Simple wall

MATLAB / Octave file:
`t02SimpleWall.m`
`fReadWeather.m`
`fSolRadTiltSurf.m`

Objectives:

- Physical analysis
- Discrete mathematical model: DAE and state-space
- Implement the model
- Discuss time step, stability and precision
- Discuss initial conditions

1 Physical analysis and mathematical model

Let's consider a simple concrete wall with indoor insulation (Figure 1).

Physical properties:
Wall surface: $S_w = 3 \text{ m} \times 3 \text{ m}$;
Air volume: $V_a = 3 \text{ m} \times 3 \text{ m} \times 3 \text{ m}$
Concrete: $\lambda_1 = 2.00 \text{ W/mK}$; $\rho_1 c_1 = 2.5 \cdot 10^6 \text{ J/K m}^2$; $w_1 = 0.20 \text{ m}$;
Insulation: $\lambda_2 = 0.04 \text{ W/mK}$; $\rho_2 c_2 = 2.0 \cdot 10^6 \text{ J/K m}^2$; $w_2 = 0.08 \text{ m}$;
A fan-coil in the indoor space.

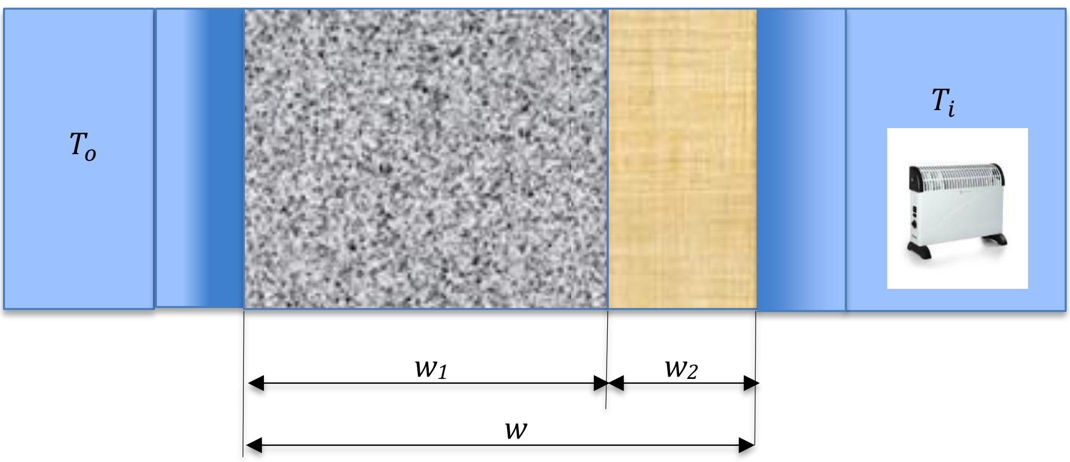


Figure 1 Simple wall: concrete and insulation

If the length and the height of the wall are much larger than the width (i.e. about 10 times larger), the heat transfer can be considered only in one direction (1D model). The heat transfer phenomena:

- convection from wall to outdoor air, $h_o = 10 \text{ W/m}^2\text{K}$
- conduction through the wall,

- convection from wall to indoor air, $h_i = 4 \text{ W/m}^2\text{K}$

The temperature of the indoor air is considered homogenous.

No other heat transfers are considered.

2 Discrete mathematical model

2.1 Choice of space and time discretization step

For explicit Euler method, Von Neumann stability analysis for heat equation requires the conditions [1]:

$$\frac{\lambda/\rho c \; \Delta t}{\Delta w^2} \leq \frac{1}{2} \tag{1}$$

or

$$\frac{\lambda S/w}{wS\rho c} \Delta t = \frac{\Delta t}{RC} \leq \frac{1}{2} \tag{2}$$

with $R = \frac{w}{\lambda S}; C = wS\rho c = V\rho c = mc$. This is applicable for heat equation. In general, the stability condition for explicit Euler method requires that all eigenvalues $\lambda_i \in \mathbf{Z}$ of the transfer matrix satisfy the condition [2]:

$$|1 + \lambda_i \; \Delta t| \leq 1, \forall i \tag{3}$$

or, if the eigenvalues are all real (as is the case for thermal networks):

$$-2 \leq \lambda_i \; \Delta t \leq 0, \forall i \tag{4}$$

Since the eigenvalues are related to the time constants:

$$\lambda_i = -\frac{1}{T_i} \tag{5}$$

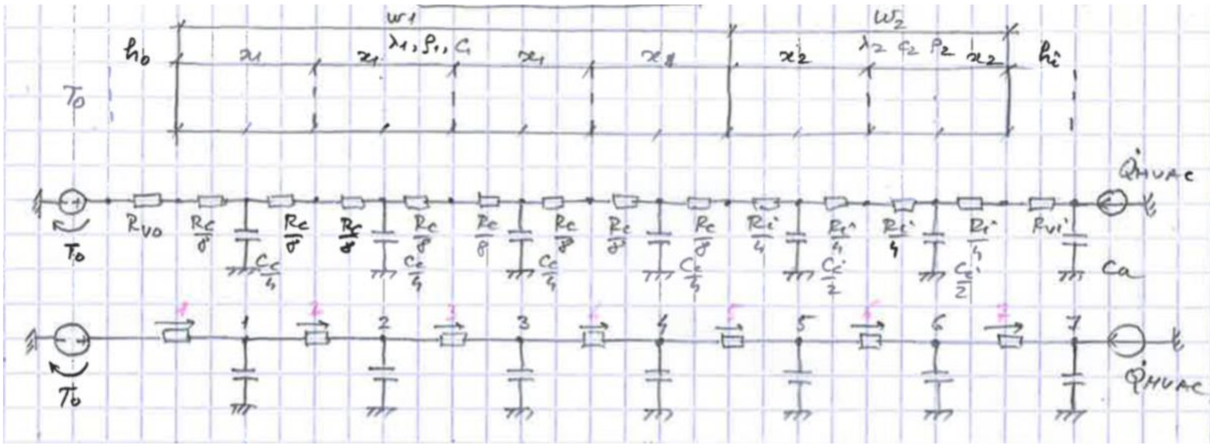
it results that the condition for stability is

$$\Delta t \leq \min T_i/2 \tag{6}$$

which is related to Shannon-Nyquist sampling theorem which requires that the sampling time is smaller than half of the shortest time constant. Note that if the discretization time step is larger than twice a time constant, that time constant can be neglected.

2.2 Thermal network

Discretization: concrete in 4 slices and insulation in 2 slices



$S_w = 3\,m \times 3\,m$ $\qquad V_a = 3\,m \times 3\,m \times 3\,m$

concrete: $\lambda_c = 2\ \frac{W}{mK}$ $\qquad w_c = 0{,}2\,m$ $\qquad \Delta x_c = 0{,}05\,m$ $\qquad \rho_c c_c = 2{,}5 \times 10^6\ \frac{J}{K m^3}$

insulation: $\lambda_c = 0{,}05\ \frac{W}{mK}$ $\qquad w_i = 0{,}08\,m$ $\qquad \Delta x_i = 0{,}04\,m$ $\qquad \rho_i c_i = 2 \cdot 10^6\ \frac{J}{K m^3}$

2.3 Differential-algebraic equations

$R_c = \frac{w_c}{\lambda_c S_w}$ $\qquad C_c = m_c c_c = V_c \rho_c c_c = S_w \cdot w_c \cdot \rho_c c_c$; $\qquad R_i = \frac{w_i}{\lambda_i S_w}$; $\qquad C_i = S_w \cdot w_i\, \rho_i c_i$

$R_1 = R_{vo} + \frac{R_c}{8}$; $\quad R_2 = R_3 = R_4 = \frac{R_c}{4}$; $\quad R_5 = \frac{R_c}{8} + \frac{R_i}{4}$; $\quad R_6 = \frac{R_i}{4}$; $\quad R_7 = \frac{R_i}{8} + R_{vi}$

$C_1 = C_2 = C_3 = C_4 = \frac{1}{4} C_c$; $\qquad C_5 = C_6 = \frac{1}{2} C_i$; $\qquad C_7 = V_a \rho_a c_a$

$$
q \backslash \theta \qquad
A = \begin{array}{c|c}
 & \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
\hline
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} &
\begin{bmatrix}
1 & & & & & & \\
-1 & 1 & & & & & \\
 & -1 & 1 & & & & \\
 & & -1 & 1 & & & \\
 & & & -1 & 1 & & \\
 & & & & -1 & 1 & \\
 & & & & & -1 & 1
\end{bmatrix}
\end{array}
\qquad
b = \begin{bmatrix} T_o \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
$$

$$f = [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ \dot{Q}_H]^T$$

The differential-algebraic equations (DAE) model is

$$\mathbf{C}\dot{\boldsymbol{\theta}} = -\mathbf{A}^T\mathbf{G}\mathbf{A}\boldsymbol{\theta} + \mathbf{A}^T\mathbf{G}\mathbf{b} + \mathbf{f} \tag{7}$$

If **C** is not singular, then the systems of equations may be put in the state-space representation

$$\dot{\boldsymbol{\theta}} = \mathbf{A_S}\boldsymbol{\theta} + \mathbf{B_S}\mathbf{u} \tag{8}$$

where
$\mathbf{A_S} \equiv -\mathbf{C}^{-1}\mathbf{A}^T\mathbf{G}\mathbf{A}$ is the transfer matrix;
$\mathbf{B_S} \equiv \mathbf{C}^{-1}([\mathbf{A}^T\mathbf{G} \quad \mathbf{I}])$ is the input matrix.

2.4 State-space representation

$$C\dot{\theta} = -A^TGA\theta + A^TGb + f \iff C\dot{\theta} = -A^TGA\theta + [A^TG \quad I_{f\times f}]\begin{bmatrix} b \\ f \end{bmatrix}$$

$$\dot{\theta} = \underbrace{-C^{-1}A^TGA}_{A}\,\theta + \underbrace{C^{-1}[A^TG \quad I_{f\times f}]}_{B}\underbrace{\begin{bmatrix} b \\ f \end{bmatrix}}_{u}$$

$$u = \begin{bmatrix} b \\ f \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} T_o \\ \vdots \\ 0 \end{bmatrix} \\ \begin{bmatrix} 0 \\ \vdots \\ \dot{Q}_H \end{bmatrix} \end{bmatrix} \rightarrow u = \begin{bmatrix} T_o \\ \dot{Q}_h \end{bmatrix} \qquad B = \begin{bmatrix} 7 \times 7 \end{bmatrix} \rightarrow B = \begin{array}{c} T_o \quad \dot{Q}_H \\ \begin{bmatrix} 7 \times 2 \end{bmatrix} \end{array}$$

$$y = C\theta + Du \qquad y = \theta_7 \rightarrow \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{C}\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \theta_7 \end{bmatrix}$$

Note that the size of **B** is initially 7 x 14:

- 7 lines for the number of temperatures in nodes;
- 14 columns for the inputs: 7 temperature sources on the seven branches and 7 heat-flow sources in the seven nodes.

Since only 2 inputs are really in the model, T_o (input 1) and $\dot{Q}_h$ (input 14), only they will be kept. This is done by MATLAB/Octave line:

```
52  B = B(:,[1 14]);        % select the 2 relevant inputs: 1->To and 14->Qh
```

2.5 Eigenvalues and eigenvectors

Matrix exponential solution of autonomous LDS

solution of $\dot{x} = Ax$, with $A \in \mathbf{R}^{n \times n}$ and constant, is

$$x(t) = e^{tA}x(0)$$

generalizes scalar case: solution of $\dot{x} = ax$, with $a \in \mathbf{R}$ and constant, is

$$x(t) = e^{ta}x(0)$$

[Boyd, 10-expm]

Time transfer property

for $\dot{x} = Ax$ we know

$$x(t) = \Phi(t)x(0) = e^{tA}x(0)$$

interpretation: the matrix e^{tA} propagates initial condition into state at time t

Dynamic interpretation

suppose $Av = \lambda v$, $v \neq 0$

if $\dot{x} = Ax$ and $x(0) = v$, then $x(t) = e^{\lambda t} v$

several ways to see this, *e.g.*,

$$\begin{aligned} x(t) = e^{tA} v &= \left(I + tA + \frac{(tA)^2}{2!} + \cdots \right) v \\ &= v + \lambda t v + \frac{(\lambda t)^2}{2!} v + \cdots \\ &= e^{\lambda t} v \end{aligned}$$

(since $(tA)^k v = (\lambda t)^k v$)

[Boyd 11-eig]

Interpretation: an eigenvector is an initial condition for which the evolution is simple:

$$x(t) = e^{\lambda t} v$$

the solution stays on the constant vector v.

Two modes http://www.youtube.com/watch?v=8RV3gXm6j2I 0:00 -
Three modes http://www.youtube.com/watch?v=jhGhSXAqXmU : 0:00 – 0:35

3 Implementation

See `t02SimpleWall.m`

3.1 Thermal network and state-space

Arc-node incidence matrix written as difference matrix (related to discretization)

```
% arc-node incidence matrix
A = eye(nq+1,nth);
A = -diff(A,1,1)';
```

Change of context from thermal network to state-space (matrices A and C change their meaning).

```
% State-space representation
B = inv(C)*[A'*G eye(nth,nth)]; % inputs u = [b; f] size(b)=nq, size(f)=nth;
B = B(:,[1 14]);         % select the 2 relevant inputs: 1->To and 14->Qh
A = inv(C)*(-A'*G*A);
C = zeros(1,7);C(7)=1;  % output: th(7)
D = 0;
```

3.2 Integration by using Euler forward and backward methods

The formulas for Euler forward (explicit)

$$\boldsymbol{\theta}_{k+1} = (\mathbf{I} + \Delta t\ \mathbf{A})\boldsymbol{\theta}_k + \Delta t\ \mathbf{B}\ \mathbf{u}_k$$

and for Euler backward (implicit)

$$\boldsymbol{\theta}_{k+1} = (\mathbf{I} + \Delta t\ \mathbf{A})^{-1}(\boldsymbol{\theta}_k + \Delta t\ \mathbf{B}\ \mathbf{u}_k)$$

are implemented as:

```
for k = 1:n-1
    th(:,k+1) = (eye(nth) + dt*A)*th(:,k) + dt*B*u(:,k);
    thi(:,k+1) = inv((eye(nth) - dt*A))*(thi(:,k) + dt*B*u(:,k));
end
```

See D. Rowell (2002) for more information.

4 Numerical experiments and discussions

4.1 Display

Change the display of the time-axis:

```
plot(Time/3600,th(7,1:n),'r', Time/3600,thi(7,1:n),'b')
xlabel('Time [h]'), ylabel('T [C]')
```

and

```
plot(Time/3600,th(7,1:n),'r', Time/3600,thi(7,1:n),'b')
xlabel('Time [h]'), ylabel('T [C]')
```

to display in seconds and in days.

4.2 Stability

The condition of stability is

$$-2 \leq \min \lambda_i \Delta t \ \leq 0$$

```
disp(['max dt = ',num2str(min(-2./eig(A))),'[s]'])
dt = 1819
```

This condition is respected if the time step is $dt < 1819$ s; if $dt > 1820$ s, the system is numerically unstable. Compare with the results for the explicit method.

Table 1 Eigenvalues and numerical stability `[sort(eig(A)*dt)]`

λ_i	$\lambda_i \Delta t$ for $dt = 1819$ s (stable)	$\lambda_i \Delta t$ for $dt < 1827$ s (instable)
-1.0994e-03	-1.9998011	-2.0085962
-6.6320e-04	-1.2063623	-1.2116679
-3.8761e-04	-0.7050680	-0.7081689
-2.2568e-04	-0.4105097	-0.4123151
-4.4772e-05	-0.0814404	-0.0817986
-1.7099e-05	-0.0311030	-0.0312398
-5.3887e-06	-0.0098021	-0.0098452

Change the value of `dt = 1827`

```
60 -     disp(['max dt = ',num2str(min(-2./eig(A))),'[s]'])
61 -     dt = 1827
```

Compare the results given by forward and backward Euler methods.

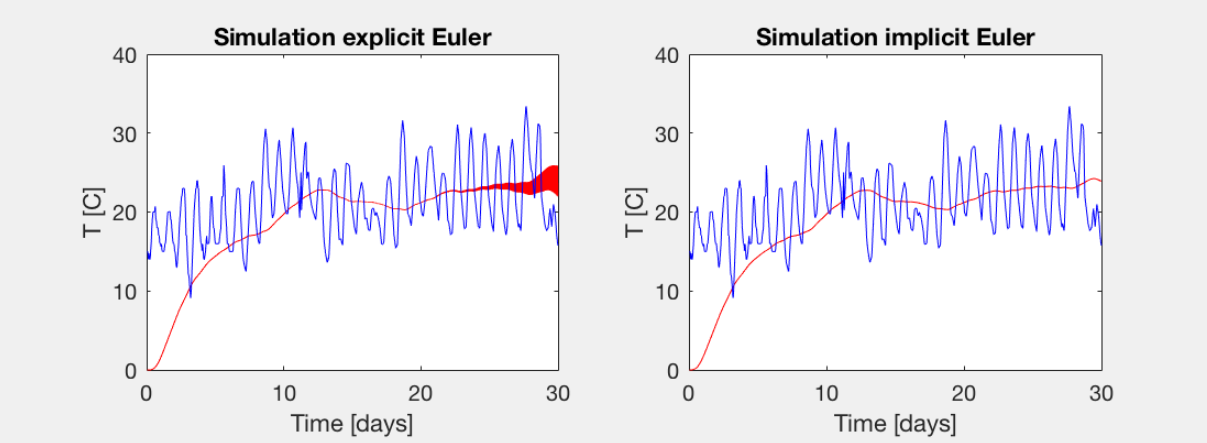


Figure 2 Comparison between forward (explicit) and backward (explicit) Euler methods for numerical integration.

4.3 Step responses

Make the system stable, by choosing in line 55: dh 1.98 or larger.

Step response for $T_o = 1$ °C

Find the heat flow rate in steady-state: $\dot{Q}_h = (\theta_6 - \theta_7)/R_{vi} \cong 0$ W

```
45 -     b = [1 0 0 0 0 0 0]'; f = [0 0 0 0 0 0 0]';
46 -     thSteadyTo = inv(A'*G*A)*(A'*G*b + f)
```

Explain the values of the temperature distribution.

Step response for $\dot{Q}_h = 1$ W

Find that the final temperature is $\theta_{7,end} = 0.2722$ °C.

Deduce the total thermal resistance and verify that it is equal to $R_{vo} + R_c + R_i + R_{vi} = 0.2722$ °C/W.

```
47 -     b = [0 0 0 0 0 0 0]'; f = [0 0 0 0 0 0 1]';
48 -     thSteadyTo = inv(A'*G*A)*(A'*G*b + f)
```

Explain the values of the temperature distribution.

Simulation for outdoor temperature

Discuss the evolution of indoor temperature for the two numerical integration methods (Euler explicit and implicit).

Compare the mean outdoor temperature with the mean indoor temperature.

5 References

[1] "Von Neumann stability analysis," 2016. [Online]. Available: https://en.wikipedia.org/wiki/Von_Neumann_stability_analysis.

[2] D. Ebery, "Stability Analysis fos Systems of Differential Equations," Geometric Tools LLC, 2008. [Online]. Available: https://www.geometrictools.com/Documentation/StabilityAnalysis.pdf.