EndSem_2

August 3, 2021

0.1 First we are making a scatter plot of the function

$g(x) = A_1 + A_2 \cdot e^{-A_3 \cdot x}$

[ ]:

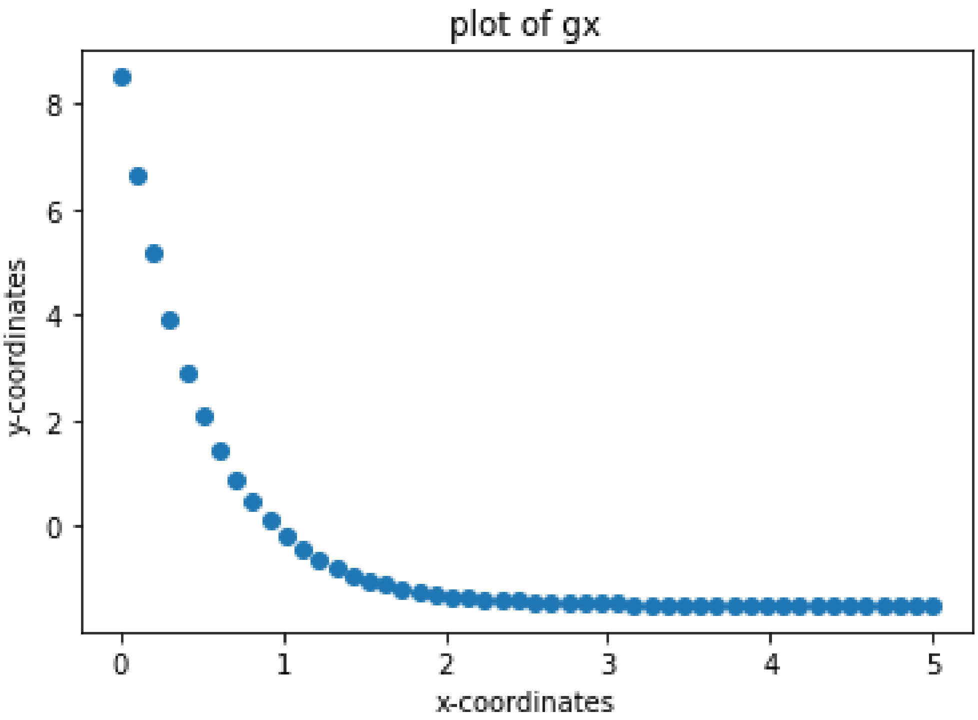
```
import math
import numpy as np
A1 = -1.5
A2 = 10
A3 = 2
x = np.linspace(0,5,50)
def myf(b):
  return A1 + (A2*math.exp(-A3*b))
gx = list(map(myf, x))
```

[ ]:

```
import matplotlib.pyplot as plt
plt.title("plot of gx")
plt.xlabel("x-coordinates")
plt.ylabel("y-coordinates")
plt.scatter(x,gx)
```

[ ]:

```
<matplotlib.collections.PathCollection at 0x7f5d48b175d0>
```



[ ]:

```
mu = 1
sd = 0.5
noiselist = np.random.normal(mu,sd,50)
def noisef(x, k):
  Amp = 0.01*k*max(gx)
  np.random.seed(0)
  noise = Amp*noiselist
  return gx + noise
g1x = noisef(x, 5)
g2x = noisef(x, 7)
g3x = noisef(x, 10)
```

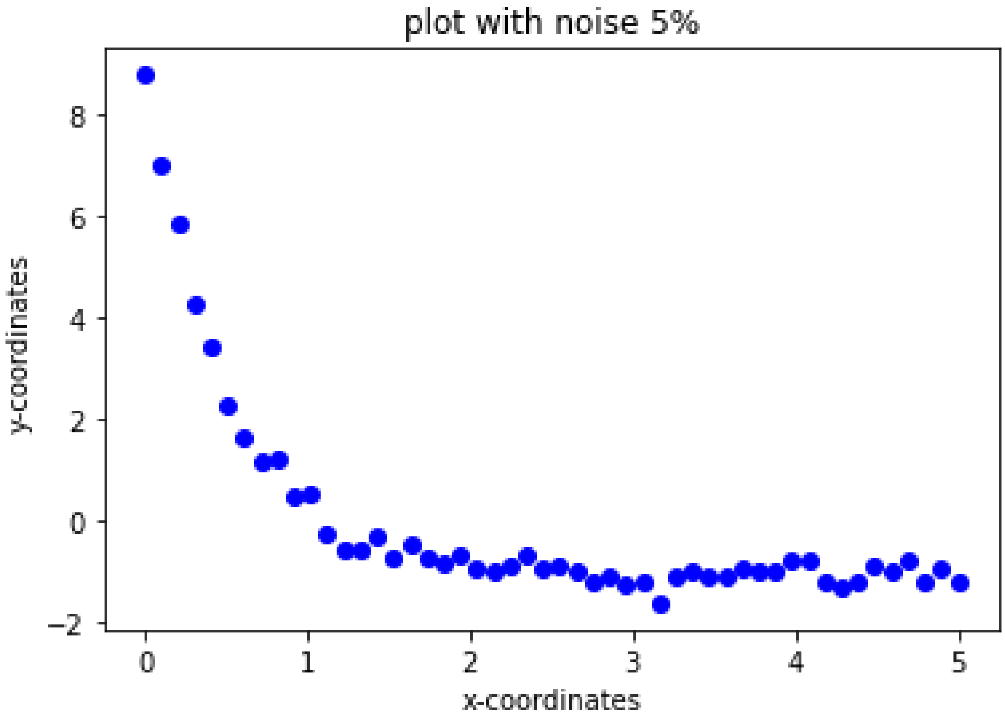
Plotting the Noise data obtaiined with 5%, 7%, 10% Noise Amplitudes

[ ]:

```
plt.title("plot with noise 5%")
plt.xlabel("x-coordinates")
plt.ylabel("y-coordinates")
plt.scatter(x, g1x, color='blue',label='noise = 5%')
```

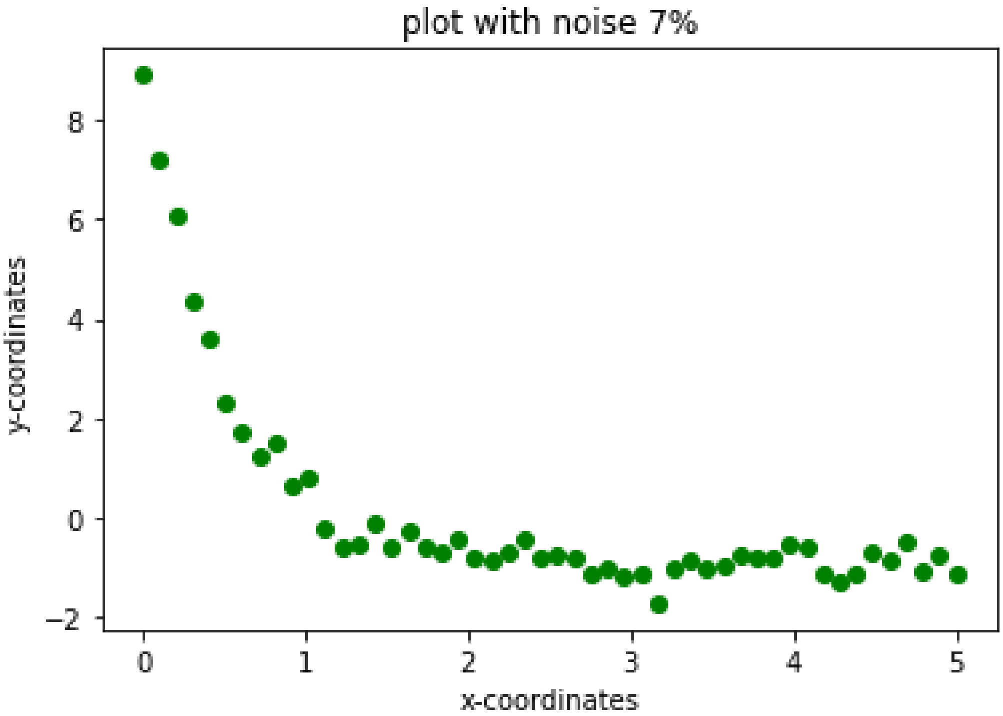
[ ]:

```
<matplotlib.collections.PathCollection at 0x7f5d48b174d0>
```



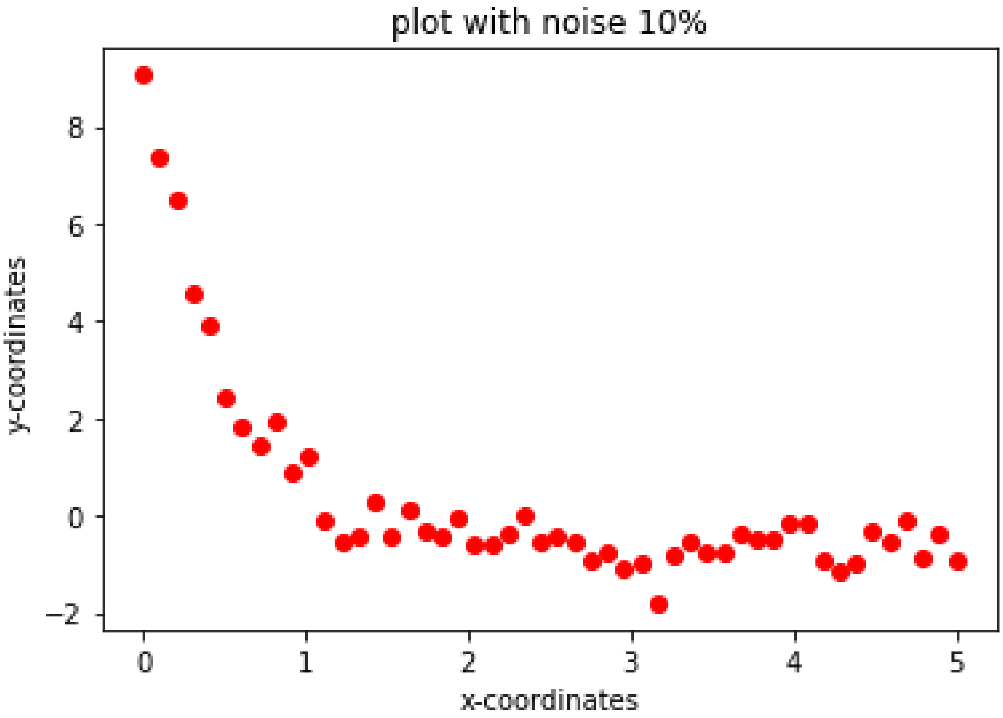
```
[ ]: plt.title("plot with noise 7%")
     plt.xlabel("x-coordinates")
     plt.ylabel("y-coordinates")
     plt.scatter(x, g2x, color='green')
```

```
[ ]: <matplotlib.collections.PathCollection at 0x7f5d46b8a110>
```



```
[ ]: plt.title("plot with noise 10%")
     plt.xlabel("x-coordinates")
     plt.ylabel("y-coordinates")
     plt.scatter(x, g3x, color='red')
```

```
[ ]: <matplotlib.collections.PathCollection at 0x7f5d46b14290>
```



0.2 Curve-Fitting

0.2.1 Now we will fit the curve using scipy

[ ]:
```
from scipy import optimize
def myfunc(x, B1, B2, B3):
  return B1 + B2*np.exp(-B3*x)
```

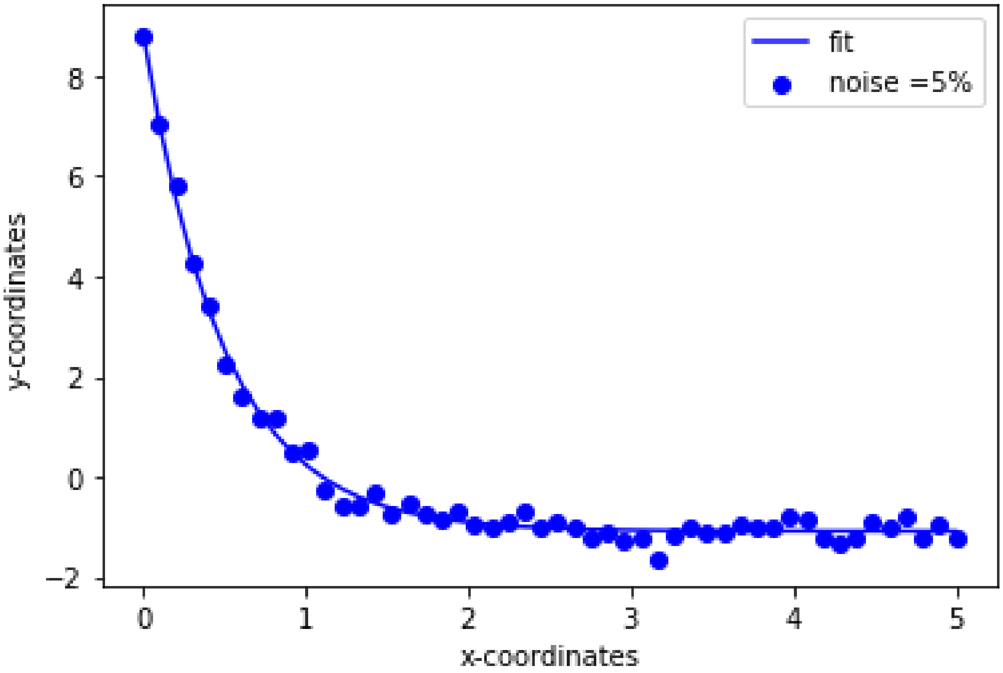
[ ]:
```
print("The difference in values of A1,B1 and A2,B2 and A3,B3 with 5%,7% and 10%␣
 ↪noise")
params1, params1_covariance = optimize.curve_fit(myfunc, x, g1x, p0=[2,2,2])
print(A1-params1[0], A2-params1[1], A3-params1[2])
params2, params2_covariance = optimize.curve_fit(myfunc, x, g2x, p0=[2,2,2])
print(A1-params2[0], A2-params2[1], A3-params2[2])
params3, params3_covariance = optimize.curve_fit(myfunc, x, g3x, p0=[2,2,2])
print(A1-params3[0], A2-params3[1], A3-params3[2])
print("It can be here-by observed that as the noise is increasing the magnitude␣
 ↪of the difference is increasing")
```

```
The difference in values of A1,B1 and A2,B2 and A3,B3 with 5%,7% and 10% noise
-0.4156088233480699 0.033053407800498036 -0.012388395815381426
-0.5818315991438257 0.0462858239317665 -0.017364239590608843
-0.8311433037759148 0.0661479777195293 -0.02484894439703922
It can be here-by observed that as the noise is increasing the magnitude of the
```

```
difference is increasing
```

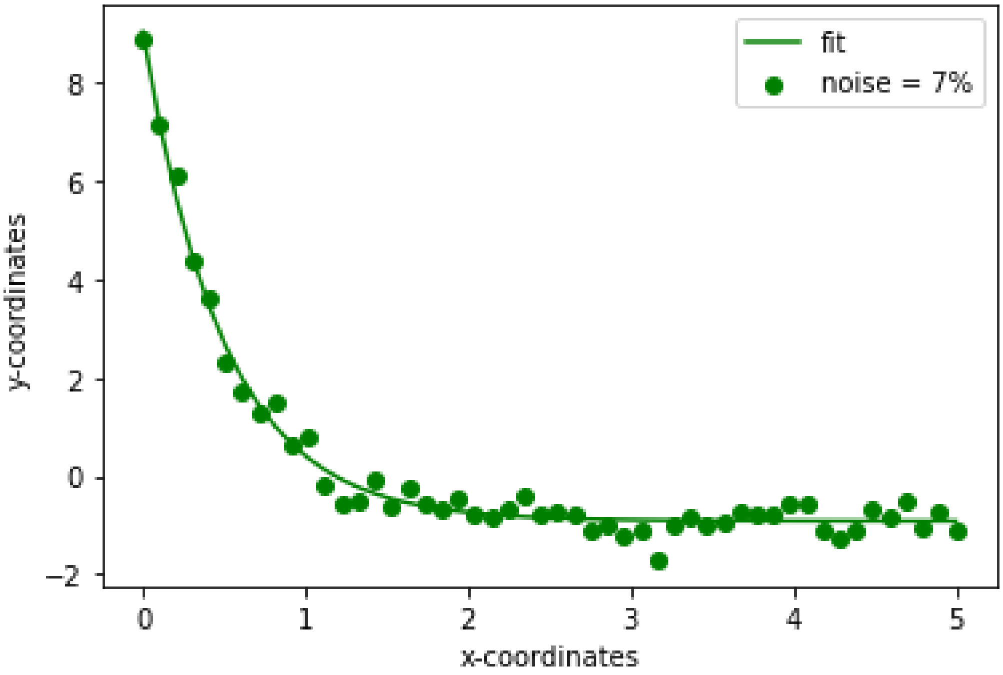
[ ]:

```
plt.scatter(x, g1x, label = 'noise =5%',color='blue')
plt.plot(x,myfunc(x, params1[0], params1[1], params1[2]), label =␣
 ↪'fit',color='blue')
plt.xlabel("x-coordinates")
plt.ylabel("y-coordinates")
plt.legend(loc = 'best')
plt.show()
```

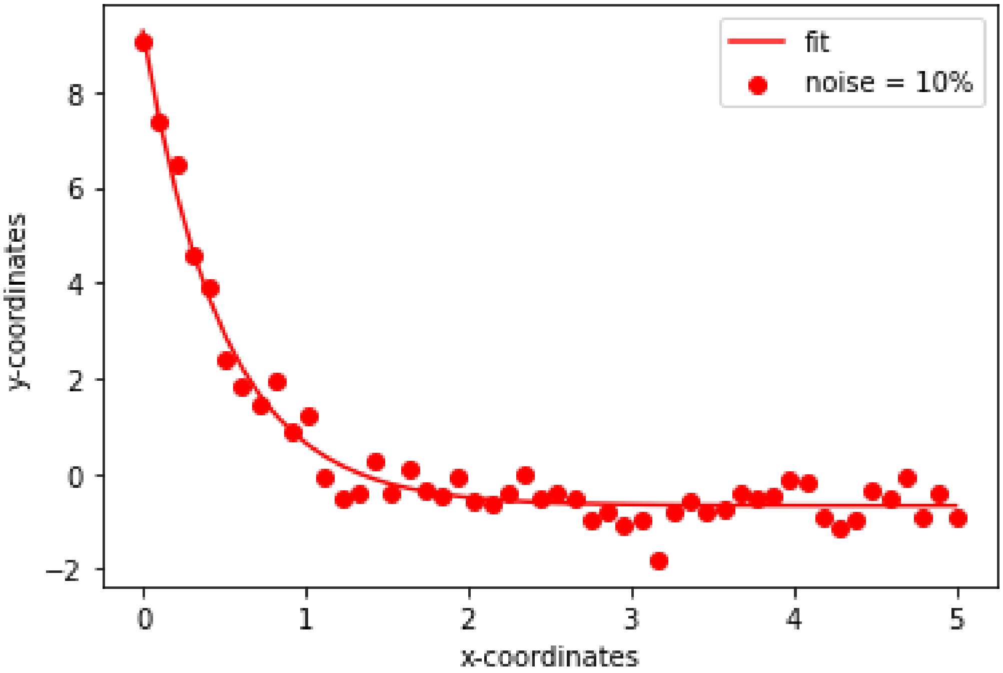


[ ]:

```
plt.scatter(x, g2x, label = 'noise = 7%',color='green')
plt.plot(x,myfunc(x, params2[0], params2[1], params2[2]), label = 'fit',color=␣
 ↪'green')
plt.xlabel("x-coordinates")
plt.ylabel("y-coordinates")
plt.legend(loc = 'best')
plt.show()
```



```
[ ]: plt.scatter(x, g3x, label = 'noise = 10%',color='red')
     plt.plot(x,myfunc(x, params3[0], params3[1], params3[2]), label = 'fit', color=␣
      ↪'red')
     plt.xlabel("x-coordinates")
     plt.ylabel("y-coordinates")
     plt.legend(loc = 'best')
     plt.show()
```


fit
noise = 10%
y-coordinates
x-coordinates
8
6
4
2
0
−2
0
1
2
3
4
5