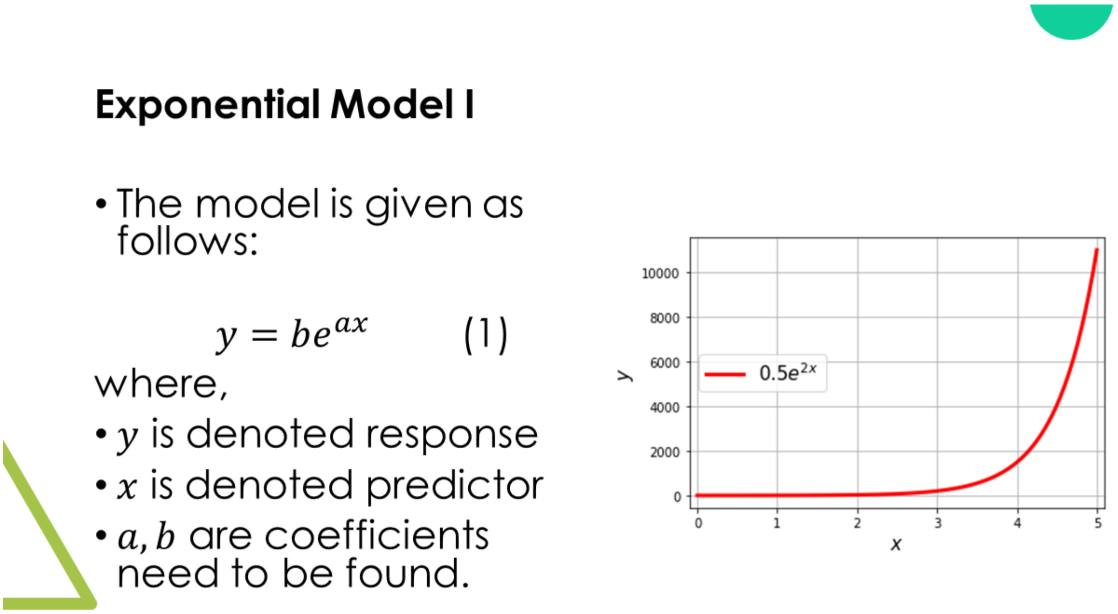
NON-LINEAR REGRESSION

Eksponensial Tipe I



Contoh

Diberikan 14 data berikut ini:

i	x_i	y_i	i	x_i	y_i
1	0	0.07	**8**	1	2.33
2	0.14	0.92	**9**	1.14	3.11
3	0.28	0.96	**10**	1.28	2.98
4	0.42	1.14	**11**	1.42	4.52
5	0.57	1.04	**12**	1.57	5.15
6	0.71	1.44	**13**	1.71	6.23
7	0.85	1.6	**14**	1.85	8.4

Tentukanlah model regresi tak linier eksponensial I

In [1]:

```
import numpy as np
x = np.array([0, 0.14, 0.28, 0.42, 0.57, 0.71, 0.85, 1, 1.14, 1.28, 1.42, 1.57, 1.71
y = np.array([0.07, 0.92, 0.96, 1.14, 1.04, 1.44, 1.6, 2.33, 3.11, 2.98, 4.52, 5.15,
```

Simple Idea

- Generally, to handle this problem we need to use logarithm such as

$$\ln y = \ln(be^{ax})$$

or can be written as

$$\ln y = \ln b + ax \qquad (2)$$

Thus (2) can be solved as in simple linear regression to find coefficients ($\ln b$ and a).

Mengubah data tabel

In [2]:
```
m = len(y)
sum_x =  np.sum(x)
sum_x2 =  np.sum(x**2)

ln_y =  np.log(y)
sum_ln_y = np.sum(ln_y)
sum_xln_y =  np.sum(ln_y*x)
```

Menentukan Matriks A dan vektor y

In [3]:
```
A = np.array([[m, sum_x],[sum_x, sum_x2]])
print(A)
b = np.array([sum_ln_y, sum_xln_y])
print(b)
```

```
[[14.      12.94   ]
 [12.94    16.5918]]
[ 8.39896312 16.03045785]
```

Cari koefisien

In [4]:
```
c = np.linalg.inv(A)@b
print('Coefficients:')
print(c)
```

```
Coefficients:
[-1.04994646  1.78502423]
```

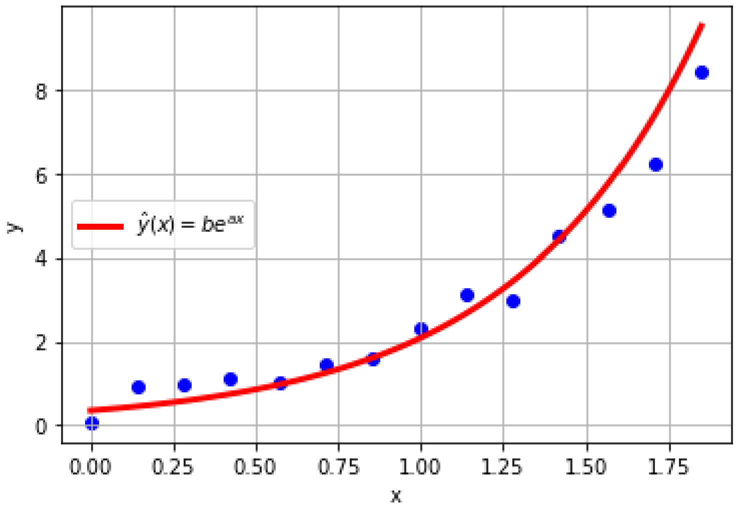
Definisikan fungsi hampiran

In [5]:
```
def Non_reg(a1,a0,x):
  return np.exp(a0)*np.exp(a1*x)
```

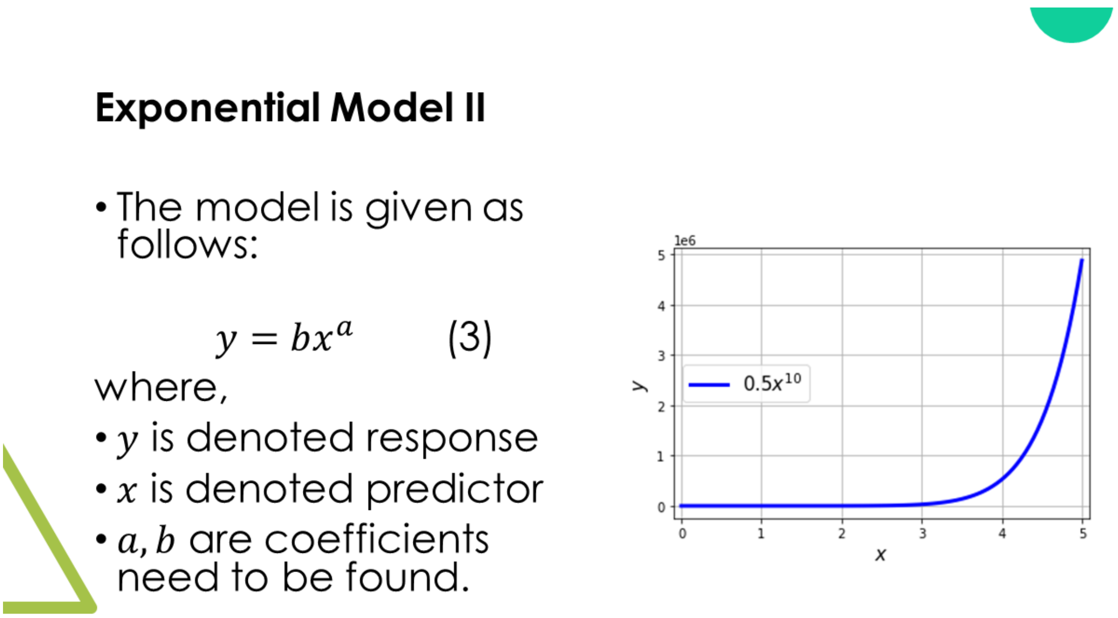
Plot data dan fungsi

In [6]:
```
import matplotlib.pyplot as plt #library untuk plot
xp = np.linspace(min(x), max(x), 100)
```

```
yp = Non_reg(c[1],c[0],xp)

plt.plot(xp,yp, color = 'red', linewidth=3)
plt.scatter(x,y,  color='blue')
plt.xlabel("x")
plt.ylabel("y")
plt.legend(('$\hat{y}(x)=b e^{ax}$',), loc='center left')
plt.grid()
plt.show()
```



Eksponensial Tipe II



Contoh

Diberikan 14 data berikut ini:

i	x_i	y_i	i	x_i	y_i
1	0.02	0.17	**8**	1.08	2.16
2	0.15	1.01	**9**	1.23	2.13
3	0.31	1.26	**10**	1.38	2.22
4	0.46	1.4	**11**	1.54	2.37
5	0.62	1.79	**12**	1.69	2.2
6	0.77	1.77	**13**	1.85	2.25
7	0.92	1.87	**14**	2	2.49

Tentukanlah model regresi tak linier eksponensial tipe II

menyiapkan data

In [7]:
```
import numpy as np
x = np.array([0.02, 0.15, 0.31, 0.46, 0.62, 0.77, 0.92, 1.08, 1.23, 1.38, 1.54, 1.69
y = np.array([0.17, 1.01, 1.26, 1.4, 1.79, 1.77, 1.87, 2.16, 2.13, 2.22, 2.37, 2.2,
```

Simple Idea

- Generally, to handle this problem we need to use logarithm such as

$$\ln y = \ln(bx^a)$$

or can be written as

$$\ln y = \ln b + a \ln x \qquad (4)$$

Thus (4) can be solved as in simple linear regression to find coefficients ($\ln b$ and a).

Mengubah data tabel

In [8]:
```
m =  len(x)

ln_x =  np.log(x)
sum_ln_x =  np.sum(ln_x)
sum_ln_x2 =  np.sum(ln_x**2)

ln_y =  np.log(y)
sum_ln_y =  np.sum(ln_y)
sum_xln_y =  np.sum(ln_y*ln_x)
```

Definisikan Matriks A dan vektor y

In [9]:
```
A = np.array([[m,sum_ln_x],[sum_ln_x,sum_ln_x2]])
print(A)
b = np.array([sum_ln_y,sum_xln_y])
print(b)
```

```
[[14.         -5.70873446]
 [-5.70873446 22.65463515]]
```

```
[6.28300868 8.29154466]
```

Hitung Koefisien

In [10]:
```
c = np.linalg.inv(A)@b
print('Coefficients:')
print(c)
```

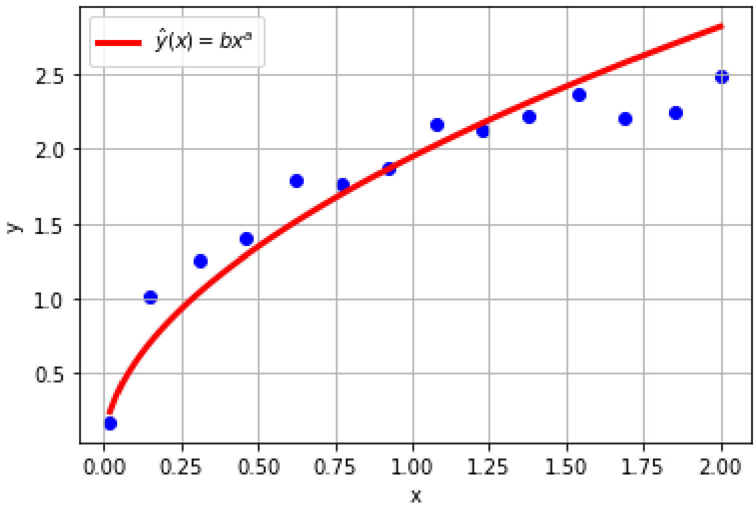
```
Coefficients:
[0.6665144  0.53395247]
```

Definisikan fungsi hampiran

In [11]:
```
def Non_reg(a1,a0,x):
  return np.exp(a0)*x**a1
```

Plot data

In [12]:
```
import matplotlib.pyplot as plt #library untuk plot
xp = np.linspace(min(x), max(x), 100)
yp = Non_reg(c[1],c[0],xp)

plt.plot(xp,yp, color = 'red', linewidth=3)
plt.scatter(x,y,  color='blue')
plt.xlabel("x")
plt.ylabel("y")
plt.legend(('$\hat{y}(x)=b x^a$',), loc='upper left')
plt.grid()
plt.show()
```



Machine Learning Approach

Langkah 1. Menyiapkan data

Pada langkah ini menyiapkan data advertising.

In [13]:
```
import pandas as pd

url = 'http://bit.ly/Test-PHN'
data = pd.read_csv(url, index_col=0)

data
```

Out[13]:

	TV	radio	newspaper	sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9
...	...	...	...	...
196	38.2	3.7	13.8	7.6
197	94.2	4.9	8.1	9.7
198	177.0	9.3	6.4	12.8
199	283.6	42.0	66.2	25.5
200	232.1	8.6	8.7	13.4

200 rows × 4 columns

Langkah 2. Membagi data menjadi 80\% training dan 20\% testing

In [14]:

```
import numpy as np
msk = np.random.rand(len(data)) < 0.8
train = data[msk]
test = data[~msk]
test.head()
```

Out[14]:

	TV	radio	newspaper	sales
4	151.5	41.3	58.5	18.5
9	8.6	2.1	1.0	4.8
23	13.2	15.9	49.6	5.6
35	95.7	1.4	7.4	9.5
39	43.1	26.7	35.1	10.1

Langkah 3. Menyiapkan data x (TV) dan y(sales)

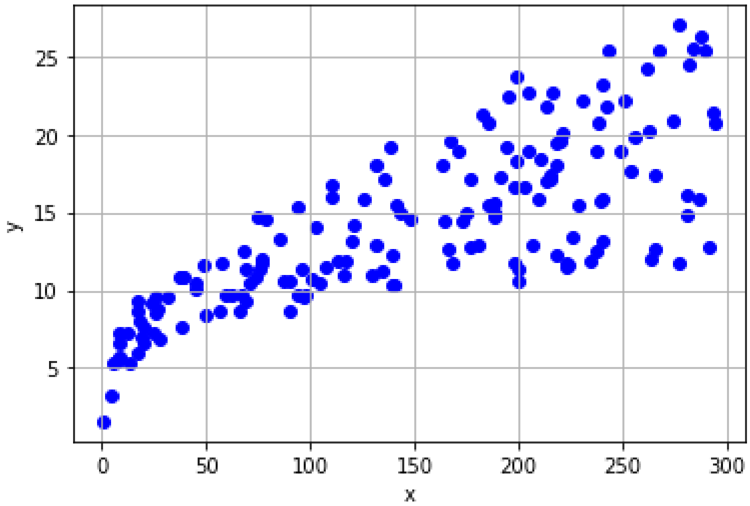
In [15]:

```
m = len(train.TV) #number of rows data

x = np.asanyarray(train[['TV']])
y = np.asanyarray(train[['sales']])
```

plot sebaran data

In [16]:

```
import matplotlib.pyplot as plt #library untuk plot

plt.scatter(x,y,  color='blue')
plt.xlabel("x")
plt.ylabel("y")
plt.grid()
plt.show()
```



Langkah 4. Menentukan model yang akan digunakan.

Dalam hal ini akan menggunakan model eksponensial II, selanjutnya mengubah data

In [17]:
```
m =  len(y)

ln_x =  np.log(x)
sum_ln_x =  np.sum(ln_x)
sum_ln_x2 =  np.sum(ln_x**2)

ln_y =  np.log(y)
sum_ln_y =  np.sum(ln_y)
sum_xln_y =  np.sum(ln_y*ln_x)
```

Mendefinisikan matriks A dan vektor y

In [18]:
```
A = np.array([[m,sum_ln_x],[sum_ln_x,sum_ln_x2]])
print(A)
b = np.array([sum_ln_y,sum_xln_y])
print(b)
```

```
[[ 161.          750.81426538]
 [ 750.81426538 3674.93612788]]
[ 413.62402289 1989.58585865]
```

mencari koefisien

In [19]:
```
c = np.linalg.inv(A)@b
print(c)
```

```
[0.93879482 0.3495912 ]
```

mendefinisikan fungsi

In [20]:
```
def Non_reg(a1,a0,x):
  return np.exp(a0)*x**a1
```

Langkah 5. Mengevaluasi model

Menentukan tabel baru yang berisi data latih/testing

In [21]:
```
m = len(test.TV)
x = np.asanyarray(test[['TV']])
```

```
y = np.asanyarray(test[['sales']])

ln_x =  np.log(x)
ln_y =  np.log(y)
```

selanjutnya tentukan $\hat{y} = be^{ax}$

In [22]:
```
yhat =  np.exp(c[0])*x**c[1]
```

Atau dalam $\ln \hat{y} = \ln b + a \ln x$

In [23]:
```
ln_yhat = c[0]+c[1]*np.log(x)
```

Menentukan Tabel ANOVA

Dalam bentuk

$\ln y = \ln b + a \ln x$

maka bentuk termasuk linier, sehingga dapat kita analisis kelinierannya dengan ANOVA yaitu

Hypotheis Testing

$$\begin{cases} H_0 : a = 0 \\ H_1 : a \neq 0 \end{cases}$$

In [24]:
```
import numpy as np
import pandas as pd
import scipy
from scipy import stats
def ANOVATAB(y,yhat,n,m):
  dfn = n
  dfd = m-n-1
  ybar = np.average(y)

  SSR = np.sum((yhat - ybar)**2)
  SSE = np.sum((y-yhat)**2)
  SST = np.sum((y-ybar)**2)
  MSR = SSR/dfn
  MSE = SSE/dfd

  Fs = MSR/MSE
  ks = 1-scipy.stats.f.cdf(Fs, dfn, dfd)
  data_table= {
    'SS': [SSR, SSE, SST],
    'df': [dfn, dfd,m-1] ,
    'MS': [MSR, MSE,'-'],
    'Fs': [Fs, '-','-'],
    'pval': [ks, '-','-']
  }

  return pd.DataFrame(data_table)
```

In [25]:
```
n= 1
tabel = ANOVATAB(ln_y,ln_yhat,n,m)
tabel
```

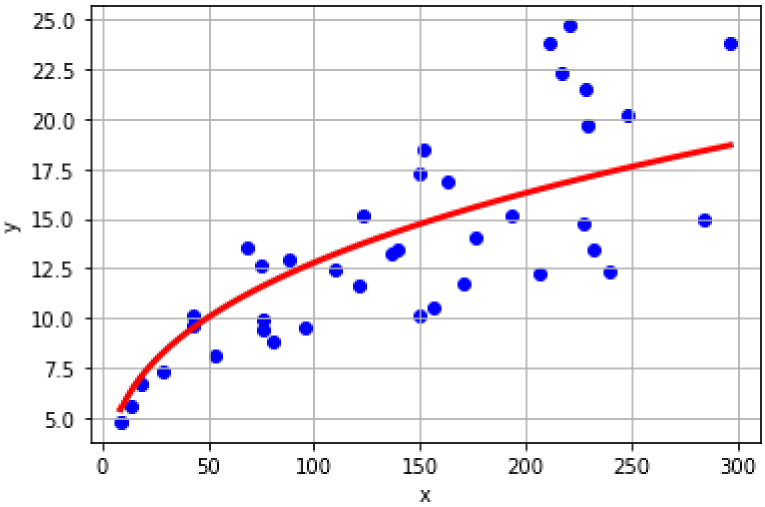
Out[25]:

	SS	df	MS	Fs	pval
0	3.432780	1	3.43278	73.359974	0.0
1	1.731364	37	0.046794	-	-
2	5.835207	38	-	-	-

Plot data testing dan fungsi hampiran

In [26]:

```
import matplotlib.pyplot as plt #library untuk plot
xp = np.linspace(min(x), max(x), 100)
yp = Non_reg(c[1],c[0],xp)

plt.plot(xp,yp, color = 'red', linewidth=3)
plt.scatter(x,y,  color='blue')
plt.xlabel("x")
plt.ylabel("y")
#plt.legend(('$\hat{y}(x)=b x^a$',), loc='upper left')
plt.grid()
plt.show()
```



Homework

1. Given the following data:

i	1	2	3	4	5	6	7	8	9	10
x	4.0	4.2	4.5	4.7	5.1	5.5	5.9	6.3	6.8	7.1
y	102.56	113.18	130.11	142.05	167.53	195.14	224.87	256.73	299.50	326.72

Construct the least squares polynomial of degree 1, and compute the error.

1. Construct the least squares polynomial of degree 2, and compute the error.
2. Construct the least squares polynomial of degree 3, and compute the error.
3. Construct the least squares approximation of the form be^{ax}, and compute the error.
4. Construct the least squares approximation of the form bx^a, and compute the error.

1. Diberikan data seperti berikut ini

W	R	W	R	W	R	W	R	W	R
0.017	0.154	0.025	0.23	0.020	0.181	0.020	0.180	0.025	0.234
0.087	0.296	0.111	0.357	0.085	0.260	0.119	0.299	0.233	0.537
0.174	0.363	0.211	0.366	0.171	0.334	0.210	0.428	0.783	1.47
1.11	0.531	0.999	0.771	1.29	0.87	1.32	1.15	1.35	2.48
1.74	2.23	3.02	2.01	3.04	3.59	3.34	2.83	1.69	1.44
4.09	3.58	4.28	3.28	4.29	3.40	5.48	4.15	2.75	1.84
5.45	3.52	4.58	2.96	5.30	3.88			4.83	4.66
5.96	2.40	4.68	5.10					5.53	6.94

tentukanlah:

a.) Tentukanlah model regrsi linar dalam bentuk $R = bW^a$ menggunakan model hampiran logaritma

$\ln R = \ln b + a \ln W$

b) Tentukan tabel ANOVA untuk hampiran bentuk model logarithm.

c). Bandingkan MSE dalam bentuk $R = bW^a$ dan $\ln R = \ln b + a \ln W$

c) Jika ditambahkan suku $(\ln W)^2$ pada model hampiran logaritma soal a), maka tentukan bentuk model hampiran logaritma polinomial orde 2.

1. Given the follwing data

   http://bit.ly/Test-PHN3

   Use Machine Learning algorithm to:

   A. Construct the least squares approximation of the form be^{ax}, and compute the error.
   B. Construct the least squares approximation of the form bx^a, and compute

Soal 1

1. Given the following data:

i	1	2	3	4	5	6	7	8	9	10
x	4.0	4.2	4.5	4.7	5.1	5.5	5.9	6.3	6.8	7.1
y	102.56	113.18	130.11	142.05	167.53	195.14	224.87	256.73	299.50	326.72

Construct the least squares polynomial of degree 1, and compute the error.

1. Construct the least squares polynomial of degree 2, and compute the error.
2. Construct the least squares polynomial of degree 3, and compute the error.
3. Construct the least squares approximation of the form be^{ax}, and compute the error.
4. Construct the least squares approximation of the form bx^a, and compute the error.

In [27]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [28]:

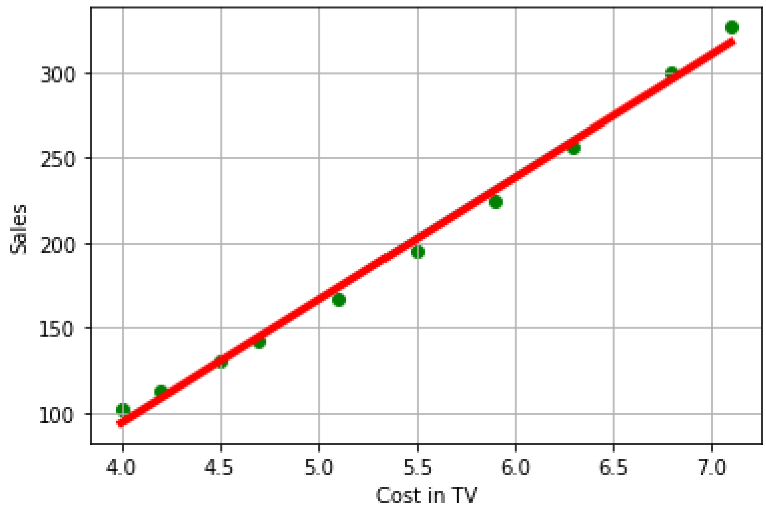
```
x = np.array([4, 4.2, 4.5, 4.7, 5.1, 5.5, 5.9, 6.3, 6.8, 7.1])
y= np.array([102.56, 113.18, 130.11, 142.05, 167.53, 195.14, 224.87, 256.73, 299.50,
```

Least square polynomial 1 degree

In [29]:

```
a11 =  len(x)
a12 =  sum(x)
a21 =  sum(x)
a22 =  sum(x**2)

b1 =  sum(y)
b2 =  sum(y*x)

A = np.array([[a11, a12], [a21, a22]])
print('Matrix A:')
print(A)
b = np.array([ b1, b2])
print('Vector b')
print(b)

# c = inv(A) b
c = np.linalg.inv(A)@b
print('Nilai coefficient:', c)

#Plot fungsi hampiran
xp = np.linspace(min(x), max(x), 100)
yp = c[1]*xp + c[0]

plt.plot(xp,yp, color = 'red', linewidth=4)
plt.scatter(x,y,  color='green')
plt.xlabel("Cost in TV")
plt.ylabel("Sales")
plt.grid()
plt.show()

#Mencari nilai error
yhat = c[1]*x +c[0]

MSE = sum(y-yhat)**2/(len(x)-1-1)
print('MSE :', MSE)
```

```
Matrix A:
[[ 10.    54.1 ]
 [ 54.1  303.39]]
Vector b
[ 1958.39  11366.843]
Nilai coefficient: [-194.13824073   72.0845177 ]
```

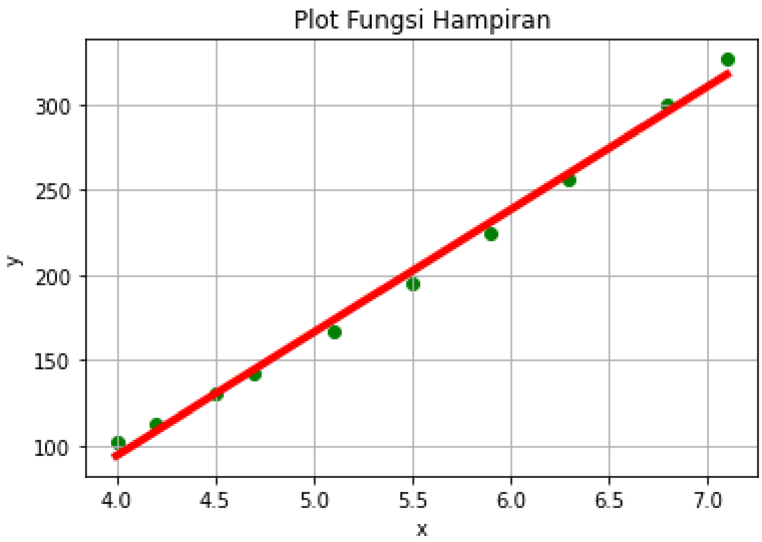



```
MSE : 3.4558418535922085e-24
```

Least square polynomial 2 degree

In [30]:
```
#Model regresi polinomial
a11 = len(x)
a12 =  sum(x)
a13 =  sum(x**2)

a21 =  sum(x)
a22 =  sum(x**2)
a23 =  sum(x**3)

a31 = sum(x**2)
a32 = sum(x**3)
a33 = sum(x**4)

b1 =  sum(y)
b2 =  sum(y*x)
b3 =  sum(y*x**2)

A = np.array([[a11, a12, a13], [a21, a22, a23], [a31, a32, a33]])
b = np.array([ b1, b2, b3])
c = np.linalg.inv(A)@b
print('Koefisien model:', c)

#Plot fungsi hampiran
plt.scatter(x, y, color='green')
plt.plot(xp, yp, color = 'red', linewidth=4)
plt.xlabel("x")
plt.ylabel("y")
plt.title("Plot Fungsi Hampiran")
plt.grid()
plt.show()

#Evaluasi model
yhat =c[2]*x**2 + c[1]*x + c[0]
MSE = sum(y-yhat)**2/(len(x)-1-1)
print("MSE = ", MSE)
```

```
Koefisien model: [ 1.23556037 -1.14352337  6.61821092]
```




```
MSE =  1.5078679548890366e-19
```

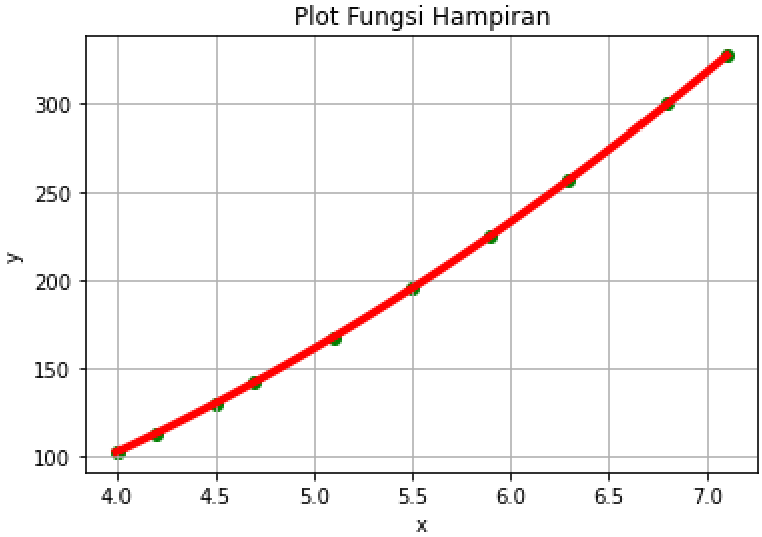
Least square polynomial 3 degree

In [31]:

```python
#Membangun model regresi polinom derajat 3
a11 = len(x)
a12 = sum(x)
a13 = sum(x**2)
a14 = sum(x**3)

a21 =  sum(x)
a22 =  sum(x**2)
a23 =  sum(x**3)
a24 = sum(x**4)

a31 =  sum(x**2)
a32 =  sum(x**3)
a33 =  sum(x**4)
a34 = sum(x**5)

a41 =  sum(x**3)
a42 =  sum(x**4)
a43 =  sum(x**5)
a44 = sum(x**6)

b1 =  sum(y)
b2 =  sum(y*x)
b3 =  sum(y*x*x)
b4 = sum(y*x*x*x)

A = np.array([[a11, a12, a13, a14], [a21, a22, a23, a24], [a31, a32, a33, a34],
              [a41, a42, a43, a44]])
b = np.array([b1, b2, b3, b4])
c = np.linalg.inv(A)@b
print('Coefficient:')
print(c)

xp= np.linspace(min(x),max(x),100)
yp = c[3]*xp**3 + c[2]*xp**2 + c[1]*xp + c[0]

#Plot fungsi hampiran
import matplotlib.pyplot as plt
plt.scatter(x, y, color='green')
plt.plot(xp, yp, color = 'red', linewidth=4)
plt.xlabel("x")
plt.ylabel("y")
```

```
plt.title("Plot Fungsi Hampiran")
plt.grid()
plt.show()

#Evaluasi model
n = 3
m = len(x)

yhat = c[3]*(x**3) + c[2]*(x**2) + c[1]*x +c[0]
ybar = np.average(y)
MSE = sum((y-yhat)**2)/(m-n-1)
print('MSE = ', MSE)
```

```
Coefficient:
[ 3.4290944  -2.37922111  6.84557777 -0.01367456]
```




```
MSE =  8.789020050961804e-05
```

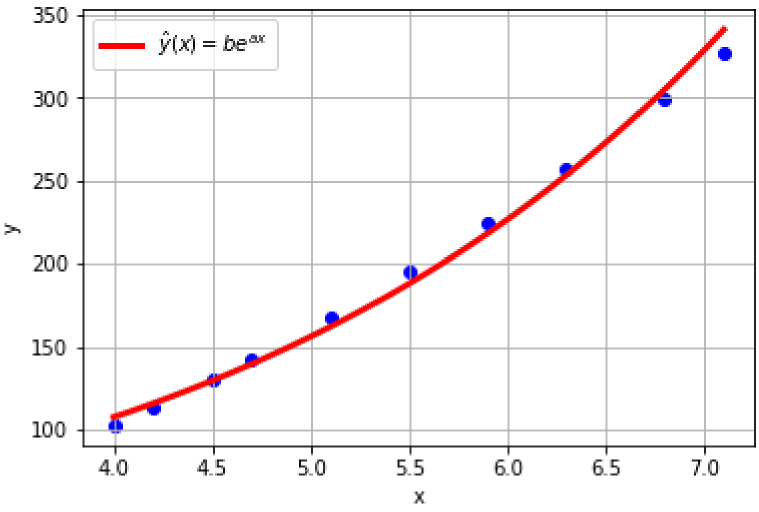
Aproksimasi least square dalam bentuk be^{ax}

In [32]:

```
m = len(y)
sum_x =  np.sum(x)
sum_x2 =  np.sum(x**2)

ln_y =  np.log(y)
sum_ln_y =  np.sum(ln_y)
sum_xln_y =  np.sum(x*ln_y)

#Menentukan matriks A dan vektor y
A = np.array([[m, sum_x],[sum_x, sum_x2]])
print(A)
b = np.array([sum_ln_y, sum_xln_y])
print(b)

#Mencari nilai c
c = np.linalg.inv(A)@b
print('Coefficients:', c)

def Non_reg(a1,a0,x):
  return np.exp(a0)*np.exp(a1*x)

#Membuat plot
import matplotlib.pyplot as plt
xp = np.linspace(min(x), max(x), 100)
yp = Non_reg(c[1],c[0],xp)

plt.plot(xp,yp, color = 'red', linewidth=3)
```

```
plt.scatter(x,y,  color='blue')
plt.xlabel("x")
plt.ylabel("y")
plt.legend(('$\hat{y}(x)=b e^{ax}$',), loc='upper left')
plt.grid()
plt.show()
```

```
[[ 10.     54.1 ]
 [ 54.1  303.39]]
[ 52.03363187 285.4897848 ]
Coefficients: [3.1887778  0.37238177]
```



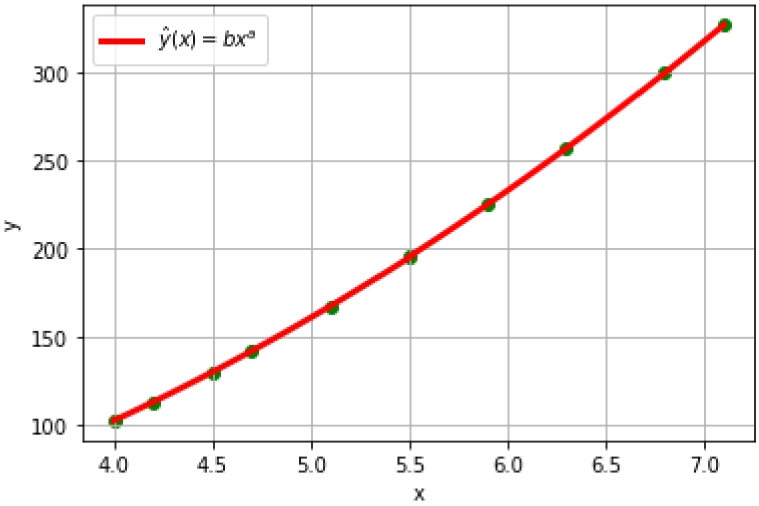
Aproksimasi least square dalam bentuk bx^a

In [33]:

```
m =  len(x)

ln_x = np.log(x)
sum_ln_x =  np.sum(ln_x)
sum_ln_x2 = np.sum(ln_x**2)

ln_y =  np.log(y)
sum_ln_y =  np.sum(ln_y)
sum_xln_y =  np.sum(ln_x*ln_y)

#Menentukan matriks A dan vektor y
A = np.array([[m,sum_ln_x],[sum_ln_x,sum_ln_x2]])
print('A')
print(A)
b = np.array([sum_ln_y,sum_xln_y])
print('y')
print(b)

#Mencari nilai koefisien
c = np.linalg.inv(A)@b
print('Coefficients:', c)

def Non_reg(a1,a0,x):
  return np.exp(a0)*x**a1

#Membuat plot
import matplotlib.pyplot as plt
xp = np.linspace(min(x), max(x), 100)
yp = Non_reg(c[1],c[0],xp)

plt.plot(xp,yp, color = 'red', linewidth=3)
plt.scatter(x,y,  color='green')
plt.xlabel("x")
```

```
plt.ylabel("y")
plt.legend(('$\hat{y}(x)=b x^a$',), loc='upper left')
plt.grid()
plt.show()
```

```
A
[[10.         16.6995268 ]
 [16.6995268  28.25371164]]
y
[52.03363187 87.63344505]
Coefficients: [1.83082464 2.01954138]
```



Soal 2

1. Diberikan data seperti berikut ini

W	R	W	R	W	R	W	R	W	R
0.017	0.154	0.025	0.23	0.020	0.181	0.020	0.180	0.025	0.234
0.087	0.296	0.111	0.357	0.085	0.260	0.119	0.299	0.233	0.537
0.174	0.363	0.211	0.366	0.171	0.334	0.210	0.428	0.783	1.47
1.11	0.531	0.999	0.771	1.29	0.87	1.32	1.15	1.35	2.48
1.74	2.23	3.02	2.01	3.04	3.59	3.34	2.83	1.69	1.44
4.09	3.58	4.28	3.28	4.29	3.40	5.48	4.15	2.75	1.84
5.45	3.52	4.58	2.96	5.30	3.88			4.83	4.66
5.96	2.40	4.68	5.10					5.53	6.94

tentukanlah:

a.) Tentukanlah model regrsi linar dalam bentuk $R = bW^a$ menggunakan model hampiran logaritma

$\ln R = \ln b + a \ln W$

b) Tentukan tabel ANOVA untuk hampiran bentuk model logarithm.

c). Bandingkan MSE dalam bentuk $R = bW^a$ dan $\ln R = \ln b + a \ln W$

d) Jika ditambahkan suku $(\ln W)^2$ pada model hampiran logaritma soal a), maka tentukan bentuk model hampiran logaritma polinomial orde 2.

In [34]:

```
import scipy
from scipy import stats
def ANOVATAB(y,yhat,n,m):
    dfn = n
    dfd = m-n-1
    ybar = np.average(y)
    SSR = np.sum((yhat - ybar)**2)
    SSE = np.sum((y-yhat)**2)
    SST = np.sum((y-ybar)**2)
    MSR = SSR/dfn
    MSE = SSE/dfd
    Fs = MSR/MSE
    ks = 1-scipy.stats.f.cdf(Fs, dfn, dfd)
    data_table= {
     'SS': [SSR, SSE, SST],
     'df': [dfn, dfd,m-1] ,
     'MS': [MSR, MSE,'-'],
     'Fs': [Fs, '-','-'],
     'pval': [ks, '-','-']
     }
    return pd.DataFrame(data_table)
W=np.array([0.017,0.087,0.174,1.11,1.74,4.09,5.45,5.96,
 0.025,0.111,0.211,0.999,3.02,4.28,4.58,4.68,
 0.02,0.085,0.171,1.29,3.04,4.29,5.3,
 0.02,0.119,0.21,1.32,3.34,5.48,
 0.025,0.233,0.783,1.35,1.69,2.75,4.83,5.53])
R=np.array([0.154,0.296,0.363,0.531,2.23,3.58,3.52,2.4,
 0.23,0.357,0.366,0.771,2.01,3.28,2.96,5.1,
 0.181,0.26,0.334,0.87,3.59,3.4,3.88,
 0.18,0.299,0.428,1.15,2.83,4.15,
 0.234,0.537,1.47,2.48,1.44,1.84,4.66,6.94])
```

Model regresi linar dalam bentuk $R = bW^a$ menggunakan model hampiran logaritma

$\ln R = \ln b + a \ln W$

In [35]:

```
ln_W=np.log(W)

a11 = len(W)
a12 = np.sum(ln_W)
a21 = a12
a22 = np.sum(ln_W**2)

ln_R = np.log(R)

b1 = np.sum(ln_R)
b2 = np.sum(ln_R*ln_W)

X=np.array([[a11,a12],[a21,a22]])
b=np.array([b1,b2])
c=np.linalg.inv(X)@b

print("Persamaan: R={} W={}".format(np.exp(c[0]),c[1]))
```

```
Persamaan: R=1.3029717779462264 W=0.5756426027724945
```

Tabel ANOVA untuk hampiran bentuk model logarithm

In [36]:

```
ln_Rhat=c[0]+c[1]*np.log(W)
Rhat=np.exp(ln_Rhat)

#lnR dengan LnR hat
tabel1=ANOVATAB(ln_R,ln_Rhat,1,a11)
```

```
#R dengan R hat
tabel2=ANOVATAB(R,Rhat,1,a11)

print(tabel1)
print(tabel2)
```

```
          SS  df         MS          Fs pval
0  45.854526   1  45.854526  371.508737  0.0
1   4.319975  35   0.123428           -    -
2  50.174501  36          -           -    -
           SS  df         MS         Fs pval
0   58.410294   1  58.410294  80.798873  0.0
1   25.301842  35    0.72291          -    -
2  108.465862  36          -          -    -
```

Bandingkan MSE dalam bentuk $R = bW^a$ dan $\ln R = \ln b + a \ln W$

In [37]:

```
print("MSE ln_R: {}".format(tabel1["MS"][1]))
print("MSE R: {}".format(tabel2["MS"][1]))
```

```
MSE ln_R: 0.12342785338713294
MSE R: 0.7229097610083884
```

Jadi berdasarkan data tersebut, dapat kita simpulkan bahwa MSE R > MSE ln_R

Bentuk model hampiran logaritma polinomial orde 2 jika ditambahkan suku $(\ln W)^2$ pada model hampiran logaritma soal a

In [38]:

```
a11 = len(W)
a12 = np.sum(ln_W)
a13 = np.sum(ln_W**2)

a21 = a12
a22 = a13
a23 = np.sum(ln_W**3)

a31 = a13
a32 = a23
a33 = np.sum(ln_W**4)

ln_R = np.log(R)

b1 = np.sum(ln_R)
b2 = np.sum(ln_R*ln_W)
b3 = np.sum(ln_R*ln_W**2)

X=np.array([[a11,a12,a13],[a21,a22,a23],[a31,a32,a33]])

b=np.array([b1,b2,b3])
c=np.linalg.inv(X)@b

print("persamaanya: lnR={}+{}lnW+{}(lnW)^2".format(c[0],c[1],c[2]))
```

```
persamaanya: lnR=0.049620213532374446+0.7006291876067215lnW+0.06695491979365836(lnW)
^2
```

Soal 3

1. Given the follwing data

http://bit.ly/Test-PHN3

Use Machine Learning algorithm to:

(a). Construct the least squares approximation of the form be^{ax}, and compute the error.

(b). Construct the least squares approximation of the form bx^{a}, and compute

In [39]:

```
url = 'http://bit.ly/Test-PHN3'
data = pd.read_csv(url, index_col=0)
data["ydata"]=abs(data["ydata"])

msk = np.random.rand(len(data)) < 0.8
train = data[msk]
test = data[~msk]

m = len(train.x) #number of rows data
x = np.asanyarray(train[['x']])
y = np.asanyarray(train[['ydata']])

a11 = len(x)
a12 = np.sum(x)
a21 = a12
a22 = np.sum(x**2)

ln_y = np.log(y)

b1 = np.sum(ln_y)
b2 = np.sum(ln_y*x)

X=np.array([[a11,a12],[a21,a22]])
b=np.array([b1,b2])
c=np.linalg.inv(X)@b
print("persamaanya: lny={}+{}x".format(c[0],c[1]))

m = len(test.x)
x = np.asanyarray(test[['x']])
y = np.asanyarray(test[['ydata']])
ln_y = np.log(y)
ln_yhat = c[0]+c[1]*x

n= 1
tabel = ANOVATAB(ln_y,ln_yhat,n,m)
print(tabel)
```

```
persamaanya: lny=6.640667402785553+0.17259129809203522x
          SS  df        MS        Fs      pval
0   3.876814   1  3.876814  6.202985  0.022176
1  11.874842  19  0.624992         -         -
2  14.021076  20         -         -         -
```

In [40]:

```
url = 'http://bit.ly/Test-PHN3'
data = pd.read_csv(url, index_col=0)
data["ydata"]=abs(data["ydata"])
data["x"]=data["x"]+0.1

msk = np.random.rand(len(data)) < 0.8
train = data[msk]
test = data[~msk]

m = len(train.x) #number of rows data
```

```python
x = np.asanyarray(train[['x']])
y = np.asanyarray(train[['ydata']])
ln_x= np.log(x)

a11 = len(x)
a12 = np.sum(ln_x)
a21 = a12
a22 = np.sum(ln_x**2)

ln_y = np.log(y)

b1 = np.sum(ln_y)
b2 = np.sum(ln_y*ln_x)

X=np.array([[a11,a12],[a21,a22]])
b=np.array([b1,b2])
c=np.linalg.inv(X)@b
print("persamaanya: lny={}+{}lnx".format(c[0],c[1]))

m = len(test.x)
x = np.asanyarray(test[['x']])
y = np.asanyarray(test[['ydata']])

ln_x=np.log(x)
ln_y = np.log(y)
ln_yhat = c[0]+c[1]*ln_x

n= 1
tabel = ANOVATAB(ln_y,ln_yhat,n,m)
print(tabel)
```

```
persamaanya: lny=7.055961121113185+0.34791435452262576lnx
          SS  df        MS        Fs     pval
0   3.889905   1  3.889905  3.879238  0.06222
1  21.057745  21   1.00275         -        -
2  21.482391  22         -         -        -
```