Manik Daya

2017140008

BE IT

EXPERIMENT 4: Data Fitting

Theory: Curve fitting is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints. Curve fitting can involve either interpolation, where an exact fit to the data is required, or smoothing, in which a "smooth" function is constructed that approximately fits the data. A related topic is regression analysis, which focuses more on questions of statistical inference such as how much uncertainty is present in a curve that is fit to data observed with random errors. Fitted curves can be used as an aid for data visualization, to infer values of a function where no data are available, and to summarize the relationships among two or more variables. Extrapolation refers to the use of a fitted curve beyond the range of the observed data, and is subject to a degree of uncertainty since it may reflect the method used to construct the curve as much as it reflects the observed data.

CODE:

EXP 4 - DATA FITTING

In [1]:
```
from pylab import*
```

In [2]:
```
from scipy.optimize import curve_fit
```

In [3]:
```
x_data,y_data =loadtxt('data.txt',unpack=True)
```

In [4]:
```
x_data
```

Out[4]:
```
array([8.299, 7.399, 6.899, 6.299, 5.499, 5.199])
```

In [5]:
```
y_data
```

Out[5]:
```
array([3.199, 2.599, 2.25 , 1.809, 1.309, 1.119])
```

In [6]:
```
def linearfunction(x,intercept,slope):
    y = intercept+slope * x
    return y
```

In [25]:
```
linearfunction(2,3,3)
```

Out[25]: 9

In [16]:
```
popt,cov=curve_fit(linearfunction,x_data,y_data)
```

In [17]:
```
print(popt)
```

```
[-2.40615359  0.67489826]
```

In [18]:
```
print(cov)
```

```
[[ 2.81352652e-03 -4.15417792e-04]
 [-4.15417792e-04  6.29516283e-05]]
```

In [19]:
```
inter = popt[0]
slope = popt[1]
```

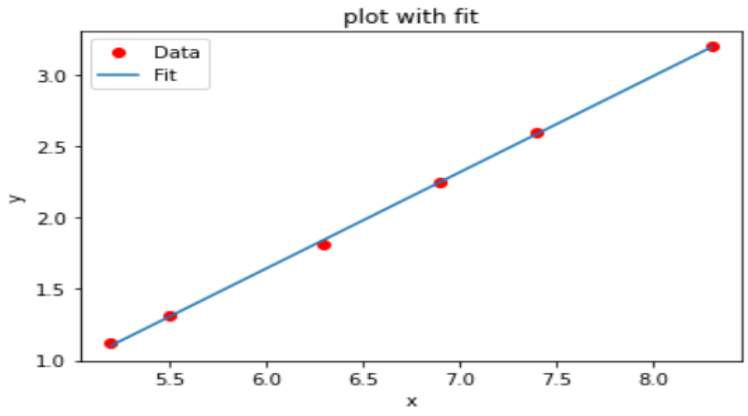
In [20]:
```
d_inter = sqrt(cov[0][0])
d_slope = sqrt(cov[1][1])
```

In [27]:
```
d_slope
```

Out[27]: 0.007934206216589924

In [21]:
```
plot(x_data,y_data,'ro',label='Data')
yfit = inter + slope*x_data
plot(x_data,yfit,label='Fit')
legend()
xlabel('x')
ylabel('y')
title('plot with fit')
```

Out[21]: Text(0.5, 1.0, 'plot with fit')



In [22]:
```
print(f'The slope = {slope}, with uncertainty {d_slope}')
```

```
The slope = 0.6748982556897472, with uncertainty 0.007934206216589924
```

In [23]:
```
print(f'The intercept = {inter}, with uncertainty {d_inter}')
```

```
The intercept = -2.4061535892966415, with uncertainty 0.053042685870017224
```

DATA FIT WITH ERROR

In [78]:
```
def linearFunc(x,intercept,slope):
 y = intercept + slope * x
 return y
```

In [79]:
```
xdata,ydata,d_y = loadtxt('data_with_er.txt',unpack=True)
```

In [80]:
```
print(xdata)
```

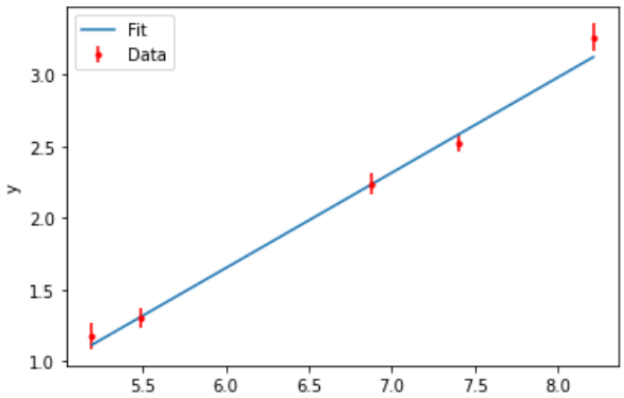
```
[8.213 7.402 6.876 5.491 5.196]
```

In [81]:
```
print(d_y)
```

```
[0.0971 0.0559 0.0708 0.0683 0.0893]
```

In [91]:
```
# Create a graph showing the data.
errorbar(xdata,ydata,yerr=d_y,fmt='r.',label='Data')
# Compute a best fit line from the fit intercept and slope.
yfit = inter + slope*xdata
# Create a graph of the fit to the data. We just use the ordinary plot
# command for this.
plot(xdata,yfit,label='Fit')
# Display a legend, label the x and y axes and title the graph.
legend()
xlabel('x')
ylabel('y')
```

Out[91]:
```
Text(0, 0.5, 'y')
```



x

In [85]:
```
print(f'The slope = {slope}, with uncertainty {d_slope}')
print(f'The intercept = {inter}, with uncertainty {d_inter}')
```

```
The slope = 0.6656028702881751, with uncertainty 0.03549213604200107
The intercept = -2.3430681719234285, with uncertainty 0.239532487804196
```

In [86]:
```
chisqr = sum((ydata-linearFunc(xdata,inter,slope))**2/d_y**2)
dof = len(ydata) - 2
chisqr_red = chisqr/dof
print(f'Reduced chi^2 = {chisqr_red}')
```

```
Reduced chi^2 = 1.2633310164063059
```

Conclusion: Data fitting is done in a linear function of form y= mx+c. In the first part, the linea r function is fit with curve_fit () and it return two parameters one contains best solution in this case they are [-2.40615359, 0.67489826] and other contains covariance matrix of 2*2 to det ermine the uncertainties and correlation between parameters. We compute the best y fit valu es and plot the graph, then we display the best values for the slope and intercept with uncert ainties. The best fit value for slope is 0.6748982556897472, with uncertainty 0.00793420621 6589924 and intercept -2.4061535892966415, with uncertainty 0.053042685870017224.

In the second part we performed the data fitting for a data with given error and performed the chi square test to check the fitting, reduced chi^2 value was 1.26 which is near to 1 so it indi cates that the data has been fitted accurately.

In the third part the error is increased in the data and got the value of reduced chi square is about 6.713 which shows it a bad fit.