indoor temperature prediction - using real datas to fit a RC analog model

Alexandre CUER
Dromotherm@Cerema
(Dated: March 15, 2020)

I. INDOOR TEMPERATURE FORECASTING WITH AN RC MODEL

c_{res} : thermal capacity of the envelope
c_s : thermal capacity of the indoor (air volume)
c_{res} and c_s are expressed in $\frac{J}{K}$
r_i : indoor thermal resistance of walls
r_0 : outdoor thermal resistance of walls
r_f : leakage resistance
r_i, r_0 and r_f are expressed in $\frac{K}{W}$
All temperatures T_i, T_{ext} and T_s being expressed in °C or K
All powers Q_{res} and Q_s are expressed in W
With $T_i(t)$ the simulated indoor temperature and $T_s(t)$ the simulated envelope temperature, we define :

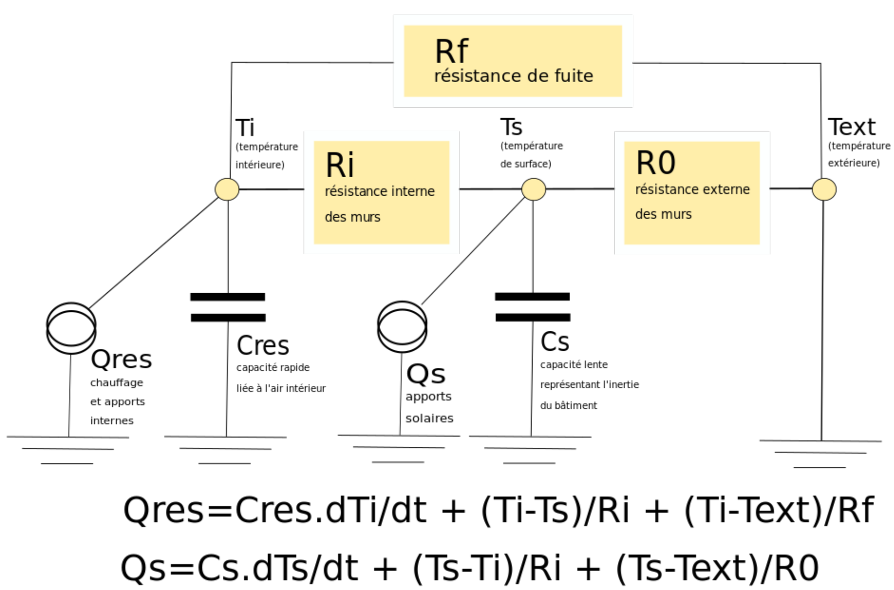
$$T_p(t) = \begin{vmatrix} T_i(t) \\ T_s(t) \end{vmatrix}$$

The equation to solve is :

$$\frac{dT_p(t)}{dt} = AT_p(t) + G(t, p)$$

Let's introduce the parameters vector :

$$p = [c_{res}, c_s, r_i, r_0, r_f]$$



R3C2 modelling of a (homogeneous) thermal zone of a building

$$A = \begin{vmatrix} -\frac{1}{c_{res}}(\frac{1}{r_i} + \frac{1}{r_f}) & \frac{1}{c_{res}r_i} \\ \frac{1}{c_s r_i} & -\frac{1}{c_s}(\frac{1}{r_i} + \frac{1}{r_0}) \end{vmatrix} = \begin{vmatrix} -\frac{1}{p[0]}(\frac{1}{p[2]} + \frac{1}{p[4]}) & \frac{1}{p[0]p[2]} \\ \frac{1}{p[1]p[2]} & -\frac{1}{p[1]}(\frac{1}{p[2]} + \frac{1}{p[3]}) \end{vmatrix}$$

$$\frac{dA}{dc_{res}} = \begin{vmatrix} \frac{1}{c_{res}^2}(\frac{1}{r_i} + \frac{1}{r_f}) & -\frac{1}{c_{res}^2 r_i} \\ 0 & 0 \end{vmatrix} = \begin{vmatrix} \frac{1}{p[0]^2}(\frac{1}{p[2]} + \frac{1}{p[4]}) & -\frac{1}{p[0]^2 p[2]} \\ 0 & 0 \end{vmatrix}$$

$$\frac{dA}{dc_s} = \begin{vmatrix} 0 & 0 \\ -\frac{1}{c_s^2 r_i} & \frac{1}{c_s^2}(\frac{1}{r_i} + \frac{1}{r_0}) \end{vmatrix}$$

$$\frac{dA}{dr_i} = \begin{vmatrix} \frac{1}{c_{res}r_i^2} & -\frac{1}{c_{res}r_i^2} \\ -\frac{1}{c_s r_i^2} & \frac{1}{c_s r_i^2} \end{vmatrix} = \begin{vmatrix} \frac{1}{p[0]p[2]^2} & -\frac{1}{p[0]p[2]^2} \\ -\frac{1}{p[1]p[2]^2} & \frac{1}{p[1]p[2]^2} \end{vmatrix}$$

$$\frac{dA}{dr_0} = \begin{vmatrix} 0 & 0 \\ 0 & \frac{1}{c_s r_0^2}) \end{vmatrix} = \begin{vmatrix} 0 & 0 \\ 0 & \frac{1}{p[1]p[3]^2} \end{vmatrix}$$

$$\frac{dA}{dr_f} = \begin{vmatrix} \frac{1}{c_{res}r_f^2} & 0 \\ 0 & 0 \end{vmatrix} = \begin{vmatrix} \frac{1}{p[0]p[4]^2} & 0 \\ 0 & 0 \end{vmatrix}$$

Let's introduce the ground truths tensor :
$\theta = [T_{ext}, T_{int}, Q_{res}, Q_s]$

$$G = \begin{vmatrix} \frac{Q_{res}}{c_{res}} + \frac{T_{ext}}{c_{res}r_f} \\ \frac{Q_s}{c_s} + \frac{T_{ext}}{c_s r_0} \end{vmatrix} = \begin{vmatrix} \frac{\theta^i[2]}{p[0]} + \frac{\theta^i[0]}{p[0]p[4]} \\ \frac{\theta^i[3]}{p[1]} + \frac{\theta^i[0]}{p[1]p[3]} \end{vmatrix}$$

$$\frac{dG}{dc_{res}} = \begin{vmatrix} -\frac{1}{c_{res}^2}(Q_{res} + \frac{T_{ext}}{r_f}) \\ 0 \end{vmatrix} = \begin{vmatrix} -\frac{1}{p[0]^2}(\theta^i[2] + \frac{\theta^i[0]}{p[4]}) \\ 0 \end{vmatrix}$$

$$\frac{dG}{dc_s} = \begin{vmatrix} 0 \\ -\frac{1}{c_s^2}(Q_s + \frac{T_{ext}}{r_0}) \end{vmatrix} = \begin{vmatrix} 0 \\ -\frac{1}{p[1]^2}(\theta^i[3] + \frac{\theta^i[0]}{p[3]}) \end{vmatrix}$$

$$\frac{dG}{dr_i} = \begin{vmatrix} 0 \\ 0 \end{vmatrix}$$

$$\frac{dG}{dr_0} = \begin{vmatrix} 0 \\ -\frac{T_{ext}}{c_s r_0^2} \end{vmatrix} = \begin{vmatrix} 0 \\ -\frac{\theta^i[0]}{p[1]p[3]^2} \end{vmatrix}$$

$$\frac{dG}{dr_f} = \begin{vmatrix} -\frac{T_{ext}}{c_{res}r_f^2} \\ 0 \end{vmatrix} = \begin{vmatrix} -\frac{\theta^i[0]}{p[0]p[4]^2} \\ 0 \end{vmatrix}$$

usually, specialists of RC models introduce a B matrix :

$B = \begin{vmatrix} \frac{1}{c_{res}r_f} & \frac{1}{c_{res}} & 0 \\ \frac{1}{c_s r_0} & 0 & \frac{1}{c_s} \end{vmatrix}$
and a U vector :
$U = \begin{vmatrix} T_{ext} \\ Q_{res} \\ Q_s \end{vmatrix}$
the equation to solve becomes :
$\frac{dT_p(t)}{dt} = A(p)T_p(t) + B(p)U(t)$
we have :
$\frac{dB}{dc_{res}} = \begin{vmatrix} \frac{-1}{c_{res}^2 r_f} & \frac{-1}{c_{res}^2} & 0 \\ 0 & 0 & 0 \end{vmatrix}$
$\frac{dB}{dc_s} = \begin{vmatrix} 0 & 0 & 0 \\ \frac{-1}{c_s^2 r_0} & 0 & \frac{-1}{c_s^2} \end{vmatrix}$
$\frac{dB}{dr_i} = \begin{vmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$
$\frac{dB}{dr_0} = \begin{vmatrix} 0 & 0 & 0 \\ \frac{-1}{c_s r_0^2} & 0 & 0 \end{vmatrix}$
$\frac{dB}{dr_f} = \begin{vmatrix} \frac{-1}{c_{res} r_f^2} & 0 & 0 \\ 0 & 0 & 0 \end{vmatrix}$

A. Discretisation according to an implicit Euler scheme

Noting i the discretisation index, we have $t^{i+1} - t^i = \delta t$
We can write :
$\frac{T_p^{i+1} - T_p^i}{\delta t} = A(p)T_p^{i+1} + B(p)U^i$
$(I - \delta t A(p))T_p^{i+1} = T_p^i + \delta t B(p)U^i$
The first discretisation scheme to code is :

$$T_p^{i+1} = (I - \delta t A(p))^{-1}[T_p^i + \delta t B(p)U^i]$$

B. Discretisation according to a Krank Nicholson scheme

$\frac{T_p^{i+1} - T_p^i}{\delta t} = A(p)\frac{T_p^{i+1} + T_p^i}{2} + B(p)\frac{U^{i+1} + U^i}{2}$
This scheme is more precise and would lead to a different algorithm :

$$T_p^{i+1} = (I - \frac{\delta t}{2}A(p))^{-1}[(I + \frac{\delta t}{2}A(p))T_p^i + \frac{\delta t}{2}B(p)(U^{i+1} + U^i)]$$

C. Optimization using a gradient descent and the Euler implicit scheme

we have to find the minimum of the cost function :
$f(p) = \frac{1}{2}\sum_{i=1}^{N}(T_p^i[0] - \theta^i[1])^2$
its gradient is :
$\nabla f = \begin{vmatrix} \frac{\delta f}{\delta p_0} \\ \frac{\delta f}{\delta p_1} \\ \frac{\delta f}{\delta p_2} \\ \frac{\delta f}{\delta p_3} \\ \frac{\delta f}{\delta p_4} \end{vmatrix} = \begin{vmatrix} \frac{\delta f}{\delta c_{res}} \\ \frac{\delta f}{\delta c_s} \\ \frac{\delta f}{\delta r_i} \\ \frac{\delta f}{\delta r_0} \\ \frac{\delta f}{\delta r_f} \end{vmatrix}$

with :
$\frac{\delta f}{\delta p_j} = \sum_{i=1}^N \frac{\delta T_p^i[0]}{\delta p_j}(T_p^i[0] - \theta^i[1])$
Let's call :
$z_j(t) = \frac{\delta T_p}{\delta p_j}$
By re-using the Euler implicit discretisation scheme, we have :
$\frac{\delta T_p^{i+1} - \delta T_p^i}{\delta t} = A(p)\delta T_p^{i+1} + \delta A(p) T_p^{i+1} + \delta B(p) U^i$
which leads to the second discretisation scheme to code :

$$z_j^{i+1} = (I - \delta t A)^{-1}(z_j^i + \delta t \frac{\delta A}{\delta p_j} T_p^{i+1} + \delta t \frac{\delta B(p)}{\delta p_j} U^i)$$

After this second scheme, we can estimate the gradients of the functional to minimize :

$$\frac{\delta f}{\delta p_j} = \sum_{i=1}^{N} z_j^i[0](T_p^i[0] - \theta^i[1])$$

D. Initial guess for the parameters

An empiric method for determining the initial thermal capacities is to appreciate the inertia of the building and its volume

According to regulatory approaches, we can define 5 classes of inertia :

$inertia\ type$	$\frac{KJ}{Km^2}$	$exchange\ surf.\ (\frac{m^2}{m^2})$
$very\ light$	80	2.5
$light$	110	2.5
$medium$	165	2.5
$hard$	260	3
$very\ hard$	370	3.5

In our use case (individual public housing estates from the 80s), the volume is 300 m3 and the inertia is medium

```
In [19]: # building volume in m3
         vb=300
         # air bulk density in kg/m3
         rho_air=1.22
         # air heat capacity in J/(kg.K)
         c_air=1004
         # heated floor area in m2
         floor=80
         # atbat in m2 - off-floor loss area
         atbat=217
         # inertia in J/(K.m2)
         inertia = 165000

         # res in J/K
         cres=c_air*rho_air*vb
         # cs in J/K
         cs=inertia*2.5*atbat

         print("cres is {}".format("{:e}".format(cres)))
         print("cs is {}".format("{:e}".format(cs)))
```

```
cres is 3.674640e+05
cs is 8.951250e+07
```

thermal resistances are generally small, something like 1e-2

```
In [ ]: ri=1e-2
        r0=1e-2
        rf=1e-2
```