UMONS
Université de Mons

Faculté Polytechnique

The use of heat transfer functions for heat flow computation through multilayer walls

Master Thesis

Xavier Botey i Bassols

POLYTECH MONS

Under the supervision of teachers
Mr. Eric DUMONT
Mr. Renato LEPORE

January 2013

ACADÉMIE UNIVERSITAIRE WALLONIE-BRUXELLES

INDEX

INTRODUCTION 1

CHAPTER I: Heat conduction in solids 3

CHAPTER II: One-layer wall 15
- II.1. Coefficients for one-layer wall 17
- II.2. Analytical solution for case 1 – step 20
- II.3. Results for case 1 – step 22
  - II.3.1. The heat transfer error 23
- II.4. Analytical solution for case 2 – cosine 26
- II.5. Results for case 2 – cosine 27
  - II.5.1. The heat transfer error 29
  - II.5.2. The shift of the maximum 31

CHAPTER III: Multilayer wall 33
- III.1. Results for standard wall 33
- III.2. Results for an homogeneous multilayer wall 36
  - III.2.1. The heat flux through an homogeneous multilayer wall 38
- III.3. Results for a house 41

CONCLUSIONS 47

NOMENCLATURE 48

BIBLIOGRAPHY 50

APPENDIX A: Pipes (1957)

APPENDIX B: The z-transform

APPENDIX C: Demonstration $cosh\left(\sqrt{-s}\right) = cos(\sqrt{\psi})$

APPENDIX D: Matlab code

INTRODUCTION

Nowadays, the computing of the heat transfer through a wall is really important in order to study the thermal insulation capacity of different materials. Being able to obtain and to keep specific thermal conditions inside a building being as energy-efficient as possible is one of the main goals of building construction. In 1933, Nessi and Nissolle [1] ran an investigation which led to the discovery of a way to calculate this heat transfer. But their formula had a very complicated final solution, which had to be solved by hand and took a lot of time.

In 1957, Pipes [2] developed a way which uses a matrix that relates the heat flux and the temperature of the two boundary surfaces of a multilayer wall. To make it possible is necessary to work in the Laplace space, so the main problem is to find the final solution in the real space.

The evolution of machine computation gave scientists the chance to discover new methods to give a solution to the same problem in a more accurate way and using a smaller period of time. A z-transform of the heat transmission matrix is necessary to reach a final solution involving the current and previous temperatures and the heat fluxes through façades and roofs. Mitalas and Stepherson [3] developed a numerical method to compute the heat flux and used to the z-transform to solve the heat transfer matrix developed by Pipes [2]. This is the method implemented in the building energy software TRNSYS.

The TRNSYS software can compute solution for a minimum sampling period of 10-15 minutes, this time can represent a problem for detailed control studies where a time-step in the order of 1 minute is required.

The current project has three goals:

- The understanding of the heat transfer function method and the derivation of the equations used to compute the associated heat transfer coefficients.
- The comparison of the solutions obtained by this method with the analytical solutions. This will be only done for monolayer walls. In the case of multilayer walls, the solutions will be compared with the solutions obtained with TRNSYS.
- The discussion of the advantages and drawbacks of the heat transfer function method.

It is composed of three chapters:

- Chapter I: Heat conduction in solids. It exposes the theory needed to understand the heating and/or the cooling loads of a building and the algorithm used to obtain the heat transfer coefficients.
- Chapter II: One-layer wall. It introduces the results obtained for a one-layer wall. These results are compared with the analytical solution from the Fourier equation for different boundary conditions.
- Chapter III. Multilayer wall. It shows the results obtained for a multilayer wall. In this case, we do not have an analytical solution to compare with, so the results are compared with the solutions obtained with TRNSYS. The final part of this chapter is an evaluation of the heating and cooling needs of a house.

As software we use *Matlab* [4] to program the different algorithms. There are two files:

- *1_layer*: calculates the coefficients for one-layer walls in Chapter II.
- *multilayer*: computes the coefficient for multilayer walls in Chapter III.

(See APPENDIX D for Matlab code)

All the results are plotted and compared using *Microsoft Excel.*

CHAPTER I: HEAT CONDUCTION IN SOLIDS

Aim: Computation of heat transfer through walls, in order to determine the heating and/or cooling loads of a building.

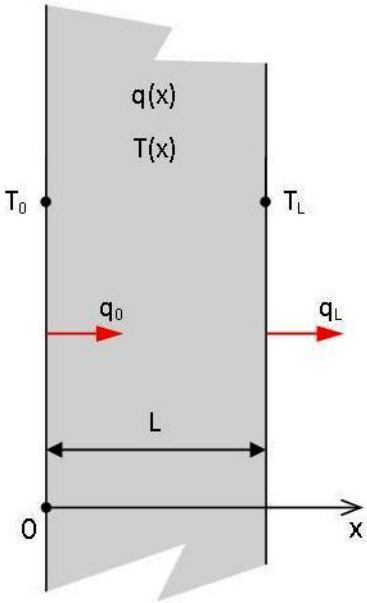


Figure I.1. Fourier's law

Walls are assumed infinite in two directions, so that the properties may way in one direction, along its thickness.

With T(x), the temperature profile and q(x), the heat flux [W/m^2] along the thickness of the wall.

Heat conduction through the wall is governed by two laws:

1) Fourier's law

$$q = -k \cdot \overline{\nabla} T \qquad (I.1)$$

Relates the heat fluxes to temperature gradient

2) 1st law of Thermodynamics (energy conservation)

$$\rho \cdot C_p \cdot \frac{\partial T}{\partial t} = -\overline{\nabla} q \qquad (I.2)$$

These laws need the thermal properties of the wall:

- k(x): thermal conductivity [kJ/(h·m·K)]
- ρ(x): density [kg/m^3]
- C_p(x): specific heat [kJ/(kg·K)]

Solving Equation I.1 and I.2 can be performed by using numerical methods (FEM, FUM,…) which can be slow.

For heat load computation, we do not need the whole flux and temperature profile along the wall T(x) and q(x), but only the values at x=0 and x=L, that is at the boundaries of the wall.

Pipes (1957) [2] solved this problem, which is analogue to an electrical quadruple: how to relate the values at one boundary T_0, q_0 to the values at the second boundary T_L, q_L.

By defining

$$R = \frac{1}{k}$$

$$C = \rho \cdot C_p$$

Equation I.1 and I.2 for a one-dimension wall become:

$$-\frac{\partial T}{\partial x} = R \cdot q \qquad (I.3)$$

$$-\frac{\partial q}{\partial x} = C\frac{\partial T}{\partial t} \qquad (I.4)$$

By taking the Laplace transform of Equation I.3 and I.4, we obtain:

$$-\frac{\partial T(s)}{\partial x} = R \cdot q(s) \qquad (I.5)$$

$$-\frac{\partial q(s)}{\partial x} = C \cdot s \cdot T(s) \qquad (I.6)$$

By solving Equation I.5 and I.6 and writing the particular solutions for x=0 and x=L, we can rewrite the system of equations in a matrix form,

$$\begin{bmatrix} T_0(s) \\ q_0(s) \end{bmatrix} = \begin{bmatrix} \cosh\,(L\sqrt{R\ C\ s}) & \sqrt{\frac{R}{C\ s}}\sinh\,(L\sqrt{R\ C\ s}) \\ \frac{\sinh\,(L\sqrt{R\ C\ s})}{\sqrt{\frac{R}{C\ s}}} & \cosh\,(L\sqrt{R\ C\ s}) \end{bmatrix} \cdot \begin{bmatrix} T_L(s) \\ q_L(s) \end{bmatrix}$$

(See APPENDIX A for the demonstration)

The matrix is called the transmission matrix, H(s). This transmission matrix is valid for a homogeneous layer of constant thermal properties. In the case of a multilayer wall, the overall transmission matrix is the product of the transmission matrices of each layer:

$$H = H_1 \cdot H_2 \cdot H_3 \cdot \ldots \cdot H_n$$

If a layer has a negligible specific heat C_{pj}, its transmission matrix becomes:

$$H(s) = \begin{bmatrix} 1 & \frac{L}{k} \\ 0 & 1 \end{bmatrix}$$

STEPHERSON AND MITALAS (1971)

They use the transfer matrix defined by Pipes (1957) [2],

$$\begin{bmatrix} T_0(s) \\ q_0(s) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \cdot \begin{bmatrix} T_L(s) \\ q_L(s) \end{bmatrix} \qquad (I.7)$$

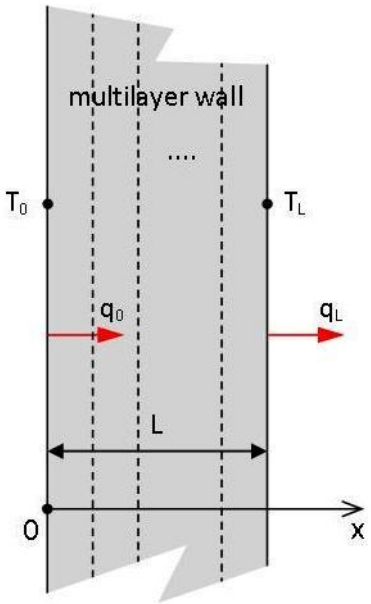


Figure I.2. Heat transfer through a multilayer wall

Where $H = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$, the overall transfer matrix.

H is the product of the transfer matrices of each layer which is part of the wall (M layers):

$$H = \prod_{j=1}^{M} H_j \qquad with \quad H_j = \begin{bmatrix} A_j & B_j \\ C_j & D_j \end{bmatrix}$$

In Pipes, H_j is written as:

$$H_j = \begin{bmatrix} \cosh(\theta_j) & Z_{0j} \cdot \sinh(\theta_j) \\ \dfrac{\sinh(\theta_j)}{Z_{0j}} & \cosh(\theta_j) \end{bmatrix} \qquad (I.8a)$$

With

- $\theta_j = L_j \cdot \sqrt{R_j \cdot C_j \cdot s}$
- $Z_{0j} = \sqrt{\frac{R_j}{C_j \cdot s}}$
- $C_j = \rho_j \cdot C_{pj}$
- $R_j = \frac{1}{k_j}$

and

- ρ_j = density of layer j
- C_{pj} = specific heat of layer j
- k_j = thermal conductivity of layer j
- L_j = thickness of layer j

In Stephenson and Mitalas [3], H_j is written as:

$$H_j = \begin{bmatrix} \cosh\left(\sqrt{\tau_j s}\right) & \dfrac{R_j \cdot sinh\left(\sqrt{\tau_j s}\right)}{\sqrt{\tau_j s}} \\ \dfrac{\sqrt{\tau_j s} \cdot sinh\left(\sqrt{\tau_j s}\right)}{R_j} & \cosh\left(\sqrt{\tau_j s}\right) \end{bmatrix} \qquad (I.8b)$$

With

- $\tau_j = \frac{{L_j}^2}{\alpha_j}$
- $\alpha_j = \frac{k_j}{\rho_j \cdot C_{pj}}$ =diffusivity of layer j
- $R_j = \frac{L_j}{k_j}$ = thermal resistance of layer j

A third way of writing H_j is:

$$H_j = \begin{bmatrix} \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) & \frac{sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right)}{k_i \cdot \sqrt{\frac{s}{\alpha_j}}} \\ k_j \cdot \sqrt{\frac{s}{\alpha_j}} \cdot sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) & \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) \end{bmatrix} \qquad (I.8c)$$

With

- L_j = thickness of layer j
- α_j = diffusivity of layer j

A forth way of writing H_j is:

$$H_j = \begin{bmatrix} \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) & \frac{R_j \cdot sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right)}{L_j \cdot \sqrt{\frac{s}{\alpha_j}}} \\ \frac{L_j \cdot \sqrt{\frac{s}{\alpha_j}} \cdot sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right)}{R_j} & \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) \end{bmatrix} \qquad (I.8d)$$

With

- L_j = thickness of layer j
- $\alpha_j = \frac{k_j}{\rho_j \cdot C_{pj}}$ = diffusivity of layer j
- $R_j = \frac{L_j}{k_j}$ = thermal resistance of layer j

This way is the easiest form to use, especially for layers with a negligible specific heat $\rho_j \cdot C_{pj} \cong 0$. In this case H_j is:

$$H_j = \begin{bmatrix} 1 & R_j \\ 0 & 1 \end{bmatrix}$$

We can see that the determinant of the matrix H_j is equal to 1, for example from Equation I.8d:

$$det\, H_j = A_j \cdot D_j - B_j \cdot C_j = cosh^2\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) - sinh^2\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) = 1$$

That means the determinant of the overall matrix H is also equal to 1, $det\, H = 1$.

We can therefore write other matrices which relate other boundary conditions to each other:

$$\begin{bmatrix} q_0(s) \\ q_L(s) \end{bmatrix} = \begin{bmatrix} \frac{D}{B} & -\frac{1}{B} \\ \frac{1}{B} & -\frac{A}{B} \end{bmatrix} \cdot \begin{bmatrix} T_0(s) \\ T_L(s) \end{bmatrix} \qquad (I.9a)$$

$$\begin{bmatrix} T_0(s) \\ T_L(s) \end{bmatrix} = \begin{bmatrix} \frac{A}{C} & -\frac{1}{C} \\ \frac{1}{C} & -\frac{D}{C} \end{bmatrix} \cdot \begin{bmatrix} q_0(s) \\ q_L(s) \end{bmatrix} \qquad (I.9b)$$

$$\begin{bmatrix} q_0(s) \\ T_L(s) \end{bmatrix} = \begin{bmatrix} \frac{C}{A} & \frac{1}{A} \\ \frac{1}{A} & -\frac{B}{A} \end{bmatrix} \cdot \begin{bmatrix} T_0(s) \\ q_L(s) \end{bmatrix} \qquad (I.9c)$$

$$\begin{bmatrix} T_0(s) \\ q_L(s) \end{bmatrix} = \begin{bmatrix} \frac{B}{D} & \frac{1}{D} \\ \frac{1}{D} & -\frac{C}{D} \end{bmatrix} \cdot \begin{bmatrix} q_0(s) \\ T_L(s) \end{bmatrix} \qquad (I.9d)$$

These new matrices cannot be obtained by the product of the equivalent matrix of each layer. Only the H_j matrix ensues that you have continuity of T_j and q_j between two adjacent layers, because this is the sole matrix which relates values of one side of a layer to values of the other side of the layer.

We just use the Equation I.3a because we want to relate the heat fluxes q_0 and q_L to the temperatures T_0 and T_L:

$$q_0(s) = \frac{D(s)}{B(s)} \cdot T_0(s) - \frac{1}{B(s)} \cdot T_L(s) \qquad (I.10)$$

$$q_L(s) = \frac{1}{B(s)} \cdot T_0(s) - \frac{A(s)}{B(s)} \cdot T_L(s) \qquad (I.11)$$

In matrix form:

$$\begin{bmatrix} q_0(s) \\ q_L(s) \end{bmatrix} = \begin{bmatrix} \frac{D(s)}{B(s)} & -\frac{1}{B(s)} \\ \frac{1}{B(s)} & -\frac{A(s)}{B(s)} \end{bmatrix} \cdot \begin{bmatrix} T_0(s) \\ T_L(s) \end{bmatrix} \qquad (I.12)$$

We call this matrix G(s) with:

$$G_{11}(s) = \frac{D(s)}{B(s)} \qquad G_{12}(s) = -\frac{1}{B(s)}$$

$$G_{21}(s) = \frac{1}{B(s)} \qquad G_{22}(s) = -\frac{A(s)}{B(s)}$$

To obtain the relation between the flux q_0 and q_L in function of T_0 and T_L in real space that means in function of time, we need to invert G(s).

Mitalas and Stepherson [3] noted that the temperatures T_0 and T_L are not known at every time t but at discrete time, usually separated by a constant time interval Δ (most of the time Δ=1h). In that case, it is more appropriate to use the z-transform instead of the Laplace transform, and we can have G(z) instead of G(s).

The genius idea of Mitalas is the assumption that each transfer function can be written as a quotient of polynomials:

$$\frac{A(z)}{B(z)} = \frac{Num_A(z)}{Den(z)} \tag{I.13}$$

$$\frac{D(z)}{B(z)} = \frac{Num_B(z)}{Den(z)} \tag{I.14}$$

$$\frac{1}{B(z)} = \frac{1}{Den(z)} \tag{I.15}$$

With

- $Num_A(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}$ $(I.16)$
- $Den(z) = d_0 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_p z^{-p}$ $(I.17)$

As $Num_A(z)$, $Num_B(z)$ and Den(z) are polynomials, they are easily inverted. For example:

$$q_L(z) = \frac{A(z)}{B(z)} \cdot T_L(z) \tag{I.18}$$

$$q_L(t) = \left(\frac{A(z)}{B(z)}\right)^{-1} \cdot T_L(t) \tag{I.19}$$

$$q_L(t)d_0 + q_L(t-\Delta)d_1 + q_L(t-2\Delta)d_2 + \cdots + q_L(t-p\Delta)d_p = \\ = T_L(t)a_0 + T_L(t-\Delta)a_1 + T_L(t-2\Delta)a_2 + \cdots + T_L(t-N\Delta)a_N \tag{I.20}$$

$$q_L(t) = \{T_L(t)a_0 + T_L(t-\Delta)a_1 + T_L(t-2\Delta)a_2 + \cdots + T_L(t-N\Delta)a_N$$
$$- [q_L(t-\Delta)d_1 + q_L(t-2\Delta)d_2 + \cdots + q_L(t-p\Delta)d_p]\} \cdot \frac{1}{d_0} \qquad (I.21)$$

That means that the heat flux $q_L(t)$ is computed with the N previous T_L temperatures and the *p* previous heat fluxes $q_L(t)$. The next chapters will show that in order to obtain a good accuracy, just a few coefficients a_n and b_n are needed (usually low than ten coefficients).

Considering the matrix G(z) as a transfer matrix, each element $G_{ij}(s)$ can be considered as a transfer function.

If one applies an input signal I(t) to the transfer function G_{ij}, we obtain its output signal O(t). By taking the Laplace transform and the z-transform of I(t) and O(t) we obtain:

$$G_{ij}(z) = \frac{O(z)}{I(z)} \qquad (I.22a)$$

$$G_{ij}(s) = \frac{O(s)}{I(s)} \qquad (I.22b)$$

Mitalas takes a ramp as an input signal, $I(t) = t$.

Then,

$$I(s) = \frac{1}{s^2} \qquad (I.23)$$

For $G_{22}(s) = \frac{A(s)}{B(s)}$, we have:

$$O(s) = G_{22}(s) \cdot I(s) = \frac{A(s)}{B(s) \cdot s^2} \qquad (I.24)$$

O(s) can be write as a sum of rational fraction (G(s) is supposed a quotient of polynomials),

$$O(s) = \frac{A(s)}{B(s) \cdot s^2} = \frac{C_0}{s^2} + \frac{C_1}{s} + \sum_{n=1}^{\infty} \frac{e_n}{s + \beta_n} \qquad (I.25)$$

Where:

- β_n are the roots of B(s)=0. These roots are negative real values.
- C_0, C_1 and e_n are constants defined by:

$$C_0 = \left[\frac{A(s)}{B(s)}\right]_{s=0} \tag{I.26}$$

$$C_1 = \left[\frac{d}{ds}\left[\frac{A(s)}{B(s)}\right]\right]_{s=0} = \left[\frac{B(s)\cdot\frac{dA(s)}{ds} - A(s)\cdot\frac{dB(s)}{ds}}{B^2(s)}\right]_{s=0} \tag{I.27}$$

$$e_n = \left[\frac{A(s)}{s^2\frac{dB(s)}{ds}}\right]_{s=-\beta_N} \tag{I.28}$$

We can rewrite the Equation I.25 in z-transform,

$$O(z) = \frac{C_0\cdot\Delta}{z\cdot(1-z^{-1})^2} + \frac{C_1}{1-z^{-1}} + \sum_{n=1}^{\infty}\frac{e_n}{1-exp(-\beta_N\cdot\Delta)\cdot z^{-1}} \tag{I.29}$$

(See APPENDIX B for equivalence of Laplace transform and z-transform)

The denominator of O(z) is:

$$Den\ O(z) = z\cdot(1-z^{-1})^2\cdot\prod_{n=1}^{\infty}[1-exp(-\beta_N\cdot\Delta)\cdot z^{-1}] \tag{I.30}$$

The denominator of $G_{22}(z)$ is:

$$Den\ G_{22}(z) = \prod_{n=1}^{\infty}[1-exp(-\beta_N\cdot\Delta)\cdot z^{-1}] \tag{I.31}$$

We can identify Equation I.31 with Equation I.17:

$$\prod_{n=1}^{\infty}[1-exp(-\beta_N\cdot\Delta)\cdot z^{-1}] = d_0 + d_1z^{-1} + d_2z^{-2} + \cdots + d_pz^{-p} \tag{I.32}$$

Mitalas cut the infinity product from Equation I.32 to the factor $exp(-\beta_N\cdot\Delta)$ when $exp(-\beta_N\cdot\Delta) < 10^{-10}$

Let's suppose that O(z) can also be described by a polynomial. Equation I.29 becomes:

$$O(z) = \frac{C_0 \cdot \Delta}{z \cdot (1 - z^{-1})^2} + \frac{C_1}{1 - z^{-1}} + \sum_{n=1}^{\infty} \frac{e_n}{1 - exp(-\beta_N \cdot \Delta) \cdot z^{-1}}$$
$$= o_0 + o_1 z^{-1} + o_2 z^{-2} + \cdots + o_N z^{-N} \qquad (I.33)$$

We have then:

- $o_0 = C_1 + \sum_{i=1}^{N} e_i$
- $o_1 = C_1 + C_0 \cdot \Delta + \sum_{i=1}^{N} e_i \cdot exp\,(-\beta_i \cdot \Delta)$
- $o_2 = C_1 + 2 \cdot C_0 \cdot \Delta + \sum_{i=1}^{N} e_i \cdot exp\,(-2 \cdot \beta_i \cdot \Delta)$
- …
- $o_N = C_1 + N \cdot C_0 \cdot \Delta + \sum_{i=1}^{N} e_i \cdot exp\,(-N \cdot \beta_i \cdot \Delta)$ $\qquad (I.34)$

The numerator of $G_{22}(z)$ is obtained by using Equation I.13

$$Num\ G_{22}(z) = \frac{Den\ G_{22}(z)}{I(z)} \cdot O(z) =$$

$$= \frac{z \cdot (1 - z^{-1})^2}{\Delta} \prod_{n=1}^{\infty} [1 - exp(-\beta_N \cdot \Delta) \cdot z^{-1}] \cdot O(z) =$$

$$= \frac{z \cdot (1 - z^{-1})^2}{\Delta} \left(d_0 + d_1 z^{-1} + d_2 z^{-2} + \cdots + d_p z^{-p}\right) \cdot (o_0 + o_1 z^{-1} + o_2 z^{-2} + \cdots + o_N z^{-N})$$

$$= a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{2N+1} z^{-2N+1} \qquad (I.35)$$

The heat transfer matrix H need not to be evaluated for complex s numbers but only for s=0 and s=-β_N, the roots, which are negative real numbers.

In these cases each layer matrix H_j reads:

$$[H_j]_{s=-\psi} = \begin{bmatrix} \cos\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right) & \frac{R_j sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{L_j\sqrt{\frac{\psi}{\alpha_j}}} \\ -\frac{L_j\sqrt{\frac{\psi}{\alpha_j}}\, sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{R_j} & \cos\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right) \end{bmatrix} \tag{I.36}$$

(See APPENDIX C for the demonstration)

$$[H] = \prod_{j=1}^{M} H_j \tag{I.37}$$

$$\left[\frac{dH}{ds}\right] = \sum_{l=1}^{M}\left[\prod_{j=1}^{l-1}[H_j]\right]\cdot\left[\frac{d}{ds}H_l\right]\cdot\left[\prod_{j=l+1}^{M}[H_j]\right] \tag{I.38}$$

With,

$$\left[\frac{dH_j}{ds}\right]_{s=-\psi} = \frac{{L_j}^2}{\alpha_j}\begin{bmatrix} \frac{\sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{L_j\sqrt{\frac{\psi}{\alpha_j}}} & \frac{R_j{L_j}^2}{\alpha_j\psi}\left(\frac{\sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{L_j\sqrt{\frac{\psi}{\alpha_j}}} - cos\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)\right) \\ \frac{1}{R_j}\left(\frac{\sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{L_j\sqrt{\frac{\psi}{\alpha_j}}} + cos\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)\right) & \frac{\sin\left(L_j\sqrt{\frac{\psi}{\alpha_j}}\right)}{L_j\sqrt{\frac{\psi}{\alpha_j}}} \end{bmatrix} \tag{I.39}$$

$$\left[\frac{dH_j}{ds}\right]_{s=0} = \frac{{L_j}^2}{\alpha_j}\begin{bmatrix} 1 & \frac{R_j}{3} \\ \frac{2}{R_j} & 1 \end{bmatrix} \tag{I.40}$$

ALGORITHM

1) Find the roots of B(s)=0 with Equation I.35 and I.36 given the thickness L_j and the thermal properties k_j, ρ_j, Cp_j of the layer j.
2) Calculate the coefficients d_j of the denominator of the transfer function G(z) with Equation I.32.
3) Calculate the constants C_0, C_1 and e_n with Equation I.26, I.27 and I.28.
4) Calculate the coefficients o_n of the output with Equation I.34
5) Calculate the coefficients a_n of the numerator of the transfer function G(z) with Equation I.35
6) With coefficients a_n and d_n, the heat flux $q_L(t)$ can be evaluated at time step Δ, Equation I.21

REMARKS

1) The matrix G(z) has four elements:

$$G_{11}(z) = \frac{D(z)}{B(z)} \qquad (I.41)$$

$$G_{12}(z) = -\frac{1}{B(z)} \qquad (I.42)$$

$$G_{21}(z) = \frac{1}{B(z)} \qquad (I.43)$$

$$G_{22}(z) = -\frac{A(z)}{B(z)} \qquad (I.44)$$

We have provided the algorithm to compute the coefficients a_n and d_n which relate $q_L(t)$ to $T_L(t)$.

We can compute the coefficients which relate:

- $q_0(t)$ to $T_0(t)$
- $q_0(t)$ to $T_L(t)$
- $q_L(t)$ to $T_0(t)$

Equation I.42 and I.43 are opposite, so that the coefficients of the numerator are the same and the algorithm described also allow to obtain the coefficients b_n.

Equation I.41 gives the coefficients c_n.

Equation I.41 to I.44 have the same denominator, so that the coefficients d_n are the same for every $G_{ii}(z)$.

We then have:

$$q_L(t) = \sum_{i=0}^{N_a} a_i \cdot T_L(t - i\Delta) - \sum_{j=0}^{N_b} b_j \cdot T_0(t - j\Delta) - \sum_{h=1}^{N_d} d_i \cdot q_L(t - h\Delta) \qquad (I.45)$$

$$q_0(t) = \sum_{i=0}^{N_b} b_i \cdot T_L(t - i\Delta) - \sum_{j=0}^{N_c} c_j \cdot T_0(t - j\Delta) - \sum_{h=1}^{N_d} d_i \cdot q_0(t - h\Delta) \qquad (I.46)$$

2) The coefficients a_n, b_n, c_n and d_n depend on the sampled period Δ. The smaller the Δ, the larger the number of coefficients needed to calculate the heat fluxes.
3) Thermal properties are constant.

CHAPTER II: ONE-LAYER WALL

The following chapter introduces the results obtained for a wall composed of a simple, homogeneous layer of material. In order to compare the heat transfer calculated with the heat transfer coefficients, we will present the heat transfer calculated by the analytical solution of the Fourier equation.

The analytical solution is only tractable for simple configuration, for example, one-layer wall and simple boundary conditions. This is the main reason of the interest of one-layer walls.

Two different cases will be considered, each one with different boundary conditions:

- Case 1: the temperatures in x=0 and x=L are subject to a step, from 20 ºC to 0 ºC. This scenario tests the method against a large spectrum of frequencies.
- Case 2: in this scenario the temperature in x=0 is a cosine wave and the inside temperature is constant and equal to 0 ºC. This scenario is more realistic, as the temperature in x=L is usually constant and the temperature in x=0 is a function of the time.

For each case, two different materials have been used:

- Material 1 (heavy density wall)
  - Wall thickness, L = 0,203 m
  - Thermal conductivity, k = 7,02 kJ/(h·m·K)
  - Density, ρ = 2240 kg/m^3
  - Specific heat, C_p = 0,9 kJ/(kg·K)
- Material 2 (light density wall)
  - Wall thickness, L = 0,076 m
  - Thermal conductivity, k = 0,108 kJ/(h·m·K)
  - Density, ρ = 43 kg/m^3
  - Specific heat, C_p = 1,21 kJ/(kg·K)

In the case of a one layer wall, the procedure described in Chapter I to get the heat transfer coefficients is simpler:

- No matrix product is needed so that the overall transfer matrix G is the layer transfer matrix H.
- The elements of the overall transfer matrix are simple explicit algebraic expressions of the thermal properties of the material of the layer.
- The numerical reach for the roots of element B(s) can be avoided because these roots can be obtained analytically.

$$G = \begin{bmatrix} \frac{D}{B} & -\frac{1}{B} \\ \frac{1}{B} & -\frac{A}{B} \end{bmatrix} \qquad (II.1)$$

With,

$$A(s) = D(s) = cosh\left(L \cdot \sqrt{\frac{s}{\alpha}}\right)$$

$$B(s) = \frac{R \cdot sinh\left(L \cdot \sqrt{\frac{s}{\alpha}}\right)}{L \cdot \sqrt{\frac{s}{\alpha}}} \qquad (II.2)$$

Where,

- $\alpha =$ Thermal diffusivity
- $R =$ Thermal resistance
- $L =$ Thickness of the wall

The roots of B(s)=0 are:

$$s = -\frac{n^2 \cdot \pi^2 \cdot \alpha}{L^2} \qquad (II.3)$$

With *n* an integer value.

Also, for a one-layer wall:

$$C_0 = \left[\frac{A(s)}{B(s)}\right]_{s=0} = \frac{1}{R} \qquad (II.4)$$

$$C_1 = \left[\frac{d}{ds}\left[\frac{A(s)}{B(s)}\right]\right]_{s=0} = \left[\frac{B(s) \cdot \frac{dA(s)}{ds} - A(s) \cdot \frac{dB(s)}{ds}}{B^2(s)}\right]_{s=0}$$

$$= -\sum_{n=1}^{\infty} e_n = \frac{2\tau}{R\pi^2} \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\tau}{3R} \qquad (II.5)$$

$$e_n = \left[\frac{A(s)}{s^2 \frac{dB(s)}{ds}}\right]_{s=-\beta_N} = \left[\frac{A(s)}{s^2} \frac{2s}{RA(s)}\right]_{s=-\frac{n^2 \cdot \pi^2 \cdot \alpha}{L^2}} = -\frac{2\tau}{Rn^2\pi^2} \qquad (II.6)$$

With,

$$\tau = \frac{L^2}{\alpha}$$

II.1. COEFFICIENTS FOR ONE-LAYER WALL

For the heavy density wall coefficients a, b, c and d are listed in Table II.1.

	Coefficients for Δ=1h - Heavy density wall							
	TRNSYS				Matlab			
	a	b	c	d	a	b	c	d
1	1,3423604E+02	1,6590755E+00	1,3423604E+02	1,0000000E+00	1,3423604E+02	1,6590755E+00	1,3423604E+02	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-1,4171487E+02	-4,7045212E-01	-1,4171487E+02	1,2473125E+01	-1,4171487E+02	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	2,6852544E+01	1,5713158E-02	2,6852544E+01	4,6355157E+00	2,6852544E+01	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-5,1838363E-01	-8,5231741E-06	-5,1838363E-01	8,7786719E-02	-5,1838363E-01	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	2,0072352E-04		2,0072352E-04	2,6962784E-05	2,0072352E-04	1,3629712E-11
6					-2,5013531E-10	2,6839242E-11	-2,5013531E-10	
∑	1,8855530E+01	1,8855530E+01	1,8855530E+01	5,4525252E-01	1,8855530E+01	1,8855530E+01	1,8855530E+01	5,4525252E-01

Table II.1. Coefficients for Δ=1h – Heavy density wall

The coefficients calculated by TRNSYS are represented at Table II.2 for a time step Δ=1h. We can see that:

- The coefficients calculated with Matlab are the same as those obtained with TRNSYS.
- The number of significant coefficients is small (6 values for a, b and c; and 5 for d)

This explains the popularity of the heat transfer coefficient method to calculate the heat transfer in walls: one does only need very few coefficients.

We can perform the calculating of the coefficients for different time steps Δ=1h, 0’5h, 0’25h, 0’1h and 0’03h; they are presented in Table II.2.

We can see that the number of significant coefficients increases when the time step decreases. This means that the heat fluxes depend on a further part time when the time step decreases.

We can check the accuracy of the coefficients by computing the steady state heat transmission coefficient U:

$$U = \frac{\sum a}{\sum d} = \frac{\sum b}{\sum d} = \frac{\sum c}{\sum d}$$

They are presented in Table II.3 for the different time steps.

Actually, the steady-state heat transmission coefficient U for the heavy density wall is:

$$U = \frac{k}{L} = 34{,}5813\ kJ/(h\ m^2 K)$$

	Coefficients - Heavy density wall - Matlab							
	Δ=0,5h				Δ=0,25h			
	a	b	c	d	a	b	c	d
1	1,8983832E+02	7,0513957E-02	1,8983832E+02	1,0000000E+00	2,6847192E+02	1,4708951E-04	2,6847192E+02	1,0000000E+00
2	-2,7682077E+02	3,1156063E+00	-2,7682077E+02	-8,7240664E-01	-5,4410199E+02	1,4052180E-01	-5,4410199E+02	-1,4408761E+00
3	1,0632012E+02	5,1968468E+00	1,0632012E+02	1,4532062E-01	3,6231038E+02	1,2376999E+00	3,6231038E+02	6,0185863E-01
4	-1,0155193E+01	9,3204452E-01	-1,0155193E+01	-3,1020463E-03	-9,1413202E+01	1,3419953E+00	-9,1413202E+01	-7,6913160E-02
5	1,4823946E-01	1,5547791E-02	1,4823946E-01	3,7837373E-06	7,9008667E+00	2,5955510E-01	7,9008667E+00	2,3657378E-03
6	-1,3930883E-04	1,3530245E-05	-1,3930883E-04	-1,1072772E-10	-1,8007116E-01	8,6690705E-03	-1,8007116E-01	-1,1834431E-05
7	3,3325703E-09	2,8403670E-10	3,3325703E-09		7,2075387E-04	3,9797252E-05	7,2075387E-04	6,2052519E-09
8					-3,1750009E-07	1,7887461E-08	-3,1750009E-07	
∑	9,3305729E+00	9,3305729E+00	9,3305729E+00	2,6981571E-01	2,9886281E+00	2,9886281E+00	2,9886281E+00	8,6423290E-02
	Δ=0,1h				Δ=0,03h			
	a	b	c	d	a	b	c	d
1	4,2449138E+02	1,7053026E-12	4,2449138E+02	1,0000000E+00	7,7501183E+02	-7,7042279E-05	7,7501183E+02	1,0000000E+00
2	-1,3390924E+03	1,3934715E-05	-1,3390924E+03	-2,5687945E+00	-4,4087239E+03	5,4721461E-04	-4,4087239E+03	-5,1028024E+00
3	1,6439527E+03	3,3435218E-03	1,6439527E+03	2,4643700E+00	1,0990579E+04	-1,7329787E-03	1,0990579E+04	1,1288398E+01
4	-9,9757828E+02	4,4046383E-02	-9,9757828E+02	-1,1041438E+00	-1,5806230E+04	3,2283267E-03	-1,5806230E+04	-1,4224131E+01
5	3,1578433E+02	1,0994595E-01	3,1578433E+02	2,3835571E-01	1,4536653E+04	-3,9309050E-03	1,4536653E+04	1,1289622E+01
6	-5,1133509E+01	7,4447363E-02	-5,1133509E+01	-2,3553529E-02	-8,9636190E+03	3,2960824E-03	-8,9636190E+03	-5,9002204E+00
7	3,9528500E+00	1,5388317E-02	3,9528500E+00	9,5672488E-04	3,7835163E+03	-1,8872586E-03	3,7835163E+03	2,0655361E+00
8	-1,3040561E-01	9,7984771E-04	-1,3040561E-01	-1,3772902E-05	-1,0983867E+03	9,6666666E-04	-1,0983867E+03	-4,8453353E-01
9	1,5796465E-03	1,8170322E-05	1,5796465E-03	5,9219678E-08	2,1784220E+02	-2,9649866E-05	2,1784220E+02	7,5259797E-02
10	-5,8890622E-06	8,8786747E-08	-5,8890622E-06	-6,3643422E-11	-2,9046520E+01	1,8209499E-04	-2,9046520E+01	-7,5718961E-03
11	5,6102201E-09	9,9939640E-11	5,6102201E-09		2,5414386E+00	3,4195522E-05	2,5414386E+00	4,7853936E-04
12					-1,4134618E-01	7,0804232E-06	-1,4134618E-01	-1,8284735E-05
13					4,8070815E-03	4,6412912E-07	4,8070815E-03	4,0400769E-07
14					-9,5586566E-05	2,1211971E-08	-9,5586566E-05	-4,9139631E-09
15					1,0572656E-06	3,0999491E-10	1,0572656E-06	3,1218849E-11
16					-6,0773349E-09	3,4238865E-11	-6,0773349E-09	
∑	2,4818358E-01	2,4818358E-01	2,4818358E-01	7,1768186E-03	6,0431256E-04	6,0431260E-04	6,0431256E-04	1,7475136E-05

Table II.2. Coefficients for Δ=1h, 0'5h, 0'25h, 0'1h and 0'03h – Heavy density wall

	Real	Δ=1h	Δ=0,5h	Δ=0,25h	Δ=0,1h	Δ=0,03h	Δ=0,02h
U	34,5813	34,5813	34,5813	34,5813	34,5813	34,5813	34,5812
$\%U_{real}$	-	100	100	100	100	99,99	99,99

Table II.3. steady-state heat transmission coefficient U for Δ=1h, 0'5h, 0'25h, 0'1h, 0'03h and 0'02 – Heavy density wall

There exists a minimum value of the time step which guaranties accurate results: the smaller time step, the larger the number of coefficients.

The sum of *a* over the sum of *d* becomes different of the steady-state value because we sum a large positive and negative numbers which give a very little sum, and the number of significant decimals decreases. This is a numerical issue due to the limited number of decimals stored in the computer memory.

In our case, the minimum time step is Δ=0,02h if we want to keep a U value at 99% of the real U value. As we can see in Table II.4 for Δ=0,01h the results are wrong because the sum of *a* is different to the sum of *b*.

	Coefficients for Δ=0.01h - Heavy density wall - Matlab			
	a	b	c	d
1	1,3436234E+03	-2,8990688E-01	1,3436234E+03	1,0000000E+00
2	-1,3050924E+04	3,2172478E+00	-1,3050924E+04	-9,1261298E+00
3	5,8750329E+04	-1,6639921E+01	5,8750329E+04	3,8462714E+01
4	-1,6273589E+05	5,3274901E+01	-1,6273589E+05	-9,9356294E+01
5	3,1059438E+05	-1,1827970E+02	3,1059438E+05	1,7607520E+02
6	-4,3345746E+05	1,9333547E+02	-4,3345746E+05	-2,2705737E+02
7	4,5815648E+05	-2,4110443E+02	4,5815648E+05	2,2055889E+02
8	-3,7488992E+05	2,3459630E+02	-3,7488992E+05	-1,6484392E+02
9	2,4073067E+05	-1,8063926E+02	2,4073067E+05	9,6014938E+01
10	-1,2227242E+05	1,1101031E+02	-1,2227242E+05	-4,3885797E+01
11	4,9299685E+04	-5,4675594E+01	4,9299685E+04	1,5778027E+01
12	-1,5778564E+04	2,1596237E+01	-1,5778564E+04	-4,4551355E+00
13	3,9955168E+03	-6,8233046E+00	3,9955168E+03	9,8285931E-01
14	-7,9529622E+02	1,7144858E+00	-7,9529622E+02	-1,6788845E-01
15	1,2318861E+02	-3,3944954E-01	1,2318861E+02	2,1907658E-02
16	-1,4638582E+01	5,2251120E-02	-1,4638582E+01	-2,1423220E-03
17	1,3082991E+00	-6,1363663E-03	1,3082991E+00	1,5276375E-04
18	-8,5526830E-02	5,3535198E-04	-8,5526830E-02	-7,6293288E-06
19	3,9262390E-03	-3,3361567E-05	3,9262390E-03	2,5026958E-07
20	-1,1865469E-04	1,3968266E-06	-1,1865469E-04	
∑	-2,0571587E-06	6,0503257E-08	-2,0571587E-06	5,4821432E-09

Table II.3. Coefficients for Δ=0,01h – Heavy density wall

II.2. ANALYTICAL SOLUTION FOR CASE 1 - STEP

We present the analytical solution of temperature of a homogeneous wall submitted to a temperature step at both sides of the wall.

T(x)=T for t<0

T(0), T(1)=0 for t>0

Matthew J. Hancock [5] presents an analytical solution in dimensionless notation. He defines:

$x^* = \frac{x}{L}$ = dimensionless position in the wall

$t^* = \frac{t}{t_R}$ = dimensionless time

$T^* = \frac{T}{T_R}$ = dimensionless temperature

Where,

- L is take as the real thickness of the wall
- T_R is take as the real initial temperature of the wall at the two boundary before performing the step
- t_R is take so that $t_R = \frac{L^2}{\alpha}$ with $\alpha = \frac{k}{\rho \cdot C_p}$, the diffusivity of the material

So, the solution of temperature is:

$$T^*(x^*, t^*) = \frac{4{T_0}^*}{\pi} \sum_{n=1}^{\infty} \frac{\sin\left[(2n-1)\pi x^*\right]}{2n-1} \exp\left[-(2n-1)^2 \pi^2 t^*\right] \qquad (II.7)$$

With ${T_0}^* = \frac{T_0}{T_R}$ the dimensionless initial uniform temperature

Finally, the dimensional solution is:

$$T(x,t) = \frac{4T_0}{\pi} \sum_{n=1}^{\infty} \frac{1}{2n-1} \sin\left[(2n-1)\pi \frac{x}{L}\right] exp\left[-(2n-1)^2 \pi^2 \frac{\alpha t}{L^2}\right] \qquad (II.8)$$

When the temperatures are known, the calculations of the heat fluxes through the surfaces are:

$$q(x,t) = -k \frac{\partial T(x,t)}{\partial x} = -k \frac{4T_0}{L} \sum_{n=1}^{\infty} \cos\left[(2n-1)\pi \frac{x}{L}\right] exp\left[-(2n-1)^2 \pi^2 \frac{\alpha t}{L^2}\right] \qquad (II.9)$$

- For x=0:

$$q(0,t) = -k \frac{4T_0}{L} \sum_{n=1}^{\infty} exp\left[-(2n-1)^2 \pi^2 \frac{\alpha t}{L^2}\right] \qquad (II.10)$$

- For x=L:

$$q(L,t) = k\frac{4T_0}{L}\sum_{n=1}^{\infty} exp\left[-(2n-1)^2\pi^2\frac{\alpha t}{L^2}\right] \qquad (II.11)$$

To compute the total flux delivered by the wall to the surroundings we calculate the integral of the heat flux up to ∞. For x=L,

$$Q(L,t) = \int_0^{\infty} q(L,t)dt = \left[\sum_{n=1}^{\infty} -\frac{4kLT_0}{\pi^2\alpha(2n-1)^2} exp\left[-(2n-1)^2\pi^2\frac{\alpha t}{L^2}\right]\right]_0^{\infty}$$

$$= \sum_{n=1}^{\infty}\frac{4kLT_0}{\pi^2\alpha(2n-1)^2} = \frac{4kLT_0}{\pi^2\alpha}\sum_{n=1}^{\infty}\frac{1}{(2n-1)^2} = \frac{4kLT_0}{\pi^2\alpha}\frac{\pi^2}{8} = \frac{\rho C_p L T_0}{2} \qquad (II.12)$$

$\rho C_p L T_0$ is the heat released for a volume of L m^3 (surface area of 1 m^3) when it passes from T_0 to 0ºC. As the heat is released symmetrically at both sides, the total heat at one side is $\frac{1}{2}\rho C_p L T_0$.

II.3. RESULTS FOR CASE 1 - STEP

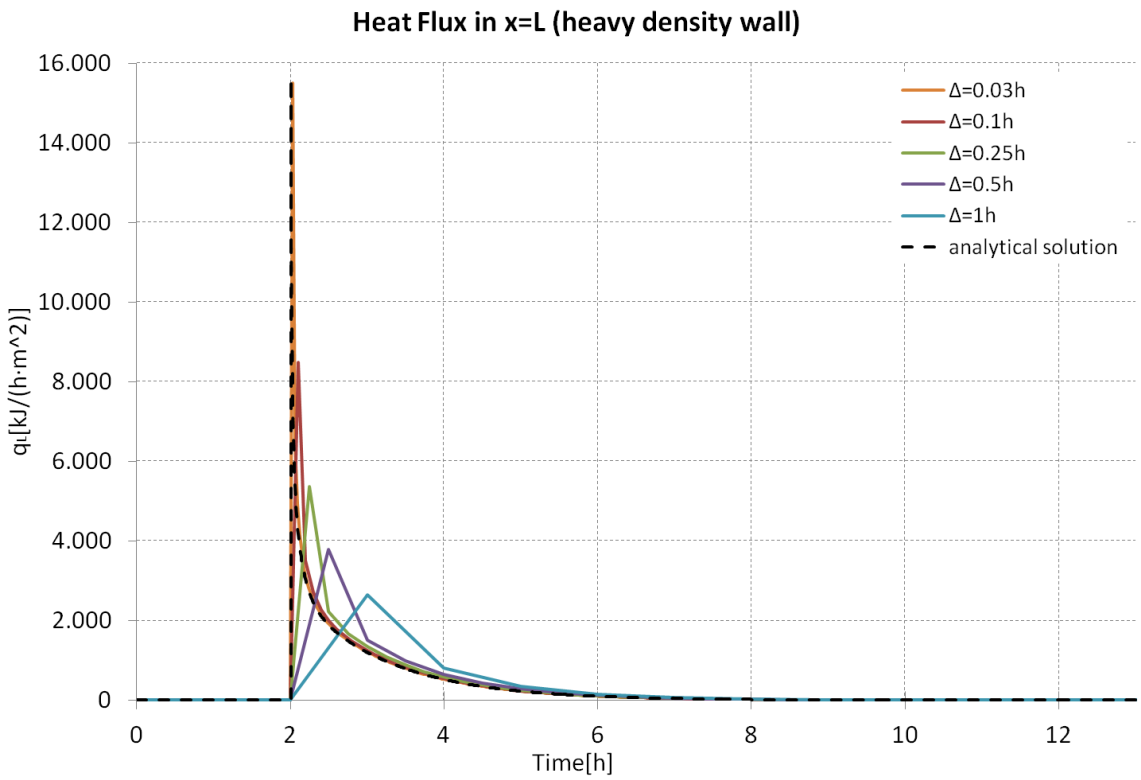


Figure II.1. Evolution of the heat flux in x=L for a heavy density wall – case 1

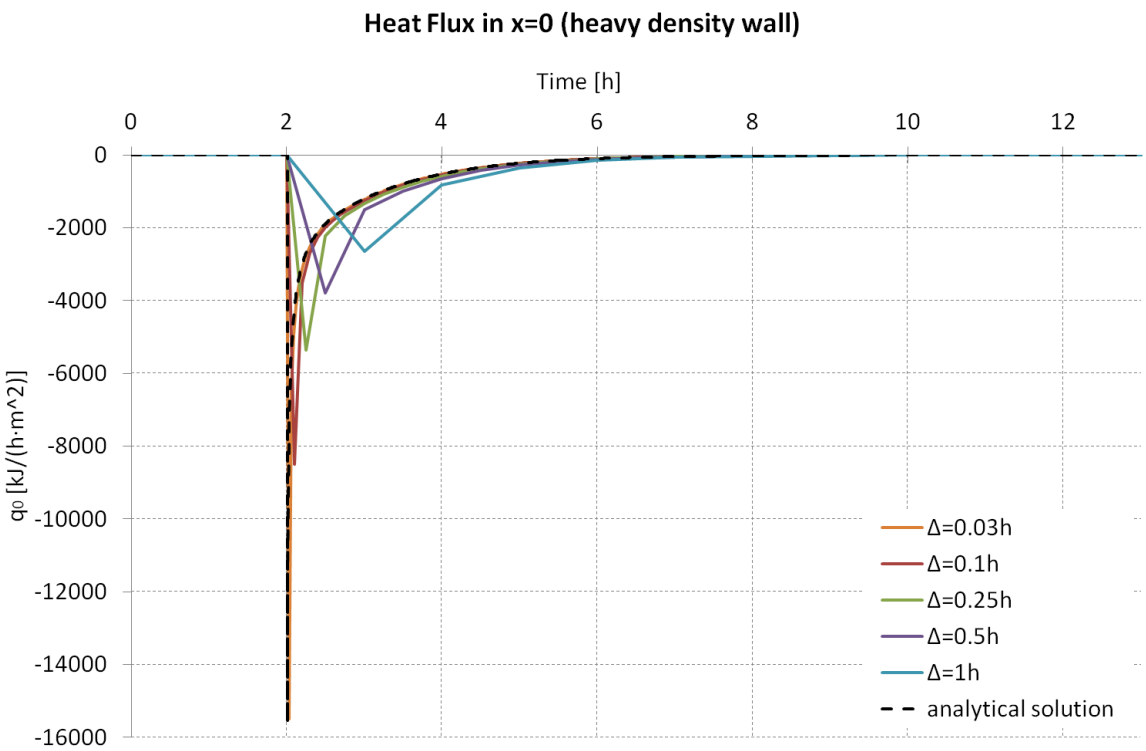


Figure II.2. Evolution of the heat flux in x=0 for a heavy density wall – case 1

Figures II.1 and II.2 present the heat flux at each side of the heavy density wall for different time steps and for the analytical solution. Figures II.4 and II.5 present the heat flux at each side of the light density wall for different time steps and for the analytical solution.

We observe that:

- Heat fluxes have the same value at both sides.
- The analytical solution gives an infinite flux just after the temperature step. The heat flux decreases exponentially to zero.

The value of peak and its position are different for different time steps.

This last remark can be explained by the fact that for discrete temperature function, the temperature step is actually a "ramp" as shown on Figure II.3. This is however the way TRNSYS simulates a temperature step.

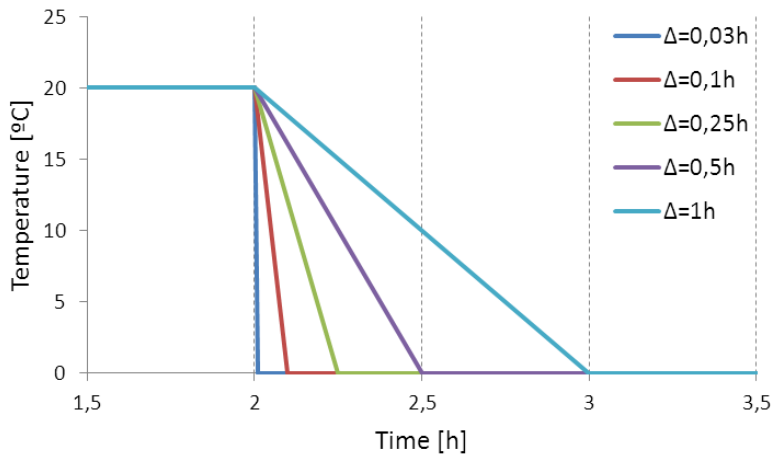


Figure II.3. Step input graph for each sampling time

It is therefore non useful to compare the shape of the heat flux curves presented in Figures II.1 II.2, II.4 and II.5 because the input signals are different.

However, we can see if the total heat released by the wall is the same for the various time steps.

II.3.1. THE HEAT TRANSFER ERROR

We calculate the energy transfer, for each curve we have to compute the integral of the heat flux. To do so, we have used a numerical method, specifically the mid-point rule that consists in multiplying each point by the corresponding sampling time. This method was chosen because it is the method used in TRNSYS. Table II.4 and II.5 show the results.

- For the heavy density wall

	analytical	Δ=0.03h	Δ=0.1h	Δ=0.25h	Δ=0.5h	Δ=1h
$\dot{q}_L$(kJ/m^2)	4092,480	4092,564	4092,475	4092,475	4092,475	4092,475
error (%)		2,0624E-03	1,2168E-04	1,2884E-04	1,2984E-04	1,297E-04

Table II.4. Heat transfer error for the heavy density material – case 1

- For the light density wall

	analytical	Δ=0.03h	Δ=0.1h	Δ=0.25h	Δ=0.5h	Δ=1h
$\dot{q}_L$ (kJ/m^2)	39,543	39,5428	39,5428	39,5428	39,5428	39,5428
error (%)		8,47E-11	1,83E-12	2,79E-10	8,31E-10	3,22E-13

Table II.5. Heat transfer error for the light density material – case 1

The results for each sampling time are quite the same and really close to the analytical solution.

For the numerical solution is impossible to perform a real step input, the Matlab software does not allow to establish two different temperatures for a single time value (Figure II.3).

However, the heat transfer through the boundary surfaces does not depend on how fast the temperature varies, it only depends on the initial and final values of the temperature, here $\Delta T = T_0 - T_L = T_0$ (Equation II.12). That is the reason why the errors between each sampling time solution and the analytical solution are so small.

The results in Figures II.1, II.2, II.4 and II.5 show a shift and a difference in the value of the maximum. Actually, the real value of that maximum must be infinity because the derivative of the step input in time t=2h is a vertical straight line but, as already stated, we cannot compute a real step input in Matlab. The displacement gets bigger with the increasing of the sampling time because the first solution point after the step is computed later when the sampling time is bigger.

We also notice that the values of heat fluxes for different types of material are different. This value is bigger for the heavy density material than for the light density one. Certainty, if we compare the thermal diffusivity α of each material we realize that they are quite the same, but those differences are caused by the thickness of each wall.

$$\alpha_H = \frac{k_H}{\rho_H C_{pH}} = 9{,}673 \cdot 10^{-7} m^2/s$$

$$\alpha_L = \frac{k_L}{\rho_L C_{pL}} = 5{,}766 \cdot 10^{-7} m^2/s$$

Furthermore, the thermal conductivity explains the time that is needed to reach the steady-state so that a material with a lower *k* value will reach earlier the steady-state. Notice that the heavy density material needs more than four hours to reach that state and the light density one less than two hours.

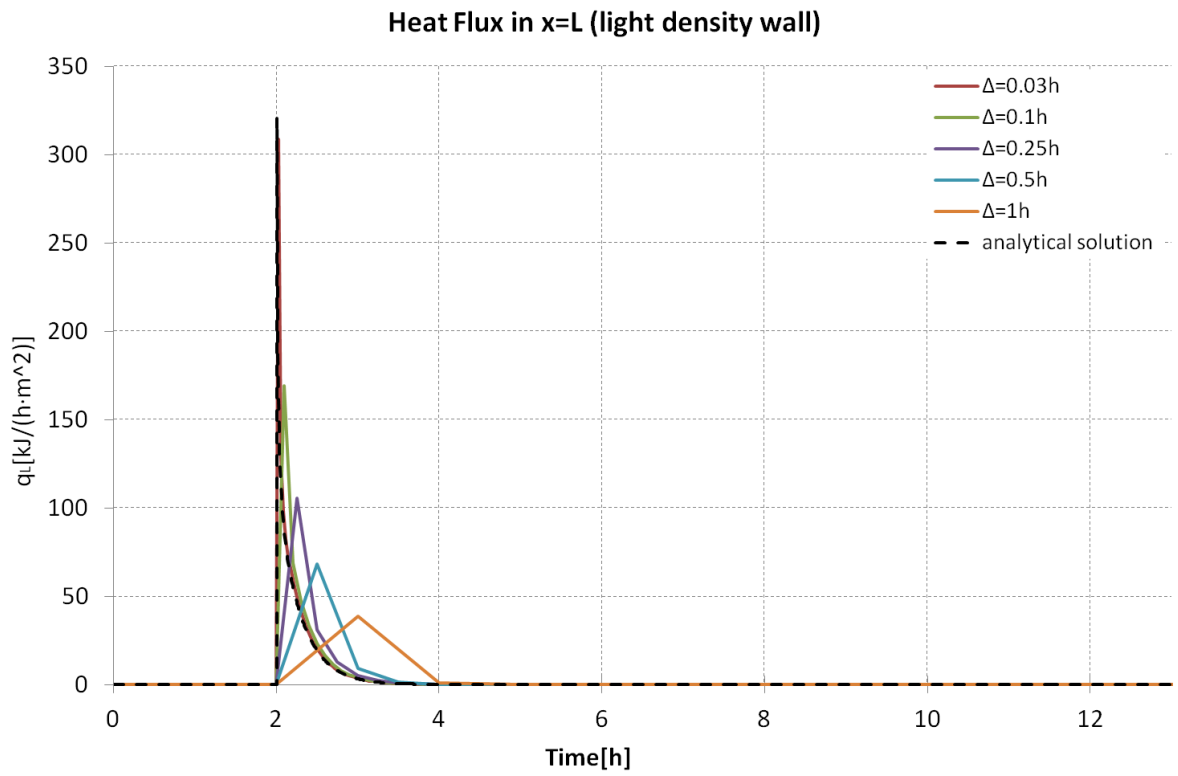


Figure II.4. Evolution of the heat flux in x=L for a light density wall – case 1

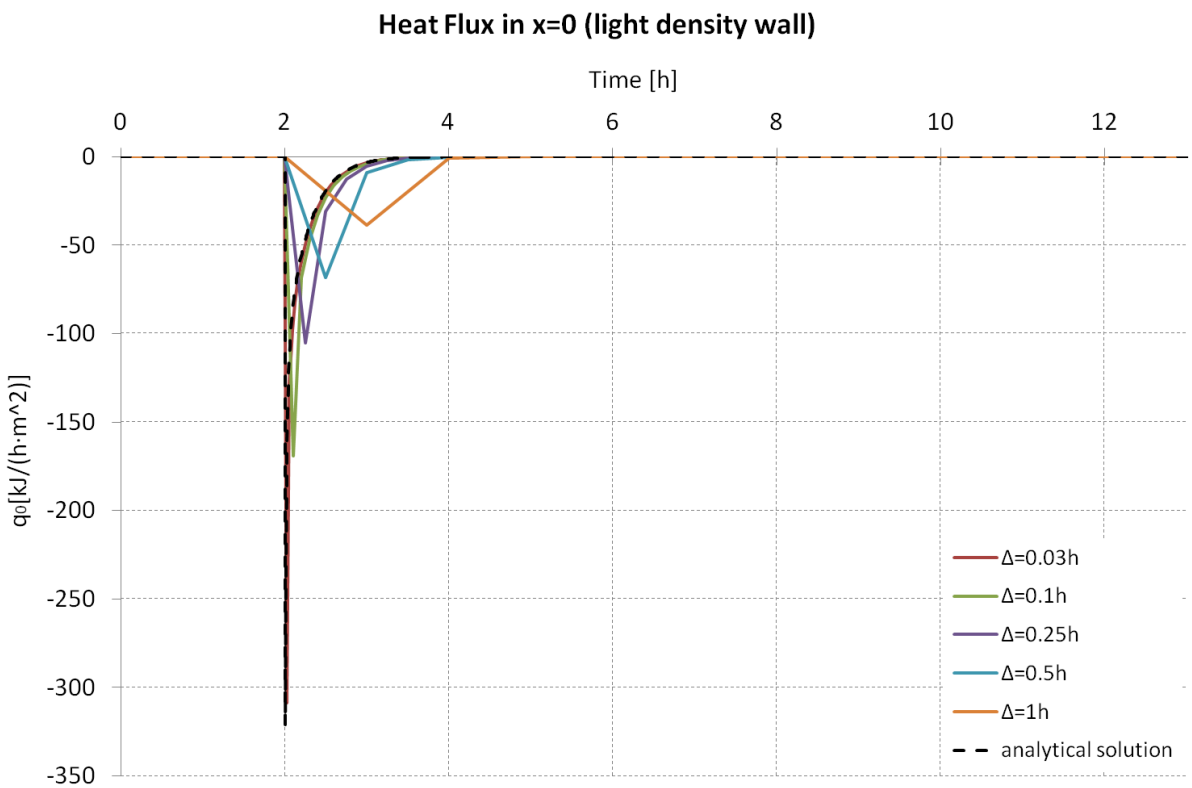


Figure II.5. Evolution of the heat flux in x=0 for a light density wall – case 1

II.4. ANALYTICAL SOLUTION FOR CASE 2 - COSINE

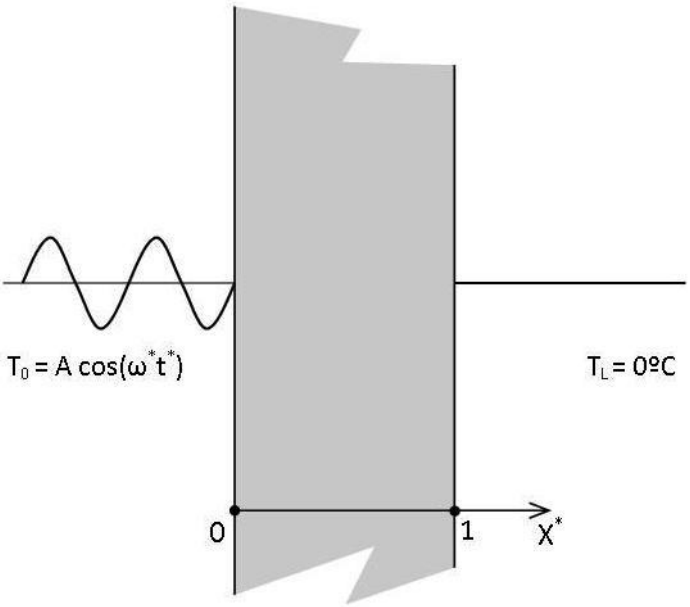


Figure II.6. Boundary conditions for case 2 - cosine

For a periodic boundary condition (Figure II.6), the analytical solution is presented by Matthew J. Hancock [5].

The dimensionless temperature $T^*(x^*, t^*)$ is the sum of a transient term and a "quasi steady-state" term. We are only interested in the quasi steady-state term, which is:

$$T^*(x^*, t^*) = Re\left\{\frac{exp\left(\sqrt{\frac{\omega^*}{2}}(1+i)(1-x^*)\right) - exp\left(-\sqrt{\frac{\omega^*}{2}}(1+i)(1-x^*)\right)}{exp\left(\sqrt{\frac{\omega^*}{2}}(1+i)\right) - exp\left(-\sqrt{\frac{\omega^*}{2}}(1+i)\right)} Aexp\,(\omega^* t^*)\right\} \tag{II.13}$$

Where the dimensionless period is:

$$\omega^* = \frac{2\pi}{\tau^*} \quad , \quad \tau^* = \frac{\tau}{t^*}$$

And A is the amplitude of cosine signal.

II.5. RESULTS FOR CASE 2 – COSINE

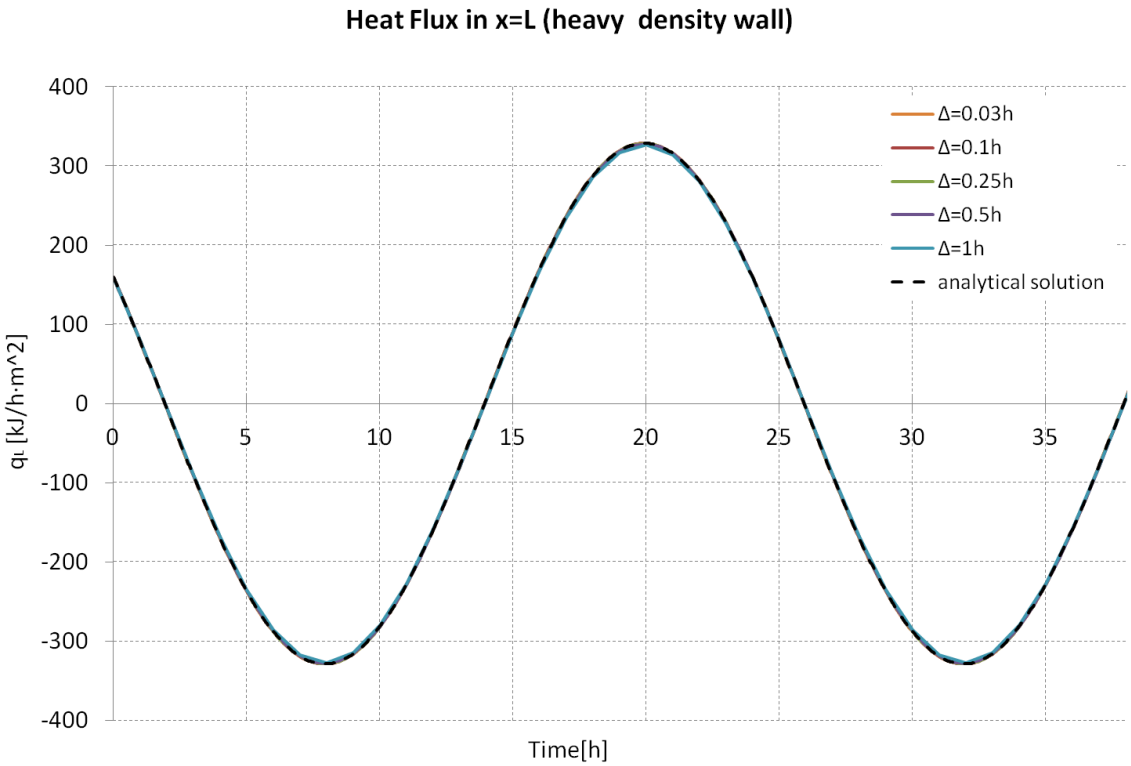


Figure II.7. Evolution of the heat flux in x=L for a heavy density wall – case 2

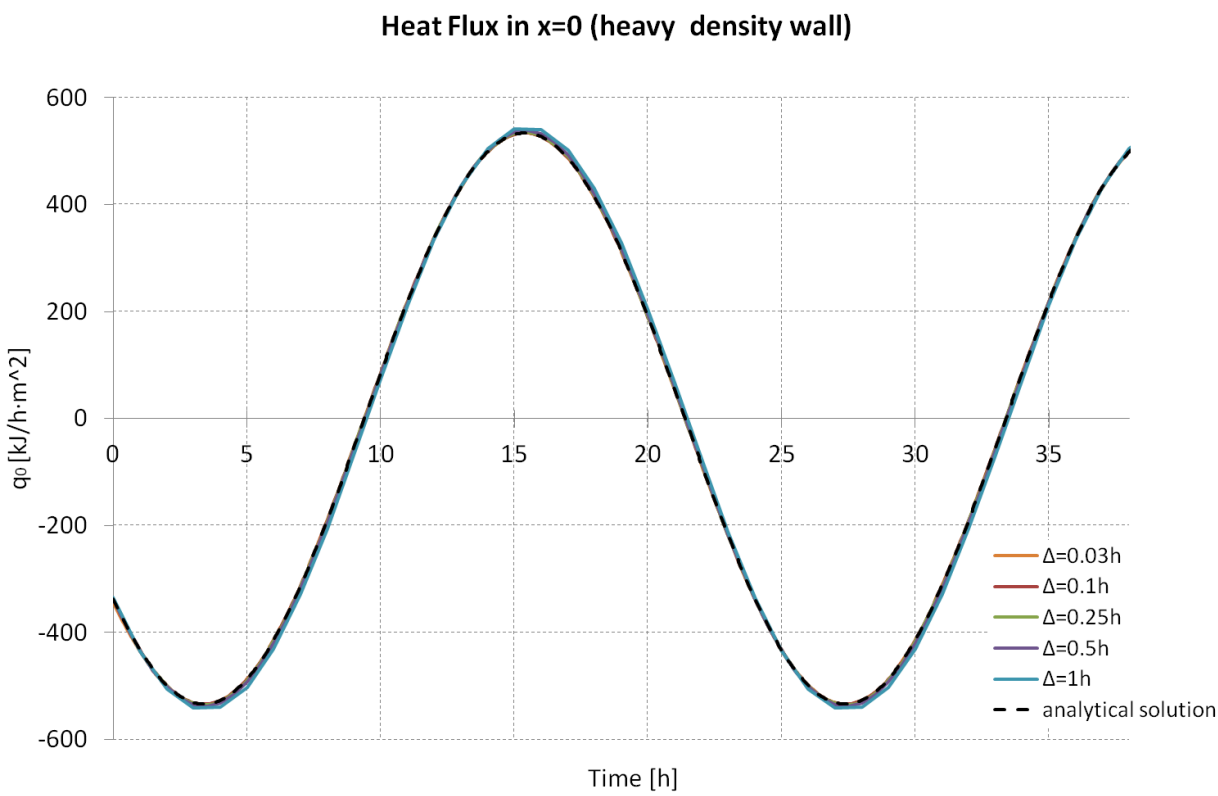


Figure II.8. Evolution of the heat flux in x=0 for a heavy density wall – case 2

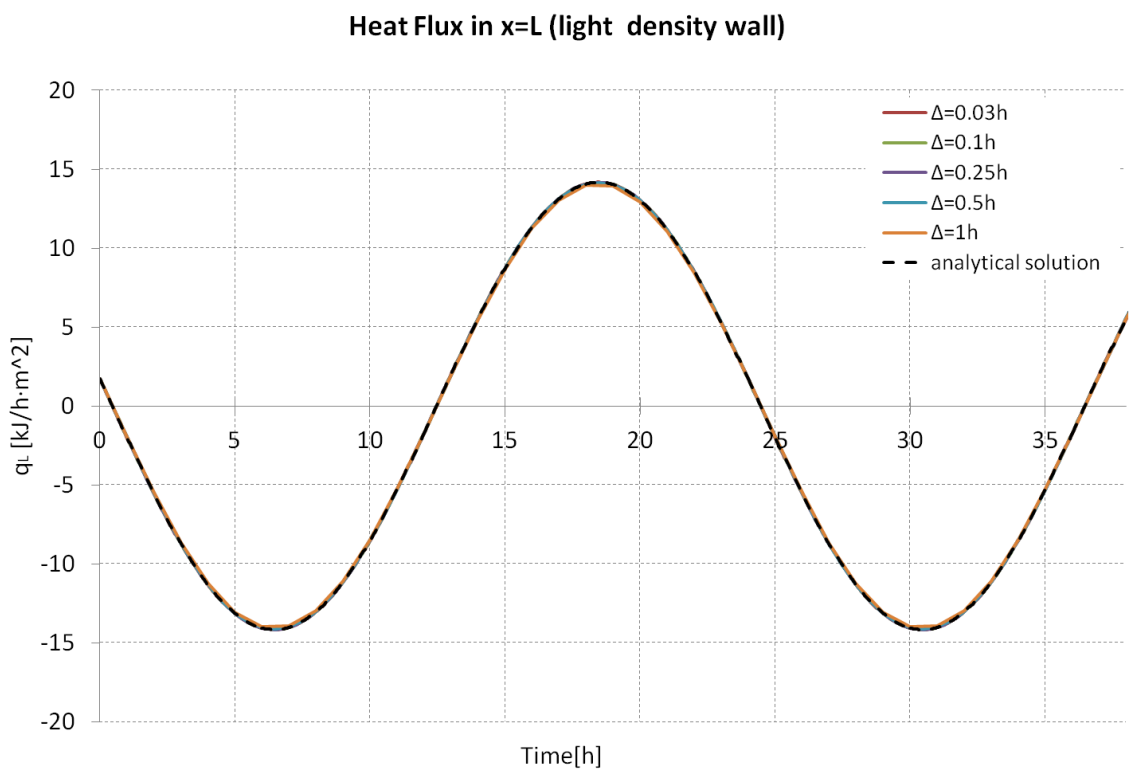


Figure II.9. Evolution of the heat flux in x=L for a light density wall – case 2

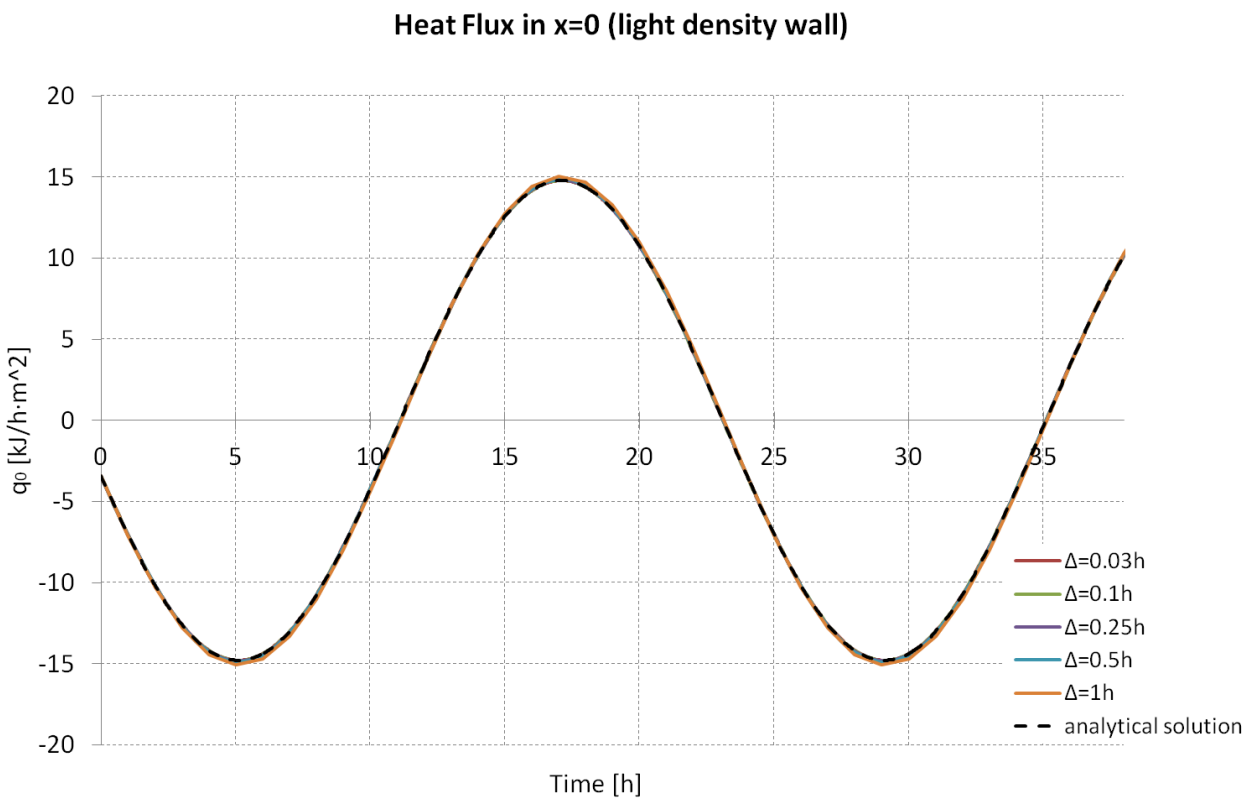


Figure II.10. Evolution of the heat flux in x=0 for a light density wall – case 2

Figures II.7, II.8, II.9 and II.10 show that the time step has quasi no influence on the shape of the heat flux and that every solution is very close to the analytical solution. There is however a small maximum shift for several cases (Figure II.11). We have used different cosine periods, from 0,5h to 24h to examine the accuracy of the solutions compared to the analytical solution.

II.5.1. THE HEAT TRANSFER ERROR

As in case 1, we compute the heat fluxes through a wall. We also use the mid-point rule to compute the total heat over one cosine period. Table II.6 and II.7 show the results for different sampling times and different cosine frequencies.

- For the heavy density material

$\dot{q}_L$ [kJ/m^2]	T=24h	T=12h	T=6h	T=3h	T=1h	T=0.5h
analytical	2510,369	1100,881	386,569	97,369	4,267	0,239
Δ=0.03h	2510,356	1100,862	386,542	97,340	4,254	0,237
Δ=0.1h	2510,312	1100,757	386,519	97,581	4,201	0,203
Δ=0.25h	2510,219	1100,255	385,416	97,124	3,837	0,002
Δ=0.5h	2511,475	1098,942	385,154	93,210	1,527	0
Δ=1h	2519,138	1104,075	371,837	49,352	0	0

Table II.6. Heat transfer values for the heavy density material – case 2

- For the light density material

$\dot{q}_L$ [kJ/m^2]	T=24h	T=12h	T=6h	T=3h	T=1h	T=0.5h
analytical	108,242	53,651	25,939	11,522	1,990	0,405
Δ=0.03h	108,241	53,650	25,936	11,518	1,984	0,400
Δ=0.1h	108,239	53,645	25,925	11,531	2,006	0,450
Δ=0.25h	108,252	53,668	25,954	11,474	1,810	0,250
Δ=0.5h	108,316	53,785	26,110	10,929	0,715	0
Δ=1h	107,827	52,820	25,810	5,797	0	0

Table II.7. Heat transfer values for the light density material – case 2

The ratio of the analytical solution and the numerical one gives the error.

- For the heavy density material

error [%]	T=24h	T=12h	T=6h	T=3h	T=1h	T=0.5h
Δ=0.03h	0,0005	0,0017	0,0071	0,0297	0,3040	1,0943
Δ=0.1h	0,0023	0,0113	0,0130	0,2176	1,5421	14,9551
Δ=0.25h	0,0060	0,0568	0,2983	0,2520	10,0695	99,0066
Δ=0.5h	0,0440	0,1762	0,3659	4,2718	64,2051	100
Δ=1h	0,3493	0,2901	3,8110	49,3146	100	100

Table II.8. Heat transfer error for the heavy density material – case 2

- For the light density material

error [%]	T=24h	T=12h	T=6h	T=3h	T=1h	T=0.5
Δ=0.03h	0,0005	0,0021	0,0082	0,0329	0,2957	1,1787
Δ=0.1h	0,0025	0,0107	0,0526	0,0778	0,7906	11,1764
Δ=0.25h	0,0089	0,0313	0,0596	0,4111	9,0716	38,3267
Δ=0.5h	0,0684	0,2506	0,6612	5,1448	64,0464	100
Δ=1h	0,3835	1,5488	0,4973	49,6867	100	100

Table II.9. Heat transfer error for the light density material – case 2

As we can see, the results of the heat transfer error are quite the same for both types of material. However, the behaviour of the error increase between each time step and period shows us some irregularity (Table II.8 and II.9). We know that those values have to increase with a bigger sampling time because we have less points. It has also to increase with the period because we have less solution points per period. In spite of it, in the Table II.8 we can see two cases where the error decreases (marked with yellow). Moreover, in the Table II.9 there is another case. We attribute those drops to the precision of the midpoint rule calculation.

We have also considered values of Δ and T not in agreement with the Shannon's law. Those values are: Δ=1h T=1h and Δ=0,5h T=0,5h. The solution for those values are Q_L=0 and 100% of error. The Shannon's law establishes that a function can be perfectly reconstructed from an infinite sequence of samples if the system frequency is no greater than half the sampling time frequency. So,

$$\omega_{system} \leq \frac{1}{2}\omega_{\Delta}$$

- For Δ=0,5h and T=1h,

$$\omega_{system} = \frac{2\,\pi}{T} = \frac{2\,\pi}{1} = 2\pi\ rad/h$$

$$\omega_{\Delta} = \frac{2\,\pi}{\Delta} = \frac{2\,\pi}{0{,}5} = 4\pi\ rad/h$$

$$\omega_{system} = \omega_{\Delta} \quad \rightarrow \quad Verifies\ the\ Shannon's\ law, limit\ case$$

- For Δ=0,5h and T=0'5h,

$$\omega_{system} = \frac{2\,\pi}{T} = \frac{2\,\pi}{0{,}5} = 4\pi\ rad/h$$

$$\omega_{\Delta} = \frac{2\,\pi}{\Delta} = \frac{2\,\pi}{0{,}5} = 4\pi\ rad/h$$

$$\omega_{system} > \frac{1}{2}\omega_{\Delta} \quad \rightarrow \quad Does\ not\ verify\ the\ Shannon's\ law$$

From the tables, if we want to limit the error to less than 1%, the signal period should be 10 times the sample period.

II.5.2. THE SHIFT OF THE MAXIMUM

By plotting the numerical results obtained with the heat transfer functions we have seen that some curves have their maxima shifted. The shift magnitude is increasing with the increase of the sampling time. Figure II.11 is a zoom of the outside heat flux for the heavy density wall.

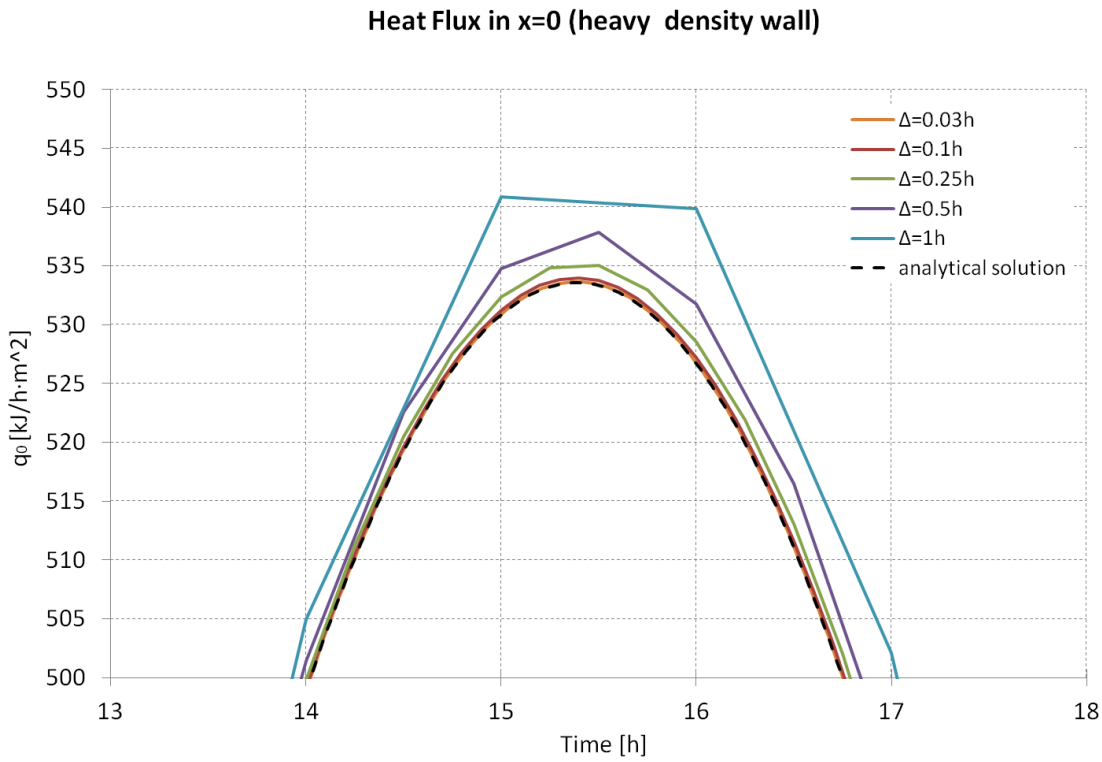


Figure II.11. Zoom of the maxima of the heat flux – case 2

It is impossible to know if the highest value of the numerical results is the maximum of the curve or not. We decided to calculate the time value that corresponds to the maximum by linear interpolation: we have calculated the two points were the function crosses the x-axis and computed the average time between those points to find the position of the maximum. Table II.10 shows the results.

		HEAVY DENSITY WALL			LIGHT DENSITY WALL		
		time (q=0) [h]	time average (q_{max})	difference (h)	time (q=0) [h]	time average (q_{max})	difference (h)
T=24h	analytical	19,93450525	25,93450525	/	18,46325432	24,46325432	/
		31,93450525			30,46325432		
	Δ=0.03	19,93450573	25,93450551	2,62E-07	18,46325436	24,46325436	3,56E-08
		31,93450529			30,46325436		
	Δ=0.1	19,93450654	25,93450631	1,06E-06	18,46325366	24,46325366	6,63E-07
		31,93450608			30,46325366		
	Δ=0.25	19,93448933	25,93448908	1,62E-05	18,46323871	24,46323871	1,56E-05
		31,93448883			30,46323871		
	Δ=0.5	19,93438392	25,93438363	1,22E-04	18,46322763	24,46322763	2,67E-05
		31,93438335			30,46322763		
	Δ=1	19,93385722	25,93385684	6,48E-04	18,46312898	24,46312898	1,25E-04
		31,93385645			30,46312898		
T=3h	analytical	21,55613504	22,30613504	/	11,6859514	12,4359514	/
		23,05613504			13,1859514		
	Δ=0.03	21,55613505	22,30613504	1,63E-09	11,6859514	12,4359514	3,62E-10
		23,05613504			13,1859514		
	Δ=0.1	21,55611332	22,30611331	2,17E-05	11,68588903	12,43588903	6,24E-05
		23,0561133			13,18588903		
	Δ=0.25	21,5572425	22,30724249	1,11E-03	11,68487884	12,43487884	1,07E-03
		23,05724248			13,18487884		
	Δ=0.5	21,56402585	22,31402583	7,89E-03	11,69559059	12,44559059	9,64E-03
		23,06402582			13,19559059		
	Δ=1	21,5202758	22,29629116	9,84E-03	11,59154226	12,41394709	2,20E-02
		23,07230652			13,23635192		
T=1h	analytical	21,59542177	21,84542912	/	15,09543034	15,34541392	/
		22,09543646			15,5953975		
	Δ=0.03	21,59542208	21,84542942	3,02E-07	15,09543034	15,34541392	1,23E-09
		22,09543675			15,59539751		
	Δ=0.1	21,59516687	21,84516686	2,62E-04	15,09515617	15,34515617	2,58E-04
		22,09516686			15,59515617		
	Δ=0.25	21,60134959	21,85134958	5,92E-03	15,1007167	15,3507167	5,30E-03
		22,10134956			15,6007167		
	Δ=0.5	/	22	1,55E-01	/	15,5	1,55E-01
		/			/		

Table II.10. Shift of the maximum – case 2

We have computed the solution for a period equal to T=24h, T=3h and T=1h because the differences between them are too small. In Table II.10 we realize that the shift is very small, so we consider that it does not impact the accuracy of the results.

CHAPTER III. MULTILAYER WALL

This chapter shows the results obtained for a wall composed of several layers. We focus our study on four aspects:

- Calculate the heat transfer coefficients for a multilayer wall.
- Check the accuracy versus results of TRNSYS.
- Check the method by comparing the results for an homogeneous wall divided in several layers and the one-layer wall.
- Apply the method to compute the heating and cooling demand of a house.

Here we do not have an analytical solution to compare the accuracy of the numerical results.

In case of a multilayer wall we remark:

- The overall transfer matrix is obtained by the products of the matrices of each layer. See it in Chapter I.
- The roots of B(s) must be searched numerically. There exists several methods to find the roots of a function, we use the bisection method because is the method used in TRNSYS.

<u>THE BISECTION METHOD</u>

The bisection method is a root-finding algorism which repeatedly divides an interval by its midpoint and selects the subinterval in which lies the root. The function *f(x)* must be continuous in the interval *[a,b]*. The method has five steps:

1. Verifies $f(a)\cdot f(b)<0$
2. Calculates the midpoint *c* of the interval *[a,b]* and computes *f(c)*. If $f(c)=0$, the root is *c*.
3. Computes $f(a)\cdot f(c)$ and $f(c)\cdot f(b)$ and selects the subinterval (*[a,c]* or *[c,b]*) with the negative product.
4. With the new interval, *[a,c]* or *[c,b]*, it returns to the first step and continues repeating the algorism until the required accuracy.

In our case we need an accuracy of 10^{-15} and 10 000 iterations to find the correct roots.

III.1. RESULTS FOR A STANDARD WALL

In this part we selected an Insulated Concrete Form (ICF) as a multilayer wall to compare the numerical results with the TRNSYS results. Table III.1 shows the properties of the wall.

MATERIAL	L [m]	k [kJ/(h·m·K)]	C_p [kJ/(kg·h)]	ρ [kg/m^3]
Gypsum board	0,016	0,16	1,09	800
EPS board	0,076	0,03	1,21	43
Heavyweight concrete	0,203	1,95	0,9	2240
EPS board	0,076	0,03	1,21	43
Stucco	0,025	0,72	0,84	1856

Table III.1. ICF properties

For the ICF wall, the coefficients calculated by TRNSYS are listed in Table III.2.

	ICF wall - TRNSYS			
	a	b	c	d
1	4,0956831E+01	4,3589097E-09	1,5845680E+01	1,0000000E+00
2	-1,0279176E+02	3,2091903E-05	-3,8852640E+01	-1,5480351E+00
3	8,5701039E+01	5,9292987E-04	3,1641651E+01	6,0808741E-01
4	-2,6096370E+01	1,1650217E-03	-9,3987358E+00	-5,8863053E-02
5	2,3108949E+00	4,3328434E-04	7,9197061E-01	2,1695377E-03
6	-7,9364346E-02	3,5113880E-05	-2,5980013E-02	-2,9585204E-05
7	9,8861187E-04	6,2116277E-07	3,1340296E-04	7,6329416E-08
8	-1,7911551E-06	2,2027218E-09	-5,8630798E-07	
∑	2,2590684E-03	2,2590694E-03	2,2590691E-03	3,3292982E-03

Table III.2. TRNSYS coefficients for ICF wall

The coefficients *a, b, c* and *d* and roots for different accuracy values are listed in Table III.3 to III.5. We can see that:

- We need an accuracy of 10^{-15} and 10 000 iterations to get the same results as TRNSYS.
- Coefficients *a, c* and *d* are nearly the same as the obtained with TRNSYS for the different accuracies.
- Coefficients *b* are very sensible to the value of the roots.
- The roots are nearly the same for the different accuracies. We need 15 decimals to see differences between their 3 last decimals but those differences are the cause of the changes on coefficients *b*.

	ICF wall - Matlab - Accuracy: 10^{-07}				
	a	b	c	d	roots
1	4,0956831E+01	-1,7776114E-05	1,5845682E+01	1,0000000E+00	0,006677189688396
2	-1,0279176E+02	8,1586765E-05	-3,8852645E+01	-1,5480351E+00	0,834018924012005
3	8,5701033E+01	5,4398177E-04	3,1641657E+01	6,0808747E-01	2,968911805340570
4	-2,6096365E+01	1,1857958E-03	-9,3987392E+00	-5,8863073E-02	3,250015358758540
5	2,3108936E+00	4,2947211E-04	7,9197143E-01	2,1695388E-03	3,603015230688470
6	-7,9364216E-02	3,5394502E-05	-2,5980070E-02	-2,9585210E-05	6,669934142163080
7	9,8860693E-04	6,1230259E-07	3,1340451E-04	7,6329214E-08	6,709814533496090
8	-1,7910902E-06	2,3084472E-09	-5,8632405E-07	-6,8600871E-11	7,551825047521970
9	9,7029321E-10	1,2210007E-12	3,3866586E-10		13,098483279400600
10					14,437866235205000
11					14,809130941113200
12					20,877008865562400
13					29,647456364950500
14					31,693554700390600
15					32,150232033129800
16					40,902766131455200
∑	2,2590695E-03	2,2590695E-03	2,2590695E-03	3,3292983E-03	

Table III.3. Coefficients a, b, c and d and roots of ICF wall for an accuracy of 10^{-07}

	ICF wall - Matlab - Accuracy: 10^{-12}				
	a	b	c	d	roots
1	4,0956831E+01	3,5382897E-09	1,5845680E+01	1,0000000E+00	0,006677189239533
2	-1,0279176E+02	3,2093543E-05	-3,8852640E+01	-1,5480351E+00	0,834018922097597
3	8,5701039E+01	5,9292925E-04	3,1641651E+01	6,0808741E-01	2,968912060006220
4	-2,6096370E+01	1,1650213E-03	-9,3987358E+00	-5,8863053E-02	3,250016757669610
5	2,3108949E+00	4,3328459E-04	7,9197061E-01	2,1695377E-03	3,603014554092740
6	-7,9364346E-02	3,5113854E-05	-2,5980013E-02	-2,9585204E-05	6,669929194456390
7	9,8861187E-04	6,2116376E-07	3,1340296E-04	7,6329416E-08	6,709810674198550
8	-1,7911549E-06	2,2027287E-09	-5,8630794E-07	-6,8601229E-11	7,551826437199800
9	9,7047906E-10	1,3575896E-12	3,3863644E-10		13,098483744822700
10					14,437864182146800
11					14,809133002915200
12					20,877008843752500
13					29,647455709800600
14					31,693558666111300
15					32,150231583426400
16					40,902766216835100
∑	2,2590694E-03	2,2590694E-03	2,2590694E-03	3,3292982E-03	

Table III.4. Coefficients a, b, c and d and roots of ICF wall for an accuracy of 10^{-12}

	ICF wall - Matlab - Accuracy: 10^{-15}				
	a	b	c	d	roots
1	4,0956831E+01	4,3590376E-09	1,5845680E+01	1,0000000E+00	0,006677189239490
2	-1,0279176E+02	3,2091903E-05	-3,8852640E+01	-1,5480351E+00	0,834018922097343
3	8,5701039E+01	5,9292987E-04	3,1641651E+01	6,0808741E-01	2,968912060020650
4	-2,6096370E+01	1,1650217E-03	-9,3987358E+00	-5,8863053E-02	3,250016757680720
5	2,3108949E+00	4,3328434E-04	7,9197061E-01	2,1695377E-03	3,603014554084080
6	-7,9364346E-02	3,5113880E-05	-2,5980013E-02	-2,9585204E-05	6,669929194741420
7	9,8861187E-04	6,2116277E-07	3,1340296E-04	7,6329416E-08	6,709810674987970
8	-1,7911549E-06	2,2027360E-09	-5,8630801E-07	-6,8601229E-11	7,551826437090320
9	9,7040960E-10	1,4070138E-12	3,3871830E-10		13,098483744782800
10					14,437864182120700
11					14,809133002843700
12					20,877008843757100
13					29,647455709769300
14					31,693558666454900
15					32,150231583125400
16					40,902766216812800
∑	2,2590694E-03	2,2590694E-03	2,2590694E-03	3,3292982E-03	

Table III.5. Coefficients a, b, c and d and roots of ICF wall for an accuracy of 10^{-15}

III.2. RESULTS OF AN HOMOGENOUS MULTILAYER WALL

By using a cosine wave as an input curve we will study the effect of the number of layers on the final results. In other words, we will examine if the accuracy of the program changes in function of the number of wall layers. The computation will be done for a sampling time equal to 1 hour, Δ=1h.

- Input temperature:

$$T_o(t) = 10 \cdot \cos\left(\frac{2 \cdot \pi}{24} \cdot t\right) \qquad (III.1)$$

- Properties of the heavy density material:

  Wall thickness, L = 0,203 m
  Thermal conductivity, k = 7,02 kJ/(h·m·K)
  Density, ρ = 2240 kg/m^3
  Specific heat, C_p = 0,9 kJ/(kg·K)

Table III.6 lists the coefficients for a one-layer wall and Table III.7 shows the coefficients for an homogeneous multilayer wall composed of 1, 2, 10, 20, 200 and 1000 layers.

	one-layer wall		
	a=c	b	d
1	1,3423604E+02	1,6590755E+00	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	1,3629712E-11
6	-2,5013531E-10	2,6839242E-11	
∑	1,8855530E+01	1,8855530E+01	5,4525252E-01

Table III.6. Coefficients for a one-layer wall

Notice that:

- Coefficients *d* are the same for a one-layer wall as an homogeneous multilayer wall.
- Coefficients *a, b* and *c* are nearly the same. For a multilayer wall, to calculate the coefficients *a, b* and *c*, the program must compute a huge number of matrix products. So, the results are less accurate because of the limited number of decimals stored in the computer memory.

	Homogeneous multilayer wall					
	1 layer			2 layers		
	a=c	b	d	a=c	b	d
1	1,3423604E+02	1,6590755E+00	1,0000000E+00	1,3423604E+02	1,6590755E+00	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-4,7045212E-01	-1,4171487E+02	1,2473125E+01	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	1,5713158E-02	2,6852544E+01	4,6355157E+00	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-8,5231741E-06	-5,1838363E-01	8,7786719E-02	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	1,3629712E-11	2,0072352E-04	2,6962784E-05	1,3629712E-11
6	-2,5021472E-10	2,6882981E-11		-2,5021472E-10	2,6882870E-11	
∑	1,8855530E+01	1,8855530E+01	5,4525252E-01	1,8855530E+01	1,8855530E+01	5,4525252E-01
	10 layers			20 layers		
	a=c	b	d	a=c	b	d
1	1,3423604E+02	1,6590755E+00	1,0000000E+00	1,3423604E+02	1,6590755E+00	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-4,7045212E-01	-1,4171487E+02	1,2473125E+01	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	1,5713158E-02	2,6852544E+01	4,6355157E+00	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-8,5231741E-06	-5,1838363E-01	8,7786719E-02	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	1,3629712E-11	2,0072352E-04	2,6962784E-05	1,3629712E-11
6	-2,5004419E-10	2,6953647E-11		-2,5008593E-10	2,6910515E-11	
∑	1,8855530E+01	1,8855530E+01	5,4525252E-01	1,8855530E+01	1,8855530E+01	5,4525252E-01
	50 layers			100 layers		
	a=c	b	d	a=c	b	d
1	1,3423604E+02	1,6590755E+00	1,0000000E+00	1,3423604E+02	1,6590755E+00	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-4,7045212E-01	-1,4171487E+02	1,2473125E+01	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	1,5713158E-02	2,6852544E+01	4,6355157E+00	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-8,5231741E-06	-5,1838363E-01	8,7786719E-02	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	1,3629712E-11	2,0072352E-04	2,6962784E-05	1,3629712E-11
6	-2,5025646E-10	2,6886035E-11		-2,5027068E-10	2,6879705E-11	
∑	1,8855530E+01	1,8855530E+01	5,4525252E-01	1,8855530E+01	1,8855530E+01	5,4525252E-01
	200 layers			1000 layers		
	a=c	b	d	a=c	b	d
1	1,3423604E+02	1,6590755E+00	1,0000000E+00	1,3423604E+02	1,6590755E+00	1,0000000E+00
2	-1,4171487E+02	1,2473125E+01	-4,7045212E-01	-1,4171487E+02	1,2473125E+01	-4,7045212E-01
3	2,6852544E+01	4,6355157E+00	1,5713158E-02	2,6852544E+01	4,6355157E+00	1,5713158E-02
4	-5,1838363E-01	8,7786719E-02	-8,5231741E-06	-5,1838363E-01	8,7786719E-02	-8,5231741E-06
5	2,0072352E-04	2,6962784E-05	1,3629712E-11	2,0072352E-04	2,6962784E-05	1,3629712E-11
6	-2,4999223E-10	2,6902853E-11		-2,5014277E-10	2,6862722E-11	
∑	1,8855530E+01	1,8855530E+01	5,4525252E-01	1,8855530E+01	1,8855530E+01	5,4525252E-01

Table III.7. Coefficients for an homogeneous multilayer wall

III.2.1. THE HEAT FLUX THROUGH AN HOMOGENEOUS MULTILAYER WALL

We compute the heat flux through a one-layer wall and through an homogeneous multilayer wall to see if there exist differences. Figure III.1 shows the evolution of the heat flux in x=L for a one layer wall and for an homogeneous multilayer wall.

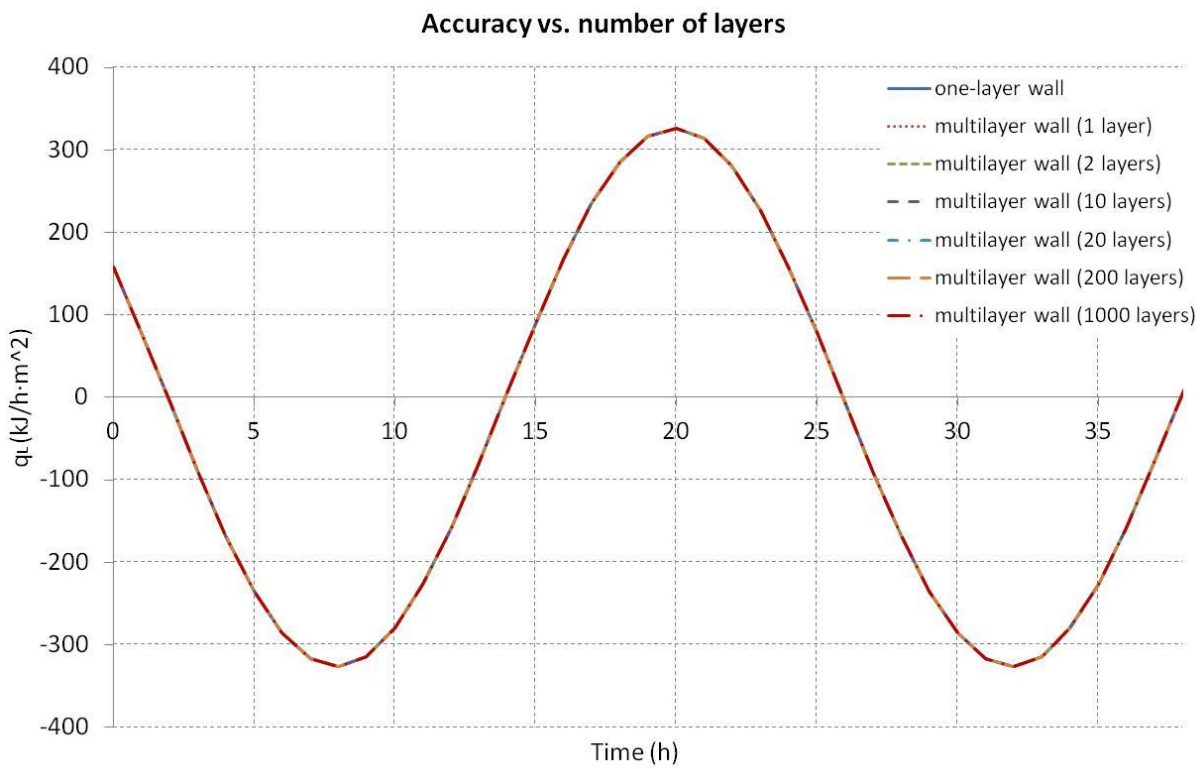


Figure III.1. Evolution of the heat flux through a one-layer wall and through an homogeneous multilayer wall

As we can see in Figure III.1, there are no differences between the results of a one-layer wall and for a wall with several numbers of layers. If we focus on the heat flux values (Table III.9) we notice that they change at the third decimal, which gives a highest difference of 0,003 kJ/m^2 in the heat fluxes (Table III.8).

		$\dot{q}_L$ [kJ/m^2]	difference [kJ/m^2]
one-layer wall		2487,2324	/
multilayer wall	1 layer	2487,2324	3,47E-10
	2 layers	2487,2324	0
	10 layers	2487,2324	1,36E-12
	20 layers	2487,2324	1,82E-12
	200 layers	2487,2287	3,68E-03
	1000 layers	2487,2287	3,96E-11

Table III.8. Total heat fluxes through a one-layer wall and through an homogeneous multilayer wall

	q_L [kJ/(h·m^2)]						
t [h]	one-layer wall	multilayer wall					
		1 layer	2 layers	10 layers	20 layers	200 layers	1000 layers
0	158,510	158,510	158,510	158,510	158,510	158,510	158,510
1	79,148	79,148	79,148	79,148	79,148	79,148	79,148
2	-5,604	-5,604	-5,604	-5,604	-5,604	-5,604	-5,604
3	-89,973	-89,973	-89,973	-89,973	-89,973	-89,973	-89,973
4	-168,211	-168,211	-168,211	-168,211	-168,211	-168,210	-168,210
5	-234,984	-234,984	-234,984	-234,984	-234,984	-234,984	-234,984
6	-285,744	-285,744	-285,744	-285,744	-285,744	-285,744	-285,744
7	-317,031	-317,031	-317,031	-317,031	-317,031	-317,030	-317,030
8	-326,712	-326,712	-326,712	-326,712	-326,712	-326,712	-326,712
9	-314,129	-314,129	-314,129	-314,129	-314,129	-314,129	-314,129
10	-280,138	-280,138	-280,138	-280,138	-280,138	-280,138	-280,138
11	-227,057	-227,057	-227,057	-227,057	-227,057	-227,056	-227,056
12	-158,502	-158,502	-158,502	-158,502	-158,502	-158,501	-158,501
13	-79,145	-79,145	-79,145	-79,145	-79,145	-79,145	-79,145
14	5,606	5,606	5,606	5,606	5,606	5,606	5,606
15	89,974	89,974	89,974	89,974	89,974	89,974	89,974
16	168,211	168,211	168,211	168,211	168,211	168,211	168,211
17	234,984	234,984	234,984	234,984	234,984	234,984	234,984
18	285,744	285,744	285,744	285,744	285,744	285,744	285,744
19	317,031	317,031	317,031	317,031	317,031	317,030	317,030
20	326,712	326,712	326,712	326,712	326,712	326,712	326,712
21	314,129	314,129	314,129	314,129	314,129	314,129	314,129
22	280,138	280,138	280,138	280,138	280,138	280,138	280,138
23	227,057	227,057	227,057	227,057	227,057	227,056	227,056
24	158,502	158,502	158,502	158,502	158,502	158,501	158,501
25	79,145	79,145	79,145	79,145	79,145	79,145	79,145
26	-5,606	-5,606	-5,606	-5,606	-5,606	-5,606	-5,606
27	-89,974	-89,974	-89,974	-89,974	-89,974	-89,974	-89,974
28	-168,211	-168,211	-168,211	-168,211	-168,211	-168,211	-168,211
29	-234,984	-234,984	-234,984	-234,984	-234,984	-234,984	-234,984
30	-285,744	-285,744	-285,744	-285,744	-285,744	-285,744	-285,744
31	-317,031	-317,031	-317,031	-317,031	-317,031	-317,030	-317,030
32	-326,712	-326,712	-326,712	-326,712	-326,712	-326,712	-326,712
33	-314,129	-314,129	-314,129	-314,129	-314,129	-314,129	-314,129
34	-280,138	-280,138	-280,138	-280,138	-280,138	-280,138	-280,138
35	-227,057	-227,057	-227,057	-227,057	-227,057	-227,056	-227,056
36	-158,502	-158,502	-158,502	-158,502	-158,502	-158,501	-158,501
37	-79,145	-79,145	-79,145	-79,145	-79,145	-79,145	-79,145
38	5,606	5,606	5,606	5,606	5,606	5,606	5,606

Table III.9. Heat flux values for a one-layer wall and for an homogeneous multilayer wall

Table III.10 shows the values of the computing time needed to compute the heat transfer coefficients. It is important to note that the heat transfer coefficients are computed once. Time values are small for a small number of layers, but increase when the number of layers is huge. The results are acceptable considered that a typical wall is made of less than five layers.

Number of layers	computation time
1	0,266 s
2	0,297 s
10	0,578 s
20	0,92 s
200	12,891 s
1000	3,302 min

Table III.10. Computation time vs. number of layers

III.3. RESULTS FOR A HOUSE

The objective is to evaluate the cooling and/or heating needs of a house in Châtelet. See Figure III.2 and Table III.11 to know the house properties.

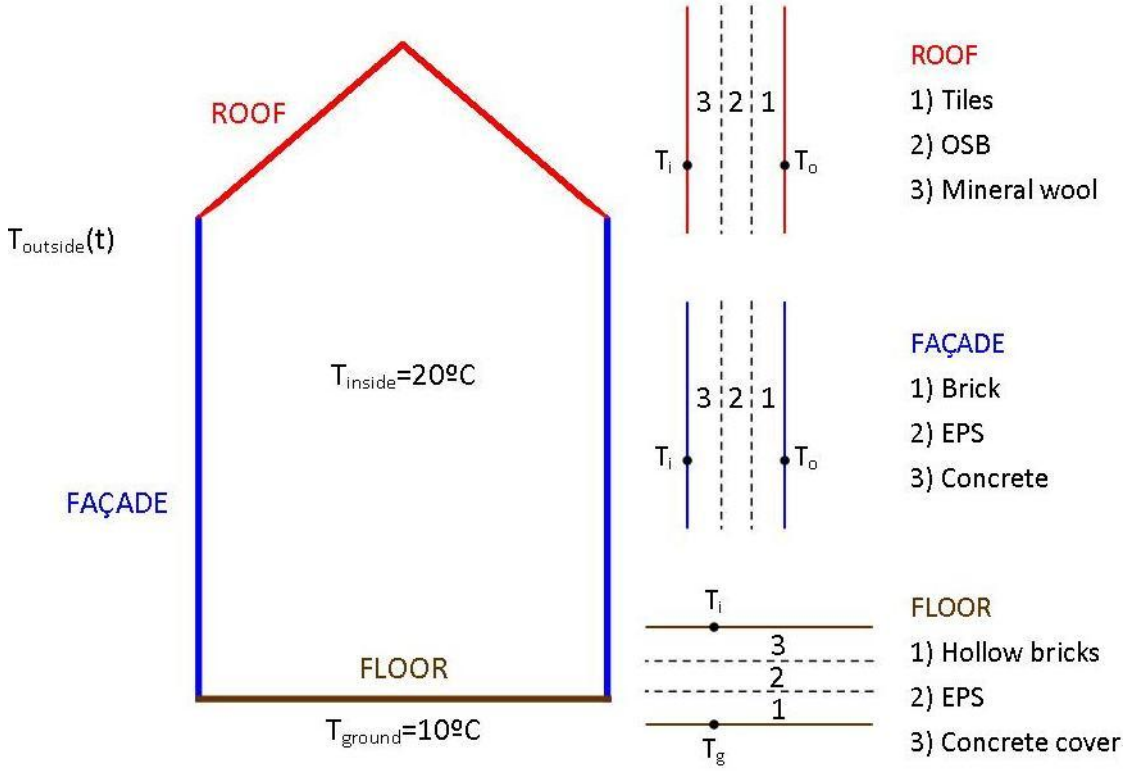


Figure III.2. Properties of the house

	MATERIAL	L [m]	k [kJ/(h·m·K)]	C_p [kJ/(kg·K)]	ρ [kg/m^3]	Area [m^2]
ROOF	1) Tiles	0,01	0,48	1	1200	131,28
	2) OSB	0,02	0,432	1,88	400	
	3) Mineral wool	0,12	0,18	0,84	100	
FAÇADE	1) Brick	0,09	2,38	0,84	1500	153,04
	2) EPS	0,03	0,162	1,47	30	
	3) Concrete	0,15	0,972	0,84	900	
FLOOR	1) Hollow bricks	0,15	3,6	0,84	2300	80,1
	2) EPS	0,15	0,162	1,47	30	
	3) Concrete cover	0,08	2,088	0,84	1300	

Table III.11. Material properties of the house

For the characteristics of each wall (roof, façade and floor) we compute numerically the heat transfer coefficients. We compute the coefficients for Δ=1h and Δ=0,2h. They are listed in Table III.12.

	ROOF							
	Δ=1h				Δ=0,2h			
	a	b	c	d	a	b	c	d
1	4,3881E+00	3,3612E-02	2,3509E+01	1,0000E+00	9,8111E+00	3,7926E-09	5,9507E+01	1,0000E+00
2	-4,6564E+00	4,3709E-01	-2,9714E+01	-4,8217E-01	-2,4861E+01	1,5434E-04	-1,5342E+02	-1,9482E+00
3	1,0877E+00	2,7949E-01	7,2784E+00	4,2853E-02	2,2995E+01	6,0389E-03	1,4368E+02	1,2990E+00
4	-5,5534E-02	1,3712E-02	-3,1079E-01	-1,1675E-04	-9,5345E+00	2,3017E-02	-6,0259E+01	-3,4880E-01
5	9,6745E-05	2,3732E-05	7,3374E-04	1,7138E-09	1,7753E+00	1,7762E-02	1,1374E+01	3,6403E-02
6	-1,2191E-09	4,4408E-10	-7,1220E-09		-1,3968E-01	3,3398E-03	-8,5871E-01	-1,3601E-03
7					4,0525E-03	1,5178E-04	2,3611E-02	1,1883E-05
8					-2,8466E-05	1,4788E-06	-1,7675E-04	-1,9049E-08
9					4,0210E-08	2,5728E-09	2,3956E-07	
∑	7,6393E-01	7,6393E-01	7,6393E-01	5,6057E-01	5,0464E-02	5,0464E-02	5,0464E-02	3,7031E-02
	FAÇADE							
	Δ=1h				Δ=0,2h			
	a	b	c	d	a	b	c	d
1	3,0588E+01	9,4347E-06	6,1655E+01	1,0000E+00	6,8397E+01	6,0396E-13	1,3817E+02	1,0000E+00
2	-6,7619E+01	7,8225E-03	-1,3877E+02	-1,6249E+00	-3,0656E+02	-2,3710E-12	-6,1929E+02	-3,8963E+00
3	5,1229E+01	6,9002E-02	1,0408E+02	8,1984E-01	5,7600E+02	1,1384E-08	1,1636E+03	6,2354E+00
4	-1,5596E+01	7,6671E-02	-2,9895E+01	-1,3498E-01	-5,9036E+02	1,9140E-06	-1,1927E+03	-5,3045E+00
5	1,5995E+00	1,5145E-02	3,1773E+00	3,8428E-03	3,6039E+02	3,5663E-05	7,2809E+02	2,6003E+00
6	-3,2180E-02	4,9906E-04	-7,6066E-02	-1,9795E-05	-1,3468E+02	1,6232E-04	-2,7201E+02	-7,4766E-01
7	1,4550E-04	2,1574E-06	2,5389E-04	1,5759E-08	3,0637E+01	2,4542E-04	6,1855E+01	1,2365E-01
8	-9,0162E-08	8,1139E-10	-1,7791E-07		-4,1135E+00	1,4161E-04	-8,3188E+00	-1,1222E-02
9					3,1046E-01	3,3109E-05	6,2858E-01	5,2820E-04
10					-1,2432E-02	3,1833E-06	-2,4978E-02	-1,1881E-05
11					2,4121E-04	1,2405E-07	4,8699E-04	1,1181E-07
12					-2,0068E-06	1,8927E-09	-4,1012E-06	-4,3188E-10
13					7,0493E-09	1,2061E-11	1,4107E-08	
∑	1,6915E-01	1,6915E-01	1,6915E-01	6,3824E-02	6,2336E-04	6,2336E-04	6,2336E-04	2,3521E-04
	FLOOR							
	Δ=1h				Δ=0,2h			
	a	b	c	d	a	b	c	d
1	5,3520E+01	2,5971E-04	9,4104E+01	1,0000E+00	1,2048E+02	6,9278E-13	2,1042E+02	1,0000E+00
2	-1,0907E+02	4,8183E-02	-1,8529E+02	-1,3828E+00	-4,7000E+02	1,1551E-09	-8,2087E+02	-3,3152E+00
3	6,9179E+01	2,0601E-01	1,1740E+02	5,3773E-01	7,4492E+02	2,2811E-06	1,3012E+03	4,3380E+00
4	-1,4200E+01	1,1107E-01	-2,7485E+01	-5,6077E-02	-6,1797E+02	1,0269E-04	-1,0791E+03	-2,8569E+00
5	9,4961E-01	8,5980E-03	1,6589E+00	3,8260E-04	2,8874E+02	7,1551E-04	5,0397E+02	1,0060E+00
6	-5,7606E-03	7,0752E-05	-8,1245E-03	-4,0566E-07	-7,6470E+01	1,3309E-03	-1,3363E+02	-1,8771E-01
7	3,6373E-06	4,1844E-08	7,8121E-06	1,7419E-11	1,1105E+01	8,1464E-04	1,9467E+01	1,7383E-02
8	-1,5153E-10	8,6266E-13		-1,3822E-18	-8,3074E-01	1,7762E-04	-1,4433E+00	-7,1535E-04
9				1,3593E-26	2,8514E-02	1,3954E-05	4,9490E-02	1,2442E-05
10				-2,7616E-37	-4,1439E-04	3,8353E-07	-7,3846E-04	-7,6068E-08
11				7,8166E-52	2,2642E-06	3,4919E-09	3,9018E-06	1,3766E-10
12				-3,3465E-67	-3,7134E-09	9,8425E-12	-6,4145E-09	
∑	3,7419E-01	3,7419E-01	3,7419E-01	9,9223E-02	3,1579E-03	3,1579E-03	3,1579E-03	8,3738E-04

Table III.12. Coefficients for the roof, façade and floor

The boundary conditions in this case are:

- Outside temperatures: extracted from TRNSYS
- Inside temperature: constant and equal to 20ºC, $T_i = 20ºC$
- Ground temperature: constant and equal to 10ºC, $T_g = 10ºC$

By using the heat transfer coefficients for each wall (roof, façade and floor) and the boundary conditions we calculate the heat flux through each wall, q [kJ/(h·m^2)], heat flux per surface area. Following, we multiply the heat flux *q* by each area *A* to obtain the heat flow rate,

$$Q\left[\frac{kJ}{h}\right] = q\left[\frac{kJ}{h \cdot m^2}\right] \cdot A[m^2] \qquad (III.2)$$

Finally the total heat flow rate inside the house is the sum of each part,

$$Q_{inside} = Q_{roof} + Q_{façade} + Q_{floor} \qquad (III.3)$$

In this part we focus on three aspects:

- Compare the results with the steady state solution computed by hand, using the following equation.

$$Q_{ss} = U \cdot (T_o - T_i) \qquad (III.4)$$

- Analyse the evolution of the heat flows for a winter week, from the 20th to the 27th of January.
- Analyse the evolution of the heat flows for a summer week, from the 25th to the 31st of July.

Figures III.3 and Figure III.4 show that there are no differences between the solutions computed with a sampling time equal to 1 hour and equal to 0,2 hours. Obviously, with a 0,2-hour sampling period more accuracy is obtained than with a 1-hour sampling period. For one week using the 1 hour sampling time results is enough.

Figure III.3 and Figure III.4 show that the heat flow rate through the floor is constant. In this case the temperatures of both sides of the floor have the same value.

During the winter's week (Figure III.3), the heat flow rate through the façade is always higher, in magnitude, than the heat flow rate through the roof. If we calculate the thermal resistance for each part we realize that the façade's value is smaller than the roof one (Table III.13), so in absolute value the heat flow rate will be always higher through the façade.

$$R_{total} = \sum_{n=1}^{\substack{number \\ of\ layers}} R_n = \sum_{n=1}^{\substack{number \\ of\ layers}} \frac{L_n}{k_n} \qquad (III.5)$$

	ROOF	FAÇADE	FLOOR
R_{total} [h·m^2·K/kJ]	0,734	0,377	1,006

Table III.13. Thermal resistances for the roof, the façade and the floor

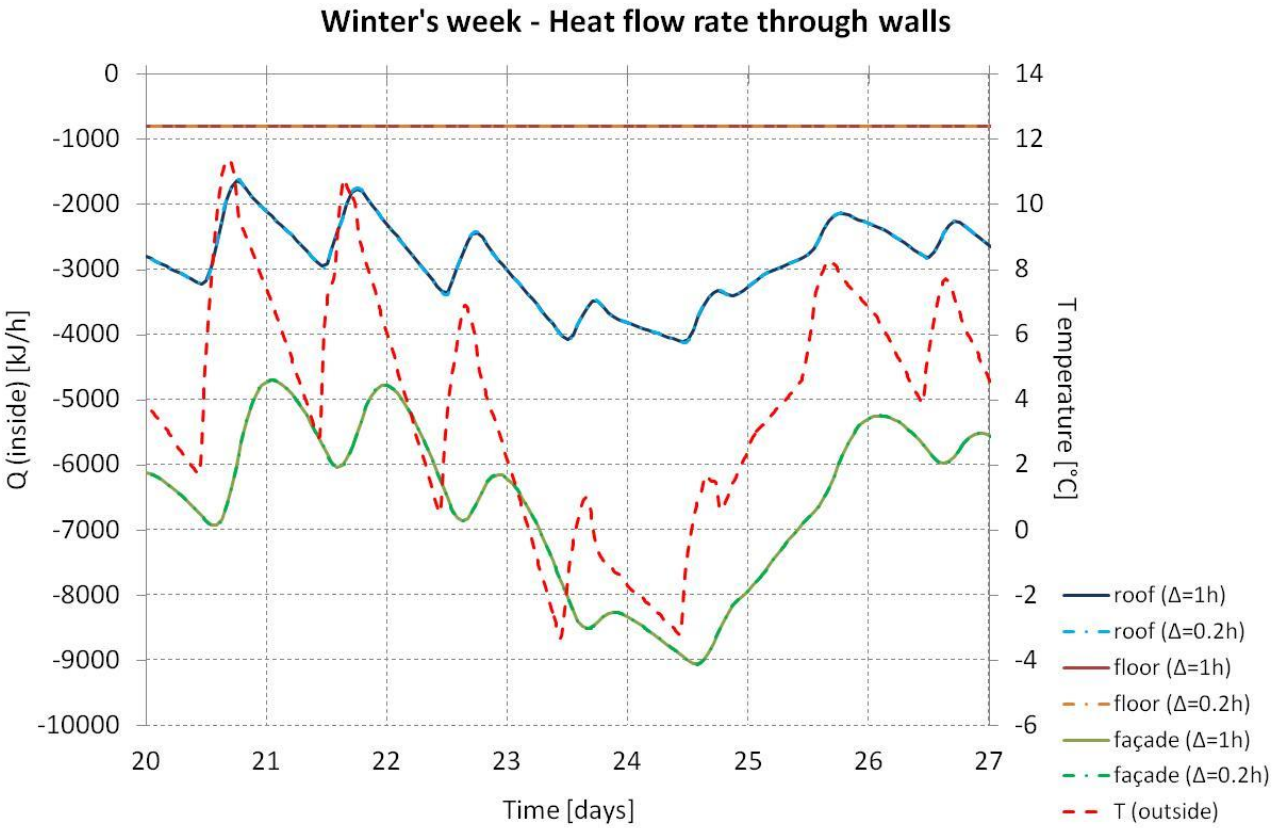


Figure III.3. Evolution of the heat flow rates through walls during a winter's week

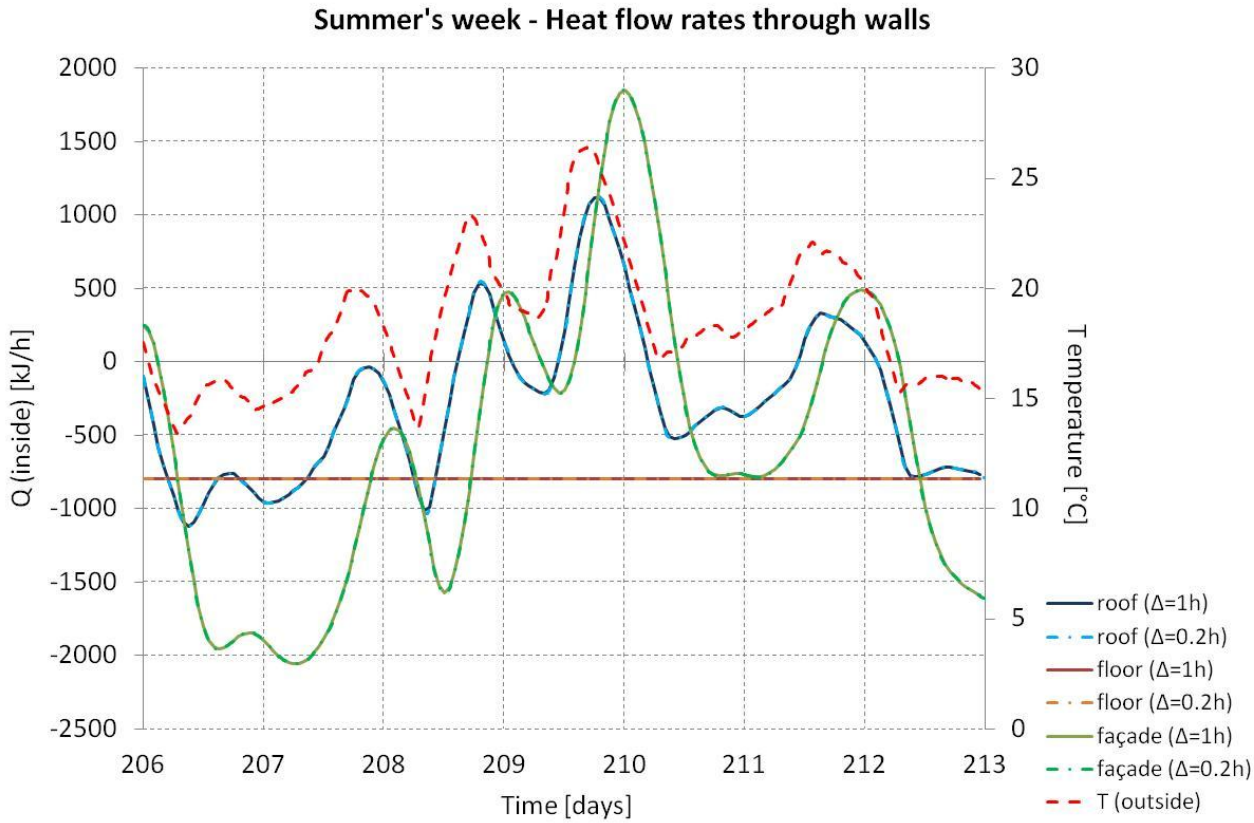


Figure III.4. Evolution of the heat flow rates through walls during a summer's week

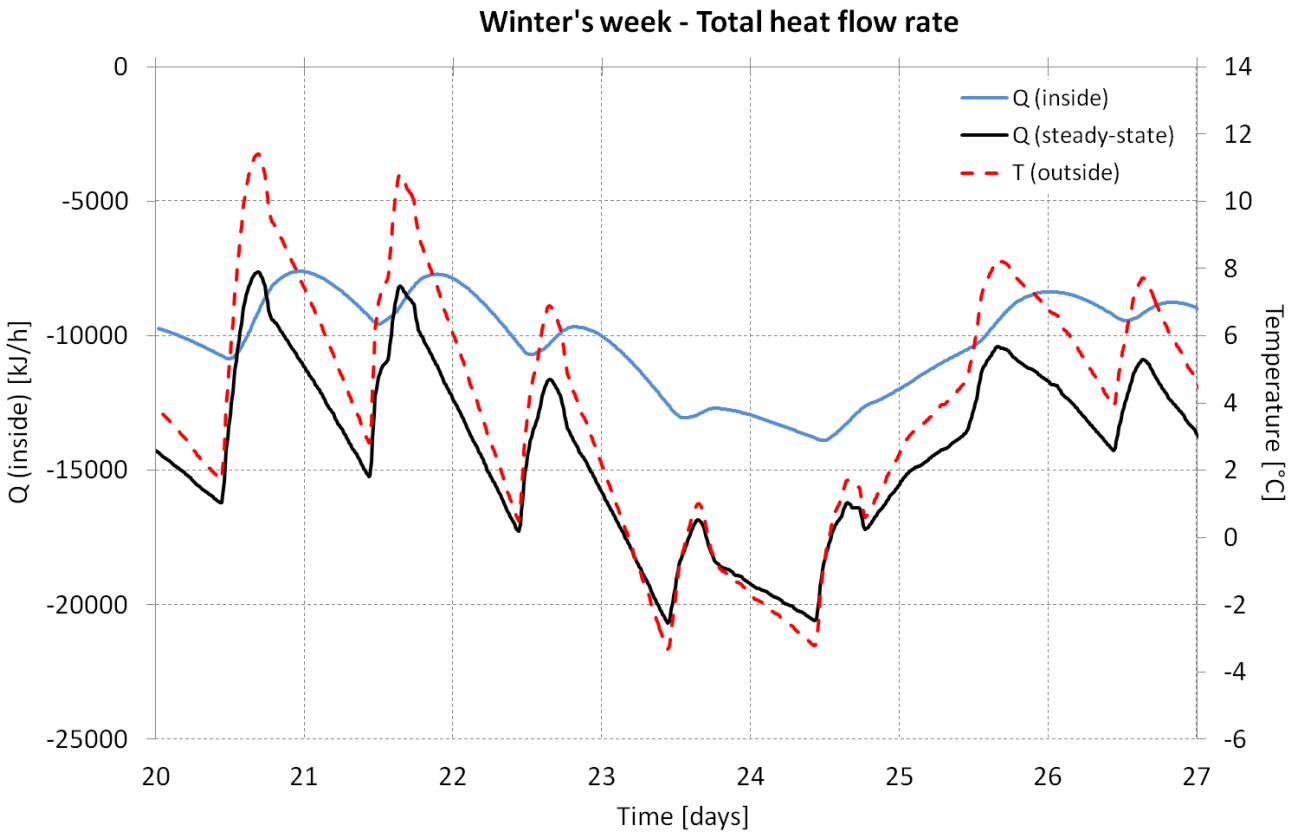


Figure III.5. Evolution of the total heat flow rate during a winter's week

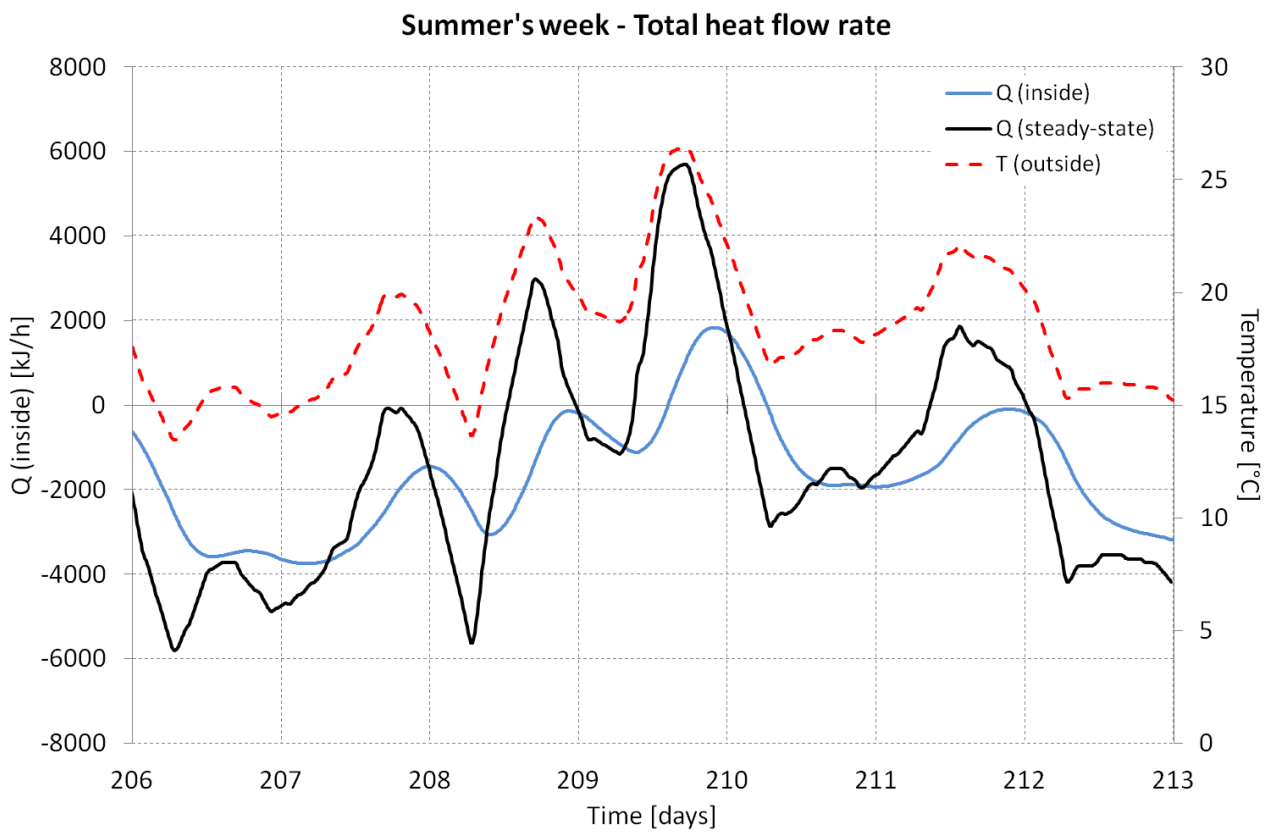


Figure III.6. Evolution of the total heat flow rate during a summer's week

In the last graph, Figure III.6, we can observe that the total heat flow rate is positive only once, although the outside temperature is higher than 20 ºC more than one time. The results show also that the values of the heat flow rate are negative during the majority of the time. The house used as a model in this scenario does not have any windows so that the cooling needs are lower than in the real home, because the solar gains are not taken into account.

The most obvious impact in Figure III.5 and Figure III.6 is that the steady-state heat flow rate is really different from the real heat flow rate. The steady-state values have been computed using the inside and outside instantaneous temperatures so the maximum values of the temperature and the steady-state flow rate happen at the same time. In the real case, the maxima are shifted and flattened, which is due to the thermal inertia of the walls.

CONCLUSIONS

The current implementation of the heat transfer function method to obtain the heat transfer coefficients has confirmed that this method compute accurate solutions really close to the analytical solution, knowing that the smaller the time period the more accurate is the solution.

In general, the coefficients obtained by the numerical method are quite the same as the coefficients obtained by TRNSYS. We do not know the standard used by TRNSYS to stop the calculation when it computes for a small time period. In our case, computing the same wall as computed by TRNSYS, we obtain the correct solution using smaller time steps, the smallest one equal to Δ=0,02h. Notice that it depends on the characteristics of the wall. The main issue of our computation is the limited number of decimals stored in the computer memory.

Our results show that it is not necessary to compute a large number of coefficients to obtain an accurate solution. The number of significant coefficients increases when the time step decreases.

We realize that coefficients *b* are very sensible to the value of the roots of B(s). So, it is important to use an accurate method to find the roots. We have used the bisection method to find the roots: it is a very simple and robust method but it takes much time to get the solutions. This drawback is not severe because coefficients are computed once. We also observed that sometimes this method do not find all the roots because they are very close one to each other, consequently, the final solution is wrong.

According to the computation time needed to compute the solutions, the time is acceptable considering that a typical wall is made of less than five layers.

For future works, a better validation for a temperature step input would have to consider the same ramp whatever the sampling period. It would be useful to analyse the computation for a wall with a large thickness because some paper reports a problem in that case.

As indicated, the numeric root-finding method used is very important to obtain accurate results. It would be interesting to try other converging methods which are faster and more accurate.

The heat transfer function method is valid with layers for which the thermal properties are constant. As we realize that it is possible to work with a wall formed of many layers (up to 1000) it would be interesting to try if it still works with a wall wherein the thermal parameters do not vary with the thickness dividing it in several layers wherein the thermal parameters are constant. If the thermal parameters vary with temperature (that is with time) the heat transfer method is not valid.

As the main problem of our computation is the computer memory to store decimals, would be also interesting to try to use the same method with dimensionless variables.

Finally, we have found some mistakes in the paper "*Conduction Transfer Functions in TRNSYS Multizone Building Model: Current Implementation, Limitations and Possible Improvements*" [6] that caused us some problems to program the Matlab code.

NOMENCLATURE

A, B, C, D = The elements of the heat transmission matrix

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = H$$

$Den(z) = \sum_{n=1}^{p} d_n \cdot z^{-l}$ = The denominator of the z-transfer function $\frac{A(z)}{B(z)}, \frac{D(z)}{B(z)}$ and $\frac{1}{B(z)}$

$[H] = \prod_{j=1}^{M} [H_j]$ = The heat transmission matrix for a wall of M layers

$$[H_j] = \begin{bmatrix} \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) & \frac{R_j \cdot sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right)}{L_j \cdot \sqrt{\frac{s}{\alpha_j}}} \\ C(s) = \frac{L_j \cdot \sqrt{\frac{s}{\alpha_j}} \cdot sinh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right)}{R_j} & \cosh\left(L_j \cdot \sqrt{\frac{s}{\alpha_j}}\right) \end{bmatrix}$$ = The heat transmission matrix for jth layer

$I(z)$ = z-transform of excitation function

L_j = Thickness of jth layer

M = Number of layer in a multilayer wall

$Num_A(z) = \sum_{n=0}^{2N+1} a_n \cdot z^{-1}$ = The numerator of the z-transfer function $\frac{A(z)}{B(z)}$

$O(z) = \frac{A(z)}{B(z)} \cdot I(z)$ = The z-transform of the output that results from excitation $I(z)$

p = Order of the polynomial $Den(z)$

q = Heat transfer flux

Q = Heat flow rate

$R_j = \frac{L_j}{k_j}$ = Thermal resistance of jth layer

$R = \sum_{j=1}^{M} R_j$ = Thermal resistance of a wall

T = Temperature of a wall surface

a_n = Coefficient of z^{-l} in $Num_A(z)$

d_n = Coefficient of z^{-l} in $Den(z)$

$C_0,\ C_1$ = Residues of $\frac{A(z)}{B(z) \cdot s^2}$ at the double pole at $s = 0$

e_n = Residue of $\frac{A(z)}{B(z) \cdot s^2}$ at $s = -\beta_N$

j = Layer number for a multilayer wall starting from the outside (x=0)

$i = \sqrt{-1}$

k_j = Thermal conductivity of jth layer

n = An index to identify terms in a series

s = Laplace transform parameter

t = Time

z = z-transform parameter

α_j = Thermal diffusivity of jth layer

$-\beta_N$ = Roots of $B(s) = 0$

Δ= Sampling interval for sampled data system

$\tau_j = \frac{{L_j}^2}{\alpha_j}$ = Characteristic time for jth layer

ψ = Is a positive real number

ρ_j = Density of jth layer

C_{pj} = Specific heat of jth layer

BIBLIOGRAPHY

[1] A. Nessi, L. Nisolle, *Régimes variables de fonctionnement dans les installations de chauffage central*, DUNOD, 1925

A. Nessi, L. Nisolle, *Résolution practique des problèmes de discontinuité de fonctionement dans les installations de chauffage central*, DUNOD, 1933

[2] Louis A. Pipes, *Matrix Analysis of Heat Transfer Problems*, Journal, Franklin Institude, Vol. 263, No. 3, 1957

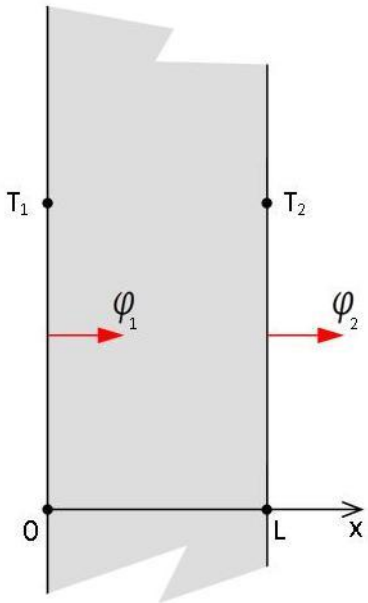
[3] Mitalas G.P. and Stephenson D.G., *Calculation of heat conduction transfer functions for multilayer slabs*, ASHRAE Transactions, 77(2), 117-126, 1971

[4] Brian Hahn and Dan Valentine, *Essential Matlab for engineers and scientists*, Newnes, 3rd Edition, 2007

[5] Matthew J. Hancock, *The 1-D Heat Equation*, Fall, 2006

[6] Benoit Delcroix, Michaël Kummert, Ahmed Daoud and Marion Hiller, *Conduction Transfer Functions in TRNSYS Multizone Building Model: Current Implementation, Limitations and Possible Improvements*, IBPSA Conferences, SIMBUILD, Madison WI (USA), 2012

APPENDIX A: PIPES (1957)



Material characteristics:

- Homogeneous material
- One-dimensional
- Constant properties ρ, C_p and k
- Thickness L

$T(x,t)$: Temperature (ºC)

$\varphi(x,t)$: Heat flux (W/m^2)

The values of temperature and heat flux at both sides of the material are:

$$T_1(t) = T(0,t)$$

$$T_2(t) = T(L,t)$$

$$\varphi_1(t) = \varphi(0,t)$$

$$\varphi_2(t) = \varphi(L,t)$$

The heat conduction transfer through the material is governed by:

- Fourier's law

$$-k\frac{\partial T}{\partial x} = \varphi \qquad (A.1)$$

- 1st law of thermodynamics (energy conservation)

$$-\frac{\partial \varphi}{\partial x} = \rho \cdot C_p \frac{\partial T}{\partial t} \qquad (A.2)$$

By solving Equation A.1 and A.2 we obtain the law of Fourier-Kirchhoff

$$\frac{\partial^2 T}{\partial x^2} = \frac{\rho \cdot C_p}{k}\frac{\partial T}{\partial t} \qquad (A.3)$$

By defining,

$$R = \frac{1}{k}$$

$$C = \rho \cdot C_p$$

Equation A.1, A.2 and A.3 become:

$$-\frac{\partial T}{\partial x} = R \cdot \varphi \qquad (A.4)$$

$$-\frac{\partial \varphi}{\partial x} = C \cdot \frac{\partial T}{\partial t} \qquad (A.5)$$

$$\frac{\partial^2 T}{\partial x^2} = R \cdot C \frac{\partial T}{\partial t} \qquad (A.6)$$

By taking the Laplace transform of functions $T(x,t)$ and $\varphi(x,t)$

$$\mathcal{L}\big(T(x,t)\big) = \int_0^\infty e^{-s \cdot t} \cdot T(x,t)dt = \mathcal{T}(x,s)$$

$$\mathcal{L}\big(\varphi(x,t)\big) = \int_0^\infty e^{-s \cdot t} \cdot \varphi(x,t)dt = \phi(x,s)$$

And supposing initial conditions, $\varphi(x,0) = 0$

$T(x,0) = 0$

Equation A.4, A.5 and A.6 become:

$$(A.4) \rightarrow \frac{\partial \mathcal{T}}{\partial x} = R \cdot \phi \qquad (A.7)$$

$$(A.5) \rightarrow \frac{\partial \phi}{\partial x} = C \cdot s \cdot \mathcal{T} \qquad (A.8)$$

$$(A.6) \rightarrow \frac{\partial^2 \mathcal{T}}{\partial x^2} = R \cdot C \cdot s \cdot \mathcal{T} \qquad (A.9)$$

We can rewrite Equation A.9 as:

$$\frac{1}{R \cdot C \cdot s} \frac{\partial^2 \mathcal{T}}{\partial x^2} - \mathcal{T} = 0 \qquad (A.10)$$

Equation A.10 is a particular form of:

$$a \cdot \frac{\partial^2 \mathcal{T}}{\partial x^2} + b \cdot \frac{\partial \mathcal{T}}{\partial x} + c \cdot \mathcal{T} = 0 \qquad (A.11)$$

Its characteristic equation is:

$$a \cdot \lambda^2 + b \cdot \lambda + c = 0 \qquad (A.12)$$

Where,

$$a = \frac{1}{R \cdot C \cdot s}$$

$$b = 0$$

$$c = -1$$

The determinant of Equation A.12 is:

$$\Delta = b^2 - 4 \cdot a \cdot c = 0 + \frac{4}{R \cdot C \cdot s} > 0$$

So, Equation A.12 has two real equivalent roots:

$$\frac{1}{R \cdot C \cdot s}\lambda^2 - 1 = 0 \quad \rightarrow \quad \lambda = \pm\sqrt{R \cdot C \cdot s}$$

Thus, a general solution of Equation A.10 is:

$$\mathcal{T}(x,s) = C_1 \cdot e^{\lambda_1 \cdot x} + C_2 \cdot e^{\lambda_2 \cdot x}$$

$$\mathcal{T}(x,s) = C_1 \cdot e^{\sqrt{R \cdot C \cdot s} \cdot x} + C_2 \cdot e^{-\sqrt{R \cdot C \cdot s} \cdot x} \qquad (A.13)$$

From Equation A.7 we compute:

$$\phi(x,s) = -\frac{1}{R}\frac{\partial \mathcal{T}}{\partial x} = -\frac{C_1}{R}\sqrt{RCs} \cdot e^{\sqrt{R \cdot C \cdot s} \cdot x} + \frac{C_2}{R}\sqrt{RCs} \cdot e^{-\sqrt{R \cdot C \cdot s} \cdot x}$$

$$\phi(x,s) = -\frac{C_1}{\sqrt{\frac{R}{Cs}}} e^{\sqrt{R \cdot C \cdot s} \cdot x} + \frac{C_2}{\sqrt{\frac{R}{Cs}}} e^{-\sqrt{R \cdot C \cdot s} \cdot x} \qquad (A.14)$$

Constants C_1 and C_2 can be computed by the boundary conditions:

$$\mathcal{T}_1(s) = \mathcal{T}(0,s)$$

$$\mathcal{T}_2(s) = \mathcal{T}(L,s)$$

$$\phi_1(s) = \phi(0,s)$$

$$\phi_2(s) = \phi(L,s)$$

From Equation A.13, A.14 and boundary conditions, we obtain:

$$\mathcal{T}_1 = C_1 + C_2 \qquad (A.15)$$

$$\mathcal{T}_2 = C_1 \cdot e^{\sqrt{R \cdot C \cdot s} \cdot L} + C_2 \cdot e^{-\sqrt{R \cdot C \cdot s} \cdot L} = C_1 \cdot a + C_2 \cdot b \qquad (A.16)$$

$$\phi_1 = -\frac{C_1}{\sqrt{\frac{R}{Cs}}} + \frac{C_2}{\sqrt{\frac{R}{Cs}}} \qquad (A.17)$$

$$\phi_2 = -\frac{C_1}{\sqrt{\frac{R}{Cs}}} e^{\sqrt{R \cdot C \cdot s} \cdot L} + \frac{C_2}{\sqrt{\frac{R}{Cs}}} e^{-\sqrt{R \cdot C \cdot s} \cdot L} = -\frac{C_1}{\sqrt{\frac{R}{Cs}}} \cdot a + \frac{C_2}{\sqrt{\frac{R}{Cs}}} \cdot b \qquad (A.18)$$

Where $a = e^{\sqrt{R \cdot C \cdot s} \cdot L}$ and $b = e^{-\sqrt{R \cdot C \cdot s} \cdot L}$

Solving the system formed by Equation A.16 and A.18, we obtain C_1 and C_2:

$$\begin{bmatrix} a & b \\ -\frac{a}{\sqrt{\frac{R}{Cs}}} & \frac{b}{\sqrt{\frac{R}{Cs}}} \end{bmatrix} \cdot \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} \mathcal{T}_2 \\ \phi_2 \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ -\frac{a}{\sqrt{\frac{R}{Cs}}} & \frac{b}{\sqrt{\frac{R}{Cs}}} \end{bmatrix}^{-1} = \frac{1}{\frac{a \cdot b}{\sqrt{\frac{R}{Cs}}} + \frac{a \cdot b}{\sqrt{\frac{R}{Cs}}}} \cdot \begin{bmatrix} \frac{b}{\sqrt{\frac{R}{Cs}}} & -b \\ \frac{a}{\sqrt{\frac{R}{Cs}}} & a \end{bmatrix} = \frac{1}{2} \begin{bmatrix} b & -b \cdot \sqrt{\frac{R}{Cs}} \\ a & a \cdot \sqrt{\frac{R}{Cs}} \end{bmatrix}$$

$$\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} b & -b \cdot \sqrt{\frac{R}{Cs}} \\ a & a \cdot \sqrt{\frac{R}{Cs}} \end{bmatrix} \cdot \begin{bmatrix} \mathcal{T}_2 \\ \phi_2 \end{bmatrix}$$

With $a = e^{\sqrt{R \cdot C \cdot s} \cdot L}$ and $b = e^{-\sqrt{R \cdot C \cdot s} \cdot L}$, C_1 and C_2 are:

$$C_1 = \frac{e^{-\sqrt{R \cdot C \cdot s} \cdot L}}{2} \cdot \left(\mathcal{T}_2 - \sqrt{\frac{R}{Cs}} \cdot \phi_2 \right)$$

$$C_2 = \frac{e^{\sqrt{R \cdot C \cdot s} \cdot L}}{2} \cdot \left(\mathcal{T}_2 + \sqrt{\frac{R}{Cs}} \cdot \phi_2 \right)$$

Knowing,

$$\sinh x = \frac{e^x - e^{-x}}{2}$$

$$\cosh x = \frac{e^x + e^{-x}}{2}$$

We can write $\mathcal{T}_1$ and ϕ_1 in function of $\mathcal{T}_2$ and ϕ_2 using Equation A.15, A.16, A.17 and A.18:

$$\begin{bmatrix} \mathcal{T}_1 \\ \phi_1 \end{bmatrix} = \begin{bmatrix} \cosh\theta & Z_0 \cdot \sinh\theta \\ \dfrac{\sinh\theta}{Z_0} & \cosh\theta \end{bmatrix} \cdot \begin{bmatrix} \mathcal{T}_2 \\ \phi_2 \end{bmatrix} \qquad (A.19)$$

Where:

$\theta = \sqrt{R \cdot C \cdot s} \cdot L =$ Propagation constant of the quadruple

$Z_o = \sqrt{\frac{R}{C \cdot s}} =$ Characteristic impedance of the quadruple

Matrix A.19 is called Heat Transfer Matrix H(θ),

$$H(\theta) = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

This transmission matrix is valid for a homogeneous layer of constant thermal properties. In the case of a multilayer wall, the overall transmission matrix is the product of the transmission matrices of each layer:

$$H = H_1 \cdot H_2 \cdot H_3 \cdot \ldots \cdot H_n$$

If a layer has a negligible heat capacity C_{pj}, its transmission matrix becomes:

$$H(\theta) = \begin{bmatrix} 1 & \frac{L}{k} \\ 0 & 1 \end{bmatrix}$$

APPENDIX B: THE Z-TRANSFORM

Table of equivalent Laplace and z-transform:

$f(t)$	$F(s)$	$F(z)$
1	$\frac{1}{s}$	$\frac{1}{1-z^{-1}}$
t	$\frac{1}{s^2}$	$\frac{\Delta}{z\cdot(1-z^{-1})^2}$
t^2	$\frac{2!}{s^3}$	$\frac{\Delta^2\cdot(1+z^{-1})}{z\cdot(1-z^{-1})^3}$
t^3	$\frac{3!}{s^4}$	$\frac{\Delta^3\cdot(1+4z^{-1}+z^{-2})}{z\cdot(1-z^{-1})^4}$
t^4	$\frac{4!}{s^5}$	$\frac{\Delta^4\cdot(1+11z^{-1}+11z^{-2}+z^{-3})}{z\cdot(1-z^{-1})^5}$
$e^{-a\cdot t}$	$\frac{1}{s+a}$	$\frac{1}{1-e^{-a\cdot\Delta}\cdot z^{-1}}$

Table B.1. Laplace and z-transform

APPENDIX C: Demonstration $\boldsymbol{cosh}\left(\sqrt{-\boldsymbol{s}}\right) = \boldsymbol{cos}(\sqrt{\boldsymbol{\psi}})$

We know that:

$$\cosh(x) = \frac{e^{x} + e^{-x}}{2} \qquad (C.1)$$

$$\cos(x) = \frac{e^{i \cdot x} + e^{-i \cdot x}}{2} \qquad (C.2)$$

For a function

$$f(s) = \cosh\left(\sqrt{-s}\right) \ , \qquad where\ s < 0 \qquad (C.3)$$

All the roots of B(s) are negative real values, so $-s = \psi$ with ψ real positive value

We can rewrite Equation C.3 as:

$$f(\psi) = \cosh\left(\sqrt{-1} \cdot \sqrt{\psi}\right) = \cosh\,(i \cdot \psi) \qquad (C.4)$$

By combining Equation C.1, C.2 and C.4,

$$\cosh\left(i \cdot \sqrt{\psi}\right) = \frac{e^{i \cdot \sqrt{\psi}} + e^{-i \cdot \sqrt{\psi}}}{2} = \cos\left(\sqrt{\psi}\right) \ , \qquad with\ \psi > 0 \qquad (C.5)$$

The same procedure must be just repeated to demonstrate that $sinh\left(\sqrt{-s}\right) = sin(\sqrt{\psi})$

APPENDIX D: Matlab code

1_LAYER WALL

```
%MONOLAYER WALL PARAMETERS
Cp = 0.9; %thermal capacity [kJ/(kg*K)]
L = 0.203; %length [m]
D = 2240; %density [kg/m^3]
k = 1.95*3.6; %thermal conductivity [J/(m*K)]

%PARAMETERS OF THE COMPUTATION
t = 0.01; %sampling time [h]
Nr = 20; %number of roots
coef_min=1.0e-10; %minimum value of coefficients a, b and c
max_coef=25; %maximum number of coefficients a, b and c

%PARAMETERS OF THE SIMULATION
%time simulation
s=1800;
%inside temperature (function f(x))
x=1:s; Ti(x)=0;
%outside temperature (function f(x))
x=1:s; To(x)=10*cos(2*pi/24*t*x);
%
%FUNCTIONS
r = B_roots (Cp,L,D,k,Nr);%computes the roots of B(s)
[ex,ey] = e_coef (Cp,L,D,k,Nr);%computes the coefficients used to compute
Ox(z) and Oy(z)
[dd,d,n] = d_coef (r,t,Nr);%computes the coefficients of D(z)
[C0x,C1x,C0y,C1y]= C0_C1 (D,Cp,k,L);%computes C0x,C1x,C0y and C1y used to
compute Ox(z) and Oy(z)
[oox,ooy,ox,oy] = o_coef (C0x,C1x,C0y,C1y,t,n,r,ex,ey,Nr);%computes the
coefficients of Ox(z) and Oy(z)
[a,b] = abc_coef (t,dd,oox,ooy,coef_min,max_coef);%computes the coefficients
of Nx(z)=Nz(z) and Ny(z)
[qi,qo,time,qt] = qi_qo (t,s,Ti,To,a,b,d);%computes the values of qL (x=L) and
q0 (x=0)

%FILES
save B(s)_roots r -double -ASCII
save coefficients_A=C a -double -ASCII
save coefficients_B b -double -ASCII
save coefficients_D d -double -ASCII
save coefficients_Ox=Oz ox -double -ASCII
save coefficients_Oy oy -double -ASCII
save coefficients_qi_qo qt -double -ASCII %first column=time; second
column=q(inside); third column=q(outside)
```

```
function [r] = B_roots (Cp,L,D,k,Nr)
%computes the roots of B(s)

a=k/(D*Cp); %thermal diffusivity

for x=1:Nr,
```

```
    r(x)=(x^2*pi^2*a)/L^2; %expression to compute the roots of B(s) for 1
layer wall
end
r=r';
end
```

```
function [ex,ey]=e_coef (Cp,L,D,k,Nr)
%computes the coefficients used to compute Ox(z)=Oz(z) and Oy(z)

a=k/(D*Cp); %thermal diffusivity
R=L/k; %thermal resistance

for x=1:Nr,
    ex(x)=-((2*L^2)/(a*R*x^2*pi^2)); %coefficients to compute Ox(z) and Oz(z)
end

for x=1:Nr,
    ey(x)=-((-1)^x)*((2*L^2)/(a*R*x^2*pi^2)); %coefficients to compute Oy(z)
end

end
```

```
function [dd,d,n] = d_coef (r,t,Nr)
%computes the coefficients of D(z)

for x=1:Nr,
    h(x)= exp(-r(x)*t); %coefficients to compute 'd' coefficients
end

%'d' coefficients are computed multiplying all the polynomials made with all
the 'h' coefficients
ddd=[1 -h(1)];
for x=2:Nr,
    u=[1 -h(x)];
    ddd=conv(ddd,u); %ddd=[1 -h(1)]*[1 -h(2)]*...*[1 -h(Nr)]
end

dd=[];
dd(1)=0;%this is the coefficient of z, we must create it to make able the
product with the ramp input I(z)
x=1;
while (x<=Nr+1),
    dd=[dd ddd(x)];
    x=x+1;
end
n=length(dd);

x=2;
while (x<=Nr),
    d(x-1)=dd(x);
    x=x+1;
end
```

```
d=d';

end
```

```
function [C0x,C1x,C0y,C1y]= C0_C1 (D,Cp,k,L)
%computes C0x,C1x,C0y and C1y used to compute Ox(z)=Oz(z) and Oy(z)

a=k/(D*Cp); %thermal diffusivity
R=L/k; %thermal resistance

%the following expressions are suitable only for 1 layer wall
C0x=1/R;
C1x=L^2/(a*3*R);

C0y=1/R;
C1y=-L^2/(a*6*R);

end
```

```
function [oox,ooy,ox,oy] = o_coef (C0x,C1x,C0y,C1y,t,n,r,ex,ey,Nr)
%computes the coefficients of Ox(z)=Oz(z) and Oy(z)

%the 2 first coefficients ('z' power 2 and 1) must be 0 to make able the
product with the ramp input I(z)
oox(1)=0;
oox(2)=0;
for x=3:n,
    sum_exp=0;
    for y=1:Nr,
        sum_exp=sum_exp+ex(y)*exp(-(x-2)*r(y)*t);
    end
    oox(x)=C1x+(x-2)*C0x*t+sum_exp; %general expression to compute the
coefficients of Ox(z)=Oz(z)
end

%The operations for Oy are the same as Ox
ooy(1)=0;
ooy(2)=0;
for x=3:n,
    sum_exp=0;
    for y=1:Nr,
        sum_exp=sum_exp+ey(y)*exp(-(x-2)*r(y)*t);
    end
    ooy(x)=C1y+(x-2)*C0y*t+sum_exp;
end

x=3;
while (x<=Nr),
    ox(x-2)=oox(x);
    oy(x-2)=ooy(x);
    x=x+1;
end
```

```
ox=ox';
oy=oy';

end
```

```
function [a,b]=abc_coef (t,dd,oox,ooy,coef_min,max_coef)
%computes the coefficients of Nx(z)=Nz(z) and Ny(z)

p=(1/t)*[1 -2 1]; %ramp input I(z)
g=conv(p,dd); %polynomial product between the  ramp input and D(z)

aa=conv(g,oox); %polynomial product between the  ramp input, D(z) and
Ox(z)=Oz(z)
bb=conv(g,ooy); %polynomial product between the  ramp input, D(z) and Oy(z)

a=[];
b=[];
x=4;
while (abs(aa(x))>coef_min),
    a=[a aa(x)];
    b=[b bb(x)];
    x=x+1;
end

for y=(length(a)+1):max_coef,
    a(y)=0;
    b(y)=0;
end

a=a';
b=b';

end
```

```
function [qi,qo,time,qt] = qi_qo (t,s,Ti,To,a,b,d)
%computes the values of qL (x=L) and q0 (x=0)

qi=[];
time=[];
for y=1:s,
    bT=0;
    for z=1:length(b), %computes the product between 'b' coefficients, present
outside temperature and previous outside temperatures
        if (y>=z),
            bT=bT+b(z)*To(y-z+1);
        end
    end
    cT=0;
    for z=1:length(a),%computes the product between 'c' coefficients, present
inside temperature and previous inside temperatures
        if (y>=z),
            cT=cT+a(z)*Ti(y-z+1);
        end
```

```
    end
    dq=0;
    for z=2:length(d), %computes the product between 'd' coefficients and
previous inside heat fluxes
        if (y>z),
           dq=dq+d(z)*qi(y-z+1);
        end
    end
    qi(y)=bT-cT-dq; %general expression for the inside heat flux
    time(y)=t*y;
end
time=time';

qo=[];
for y=1:s,
    aT=0;
    for z=1:length(a), %computes the product between 'a' coefficients, present
outside temperature and previous outside temperatures
        if (y>=z),
            aT=aT+a(z)*To(y-z+1);
        end
    end
    bT=0;
    for z=1:length(b), %computes the product between 'b' coefficients, present
inside temperature and previous inside temperatures
        if (y>=z),
            bT=bT+b(z)*Ti(y-z+1);
        end
    end
    dq=0;
    for z=2:length(d), %computes the product between 'd' coefficients and
previous outside heat fluxes
        if (y>z),
            dq=dq+d(z)*qo(y-z+1);
        end
    end
    qo(y)=aT-bT-dq; %general expression for the outside heat flux
end

qi=qi';
qo=qo';
time=time';
qt(:,1)=time;
qt(:,2)=qi;
qt(:,3)=qo;

end
```

MULTILAYER WALL

```
%MULTILAYER WALL PARAMETERS
k=[0.16 0.03 1.95 0.03 0.72]*3.6; %thermal conductivity of each layer [J/m*K]
d=[800 43 2240 43 1856]; %density of each layer [kg/m^3]
Cp=[1.09 1.21 0.9 1.21 0.84]; %thermal capacity of each layer [kJ/(kg*K)]
L=[0.025 0.72 1856 0.84 0.035]; %thickness of each layer [m]

%PARAMETERS TO FIND THE ROOTS OF B(s)
s_max=300; %range of 's' values to find the roots of B(s), s=[0,s_max]
n=10000; %number of intervals inside [0,s_max]
Ni=100000; %number of iterations in the bisection method
eps_abs=10e-15; %maximum value to consider that a point is the root
eps_step=10e-8; %maximum width of the interval inside the bisection method

%PARAMETERS OF THE COMPUTATION
t =1; %sampling time [h] of the ramp input
coef_min=1.0e-10; %minimum value of coefficients a, b and c
max_coef=25; %maximum number of coefficients a, b and c

%PARAMETERS OF THE SIMULATION
%time simulation:
s=54;
%inside temperature (function Ti(x)):
x=1:s; Ti(x)=0;
%outside temperature (function To(x)):
x=1:s; To(x)=10*cos(2*pi/24*t*x);

%FUNCTIONS
%Function to find the roots of B(s):
[r,Nr]=B_roots(n,s_max,Ni,eps_abs,eps_step,k,d,Cp,L);
%Function to computes C0x,C1x,C0y,C1y,C0z and C1z used to compute Ox(z),
%Oy(z) and Oz(z):
[C0x,C1x,C0y,C1y,C0z,C1z] = C0_C1(k,d,Cp,L);
%Function to computes the coefficients used to compute Ox(z), Oy(z) and
%Oz(z):
[ex,ey,ez] = e_coef(r,Nr,k,d,Cp,L);
%Function to computes the coefficients of D(z):
[dd,d,n] = d_coef (r,t,Nr);
%Function to computes the coefficients of Ox(z), Oy(z) and Oz(z):
[oox,ooy,ooz,ox,oy,oz] = o_coef (C0x,C1x,C0y,C1y,C0z,C1z,t,n,r,ex,ey,ez,Nr);
%Function to computes the coefficients of Nx(z), Ny(z) and Nz(z):
[a,b,c]=abc_coef (t,dd,oox,ooy,ooz,coef_min,max_coef);
%Function to computes the values of qi (inside the wall) and qo (outside):
[qi,qo,time,qt] = qi_qo (t,s,Ti,To,a,b,c,d);

%COEFFICIENTS TEST
format long
disp('sum(a)='), disp (sum(a))
disp('sum(b)='), disp (sum(b))
disp('sum(c)='), disp (sum(c))
disp('sum(d)='), disp (sum(d))
disp('U=sum(a)/sum(d)='), disp(sum(a)/sum(d))

%FILES
```

```
save B(s)_roots r -double -ASCII
save coefficients_A a -double -ASCII
save coefficients_B b -double -ASCII
save coefficients_C c -double -ASCII
save coefficients_D d -double -ASCII
save coefficients_Ox ox -double -ASCII
save coefficients_Oy oy -double -ASCII
save coefficients_Oz oz -double -ASCII
save coefficients_qi_qo qt -double -ASCII
%first column=time; second column=q(x=L); third column=q(x=0)
```

```
function [T] = HTM(s,k,d,Cp,L)
%Function to compute the heat transmission matrix and B(s) for a multilayer
%wall

n=length(L);
for x=1:n,
    a(x)=k(x)/(d(x)*Cp(x)); %Thermal diffusivity of each layer
    R(x)=L(x)/k(x); %Thermal resistance of each layer
end

for x=1:n,%Computes A(s)=D(s), B(s) and C(s) of each layer
    A(x)= cos(L(x)*sqrt(s/a(x)));
    B(x)= R(x)*sin(L(x)*sqrt(s/a(x)))/(L(x)*sqrt(s/a(x)));
    C(x)= -(L(x)*sqrt(s/a(x))*sin(L(x)*sqrt(s/a(x)))/R(x));
end

T=eye(2);
for x=1:n,
    T=T*[A(x) B(x); C(x) A(x)]; %Computes the transmission heat matrix T(s)
end

end
```

```
function [Tn] = HTM_layer(s,k,d,Cp,L,n)
%Computes the heat transfer matrix for the layer number 'n'

a=k(n)/(d(n)*Cp(n)); %Thermal diffusivity of each layer
R=L(n)/k(n); %Thermal resistance of each layer

An= cos(L(n)*sqrt(s/a));
Bn= R*sin(L(n)*sqrt(s/a))/(L(n)*sqrt(s/a));
Cn= -(L(n)*sqrt(s/a)*sin(L(n)*sqrt(s/a))/R);

Tn=[An Bn;Cn An];
end
```

```
function [dT] = dHTM(s,k,d,Cp,L)
%
n=length(L);
for x=1:n,
    a(x)=k(x)/(d(x)*Cp(x)); %Thermal diffusivity of each layer
```

```
    R(x)=L(x)/k(x); %Thermal resistance of each layer
end

for x=1:n,
    tau(x)=L(x)^2/a(x);%Characteristic time for each layer
end

for x=1:n,%Computes the derivative of A(s)=D(s),B(s) and C(s)
    dA(x)=(tau(x)/2)*sin(sqrt(tau(x)*s))/sqrt(tau(x)*s);
    dB(x)=tau(x)/2*R(x)/(tau(x)*s)*(sin(sqrt(tau(x)*s))/sqrt(tau(x)*s)-
cos(sqrt(tau(x)*s)));

dC(x)=(tau(x)/(2*R(x)))*(sin(sqrt(tau(x)*s))/sqrt(tau(x)*s)+cos(sqrt(tau(x)*s)
));
end

dT=zeros(2);
for l=1:n,%Computes the derivative of T(s)
    p1=eye(2);
    for j=1:(l-1),
        p1=p1*HTM_layer(s,k,d,Cp,L,j);
    end
    p2=eye(2);
    for j=(l+1):n,
        p2=p2*HTM_layer(s,k,d,Cp,L,j);
    end
    dT=dT+p1*[dA(l) dB(l);dC(l) dA(l)]*p2;
end

end
```

```
function [ r,Nr ] = B_roots(n,s_max,Ni,eps_abs,eps_step,k,d,Cp,L)
%This function finds the interval where the root is contained

int=s_max/n; %Width of the interval
a=eps_abs; %B(0) does not have solution, the first point has to be close to
s=0
b=int;
z=1;
for x=1:n,
    Ta=HTM(a,k,d,Cp,L);
    Tb=HTM(b,k,d,Cp,L);
    fa=Ta(1,2);
    fb=Tb(1,2);
    if sign(fa)~=sign(fb),%There is a root in the interval [a,b]
        r(z)=bisection(a,b,Ni,eps_step,eps_abs,k,d,Cp,L);
        z=z+1;
    elseif fa==0, %The root is in a
        r(z)=a;
        z=z+1;
    elseif fb==0; %The root is in b
        r(z)=b;
        z=z+1;
    else
    end
```

```
    a=b+eps_abs; %next interval [a,b]
    b=(x+1)*int;
end
Nr=length(r); %number of roots found
r=r';
end
```

```
function [r]=bisection(a,b,Ni,eps_step,eps_abs,k,d,Cp,L)
%This function finds the roots of B(s)

for x = 1:Ni
        c = (a + b)/2; % Find the mid-point
        Ta=HTM(a,k,d,Cp,L);
        Tb=HTM(b,k,d,Cp,L);
        Tc=HTM(c,k,d,Cp,L);
        fa=Ta(1,2);
        fb=Tb(1,2);
        fc=Tc(1,2);
       if (fc==0) %the root is in c
            r=c;
            return;
        elseif ( (fc*fa)< 0 )%the root lies in [a,c]
            b = c;
        else %the root lies in [c,b]
            a = c;
        end
        if ( b - a < eps_step )%minimum interval width
            if ( abs(fa) < abs(fb) && abs(fa) < eps_abs )
                r = a; %a is considered a root because it is really close to
it
                return;
            elseif ( abs(fb) < eps_abs )
                r = b;%b is considered a root because it is really close to it
                return;
            end
        end
end
error( 'the bisection method did not converge' );
end
```

```
function [C0x,C1x,C0y,C1y,C0z,C1z] = C0_C1(k,d,Cp,L)
%
n=length(L);
for x=1:n,
    a(x)=k(x)/(d(x)*Cp(x)); %Thermal diffusivity of each layer
    R(x)=L(x)/k(x); %Thermal resistance of each layer
end

T=eye(2);
for x=1:n,%Computes the heat transmission matrix for s=0
    T=T*[1 R(x);0 1];
end

for x=1:n,%Computes the derivative of A(s)=D(s), B(s) and C(s) for s=0
```

```
    dA(x)=L(x)^2/(a(x)*2);
    dB(x)=(R(x)*L(x)^2)/(a(x)*6);
    dC(x)=L(x)^2/(a(x)*R(x));
end

dT=zeros(2);
for l=1:n,%Computes the derivative of T(s) for s=0
    p1=eye(2);
    for j=1:(l-1),
        p1=p1*[1 R(j);0 1];
    end
    p2=eye(2);
    for j=(l+1):n,
        p2=p2*[1 R(j);0 1];
    end
    dT=dT+p1*[dA(l) dB(l);dC(l) dA(l)]*p2;
end

%Expressions to compute C0 and C1 for a multilayer wall
C0x=T(1)/T(3);
C1x=(dT(1)*T(3)-T(1)*dT(3))/T(3)^2;

C0y=1/T(3);
C1y=-dT(3)/T(3)^2;

C0z=T(4)/T(3);
C1z=(dT(4)*T(3)-T(4)*dT(3))/T(3)^2;

end
```

```
function [ex,ey,ez] = e_coef(r,Nr,k,d,Cp,L)
%Function to calculate the coefficients to compute Ox(z), Oy(z) and Oz(z)

for x=1:Nr,
    Tr=HTM(r(x),k,d,Cp,L);%Heat transfer function in s=root
    dTr=dHTM(r(x),k,d,Cp,L);%Derivative of the heat transfer function in
s=root
    ex(x)=Tr(1)/(r(x)^2*dTr(3));
    ey(x)=1/(r(x)^2*dTr(3));
    ez(x)=Tr(4)/(r(x)^2*dTr(3));
end

end
```

```
function [dd,d,n] = d_coef (r,t,Nr)
%computes the coefficients of D(z)

for x=1:Nr,
    h(x)= exp(-r(x)*t); %coefficients to compute 'd' coefficients
end

%'d' coefficients are computed multiplying all the polynomials made with all
the 'h' coefficients
```

```
ddd=[1 -h(1)];
for x=2:Nr,
    u=[1 -h(x)];
    ddd=conv(ddd,u); %ddd=[1 -h(1)]*[1 -h(2)]*...*[1 -h(Nr)]
end

dd=[];
dd(1)=0;%this is the coefficient of z, we must create it to make able the
product with the ramp input I(z)
x=1;
while (x<=Nr+1),
    dd=[dd ddd(x)];
    x=x+1;
end
n=length(dd);

x=2;
while (x<=Nr),
    d(x-1)=dd(x);
    x=x+1;
end

d=d';

end
```

```
function [oox,ooy,ooz,ox,oy,oz] = o_coef
(C0x,C1x,C0y,C1y,C0z,C1z,t,n,r,ex,ey,ez,Nr)
%Computes the coefficients of Ox(z)=Oz(z) and Oy(z)

%The 2 first coefficients ('z' power 2 and 1) must be 0 to make able the
product with the ramp input I(z)
oox(1)=0;
oox(2)=0;
for x=3:n,
    sum_exp=0;
    for y=1:Nr,
        sum_exp=sum_exp+ex(y)*exp(-(x-2)*r(y)*t);
    end
    oox(x)=C1x+(x-2)*C0x*t+sum_exp; %general expression to compute the
coefficients of Ox(z)=Oz(z)
end

%The operations for Oy are the same as Ox
ooy(1)=0;
ooy(2)=0;
for x=3:n,
    sum_exp=0;
    for y=1:Nr,
        sum_exp=sum_exp+ey(y)*exp(-(x-2)*r(y)*t);
    end
    ooy(x)=C1y+(x-2)*C0y*t+sum_exp;
end
```

```
%The operations for Oz are the same as Ox and Oy
ooz(1)=0;
ooz(2)=0;
for x=3:n,
    sum_exp=0;
    for y=1:Nr,
        sum_exp=sum_exp+ez(y)*exp(-(x-2)*r(y)*t);
    end
    ooz(x)=C1z+(x-2)*C0z*t+sum_exp;
end

x=3;
while (x<=Nr),
    ox(x-2)=oox(x);
    oy(x-2)=ooy(x);
    oz(x-2)=ooz(x);
    x=x+1;
end

ox=ox';
oy=oy';
oz=oz';

end
```

```
function [a,b,c]=abc_coef (t,dd,oox,ooy,ooz,coef_min,max_coef)
%computes the coefficients of Nx(z)=Nz(z) and Ny(z)

p=(1/t)*[1 -2 1]; %ramp input I(z)
g=conv(p,dd); %polynomial product between the  ramp input and D(z)

aa=conv(g,oox); %polynomial product between the  ramp input, D(z) and Ox(z)
bb=conv(g,ooy); %polynomial product between the  ramp input, D(z) and Oy(z)
cc=conv(g,ooz); %polynomial product between the  ramp input, D(z) and Oz(z)

a=[];
b=[];
c=[];
x=4;
while (abs(aa(x))>coef_min),%Process to erase the last values of the vector,
the wrong ones
    a=[a aa(x)];
    b=[b bb(x)];
    c=[c cc(x)];
    x=x+1;
end

for y=(length(a)+1):max_coef,%Process to limit the lenght of the vector
    a(y)=0;
    b(y)=0;
    c(y)=0;
end

a=a';
```

```
b=b';
c=c';

end
```

```
function [qi,qo,time,qt] = qi_qo (t,s,Ti,To,a,b,c,d)
%computes the values of qi (inside the wall) and qo (outside)

qi=[];
time=[];
for y=1:s,
    bT=0;
    for z=1:length(b), %computes the product between 'b' coefficients, present
outside temperature and previous outside temperatures
        if (y>=z),
            bT=bT+b(z)*To(y-z+1);
        end
    end
    cT=0;
    for z=1:length(c),%computes the product between 'c' coefficients, present
inside temperature and previous inside temperatures
        if (y>=z),
            cT=cT+c(z)*Ti(y-z+1);
        end
    end
    dq=0;
    for z=2:length(d), %computes the product between 'd' coefficients and
previous inside heat fluxes
        if (y>z),
           dq=dq+d(z)*qi(y-z+1);
        end
    end
    qi(y)=bT-cT-dq; %general expression for the inside heat flux
    time(y)=t*y;
end
time=time';

qo=[];
for y=1:s,
    aT=0;
    for z=1:length(a), %computes the product between 'a' coefficients, present
outside temperature and previous outside temperatures
        if (y>=z),
            aT=aT+a(z)*To(y-z+1);
        end
    end
    bT=0;
    for z=1:length(b), %computes the product between 'b' coefficients, present
inside temperature and previous inside temperatures
        if (y>=z),
            bT=bT+b(z)*Ti(y-z+1);
        end
    end
    dq=0;
```

```
    for z=2:length(d), %computes the product between 'd' coefficients and
previous outside heat fluxes
        if (y>z),
            dq=dq+d(z)*qo(y-z+1);
        end
    end
    qo(y)=aT-bT-dq; %general expression for the outside heat flux
end

qi=qi';
qo=qo';
time=time';
qt(:,1)=time;
qt(:,2)=qi;
qt(:,3)=qo;

end
```