```
clc; close all; clear;
```

Rigoni Luca

Renewable Energy Convertion Systems - Assigment 2 - Energy analysis of a thermal solar system for DHW

Table of Contents

Data

```
A_collector = 2.14;          % Collector area [m^2]
Q_num = 64.2 / 3600 / 1000; % flow rate [m^3/s]
eta_0_num = 0.802;           % collector efficiency when T_in-T_a = 0 [-]
U_num = 4.3;                 % heat loss coefficient [W/(m^2·K)]
M_tank = 200;                % tank volume [kg]
hA = 2.5;                    % tank loss coefficent [W/K], CONVECTION COEFFICENT
epsilon_num = 0.6;           % heat exchanger effectiveness [-]

rho = 0.2;                   % albedo component of the ground reflectance
c_p_num = 4186;              % specific heat capacity of water [J/(kg·K)]
rho_water = 1000;            % density of water [kg/m^3]
```

Relevant solar data

```
n = [15,45,74,105,135,166,196,227,258,288,319,349]; % 15th day of each month over a
year
Month_day = [31,28,31,30,31,30,31,31,30,31,30,31];  % days in each month

t_max = 14; % assume time at which the max temperature achived during the day [h]
```

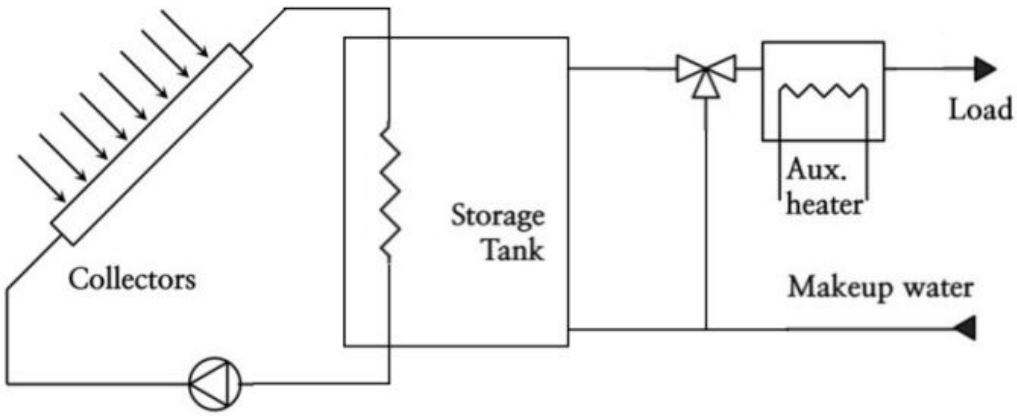
Temperature data

```
T_min = [4,3,5,8,12,16,18,18,15,12,7,5];     % Min temperature [°C]
T_max = [8,8,10,13,18,23,25,25,21,16,12,9]; % maximum ambient temperature per each
month[°C]
T_roof = 18;                                 % constant room temp [°C]
T_in_num = 30;                               % input temperature to the collector
[°C]
T_mains_s = 16;                              % mains temperature during the summer
[°C]
T_mains_w = 12;                              % mains temperature during the winter
[°C]
T_start = 30;                                % starting temperature of the tank [°C]
T_cellar = 18;                               % temperature of the room in which the
tank is placed [°C]
```

Location data (Trento)

```
H = 3.6*[1.649,2.514,3.889,4.903,5.636,6.471,6.488,5.575,4.313,2.812,1.653,1.277];
% Global horizontal irradiation [MJ/m²]
D = 3.6*[0.713,1.039,1.502,2.021,2.408,2.528,2.413,2.138,1.707,1.236,0.818,0.624];
% Diffuse horizontal irradiation [MJ/m²]
phi = 46.07*pi/180;     % latitude [rad]
gamma = 0*pi/180;       % panel azimuth [rad]
beta = 30*pi/180;       % tilt angle [rad]
```

A two-person household in Trento has a flat plate collector installed on a rooftop, with a slope = 30° facing South. The collector has the features summarized in the table below (see also datasheet), and it is fed by an on-off pump, which either provides no mass flow rate (at night, or in those situations in which it could provide no positive power output), or a constant flow rate. The collector is coupled with a storage tank, according to the topology shown in the picture below.



Static analysis

Compute the total thermal energy yield of the collector.

Assumptions:

- inlet water temperature is constant $T_{\text{in}} = 30°C$
- Daily radiation profiles on the collector, for the reference location and the reference collector slope
- Ambient temperature variations during the typical day of a month (note that the collector is installed outside, i.e., it subject to the external ambient temperature)
- The average behaviour of the collector in a month is represented by the response in the 15th day of the month
- Use numerical data from class 17 on PV systems simulation

The first things to be calculated are the the remainang relelvant geometry quantities in order to estimate the amount of energy that reaches Trento.

Declination angle

$$\delta = 23.45 \frac{\pi}{180} \sin\left(2\pi \frac{284+n}{365.25}\right)$$

Sunrise/sunset angle

$$\cos(\omega_s) = -\tan(\phi)\tan(\delta)$$

Sunrise/sunset time

$$t_{\text{sunrise/sunset}} = - + \left(\frac{12}{\pi}\omega_s + 12\right)$$

```
% angle taht represents the position of the sun at noon at the equator
delta = 23.45*pi/180*sin(2*pi*(284+n)/365.25);  % solar declination angle [rad]
omega_s = acos(-tan(phi)*tan(delta));           % sunrise/sunset angle on ground
[rad] --> hour angle. at 0° it's noon
t_sunrise = -12/pi*omega_s+12;                  % sunset time [h]
t_sunset = 12/pi*omega_s+12;                    % sunrise time [h]
```

To determine the daily average irradiation for each month on the panel, some other adjustment factors are to be defined

```
% adjustment factors
a = 0.409+0.5016*sin(omega_s-pi/3);
b = 0.6609-0.4767*sin(omega_s-pi/3);
```

The average energy collected in each month in one year can be calculated, together with the average monthly temperature and the avergae monthly irradiation.

Irradiation calculations

1) Energy falling on an extraterrestrial horizontal surface

Radiation reaching earth:

$$I_{\text{SC}} = 1383 \frac{W}{m^2}$$

Has to be corrected due to **elliptical orbit**:

$$I_0 = I_{SC}\left[1 + 0.034\cos\left(2\pi\ \frac{n}{365.25}\right)\right]$$

Compute the **extraterrestrial radiation on a horizontal plane**:

$$I_{0h} = I_0\cos(\theta_z)$$

where θ_z is the **zenith angle** $\theta_z = \sin(\delta)\sin(\phi) + \cos(\delta)\cos(\phi)\cos(\omega)$

- ω hour angle of sun $\omega = 15\ \frac{\pi}{180}(t - 12)$
- ϕ Trento latitude angle $\phi = 47.07\ \frac{\pi}{180}$

$$I_{0h} = I_{SC}\left[1 + 0.034\cos\left(2\pi\ \frac{n}{365.25}\right)\right](\sin(\delta)\sin(\phi) + \cos(\delta)\cos(\phi)\cos(\omega)) \quad \text{[W/m^2]}$$

To have the **extraterrestrial irradiation on horizontal plane throughout a whole day** integrate I_{0h} in time:

$$\overline{H}_{0h} = \frac{86400}{\pi} I_{SC}\left[1 + 0.034\cos\left(2\pi\ \frac{n}{365.25}\right)\right](\omega_s\sin(\delta)\sin(\phi) + \cos(\delta)\cos(\phi)\cos(\omega))$$

The **extraterrestrial irradiation on horizontal plane throughout a whole day on a tilted plane:**

$$\overline{H}_{0h}(\beta) = \frac{86400}{\pi} I_{SC}\left[1 + 0.034\cos\left(2\pi\ \frac{n}{365.25}\right)\right](\omega_s\sin(\delta)\sin(\phi - \beta) + \cos(\delta)\cos(\phi - \beta)\cos(\omega))$$

2) Relation between global, diffuse and beam components

Assume no radiation is reflected from ground on a horizontal surafce, the Global irradiation $\overline{H}$ has two components:

$$\overline{H} = \overline{B} + \overline{D}$$

- $\overline{B}$ beam irradiation
- $\overline{D}$ diffused irradiation

Note: we now refer to monthly average daily value

$$\overline{B} = \overline{H} - \overline{D}$$

3) Calculation of hourly irradiation values

The average daily irradiation vaues for each month are broken down into hourly values using the ratio r_t and r_d:

$$r_d = \frac{\pi}{24}\frac{\cos\omega - \cos\omega_s}{\sin\omega_s - \omega_s\cos\omega_s}$$

$$r_t = (a + b\cos(\omega))\, r_d$$

Using the correction factors

- $a = 0.409 + 0.5016\sin\left(\omega_s - \frac{\pi}{3}\right)$
- $b = 0.6609 + 0.4767\sin\left(\omega_s - \frac{\pi}{3}\right)$

the hourly total irradiation $\dot{H}$ of the average day of each month is:

$$\overline{H_h} = r_t\,\overline{H}$$

the hourly diffused irradiation $\dot{D}$ of the average day of each month is:

$$\overline{D_h} = r_d\,\overline{D}$$

the hourly beam irradiation $\dot{B}$ of the average day of each month is:

$$\overline{B_h} = \overline{H_h} - \overline{D_h}$$

4) Calculatin of the Beam component on a tilted surface

For a tilted plane we can now define the ratio:

$$\overline{B_h}(\beta) = R_B\,\overline{B_h} \qquad \text{with } R_B = \frac{\overline{H_0(\beta)}}{\overline{H_{0h}}} = \frac{\cos(\theta_i)}{\cos(\theta_z)}$$

5) Calculatin of the Diffused component on a tilted surface

$$\overline{D_h}(\beta) = R_D\,\overline{D_h} \qquad \text{with } R_D = \frac{1+\cos\beta}{2}$$

6) Calculatin of the Reflected component on a tilted surface

$$\overline{A_h}(\beta) = R_A\,\overline{H_h} \qquad \text{with } R_A = \rho\frac{1-\cos\beta}{2}$$

7) Calculatin of the Total irradiation on a tilted surface

$$\overline{H_h}(\beta) = \overline{B_h}(\beta) + \overline{D_h}(\beta) + \overline{A_h}(\beta)$$

Collector efficiency

In order to compute the monthly energy, the efficiency of the panel needs to be "weighted" by the difference between the collector inlet temperature T_{in} and the ambient temperature T_a, which changes based on the season. The formula for the collector efficiency becomes:

$$\eta = \eta_0 - U \frac{T_{\text{in}} - T_a}{H_{\text{bh}}}$$

When $T_{\text{in}} - T_a$ is large (i.e., when the water entering the collector is much warmer than the surrounding air), the collector will lose more heat to the environment, which lowers its efficiency. Meanwhile, when $T_{\text{in}} - T_a$ is small (i.e., the temperature of the water entering the collector is closer to the ambient temperature), there is less heat loss, and the collector operates more efficiently. As a result, the greater the temperature difference between the inlet water and the surrounding air, the more work the collector has to keep the water hot, resulting in a decrease in its efficiency.

Also, the values for η, T_a and H_{bt} for the months of january and june are memorized separately for subsequent calculations.

Thermal power

$$q = \eta \overline{H_\beta} A$$

- η hourly efficiency
- $\overline{H_\beta}$ hourly irradiance on tilted flad collector [W/m^2]
- A area of flat plate collector [m^2]
- q thermal power sent into the tank [W]

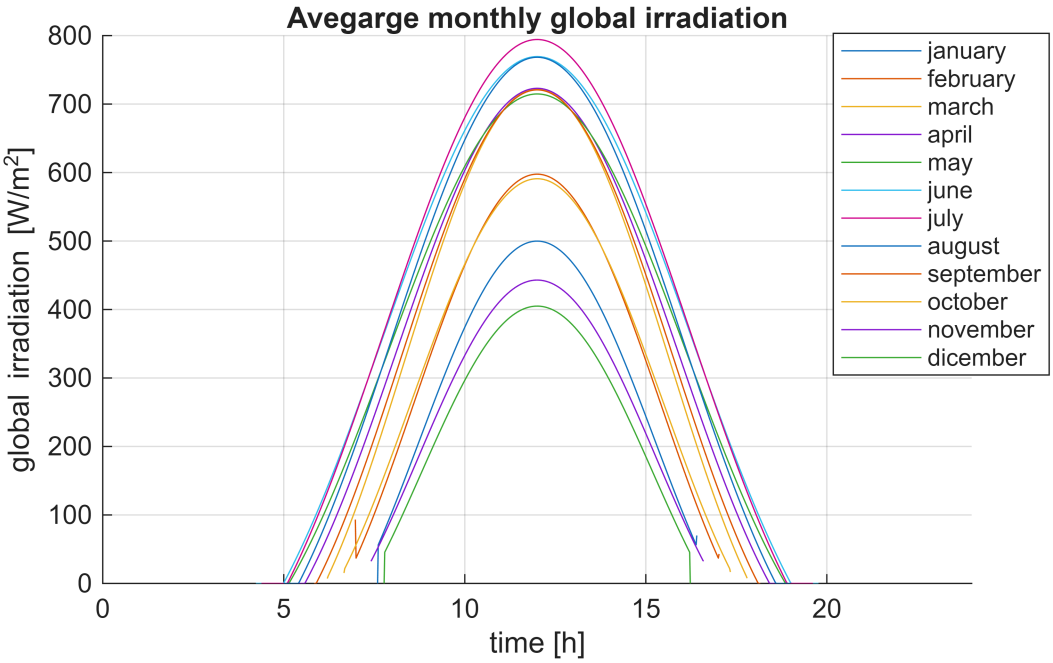
```
E_month = zeros(1,12);
T_a_15_jan = [];
T_a_15_jun = [];
eta_15_jan = [];
eta_15_jun = [];
Hbt_15_jan = [];
Hbt_15_jun = [];

% loop for every 15th day of the month
for i = 1:12
    t = linspace(t_sunrise(i),t_sunset(i),500)';    % daily time interval [h]
    omega = 15*pi/180*(t-12);                       % hour angle [rad]
    dt = t(2)-t(1);                                 % time step [h]

    % ambient temperature profile [°C]
    T_a = (T_max(i)+T_min(i))/2+(T_max(i)-T_min(i))/2*cos(2*pi/24*(t-t_max));

    % relevant angle quantities
      % cosine of the zenith angle
    cosThetaz = cos(omega)*cos(delta(i))*cos(phi)+sin(delta(i))*sin(phi);
      % cosine of the incidence angle
```

```
    cosThetai = cos(beta)*(sin(delta(i))*sin(phi)
+cos(delta(i))*cos(omega)*cos(phi))-sin(beta)*(cos(gamma)*(cos(phi)*sin(delta(i))-
cos(delta(i))*cos(omega)*sin(phi))-cos(delta(i))*sin(gamma)*sin(omega));
    Rb = cosThetai./cosThetaz;

    % hourly radiation components
    rd = pi/24*(cos(omega)-cos(omega_s(i)))/(sin(omega_s(i))-
omega_s(i)*cos(omega_s(i))); % conversion factor for the diffused component of the
irradiation D [-]
    rt = rd.*(a(i)+b(i)*cos(omega)); % conversion factor for the Global irradiation
H. rt  is the average daily irradiation broken down into hours [-]
    Dh = rd*D(i)*1/3600*1e6;         % diffuse component of the irradiation [Wh/m^2]
    Hh = rt*H(i)*1/3600*1e6;         % global radiation (beam + diffused elements)
[Wh/m^2]
    Bh = Hh-Dh;                      % beam radiation [Wh/m^2]

    % global irradiation calculation on the tilted surface
    Bbh = Bh.*Rb;                        % beam irradiation component for a tilted
surface, per hour
    Dbh = Dh.*((1+cos(beta))/2);         % diffused irradiation component
    Abh = ((1-cos(beta))/2)*rho.*Hh;     % albedo component
    Hbh = Bbh + Dbh + Abh;               % global irradiation on the collector
surface [Wh/m^2] or [W/m^2]
    Hbh(Hbh<0|isnan(Hbh)|isinf(Hbh)) = 0;

    % efficiency of the panel w.r.t. the inlet temperature
    eta = eta_0_num-U_num*((T_in_num-T_a)./Hbh);
    eta(eta<0|isnan(eta)|isinf(eta)) = 0;

    % Thermal energy calculation
    q = eta.*Hbh*A_collector;             % instantaneous thermal power [W]
    q(isnan(q)|isinf(q)) = 0;
    E_month(i) = trapz(t,q)*Month_day(i); % Monthly energy [Wh]

    disp(['Month ',num2str(i),': monthly energy = ',num2str(E_month(i)),' [Wh]; max
global irradiation = ',num2str(max(Hbh)),' [W/m^2]']);

    % memorization of the values of january and june
    if i==1
        t_january = t;
        T_a_january = T_a;
        eta_january = eta;
        Hbt_january = Hbh;
    elseif i==6
        t_june = t;
        T_a_june = T_a;
        eta_june = eta;
        Hbt_june = Hbh;
    end
```

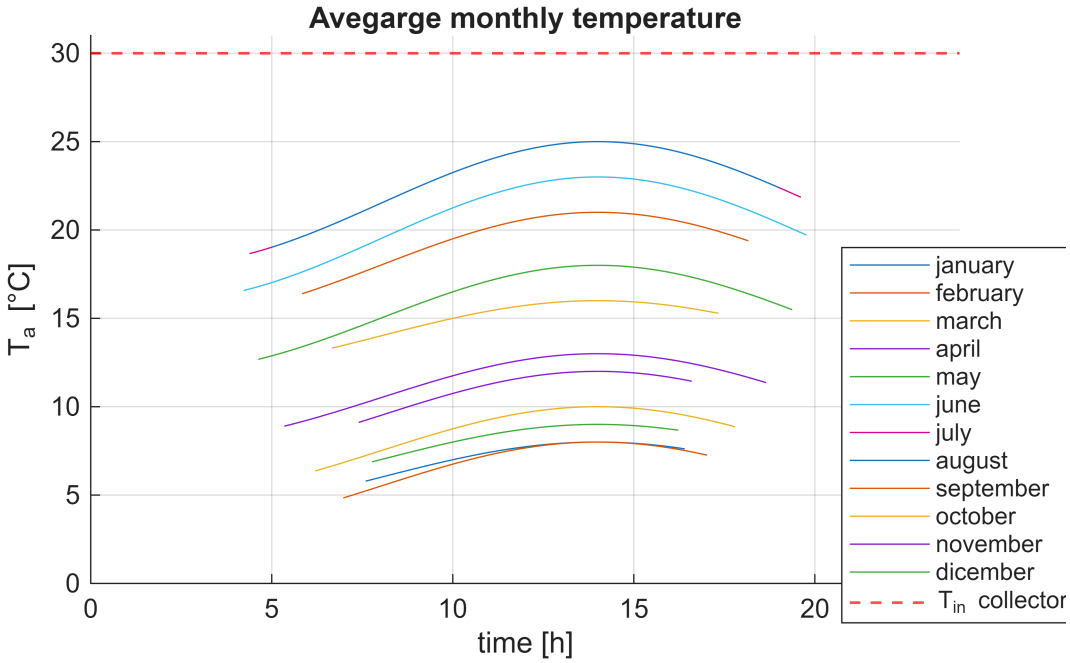
```
    % recursive plots
    % Hourly global irradiation on the collector surface for each month
    figure(1)
    hold on
    grid on
    plot(t,Hbh)

    % Hourly ambient temperature for each month
    figure(2)
    hold on
    grid on
    plot(t,T_a)

    % Hourly efficiency
    figure(3)
    hold on
    grid on
    plot(t,eta)
end
```

```
Month 1: monthly energy = 97460.7852 [Wh]; max global irradiation = 499.879 [W/m^2]
Month 2: monthly energy = 122159.0611 [Wh]; max global irradiation = 597.6333 [W/m^2]
Month 3: monthly energy = 196065.5509 [Wh]; max global irradiation = 722.5697 [W/m^2]
Month 4: monthly energy = 214794.365 [Wh]; max global irradiation = 722.9938 [W/m^2]
Month 5: monthly energy = 249145.5253 [Wh]; max global irradiation = 714.7575 [W/m^2]
Month 6: monthly energy = 286643.2352 [Wh]; max global irradiation = 769.3182 [W/m^2]
Month 7: monthly energy = 310182.1067 [Wh]; max global irradiation = 794.2701 [W/m^2]
Month 8: monthly energy = 285484.1178 [Wh]; max global irradiation = 768.3336 [W/m^2]
Month 9: monthly energy = 227702.8876 [Wh]; max global irradiation = 720.7247 [W/m^2]
Month 10: monthly energy = 158602.2702 [Wh]; max global irradiation = 590.9777 [W/m^2]
Month 11: monthly energy = 86983.4985 [Wh]; max global irradiation = 442.9942 [W/m^2]
Month 12: monthly energy = 69432.1621 [Wh]; max global irradiation = 404.9661 [W/m^2]
```

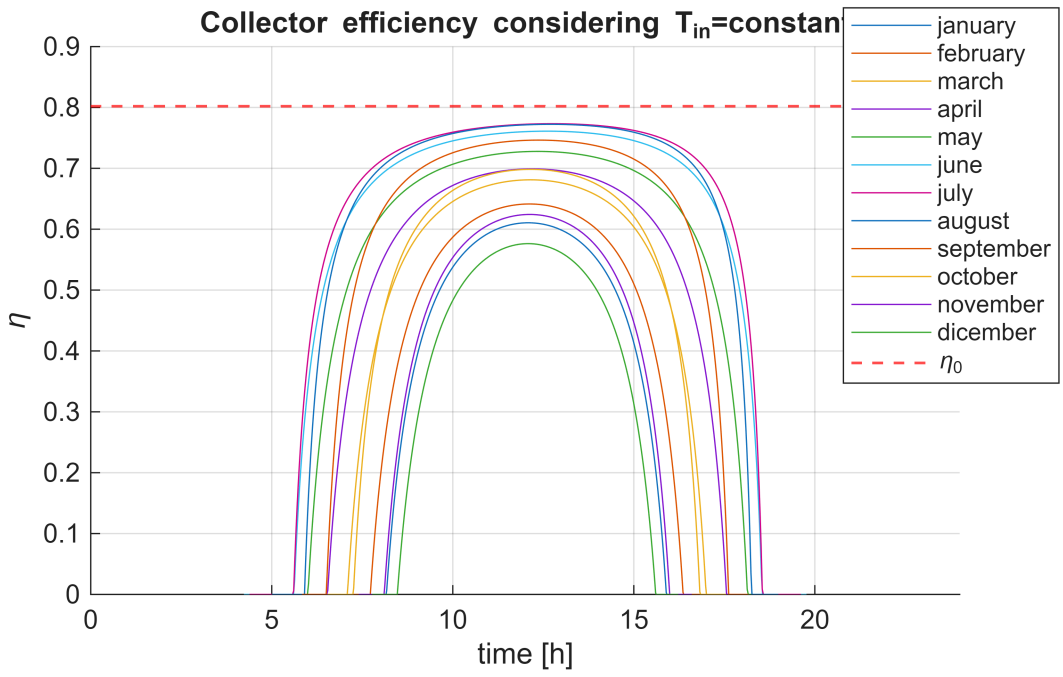
```
figure(1)
xlabel('time [h]')
ylabel('global irradiation [W/m^2]')
xlim([0 24])
title('Avegarge monthly global irradiation')
legend('january','february','march','april','may','june','july','august','september'
,'october','november','dicember','Position',[0.7816 0.4636 0.1929, 0.4226])
hold off
```




```
figure(2)
yline(T_in_num,'Color','r','LineWidth',1,'LineStyle','--');
ylim([0 31])
xlabel('time [h]')
ylabel('T_a [°C]')
xlim([0 24])
title('Avegarge monthly temperature')
legend('january','february','march','april','may','june','july','august','september'
,'october','november','dicember','T_{in} collector','Position',[0.8089 0.1207
0.1929, 0.4226])
hold off
```




```
figure(3)
yline(eta_0_num,'Color','r','LineWidth',1,'LineStyle','--');
xlabel('time [h]')
ylabel('\eta')
xlim([0 24])
title('Collector efficiency considering T_{in}=constant')
legend('january','february','march','april','may','june','july','august','september'
,'october','november','dicember','\eta_0','Position',[0.8011 0.4927 0.1929, 0.4226])
hold off
```



The annual production of energy can now be calculated by summing up all contributions for each month:

```
E_year = sum(E_month);
disp(['Annual energy prduction by the solar panel = ',num2str(E_year),' [Wh]']);
```

```
Annual energy prduction by the solar panel = 2304655.5655 [Wh]
```

Dynamic analysis

Focus on the dynamic behaviour of the system in a couple of relevant prototypical days of the year, the **15th of January** and the **15th of June**.

The profiles of the thermal power consumptions are given as two time series.

A **simulink model** has been developed to model the **system dynamics** (collector + tank + loads).

The **inputs** are:

- H solar radiation
- T_a ambient temperature profile
- $\dot{m}_u$ consumpions profiles

The **outputs** are:

- T tank temperature profile
- T_{in} inlet water temperature in the collector
- T_{out} outlet water temperature from the collector
- $\dot{m}_c$ profiles of the flow rate in the collector
- $\dot{Q}_H$ power profile supplied by the heater

Assume the mains temperature is

- $T_m = 12°C$ in winter
- $T_m = 16°C$ in summer

The notable values for the irradiaation values and the temperature profiles for the 15th of January and the 15th of June have already been extracted in the previous loop.

The efficiency values are not relevant in this simulation as they hve been calculated considering an inlet temperature $T_{\text{in}} = \text{const}$. This assumption cannot be made in a dynamic condition.

The first thing to do is to convert all relevant data series into seconds, as thius is the expected unit for simulink. A time step of 10 seconds is then choosen, as temperature changes are not immediate so a finer resolution is

not needed. The simulation duration has been set for n_{days} days, inorder to visualize the progressive stabilization of the profile of the temperature inside the water tank T_{tank} twards a "staedy-state" value.

Every simulated day is the same as the prototypical choosen 15th day of the month in it's requests from the user, ambient temperature profile and irradiation profile.

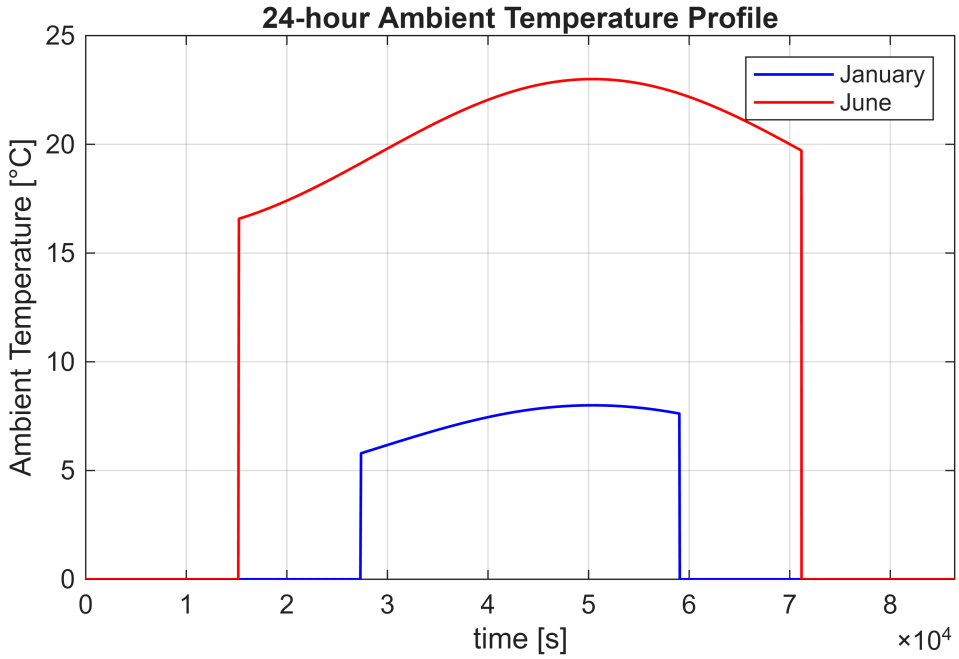
Simulation parameters:

```
resolution = 1;                  % time step duration
t_1day = (0:resolution:24*3600)'; % time vector for 1 day in seconds
n_days = 5;                       % number of days
T = 24*3600*n_days;
t_sim = (0:60:T)';                % 24-hour time vector in [s]
```

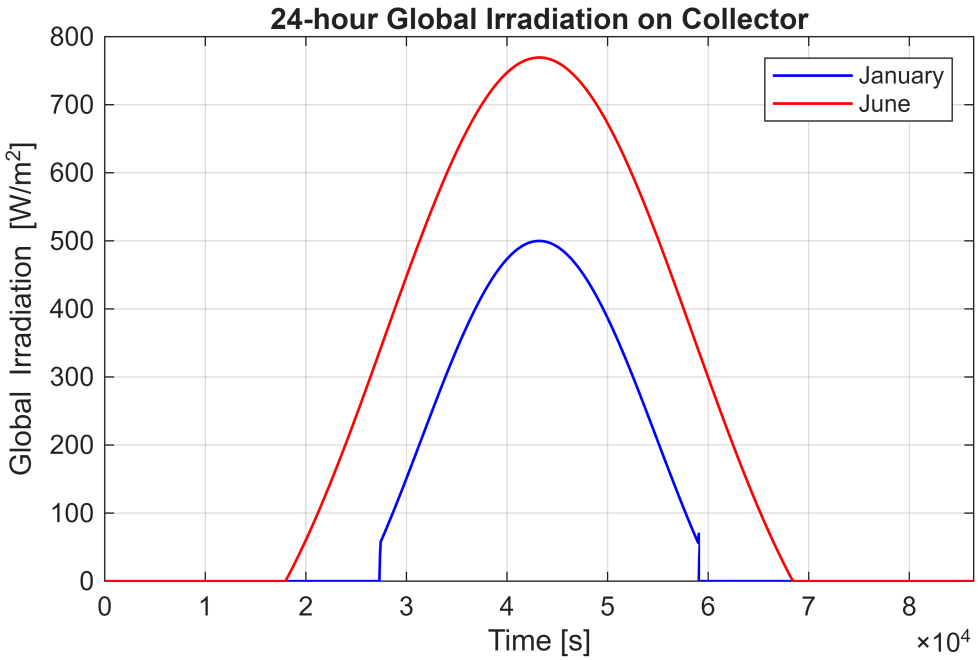
INPUT PROFILE SIGNALS

The relevant profiles also need to be changed into seconds. To do that, an interpolation of the profiles onto the new time vector is performed. tha value 0 is assigned to fill all values not defined in the original profiles

```
% january
t_jan_1day = t_january*3600;  % [h] -> [s]
% interpolate for each second of the day
T_a_jan_1day = interp1(t_jan_1day,T_a_january,t_1day,'linear',0);
Hbt_jan_1day = interp1(t_jan_1day,Hbt_january,t_1day,'linear',0);

% june
t_jun_1day = t_june*3600;
T_a_jun_1day = interp1(t_jun_1day,T_a_june,t_1day,'linear',0);
Hbt_jun_1day = interp1(t_jun_1day,Hbt_june,t_1day,'linear',0);

% plots to verifiy that the signals are consistent with the ones defined previuosly
figure;
plot(t_1day,T_a_jan_1day,'b','LineWidth',1)
hold on
grid on
plot(t_1day,T_a_jun_1day,'r','LineWidth',1)
xlabel('time [s]')
ylabel('Ambient Temperature [°C]')
xlim([0 3600*24])
title('24-hour Ambient Temperature Profile')
legend('January','June')
hold off
```

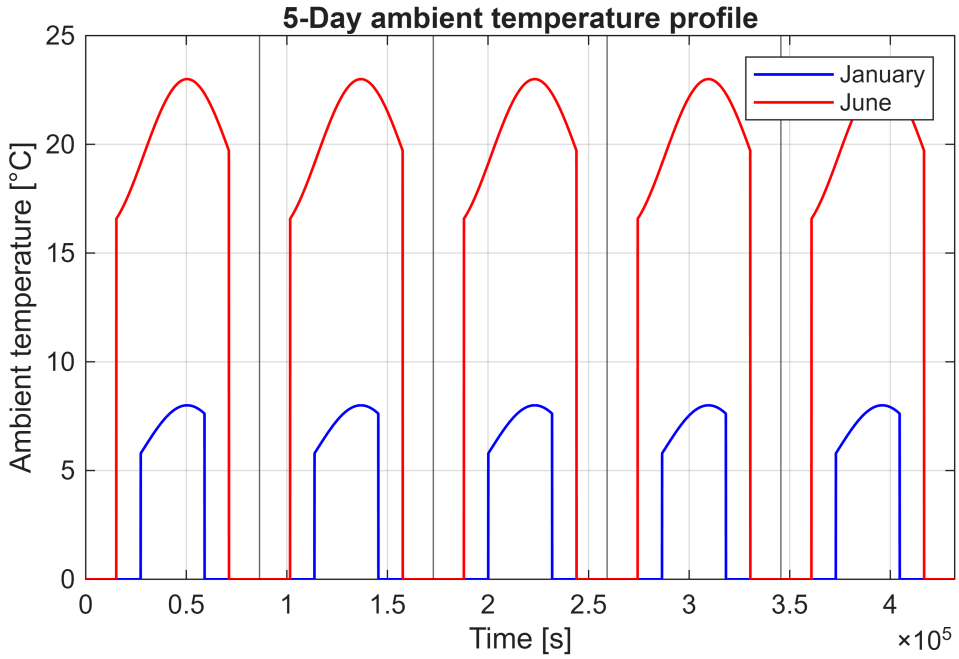



```
figure;
plot(t_1day,Hbt_jan_1day,'b','LineWidth',1)
hold on
grid on
plot(t_1day,Hbt_jun_1day,'r','LineWidth',1)
xlabel('Time [s]')
ylabel('Global Irradiation [W/m^2]')
xlim([0 3600*24])
title('24-hour Global Irradiation on Collector')
legend('January','June')
hold off
```

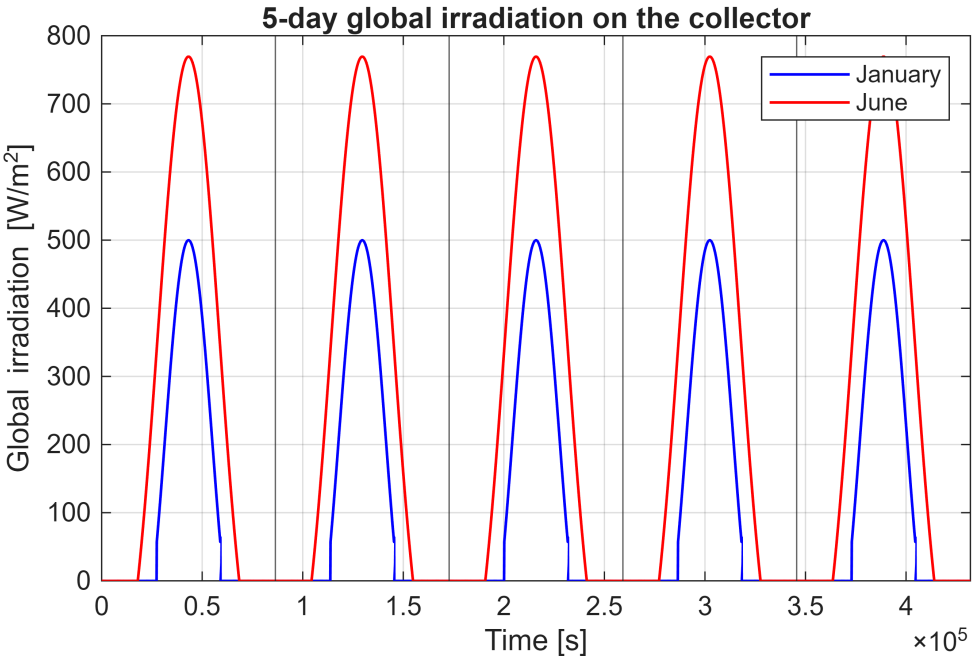


Now, these can be repeated identically for the number of days defined by the variable n_{days}

```
T_a_jan_sim = repmat(T_a_jan_1day,n_days,1);
T_a_jan_sim =
interp1(linspace(0,T,length(T_a_jan_sim)),T_a_jan_sim,t_sim,'linear',0);
Hbt_jan_sim = repmat(Hbt_jan_1day,n_days,1);
Hbt_jan_sim =
interp1(linspace(0,T,length(Hbt_jan_sim)),Hbt_jan_sim,t_sim,'linear',0);

T_a_jun_sim = repmat(T_a_jun_1day,n_days,1);
T_a_jun_sim =
interp1(linspace(0,T,length(T_a_jun_sim)),T_a_jun_sim,t_sim,'linear',0);
Hbt_jun_sim = repmat(Hbt_jun_1day,n_days,1);
Hbt_jun_sim =
interp1(linspace(0,T,length(Hbt_jun_sim)),Hbt_jun_sim,t_sim,'linear',0);

% plot; grey vertical lines at each day have been added to identify the days

% Ambient temperature
figure;
plot(t_sim,T_a_jan_sim,'b','LineWidth',1)
hold on
grid on
plot(t_sim,T_a_jun_sim,'r','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('Time [s]')
ylabel('Ambient temperature [°C]')
xlim([0 T])
title([num2str(n_days),'-Day ambient temperature profile'])
legend('January','June')
hold off
```




```
% sun irradiance on a tilted flat plate
figure;
plot(t_sim,Hbt_jan_sim,'b','LineWidth',1)
hold on
grid on
plot(t_sim,Hbt_jun_sim,'r','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('Time [s]')
ylabel('Global irradiation [W/m^2]')
xlim([0 T])
title([num2str(n_days),'-day global irradiation on the collector'])
legend('January', 'June')
hold off
```



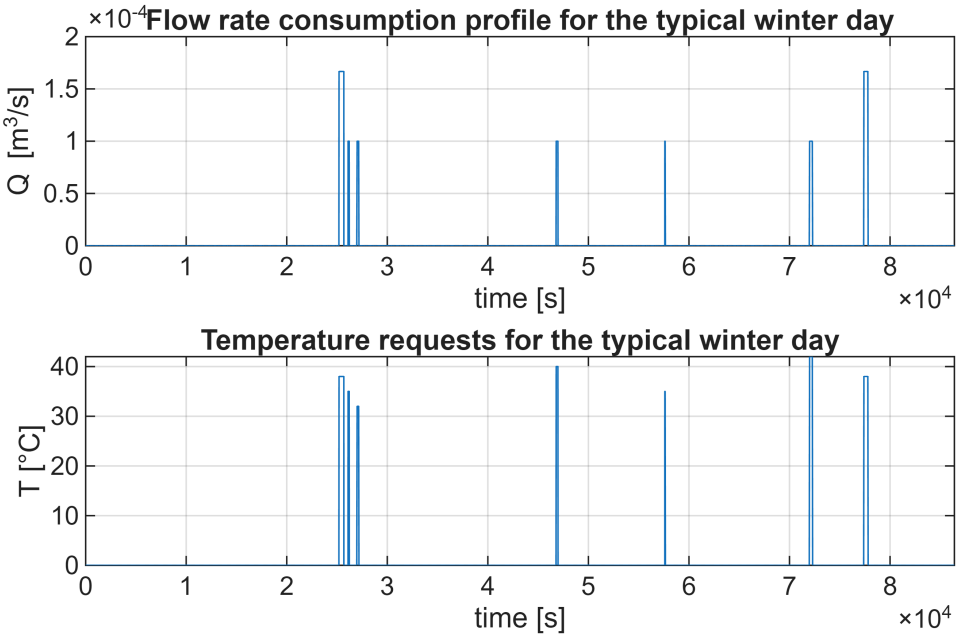
INPUT CONSUMPTION SIGNALS

before any simulation, che 1day-consumption data needs to be imported and converted into seconds for both prototipical usecases (winter and summer)

Winter consumptions

```
% for 1 day
Q_req_w_1day = zeros(length(t_1day),1);     % defined in [m^3/s]
T_req_w_1day = zeros(length(t_1day),1);     % defined in [°C]

Q_req_w_1day(t_1day >= 60*60*7    & t_1day < 60*60*7+60*8)    = (10/60)/1000;  % Q
shower 1 [m^3/s]
T_req_w_1day(t_1day >= 60*60*7    & t_1day < 60*60*7+60*8)    = 38;            % T
shower 1 [°C]
Q_req_w_1day(t_1day >= 60*60*7.25 & t_1day < 60*60*7.25+60*2) = (6/60)/1000;   % Q
face wash 1 [m^3/s]
T_req_w_1day(t_1day >= 60*60*7.25 & t_1day < 60*60*7.25+60*2) = 35;            % T
face wash 1 [°C]
Q_req_w_1day(t_1day >= 60*60*7.5  & t_1day < 60*60*7.5+60*3)  = (6/60)/1000;   % Q
breakfast wash [m^3/s]
T_req_w_1day(t_1day >= 60*60*7.5  & t_1day < 60*60*7.5+60*3)  = 32;            % T
breakfast wash [°C]
Q_req_w_1day(t_1day >= 60*60*13   & t_1day < 60*60*13+60*3)   = (6/60)/1000;   % Q
lunch wash [m^3/s]
T_req_w_1day(t_1day >= 60*60*13   & t_1day < 60*60*13+60*3)   = 40;            % T
lunch wash [°C]
```

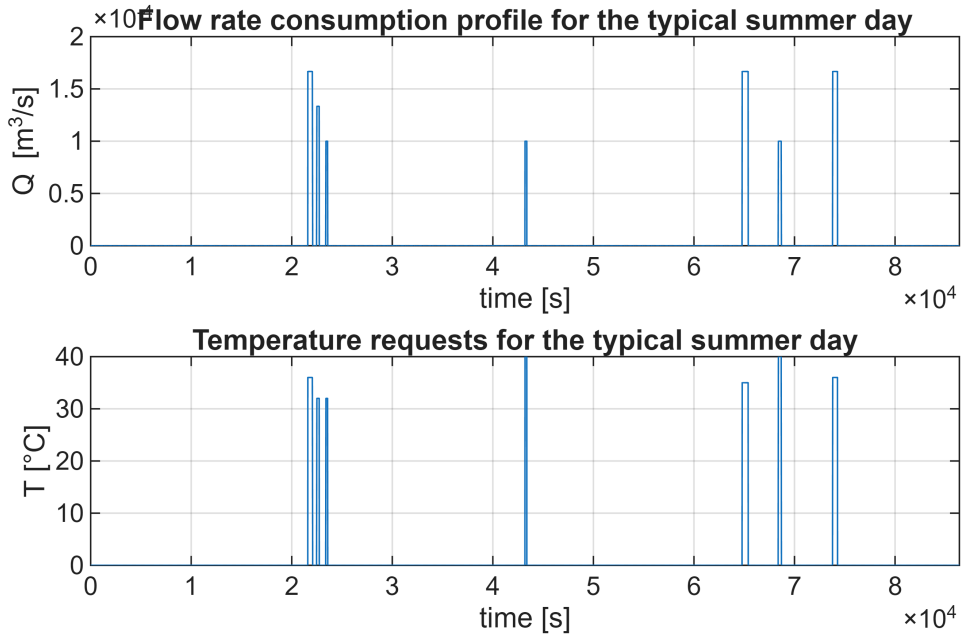
```
Q_req_w_1day(t_1day >= 60*60*16   & t_1day < 60*60*16+60*1)   = (6/60)/1000;   % Q
face wash 2 [m^3/s]
T_req_w_1day(t_1day >= 60*60*16   & t_1day < 60*60*16+60*1)   = 35;            % T
face wash 2 [°C]
Q_req_w_1day(t_1day >= 60*60*20   & t_1day < 60*60*20+60*5)   = (6/60)/1000;   % Q
dinner wash [m^3/s]
T_req_w_1day(t_1day >= 60*60*20   & t_1day < 60*60*20+60*5)   = 42;            % T
dinner wash [°C]
Q_req_w_1day(t_1day >= 60*60*21.5 & t_1day < 60*60*21.5+60*7) = (10/60)/1000;  % Q
shower 2 [m^3/s]
T_req_w_1day(t_1day >= 60*60*21.5 & t_1day < 60*60*21.5+60*7) = 38;            % T
shower 1 [°C]

figure;
subplot(2,1,1)
plot(t_1day,Q_req_w_1day)
hold on
grid on
xlabel('time [s]')
ylabel('Q [m^3/s]')
xlim([0 3600*24])
hold off
title('Flow rate consumption profile for the typical winter day')
subplot(2,1,2)
plot(t_1day,T_req_w_1day)
hold on
grid on
xlabel('time [s]')
ylabel('T [°C]')
xlim([0 3600*24])
hold off
title('Temperature requests for the typical winter day')
```



Summer consumptions

```
Q_req_s_1day = zeros(length(t_1day),1);                                    %
defined in [m^3/s]
T_req_s_1day = zeros(length(t_1day),1);                                    %
defined in [°C]

Q_req_s_1day(t_1day >= 60*60*6 & t_1day < 60*60*6+60*8) = (10/60)/1000;     % Q
shower 1 [m^3/s]
T_req_s_1day(t_1day >= 60*60*6 & t_1day < 60*60*6+60*8) = 36;               % T
shower 1 [°C]
Q_req_s_1day(t_1day >= 60*60*6.25 & t_1day < 60*60*6.25+60*4) = (8/60)/1000; % Q
face wash 1 [m^3/s]
T_req_s_1day(t_1day >= 60*60*6.25 & t_1day < 60*60*6.25+60*4) = 32;         % T
face wash 1 [°C]
Q_req_s_1day(t_1day >= 60*60*6.5 & t_1day < 60*60*6.5+60*3) = (6/60)/1000;  % Q
breakfast wash [m^3/s]
T_req_s_1day(t_1day >= 60*60*6.5 & t_1day < 60*60*6.5+60*3) = 32;           % T
breakfast wash [°C]
Q_req_s_1day(t_1day >= 60*60*12 & t_1day < 60*60*12+60*3) = (6/60)/1000;    % Q
lunch wash [m^3/s]
T_req_s_1day(t_1day >= 60*60*12 & t_1day < 60*60*12+60*3) = 40;             % T
lunch wash [°C]
Q_req_s_1day(t_1day >= 60*60*18 & t_1day < 60*60*18+60*10) = (10/60)/1000;  % Q
face wash 2 [m^3/s]
T_req_s_1day(t_1day >= 60*60*18 & t_1day < 60*60*18+60*10) = 35;            % T
face wash 2 [°C]
Q_req_s_1day(t_1day >= 60*60*19 & t_1day < 60*60*19+60*5) = (6/60)/1000;    % Q
dinner wash [m^3/s]
```

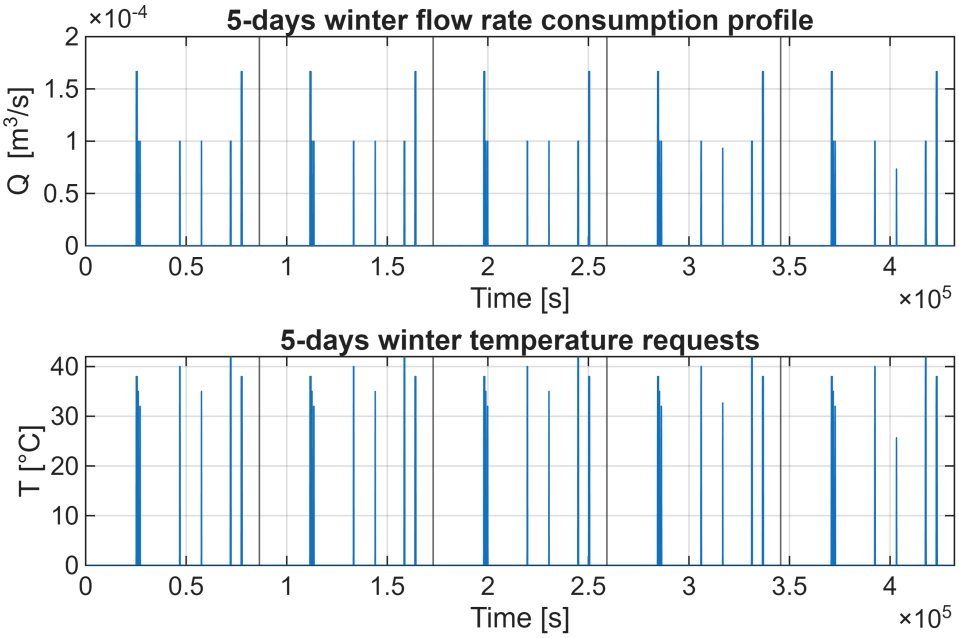
```
T_req_s_1day(t_1day >= 60*60*19 & t_1day < 60*60*19+60*5) = 40;                % T
dinner wash [°C]
Q_req_s_1day(t_1day >= 60*60*20.5 & t_1day < 60*60*20.5+60*8) = (10/60)/1000;  % Q
shower 2 [m^3/s]
T_req_s_1day(t_1day >= 60*60*20.5 & t_1day < 60*60*20.5+60*8) = 36;            % T
shower 2 [°C]

figure;
subplot(2,1,1)
plot(t_1day,Q_req_s_1day)
hold on
grid on
xlabel('time [s]')
ylabel('Q [m^3/s]')
xlim([0 3600*24])
hold off
title('Flow rate consumption profile for the typical summer day')
subplot(2,1,2)
plot(t_1day,T_req_s_1day)
hold on
grid on
xlabel('time [s]')
ylabel('T [°C]')
xlim([0 3600*24])
hold off
title('Temperature requests for the typical summer day')
```



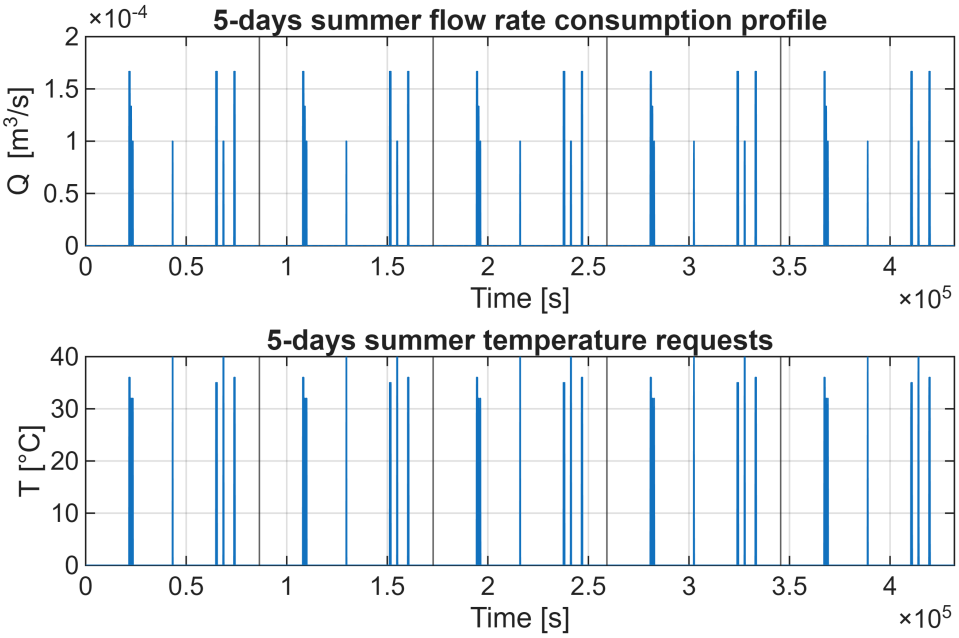
Now, these signals can be repeated for n_{days}

For the simulation purposes, we resample the signals to have a time space from 1s to 60s.

```
Q_req_w_sim = repmat(Q_req_w_1day,n_days,1);
Q_req_w_sim =
interp1(linspace(0,T,length(Q_req_w_sim)),Q_req_w_sim,t_sim,'linear',0);
T_req_w_sim = repmat(T_req_w_1day,n_days,1);
T_req_w_sim =
interp1(linspace(0,T,length(T_req_w_sim)),T_req_w_sim,t_sim,'linear',0);

Q_req_s_sim = repmat(Q_req_s_1day,n_days,1);
Q_req_s_sim =
interp1(linspace(0,T,length(Q_req_s_sim)),Q_req_s_sim,t_sim,'linear',0);
T_req_s_sim = repmat(T_req_s_1day,n_days,1);
T_req_s_sim =
interp1(linspace(0,T,length(T_req_s_sim)),T_req_s_sim,t_sim,'linear',0);

% Winter Plots
figure;
subplot(2,1,1)
plot(t_sim,Q_req_w_sim)
hold on
grid on
for i = 1:n_days-1
    xline(i*24*3600)
end
xlim([0 T])
xlabel('Time [s]')
ylabel('Q [m^3/s]')
title([num2str(n_days),'-days winter flow rate consumption profile'])
hold off
subplot(2,1,2)
plot(t_sim,T_req_w_sim)
hold on
grid on
for i = 1:n_days-1
    xline(i*24*3600)
end
xlim([0 T])
xlabel('Time [s]')
ylabel('T [°C]')
title([num2str(n_days),'-days winter temperature requests'])
hold off
```




```
% Summer Plots
figure;
subplot(2,1,1)
plot(t_sim,Q_req_s_sim)
hold on
grid on
for i = 1:n_days-1
    xline(i*24*3600)
end
xlim([0 T])
xlabel('Time [s]')
ylabel('Q [m^3/s]')
title([num2str(n_days),'-days summer flow rate consumption profile'])
subplot(2,1,2)
plot(t_sim,T_req_s_sim)
hold on
grid on
for i = 1:n_days-1
    xline(i*24*3600)
end
xlim([0 T])
xlabel('Time [s]')
ylabel('T [°C]')
title([num2str(n_days),'-days summer temperature requests'])
```



MODEL EQUATIONS

The equations to be implemented in the model are the following:

ALGEBRAIC EQUATIONS 1 & 2

$$\begin{cases} \dot{m_c}\, c_p\, (T_{\text{out}} - T_{\text{in}}) = \eta\, I(t)\, A_{\text{collector}} \\ T_{\text{in}} = T_{\text{out}} - \epsilon\, (T_{\text{out}} - T_{\text{tank}}) \end{cases}$$

DIFFERENTIAL EQUATION 3

$$\dot{m_c}\, c_p\, (T_{\text{out}} - T_{\text{in}}) - \dot{Q}_L - h\, A\, (T_{\text{tank}} - T_{\text{cellar}}) = M_{\text{tank}}\, c_p\, \dot{T_{\text{tank}}}$$

1. decribes the heat energy gained by the collector, where T_{in} is the temperature of the water from the tank to the collector and T_{out} is the temperature from the collector to the tank
2. describes the heat exchanger temperature behaviour according to the collector effectiveness
3. describes the overall thermal balance of the system, the components:

   $\dot{m_c}\, c_p\, (T_{\text{out}} - T_{\text{in}})$ → heat from collector to tank $\dot{Q}_L$ → requested load from the user $h\, A\, (T_{\text{tank}} - T_{\text{cellar}})$ → dissipated power due to tank convection

   $M_{\text{tank}}\, c_p\, \dot{T_{\text{tank}}}$ → tank energy changing over time

We differentiate between two cases for **load power request:**

$$\dot{Q}_L = \begin{cases} \dot{m_{\text{req}}}\, c_p\, (T_{\text{req}} - T_{\text{mains}}) & \text{if } T_{\text{req}} \le T_{\text{tank}} \\ \dot{m_{\text{req}}}\, c_p\, (T_{\text{tank}} - T_{\text{mains}}) & \text{otherwise} \end{cases} \qquad \text{EQUATION 4}$$

- if the Requested Temperature is lower than the Tank Temperature, the tank energy variation is the one needed to warm up the water fomr mains up to the Requested Temperaure. (all the heat is provided from the tank)
- if the Requested Temperature is higher than the Tank Temperature, the tank energy variation is the one needed to warm up the water form mains up to the Tank temperature. (the share that covers the range $T_{\text{tank}} - T_{\text{mains}}$, because the other share will be covered by the boiler)

We differentiate between two cases for the **heater power request:**

It's the law that dictates the amount of power that the auxiliary heater needs based on the required temperature by the user, and it is complementary to the heat delivered from the tank

$$\dot{Q}_H = \begin{cases} 0 & \text{if } T_{\text{req}} \leq T_{\text{tank}} \\ \dot{m}_{\text{req}}\, c_p\, (T_{\text{req}} - T_{\text{tank}}) & \text{otherwise} \end{cases} \qquad \text{EQUATION 5}$$

- if the Requested Temperature is lower than the Tank Temperature, the heater is switched off. (all the heat is provided from the tank)
- if the Requested Temperature is higher than the Tank Temperature, the heater provides the heat share that the tank cannot deliver. (the share that covers the range $T_{\text{req}} - T_{\text{tank}}$)

Two cases for pump that makes the **water flow circulate through the collector**:

implies that no water is made to flow into the solar collector by the pump if there is no irradiation or the efficiency is negative.

$$\dot{m_c} = \begin{cases} 0 & \text{if } \eta(T_{\text{in}}) \leq 0 \text{ or } I(t) \leq 0 \\ Q\, \rho_{\text{water}} & \text{otherwise} \end{cases} \qquad \text{EQUATION 6}$$

Other considerations:

- T_{tank} will never be lower than T_{mains} because the water tank is connected directly to the mains.
- T_{in} and T_{out} will never have values below T_{tank}, as the opposite would mean that the collector expended enegy rather than collecting it
- $T_{\text{in}} \leq T_{\text{out}}$ as for the same reason
- all temperatures must be $\geq 0\,^{\circ}C$ and $\leq 100\,^{\circ}C$

In order to simplify the model, equations 1,2,3 and 6 can be solved algebrically in the variables T_{in}, T_{out}, η and m_c.

The implementation logic is the following:

- if $I(t) > 0 \quad \longrightarrow$ equations can be solved analitically considering $\dot{m_c} = Q\, \rho_{\text{water}}$

```
syms m_dot_c c_p T_out T_in eta T_tank I_t A_c Q rho_w U T_amb eta_0 epsilon

eq1 = m_dot_c*c_p*(T_out - T_in) == eta*I_t*A_c;
eq2 = T_in == T_out-epsilon*(T_out - T_tank);
eq3 = eta == eta_0-U*(T_in - T_amb)/I_t;
eq4 = m_dot_c == Q*rho_w;

sol = solve([eq1,eq2,eq3,eq4],[T_in,T_out,eta,m_dot_c]);
T_in_sol = sol.T_in
```

```
T_in_sol =
```

$$\frac{A_c\,T_{\text{amb}}\,U + A_c\,I_t\,\eta_0 - A_c\,T_{\text{amb}}\,U\,\varepsilon - A_c\,I_t\,\varepsilon\,\eta_0 + Q\,T_{\text{tank}}\,c_p\,\varepsilon\,\rho_w}{A_c\,U - A_c\,U\,\varepsilon + Q\,c_p\,\varepsilon\,\rho_w}$$

```
T_out_sol = sol.T_out
```

```
T_out_sol =
```

$$\frac{A_c\,T_{\text{amb}}\,U + A_c\,I_t\,\eta_0 - A_c\,T_{\text{tank}}\,U\,\varepsilon + Q\,T_{\text{tank}}\,c_p\,\varepsilon\,\rho_w}{A_c\,U - A_c\,U\,\varepsilon + Q\,c_p\,\varepsilon\,\rho_w}$$

```
eta_sol = sol.eta
```

```
eta_sol =
```

$$\frac{Q\,c_p\,\varepsilon\,\rho_w\,(T_{\text{amb}}\,U - T_{\text{tank}}\,U + I_t\,\eta_0)}{I_t\,(A_c\,U - A_c\,U\,\varepsilon + Q\,c_p\,\varepsilon\,\rho_w)}$$

```
m_dot_c_sol = sol.m_dot_c
```

```
m_dot_c_sol =
```

$Q\,\rho_w$

Having extracted the solutions, a second verification can be made on the value of η $\longrightarrow$

$$\begin{cases} \text{if } \eta \leq 0, \text{then all equations need to be recalulated using } \dot{m_c} = 0, T_{\text{in}} = T_{\text{out}} = T_{\text{tank}} \\ \text{if } \eta > 0, \text{the the solutions hold} \end{cases}$$

- if $I(t) \leq 0$ $\longrightarrow$ equations can be solved analitically considering $\dot{m_c} = 0, T_{\text{in}} = T_{\text{out}} = T_{\text{tank}}$ and $\eta = 0$.

For the initial state of the temperature of the tank, the vale $T_{\text{tank}} = T_{\text{in}} = 30\,^{\circ}C$ has been considered.

SIMULATION

```
% winter data
Hbt_w_data = [t_sim,Hbt_jan_sim];
T_a_w_data = [t_sim,T_a_jan_sim];
Q_req_w_data = [t_sim,Q_req_w_sim];
T_req_w_data = [t_sim,T_req_w_sim];

%summer data
Hbt_s_data = [t_sim,Hbt_jun_sim];
```

```
T_a_s_data = [t_sim,T_a_jun_sim];
Q_req_s_data = [t_sim,Q_req_s_sim];
T_req_s_data = [t_sim,T_req_s_sim];

T_tank_w_0 = 22.81; % [°C]
T_tank_s_0 = 33.66; % [°C]

% sim_1 = sim('recs_HW2_luca')
sim_1 = sim('SP_DHW_1')
```

```
sim_1 =
  Simulink.SimulationOutput:

                  Hbt_s: [1x1 timeseries]
                  Hbt_w: [1x1 timeseries]
          Q_dot_heater_s: [1x1 timeseries]
          Q_dot_heater_w: [1x1 timeseries]
            Q_dot_load_s: [1x1 timeseries]
            Q_dot_load_w: [1x1 timeseries]
                 Q_req_s: [1x1 timeseries]
                 Q_req_w: [1x1 timeseries]
                  T_in_s: [1x1 timeseries]
                  T_in_w: [1x1 timeseries]
                 T_out_s: [1x1 timeseries]
                 T_out_w: [1x1 timeseries]
                 T_req_s: [1x1 timeseries]
                 T_req_w: [1x1 timeseries]
                T_tank_s: [1x1 timeseries]
                T_tank_w: [1x1 timeseries]
                   eta_s: [1x1 timeseries]
                   eta_w: [1x1 timeseries]
               m_c_dot_s: [1x1 timeseries]
               m_c_dot_w: [1x1 timeseries]
                    tout: [432001x1 double]

      SimulationMetadata: [1x1 Simulink.SimulationMetadata]
            ErrorMessage: [0x0 char]
```

RESULTS

Results can be visualized and contextualized.

```
% WINTER RESULTS
I_w_values = sim_1.Hbt_w.Data;
I_w_t = sim_1.Hbt_w.Time;
Q_dot_heater_w_values = sim_1.Q_dot_heater_w.Data;
Q_dot_heater_w_t = sim_1.Q_dot_heater_w.Time;
Q_dot_load_w_values = sim_1.Q_dot_load_w.Data;
Q_dot_load_w_t = sim_1.Q_dot_load_w.Time;
Q_req_w_values = sim_1.Q_req_w.Data;
Q_req_w_t = sim_1.Q_req_w.Time;
T_in_w_values = sim_1.T_in_w.Data;
T_in_w_t = sim_1.T_in_w.Time;
T_out_w_values = sim_1.T_out_w.Data;
T_out_w_t = sim_1.T_out_w.Time;
T_req_w_values = sim_1.T_req_w.Data;
```

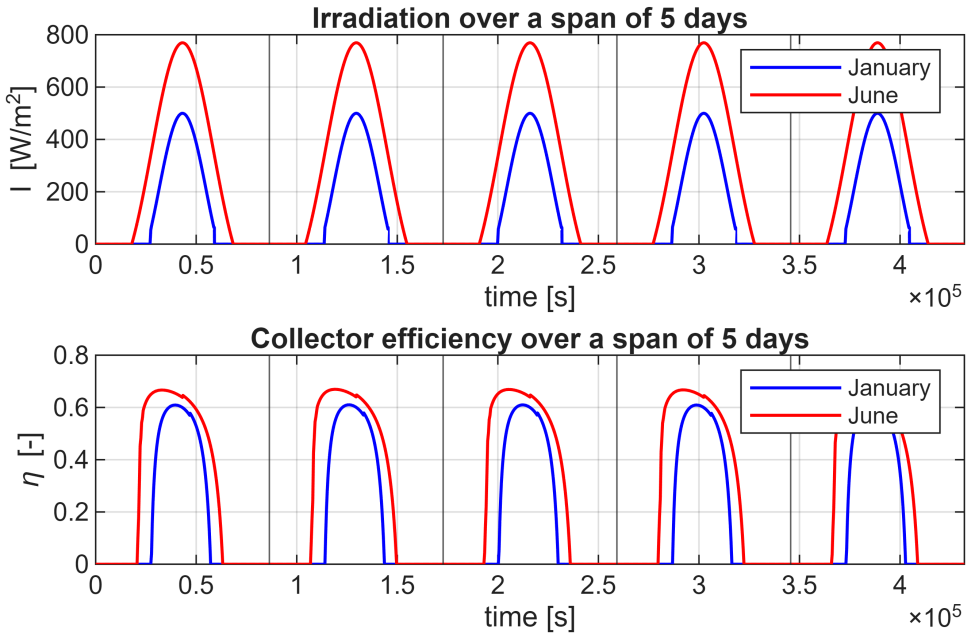
```
T_req_w_t = sim_1.T_req_w.Time;
T_tank_w_values = sim_1.T_tank_w.Data;
T_tank_w_t = sim_1.T_tank_w.Time;
eta_w_values = sim_1.eta_w.Data;
eta_w_t = sim_1.eta_w.Time;
m_c_dot_w_values = sim_1.m_c_dot_w.Data;
m_c_dot_w_t = sim_1.m_c_dot_w.Time;

% SUMMER RESULTS
I_s_values = sim_1.Hbt_s.Data;
I_s_t = sim_1.Hbt_s.Time;
Q_dot_heater_s_values = sim_1.Q_dot_heater_s.Data;
Q_dot_heater_s_t = sim_1.Q_dot_heater_s.Time;
Q_dot_load_s_values = sim_1.Q_dot_load_s.Data;
Q_dot_load_s_t = sim_1.Q_dot_load_s.Time;
Q_req_s_values = sim_1.Q_req_s.Data;
Q_req_s_t = sim_1.Q_req_s.Time;
T_in_s_values = sim_1.T_in_s.Data;
T_in_s_t = sim_1.T_in_s.Time;
T_out_s_values = sim_1.T_out_s.Data;
T_out_s_t = sim_1.T_out_s.Time;
T_req_s_values = sim_1.T_req_s.Data;
T_req_s_t = sim_1.T_req_s.Time;
T_tank_s_values = sim_1.T_tank_s.Data;
T_tank_s_t = sim_1.T_tank_s.Time;
eta_s_values = sim_1.eta_s.Data;
eta_s_t = sim_1.eta_s.Time;
m_c_dot_s_values = sim_1.m_c_dot_s.Data;
m_c_dot_s_t = sim_1.m_c_dot_s.Time;
```

Firstly, the results about **irradiation and efficiency** can be visualized.

- irradiation profile do not change
- the efficiency curve depends from T_{in}, T_a, $I(t)$, it have a deformation in the morning, because the irradiation grows faster than the temperature difference $T_{\text{in}} - T_a$

```
figure;
subplot(2,1,1)
plot(I_w_t,I_w_values,'b','LineWidth',1)
hold on
grid on
plot(I_s_t,I_s_values,'r','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('I [W/m^2]')
xlim([0 T])
legend('January','June')
```

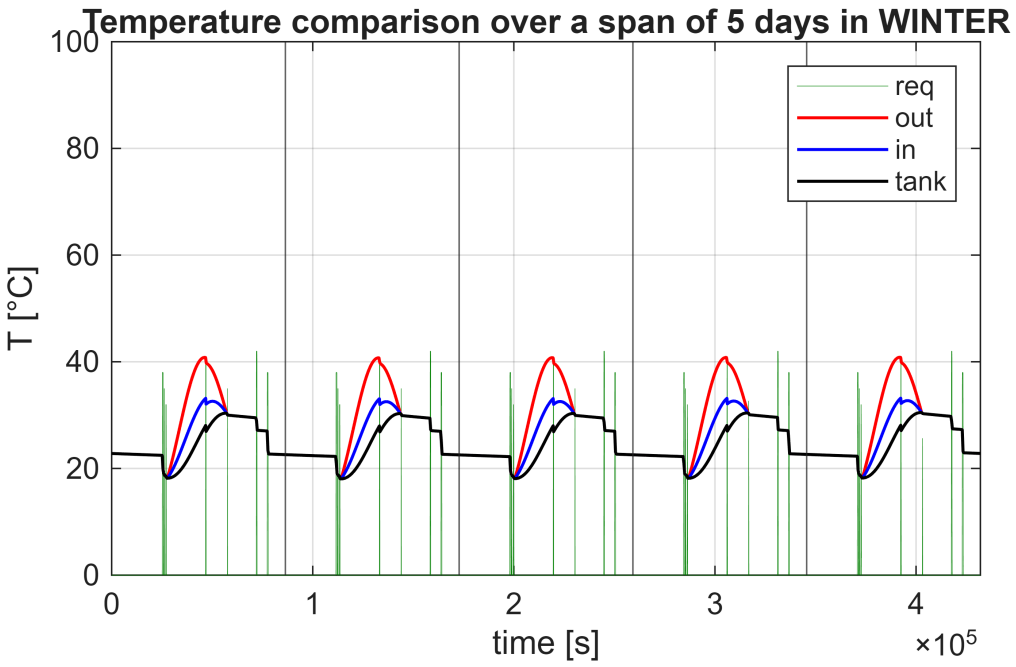
```
title(['Irradiation over a span of ',num2str(n_days),' days'])
hold off
subplot(2,1,2)
plot(eta_w_t,eta_w_values,'b','LineWidth',1)
hold on
grid on
plot(eta_s_t,eta_s_values,'r','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('\eta [-]')
xlim([0 T])
legend('January','June')
title(['Collector efficiency over a span of ',num2str(n_days),' days'])
hold off
```



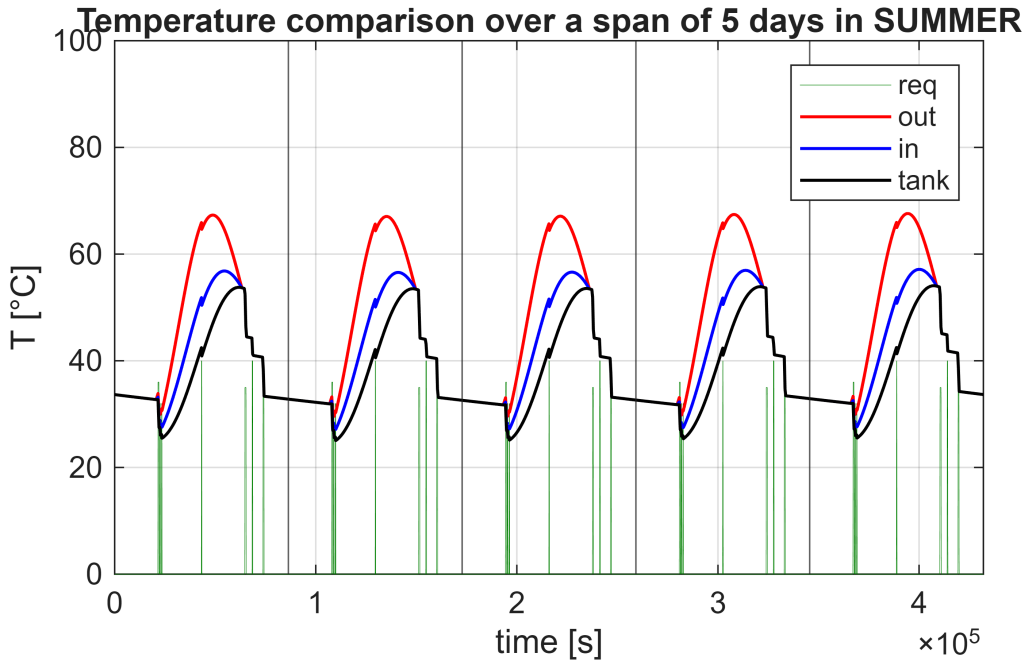
A second comparison can be made on the temperatures $T_{\text{in}}, T_{\text{out}}$ and T_{tank}. All these temperatures should start from an equal point, and evolve accordingly to the required usage patterns defined by T_{req}

```
figure;
plot(T_req_w_t,T_req_w_values,'Color',[0 0.5 0],'LineWidth',0.1)
hold on
grid on
plot(T_out_w_t,T_out_w_values,'r','LineWidth',1)
plot(T_in_w_t,T_in_w_values,'b','LineWidth',1)
plot(T_tank_w_t,T_tank_w_values,'k','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
```

```
end
xlabel('time [s]')
ylabel('T [°C]')
xlim([0 T])
ylim([0 100])
legend('req','out','in','tank')
title(['Temperature comparison over a span of ',num2str(n_days),' days in WINTER'])
hold off
```




```
figure;
plot(T_req_s_t,T_req_s_values,'Color',[0 0.5 0],'LineWidth',0.1)
hold on
grid on
plot(T_out_s_t,T_out_s_values,'r','LineWidth',1)
plot(T_in_s_t,T_in_s_values,'b','LineWidth',1)
plot(T_tank_s_t,T_tank_s_values,'k','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('T [°C]')
xlim([0 T])
ylim([0 100])
legend('req','out','in','tank')
title(['Temperature comparison over a span of ',num2str(n_days),' days in SUMMER'])
hold off
```



After ~ 3 days, it can be seen that the whole system reaches what can be considered as a steady-state solution. Profiles happear to be periodic

The steady state solution can be found by conidering the vey last temperature on the tank:

```
disp(['Steady state solution WINTER: ',num2str(T_tank_w_values(end)),' °C']);
```

```
Steady state solution WINTER: 22.8159 °C
```

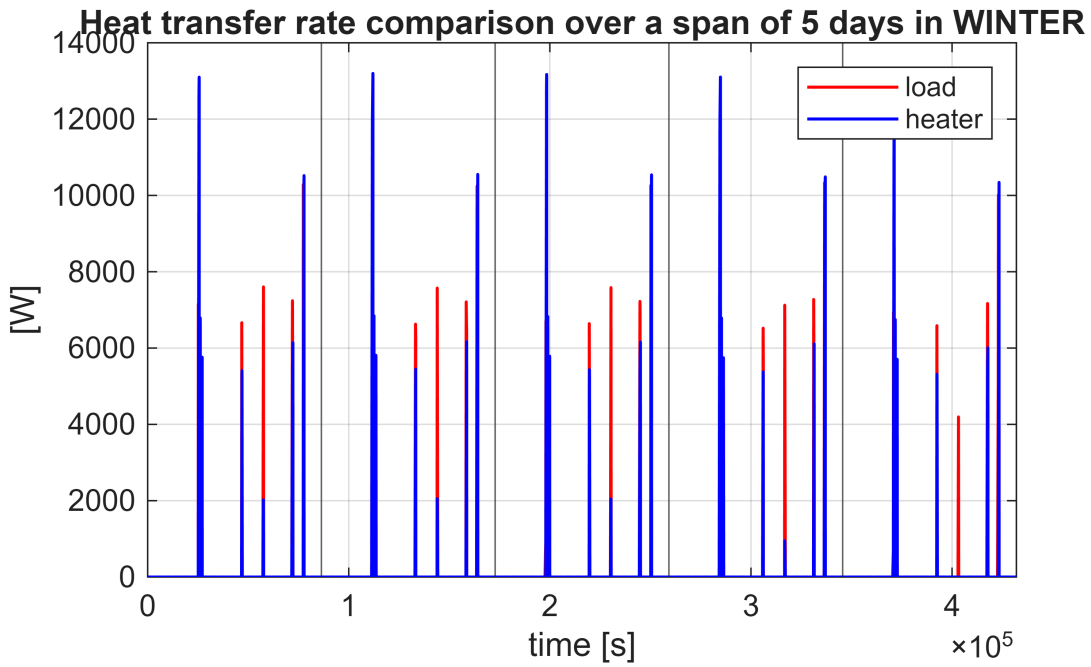
```
disp(['Steady state solution SUMMER: ',num2str(T_tank_s_values(end)),' °C']);
```

```
Steady state solution SUMMER: 33.666 °C
```

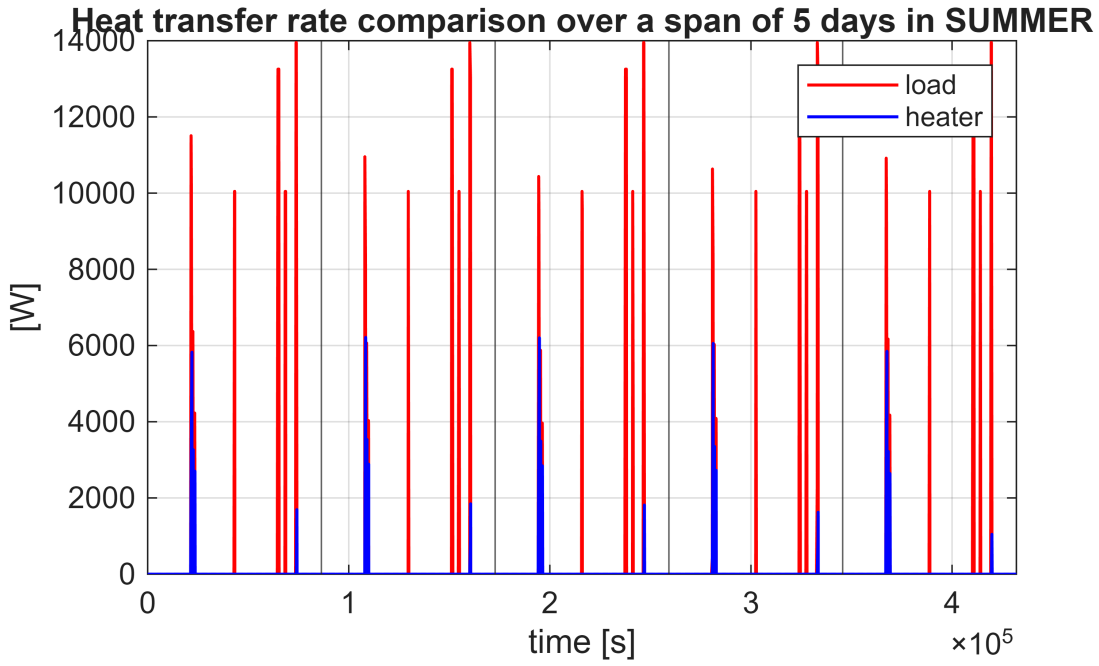
Another comparison can be made by the heat transfer rates required by the load, in each moment the sum of the power delivered by the tank and the power delivered by the heater is equal to the power heat requested by the user.

```
figure;
plot(Q_dot_load_w_t,Q_dot_load_w_values,'r','LineWidth',1)
hold on
grid on
plot(Q_dot_heater_w_t,Q_dot_heater_w_values,'b','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('[W]')
xlim([0 T])
legend('load','heater')
```

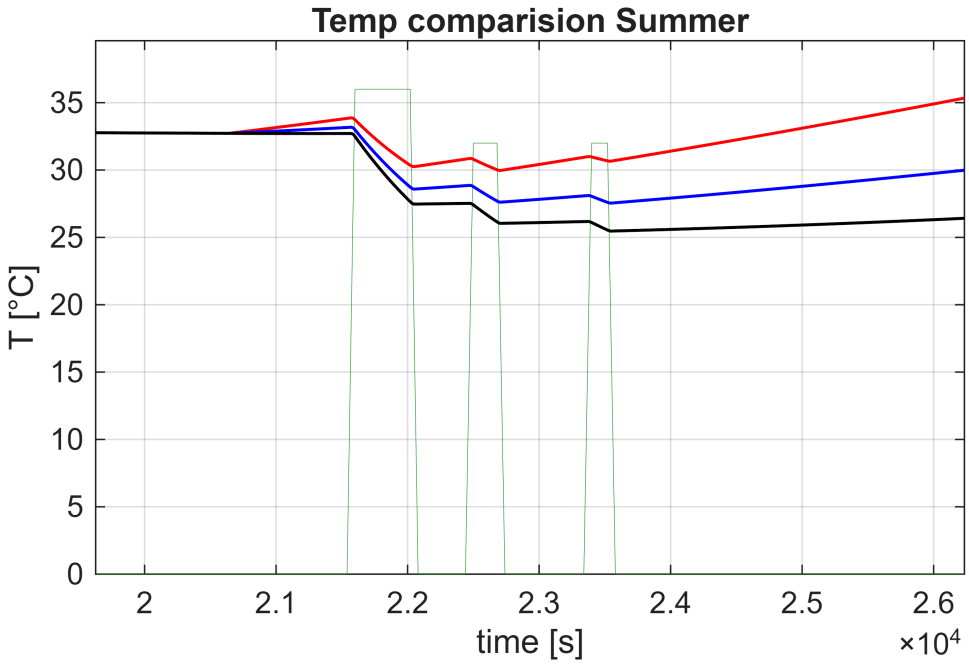
```
title(['Heat transfer rate comparison over a span of ',num2str(n_days),' days in
WINTER'])
hold off
```



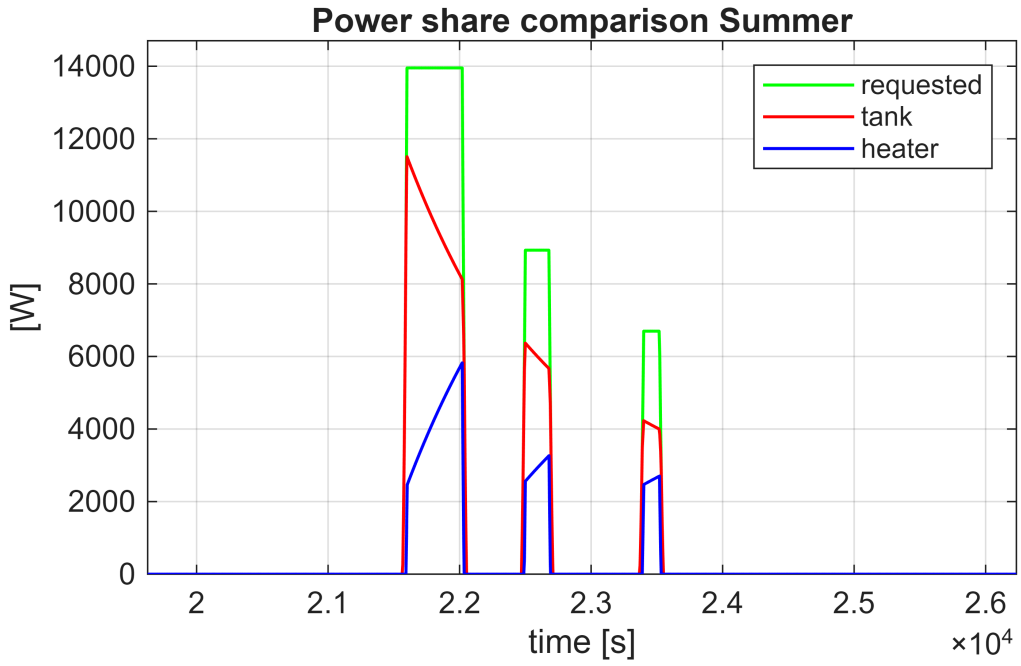

```
figure;
plot(Q_dot_load_s_t,Q_dot_load_s_values,'r','LineWidth',1)
hold on
grid on
plot(Q_dot_heater_s_t,Q_dot_heater_s_values,'b','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('[W]')
xlim([0 T])
legend('load','heater')
title(['Heat transfer rate comparison over a span of ',num2str(n_days),' days in
SUMMER'])
hold off
```




```
% Zoom in (summer)

figure;
plot(T_req_s_t,T_req_s_values,'Color',[0 0.5 0],'LineWidth',0.1)
hold on
grid on
plot(T_out_s_t,T_out_s_values,'r','LineWidth',1)
plot(T_in_s_t,T_in_s_values,'b','LineWidth',1)
plot(T_tank_s_t,T_tank_s_values,'k','LineWidth',1)
xlim([19628 26234])
ylim([0.0 39.6])
xlabel('time [s]')
ylabel('T [°C]')
title('Temp comparision Summer')
hold off
```



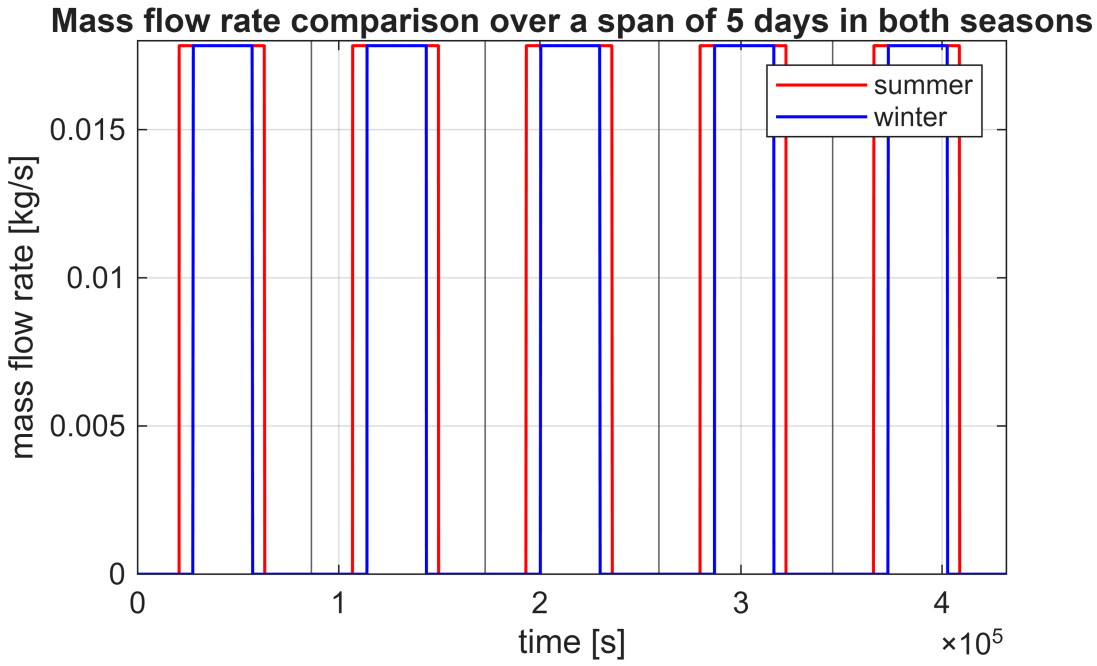

```
figure;
plot(Q_dot_heater_w_t,Q_dot_heater_s_values+Q_dot_load_s_values,'g','LineWidth',1)
hold on
grid on
plot(Q_dot_load_w_t,Q_dot_load_s_values,'r','LineWidth',1)
plot(Q_dot_heater_w_t,Q_dot_heater_s_values,'b','LineWidth',1)
xlim([19628 26234])
ylim([0 14703])
xlabel('time [s]')
ylabel('[W]')
legend('requested','tank','heater')
title('Power share comparison Summer')
hold off
```



As a last thing, the mass flow rate across the solar array can be plotted

The flow rate in witer last less, because of minor hour of light and the panel efficiency is reached slightly later.

```
figure;
plot(m_c_dot_s_t,m_c_dot_s_values,'r','LineWidth',1)
hold on
grid on
plot(m_c_dot_w_t,m_c_dot_w_values,'b','LineWidth',1)
for i = 1:n_days-1
    xline(i*24*3600)
end
xlabel('time [s]')
ylabel('mass flow rate [kg/s]')
xlim([0 T])
legend('summer','winter')
title(['Mass flow rate comparison over a span of ',num2str(n_days),' days in both
seasons'])
hold off
```



Use the dynamic model to determine what shares of the thermal energy demand are respectively covered by the solar collector and the heater in the two days.

the variuos contributes to the overall energy of the system can be determined using integration over the whole simulation time of n_{days}

```
% total energy provided by the collector
% E_collector_w = trapz(eta_w_t,eta_w_values.*I_w_values);
% E_collector_s = trapz(eta_s_t,eta_s_values.*I_s_values);

% total energy from the external heater
E_heater_w = trapz(Q_dot_heater_w_t,Q_dot_heater_w_values);
E_heater_s = trapz(Q_dot_heater_s_t,Q_dot_heater_s_values);

% total load energy
E_load_w = trapz(Q_dot_load_w_t,Q_dot_load_w_values);
E_load_s = trapz(Q_dot_load_s_t,Q_dot_load_s_values);

% total system energy
E_w = E_heater_w + E_load_w; % E_collector_w
E_s = E_heater_s + E_load_s; % E_collector_s
```

Now the shares can be calculated as percentages

```
share_heater_w =  E_heater_w/E_w*100;
share_heater_s =  E_heater_s/E_s*100;
% share_tank_w = E_collector_w/E_w*100;
% share_tank_s = E_collector_s/E_s*100;
share_tank_w = E_load_w/E_w*100;
share_tank_s = E_load_s/E_s*100;
```

```
fprintf(['The share of the power provided by the external heater during \nWINTER
days is %.3f %% of the total energy of the system\n\n' ...
         'The share of the power provided by the tank-collector system during
\nWINTER days is, %.3f %% of the total energy of the system\n\n' ...
         'The share of the power provided by the external heater during \nSUMMER
days is, %.3f %% of the total energy of the system\n\n' ...
         'The share of the power provided by the tank-collector system during
\nSUMMER days is, %.3f %% of the total energy of the system\n'], ...
         share_heater_w,share_tank_w,share_heater_s,share_tank_s);
```

```
The share of the power provided by the external heater during
WINTER days is 53.599 % of the total energy of the system

The share of the power provided by the tank-collector system during
WINTER days is, 46.401 % of the total energy of the system

The share of the power provided by the external heater during
SUMMER days is, 10.543 % of the total energy of the system

The share of the power provided by the tank-collector system during
SUMMER days is, 89.457 % of the total energy of the system
```

Clearly the heater has a more important role in Winter.

Perform a sensitivity analysis of the system behaviour (temperature profiles, consumptions, etc.) with respect to the tank volume, the external ambient temperature, and the heat exchanger effectiveness factor.

The goal of this analysis is to see how things change when different parameters are modified in their values. The parameters that can be modified are $M_{\text{tank}}, \text{hA}, \epsilon, T_{\text{amb}}, T_{\text{in}}$

The trials have been perfomed in simulink only under a new file name. Relevant values can be modified directly into that file, aside from new external temperature profiles tant can be modified below. simulations are again performed on n_{days}

```
% REFERENCE TEMPERATURE PROFILE

% T_min_sa = [4,3,5,8,12,16,18,18,15,12,7,5];     % Min temperature [°C]
% T_max_sa = [8,8,10,13,18,23,25,25,21,16,12,9];  % maximum ambient temperature per
each month[°C]
% t_max_sa = 14;                                  % time at which the max
temperature achived during the day [h]
% for i = [1,6]
%     t = linspace(t_sunrise(i),t_sunset(i),500)';
%     Ta_sa = (T_max_sa(i)+T_min_sa(i))/2+(T_max_sa(i)-T_min_sa(i))/2*cos(2*pi/
24*(t-t_max_sa));
%     if i == 1
%         T_a_w_sa = Ta_sa;
%         t_w_sa = t;
%     end
%     if i == 6
```

```
%         T_a_s_sa = Ta_sa;
%         t_s_sa = t;
%     end
% end

% NEW FLATTER TEMPERATURE PROFILE
% T_min_sa = [6,5,7,10,14,18,20,20,17,14,10,8];
% T_max_sa = [10,10,12,15,20,24,26,26,23,18,14,11];
% t_max_sa = 14;
% ampl = 0.5;
% for i = [1,6]
%     t = linspace(t_sunrise(i),t_sunset(i),500)';
%     Ta_sa = (T_max_sa(i)+T_min_sa(i))/2+ampl*(T_max_sa(i)-T_min_sa(i))/2*cos(2*pi/
24*(t-t_max_sa));
%     if i == 1
%         T_a_w_sa = Ta_sa;
%         t_w_sa = t;
%     end
%     if i == 6
%         T_a_s_sa = Ta_sa;
%         t_s_sa = t;
%     end
% end

% NEW HIGHER-VARIATION TEMPERATURE PROFILE
T_min_sa = [-3,-2,0,5,10,15,20,20,15,10,5,0];
T_max_sa = [5,8,15,22,30,35,40,40,30,22,15,8];
t_max_sa = 14;
for i = [1,6]
    t = linspace(t_sunrise(i),t_sunset(i),500)';
    Ta_sa = (T_max_sa(i)+T_min_sa(i))/2+(T_max_sa(i)-T_min_sa(i))/2*cos(2*pi/24*(t-
t_max_sa));
    if i == 1
        T_a_w_sa = Ta_sa;
        t_w_sa = t;
    end
    if i == 6
        T_a_s_sa = Ta_sa;
        t_s_sa = t;
    end
end

t_w_sa_1day = t_w_sa*3600;
Ta_w_sa_1day = interp1(t_w_sa_1day,T_a_w_sa,t_1day,'linear',0);
Ta_w_sa_sim = repmat(Ta_w_sa_1day,n_days,1);
Ta_w_sa_sim =
interp1(linspace(0,T,length(Ta_w_sa_sim)),Ta_w_sa_sim,t_sim,'linear',0);

t_s_sa_1day = t_s_sa*3600;
Ta_s_sa_1day = interp1(t_s_sa_1day,T_a_s_sa,t_1day,'linear',0);
```

```
Ta_s_sa_sim = repmat(Ta_s_sa_1day,n_days,1);
Ta_s_sa_sim =
interp1(linspace(0,T,length(Ta_s_sa_sim)),Ta_s_sa_sim,t_sim,'linear',0);

Ta_w_sa_data = [t_sim,Ta_w_sa_sim];
Ta_s_sa_data = [t_sim,Ta_s_sa_sim];
```

Other data to be changed

```
M_tank_sa = 50;          % nominal value 200
hA_sa = 20;              % nominal value 2.5
epsilon_num_sa = 0.9;    % nominal value 0.6
T_in_num_sa = 30;        % nominal value 30
```

Sensitivity analysis simulation

```
sim_sa = sim('SP_DHW_1_sens_a')
```

```
sim_sa =
  Simulink.SimulationOutput:

                 Hbt_s: [1x1 timeseries]
                 Hbt_w: [1x1 timeseries]
        Q_dot_heater_s: [1x1 timeseries]
        Q_dot_heater_w: [1x1 timeseries]
          Q_dot_load_s: [1x1 timeseries]
          Q_dot_load_w: [1x1 timeseries]
               Q_req_s: [1x1 timeseries]
               Q_req_w: [1x1 timeseries]
                T_in_s: [1x1 timeseries]
                T_in_w: [1x1 timeseries]
               T_out_s: [1x1 timeseries]
               T_out_w: [1x1 timeseries]
               T_req_s: [1x1 timeseries]
               T_req_w: [1x1 timeseries]
              T_tank_s: [1x1 timeseries]
              T_tank_w: [1x1 timeseries]
                 eta_s: [1x1 timeseries]
                 eta_w: [1x1 timeseries]
             m_c_dot_s: [1x1 timeseries]
             m_c_dot_w: [1x1 timeseries]
                  tout: [432001x1 double]

    SimulationMetadata: [1x1 Simulink.SimulationMetadata]
          ErrorMessage: [0x0 char]
```

CHANGING ϵ (heat exchanger effectiveness)

By changing the value of ϵ the overall difference between T_{in} and T_{out} increases significantely. This allows the tank to absorb more heat coming from the solar collector and increase it temperature. This results in a reduced reliance on the external heater. The steady-state temperature of the tank is around 20°C (for $\epsilon = 0.9$). The opposite behaviour is obtained when the value of ϵ decreases.

CHANGING hA (tank loss coefficent)

By increasing the value of hA (ex. $\mathrm{hA} = 50$), the temeperature of the tank has much sharper declines even if solar efficicency is increased. This is due to the hirgher loss rate that the tank exprerienices. Other quantities such ar internal temperatures are not modified. Due to the higher loss coefficent, the external heater is engaged more often.

The opposite result is obtained when $\mathrm{hA} \approx 0$, as the temperature in the tank stays constant when no load is applied.

CHANGING T_{in}

By substantially rising (ex. $T_{\mathrm{in}} = 60\,^{\circ}C$) the initial tank temperature, the initial contribution of the external heater is reduced to 0; the efficiency of the solar collector is also reduced. However, all quantites seem to be stabilized to values similar to the orginal case of study after $\approx$ 3 days.

On the contrary, if the temperature is reduced and posed $\approx T_{\mathrm{mains}}$, the contribution of the external heater will be every time activated. The system will reach the steady-state condition much faster.

CHANGING M_{tank}

By lowering the mass/volume of the tank (ex. $M_{\mathrm{tank}} = 20$), T_{tank} variates much more quickly in both heatin up and cooling down, as there is less "thermal inertia". However, variations leas to reduced efficiency and a more prominent utilization of the external heater for all load requests longer than $\approx$ 1 min

On the opposite side, if the tank is enlarged, the inertia can be seen much more clearly. However, large volumes at high temperatures allow to satisfy the load without using the external heater as much.

Also, the T_{tank} curve is more regular but takes longer to reach a steady-state solution.

CHANGING T_{amb}

Different profiles can be implemented, with much bigger and smaller differences between max and min values. changes are not immediate