Contents


```
clc,clear,close all
```

FDM solver for Laplace's equation in 2D %%

Setting

```
disp('Heat Conduction Equation solver for 2D Li-ion Battery')
disp(' ')

% Resolution
res = 120;              % Multiples of 15 only
left1 = res/15;
left2 = res*7/15;
right1 = res*8/15;
right2 = res*14/15;

% Width of domain
W = 0.2;

% Height of domain
L = 0.2;

% Grid spacing
gs = W / (res - 1);
```

Battery and Thermal Characteristics

```
k = 28;                         % thermal conductivity (W/m.K)
cp = 1100;                      % specific heat (J/kg K)
rho = 2551.7;                   % density (kg/m^3)
alpha = k/(cp*rho);             % alpha (1/sec)

BTMS=input('Is the BTMS convection based (1) or conduction based (2) ? ');
disp(' ')

C_rate = input('C rate (C) = '); % Charge/discharge rate

tfinal = input('Time Final (s) = ');

if BTMS==1
    h = input('Convective Heat Transfer coefficient (W/Km2) (Range:0 to 10e5) h = ');
    Bi = h*gs/k;
    term = Bi;
    n = h;
    name = 'h';
elseif BTMS==2
    k_cool=input('Conductive Heat Transfer coefficient (W/mK) (Range:0 to 27) k = ');
    term = (k_cool/k);
    n = k_cool;
    name = 'k';
end

R             = 1.33e-3 ;               % Ohm Resistance of the battery
R_tabP        = 3.37e-5 ;               % Ohm Resistance of the positive tab
W_tabP        = 0.08 ;                  % Positive tab width
R_tabN        = 3.48e-5 ;               % Ohm Resistance of the negative tab
```

```
W_tabN        = 0.08 ;                  % Negative tab width
Bat_cap       = 53.0 ;                  % Battery capacity (Ah)
I             = Bat_cap * C_rate ;      % Current of the battery: 5C * 53Ah
```

Time parameters

```
time = 0;
dt = 0.9*(gs^2)/(4*alpha);

Fo = (alpha*dt)/(gs^2);            % Fourier Number
```

Setting boundary conditions

```
Twall = 25;
phi = Twall*ones(res);
```

Sensor 1

```
X_sensor1 = 60;
Y_sensor1 = 60;
Tt1 = [time phi(Y_sensor1, X_sensor1)] ;   % Tt=[0 25]
```

Sensor 2

```
X_sensor2 = 10;
Y_sensor2 = 15;
Tt2 = [time phi(Y_sensor2, X_sensor2)] ;   % Tt=[0 25]
```

Heat Flux Calculation

```
qtab_neg = ((I^2)*R_tabN/(W_tabN^2));
qtab_pos = ((I^2)*R_tabP/(W_tabP^2));

% Creating a copy of phi
phi_old = phi;

tic
while time < tfinal
    for i=1:res
        for j=1:res
            % Bottom Left Corner
            if (i == 1 && j == 1)
                phi(i,j) = 0.5*(phi_old(i,j+1)+phi_old(i+1,j));

                % Bottom Right Corner
            elseif (i == res && j == 1)
                phi(i,j) = 0.5*(phi_old(i,j+1)+phi_old(i-1,j));

                % Top Left Corner
            elseif (i == 1 && j == res)

                phi(i,j) = 0.5*(phi_old(i,j-1)+phi_old(i+1,j));

                % Top Right Corner
            elseif (i == res && j == res)

                phi(i,j) = 0.5*(phi_old(i,j-1)+phi_old(i-1,j));

                % Bottom Edge
            elseif (j == 1)

                phi(i,j) = Fo*(phi_old(i-1,j) +2*phi_old(i,j+1) +phi_old(i+1,j) +2*term*Twall +((1/Fo) -4 -(2*term))*phi_old(i,j));

                % Right Edge
            elseif (j ~= res && i == res)

                phi(i,j) = Fo*(phi_old(i,j+1) +2*phi_old(i-1,j) +phi_old(i,j-1) +2*term*Twall +((1/Fo) -4 -(2*term))*phi_old(i,j));

                % Left Edge
```

```
            elseif (j ~= res && i == 1)

                phi(i,j) = Fo*(phi_old(i,j+1) +2*phi_old(i+1,j) +phi_old(i,j-1) +2*term*Twall +((1/Fo) -4 -(2*term))*phi_old(i,j));

            end
        end
    end

    % Top Edge Gaps
    for i = [2:left1-1  left2+1:right1-1  right2+1:res-1]
        for j = res
            phi(i,j) = Fo*(phi_old(i-1,j) +2*phi_old(i,j-1) +phi_old(i+1,j) +2*term*Twall +(1/Fo -4 -2*term)*phi_old(i,j));
        end
    end

    % Negative Tab
    for i = left1:left2
        for j = res
            phi(i,j) = Fo*(phi_old(i-1,j) +2*phi_old(i,j-1) +phi_old(i+1,j) +2*gs*qtab_neg/387 +(1/Fo -4)*phi_old(i,j));
        end
    end

    % Positive Tab
    for i = right1:right2
        for j = res
            phi(i,j) = Fo*(phi_old(i-1,j) +2*phi_old(i,j-1) +phi_old(i+1,j) +2*gs*qtab_pos/387 +(1/Fo -4)*phi_old(i,j));
        end
    end

    % Internal Domain
    for i = 2:res-1
        for j = 2:res-1
            qgen = ((I^2)*R)/(W*L*0.011);
            phi(i,j) = phi_old(i,j)+Fo*(phi_old(i,j+1)-4*phi_old(i,j) +phi_old(i,j-1) +phi_old(i+1,j) +phi_old(i-1,j)) +(qgen*dt)/(rho*cp);
        end
    end

    phi_old = phi;
    time = time + dt;

    Tt1 = [Tt1 ; time phi(Y_sensor1, X_sensor1)] ; %% Tt1= [0 25 ; 0.1 25.1]
    Tt2 = [Tt2 ; time phi(Y_sensor2, X_sensor2)] ; %% Tt2= [0 25 ; 0.1 25.1]

end
toc
phi=transpose(phi);
```

Plot results

```
load colormap.mat

% Create meshgrid of plotting points
[xplot, yplot] = meshgrid(linspace(0, W, res), linspace(0, L, res));

% Find Temp Gradient
[ux, uy] = gradient(phi);
ux = -ux;
uy = -uy;

mag = sqrt(ux.^2 + uy.^2);
uxn = ux ./ mag;
uyn = uy ./ mag;

disp(' ')
disp('Value of Maximum Temperature (C)')
disp(max(max(phi)))
disp('Nodal Position of Maximum Temperature')
[X, Y] = find(ismember(phi, max(phi(:))));
fprintf('i = %d j = %d',Y,X)
disp(' ')
disp('Largest Temperature Difference (C)')
disp(max(max(phi))-min(min(phi)))

figure
quiver(xplot, yplot, uxn, uyn);
```

```
title('Normalised Heat Flux Field','interpreter','latex','Fontsize',14)
xlabel('x','interpreter','latex','Fontsize',14)
ylabel('y','interpreter','latex','Fontsize',14)
axis equal
axis tight

% Sensor 1
figure
plot(Tt1(1:end,1),Tt1(1:end,2));
title_text=sprintf('Temperature variation vs time at point (%d,%d) ',X_sensor1,Y_sensor1);
title(title_text,'interpreter','latex','Fontsize',14);
xlabel('Time (s)','interpreter','latex','Fontsize',14)
ylabel('Temperature (C)','interpreter','latex','Fontsize',14)

% Sensor 2
figure
plot(Tt2(1:end,1),Tt2(1:end,2));
title_text=sprintf('Temperature variation vs time at point (%d,%d) ',X_sensor2,Y_sensor2);
title(title_text,'interpreter','latex','Fontsize',14);
xlabel('Time (s)','interpreter','latex','Fontsize',14)
ylabel('Temperature (C)','interpreter','latex','Fontsize',14)

figure
contourf(xplot, yplot, phi,10);
colorbar
colormap(map)
caxis([26.8 64.2])
title_text=sprintf('%dC charging for %ds, %s = %d ',C_rate,tfinal,name,n);
title(title_text,'interpreter','latex','Fontsize',14);
xlabel('x','interpreter','latex','Fontsize',14)
ylabel('y','interpreter','latex','Fontsize',14)
hcb=colorbar;
title(hcb,'Temperature','interpreter','latex','Fontsize',10)
view(2)
axis equal

file_name = sprintf('CR%d_tf%d ',C_rate,tfinal);
saveas(gcf,file_name,'pdf');
```

Published with MATLAB® R2018a