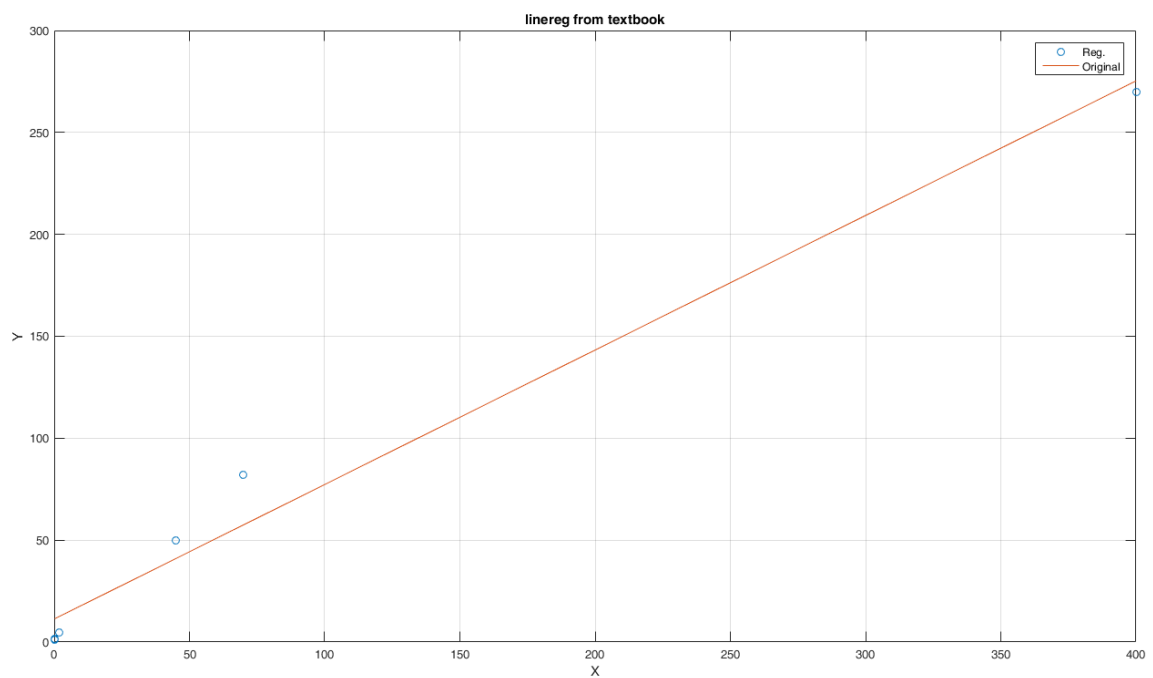


Problem 14.11

Use linear regression to determine an equation to predict metabolism rate as a function of mass

One can use `linreg` from our textbook



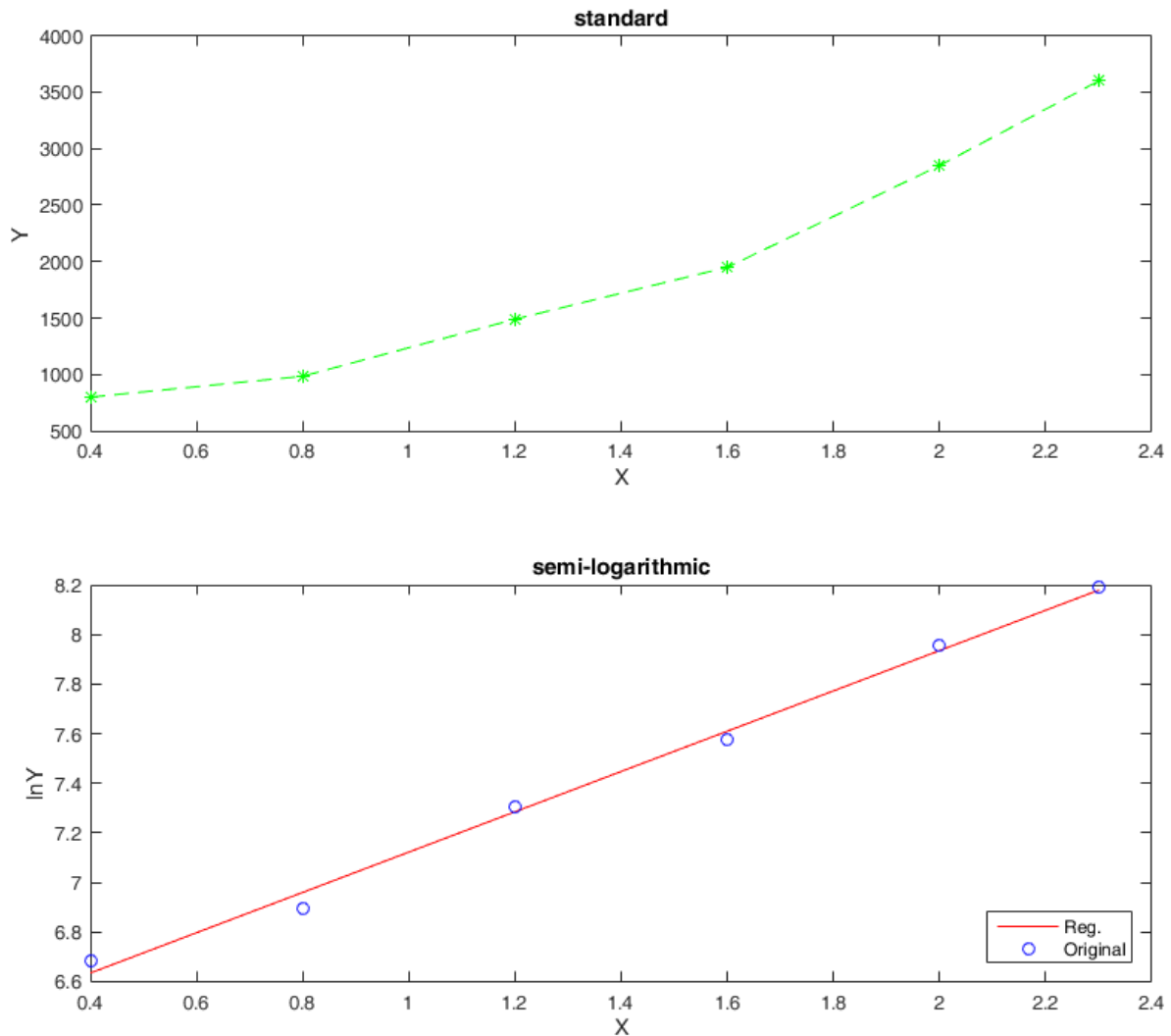
$$y(x) = 11.2908 + 0.6599x$$

When $x=200\text{kg}$, $y(x=200) = 143.27$

Problem 14.13

Exponential model: $y = \alpha_1 e^{\beta_1 x}$ → Linearization → Slope: β_1 , Intercept: $\ln \alpha_1$

Using linear regression with x and $\log(y)$



Finding fitting parameters: $\alpha_1 = 549.8$ and $\beta_1 = 0.8127$

Problem 15.3

Using matrix operations to fit a cubic polynomial to the given data

One can find a detail of this in Chapter 15.3 and Example 15.3 in the textbook

```
clear;clc;

format short

X = [3 4 5 7 8 9 11 12]';
Y = [1.6 3.6 4.4 3.4 2.2 2.8 3.8 4.6]';

%find r^2 and s_y/x
Z = [ones(size(X)) X X.^2 X.^3]; → cubic polynomial

a = (Z'*Z)\(Z'*Y); → gives coefficients
Sr = sum((Y-Z*a).^2);
r2 = 1-Sr/sum((Y-mean(Y)).^2);
syx = sqrt(Sr/(length(X)-length(a)));
```

$$y = 0.0467x^3 - 1.0412x^2 + 7.1438x - 11.4887$$

$$r^2 = 0.8290 \text{ and } s_{y/x} = 0.57$$

Problem 15.6

Using generalized linear regression with matrix operation to derive predictive equation

```
clear;clc;

X1 = [0 5 10 15 20 25 30 0 5 10 15 20 25 30 0 5 10 15 20 25 30]';
X2 = [0 0 0 0 0 0 0 10 10 10 10 10 10 10 20 20 20 20 20 20 20]';

Y = [14.6 12.8 11.3 10.1 9.09 8.26 7.56 12.9 11.3 10.1 9.03 8.17 7.46
6.85 11.4 10.3 8.96 8.08 7.35 6.73 6.20]';

%find r^2 and s_y/x
Z = [ones(size(X1)) X1 X2]; → multiple regression (X1 is T, X2 is c)

a = (Z'*Z)\(Z'*Y); → gives coefficients

display(a)

f_estimate = a(1) + a(2)*(12) + a(3)*(15); → T = 12°C and c = 15 g/L
error = abs((9.09-f_estimate)/9.09)*100;
display(f_estimate)
display(error)
```

$$y = -0.1049x_1 - 0.2012x_2 + 13.5221$$

$$y(x_1 = 12, x_2 = 15) = 9.5334 \text{ and error is } 4.88\%$$

Problem 17.12

Using `polyfit` and `polyval` function to fit the data with 5th-order polynomial

$$V(i) = 592i^3 + 45i$$

$$V(i = 0.1) = 592(0.1)^3 + 45(0.1) = 5.092$$

Problem 17.19

One can use `polyfit` and `polyval` function to solve this problem

However, it has ill-conditioning problem so we should do scaling → Changing length scale

```
clear;clc;

format short
y = [120 90 60 30 0]; %scaling from m to km to avoid condition problem
g = [9.5682 9.6278 9.6879 9.7487 9.8100 ];

p = polyfit(y,g,2);
display(p)

d = polyval(p,55);
display(d)
```

$$g(y) = -0.021y + 9.81$$

$$g(y = 55km) = -0.021(55) + 9.81 = 9.698$$

Above 2nd order, a polynomial fits give a same result

Problem 18.6

Develop m-file to do spline interpolation with natural end condition

```
function [yy,dy,d2] = natspline(x,y,xx)
% natspline: natural spline with differentiation
% [yy,dy,d2] = natspline(x,y,xx): uses a natural cubic spline
% interpolation to find yy, the values of the underlying function
% y at the points in the vector xx. The vector x specifies the
% points at which the data y is given.
% input:
% x = vector of independent variables
% y = vector of dependent variables
% xx = vector of desired values of dependent variables
% output:
% yy = interpolated values at xx
% dy = first derivatives at xx
% d2 = second derivatives at xx
n = length(x);
if length(y)~=n, error('x and y must be same length'); end
if any(diff(x)<=0),error('x not strictly ascending'),end
m = length(xx);
b = zeros(n,n);
aa(1,1) = 1; aa(n,n) = 1; %set up Eq. 18.27
bb(1)=0; bb(n)=0;
for i = 2:n-1
    aa(i,i-1) = h(x, i - 1);
    aa(i,i) = 2 * (h(x, i - 1) + h(x, i));
    aa(i,i+1) = h(x, i);
    bb(i) = 3 * (fd(i + 1, i, x, y) - fd(i, i - 1, x, y));
end
c=aa\bb'; %solve for c coefficients
for i = 1:n - 1 %solve for a, b and d coefficients
    a(i) = y(i);
    b(i) = fd(i + 1, i, x, y) - h(x, i) / 3 * (2 * c(i) + c(i + 1));
    d(i) = (c(i + 1) - c(i)) / 3 / h(x, i);
end
for i = 1:m %perform interpolations at desired values
    [yy(i),dy(i),d2(i)] = SplineInterp(x, n, a, b, c, d, xx(i));
end
end

function hh = h(x, i)
    hh = x(i + 1) - x(i);
end

function fdd = fd(i, j, x, y)
    fdd = (y(i) - y(j)) / (x(i) - x(j));
end

function [yyy,dy,d2y]=SplineInterp(x, n, a, b, c, d, xi)
    for ii = 1:n - 1
        if xi >= x(ii) - 0.000001 & xi <= x(ii + 1) + 0.000001
            yyy=a(ii)+b(ii)*(xi-x(ii))+c(ii)*(xi-x(ii))^2+d(ii)...
                *(xi-x(ii))^3;
            dyy=b(ii)+2*c(ii)*(xi-x(ii))+3*d(ii)*(xi-x(ii))^2;
            d2y=2*c(ii)+6*d(ii)*(xi-x(ii));
            break
        end
    end
end
end
```

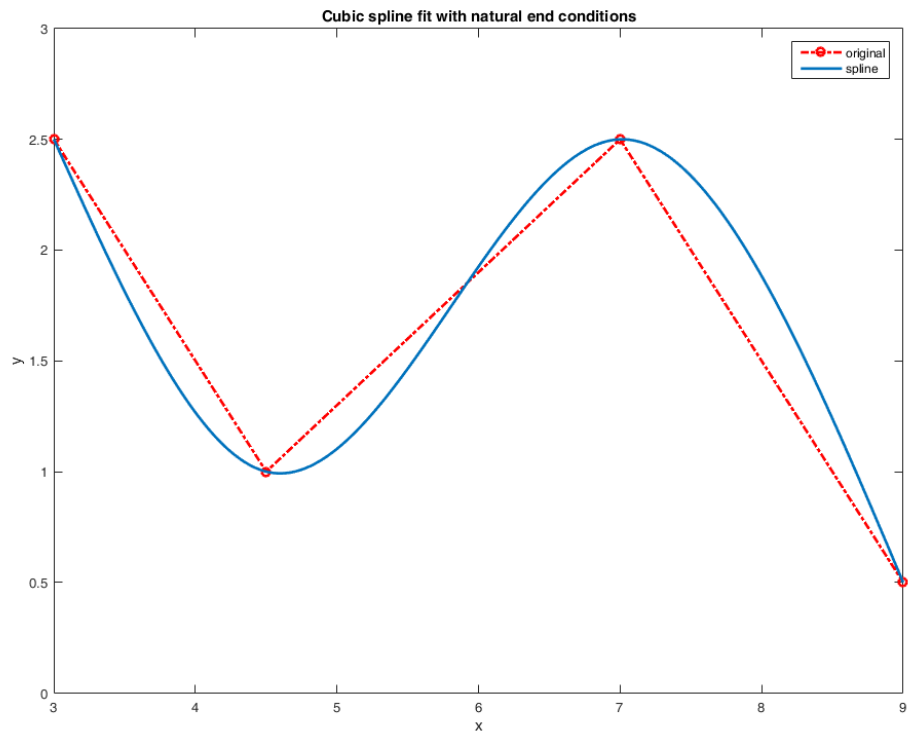
```

clc; clear; clf;

% Data is from table 18.1
x = [3.0 4.5 7.0 9.0];
y = [2.5 1.0 2.5 0.5];

xx = linspace(3,9,601);
yy = natspline(x,y,xx);
plot(x,y,'r-.o',xx,yy,'LineWidth',2);
title('Cubic spline fit with natural end conditions');
xlabel('x');
ylabel('y');

```



Problem 18.9

Using `interp1` and `polyfit` function to fit the data and estimate $o(27)$

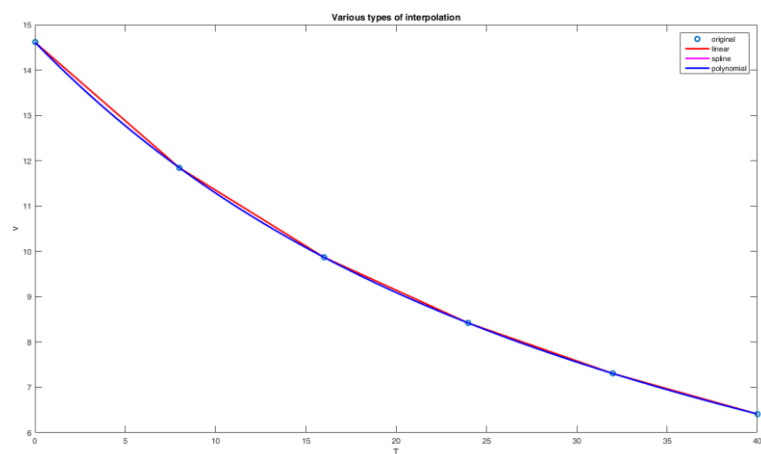
```
clear; clc; clf;

format short
t = [0 8 16 24 32 40];
v = [14.612 11.843 9.870 8.418 7.305 6.413];
tt = linspace(0,40,4001);

v1 = interp1(t,v,tt,'linear');
vs = interp1(t,v,tt,'spline');

p5 = polyfit(t,v,5);
d5 = polyval(p5,27);
display(p5)
display(d5, 'Fifth-order polynomial')
display(v1(2701), 'Linear interpolation')
display(vs(2701), 'Spline')
f5 = p5(6) + p5(5)*tt + p5(4)*tt.^2 + p5(3)*tt.^3 + p5(2)*tt.^4 + p5(1)*tt.^5;

plot(t,v,'o',tt,v1,'r',tt,vs,'m',tt,f5,'b','LineWidth',2)
title('Various types of interpolation');
xlabel('T')
ylabel('v')
legend('original','linear','spline','polynomial')
```



Estimation of $o(27)$

Fifth-order polynomial $\rightarrow 7.9683$

Linear interpolation $\rightarrow 8.0006$

Spline $\rightarrow 7.9679$