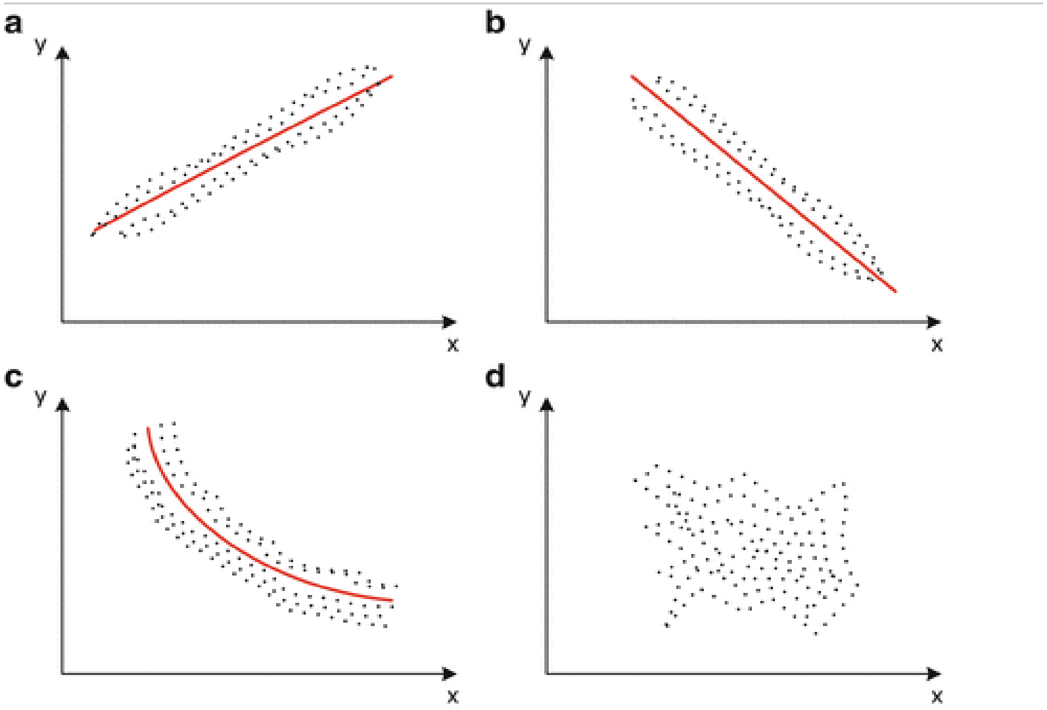
Machine Learning
Practical-1
Non-linear regression (NLR)

Lecturer:

Dr Hamid Khayyam (Australia)
Email: hamid.khayyam@rmit.edu.au

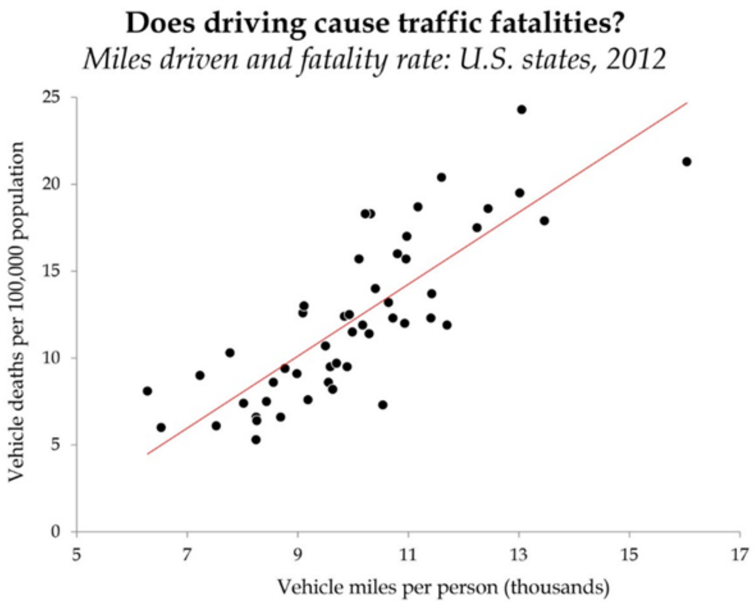
Regression analysis

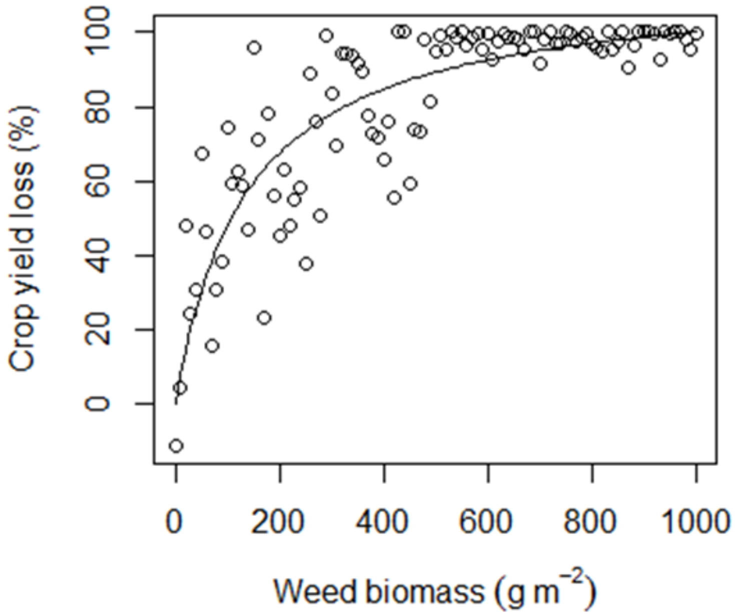
Regression models describe the correlation between input and output variables.



Different types of correlation between *Y* and *X*. (**a**) Positive linear correlation (**b**) negative linear correlation (**c**) nonlinear correlation (**d**) no correlation

Example of linear and non-linear regression model





Linear Regression Model

Nonlinear Regression model

Linear regression model:

Linear regression is used to model a linear relationship between a continuous dependent variables *Y(output)* and independent variables *X(input)*.

- **fitlm:** Creates linear regression model.

Syntax:

```
mdl = fitlm(x,y);

mdl = fitlm(x,y,modelspec);
```

Description:

x: Predictor variables(Input)

y: Response variable(output)

modelspec: Model specification

- **Predict:** Predict response of linear regression model

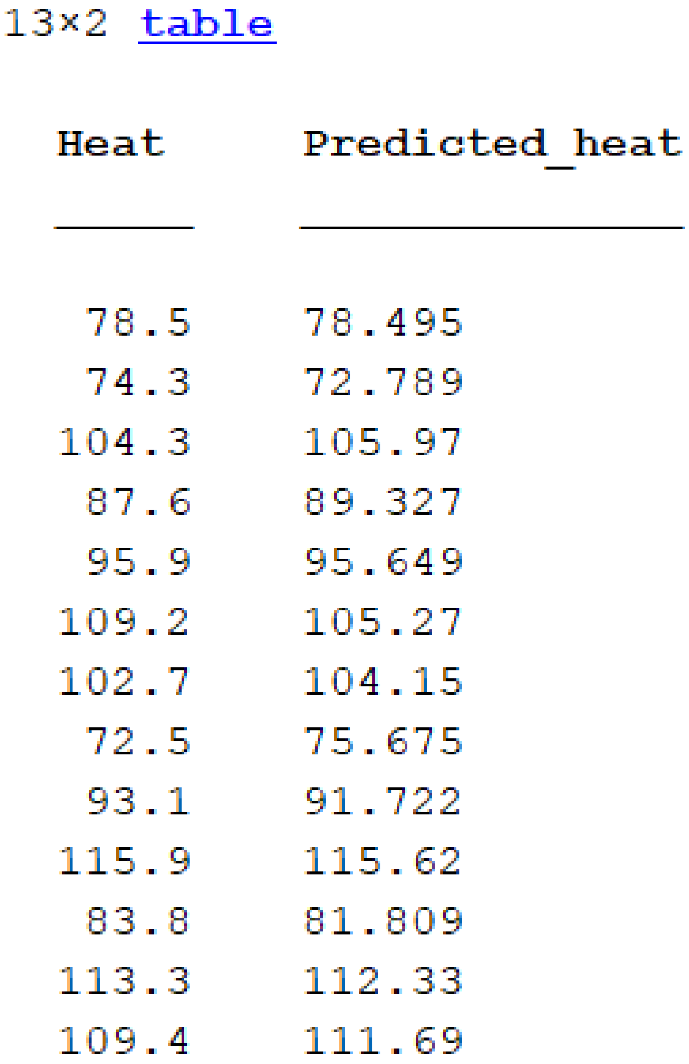
Syntax:

```
ypred = predict(mdl,Xnew);
```

Example of Linear regression function: fitlm

```
clear;clc;
rng(1);
load hald % built-in data set in Matlab
mdl = fitlm(ingredients,heat,'linear') ;
 heatpred = predict(mdl,ingredients);
table( heat( : ),heatpred( : ), 'VariableNames',...
    {' Heat',' Predicted_heat'}) %Show the results of output data and predicted output.
```

Example of Linear regression function: fitlm (cont.)

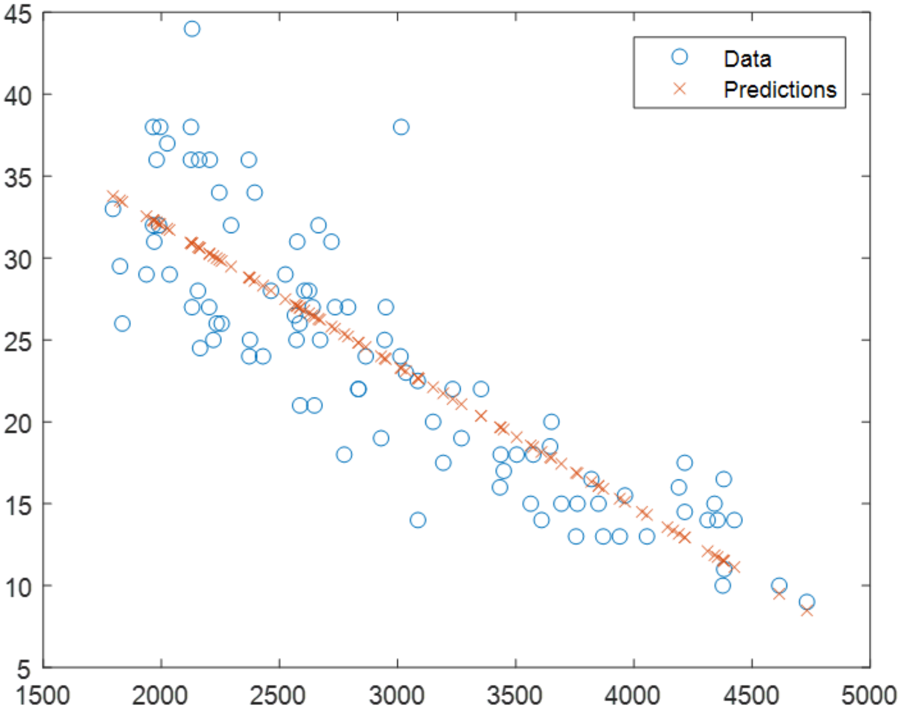


```
13×2 table

 Heat      Predicted_heat
 _____     ______________

  78.5     78.495
  74.3     72.789
 104.3     105.97
  87.6     89.327
  95.9     95.649
 109.2     105.27
 102.7     104.15
  72.5     75.675
  93.1     91.722
 115.9     115.62
  83.8     81.809
 113.3     112.33
 109.4     111.69
```

Example of Linear regression function (2): fitlm (cont.)

```
clear;clc;

rng(1);

Filename='regression1.xlsx';

Sheetread='Sheet1';

Input1='A1:A100';

output1='B1:B100';

Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft Excel

Target=xlsread(Filename,Sheetread,output1 );

x=Input;

t=Target;

t= fillmissing(t,'spline'); %fill in the missing output data
```

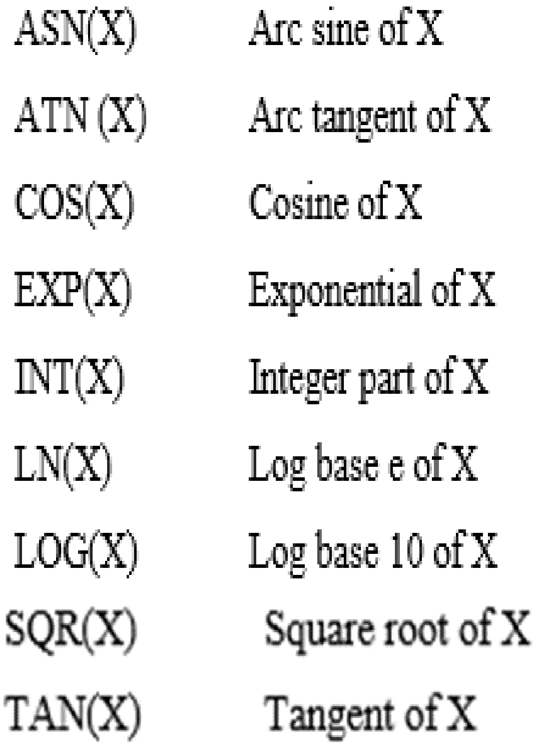
```
mdl = fitlm(X,y,'linear');
ypred = predict(mdl,X);
plot(X,y,'o',X,ypred,'x')
legend('Data','Predictions')
```



Nonlinear regression models/functions

Nonlinear regression model is a nonlinear correlation of a continuous dependent variables *Y(output)* and independent variables *X(input)*.

- **Examples of Nonlinear functions**



ASN(X)	Arc sine of X
ATN (X)	Arc tangent of X
COS(X)	Cosine of X
EXP(X)	Exponential of X
INT(X)	Integer part of X
LN(X)	Log base e of X
LOG(X)	Log base 10 of X
SQR(X)	Square root of X
TAN(X)	Tangent of X

Examples of nonlinear models in Matlab

1. Y=b(1) + b(2)*x(:,1).^b(3) +b(4)*x(:,2).^b(5)

2. Y=x(:,1).*exp(x(:,2))

3. Y = (b(1)*x2 – x(:,3)/b(5))./(1+b2*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3));

4. Y=b(1)+b(2) *x(:,1)+b(3) *x(:,2)+b(4) *x(:,3)

5. Y=b(1)+b(2) *x(:,1)+b(3) *x(:,2)+b(4) *x(:,3)+b(5) *(x(:,1).^2)+b(6) *x(:,3) +b(7) *((x(:,1).*x(:,2).*x(:,3)));

Matlab function for non-linear polynomial models: polyfit

- **polyfit:** A Matlab solver for non-linear polynomial models.

Syntax:

```
p = polyfit(x,y,n)
```

Describe:

Returns the coefficients for a polynomial p(x) of degree n that is a best fit for the data in y. x is the input ,y is the output.

- **polyval**

Syntax:

```
ypredicted = polyval(p,x)
```

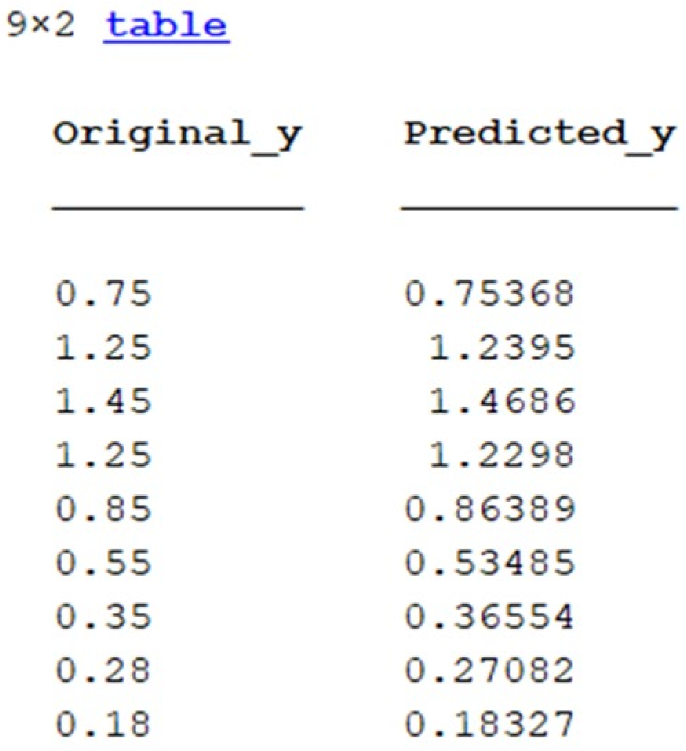
Describe:

To predict y based on x.

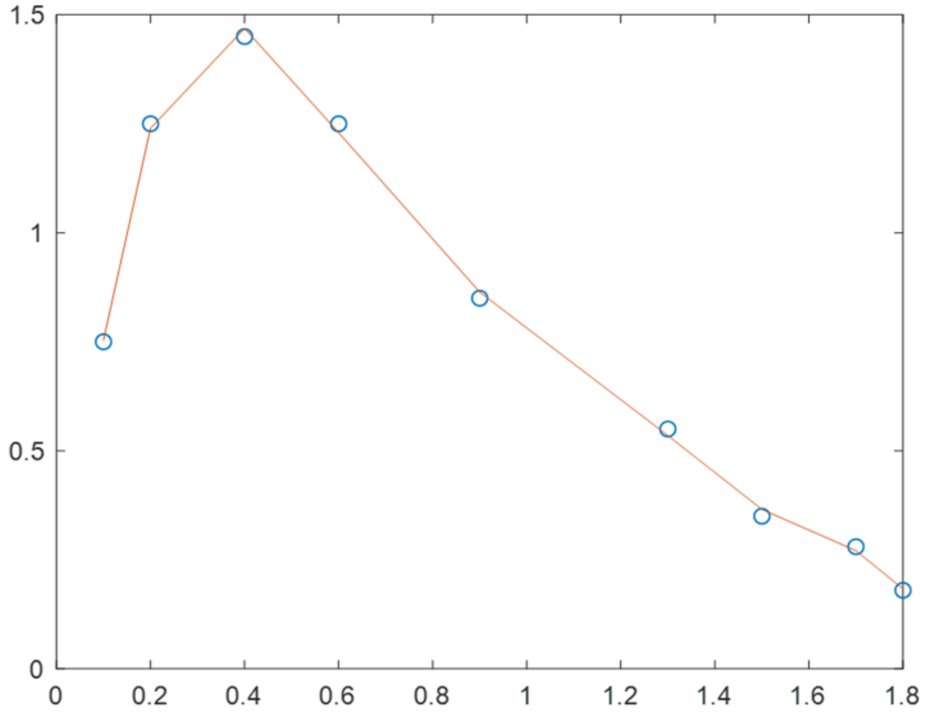
Example of ‘polyfit’ Matlab function 1:

```
clear ;clc;
x=[0.1 0.2 0.4 0.6 0.9 1.3 1.5 1.7 1.8]
y=[0.75 1.25 1.45 1.25 0.85 0.55 0.35 0.28 0.18]
p1=polyfit(x,y,7);% returns the coefficients for a polynomial p(x) of degree 7
y2=polyval(p1,x);
plot(x,y,'o',x,y2,'-');
table( y( : ),y2( : ), 'VariableNames',...
    {' Original_y',' Predicted_y'}) %Show the results of data of the output and predicted output.
```

Example of ‘polyfit’ Matlab function (cont.) 2:



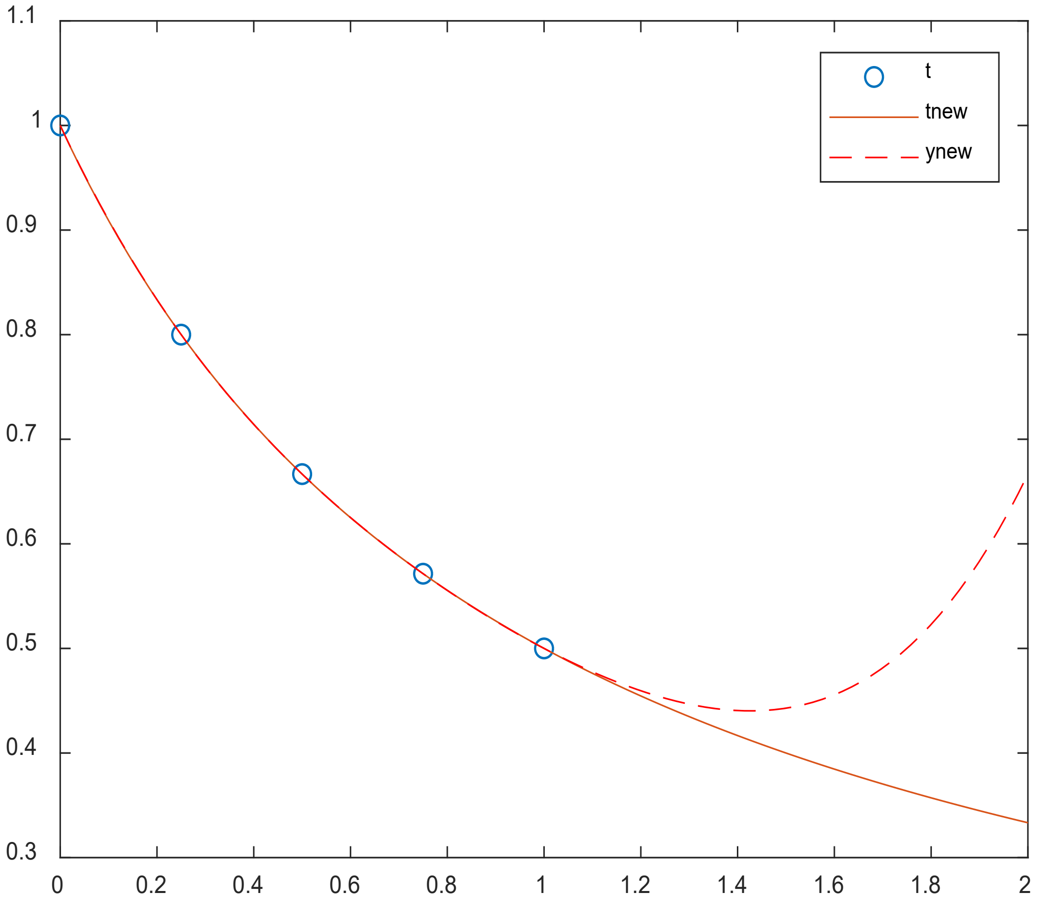
```
9×2 table

  Original_y    Predicted_y
  __________    ___________

  0.75          0.75368
  1.25           1.2395
  1.45           1.4686
  1.25           1.2298
  0.85          0.86389
  0.55          0.53485
  0.35          0.36554
  0.28          0.27082
  0.18          0.18327
```



Example of ‘polyfit’ Matlab function 2 (cont.):

```
clear;clc;
x = linspace(0,1,5); % input
t = 1./(1+x);% output
p = polyfit(x,t,4); %gives the coefficient
xnew = linspace(0,2); % new x (generate linearly)
tnew = 1./(1+xnew); % new y
ynew = polyval(p,xnew); % polynomial evaluation
figure
plot(x,t,'o')
hold on
plot(xnew,tnew)
plot(xnew,ynew,'r--')
legend('t','tnew','ynew')
```

Example of ‘polyfit’ Matlab function 2 (cont.):



Example of ‘polyfit’ Matlab function 3:

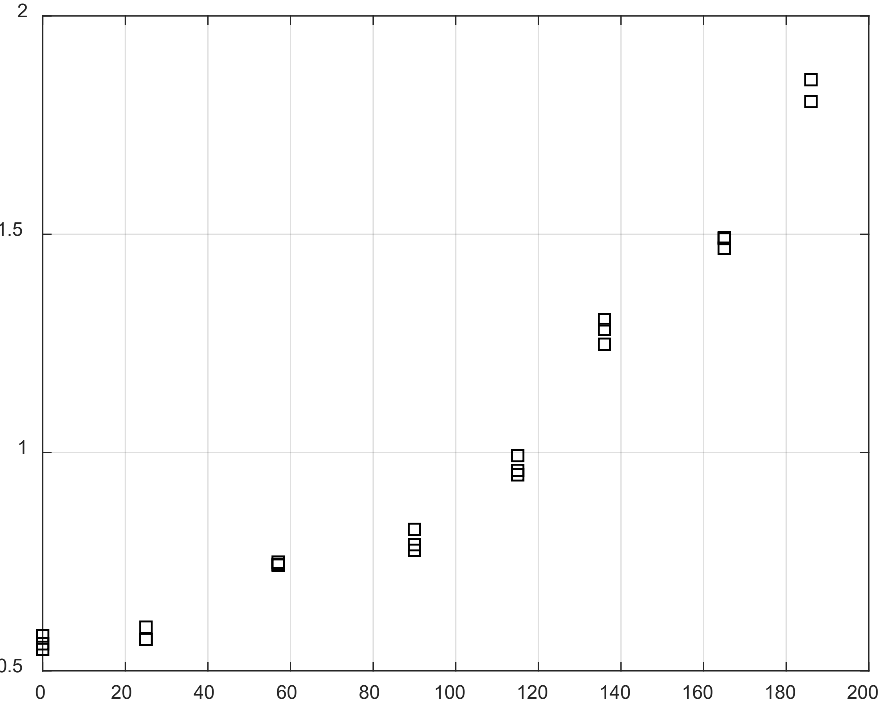
$$\textbf{Data} = \begin{pmatrix} 0 & 0.562 \\ 0 & 0.58 \\ 0 & 0.549 \\ 25 & 0.572 \\ 25 & 0.6 \\ 25 & 0.572 \\ 57 & 0.744 \\ 57 & 0.749 \\ 57 & 0.742 \\ 90 & 0.776 \\ 90 & 0.789 \\ 90 & 0.824 \\ 115 & 0.959 \\ 115 & 0.993 \\ 115 & 0.949 \\ 136 & 1.248 \\ 136 & 1.282 \\ 136 & 1.304 \\ 165 & 1.492 \\ 165 & 1.468 \\ 165 & 1.49 \\ 186 & 1.854 \\ 186 & 1.804 \end{pmatrix}$$

Example of ‘polyfit’ Matlab function 3 (cont.):

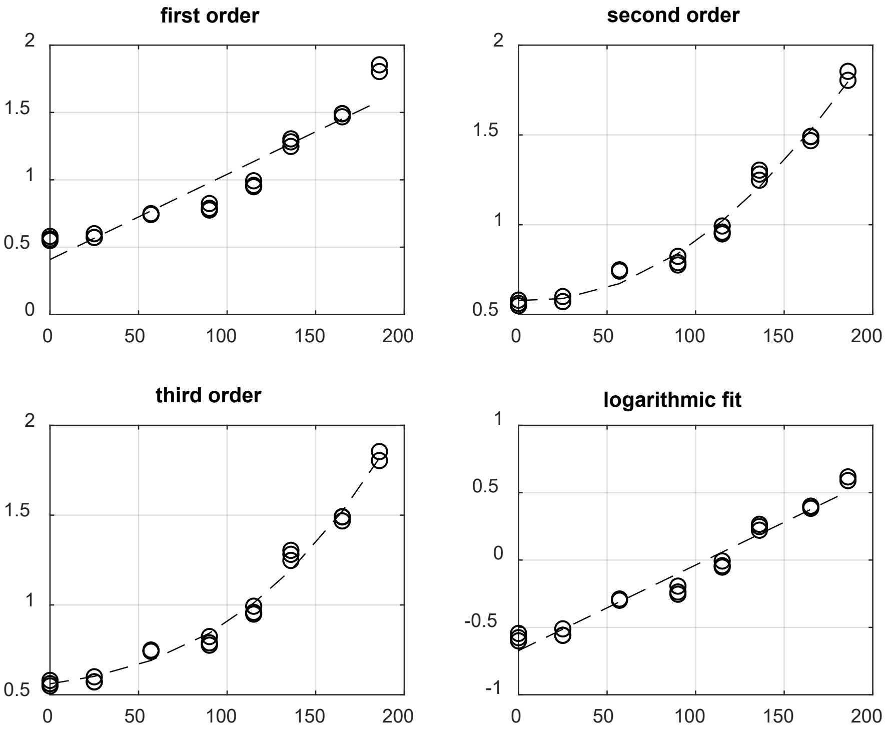
```
clear;clc;

figure(1);

plot(data(:,1),data(:,2),'ks');grid on;% plotting with black squares

x=data(:,1);

y=data(:,2);

m1=polyfit(x,y,1); %build a first order polynomial model ,m1 are the coefficients

p1=polyval(m1,x);

m2=polyfit(x,y,2);%build a second order polynomial model ,m2 are the coefficients

p2=polyval(m2,x);

m3=polyfit(x,y,3);%build a second order polynomial model ,m3 are the coefficients
```

Example of ‘polyfit’ Matlab function 3 (cont.):

```
p3=polyval(m3,x);

m4=polyfit(x,log(y),1);%build a logarithmic model of order 1
,m4 are the coefficients

p4=polyval(m4,x);

figure(2);

subplot(2,2,1),plot(x,y,'ko'),hold on,plot(x,p1,'k--');grid
on,title('first order');

subplot(2,2,2),plot(x,y,'ko'),hold on,plot(x,p2,'k--');grid
on,title('second order');

subplot(2,2,3),plot(x,y,'ko'),hold on,plot(x,p3,'k--');grid
on,title('third order');

subplot(2,2,4),plot(x,log(y),'ko'),hold on,plot(x,p4,'k--
');grid on,title('logarithmic fit');
```



Original data


first order

second order

third order

logarithmic fit

Original data and polynomial fit

Nonlinear functions solver: lsqcurvefit

- **lsqcurvefit:** A Matlab solver for non-linear models.

Syntax:

```
m = lsqcurvefit(modelfun, beta0,x,t)
```

Description:

Solve non-linear curve-fitting (data-fitting) problems in least-squares sense.

fun : Function you want to fit.

beta0 — Coefficients

x: Input data for model

t : Out put for model

- **Prediction**

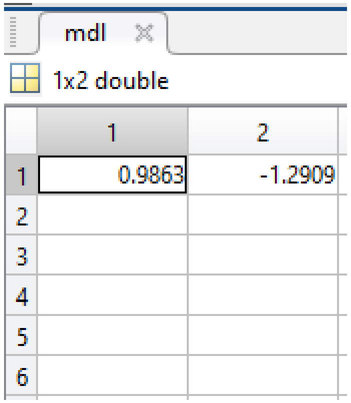
Syntax:

```
Ypredicted=modelfun(m,x);
```

Example of Non-linear regression function (1) : lsqcurvefit

```
clear;

clc;

rng(1);

x = linspace(0,3);

t = exp(-1.3*x) + 0.05*randn(size(x));

fun = @(b,x)b(1)*exp(b(2)*x); % Defining the Model function

b0 = [1/2,-2]; %Coefficient initiation

mdl = lsqcurvefit(fun,b0,x,t); % obtaining the coefficients.

ypredicted=fun(mdl,x);

plot(x,t,'ko',x,ypredicted,'b-')

legend('Data','Fitted exponential')

title('Data and Fitted Curve')
```

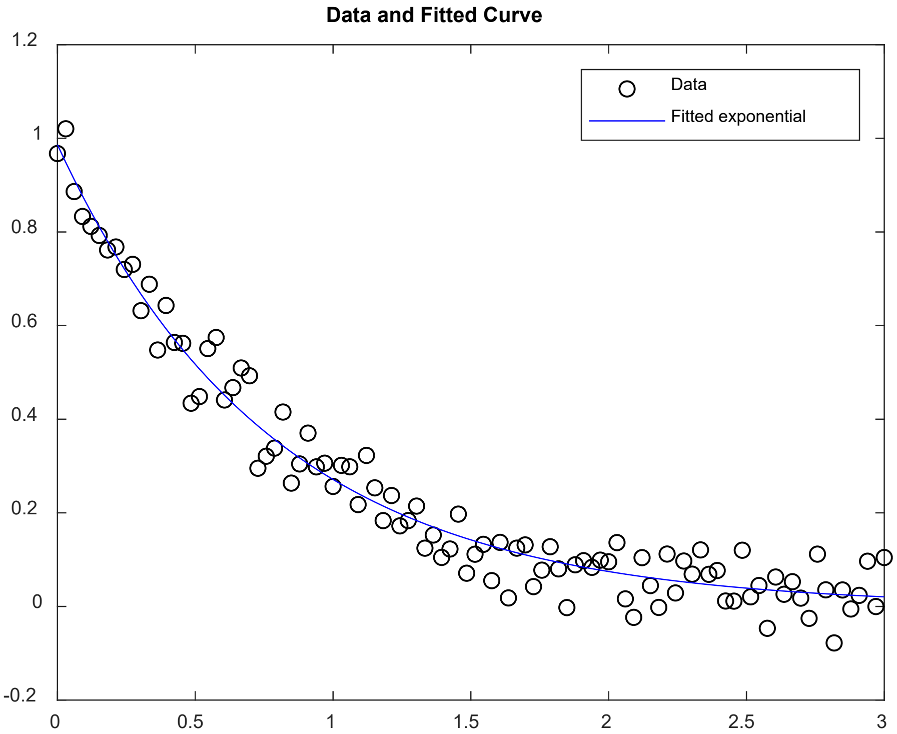
Example of Non-linear regression function (1) : lsqcurvefit (cont.)



mdl

1x2 double

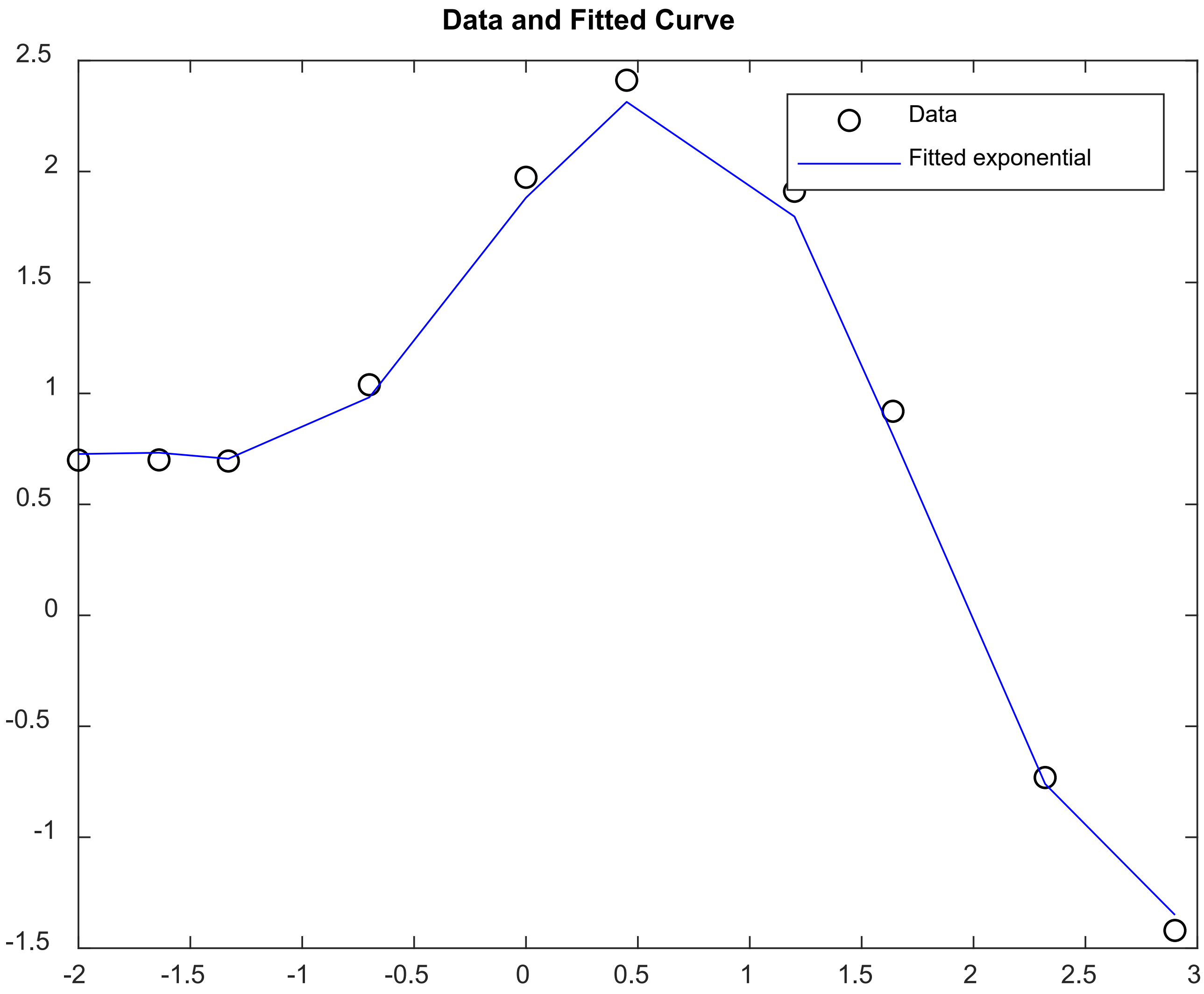
	1	2
1	0.9863	-1.2909
2		
3		
4		
5		
6		



Example of Non-linear regression function (2) : lsqcurvefit

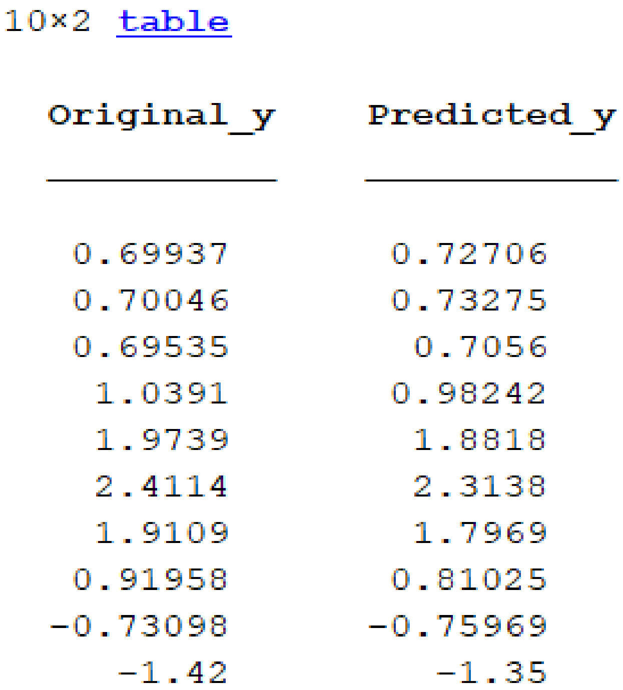
```
clear ;

clc;

rng(1);

x = [-2,-1.64,-1.33,-0.7,0,0.45,1.2,1.64,2.32,2.9];

t =
[0.699369,0.700462,0.695354,1.03905,1.97389,2.41143,
1.91091,0.919576,-0.730975,-1.42001];

fun = @(b,x)b(1)*cos(b(2)*x)+b(2)*sin(b(1)*x);

b = [1,0.2];

mdl = lsqcurvefit(fun,b,x,t);% mdl will be the
coefficients

ypredicted=fun(mdl,x);%This equals to yexpected
=mdl(:,1)*cos(mdl(:,2)*x)+mdl(:,2)*sin(mdl(:,1)*x)

plot(x,t,'ko',x,ypredicted,'b-')
```

Example of Non-linear regression function (2) : lsqcurvefit (cont.)



Example of Non-linear regression function (2) : lsqcurvefit (cont.)

```
legend('Data','Fitted exponential')

title('Data and Fitted Curve')

table( t(: ),ypredicted(:), 'VariableNames',...
    {' Original_y',' Predicted_y'})
```



```
10×2 table

 Original_y    Predicted_y
 __________    ___________

   0.69937       0.72706
   0.70046       0.73275
   0.69535        0.7056
    1.0391       0.98242
    1.9739        1.8818
    2.4114        2.3138
    1.9109        1.7969
   0.91958       0.81025
  -0.73098      -0.75969
     -1.42         -1.35
```

Non-linear regression Matlab function (fitnlm):

- **fitnlm:** A non-linear regression Matlab solver for non-linear models.

Syntax:

```
mdl = fitnlm(x,y,modelfun,beta0)
```

x: Predictor variables(Input)

y: Response variable(output)

Modelfun: Functional form of the model (model you want to fit)

Beta0: Coefficients

- **Predict:** Predict response of the regression model

Syntax:

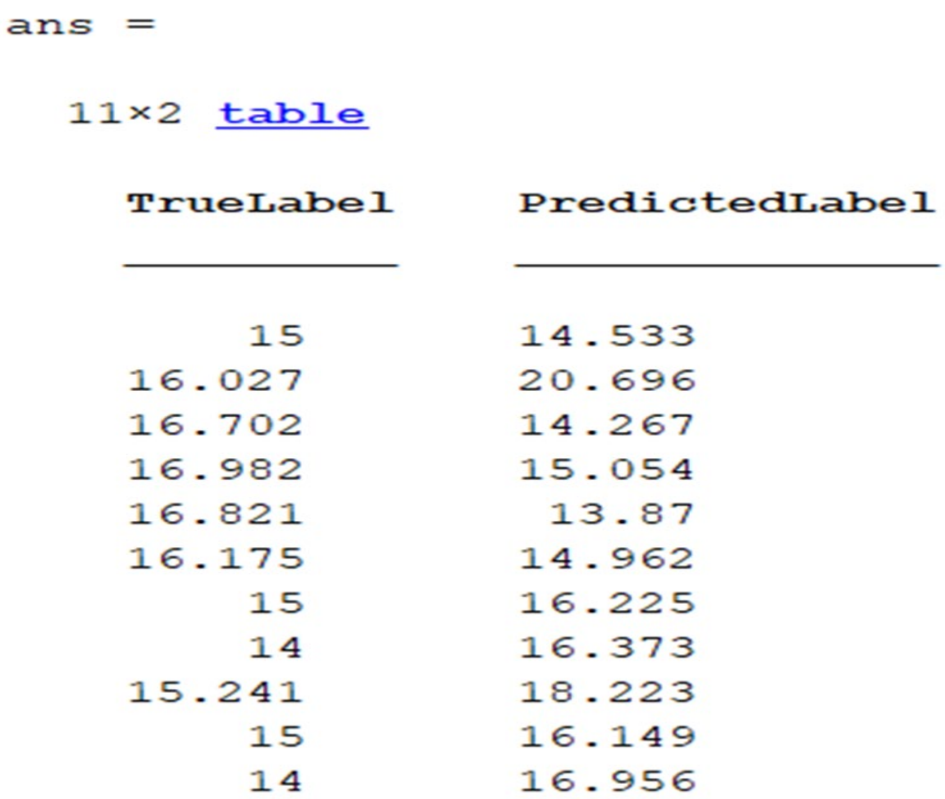
```
ypred = predict(mdl,Xnew);
```

Example of Nonlinear regression function 1: fitnlm

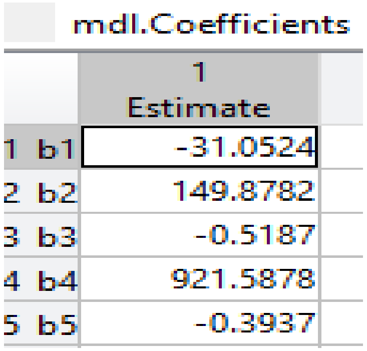
```
rng(1);
Filename='regression2.xlsx';
Sheetread='Sheet1';
Input1='A1:B406';
output1='C1:C406';
Input=xlsread(Filename,Sheetread,Input1); %Read
Microsoft Excel
Target=xlsread(Filename,Sheetread,output1 );
x=Input;
t=Target;
modelfun = @(b,x)(b(1) + b(2)*x(:,1).^b(3) +...
b(4)*x(:,2).^b(5)); % nonlinear model
```

Example of Nonlinear regression function 1: fitnlm (cont.)

```
beta0 = [-50 500 -1 500 -1]; % coefficients
initiation

mdl = fitnlm(x,t,modelfun,beta0);

y_expected = predict(mdl,x);

table( t (10:20 ),y_expected( 10:20 ),
'VariableNames',...

    {' TrueLabel',' PredictedLabel'}) %Show the
results of 1st  to 10th data of the output and
predicted output.

MSE_training= (mean((t - y_expected).^2));

RMSE_training = sqrt(mean((t - y_expected).^2));
```



```
ans =

  11×2 table

    TrueLabel      PredictedLabel
    _________      ______________

         15            14.533
     16.027            20.696
     16.702            14.267
     16.982            15.054
     16.821             13.87
     16.175            14.962
         15            16.225
         14            16.373
     15.241            18.223
         15            16.149
         14            16.956
```



mdl.Coefficients

	1 Estimate
1 b1	-31.0524
2 b2	149.8782
3 b3	-0.5187
4 b4	921.5878
5 b5	-0.3937

MSE_training =16.1174

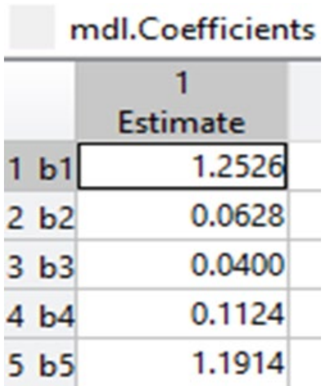
MSE_testing =4.0147

Example of Nonlinear regression function 2: fitnlm

```
clear;clc;
rng(1);
Filename='regression3.xlsx';
Sheetread='Sheet1';
Input1='A1:C13';
output1='D1:D13';
Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft
Excel
Target=xlsread(Filename,Sheetread,output1 );
x=Input;
t=Target;
beta = [1 1 1 1 1]; % coefficient initiation
fun = @(b,x)((b(1)*x(:,2)-
x(:,3)/b(5))./(1+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)));
mdl = fitnlm(x,t,fun,beta)
```

Example of Nonlinear regression function 2: fitnlm (cont.)

```
y_expected = predict(mdl,x);
table( t( 5:10 ),y_expected( 5:10 ), 'VariableNames',...
    {' Actual_Y',' PredictedY'}) %Show the results of 5th
to 10th data of the output and predicted output.
MSE_training= (mean((t - y_expected).^2));
RMSE_training = sqrt(mean((t - y_expected).^2));
```



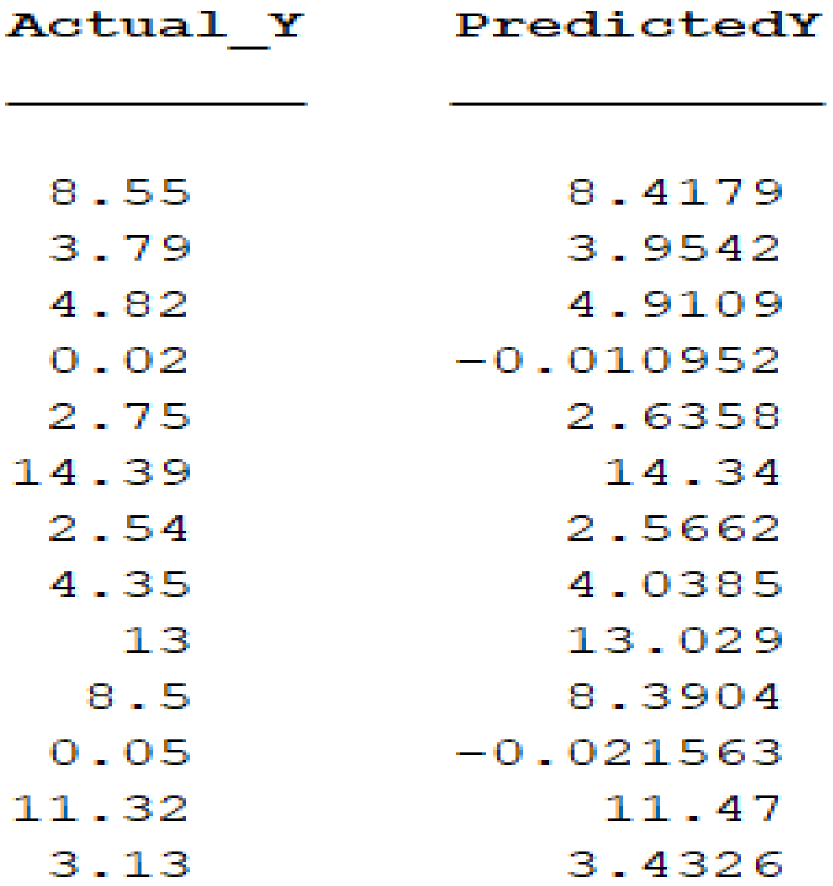
mdl.Coefficients

	1
	Estimate
1 b1	1.2526
2 b2	0.0628
3 b3	0.0400
4 b4	0.1124
5 b5	1.1914

MSE_training =0.0230

MSE_testing =0.1516

Example of Nonlinear regression function 2: fitnlm (cont.)



```
Actual_Y     PredictedY
________    ___________

  8.55          8.4179
  3.79          3.9542
  4.82          4.9109
  0.02       -0.010952
  2.75          2.6358
 14.39           14.34
  2.54          2.5662
  4.35          4.0385
    13          13.029
   8.5          8.3904
  0.05       -0.021563
 11.32           11.47
  3.13          3.4326
```

Example of Nonlinear regression function 3: fitnlm

```
Clear;clc;
rng(1);
Filename='regression4.xlsx';
Sheetread='Sheet1';
Input1='A1:C72';
output1='D1:D72';
Input=xlsread(Filename,Sheetread,Input1); %Read Microsoft
Excel
Target=xlsread(Filename,Sheetread,output1 );
x=Input;
t=Target;
[xn,sxn] = mapminmax(x');% Standardize x
[tn,stn]= mapminmax(t');% Standardize t
Sheetread1='Sheet2';
Input2='A1:C3';
```

Example of Nonlinear regression function 3: fitnlm (cont.)

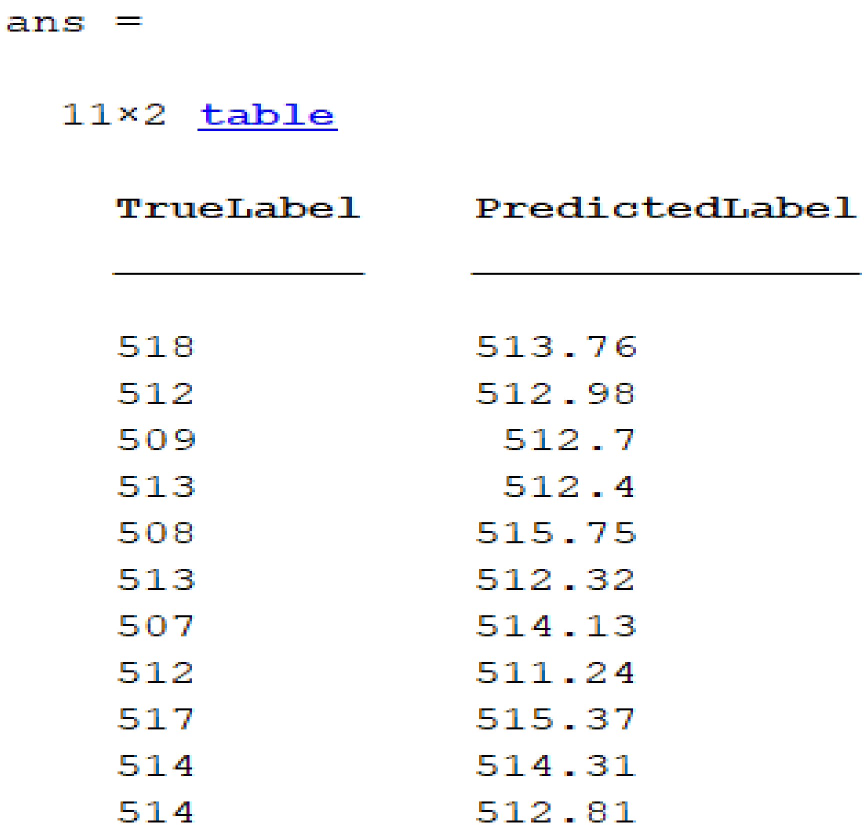
```
Target2 ='D1:D3';
Inputnew=xlsread(Filename,Sheetread1,Input2);
Targetnew=xlsread(Filename,Sheetread1,Target2 );
xnew=Inputnew;
tnew=Targetnew;
xnewn = mapminmax('apply',xnew',sxn); % The same Process
setting of
%standardization for x should also be applied for xnew.
%xnewn is the
%standardized xnew
xn=xn';% standardized x
tn=tn';% standardized t
xnewn=xnewn';%standardized xnew
beta = [1 1 1 1 1 1]; % coefficient initiation
```

Example of Nonlinear regression function 3: fitnlm (cont.)

```
fun=@(b,xn)b(1)+b(2)*xn(:,1)+b(3)*xn(:,2)+b(4)*xn(:,3)+b(5
)*(xn(:,1).^2)+b(6)*((xn(:,1).*xn(:,2).*xn(:,3)));%
nonlinear model with standardized x

mdl = fitnlm(xn,tn,fun,beta);% find coeffcients(beta) of
model(fun )using normalized x and t

yfitn = predict(mdl,xn);% make prediction based on
normalized x

yfit = mapminmax('reverse', yfitn,stn); % To reverse the
prediction to original state using the same process
setting of t

table( t( 10:20 ),yfit( 10:20 ), 'VariableNames',...

    {' TrueLabel',' PredictedLabel'}) %Show the results of
5th  to 10th data of the output and predicted output.

MSE_training=sum((yfit-t).^2)/numel(t); % Calculate MSE
for data

RMSE_training=sqrt(sum((yfit-t).^2)/numel(t)); % Calculate
RMSE for data
```

Example of Nonlinear regression function 3: fitnlm (cont.)

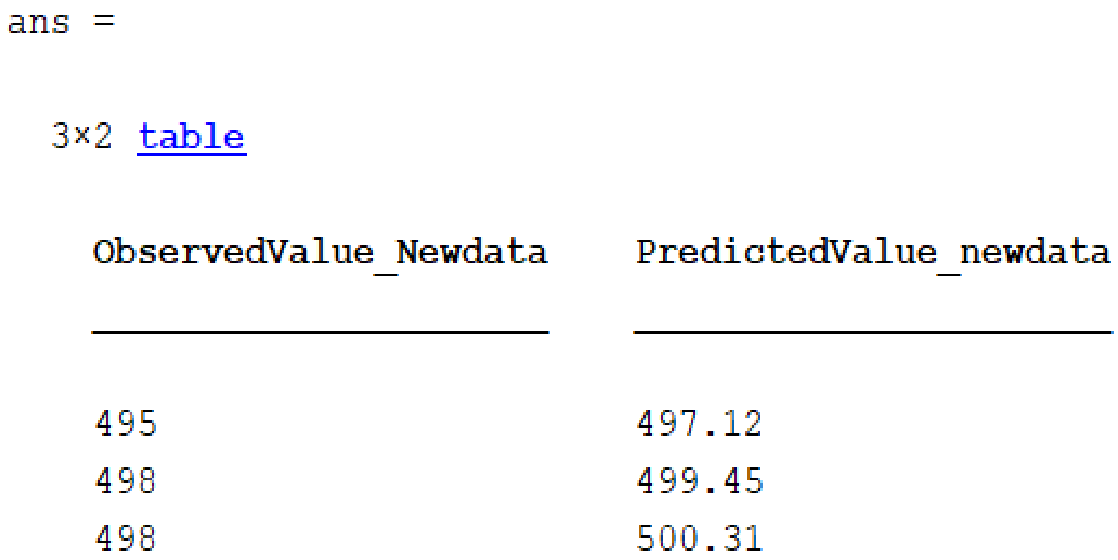
```
ynewn=predict(mdl,xnewn);% make prediction based on
normalized new data

ynew = mapminmax('reverse', ynewn,stn); % To reverse the
normalized ynew and use the processing setting of t

table(tnew(:),ynew(:),'VariableNames',{'ObservedValue_Newd
ata',' PredictedValue_newdata'}) % show data in output and
predicted output

% MSE_testing1=mse(tnew,ynew);

MSE_testing=sum((tnew-ynew).^2)/numel(tnew); % Calculate
MSE for new data

RMSE_testing=sqrt(sum((tnew-ynew).^2)/numel(tnew)); %
Calculate RMSE for new data

Errorpercentage=((ynew-tnew)./tnew)*100; % Calculate error
percentage for tnew and ynew
```



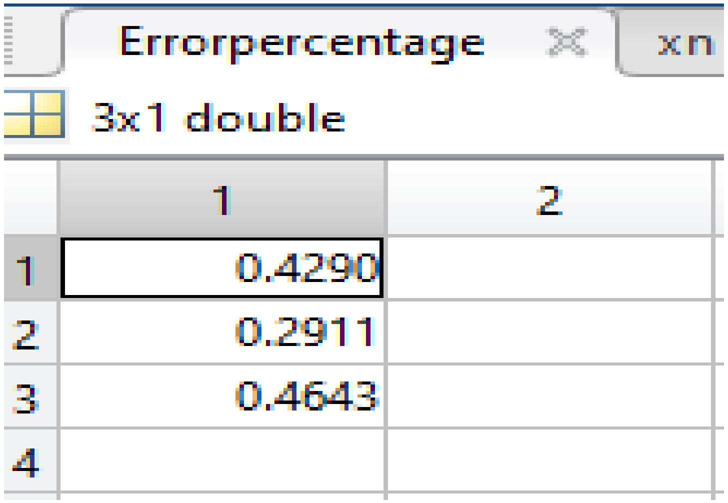
```
ans =

  11×2 table

    TrueLabel    PredictedLabel
    _________    ______________

    518          513.76
    512          512.98
    509           512.7
    513           512.4
    508          515.75
    513          512.32
    507          514.13
    512          511.24
    517          515.37
    514          514.31
    514          512.81
```

MSE_training =11.0469

MSE_testing =3.98557



```
ans =

  3×2 table

    ObservedValue_Newdata    PredictedValue_newdata
    _____________________    ______________________

    495                      497.12
    498                      499.45
    498                      500.31
```



	1	2
1	0.4290	
2	0.2911	
3	0.4643	
4		

RMSE_training =3.3237

RMSE_testing =1.9964

Examples of non-linear model functions (Using combination of Power , Exponential, polynomial functions) :

1.
```
fun=@(b,x)b(1)+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)+b(5)*(x(:,1).^2)+b(6)*((x(:,1).*x(:,2).*x(:,3)));
```

2.
```
fun=@(b,x)b(1)+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)+b(5)*(x(:,1).^2);
```

3.
```
fun=@(b,x)b(1)+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)+b(5)*(x(:,1).^2)+b(6)*x(:,2).^2;
```

4.
```
fun=@(b,x)b(1)+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)+b(5)*(x(:,1).^2)+b(6)*exp(x(:,1));
```

5.
```
fun=@(b,x)b(1)+b(2)*x(:,1)+b(3)*x(:,2)+b(4)*x(:,3)+b(5)*(x(:,1).^2)+b(6)*x(:,3)  +b(7)*((x(:,1).*x(:,2).*x(:,3)));
```

6.
```
fun=@(b,xn)b(1)+b(2)*xn(:,1)+b(3)*xn(:,2)+b(4)*xn(:,3)+b(5)*(xn(:,1).^2)+b(6)*((xn(:,1).*xn(:,2).*xn(:,3))+b(7).*exp(b(8)*xn(:,3)))
```

References for regression functions and methods:

- https://au.mathworks.com
- S. Araghinejad, Data-Driven Modeling: Using MATLAB® in Environmental Engineering