Mathematical basis

All the modeling is based on the heat conduction exchange equation in cylindric coordinates:

$$\begin{cases} T_t = \alpha\{T_{rr} + \beta\frac{T_r}{r}\}, \\ T(r,0) = T_0, \quad T_r(0,t) = 0, \\ T_r(R,t) = \frac{h}{k}\{T_{\text{Water}} - T(R,t)\}, \end{cases} \tag{*}$$

Where: * $T_t \equiv \partial T/\partial t$, $T_r \equiv \partial T/\partial r$, $T_{rr} \equiv \partial^2 T/\partial r^2$ * $r \in [0, R]$, representing the distance from the center of the food 0 is the center, R is the border at direct contect with the water * T_0, is the initial temprature of the food, generally 5°C * $T(r,t)$, representing the temperature of the food at distance r from the center at time t * β, representing the geometry of the shape (0 for slab, 1 for cylinder and 2 for sphere) * $T_{Water} $, representing the temperature set and maintained by the Roner

Furthermore, the reduction of the pathogens the Logaritmic Reduction (LR) is computed as follows:

$$\text{LR} = \frac{1}{D_{\text{Ref}}} \int_0^t 10^{\frac{T(t')-T_{\text{Ref}}}{z}} dt', \tag{**}$$

Where

- D_{Ref} is equal to $20s^{-1}$

Discrete version

The simulator is based on `Scipy` solver, accordingly a discretize version of the equation $(*)$ has been used and below you may find the considerations behind the code.

Heat conduction

The main equation in spherical coordinates $T_t = \alpha\{T_{rr} + \beta\frac{T_r}{r}\}$ is discretizez as follows:

- for $r = 0$ (center of the heated body):

$$\frac{\partial T}{\partial t} = \alpha\frac{\partial^2 T}{\partial r^2} \approx \frac{T(\Delta r, t) - T(0,t)}{\Delta r^2/2} \tag{i}$$

- for $r \in (0, R)$, each term will be approximated with discretization, particularly:

$$\frac{\partial^2 T}{\partial r^2} \approx \frac{T(r+\Delta r,t) - 2T(r,t) + T(r-\Delta r,t)}{\Delta r^2},$$

$$\frac{\partial T}{\partial r} \approx \frac{T(r+\Delta r,t) - T(r-\Delta r)}{2\Delta r}$$

and putting all together

$$\frac{\partial T}{\partial t} \approx \alpha \left[\frac{T(r+\Delta r,t) - 2T(r,t) + T(r-\Delta r,t)}{\Delta r^2} + \frac{T(r+\Delta r,t) - T(r-\Delta r)}{2\Delta r} \right] \tag{ii}$$

Boundary Conditions

Heat transfer at the border with the fluid (i.e., $r = R$) are modeled putting together Newton's law of heating and Fourier's thermal conductivity law, giving:

$$\text{Heat transfer} = k \frac{\partial T}{\partial r}\bigg|_{r=R} = -h\big(T(R,t) - T_{\text{Water}}\big)$$

and performing a discrete approximation of the left hand side, where $T(R+\Delta r,t)$ is a fictious point outside the food:

$$k\frac{T(R+\Delta r,t) - T(R,t)}{\Delta r} \approx -h\big(T(R,t) - T_{\text{Water}}\big)$$

and rearranging to explicitly write for the fictious point $T(R+\Delta r,t)$:

$$T(R+\Delta r,t) = T(R,t) - \frac{h\Delta r}{k}\big(T(R,t) - T_{\text{Water}}\big)$$

Coming back to the main heat conduction equation, evaluating for $r = R$, performing discrete approximation and susbsituting previously computed formula for the temperature of fictious point:

$$\begin{aligned}
\frac{\partial T}{\partial t}\bigg|_{r=R} &= \alpha \frac{\partial^2 T}{\partial r^2} \\
&\approx \alpha \frac{T(R-\Delta r,t) - 2T(R,t) + T(R+\Delta r,t)}{\Delta r^2} \\
&= \alpha \frac{T(R-\Delta r,t) - 2T(R,t) + T(R,t) - \frac{h\Delta r}{k}\big(T(R,t) - T_{\text{Water}}\big)}{\Delta r^2} \\
\frac{\partial T}{\partial t}\bigg|_{r=R} &= \alpha \frac{T(R-\Delta r,t) - T(R,t) - \frac{h\Delta r}{k}\big(T(R,t) - T_{\text{Water}}\big)}{\Delta r^2}
\end{aligned} \tag{iii}$$

By mapping equation terms into the following variables:

- $T(r, t)$ as `T[r]` for a given `t`
- $\partial T/\partial t(r, t)$ as `DTdt[r]` for a given `t`
- Δr as `dr`

equations (i), (ii), (iii) can be coded in Python as follows:

```
def heat_equation(t, T):
    dTdt = np.zeros_like(T)

    # Symmetry condition at r = 0
    dTdt[0] = msp.alpha * (2 / dr**2) * (T[1] - T[0])

    # Interior points
    for i in range(1, msp.N - 1):
        d2T_dr2 = (T[i+1] - 2*T[i] + T[i-1]) / dr**2
        radial_term = (msp.Beta / r[i]) * (T[i+1] - T[i-1]) / (2 * dr) if r[i] != 0 else 0
        dTdt[i] = msp.alpha * (d2T_dr2 + radial_term)

    # Convective boundary condition at the outer radius
    dTdt[-1] = msp.alpha * (1 / dr**2) *
                (T[-2] - T[-1] - (dr * msp.h / msp.k) * (T[-1] - msp.T_fluid))

    return dTdt
```