$$
\begin{bmatrix} C_{air} & 0 \\ 0 & C_{wall} \end{bmatrix} \cdot \begin{bmatrix} \frac{dT_{air}}{dt} \\ \frac{dT_{wall}}{dt} \end{bmatrix} + \begin{bmatrix} \frac{1}{R_{air,amb}} + \frac{1}{R_{air,wall}} & \frac{-1}{R_{air,wall}} \\ \frac{-1}{R_{air,wall}} & \frac{1}{R_{wall,amb}} + \frac{1}{R_{air,wall}} \end{bmatrix} \cdot \begin{bmatrix} T_{air} \\ T_{wall} \end{bmatrix} = \begin{bmatrix} \frac{1}{R_{air,amb}} \cdot T_{amb} + \dot{Q}_{heat,air} + \dot{Q}_{int,air} + \dot{Q}_{solar,air} \\ \frac{1}{R_{wall,amb}} \cdot T_{amb} + \dot{Q}_{solar,wall} \end{bmatrix} \tag{72}
$$

it is clear that in this example, where multiple nodes in the thermal network are connected to the ambient surroundings, the approach of Section 5.10 becomes more adventageous:

5.12 3R3C model

The previous 3R2C model representation necessitates an *ad hoc* term in the heat supply vector $\dot{\mathbf{q}}$. Analogous to Section 5.10, we can include the ambient surroundings as a (large) heat capacity into the model. This will change the 3R2C model into a 3R3C model. The equations become:

$$
\mathbf{C} \cdot \dot{\boldsymbol{\theta}} = \begin{bmatrix} C_{amb} & 0 & 0 \\ 0 & C_{air} & 0 \\ 0 & 0 & C_{wall} \end{bmatrix} \cdot \begin{bmatrix} \frac{dT_{amb}}{dt} \\ \frac{dT_{air}}{dt} \\ \frac{dT_{wall}}{dt} \end{bmatrix} \tag{73}
$$

For $\mathbf{K}$ we can start with filling out the non-diagonal symmetic matrix elements:

$$
\mathbf{K} = \begin{bmatrix} 0 & \frac{-1}{R_{amb,air}} & \frac{-1}{R_{amb,wall}} \\ \frac{-1}{R_{amb,air}} & 0 & \frac{-1}{R_{air,wall}} \\ \frac{-1}{R_{amb,wall}} & \frac{-1}{R_{air,wall}} & 0 \end{bmatrix} \tag{74}
$$

Then we can complete the diagonal elements, so that the sum over each row becomes zero:

$$
\mathbf{K} \cdot \boldsymbol{\theta} = \begin{bmatrix} \frac{1}{R_{amb,air}} + \frac{1}{R_{air,wall}} & \frac{-1}{R_{amb,air}} & \frac{-1}{R_{amb,wall}} \\ \frac{-1}{R_{amb,air}} & \frac{1}{R_{amb,air}} + \frac{1}{R_{air,wall}} & \frac{-1}{R_{air,wall}} \\ \frac{-1}{R_{amb,wall}} & \frac{-1}{R_{air,wall}} & \frac{1}{R_{amb,wall}} + \frac{1}{R_{air,wall}} \end{bmatrix} \cdot \begin{bmatrix} T_{amb} \\ T_{air} \\ T_{wall} \end{bmatrix} \tag{75}
$$

$$
\dot{\mathbf{q}} = \begin{bmatrix} \dot{Q}_{amb} \\ \dot{Q}_{air} \\ \dot{Q}_{wall} \end{bmatrix} \tag{76}
$$

5.13 Coupling the housemodel elements

The housemodel is to be extended with modular elements representing the installations that supply the heat demanded by the building. Each subsystem contributes its own set of differential equations to the total system. In Fig. 23, the subsystems are indicated with a color code.

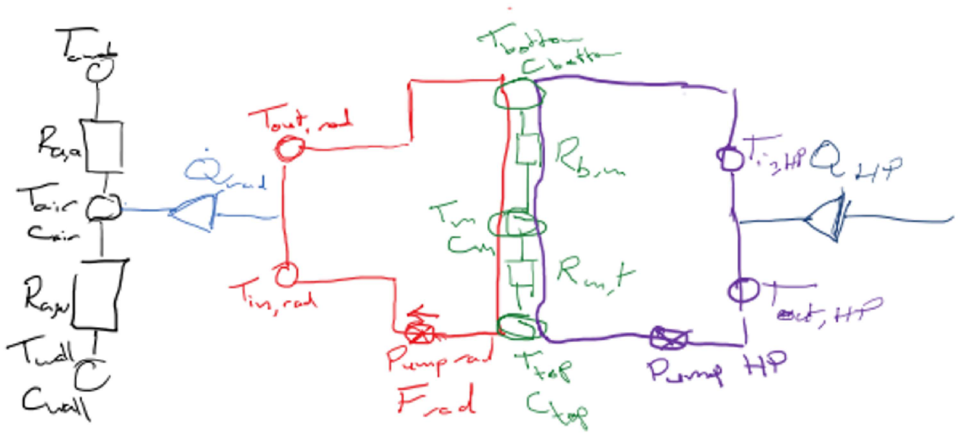

Figure 23: Grand Model

The building itself, in black, generates the differential equations:

$$
\begin{aligned}
T_{air}: \quad & C_{air}\frac{dT_{air}}{dt} = \frac{T_{amb}-T_{air}}{R_{air,amb}} + \frac{T_{wall}-T_{air}}{R_{air,wall}} + \dot{Q}_{rad,air} \\
T_{wall}: \quad & C_{wall}\frac{dT_{wall}}{dt} = \frac{T_{air}-T_{wall}}{R_{air,wall}}
\end{aligned}
\tag{77}
$$

T_{amb}: given as piecewise constant function, in interval $[t_i, t_{i+1}]$, $T_{amb} = T_{amb,i}$

The radiator element, coupled to the node T_{air}, transfers heat to the building at a rate $\dot{Q}_{rad}$. It has a feed temperature T_{feed} and a return temperature T_{return}. The radiator is modeled as a "cross-flow" heat exchanger, obeying the "radiator equation":

$$
\begin{aligned}
\dot{Q}_{rad}: \quad & \dot{Q}_{rad} = C_{rad}\cdot(\Delta T_{LMTD})^n, \text{ with } \Delta T_{LMTD} = \frac{T_{feed}-T_{return}}{\ln\left(\frac{T_{feed}-T_{air}}{T_{return}-T_{air}}\right)} \\
\text{also, :} \quad & \dot{Q}_{rad} = F_{rad}\cdot c_w \cdot (T_{feed}-T_{return})
\end{aligned}
\tag{78}
$$

when T_{feed}, T_{air} and F_{rad} are known, we have two equations with two unknowns $\dot{Q}_{rad}$ and T_{return}.

Question: when do you solve this system? Do you need to solve this within the time interval $[t_i, t_{i+1}]$?

Further equations:

$$
\begin{aligned}
& T_{feed} = T_{top} \\
& T_{return} \text{ should follow from equations above.}
\end{aligned}
\tag{79}
$$

$$T_{top}\colon \quad C_{top}\frac{dT_{top}}{dt} = \frac{T_{top}-T_{mid}}{-R_{mid,top}} + F_{HP}\cdot c_w \cdot (T_{HP,out} - T_{top}) + \max(F_{rad} - F_{HP}, 0)\cdot c_w \cdot (T_{mid} - T_{top})$$

$$T_{mid}\colon \quad C_{mid}\frac{dT_{mid}}{dt} = \frac{T_{top}-T_{mid}}{R_{mid,top}} + \frac{T_{mid}-T_{bot}}{-R_{bot,mid}} + \max(F_{HP} - F_{rad}, 0)\cdot c_w \cdot (T_{top} - T_{mid}) + \max(F_{rad} - F_{HP}, 0)\cdot c_w \cdot (T_{mid} - T_{bot})$$

$$T_{bot}\colon \quad C_{bot}\frac{dT_{bot}}{dt} = \frac{T_{mid}-T_{bot}}{R_{bot,mid}} + F_{rad}\cdot c_w \cdot (T_{return} - T_{bot}) + \max(F_{HP} - F_{rad}, 0)\cdot c_w \cdot (T_{mid} - T_{bot}) \tag{80}$$

$$\begin{aligned} T_{HP,in} &= T_{bot} \\ T_{HP,out}\colon \quad \dot{Q}_{HP} &= F_{HP}\cdot c_w \cdot (T_{HP,out} - T_{HP,in}\dot{Q}_{HP} \quad = f(T_{HP,in}, T_{HP,out}, T_{src,in}, T_{src,out})) \end{aligned} \tag{81}$$

heat pump function? Also here the question is: when to solve this equation?

5.14 Package "housemodel"

The repository "twozone_housemodel-git" contains the modules for the house model. The customary way to organize the modules is to make a *Python package* with *subpackages*. This opens up the possibility of publishing the package on PyPi, so that it can be imported.

See: https://pypi.org/

From commit e74ce58 the files in the twozone_housemodel-git repository are organized as a package. The proposed structure, implemented in this commit, is:

```
twozone_housemodel-git
├── housemodel
│   ├── __init__.py
│   ├── controls
│   │   └── __init__.py
│   ├── solvers
│   │   └── __init__.py
│   ├── sourcesink
│   │   └── __init__.py
│   └── tools
│       └── __init__.py
└── tests
    ├── __init__.py
    ├── context1.py
    └── test_*.py
```

- the *repository root* twozone_housemodel-git contains the simulation scripts and configuration files (for now)
- the *package root* housemodel contains the complete package. This can be seen since it contains an (empty) __init__.py module.
- the *subpackage* folders contain the modules with common functions and classes for all simulations. They