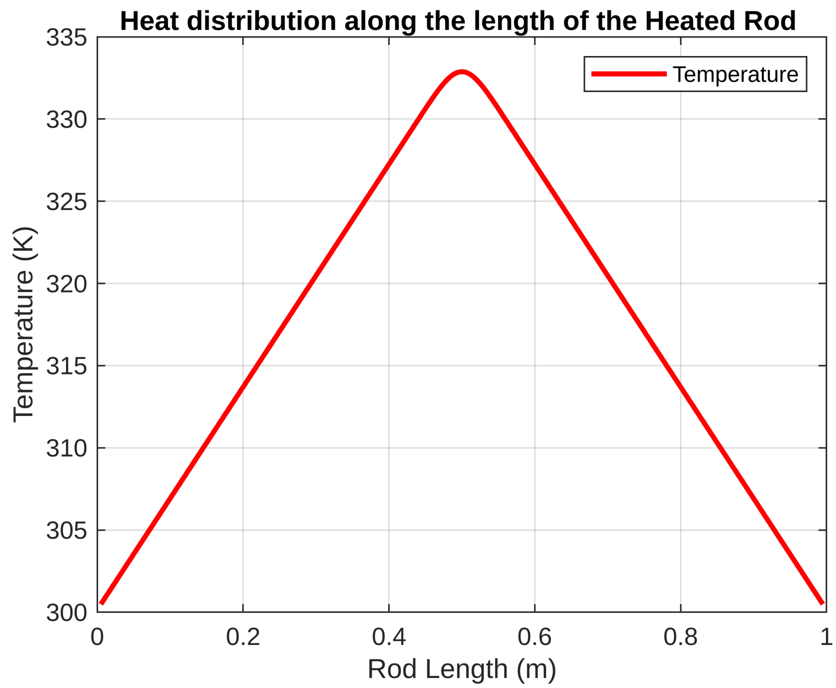
Homework 4

Question 1 (Heat transfer, 30pts)

```
L = 1; % (m)
N = 200; %sections
h = L/N;
S = 2.7e4; %(W/m3)
epsilon = 3e-2;
beta = 2.1e-3;%(m2/s)
heat_capacity = 208.5; %(J/(kg(K)))
density = 12.1; %(kg/m³)

% flame model
h = L/N;
x = (h:h:(L-h))';
f_x = S*exp(-((x-(L/2))/epsilon).^2);

% sparse diagonal matrix
A = spdiags(-ones(N-1,1)*[1 -2 1], -1:1, N-1,N-1);
full(A);

k = beta*heat_capacity*density;
b = (h^2/(2*k))*f_x; % thermal conductivity along the length of the rod

% boundary conditions
b(1,1) = b(1)+300.15;
b(N-1) = b(N-1)+300.15;

Temp = A\b;
% Plot
plot(x, Temp, 'red', LineWidth=2);
xlabel('Rod Length (m)');
ylabel('Temperature (K)');
title('Heat distribution along the length of the Heated Rod ');
legend('Temperature');
% isplay
grid on;
hold off;
```




```
% temperature when skin starts to burn
a = find(Temp > 325.15,1);
 center_rod = L/2;
% distance from the center rod
distance = abs(x*a-center_rod);
format short
disp(distance(1))
```

```
    0.1300
```

Question 2 (Polynomial least squares fitting, 15pts)

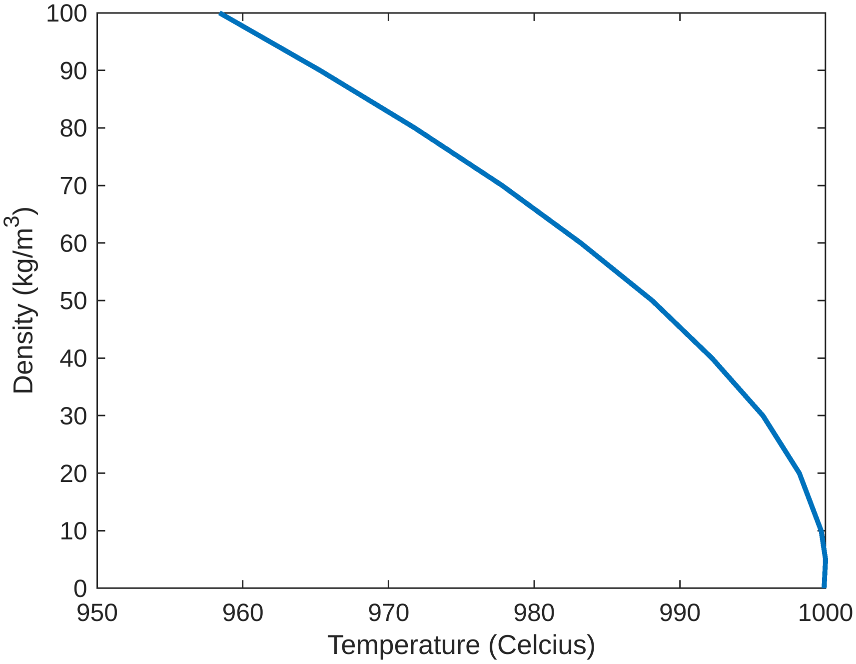
```
d = load('H2Odensity.dat');
density = d(:,1);
temp = d(:,2); % temperature is the independant variable, density is the
dependant variable.

n = 0:4;
for i = n
    % polynomial
    pn = polyfit(temp, density, i);
end
```

```
Warning: Polynomial is badly conditioned. Add points with distinct X values, reduce the degree of
the polynomial, or try centering and scaling as described in HELP POLYFIT.
Warning: Polynomial is badly conditioned. Add points with distinct X values, reduce the degree of
the polynomial, or try centering and scaling as described in HELP POLYFIT.
```

```
% compute error
err = norm(density - polyval(pn, temp),inf)
```

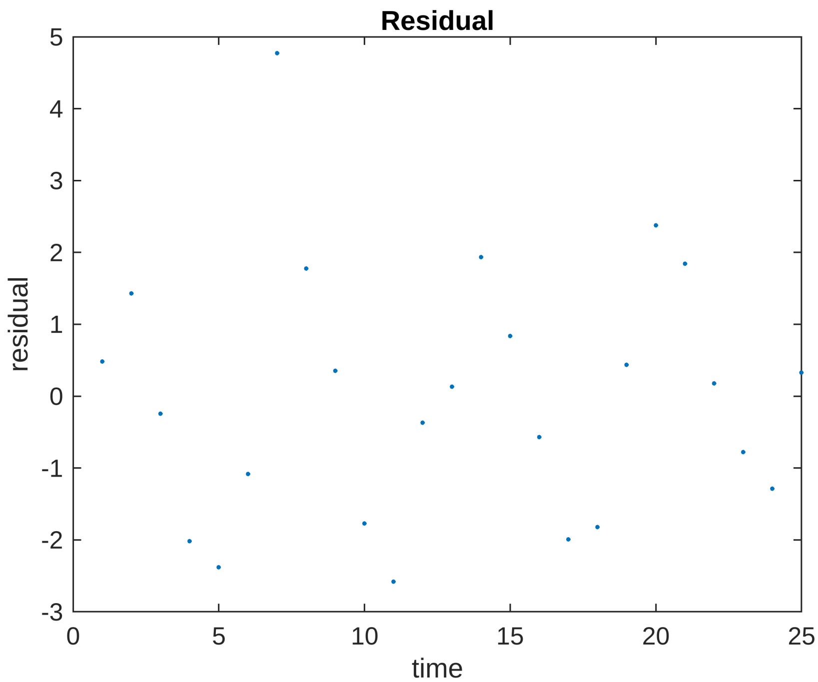
```
err = 5.7310
```

```
plot(temp, density, LineWidth=2)
xlabel('Temperature (Celcius)')
ylabel('Density (kg/m^3)')
```

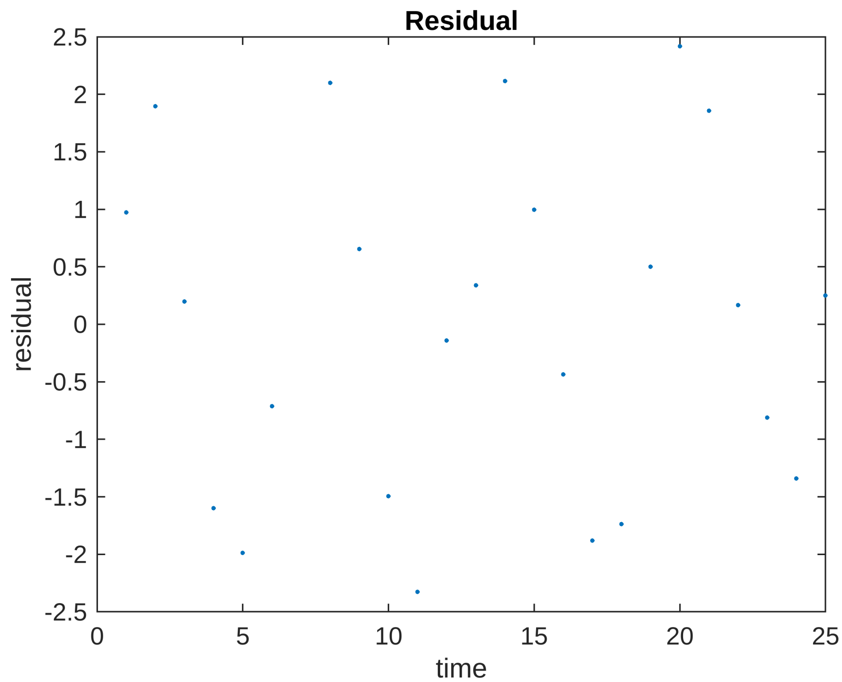


Question 3 (Oscillatory data fitting, 20pts)

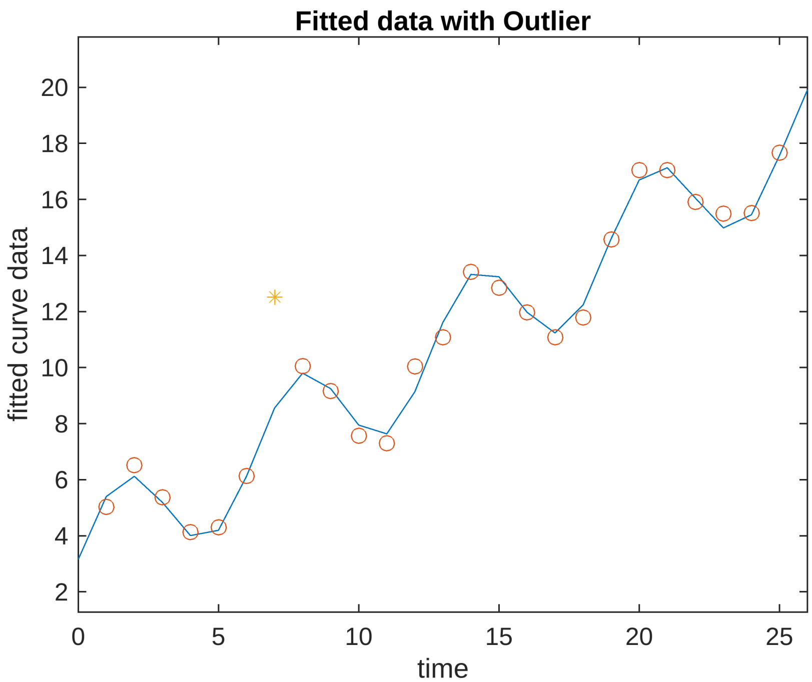
```
t = 1:25;
y = [ 5.0291   6.5099   5.3666   4.1272   4.2948
6.1261  12.5140  10.0502   9.1614   7.5677
7.2920  10.0357  11.0708  13.4045  12.8415
11.9666  11.0765  11.7774  14.5701  17.0440
17.0398  15.9069  15.4850  15.5112  17.6572];

y = y';
y = y(:);

%(a)
l = [ones(25,1) t'];
x2 = l\y;
% residual
r = y-l*x2;

% Plot residual
plot(t,r,'.')
xlabel('time')
ylabel('residual')
title('Residual')
```




```
%(b)

% find the outlier
outlier = find(r == max(r));
% save outliers
outlier_t = t(7);
outlier_y = y(7);
% removes value
t(7) = []; % empty
y(7) = [];

l = [ones(24,1) t'];
x2 = l\y;

% residual
r = y-l*x2;

% Plot residual w/o outlier
clc
plot(t,r,'.')
xlabel('time')
ylabel('residual')
title('Residual')
```



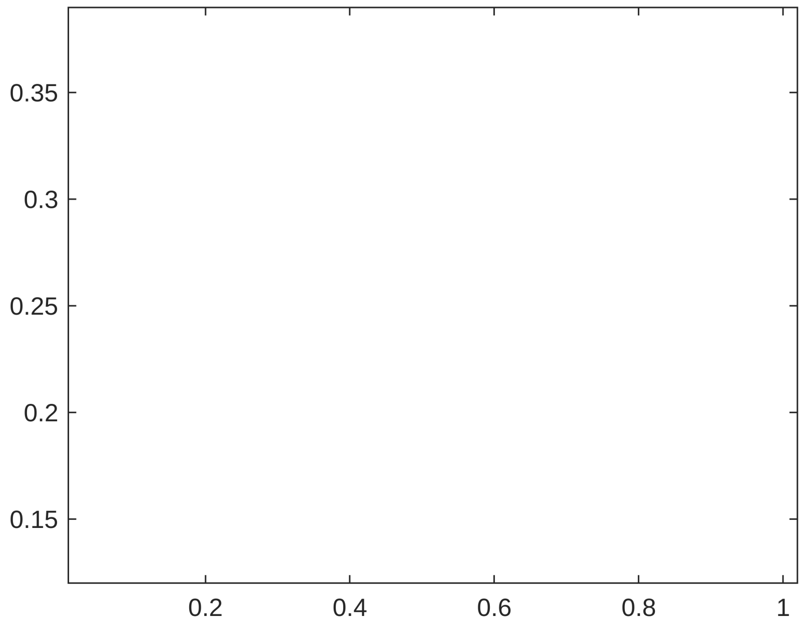

```
% (c)
length(t);
l1 = [ones(24,1) t',sin(t)'];
% backslash
x2 = l1\y;

% residual
r = y-l1*x2;

%(d)
t1 = (0:26);
l2 = [ones(27,1) t1',sin(t1)'];

%plot
plot(t1,l2*x2, '-')
hold on;
plot(t, y, 'o')
hold on;
plot(outlier_t, outlier_y,'*')
xlabel('time')
ylabel('fitted curve data')
title('Fitted data with Outlier')
axis equal;
hold off;
```



Question 4 (Fitting planetary orbits, 20pts)

```
x = [1.02 .95 .87 .77 .67 .56 .44 .30 .16 .01];
y = [0.39 .32 .27 .22 .18 .15 .13 .12 .13 .15];

% part (a)
[X,Y] = meshgrid(x,y);

% calculate [X,Y]
xmin=min(x);
xmax=max(x);
ymin=min(y);
ymax=max(y);
deltax = norm(xmax-xmin);
deltay = norm(ymax-ymin);

% Copy x y with meshgrid
[X,Y] = meshgrid(xmin:deltax:xmax, ymin:deltay:ymax);

b1 = ones(1,length(x));
Z = x.^2 + x.*y + y.^2 + x + y + 1;

C = Z\b1;

% apply C to coeficients
```

```
a=C(1); b=C(2); c=C(3); d=C(4); e=C(5); f=(1);

Z = a.*X.^2 + b.*X.*Y + c.*Y.^2 + d.*X + e.*Y + f;
X;Y;

contour(X,Y,Z,[0 0])
```




```
% (b)
```

Question 5 (Least squares and rotation, 10pts)

```
load('mathrules.mat');

% Center the data
xcbar = mean(xc); %average xc values
ycbar = mean(yc);
xc = xc - xcbar;% original - average value
yc = yc - ycbar;

x1 = -xc\yc;
% calculate rotation angle
rotation = atan(x1);
% rotation matrix R
R = [cos(rotation) -sin(rotation);
 sin(rotation) cos(rotation)];

% apply rotation
 rotation_matrix = transpose(R*[xc, yc]');
```

```
rxc = rotation_matrix(:,1); % column 1
ryc = rotation_matrix(:,2);% column 2

% Plot
 figure;
plot(rxc, ryc, 'r.');
title('Aligned Signature ');
axis equal;
```