Heat and moisture transfer modelling

1st year project

Centre de modélisation et de simulation de Strasbourg

Master 1 CSMI

UFR de Mathématique et d'Informatique

Université de Strasbourg

Thomas Saigre

supervised by

Vincent Chabannes, Zohra Djatouti,
Romain Hild & Christophe Prud'homme

Second semester 2019 – 2020

Acknowledgements

Before the hard work begins, I want to thank the persons who helped me achieving this project.

I thank Zohra Djatouti for all the physics elements she enlightens me, allowing me to not be too lost in all those papers ! We also spent some time trying to implement the benchmark at the beginning, to handle Feel++.

I thank Romain Hild and Vincent Chabannes for their help with the functioning and the keywords of atlas and Feel++.

I also thank Christophe Prud'homme for the technical and theoretical help he gave me, and also for programming the main part of the program `feelpp_hm_heat_moisture` we set up during the project.

In spite of the context of confinement with teleworking in parallel with the study, I found the subject very interesting, and having good results, in the end, is quite satisfying.

The application that will be developed in the future, and which this project has been the beginning will, I hope, allow to prevent energy drain in building.

Contents

1 Introduction

1.1 Context

According to the IPCC [14], the planet is getting warmer and it is due partly because of human activities. Global warming is particularly due to high energy consumption. Numerous studies have been carried out on this consumption and one main culprit has been identified: the building sector (figure 1). Moreover, the building sector alone accounts for 20% of greenhouse gas emissions in 2017 against 17% in 1990 [18].

Figure 1: Breakdown of consumption in France, adapted from [7] (the 3% correspond to agriculture)

To face the challenge which is mitigating the global warming, many measures have been taken at different scales.

It is in response to this that in 2015 the *Loi relative à la transition énergétique pour la croissance verte* [17] was adopted to commit France to an energy transition by setting long term objectives to "Better renovate buildings to save energy" :

- Renovate 500,000 homes a year from 2017 to target a 15% reduction in fuel poverty by 2020 (*article 3*).
- By 2025, all private buildings that consume too much (*geqslant*330 kWh per square meter) must have undergone an energy retrofit (*article 4*).

In addition, at the end of 2015, the Paris Agreement [6] was adopted following COP21, aiming to reduce the rise in temperature on the planet.

To achieve these objectives, the *Plan Transition Numérique dans le Bâtiment* (*PTNB*) was implemented at the beginning of 2015, and since 2017 the *Building Information Model* has been used for new buildings.

The *BIM*, francized in *bâti immobilier modélisé* but which should be translated by *Modélisation des Informations d'une Construction*, is a working method and a 3D parametric model that contains intelligent and structured data. It is the sharing of reliable information throughout the life of a building, from its construction to its demolition (see figure 2).

In particular, BIM contains the 3D representation of a building and makes it possible to analyze, control and simulate certain behaviors. The BIM is a data exchange medium and must therefore ensure numerical compatibility, regardless of the software used.

Solutions have been put in place to integrate an equivalent of BIM into older buildings that did not have BIM at the time of their construction [13].

One of the main sources of energetic leak source is due to moisture within the walls. Models have been created to simulate them in order to prevent such leaks. The project 4fastsim-ibat, initiated in the Cemosis Laboratory deals with it.

1.2 Presentation of Cemosis

Cemosis, is the technological platform in mathematics at the University of Strasbourg, which links mathematics with business and other disciplines. It was created in January 2013 and is hosted by the IRMA (Institut de Recherche de Mathématique avancée), and is based on the Modelling and Control team.

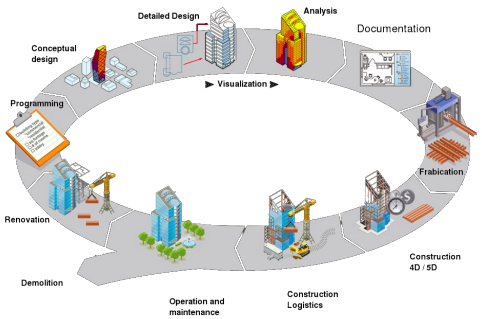


Figure 2: BIM model (from `buildipedia.com`)

1.3 4fastsim-ibat project

The aim of the 4fastsim-ibatproject [4] is to develop a fast and efficient tool combining physical models, artificial intelligence and *in-situ* measurements to identify potential energy savings in existing buildings, and to evaluate design and renovation solutions before they are implemented. The goals of this project are :

- Develop a BIM for existing buildings (here the building of the API cluster in Illkirch),
- Develop a fast 3D temperature model using artificial intelligence and physical simulations,
- Using sensors deployed as part of *IBat* (intelligent building).

It is a collaboration between CEMOSIS and Synapse Concept, a company specialized in fluid and thermal simulation and virtual and augmented reality [21]. This project will seek to integrate moisture transfer models and coupled heat and moisture transfer models to provide the user with a rapid simulation tool that allows to take into account moisture on thermal comfort and energy consumption.

In this project, we will study the moisture phenomenon in building, and try to simulate them.

1.4 Course of action

We will begin by studying the modes of heat and moisture transfer in buildings. Then, we will look at an application in the case of a building envelope, with an implementation of one or several models thanks to Feel++.

Here is the road-map of how the project shall proceed :

Heat and moisture transfer modelling

Thomas Saigre

Main objective : Implement model of heat and moisture transfer with Feel++

Asana : `https://app.asana.com/0/1164704746511025/list`

Unfolding (with approximate time / redaction time) :

- **March :** beginning of the project
- Handling Feel++ (10 h with the EDP course / 2 h)
- **Beginning of April :** Benchmark boundary layer. This benchmark has not been implemented using Feel++ (10 h / 2 h)
- **Modelisation heat transfer :** with documentation, model heat moisture transfer (4 h / 2 h)
- **14 / 04 :** Report V0
- **End of April :** Modelisation heat moisture coupling (10 h / 4 h)
- **Resolution strategies :** variational formulation, discretisation selection (both in time and space), solver selection (5 h / 3 h)
- **May :** implementation of the model, and validation with the benchmarks (15 h / 3 h)
- **22 / 05 :** Report V1
- Last verification
- Redaction of the slides (done the most possible during the previous steps 7 h)
- **27 / 05 :** Final version of the report
- **29 / 05 :** Defense

Resources :

- Docker environment in Visual Studio Code
- Machine Atlas and cemosis.feelpp to run Feel++ toolboxes

2 Feel++

2.1 Presentation

Feel++[9] (for *Finite Element Embedded Library in C++*) is an implementation of Galerkin's 1D, 2D or 3D methods for solving partial differential equations. The project was initiated in 2006 by Christophe Prud'homme. It is developed by the Cemosis laboratory and uses MPI for parallel computing. Moreover, it allows to export results in order to visualize them with Paraview, and calculate measures such as the norm of the results.

Here we will mainly use *toolboxes*, which allow expanding conventional models for one type of physics into a multi-physics model that solves coupled equations.

The toolboxes provide :

- Libraries to manipulate physics-based models and couple thermal
- a set of mono and multi-physics application. Within those toolboxes, we will use the *heat transfer*.

Heat transfer corresponds to the energy transfer process as a result of temperature difference. This flow can happen through three different means :

- **Conduction :** the heat goes from the hottest material toward the coldest,
- **Convection :** it is the heat flow between a solid body and the surrounding atmosphere,
- **Radiation :** the heat flows by electromagnetic radiation.

Feel++ uses three file formats to work : `geo`, `json` and `cfg`.

The `geo` fils contains the geometry of the domain we use, define with physical surfaces, lines... We can visualize that geometry with the software Gmsh. From this file, a mesh is generated to apply the Galerkin method on it.

Such a file can look like this. Here, we create a simple square.

```
h = 0.1 ;
Point (1) = {1,0,0,h} ;
Point (2) = {1,1,0,h} ;
Point (3) = {0,1,0,h} ;
Point (4) = {0,0,0,h} ;

Line (1) = {1,2} ;
Line (2) = {2,3} ;
Line (3) = {3,4} ;
Line (4) = {4,1} ;

Curve Loop (1) = {1,2,3,4} ;
Plane Surface (1) = {1} ;

Physical Curve ("Gamma") = {1,2,3,4} ;
Physical Surface ("Omega") = {1} ;
```

Listing 1: Geomerty of the square

We define the four points with their coordinates in 3D and their characteristic size (for the mesh). Then we draw the line between those points. Finally, we need to define physical objects with names that will be used by Feel++ during the simulation. Here we call the surface `Omega` and its border `Gamma`.

Then, we can configure the parameters of the simulation in the JSON file. In that file, we define what is needed by Feel++, such as the values of the parameters, the initial conditions, the boundary condition... A lot of calculation can be made in a section `PostProcess`.

For example, to set a Dirichlet condition on `Gamma` for the field velocity, we write :

Listing 2: Example of JSON file

```
{
    "BoundaryConditions":{
        "velocity":{
            "Dirichlet":{
                "Gamma":{"expr":"0"}
```

```
6                 }
7             }
8         }
9 }
```

In the JSON file, we can also control which quantities we want to export to visualize them.

Finally, the cfg file allows passing command-line options to Feel++. It particularly allows to set up the output directory, the mesh, the time stepping... For example, in those lines, we ask to use the geometry `cfd.geo`, using a mesh size of `0.03`.

Listing 3: Example of cfg file

```
1 [fluid]
2 mesh.filename=$cfgdir/cfd.geo
3 gmsh.hsize=0.03
```

2.2 Study of a case

Let's start by studying a case of heat transfer in a 2D building[10] already implemented in Feel++. We consider a building consisting of two rooms (domains Ω_1 and Ω_3) separated by a plaster wall (Ω_2). Each room has a heat source, a radiator. The model is shown in figure 3. The temperature T is calculated using the heat equation.

In the model, the conditions at the edges, i.e. on the walls of both rooms and the wall, must also be taken into account. On the borders of the two radiators, we put Dirichlet conditions, i.e. we consider that the radiators are always at the same temperature: $T(t) = T_{r_i} \forall t$. On the walls of the domains Ω_i (i.e. of the two rooms and the wall), we put Robin's conditions, the temperature on these borders depends on the temperature outside the domain (which can also vary):

$$-k(y)\nabla T \cdot \mathbf{n} = h(T - T_{\text{ext}}) \tag{2.1}$$

($k(y)$ is the thermal conductivity of the air at the height y in the room, which we consider here as linear between the floor and the ceiling).

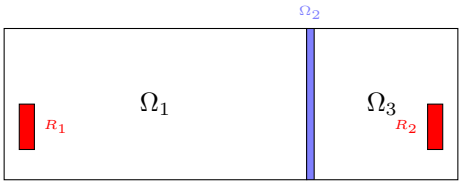


Figure 3: Model geometry

We can simulate this model with the command `mpirun -np 4 feelpp_toolbox_heat_2d -config-file thermo2dCase2.cfg` (we run it simultaneously on 4 processors). Once the simulation has been carried out, it is possible to view the temperature value over time in the building using Paraview. We can see how the radiators heat the air in both rooms, and how the heat is transmitted into the wall Ω_2. We can also draw level lines (we didn't do it here because the result is already visible). The result after 63 seconds is given in figure 4.

These tests can also be done on other shapes (e.g. a 3D building or even on a car engine to find out the temperature distribution on the different materials).

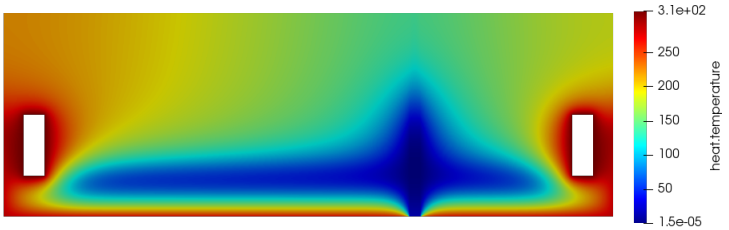


Figure 4: Result of `feelpp_toolbox_heat_2d`

2.3 Benchmark boundary layer

We want now to create a benchmark about boundary layer[8], which has not yet been implemented using Feel++. With Zohra Djatouti, we have to create the geometry of the model, and the `json` and `cfg` files.

The boundary layer is the interface zone between a body and the surrounding fluid during a relative movement between the two. In this layer, near the wall, the fluid velocity changes from the wall velocity (that we consider equal to 0 here) to the free stream velocity (u_∞). For that problem over a flat plate, there is an analytical approximation of the solution which exists. We study here the development of the momentum and the thermal boundary layer over an isothermal plate.

The geometry of the model is given in figure 5. We consider here a surface in two dimensions. The flow enters the domain through the left border (the *inlet*), with a velocity of 1 $\mathrm{m}\cdot\mathrm{s}^{-1}$, and at a temperature of $T_{in} = 20°\mathrm{C}$. The formation of the thermal boundary layer is initiated by a sudden increase in the wall temperature (at the red point in the figure), from $T_{in} = 20°\mathrm{C}$ to $T_w = 30°\mathrm{C}$.

Furthermore, initial conditions can be chosen arbitrarily in this problem.

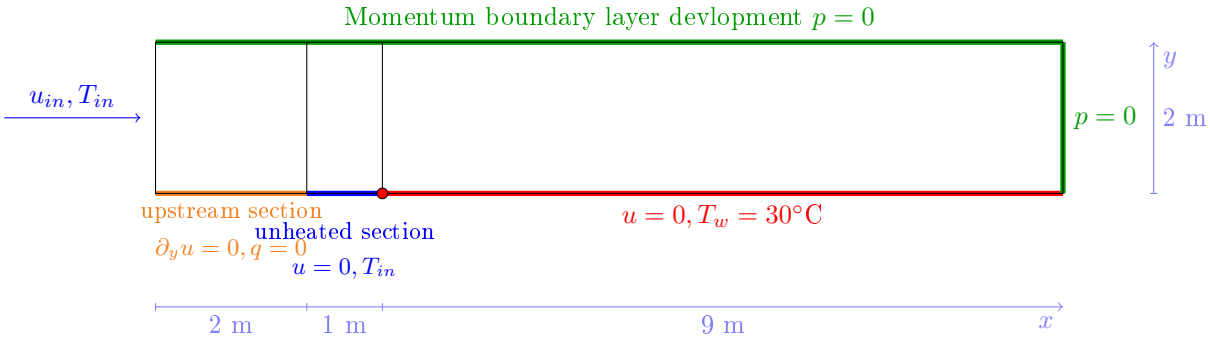


Figure 5: Geometry of the model

The first step is to create the `geo` file to generate a mesh. This step is not so hard because the geometry of our case is quite simple. We define the 8 points delimiting the domain, and then the lines that we need for the surfaces and the boundary conditions. Moreover, we add to the geometry vertical lines every meter to take measures in the post-processing of the simulation.

Then we want to create the JSON file, which is not quite easy because there are a lot of special conditions where the syntax used by Feel++ is hard to handle.

Finally, we write the cfg file. We do the simulation over a time of 10 seconds, with a time-step of 0.02 seconds.

Listing 4: Configuration of the time

```
1 [ts]
2 time-step=0.02
3 time-final=10
```

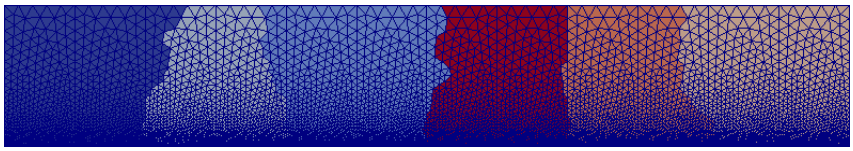
To run the simulation with the toolbox `heatfluid`, we have to go to the Atlas machine with a remote ssh connection (thanks to Visual Studio Code, we can do it easily !)

In the folder with the geo, json and cgf files, the command line to get the results is :

```
user@atlas:boundlayer$ feelpp_toolbox_heatfluid --config-file boundlayer.cfg
```

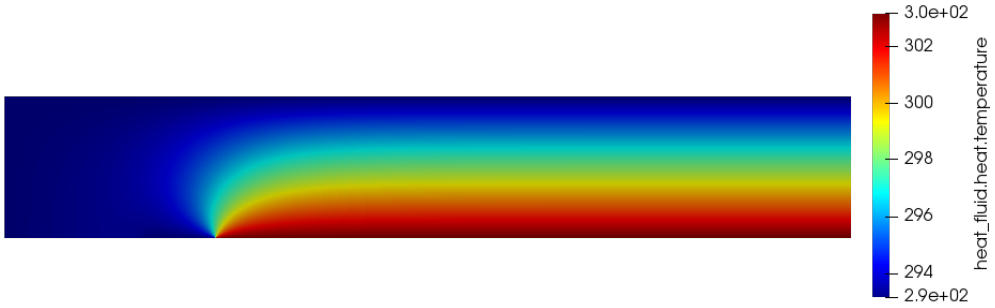
In figure 6(a), we represent the elements of the mesh, with the separation made by Feel++ on the differents process (with used 6 of them). To have a better vizualisation of the boundary layer, we have a tinier mesh on the bottom border of the domain. On the figure 6(b) we can see the formation of the boundary layer. With Paraview, we can display the result over time and see the formation of the layer. We represented on figure 6(c) the temperature, wich seems to be independant of the time.

However, the calculations proposed in the benchmark don't work totally...



(a) Mesh and `pid`

(b) Velocity (at $t = 4.22$s)



(c) Temperature

Figure 6: Results of the boundary layer becnmark

3 Coupled heat and moisture transfer in building

Let's focus on the problems linked to humidity in buildings. We will mainly study the transfer in porous material. The moisture transfer is linked to heat transfer, that is why we will study a coupled model of heat and transfer.

3.1 Moisture transfer

Moisture is a problem known for a long time in the building industry. It may cause serious deterioration in the construction. We study some causes of moisture in buildings. The content of this section is adapted from [20].

Moisture can come from several sources, but it can also cause damages in several sources. It can cause :

- Electrochemical corrosion of metal,
- Chemical deterioration and dissolution of materials (such as wood),
- discoloration of building finishes,
- growth of biological form...

To get a moisture-related problem, at least four conditions have to be satisfied :

1. A moisture source must be available,
2. There must be a means for this moisture to travel,
3. There also have to be a driving force to help moisture to move,
4. The materials involved in the construction must be susceptible to moisture damages.

A first solution to avoid moisture problems is to eliminate any one of those conditions. But it is impossible to delete the four of them, for practical and economical reasons. A more practical approach is to control or manage moisture. We can figure that problems with a *moisture balance* : one side represent how much the materials can support balance, and the other represent the sources of moisture and the possible way to eliminate it (cf figure 7) : we must keep the balance.

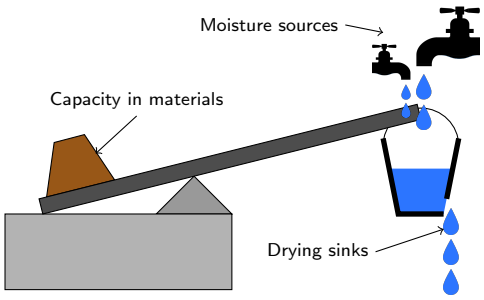


Figure 7: Moisture balance

It is well known that building construction won't be perfect (isolation problem, humidity leak...), but more strategies have been thought in order to increase drying potential and storage capacity.

Here is a list of four main sources of moisture in buildings :

- **Piping leaks and rain penetration :** they are obvious sources of moisture and must be avoided. A little leak can lead to a serious problem. For example, a tiny hole in which let pass one drop per hour onto a drywall ceiling will cause rapid and serious mold growth.
- **Water vapor:** it can be almost as problematic as direct liquid water sources, but the magnitude of the moisture involved is much less. It can be caused by condensation of water vapor on a cold water pipe, but this condensation can also occur within the wall, which is quite problematic. Moreover, a significant amount of moisture can be generated by the occupants and their activities.
- **Soil :** It is a large source of moisture in liquid and vapor forms. Liquid water penetrate in it through cracks or holes. It also provides an inexhaustible supply of water vapor, which is also a large moisture source. Water vapor from the floor enters buildings by diffusion.
- **Built-in moisture :** This source is specific to the type of building and only play a role for the first few years of a building's life.

3.2 Transfer mechanisms

Now, let's focus on principle moisture transport processes. It is possible that these processes are mixed. We also give the model and equation associated to each transport process, taken from [16] and [15].

3.2.1 Vapor diffusion

It moves water vapor from region of high concentration to low concentration, through the air or through porous materials. It will cause wetting and corrosion if it acts for long. It is not the cause of moisture damage in walls, but it may cause damage into the roof, because of the sun heating the water.

Depending on the middle where the vapor is, there are several equations describing the flow of vapor. They all depend on the gradient of the pressure.

The diffusion of water vapor in the air is given with this equation :

$$g_v = -\delta \nabla p \tag{3.1}$$

With :

- g_v [kg m^{-2}s^{-1}] vapor diffusion flux density
- p [Pa] water vapor partial pressure
- δ [kg m^{-1}s^{-1}Pa^{-1}] water vapour diffusion coefficient in air. This quantity is given by the formula $\delta = 2.0 \times 10^{-7} \frac{T^{0.81}}{P_L}$, with T the ambient temperature and P_L the ambient air pressure.

We are interested in the vapor transport thought porous materials. The new equation is given by simply introducing a water diffusion resistance factor :

$$g_v = -\frac{\delta}{\mu} \nabla p \tag{3.2}$$

The coefficient μ is the water vapour diffusion resistance factor. The two coeficients δ and μ both depend on the building material. Let's name $\delta_p = \frac{\delta}{\mu}$, the vapor permeability.

3.2.2 Capillary suction

It moves liquid moisture slowly through porous materials, and the smaller the pores are, the more powerful the capillary suction is, even if the flow is slower. This flow is due to a humidity gradient, between

$$g_w = -D_w(w) \nabla \phi \tag{3.3}$$

With :

- g_w [kg m^{-2}s^{-1}] : liquid flux density
- ϕ [–] : vapor content
- D_w [m^{2}s^{-1}] : capillary transport coefficient

The equation leading moisutre transfert is given here :

$$\xi \frac{\partial \phi}{\partial t} + \nabla \left(-\xi D_w \nabla \phi - \delta_p \nabla (\phi p_{\text{sat}}) \right) = G \tag{3.4}$$

The time partial derivate corresponds to the humidity storage of the material, and the two other terms (in $\nabla \phi$) correspond to the transfer mechanism we saw above (vapor diffusion and capillary suction).

With all of those moisture sources and transport processes in mind, the professionals can design better building and conditioning systems. Furthermore, a high level of moisture in the building may lead to isolation problems or health problems (and it is unsightly).

3.3 Heat transfer

The model [5] proposes this equation of heat transfer is :

$$(\rho C_p)_{\text{eff}} \frac{\partial T}{\partial t} + \nabla \left(-k_{\text{eff}} \nabla T + L_{\text{v}} \delta_{\text{p}} \nabla (\phi p_{\text{sat}}) \right) = Q \tag{3.5}$$

With :

- T [K] : temperature
- Q [W m^{-3}] : heat source
- ρC_p [$\text{J m}^{-3}\text{K}^{-1}$] : volumic heat capacity at constant pressure
- $-k_{\text{eff}}$ [$\text{W m}^{-1}\text{K}^{-1}$] : effective thermal conductivity
- L_{v} [J kg^{-1}] : latent heat of evaporation
- δ_p [s] : vapor permeability

The time partial derivation corresponds to the heat storage of the material, and the term $-k_{\text{eff}} \nabla T$ corresponds to the conduction, convection, and radiation transfer. The last therm is due to the humidity.

3.4 Coupled heat/moisture transfer models

Doing a mass balance, we get those equations taken from the model [5] :

$$(\rho C_p)_{\text{eff}} \frac{\partial T}{\partial t} + \nabla \left(-k_{\text{eff}} \nabla T - L_v \delta_p \nabla (\phi p_{\text{sat}}) \right) = Q \tag{3.6}$$

$$\xi \frac{\partial \phi}{\partial t} + \nabla \left(-\xi D_w \nabla \phi - \delta_p \nabla (\phi p_{\text{sat}}) \right) = G \tag{3.7}$$

With :

- ϕ [–] : relative humidity
- T [K] : temperature
- $(\rho C_p)_{\text{eff}}$ [$\text{J m}^{-3}\text{K}^{-1}$] : effective volumic heat capacity at constant pressure
- k_{eff} [$\text{W m}^{-1}\text{K}^{-1}$] : effective thermal conductivity
- L_v [J kg^{-1}] : latent heat of evaporation
- δ_p [s] : vapor permeability
- ξ [kg m^{-3}] : moisture storage capacity
- D_w [m^2s^{-1}]: moisture diffusivity
- p_{sat} [Pa] : water vapor saturation
- Q [W m^{-3}] : heat source
- G [W m^{-3}] : moisture source

3.5 Data

The model requires the following hygrothermal properties [3] :

- **Sorption isotherm :** It's a function that links the water content w to the relative humidity ϕ. It depends on the material.

$$\begin{aligned} w &= f(\phi) \\ \phi &= f(w) \end{aligned} \tag{3.8}$$

- **Water retention curve :** It's a function that links the water content w to the suction pressure p_{suc} :

$$\begin{aligned} w &= f(p_{\text{suc}}) \\ p_{\text{suc}} &= f(w) \end{aligned} \tag{3.9}$$

- **Water storage capacity :**

$$\xi = \frac{\mathrm{d}w}{\mathrm{d}\phi} \tag{3.10}$$

- **Vapor permeability :** It's a function of the temperature and the relative humidity.

$$\delta_{\mathrm{p}} = f(T, \phi) \tag{3.11}$$

- **Moisture diffusivity :**

$$D_{\mathrm{w}}(w) = -\frac{K(p_{\mathrm{suc}})}{\frac{\partial w}{\partial p_{\mathrm{suc}}}} \tag{3.12}$$

  Where $K(p_{\mathrm{suc}})$ is the material liquid water permeability.

- **Effective heat capacity :**

$$(\rho C_p)_{\mathrm{eff}} = \rho_{\mathrm{s}} C_{p,\mathrm{s}} + w C_{p,\mathrm{w}} \tag{3.13}$$

  Where :

  - ρ_{s} [kg m^{-}3] is the dry solid density
  - $C_{p,\mathrm{s}}$ [J kg^{-1}K^{-1}] is the dry solid specific heat capacity
  - w [kg m^{-3}] is the water content
  - $C_{p,\mathrm{w}}$ [J kg^{-1}K^{-1}] is the water heat capacity at constant pressure

- **Effective thermal conductivity :**

$$k_{\mathrm{eff}} = k_{\mathrm{s}} \left(1 + \frac{bw}{\rho_{\mathrm{s}}}\right) \tag{3.14}$$

  Where :

  - k_{s} [W m^{-1}K^{-1}] is the dry solid thermal conductivity
  - ρ_{s} [kg m^{-}3] is the dry solid density
  - b [–] is the thermal conductivity supplement (indicates how many percent the thermal conductivity increases per mess percent of moisture)

- **Vapor saturation pressure :** [15] gives an empirical value of p_{sat}, as function of the temperature :

$$p_{\mathrm{sat}} = 611 \cdot \exp\left(\frac{aT}{T_0 + T}\right) \tag{3.15}$$

  With :

  - $a = 22.44$ and $T_0 = 272.44$°C if $T < 0$°C
  - $a = 17.08$ and $T_0 = 234.18$°C if $T \geqslant 0$°C

4 Developing with Feel++

4.1 First program : heat equation in time

To become familiar with the development using Feel++, we focus on the heat equation depending on the time.

$$\begin{cases} \dfrac{\partial u}{\partial t} - \Delta u = 0 & \text{on } \Omega \times \mathbb{R}_+ \\ u(x,0) = u_0(x) & \text{on } \Omega \\ u(x,t) = g_D(x,t) & \text{on } \partial\Omega_D \times \mathbb{R}_+ \end{cases} \tag{4.1}$$

To make it, we use the tutorial of the Feel++ Developer Manual [11], which gives an introduction to Feel++ programming to solve a PDE.

The first step consists in writing the variationnal problem of the problem. We approximate the time derivation :

$$\frac{\partial u}{\partial t} \approx \frac{u^n - u^{n-1}}{\Delta t} \tag{4.2}$$

where $u^n(x) = u(x, t^n)$, with $t^n = n\Delta t$.

Knowing the solution at time t^{n-1}, we want to know it at time t^n.

Making calculations we find the variationnal problem : Find $u \in H_0^1(\Omega)$ such as $\forall v \in H_0^1(\Omega)$

$$\int_\Omega u^n v + \Delta t \nabla u^n \cdot \nabla v \, \mathrm{d}x = \int_\Omega \left(u^{n-1} + \Delta t f^n\right) v \, \mathrm{d}x \tag{4.3}$$

This problem is equivalent to : Find $u \in H_0^1(\Omega)$ such that $\forall v \in H_0^1(\Omega)$, $a(u,v) = l(v)$ with :

$$a(u,v) = \int_\Omega uv + \Delta t \nabla u \cdot \nabla v \tag{4.4}$$

$$l(v) = \int_\Omega \left(u^{n-1} + \Delta t f^n\right) v \tag{4.5}$$

To get the values of the problem (such as $f, g_D \ldots$), we have to put them either as option at the execution (`--function.f=1`) or in a cfg file. We also have to use a geometry to load a mesh of where the simulation will be done.

Finaly, we create the objects of the problem : the objects representing the solution u and the test function v. Those functions live on an approximated space `Vh`. We also create the objects associated to the two forms l and a.

The code (compacted) is given on listing 5.

Listing 5: Heat equation on time

```
1  int main (int argc, char ** argv) {
2      // boundary value   (idem for f, u0)
3      auto g_D = expr (soption (_name="functions.d")) ;
4
5      // time parameters
6      double dt = doption (_name="ts.time-step") ;
7      double tFinal = doption (_name="ts.time-final") ;
8
9      // load the mesh
10     auto mesh = loadMesh (_mesh = new Mesh<Simplex<2>>) ;
11     auto Vh = Pch<2> (mesh) ;
12     auto u = Vh->element (u0) ;
13     auto v = Vh->element () ;
14
15     auto l  = form1 (_test = Vh) ;
16     auto lt = form1 (_test = Vh) ;
17     auto a  = form2 (_trial=Vh, _test=Vh) ;
18     auto at = form2 (_trial=Vh, _test=Vh) ;
19     auto e  = exporter (_mesh = mesh, _name = "heat") ;
20     e->step( 0 )->add ("u", u) ;
21
```

```
22      // don't change at each step
23      l = integrate (_range = elements (mesh), _expr = dt * f * id(v)) ;
24      a = integrate (_range = elements (mesh),
25                     _expr = rho*Cp * idt(u) * id(v) + dt*k * gradt(u) * trans
    (grad(v)) ) ;
26      for( double time = dt; time <= tFinal; time += dt ) {
27          lt = l;
28          lt += integrate (_range = elements (mesh),
29                           _expr = rho*Cp * idv(u)  * id(v)) ;
30
31          at = a ;
32          // Dirichlet conditions, we have to do this at the end
33          at += on (_range = markedfaces (mesh, "Gamma_D"),
34                    _rhs = lt, _element = u, _expr = g_D ) ;
35
36          at.solve (_rhs=lt, _solution=u) ;
37
38          e->step( time )->add ("u", u) ;
39      }
40      e->save () ;
41  }
```

Definition 4.1. Let $a\colon X \times Y \to \mathbb{R}$ be a bilinear form and $l\colon X \to \mathbb{R}$ a linear form. We seek $u \in Y$ such as $\forall v \in X, a(u,v) = l(v)$

- Y is the space where the solution u lives, it is called the *trial* space.
- X is the space where there are the functions allowing to test the equation. This space is called *test*.

The algebric representation of the bilinear form is a matrix $A = \left(a(\phi_i, \psi_j)\right)_{i,j}$ where $(\phi_i)_i$ (resp $(\psi_i)_i$) is a basis of X (Y). To built this matrix in Feel++, the command `idt(u)` gives the basis functions associated to u, living in Y (`t` for trial). The command `id(v)` gives the basis of the test space.

We may also want to access to a function of an space (X, Y or another). The command `idv(u)` returns the interpolant of u on his space.

In our case, the trial space and the test space are the same, that is why on lines 16 to 19 we have `Vh` on both parameters. But it is not always like that... At each step, we save the result in order to visualize it in paraview.

The resolution strategy used here is not very efficient because at each step we define again the forms with the solution calculated on the previous iteration.

Now, we consider a more complet problem, with Neumann and Robin conditions :

$$\begin{cases} \rho C_p \dfrac{\partial u}{\partial t} - \Delta u = 0 & \text{on } \Omega \times \mathbb{R}_+ \\ u(x,0) = u_0(x) & \text{on } \Omega \\ u(x,t) = g_D(x,t) & \text{on } \partial\Omega_D \times \mathbb{R}_+ \\ \frac{\partial u}{\partial n} = g_N & \text{on } \partial\Omega_N \times \mathbb{R}_+ \\ -k\frac{\partial u}{\partial n} = h(u - u_\infty) & \text{on } \partial\Omega_R \times \mathbb{R}_+ \end{cases} \tag{4.6}$$

where ρ is the volumic mass, C_p the heat capacity, h the convection heat transfer coefficient and u_∞ the temperature far from the border[1]. The variationnal problem is almost the same : find $u \in H$ such that

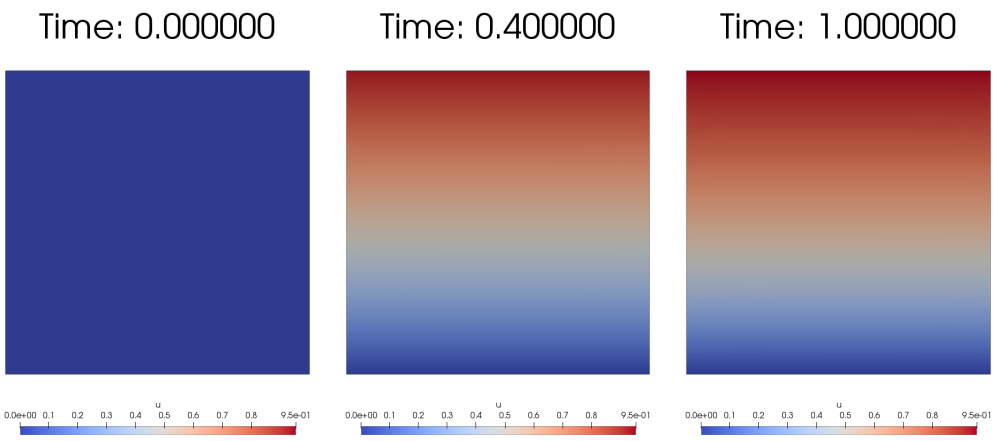
$$a(u,v) = l(v) \qquad \forall v \in H \tag{4.7}$$

where $H = \left\{u \in H^1(\Omega) \middle| u_{|\partial\Omega_D} = 0\right\}$ and

$$a(u,v) = \rho C_p \int_\Omega uv + \Delta t \int_\Omega \nabla u \cdot \nabla v + \int_{\Gamma_R} huv \tag{4.8}$$

$$l(v) = \int_\Omega \left(\rho C_p u^{n-1} + \Delta t f^n\right) v + \int_{\Gamma_N} g_N v + \int_{\Gamma_R} h u_{\text{inf}} v \tag{4.9}$$

[1] NB : in the code, this quantity is named `Tinf`



(a) $t = 0$ (b) $t = 0.4$ (c) $t = 10$

Figure 8: Result of `heat_time`

We just have to add a few lines in the program. To launch that program, we either have to launch CTest, or go in the build folder and write :

```
user@cemosis:src$ ./feelpp_p_heat_time --config-file heat_time.cfg
```

The result, with default values (all the parameters are 1) is given on figure 8. We are on the unit square where the Dirichlet border Γ_D is the bottom border (and $g_D = 0$), Γ_N is on the left and right borders, and Γ_R is the top border.

5 Strategy of resolution

The equation for heat and moisture transport are given in (3.6) and (3.7).

5.1 Linearisation of the problem

Let $\Delta t > 0$, we approximate the time derivation at the first order the quantities :

$$\frac{\partial T}{\partial t} \approx \frac{T^{n+1} - T^n}{\Delta t} \qquad \frac{\partial \phi}{\partial t} \approx \frac{\phi^{n+1} - \phi^n}{\Delta t} \tag{5.1}$$

where T^n and ϕ^n are respectively the heat and the moisture quantities at the time $t^n = n\Delta t$.

Furthermore, we linearize the term p^n_{sat} at time t^n at the first order (we recall that p_{sat} depends on the temperature, cf (3.15)) :

$$p^{n+1}_{\text{sat}} \approx p^n_{\text{sat}} \tag{5.2}$$

We make the same approximation for all the other coefficients $(\rho C_p)_{\text{eff}}, k_{\text{eff}}, L_{\text{v}}, \delta_{\text{p}}, \xi$ and D_{w}.

The problem become :

$$\begin{aligned} &(\rho C_p)^n_{\text{eff}} \frac{T^{n+1} - T^n}{\Delta t} + \nabla \left(-k^n_{\text{eff}} \nabla T^{n+1} - L^n_v \delta^n_p \nabla (\phi^{n+1} p^n_{\text{sat}}) \right) = Q^{n+1} \\ &\xi \frac{\phi^{n+1} - \phi^n}{\Delta t} + \nabla \left(-\xi^n D^n_w \nabla \phi^{n+1} - \delta^n_p \nabla (\phi^{n+1} p^{n+1}_{\text{sat}}) \right) = G \end{aligned} \tag{5.3}$$

Let's calculate the variationnal problem of (5.3). Let $v \in X$ and $q \in M$. At the time step n, the first equation become :

$$\begin{aligned} \int_\Omega (\rho C_p)^n_{\text{eff}} \frac{T^{n+1}}{\Delta t} v + \int_\Omega \nabla \cdot \left(-k^n_{\text{eff}} \nabla T^{n+1} \right) v + \int_\Omega \nabla \cdot \left(-L^n_{\text{v}} \delta^n_{\text{p}} p^n_{\text{sat}} \left(\nabla \phi^{n+1} \right) - L^n_{\text{v}} \delta^n_{\text{p}} \left(\nabla p^n_{\text{sat}} \right) \phi^{n+1} \right) v \\ = \int_\Omega \left((\rho C_p)^n_{\text{eff}} \frac{T^n}{\Delta t} + Q^{n+1} \right) v \end{aligned} \tag{5.4}$$

Which leads to, with a partial integration :

$$\begin{aligned} \int_\Omega (\rho C_p)^n_{\text{eff}} \frac{T^{n+1}}{\Delta t} v + \int_\Omega \left(k^n_{\text{eff}} \nabla T^{n+1} \right) \cdot \nabla v + \int_{\partial\Omega} \left(-k^n_{\text{eff}} \nabla T^{n+1} \cdot \mathbf{n} \right) v \\ + \int_\Omega L^n_{\text{v}} \delta^n_{\text{p}} p^n_{\text{sat}} \left(\nabla \phi^{n+1} \right) \cdot \nabla v + \int_{\partial\Omega} \left(-L^n_{\text{v}} \delta^n_{\text{p}} p^n_{\text{sat}} \left(\nabla \phi^{n+1} \right) \cdot \mathbf{n} \right) v \\ + \int_\Omega \left(L^n_{\text{v}} \delta^n_{\text{p}} \left(\nabla p^n_{\text{sat}} \right) \phi^{n+1} \right) \cdot \nabla v + \int_{\partial\Omega} \left(-L^n_{\text{v}} \delta^n_{\text{p}} \left(\nabla p^n_{\text{sat}} \right) \phi^{n+1} \cdot \mathbf{n} \right) v \\ = \int_\Omega \left((\rho C_p)^n_{\text{eff}} \frac{T^n}{\Delta t} + Q^{n+1} \right) v \end{aligned} \tag{5.5}$$

And the second one :

$$\begin{aligned} \int_\Omega \left(\xi^n \frac{\phi^{n+1}}{\Delta t} \right) q + \int_\Omega \nabla \cdot \left(-\xi^n D^n_{\text{w}} \nabla \phi^{n+1} \right) q \\ + \int_\Omega \nabla \cdot \left(-\delta^n_{\text{p}} (\nabla \phi^{n+1}) p^n_{\text{sat}} \right) q + \int_\Omega \nabla \cdot \left(-\delta_{\text{p}} \phi^{n+1} \nabla p^n_{\text{sat}} \right) q \\ = \int_\Omega \left(\xi^n \frac{\phi^n}{\Delta t} + G^{n+1} \right) q \end{aligned} \tag{5.6}$$

Which leads to :

$$\begin{aligned} \int_\Omega \left(\xi^n \frac{\phi^{n+1}}{\Delta t} \right) q + \int_\Omega \left(\xi^n D^n_{\text{w}} \nabla \phi^{n+1} \right) \cdot \nabla q + \int_{\partial\Omega} \left(-\xi^n D^n_{\text{w}} \nabla \phi^{n+1} \cdot n \right) q + \int_\Omega \left(\delta^n_{\text{p}} (\nabla \phi^{n+1}) p^n_{\text{sat}} \right) \nabla q \\ + \int_{\partial\Omega} \left(-\delta^n_{\text{p}} (\nabla \phi^{n+1}) p^n_{\text{sat}} \cdot \mathbf{n} \right) q + \int_\Omega \left(\delta_{\text{p}} \phi^{n+1} \nabla p^n_{\text{sat}} \right) \nabla q + \int_{\partial\Omega} \left(-\delta_{\text{p}} \phi^{n+1} \nabla p^n_{\text{sat}} \cdot \mathbf{n} \right) q \\ = \int_\Omega \left(\xi^n \frac{\phi^n}{\Delta t} + G^{n+1} \right) q \end{aligned} \tag{5.7}$$

At time $n+1$, knowing the result at the iteration n, the variationnal problem is : Find $(T, \phi) \in X \times M$ such as

$$\begin{cases} a(T,v) + b(\phi,v) &= f(v) \qquad \forall v \in X \\ c(T,q) + d(\phi,q) &= g(q) \qquad \forall q \in M \end{cases} \tag{5.8}$$

with :

$$a(T^{n+1},v) = \int_\Omega (\rho C_p)^n_{\text{eff}} \frac{T^{n+1}}{\Delta t} v + \int_\Omega \left(k^n_{\text{eff}} \nabla T^{n+1}\right) \cdot \nabla v - \int_{\partial\Omega} \left(k^n_{\text{eff}} \nabla T^{n+1} \cdot \mathbf{n}\right) v \tag{5.9}$$

$$\begin{aligned} b(\phi^{n+1},v) = &\int_\Omega L^n_{\text{v}} \delta^n_{\text{p}} p^n_{\text{sat}} \left(\nabla \phi^{n+1}\right) \cdot \nabla v - \int_{\partial\Omega} \left(L^n_{\text{v}} \delta^n_{\text{p}} p^n_{\text{sat}} \left(\nabla \phi^{n+1}\right) \cdot \mathbf{n}\right) v \\ &+ \int_\Omega \left(L^n_{\text{v}} \delta^n_{\text{p}} \left(\nabla p^n_{\text{sat}}\right) \phi^{n+1}\right) \cdot \nabla v - \int_{\partial\Omega} \left(L^n_{\text{v}} \delta^n_{\text{p}} \left(\nabla p^n_{\text{sat}}\right) \phi^{n+1} \cdot \mathbf{n}\right) v \end{aligned} \tag{5.10}$$

$$c(\phi^{n+1},q) = 0 \tag{5.11}$$

$$\begin{aligned} d(\phi^{n+1},q) = &\int_\Omega \left(\xi^n \frac{\phi^{n+1}}{\Delta t}\right) q + \int_\Omega \left(\xi^n D^n_{\text{w}} \nabla \phi^{n+1}\right) \cdot \nabla q - \int_{\partial\Omega} \left(\xi^n D^n_{\text{w}} \nabla \phi^{n+1} \cdot \mathbf{n}\right) q \\ &+ \int_\Omega \left(\delta^n_{\text{p}} (\nabla \phi^{n+1}) p^n_{\text{sat}}\right) \nabla q - \int_{\partial\Omega} \left(\delta^n_{\text{p}} (\nabla \phi^{n+1}) p^n_{\text{sat}} \cdot \mathbf{n}\right) q \\ &+ \int_\Omega \left(\delta_{\text{p}} \phi^{n+1} \nabla p^n_{\text{sat}}\right) \nabla q - \int_{\partial\Omega} \left(\delta_{\text{p}} \phi^{n+1} \nabla p^n_{\text{sat}} \cdot \mathbf{n}\right) q \end{aligned} \tag{5.12}$$

$$f(v) = \int_\Omega \left((\rho C_p)^n_{\text{eff}} \frac{T^n}{\Delta t} + Q^{n+1}\right) v \tag{5.13}$$

$$g(q) = \int_\Omega \left(\xi^n \frac{\phi^n}{\Delta t} + G^{n+1}\right) q \tag{5.14}$$

We approximate X with X_h and M with M_h, both of finie dimensions. In our case, those space will be $P_{c,h}$. Let $(x_{h,i})_{0 \leqslant i \leq N_T}$ and $(m_{h,i})_{0 \leqslant i \leq N_\phi}$ two basis of those spaces. The aproximate form of the variationnal problem become : Find $(T, \phi) \in X_h \times M_h$ such as

$$\begin{cases} a(T_h,v_h) + b(\phi,v_h) &= f(v_h) \qquad \forall v_h \in X_h \\ d(\phi,q_h) &= g(q_h) \qquad \forall q_h \in M_h \end{cases} \tag{5.15}$$

Let be the following matrices, associated to the three forms :

$$A = \left(a(x_{h,i}, x_{h,j})\right) \in \mathbb{R}^{N_T \times N_T} \tag{5.16}$$

$$B = \left(b(x_{h,i}, m_{h,j})\right) \in \mathbb{R}^{N_T \times N_\phi} \tag{5.17}$$

$$D = \left(d(m_{h,i}, m_{h,j})\right) \in \mathbb{R}^{N_\phi \times N_\phi} \tag{5.18}$$

and the two vectors corresponding to the linear forms :

$$F = \left(f(x_{h,j})\right) \in \mathbb{R}^{N_T} \tag{5.19}$$

$$G = \left(g(m_{h,j})\right) \in \mathbb{R}^{N_\phi} \tag{5.20}$$

We can notice that only A is indepedendant of the time.
The problem is equivalent to solve that matricial system :

$$\begin{bmatrix} A & B \\ 0 & D \end{bmatrix} \begin{bmatrix} T \\ \phi \end{bmatrix} = \begin{bmatrix} F \\ G \end{bmatrix} \tag{5.21}$$

This system can be solved in "waterfall". The solution is :

$$\begin{cases} \phi = D^{-1} G \\ T = A^{-1} \left(F - B\phi\right) \end{cases} \tag{5.22}$$

That gives us the algorithm 1 to solve the problem.

Algorithm 1: Linear problem

Input: T^0, ϕ^0 initial heat and moisture
Assemble A (indepedant of time)
$t \leftarrow 0$
while $t < t_{\max}$ **do**
  Assemble B, D, F and G
  $\phi \leftarrow D^{-1}G$
  $T \leftarrow A^{-1}(F - B\phi)$
  $t \mathrel{+}= \Delta t$
end
Output: (T, ϕ)

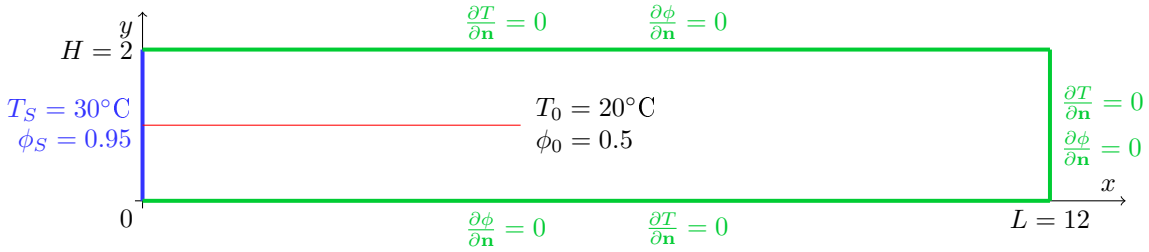


Figure 9: Geometry of the model

5.2 Benchmark : Moisture uptake within a semi-infinite region (EN 15026:2007)[2]

5.2.1 Geometry

We have $\Omega = [0, 12] \times [0, 2]$ (cf figure 9). We write $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ where :

- $\partial\Omega_D$ is the left border of the domain on which Dirichlet conditions are applied,
- $\partial\Omega_N$ is composed of the three other borders, on which homogenous Neunamm conditions stand.

5.2.2 Material data

The data from section 3.5 for this benchmark are given here. They are taken from the norm EN 15026 [5].
General data :

- $T_{\text{ref}} = 293.15\text{K}$
- $\rho_{\text{w}} = 1\,000\,\text{kg} \cdot \text{m}^{-3}$
- $L_v = 2\,500\,\text{kJ}\,\text{kg}^{-1}$
- $R_{\text{H}_2\text{O}} = 462\,\text{J} \cdot \text{kg}^{-1} \cdot \text{K}^{-1}$

Material data :

- **Water retention curve :** (3.9)

$$\begin{aligned} w &= \frac{146}{\left(1 + (8 \times 10^{-8} p_{\text{suc}})^{1.6}\right)^{0.375}} \\ p_{\text{suc}} &= 0.125 \times 10^8 \left(\left(\frac{146}{w}\right)^{\frac{1}{0.375}} - 1\right)^{0.625} \end{aligned} \tag{5.23}$$

- **Sorption isoterm :** (3.8)

$$\begin{aligned} w &= \frac{146}{\left(1 + (-8 \times 10^{-8} \cdot R_{\text{H}_2\text{O}} T_{\text{ref}} \rho_w \ln(\phi))^{1.6}\right)^{0.375}} \\ \phi &= \exp\left(-\frac{1}{R_{\text{H}_2\text{O}} T_{\text{ref}} \rho_{\text{w}}} 0.125 \times 10^8 \left(\left(\frac{146}{w}\right)^{\frac{1}{0.375}} - 1\right)^{0.625}\right) \end{aligned} \tag{5.24}$$

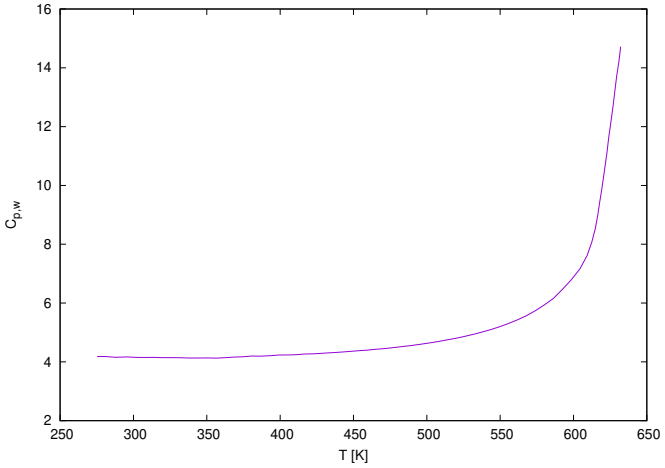


Figure 10: Heat capacity of water at constant pressure

- **Vapour diffusion :** (3.11)

$$\delta_{\mathrm{p}} = \frac{M_{\mathrm{w}}}{R_{\mathrm{H_2O}} T_{\mathrm{ref}}} \frac{26.1 \times 10^{-6}}{200} \frac{1 - \frac{w}{146}}{0.503 \left(1 - \frac{w}{146}\right)^2 + 0.497} \tag{5.25}$$

- **Liquid water permeability :** (3.12)

$$\begin{aligned} K = \exp\big(&-39.2619 + 0.0704 \cdot (w-73) - 1.7420 \times 10^{-4} \cdot (w-73)^2 - 2.7953 \times 10^{-6} \\ &\cdot (w-73)^3 - 1.1566 \times 10^{-7} \cdot (w-73)^4 + 2.5969 \times 10^{-9} \cdot (w-73)^5\big) \end{aligned} \tag{5.26}$$

- **Thermal conductivity :** (3.14)

$$k_{\mathrm{eff}} = 1.5 + \frac{15.8}{1000} w \tag{5.27}$$

- **Heat capacity for dry material :** (3.13)

$$\rho_{\mathrm{s}} C_{p,\mathrm{s}} = 1.824 \times 10^6 \tag{5.28}$$

- **Heat capacity at constant pressure :** (3.13) From [22], we can get the values at different temperatures of the heat capacity $C_{p,\mathrm{w}}$: a graphic of those values is given. Thanks to WebPlotDigitizer we can get a csv file that we will fit with Feel++. The graph of the values from the website is given in figure 10.

5.2.3 Variationnal problem

We take $X = M = \left\{ v \in H^1(\Omega) \,\middle|\, v|_{\partial\Omega_D} = 0 \right\}$ The problem (5.8) is the same, but the values of the forms are simplified :

$$a(T^{n+1}, v) = \int_\Omega (\rho C_p)^n_{\mathrm{eff}} \frac{T^{n+1}}{\Delta t} v + \int_\Omega \left(k^n_{\mathrm{eff}} \nabla T^{n+1}\right) \cdot \nabla v \tag{5.29}$$

$$\begin{aligned} b(T^{n+1}, v) = &\int_\Omega L^n_{\mathrm{v}} \delta^n_{\mathrm{p}} p^n_{\mathrm{sat}} \left(\nabla \phi^{n+1}\right) \cdot \nabla v + \int_\Omega \left(L^n_{\mathrm{v}} \delta^n_{\mathrm{p}} \left(\nabla p^n_{\mathrm{sat}}\right) \phi^{n+1}\right) \cdot \nabla v \\ &- \int_{\partial\Omega_N} \left(L^n_{\mathrm{v}} \delta^n_{\mathrm{p}} \left(\nabla p^n_{\mathrm{sat}}\right) \phi^{n+1} \cdot \mathbf{n}\right) v \end{aligned} \tag{5.30}$$

$$c(\phi^{n+1}, q) = 0 \tag{5.31}$$

$$\begin{aligned} d(\phi^{n+1}, v) = &\int_\Omega \left(\xi^n \frac{\phi^{n+1}}{\Delta t}\right) q + \int_\Omega \left(\xi^n D^n_{\mathrm{w}} \nabla \phi^{n+1}\right) \cdot \nabla q + \int_\Omega \left(\delta^n_{\mathrm{p}} (\nabla \phi^{n+1}) p^n_{\mathrm{sat}}\right) \nabla q \\ &+ \int_\Omega \left(\delta_{\mathrm{p}} \phi^{n+1} \nabla p^n_{\mathrm{sat}}\right) \nabla q - \int_{\partial\Omega_N} \left(\delta_{\mathrm{p}} \phi^{n+1} \nabla p^n_{\mathrm{sat}} \cdot \mathbf{n}\right) q \end{aligned} \tag{5.32}$$

The values of f and g are the same is in (5.13) and (5.14).

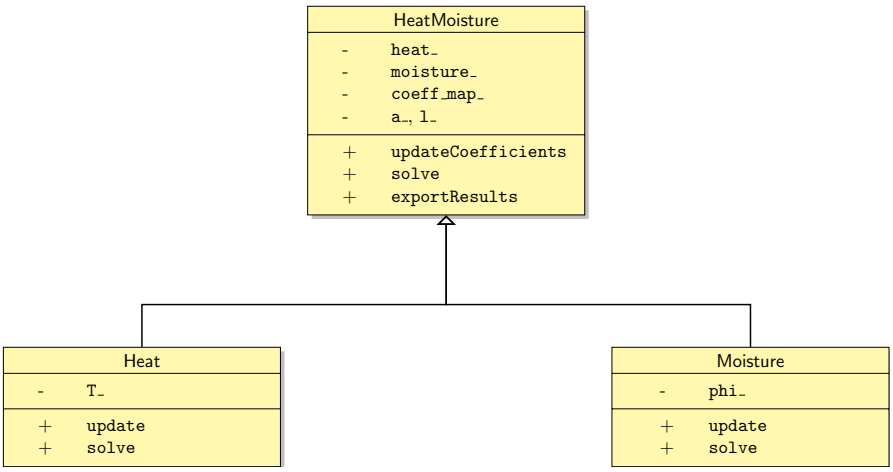


Figure 11: UML diagram

5.3 Implementation

To deal with the fact that the coefficients are variable on Ω, we create for each coefficient a field on the mesh that will contain the values. To group all of them, we put them in a dictionnary, where the keys are the names of the coefficients (`xi`, `p_sat`...).

We create three ojects classes :

- `Heat` delaing with the heat equation (3.6)
- `Moisture` for the moisture equation (3.7)
- `HeatMoisture` doing the coupling and managing the coefficients

The class `HeatMoisture` contains an object `Heat` and an object `Moisture`. It has a method `updateCoefficients` which do the calculation of the new coefficients with the actual values of T and ϕ. The UML diagram of theses classes is given in figure 11.

Each sub-class contains a method `solve` which execute the resolution of the equation associed :

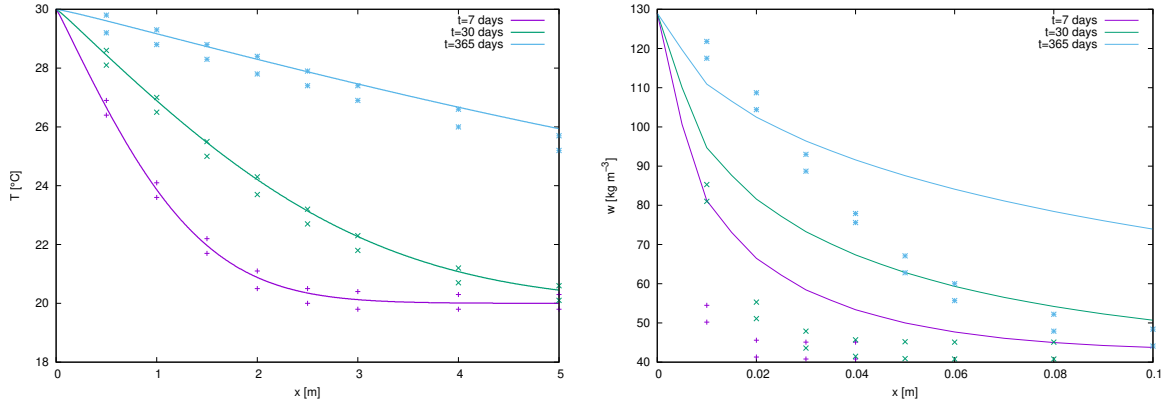
- First, `Moisture.solve` make the resolution $D\phi = G$ of the system 5.22
- Then, `Heat.solve` create the object associated to the linear form $F - B\phi$ (ϕ is the result of the previous step) and make the resolution $AT = (F - B\phi)$

We also add a *Picard loop* [12] which is a method of successive substitution. At each time step, we introduce sub-iterations in order to have a better solution. The principle of the loop is the same as the general loop : from the solution calculated previously, we calculate the new solution. But here we do this until the solution is stabilized, *i.e.* when the norm of the difference between two successive results (for T and ϕ) is small enough.
At each time step, we save all the coefficients of the dictionary.

To compile and launch the application, we have to make a compilation with `ninja` (cf the Readme file at the root of the github repository) and write this command in the build folder :

```
user@cemosis:build/src$ mpirun -np 4 ./feelpp_hm_heat_moisture --config-file
cases/moisture-uptake/moisture-uptake.cfg --heat.ksp-monitor=1 --moisture.ksp-monitor=1
--ts.time-step=86400 --ts.time-final=31536000 --hm.picard=false
--hm.export={604800,2592000,31536000}
```

To avoid a large number of export file, we have implemented the option `hm.export`. The solution will be exported at the time given in the parameters. In the example above, we want to export it after 7 days (604 800 seconds), 30 days (2 592 000 seconds) and 1 year (31 536 000 seconds). If we want to get all the export files, we just have to put `hm.export=all`.



(a) Heat T (b) Moisture w

Figure 12: Results

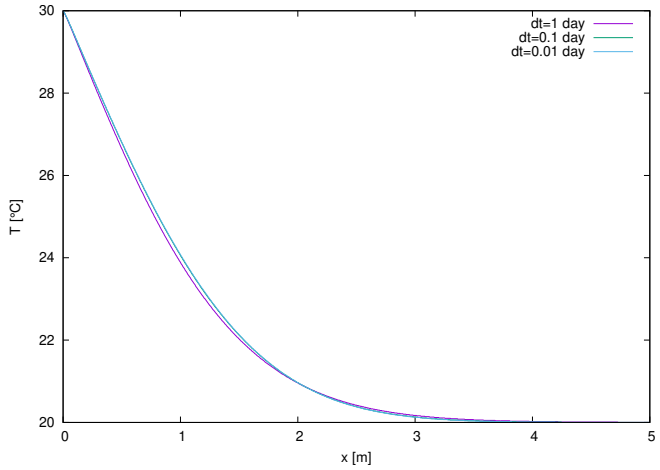


Figure 13: Results for differents time step

5.4 Results

We simulate over a time of 1 year (*ie* 31 536 000 seconds), which takes 27 minutes to run with a time-step of 10 hours.

On figure 12(a) is represented the temperature over the red line on figure 9 at different time of the simulation (7 days, 30 days and 365 days). On the figure, the continuous line represents our results, and the dots represent the range where they should be, taken from [5]. We see that for the temperature, the more we advance in time the more we get outside that range.

We draw on figure 12(b) the water content (for $0 \leqslant x \leqslant 0.1$). This time the result is different as it should be ! There must be a problem in the program (and that problem implies the decay of the temperature).

We also execute the simulation with a smaller `time-step`. In the previous simulation, we took 1 day for each step. We see in figure 13 that the results stay the same for a tinier step. The same behavior can be seen for w.

6 Conclusion

For this project, we studied a theoretical part about heat and moisture transfer (and especially moisture). Then we model them with equations.

Finally, we managed to implement a model of coupled heat / moisture transfer, which gives good results. For now, we can only simulate this transfer in a simple porous wall, but in the future, it will be possible to simulate them in more complex models, such as the building of Illkirch.

The following part, in an internship, is to keep looking at what didn't work on our model, then test the model on another stable case [1]. After that, we will be able to test it on more complex cases, such as the exercises described in [19].

A bridge can be made with the project data when the model will be working. With machine learning, they may be able to complete missing data, and it would be interesting to see if is found for those data correspond to what could be simulated.

A Notations and Units

Symbol	Quantity	Unit
c_{m}	specific heat capacity of dry material	$\mathrm{J/(kg \cdot K)}$
c_{w}	specific heat capacity of liquid water	$\mathrm{J/(kg \cdot K)}$
D_{w}	moisture diffusivity	$\mathrm{m^2/s}$
E_{sol}	total flux density of incident solar radiation	$\mathrm{W/m^2}$
g	density of moisture flow rate	$\mathrm{kg/\left(m^2 \cdot s\right)}$
g_{p}	density of moisture flow rate of available water from precipitation	$\mathrm{kg/\left(m^2 \cdot s\right)}$
g_{v}	density of water vapour flow rate	$\mathrm{kg/\left(m^2 \cdot s\right)}$
g_{w}	density of liquid water flow rate	$\mathrm{kg/\left(m^2 \cdot s\right)}$
$g_{\mathrm{w,max}}$	density of water flow rate which can be absorbed at the surface of a material	$\mathrm{kg/\left(m^2 \cdot s\right)}$
h	surface heat transfer coefficient	$\mathrm{W/\left(m^2 \cdot K\right)}$
h_{c}	convective heat transfer coefficient	$\mathrm{W/\left(m^2 \cdot K\right)}$
h_{e}	specific latent enthalpy of evaporation or condensation	$\mathrm{J/kg}$
h_{r}	radiative heat transfer coefficient	$\mathrm{W/\left(m^2 \cdot K\right)}$
K	liquid conductivity	$\mathrm{s/m}$
p_{a}	ambient atmospheric pressure	Pa
p_{suc}	suction pressure	Pa
p_{v}	partial water vapour pressure	Pa
$p_{\mathrm{v,a}}$	partial water vapour pressure in the air	Pa
$p_{\mathrm{v,s}}$	partial water vapour pressure at a surface	Pa
$p_{\mathrm{v,sat}}$	saturated water vapour pressure	Pa
p_{w}	water pressure inside pores	Pa
q	density of heat flow rate	$\mathrm{W/m^2}$
q_{lat}	density of latent heat flow rate	$\mathrm{W/m^2}$
q_{sens}	density of sensible heat flow rate	$\mathrm{W/m^2}$
R_{w}	liquid moisture flow resistance of interface	$\mathrm{m/s}$
R_{H2O}	gas constant of water vapour	$\mathrm{J/(kg \cdot K)}$
$s_{\mathrm{d,s}}$	equivalent vapour diffusion thickness of a surface layer	m
T	thermodynamic temperature	K
T_{a}	air temperature of the surrounding environment	K
T_{eq}	equivalent temperature of the surrounding environment	K
T_{r}	mean radiant temperature of the surrounding environment	K
T_{surf}	surface temperature	K
t	time	s
v	wind speed	$\mathrm{m/s}$
w	moisture content	$\mathrm{kg/m^3}$
α_{sol}	distance	m
δ_0	solar absorptance	$\mathrm{kg/(m \cdot s \cdot Pa)}$
δ_{p}	vapour permeability of still air	$\mathrm{kg/(m \cdot s \cdot Pa)}$
ε	vapour permeability of material	$-$
λ	longwave emissivity of the external surface	$\mathrm{W/m \cdot K)}$
φ	thermal conductivity	$-$
μ	relative humidity	$\mathrm{kg/m^3}$
ρ_{a}	diffusion resistance factor	$\mathrm{kg/m^3}$
ρ_{m}	density of air	$\mathrm{kg/m^3}$
ρ_{w}	density of liquid water	$\mathrm{kg/m^3}$
σ_{s}	Stefan-Boltzmann constant	$\mathrm{W/\left(m^2 \cdot K^4\right)}$

References

[1] Cemosis. Analytical verification: drying out of a layer. http://docs.cemosis.fr/hygrothermy/0.1.0/benchmark-moisture/.

[2] Cemosis. Benchmark: Moisture uptake within a semi-infinite region (EN 15026:2007). http://docs.cemosis.fr/hygrothermy/0.1.0/benchmark_EN15026_2007/.

[3] Cemosis. Heat and Moisture Transfer in Building Modeling. http://docs.cemosis.fr/hygrothermy/0.1.0/cen2007/#_required_data.

[4] Cemosis. 4fastsim-ibat. http://www.cemosis.fr/projects/4fastsim-ibat/, 21 octobre 2019.

[5] CEN. Hygrothermal performance of building components and building elements - Assessment of moisture transfer by numerical simulation, 2007.

[6] Convention-cadre des Nations unies sur les changements climatiques. Adoption of the paris agreement, 12 décembre 2015.

[7] EDF. Le bâtiment, premier poste de consomation. https://www.edf.fr/edf/le-batiment-premier-poste-de-consommation-d-energie, 2013.

[8] Feel++. http://docs.feelpp.org/cases/0.108/heatfluid/boundary_layer_development/README/.

[9] Feel++. Documentation. http://docs.feelpp.org/feelppdocs/stable/.

[10] Feel++. Example Toolboxes 2D building. http://docs.feelpp.org/cases/0.108/heat/2Dbuilding/README/.

[11] Feel++. Feel++ developer manual. http://docs.feelpp.org/tutorial-dev/latest/.

[12] Feel++. Non linear problems. http://docs.feelpp.org/math/fem/nonlinear/#_picard_strategy.

[13] Idrissa, Niakh. Réduction d'ordre et assimilation de données, application à l'aérothermie, 2020.

[14] Intergovernmental Panel on Climate Change, 2019.

[15] Künzel, Hartwig M. *Simultaneous Heat and Moisture Transport in Building Components.* PhD thesis, Fraunhofer Institute of Building Physics, 1995.

[16] Dylan Lelièvre. *Numerical simulation of heat and moisture transfers in a building multi-layer wall made of bio-based materials.* Theses, Université de Bretagne Sud, January 2015.

[17] Légifrance. LOI n. 2015-992 du 17 août 2015 relative à la transition énergétique pour la croissance verte. https://www.legifrance.gouv.fr/affichTexte.do?cidTexte=JORFTEXT000031044385&dateTexte=vig, 2020. Consulté en mars 2020.

[18] Ministère de la Transition écologique et solidaire. Panorama des émissions françaises de gaz à effet de serre. https://ree.developpement-durable.gouv.fr/themes/defis-environnementaux/changement-climatique/emissions-de-gaz-a-effet-de-serre/article/panorama-des-emissions-francaises-de-gaz-a-effet-de-serre. Consulté en mars 2020.

[19] Carsten Rode and Monika Woloszyn. Common exercises in whole building ham modelling. 07 2009.

[20] John Straube. Moisture in buildings. *ASHRAE Journal*, 44:15–19, 01 2002.

[21] Synapse Concept. https://www.synapse-concept.com/.

[22] the Engineering ToolBox. Water - Specific Heat. https://www.engineeringtoolbox.com/specific-heat-capacity-water-d_660.html. Consulted in May 2020.