(\rho_j, T_j^n) + \sigma_c(\rho_{j+1}, T_{j+1}^n)\right), \quad \sigma_{j-\frac{1}{2}} = \frac{1}{2}\left(\sigma_c(\rho_{j-1}, T_{j-1}^n) + \sigma_c(\rho_j, T_j^n)\right)$$

Not to forget the CFL condition:

$$\Delta t < \frac{\Delta x}{c}$$

We can rewrite (5) as:

$$\begin{cases} E_j^{n+1} = E_j^* - \frac{c\Delta t}{\Delta x}\left(F_{j+\frac{1}{2}} - F_{j-\frac{1}{2}}\right) \\ F_j^{n+1} = \mathrm{A}F_j^* - \mathrm{B}\left(E_{j+\frac{1}{2}} - E_{j-\frac{1}{2}}\right) \\ T_j^{n+1} = T_j^* \end{cases} \quad (6)$$

With

$$\mathrm{A} = \frac{1}{\Delta t}\left(\frac{1}{\Delta x} + \frac{c}{2}\left(M_{j+\frac{1}{2}}\sigma_{j+\frac{1}{2}} + M_{j-\frac{1}{2}}\sigma_{j-\frac{1}{2}}\right)\right)^{-1} \text{ and } \quad \mathrm{B} = \frac{c}{\Delta x}\left(\frac{1}{\Delta x} + \frac{c}{2}\left(M_{j+\frac{1}{2}}\sigma_{j+\frac{1}{2}} + M_{j-\frac{1}{2}}\sigma_{j-\frac{1}{2}}\right)\right)^{-1}$$

Once all of this is done, we move back to step 1, then back to step 2 and so on until we reach a predefined time at which we want the solution.

Concerning the deadlines, it is important to mention that considerable steps towards solving the PDE (1) have already been taken. However, a good deadline estimate for the total completion of the phase is **April 15, 2020.**

2. Phase 2: Machine learning with Neural Networks

Convoluted neural networks are the machine learning choice for this phase.

a. Step 1: Finding the density

Knowing not only the signals $E(x,t)$, F, and T at all times, but also the scattering and absorption opacities ($\sigma_c(\rho,T)$ and $\sigma_a(\rho,T)$), we will rebuild the domain's density $\rho(x)$. We can see that the inputs for the neural network will be 2D tensors (indexed by t and x) while the output will be 1D (indexed only by x).

First, we will train and validate the neural network with the data obtained from solving the PDE (1) in the first phase. Second, we will test it hoping it generalizes well.

This part shall be completed before **April 29, 2020.**

b. Setp2: The absorption and scattering opacities

This step is quite similar to the previous step with the only difference that we will be trying to predict $\sigma_c(\rho,T)$ and $\sigma_a(\rho,T)$ knowing $\rho(x)$ this time. However, it requires a more complex neural network.

This part shall be completed before **May 15, 2020.**

II. RESSOURCES AND BUDGET

Considerable resources are needed to complete this project. For computing purposes, a server on Atlas is available at v100.math.unistra.fr. Extra computing resources are also allocated through Google Colab.

1. Phase 1: Solving the PDE

This part will be almost entirely coded in C++. We will need the Eigen library for vector operations. A function parser like cparse could also be useful. The output files will be exported in the CSV format for visualization with Matplotlib in Python.

2. Phase 2: Neural Network

Since Python is the language of choice for data science, we will use it during this phase. For this to work, we will have to find a way of storing all the data from our PDE simulations in their 2D shapes, all this in one single file. We will then load the data and use it to train and test the neural network using the open-source neural-network library Keras.

III. DELIVRABLES AND OUTCOME

What is expected to be delivered at the end of this project are:

- A software that can model light spreading according to the PDE (1)
- A software that can accurately predict the optical properties on a given domain
- A typewritten report by **May 15, 2020**

All these files can be found on the GitHub repository feelpp/csmi-m1-2020-moco-inverse along with instructions on how to run the software.

III. REFERENCES

- Franck, E. April 1, 2020. "*Projets de M1*". Personal notes from Emmanuel Franck summarizing the guidelines for the project.
- Franck, E. October 23, 2012. "*Construction et analyse numérique de schéma asymptotic preserving sur maillages non structurés. Application au transport linéaire et aux systèmes de Friedrichs*". Retrieved from https://tel.archives-ouvertes.fr/file/index/docid/744371/filename/theseFranckv3.pdf