Modeling of Physical Systems

Heat transfer simulation

Dominik Katszer

27 March 2018

1 Aim of laboratory

Create numerical simulation of heat transfer in a plate. Compare resutls for different materials and compare simulation result with mathematically calculated result.

2 Algorithm

Algorithm divide plate to $N \times N$ grid. $T_{i,j}^{n}$ holds temperature value. It is 3 dim matrix, where first 2 dimensions are responisble for node location on the plate and 3rd describes time.

$$T_{i,j}^{n} = T_{i,j}^{n-1} + \frac{K\Delta t}{c_w \rho (\Delta x)^2} \left[T_{i-1,j}^{n-1} + T_{i+1,j}^{n-1} + T_{i-1,j+1}^{n-1} - 4T_{i,j}^{n-1} \right]$$

where

- n – timestep number
- i, j – node coordinates
- K – thermal conductivity coefficient of material
- Δt – timestep size
- c_w – specific heat of material
- ρ – material density
- Δx – distance between nodes

2.1 Parameters

3 Boundary Condition 1

Part of plate which has contact with heater is constant during simulation and is equal $80[C]$, while the edge of plate has also constant temperature but equal $10[C]$.

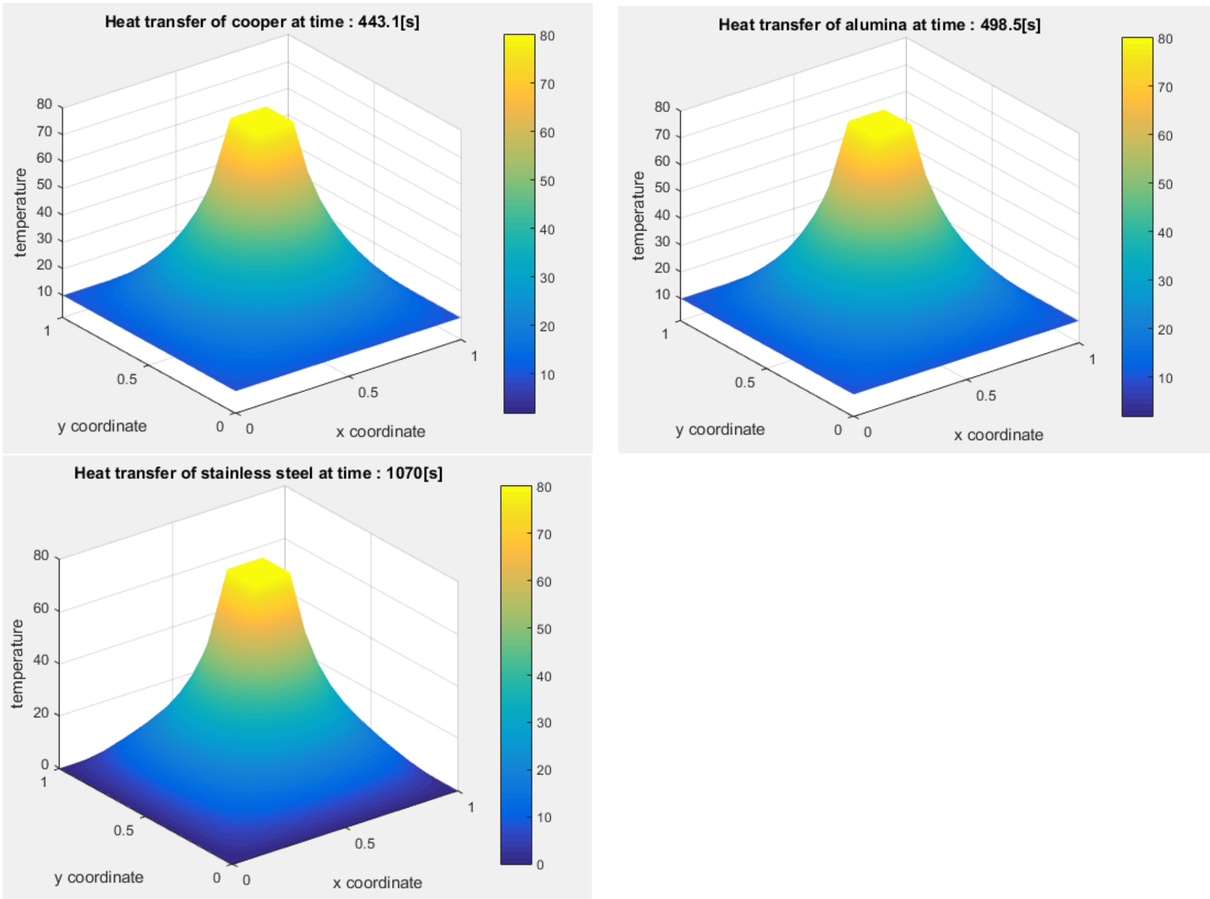

Figure 1: Simulation states at the time of stabilization for different materials

Comparing plots we can observe that time needed for stabilization was the highest for stainless steel material. It means that heat propagation in this material is the worst among 3 materials used in simulation. On the other hand cooper's heat propagation is the best, but only little better than alumina.

4 Boundary Condition 2

This boundary bondition assumes that heater works only for first 10 seconds and then shutdown. Power of heater is constant and equal $100[W]$. What is more, edges are thermally isolated from the environment. Temperature on the whole plate is $20[C]$.

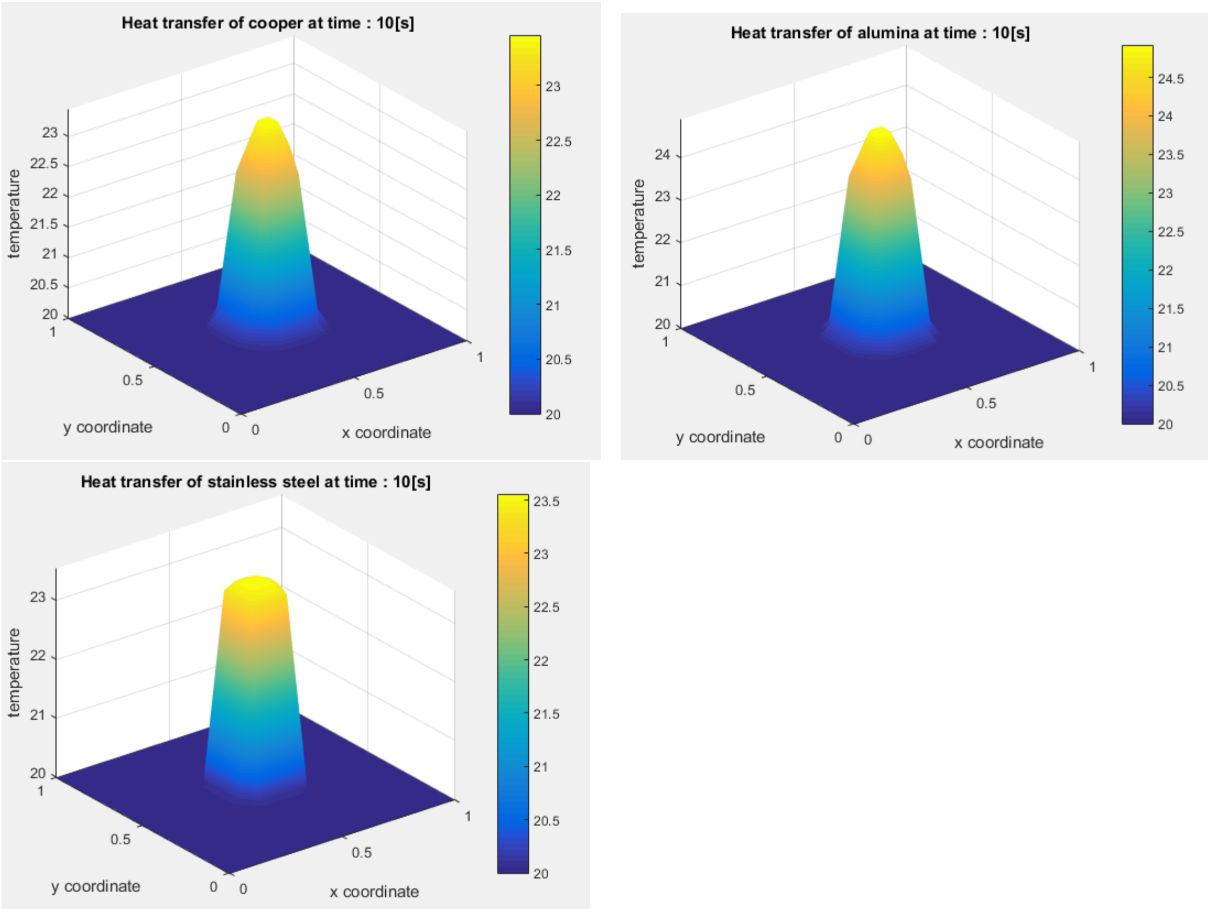

Figure 2: Simulation states at the time of heater shutdown for different materials

In this figure we can observe how material reacts during heating.

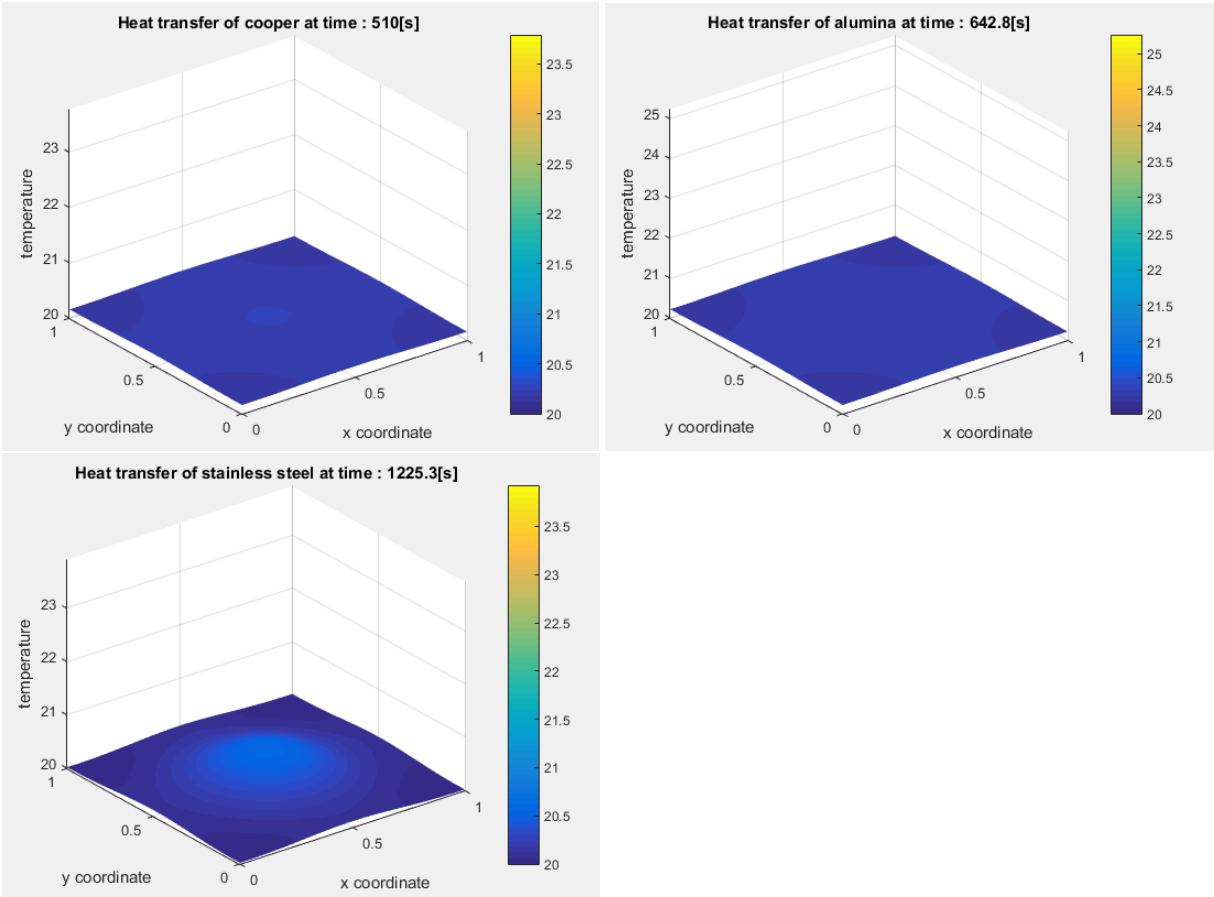

Figure 3: Simulation states at the time of stabilization for different materials

Here we can see how much time is needed for uniform heat distribution on the plate for specific material. Results could be better if the acceptable error was smaller, but then time needed for stabilization would be greater. As we could assume the worst material for heat distribution is stainless steel.

This plots also show how much even temperature of the plate will grow. Unfortunatelly, for used set of data change is very small ($\Delta T \approx 0.176$) so it is barely noticeable.

4.1 Comparison with theorethical temperature change

The heat delivered during heating is expressed by the following equation

$$Q = c_w * m * \Delta T$$

where:

- Q - Heat
- c_w - specific heat of the plate material
- ΔT - difference in heat

We want to know how much temperature changed so we need to calculate ΔT

$$\Delta T = \frac{Q}{C_w * m}$$

substituting $Q = P * t$ and $m = A^2 * h * \rho$ we get

$$\Delta T = \frac{P * t}{C_w * A^2 * h * \rho}$$

where:

- P - power of the heater
- A^2 - plate area
- h - plate thickness
- ρ - plate's material density

Using the data used in simulation for cooper we can calculate theoretical value of ΔT

- $P = 100[W]$
- $t = 10[s]$
- $C_w = 380[\frac{J}{Kg*C}]$
- $A = 1[m]$ - plate area
- $h = 0.002[m]$ - plate thickness
- $\rho = 8920[\frac{Kg}{m^3}]$ - plate's material density

The result is

$$\Delta T \approx 0.148[C]$$

Adding ΔT to initial plate's temperature $T_0 = 20$ and comparing to simulation result $T_0 + \Delta T_{sim} = 20.176$ we can say that values are not the same because of numerical errors and choosen precision for stopping the simulation. What is more calculating it by simulation is much more time consuming, however, sometimes for more complex situations, simulation seems to be the best way of calculating.Difference in values is acceptable.

5 Numerical stability

For the simulation presented in the figure below, I have used alumina material, $\delta t = 0.4$, $\delta x = 0.01$. It is easy to observe that it is not acceptable result, it is coused by used equastion which for this values is not stable.

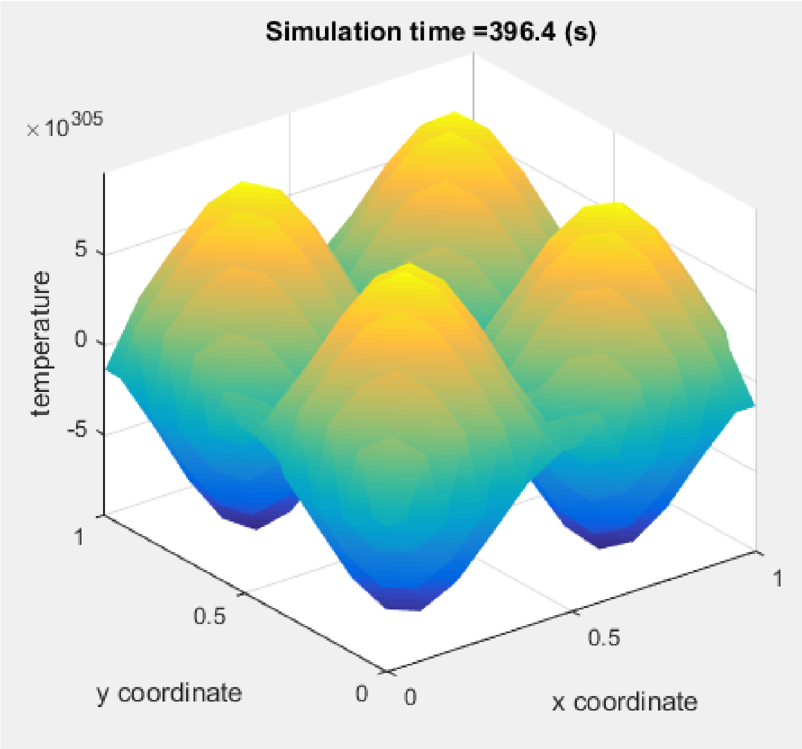

Figure 4: Numerical oscilation

6 Conclusions

We have simulated heat transfer on the plate. It is great way to see how it changes in time. This is a big advantage. What is more as it was proved, it is possible to get results very similar to theoretical calculations. We also observed how plate's material matters, and how different values can be.

7 Source code

Code is divided into sections responsible for data, first boundary , second boundary, numerical stability. What is more I extracted functions into another files which also are attached.

```
clear;
clc;

STEPS = 3000;
%Different plate's materials
data_alumina = struct('density',2700,'heat',900,'conductivity',237);
data_cooper = struct('density',8920,'heat',380,'conductivity',401);
data_stainless_steel = struct('density',7860,'heat',450,'conductivity
    ↪ ',58);
materials = [data_cooper,data_alumina,data_stainless_steel];
materials_names = {'cooper', 'alumina', 'stainless steel'};

data_heater = struct('edge',0.2,'const_temp',80,'power',100,'working_time
    ↪ ',10);
data_plate = struct('edge',1,'const_edge_temp',10, 'init_temp',20, '
    ↪ thickness',0.002);
% distance per one step. If edge = 1 and step_distance is 0.05 then it
    ↪ will
% be divided into 20 pieces.
step_distance = 0.05;

heater_size = data_heater.edge / step_distance;
plate_size = data_plate.edge / step_distance ;

%top left corner of heater in 2d view
heater_location = floor((plate_size - heater_size) / 2) + 1;
heater_in_plate = heater_location:(heater_location+heater_size-1);

%set init temp of plate everywhere
plate(1:plate_size, 1:plate_size, 1:STEPS) = data_plate.init_temp; %
    ↪ initializing plate


data_simulation = struct('dx',step_distance,'dy',step_distance,'dt',0.1,'
    ↪ Nt',0,'NX',plate_size,'NY',plate_size);
data_simulation.Nt = STEPS / data_simulation.dt;


%%
```

```
%BOUNDARY 1

%initializing blue edge
plate(1:plate_size,1,:) = data_plate.const_edge_temp;
plate(1:plate_size,plate_size,:) = data_plate.const_edge_temp;
plate(1,1:plate_size,:) = data_plate.const_edge_temp;
plate(plate_size,1:plate_size,:) = data_plate.const_edge_temp;


for m = 1:length(materials)

s = 2;
mean_diff = 1;
while(mean_diff > 0.0003)%0.0003 choosen arbitrarly
%for s = 1:STEPS %Small changes required for live simulation
    %initializing heater
    plate(heater_in_plate,heater_in_plate,s) = data_heater.const_temp;
    for i = 2:plate_size-1
        for j = 2:plate_size-1
            plate(i,j,s+1) = plate(i,j,s) + equastion_fraction(materials(m
                ↪ ),data_simulation) * (plate(i+1,j,s) + plate(i,j+1,s) -
                ↪ 4 * plate(i,j,s) + plate(i-1,j,s) + plate(i,j-1,s));
        end
    end
    mean_diff = mean(mean(abs(plate(:,:,s-1)-plate(:,:,s))));
    s = s + 1;
end

display_last_step(plate,data_plate,plate_size, data_simulation,
    ↪ materials_names{m},s);
end
%%
display_simulation(plate,data_plate,plate_size, data_simulation,STEPS);
%%
%BOUNDARY 2
for m = 1:1

s = 2;
mean_diff = 1;
while(mean_diff > 0.00001)%0.00001 choosen arbitrarly
%for s = 1:STEPS %Small changes required for live simulation
    %initializing heater
    if data_simulation.dt * s < 10
        deltaT_heater = deltaT_heater_equastion(materials(m),data_heater,
            ↪ data_plate.thickness,data_simulation);
```

```
        plate(heater_in_plate,heater_in_plate,s) = plate(heater_in_plate,
            ↪ heater_in_plate,s) + deltaT_heater;
    %uncomment if you want to check state of plate during heater shutdown
    %else
    % break
    end
    for i = 2:plate_size-1
        for j = 2:plate_size-1
            plate(i,j,s+1) = plate(i,j,s) + equastion_fraction(materials(m
                ↪ ),data_simulation) * (plate(i+1,j,s) + plate(i,j+1,s) -
                ↪ 4 * plate(i,j,s) + plate(i-1,j,s) + plate(i,j-1,s));
        end
        plate(1,i,s+1) = plate(2,i,s);
        plate(plate_size,i,s+1) = plate(plate_size-1,i,s);
        plate(i,1,s+1) = plate(i,2,s);
        plate(i,plate_size,s+1) = plate(i,plate_size-1,s);
    end
    plate(1,1,s+1) = plate(2,2,s);
    plate(1,plate_size,s+1) = plate(2,plate_size-1,s);
    plate(plate_size,1,s+1) = plate(plate_size-1,2,s);
    plate(plate_size,plate_size,s+1) = plate(plate_size-1,plate_size-1,s);

    mean_diff = mean(mean(abs(plate(:,:,s-1)-plate(:,:,s))));
    s = s + 1;
end
%display_simulation(plate,data_plate,plate_size, data_simulation,s);
display_last_step(plate,data_plate,plate_size, data_simulation,
    ↪ materials_names{m},s);
mean_tmp = mean(mean(plate(:,:,s)))
end
%% STADABILITY

data_simulation.dt = 0.4;
STEPS = 1000;
data_simulation.dx = 0.01;

for s = 1:STEPS %Small changes required for live simulation
    %initializing heater
    if data_simulation.dt * s < 10
        deltaT_heater = deltaT_heater_equastion(materials(2),data_heater,
            ↪ data_plate.thickness,data_simulation);
        plate(heater_in_plate,heater_in_plate,s) = plate(heater_in_plate,
            ↪ heater_in_plate,s) + deltaT_heater;
    %uncomment if you want to check state of plate during heater shutdown
    %else
```

```
    % break
    end
    for i = 2:plate_size-1
        for j = 2:plate_size-1
            plate(i,j,s+1) = plate(i,j,s) + equastion_fraction(materials
                ↪ (2),data_simulation) * (plate(i+1,j,s) + plate(i,j+1,s)
                ↪ - 4 * plate(i,j,s) + plate(i-1,j,s) + plate(i,j-1,s));
        end
        plate(1,i,s+1) = plate(2,i,s);
        plate(plate_size,i,s+1) = plate(plate_size-1,i,s);
        plate(i,1,s+1) = plate(i,2,s);
        plate(i,plate_size,s+1) = plate(i,plate_size-1,s);
    end
    plate(1,1,s+1) = plate(2,2,s);
    plate(1,plate_size,s+1) = plate(2,plate_size-1,s);
    plate(plate_size,1,s+1) = plate(plate_size-1,2,s);
    plate(plate_size,plate_size,s+1) = plate(plate_size-1,plate_size-1,s);
end
display_simulation(plate,data_plate,plate_size, data_simulation,STEPS);
%%

%NUMERICAL STADABILITY
%CHECK TIME NEEDED FOR STABILISATION
%find the criteria (border), relationship between KX,KY and resolution of
%the model and the dt. When the values are oscilating




% aim
% boundary 1 ,2
% stability of algorithm and time neede to stabilizate
% 3rd task , comparing with theorethical value and heat dissapation

% Compare result when temerature stabilised to constant value in every
% pixel.
% Delta Tt = Q / Cw * m = (P * Theat) / (Cw * V * ro) = (P * Theat) / (Cw
    ↪  * A^2 * h * ro)
% h - grubosc materialu, A^2 pole powierzchni plytki , ro - gestosc, Cw -
% cieplo wlasciwe materialu
% compare to
% Delta Tm z symulacji (jak bardzo wzrosla wrotsc boarda

%3rd task - simulate it in environment Q=Cexchange(Tij - To) for each
    ↪ node
```

```
%%

[XX YY] = meshgrid(0:dx:A,o:dy:A);

surf(XX,YY,T(:,:,n+1));
title(['Simulation time = ' num2str(n*dt) ' (s)']);
xlabel('x (m)');
ylabel('y (m)');
zlabel('Temperature (degC)');

function [ result ] = deltaT_heater_equastion( material, data_heater,
    ↪ plate_thickness, data_simulation)
    result = (data_heater.power * data_simulation.dt) / (material.heat *
        ↪ data_heater.edge * data_heater.edge * plate_thickness *
        ↪ material.density);
end

function [ result ] = equastion_fraction( material, data_simulation)
    result = (material.conductivity * data_simulation.dt) / (material.heat
        ↪  * material.density * data_simulation.dx * data_simulation.dx);
end

function display_last_step( plate, data_plate,plate_size, data_simulation
    ↪ ,materialName, STEPS)
[XX YY] = meshgrid(linspace(0,data_plate.edge,plate_size),linspace(0,
    ↪ data_plate.edge,plate_size));
min_temp_value = min(min(min(plate)));
max_temp_value = max(max(max(plate)));
z_axis = [min_temp_value max_temp_value]; %each min/max reduce Dim by 1.

%figure(1);
%pause(0.5);

%caxis(z_axis);%for color axis
%colorbar;

    figure();
    surf(XX,YY,plate(:,:,STEPS));
    shading interp;
    title(sprintf('Heat transfer of %s at time : %s[s]',materialName,
        ↪ num2str(STEPS*data_simulation.dt)));
    zlim(z_axis);

    xlabel('x coordinate');
    ylabel('y coordinate');
    zlabel('temperature');
```

```
    colorbar;
    caxis(z_axis);


end

function display_simulation( plate, data_plate,plate_size,
   ↪ data_simulation, STEPS)
[XX YY] = meshgrid(linspace(0,data_plate.edge,plate_size),linspace(0,
   ↪ data_plate.edge,plate_size));
min_temp_value = min(min(min(plate)));
max_temp_value = max(max(max(plate)));
z_axis = [min_temp_value max_temp_value]; %each min/max reduce Dim by 1.

%figure(1);
%pause(0.5);

%caxis(z_axis);%for color axis
%colorbar;

for i = 1:10:STEPS-1
    figure(1);
    surf(XX,YY,plate(:,:,i));
    shading interp;
    title(strcat('Simulation time = ', num2str(i*data_simulation.dt), ' (s
       ↪ )'));
    zlim(z_axis);

    xlabel('x coordinate');
    ylabel('y coordinate');
    zlabel('temperature');

    colorbar;
    caxis(z_axis);

    drawnow;
end

end
```