

---

## Table of Contents

.....	1
.....	2
INITIALIZATION .....	2
.....	2
Linearizing the data .....	2
Linearize the data using linear function .....	2
Linearize the data using power function .....	2
Linearize the data using exponential function .....	2
Linearize the data using logarithmic function .....	3
.....	3
FORMATTED TEXT/FIGURE DISPLAYS .....	3
.....	5
COMMAND WINDOW OUTPUT .....	5
.....	6
ACADEMIC INTEGRITY STATEMENT .....	6

```
function [SSE_final, SST] = M3_Regression_001_30

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENGR 132
% Program Description
% The program finds a model for estimating price of enzyme according
% to
% their Km value and output the estimate price.
%
% Function Call
% function [SSE_final, SST] = M3_Regression_001_30
%
% Input Arguments
% No input argument
%
% Output Arguments
% SSE_final: the SSE value of the linearized data
% SST: the SST value of the linearized data
%
% Assignment Information
%   Assignment:      M3
%   Team Mmember:    Luming Lin, lin971@purdue.edu
%                   Surya Manikhandan, smanikha@purdue.edu
%                   Julius Mesa, jmesa@purdue.edu
%                   Alex Norkus, anorkus@purdue.edu
%   Team ID:         001-30
%   Academic Integrity:
%   [ ] We worked with one or more peers but our collaboration
%       maintained academic integrity.
%   Peers we worked with: Name, login@purdue [repeat for each]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

---

# INITIALIZATION

---

```
clc;clearvars;
data = readmatrix('Data_NovelEnzymes_priceCatalog'); %import the data
      from the csv file
price_measured = data(:,2); % separate the price from the data
Km_measured = data(:,1); %separates the Km value from the data
```

---

## Linearizing the data

### Linearize the data using linear function

```
coe1 = polyfit(Km_measured,price_measured,1); %find the coefficient of
      trendline of the model
M1 = coe1(1);
B1 = coe1(2);
m1 = M1; %calculate the original model coefficient
b1 = B1;
linearized_predict1 = M1 .* Km_measured + B1; %calculate the predict
      value of price using the linearized model
predict1 = polyval(coe1,Km_measured); % calculate the predict value of
      price using the original model
SSE1 = sum((price_measured - linearized_predict1).^2); %calculate the
      SSE value of the model
```

### Linearize the data using power function

```
coe2 = polyfit(log10(Km_measured),log10(price_measured),1); %find the
      coefficient of trendline of the model
M2 = coe2(1);
B2 = coe2(2);
m2 = M2; %calculate the original model coefficient
b2 = 10 ^ B2;
linearized_predict2 = M2 .* log10(Km_measured) + B2; %calculate the
      predict value of price using the linearized model
predict2 = b2 .* Km_measured .^ m2; % calculate the predict value of
      price using the original model
SSE2 = sum((log10(price_measured) -
      linearized_predict2).^2); %calculate the SSE value of the model
```

### Linearize the data using exponential function

```
coe3 = polyfit(Km_measured,log10(price_measured),1); %find the
      coefficient of trendline of the model
M3 = coe3(1);
```

---

```

B3 = coe3(2);
m3 = M3; %calculate the original model coefficient
b3 = 10 ^ B3;
linearized_predict3 = M3 .* Km_measured + B3; %calculate the predict
value of price using the linearized model
predict3 = b3 .* 10 .^ (Km_measured .* m3); % calculate the predict
value of price using the original model
SSE3 = sum((log10(price_measured) -
linearized_predict3).^2); %calculate the SSE value of the model

```

## Linearize the data using logarithmic function

```

coe4 = polyfit(log(Km_measured),price_measured,1); %find the
coefficient of trendline of the model
M4 = coe4(1);
B4 = coe4(2);
m4 = M4; %calculate the original model coefficient
b4 = B4;
linearized_predict4 = M4 .* log(Km_measured) + B4; %calculate the
predict value of price using the linearized model
predict4 = m4 .* Km_measured + b4; % calculate the predict value of
price using the original model
SSE4 = sum((price_measured - linearized_predict4).^2); %calculate the
SSE value of the model

```

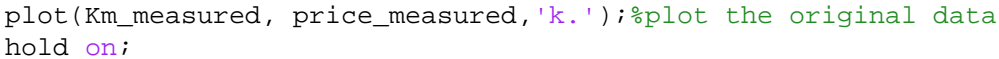
---

## FORMATTED TEXT/FIGURE DISPLAYS

```

SSE_value = [SSE1,SSE2,SSE3,SSE4]; %combine the SSE value of all four
models
SSE_final = min(SSE_value); %find the minimum SSE value which
indicates the best model
choose = find(SSE_value == SSE_final); %find which model is the best

figure(1)
if choose == 1 % if the linear function best fits the data
    SST = sum((price_measured - mean(price_measured)).^2); %calculate
the SST value of the linearized data
    r2_1 = 1-(SSE1/SST); %calculate the r^2 value of the linearized
data
    r2 = r2_1;
    plot(Km_measured, price_measured,'k.');
```



```

    hold on;
    plot(Km_measured, predict1,'r-'); %plot the model
    title({'The Ezyme USD Price Per Pound of each'; 'Michaelis Constant
from 157-350'})
    xlabel('Michaelis Constant (uM)');
    ylabel('Price (USD($)/lb)')
    function_output = sprintf("Model function: %.3f * Km + %.3f\nSSE:
%.3f, SST: %.3f, r^2: %.3f",b1,m1,SSE_final,SST,r2); %display the
funciton of the model also the SST,SSE and r^2 value

```

---

```

        legend('Novel Enzymes Price Catalog per Michaelis Constant',
function_output, 'location', 'best')
        grid on;

elseif choose == 2 % if the power function best fits the data
    SST = sum((log(price_measured) -
mean(log(price_measured))).^2); %calculate the SST value of the
linearized data
    r2_2 = 1-(SSE2/SST); %calculate the r^2 value of the linearized
data
    r2 = r2_2;
    plot(Km_measured, price_measured,'k.') %plot the original data
    hold on;
    plot(Km_measured, predict2,'r-'); %plot the model
    title({'The Ezyme USD Price Per Pound of each';'Michaelis Constant
from 157-350'})
    xlabel('Michaelis Constant (uM)');
    ylabel('Price (USD($)/lb)')
    function_output = sprintf("Model function:
%.3f * Km ^ (%.3f)\nSSE: %.3f, SST: %.3f, r^2:
%.3f",b2,m3,SSE_final,SST,r2); %display the funciton of the model
also the SST,SSE and r^2 value
    legend('Novel Enzymes Price Catalog per Michaelis Constant',
function_output, 'location', 'best')
    grid on

elseif choose == 3 % if the exponential function best fits the data
    SST = sum((log(price_measured) -
mean(log(price_measured))).^2); %calculate the SST value of the
linearized data
    r2_3 = 1-(SSE3/SST); %calculate the r^2 value of the linearized
data
    r2 = r2_3;
    plot(Km_measured, price_measured,'k.') %plot the original data
    hold on;
    plot(Km_measured, predict3,'r-'); %plot the model
    title({'The Ezyme USD Price Per Pound of each';'Michaelis Constant
from 157-350'})
    xlabel('Michaelis Constant (uM)');
    ylabel('Price (USD($)/lb)')
    function_output = sprintf("Model function: %.3f * 10 ^ (%.3f * Km)
\nSSE: %.3f, SST: %.3f, r^2: %.3f",b3,m3,SSE_final,SST,r2); %display
the funciton of the model also the SST,SSE and r^2 value
    legend('Novel Enzymes Price Catalog per Michaelis Constant',
function_output, 'location', 'best')
    grid on

elseif choose == 4 % if the power logarithmic best fits the data
    SST = sum((price_measured - mean(price_measured)).^2); %calculate
the SST value of the linearized data
    r2_4 = 1-(SSE4/SST); %calculate the r^2 value of the linearized
data

```

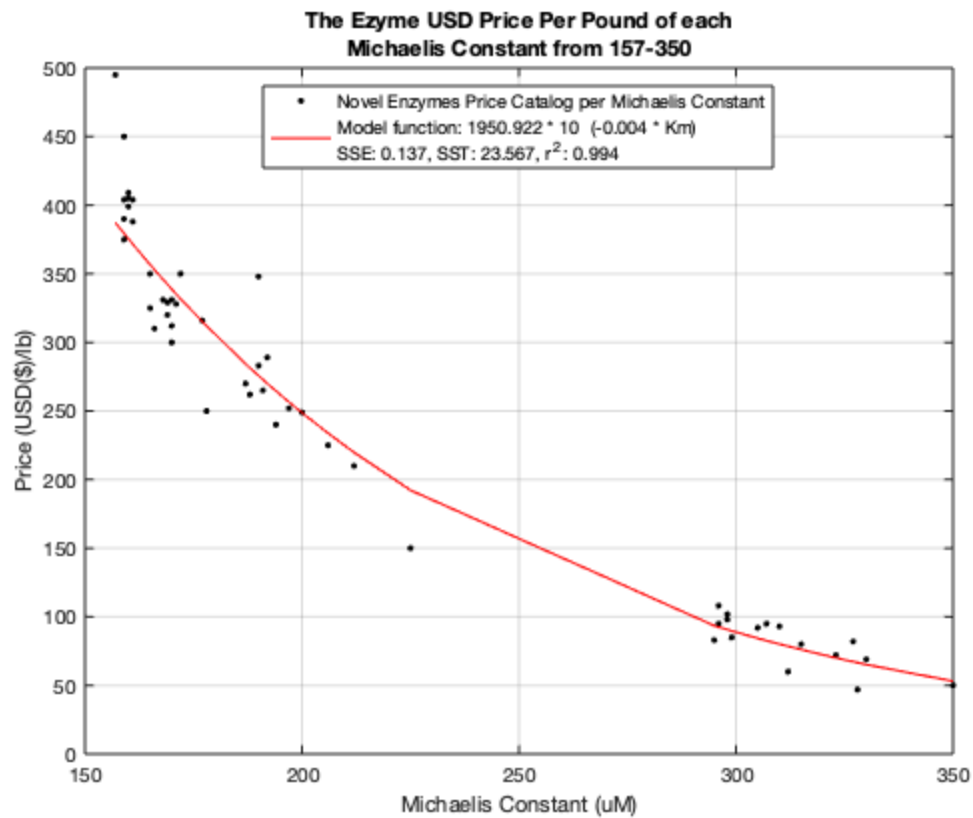
---

```

r2 = r2_4;
plot(Km_measured, price_measured, 'k.') %plot the original data
hold on;
plot(Km_measured, predict4, 'r-'); %plot the model
title({'The Ezyme USD Price Per Pound of each'; 'Michaelis Constant
from 157-350'})
xlabel('Michaelis Constant (uM)');
ylabel('Price (USD($)/lb)')
function_output = sprintf("Model function: %.3f * log10(Km) + %.3f
\nSSE: %.3f, SST: %.3f, r^2: %.3f", b4, m4, SSE_final, SST, r2); %display
the functon of the model also the SST, SSE and r^2 value
legend('Novel Enzymes Price Catalog per Michaelis Constant',
function_output, 'location', 'best')
grid on

```

end



## COMMAND WINDOW OUTPUT

```

fprintf("The chosen model is Exponential funciton: %.3f * 10 ^ (%.3f *
Km)\nSSE: %.3f, SST: %.3f, r^2: %.3f", b3, m3, SSE_final, SST, r2);

```

---

*The chosen model is Exponential function:  $1950.922 * 10^{(-0.004 * Km)}$   
SSE: 0.137, SST: 23.567,  $r^2$ : 0.994*

---

## ACADEMIC INTEGRITY STATEMENT

We have not used source code obtained from any other unauthorized source, either modified or unmodified. Neither have we provided access to my code to another. The function we are submitting is our own original work.

*Published with MATLAB® R2019b*