625.661 - Homework Seven

Eric Niblock

April 26, 2022

1. **The data included represents the fraction of active chlorine in a chemical product as a function of time after manufacturing.**

   (a) **Construct a scatterplot of the data.**

   Results for this problem are provided in the attached PDF. The scatter plot was constructed and includes both the data as well as the fitted regression.

   (b) **Fit the Mitcherlich law (see Problem 12.10) to these data. Discuss how you obtained the starting values.**

   Results for this problem are provided in the attached PDF. The coefficients were determined to have the following values: $\theta_1 = 0.3829$, $\theta_2 = -0.1986$, $\theta_3 = 0.0802$. Thus, the model can be represented by the following,

$$y = 0.3829 + 0.1986e^{-0.0802x} \tag{1}$$

   The starting values for the fitting function were obtained by analyzing the scatter plot and expectation function.

   (c) **Test for significance of regression.**

   Results for this problem are provided in the attached PDF. An F-value of 120.635 was calculated, which has a corresponding p-value of approximately zero. Therefore, the null hypothesis can be rejected, and we note that the regression is significant.

(d) Find approximate 95% confidence intervals on the parameters θ_1, θ_2, and θ_3. Is there evidence to support the claim that all three parameters are different from zero?

Results for this problem are provided in the attached PDF. Observe the 95% confidence intervals constructed for each of the parameters, θ_1, θ_2, and θ_3. There is evidence to support that all three parameters differ from zero, since none of the confidence intervals contain zero.

(e) Analyze the residuals and comment on model adequacy.

Results for this problem are provided in the attached PDF. A normal probability plot of the residuals was constructed. Though the residuals tend to oscillate around the normal line, there does appear to be a distinct pattern around this oscillation. This calls the normality assumption into question, and the residuals may not be entirely normal. Furthermore, the residuals were plotted as a function of y, and the magnitude of the residuals does appear to be correlated with the value of y. Therefore, the data does display some heteroscedasticity.

Data Cleaning and Selection

In [141]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [142]:
```
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
```

In [143]:
```
data = [[0.49,  0.49,    8]    ,
[0.48,  0.47,  0.48,  0.47,    10    ],
[0.46,  0.46,  0.45,  0.43,    12    ],
[0.45,  0.43,  0.43,    14    ],
[0.44,  0.43,  0.43,    16 ]  ,
[0.46,  0.45,    18    ],
[0.42,  0.42,  0.43,    20    ],
[0.41,  0.41,  0.40 ,   22   ] ,
[0.42,  0.40,  0.40,    24  ] ,
[0.41,  0.40,  0.41,    26 ] ,
[0.41,  0.40,    28    ],
[0.40,  0.40,  0.38,    30 ]   ,
[0.41,  0.40,    32  ]  ,
[0.40,    34    ],
[0.41,  0.38,    36  ]  ,
[0.40,  0.40,    38 ] ,
[0.39,    40  ]  ,
[0.39,    42]]

choices = np.random.choice(len(data),14,replace=False)
print('Sample rows from provided table:')
sample = [data[e] for e in choices]
sample
```

```
Sample rows from provided table:
```

Out[143]:
```
[[0.41, 0.38, 36],
 [0.41, 0.4, 28],
 [0.4, 0.4, 0.38, 30],
 [0.48, 0.47, 0.48, 0.47, 10],
 [0.39, 40],
 [0.49, 0.49, 8],
 [0.42, 0.42, 0.43, 20],
 [0.41, 0.4, 32],
 [0.41, 0.4, 0.41, 26],
 [0.46, 0.45, 18],
 [0.4, 34],
 [0.44, 0.43, 0.43, 16],
 [0.42, 0.4, 0.4, 24],
 [0.46, 0.46, 0.45, 0.43, 12]]
```

In [144]:
```
fulldata = []
for l in sample:
    for i in l[:-1]:
        fulldata.append([i,l[-1]])
fulldata = np.array(fulldata)
print('Data organized into points:')
fulldata
```

```
Data organized into points:
```

Out[144]:
```
array([[ 0.41, 36.  ],
       [ 0.38, 36.  ],
       [ 0.41, 28.  ],
       [ 0.4 , 28.  ],
       [ 0.4 , 30.  ],
       [ 0.4 , 30.  ],
       [ 0.38, 30.  ],
       [ 0.48, 10.  ],
       [ 0.47, 10.  ],
       [ 0.48, 10.  ],
       [ 0.47, 10.  ],
       [ 0.39, 40.  ],
       [ 0.49,  8.  ],
       [ 0.49,  8.  ],
       [ 0.42, 20.  ],
       [ 0.42, 20.  ],
       [ 0.43, 20.  ],
       [ 0.41, 32.  ],
       [ 0.4 , 32.  ],
       [ 0.41, 26.  ],
       [ 0.4 , 26.  ],
       [ 0.41, 26.  ],
       [ 0.46, 18.  ],
       [ 0.45, 18.  ],
       [ 0.4 , 34.  ],
       [ 0.44, 16.  ],
       [ 0.43, 16.  ],
       [ 0.43, 16.  ],
       [ 0.42, 24.  ],
       [ 0.4 , 24.  ],
       [ 0.4 , 24.  ],
       [ 0.46, 12.  ],
       [ 0.46, 12.  ],
       [ 0.45, 12.  ],
       [ 0.43, 12.  ]])
```

Parts (a) (b)

In [145]:
```
def func(x, b1, b2, b3):
    return b1 - b2*np.exp(-1*b3*x)
```

In [146]:
```
popt, pcov = curve_fit(func, fulldata[:,1], fulldata[:,0], p0=[0,-0.1,0])
```

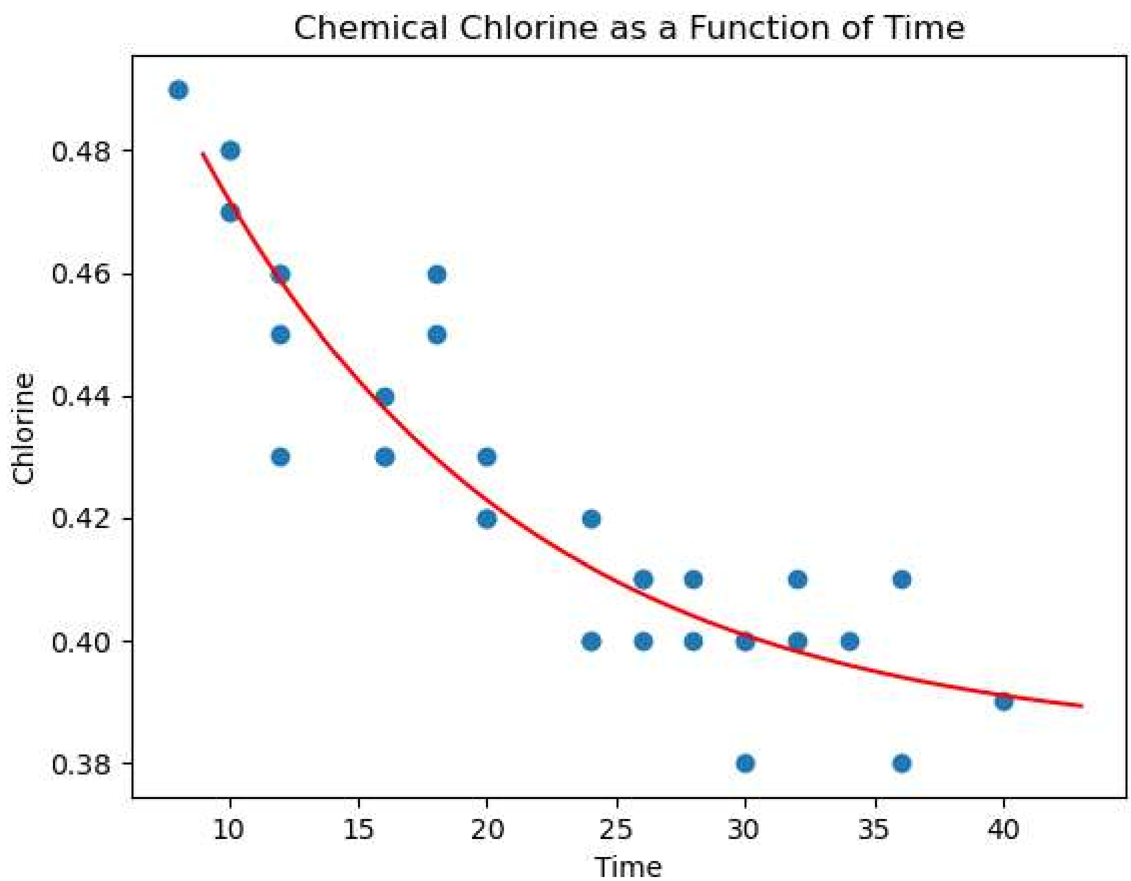
In [147]:
```
print('Fitted coefficients: ')
popt
```

```
Fitted coefficients:
```

Out[147]:
```
array([ 0.38295219, -0.19862392,  0.08025721])
```

In [148]:
```
xs = np.linspace(9,43,100)
ys = [popt[0] - popt[1]*np.exp(-1*popt[2]*x) for x in xs]
```

In [149]:
```
plt.scatter(fulldata[:,1], fulldata[:,0])
plt.plot(xs,ys, c='r')
plt.title('Chemical Chlorine as a Function of Time')
plt.xlabel('Time')
plt.ylabel('Chlorine')
plt.show()
```



Part (c)

In [157]:
```
ypred = [popt[0] - popt[1]*np.exp(-1*popt[2]*x) for x in fulldata[:,1]]
```

In [158]:
```
ss_tot = sum((fulldata[:,0] - np.mean(fulldata[:,0]))**2)
ss_res = sum((fulldata[:,0] - ypred)**2)
ms_res = ss_res/(len(fulldata)-3)
```

In [163]:
```
F = (ss_tot-ss_res)/(2*ms_res)
print('F-statistic:    ', F)
```

```
F-statistic:     120.63526928616666
```

Part (d)

In [170]:
```
print('Confidence Interval for B_1:   ')
print(popt[0]-(2.037*pcov[0,0]**0.5), ',', popt[0]+(2.037*pcov[0,0]**0.5))
```

```
Confidence Interval for B_1:
0.3618467068905411 , 0.4040576757902321
```

In [171]:
```
print('Confidence Interval for B_2:   ')
print(popt[1]-(2.037*pcov[1,1]**0.5), ',', popt[1]+(2.037*pcov[1,1]**0.5))
```

```
Confidence Interval for B_2:
-0.24806890917051916 , -0.149178930091885
```

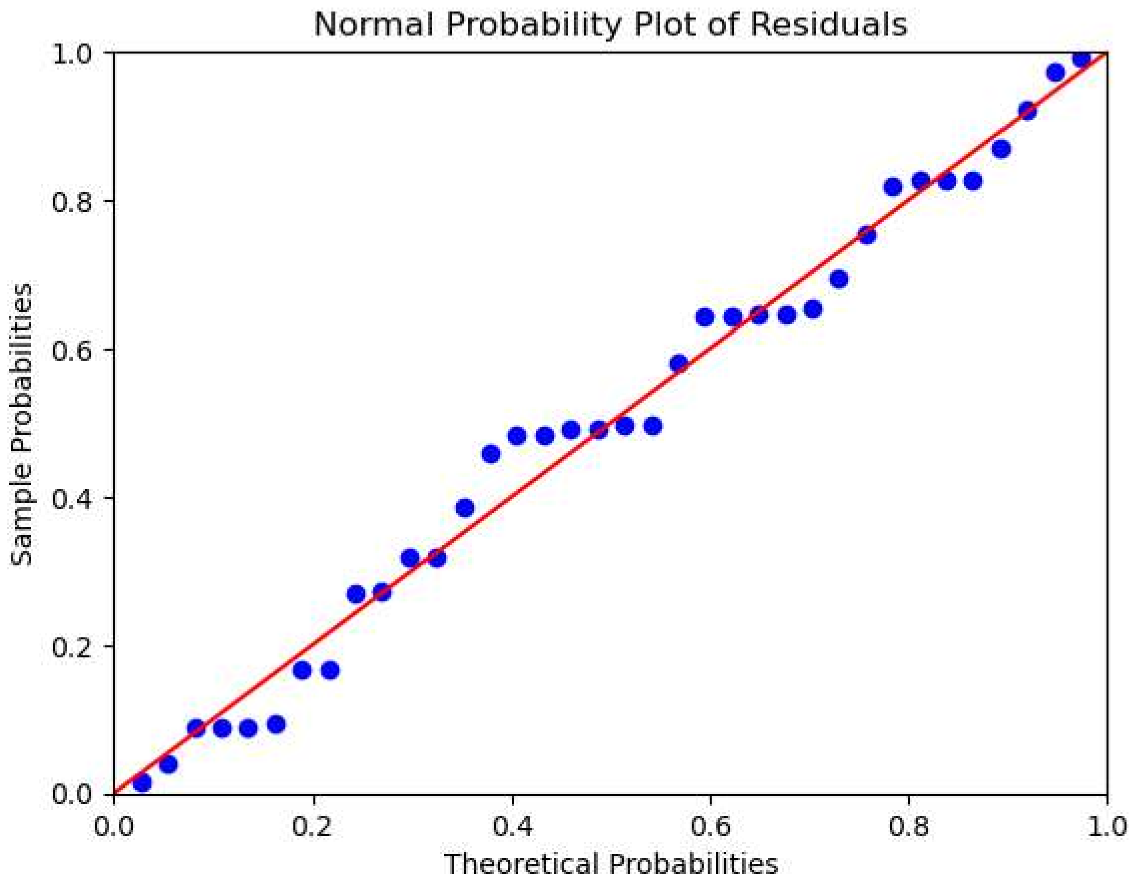
In [172]:
```
print('Confidence Interval for B_3:   ')
print(popt[2]-(2.037*pcov[2,2]**0.5), ',', popt[2]+(2.037*pcov[2,2]**0.5))
```

```
Confidence Interval for B_3:
0.03893443450520916 , 0.12157999019601332
```

Part (e)

In [128]:
```
res = fulldata[:,0] - ypred
```

In [132]:
```
import statsmodels.api as sm
import scipy.stats as stats
pplot = sm.ProbPlot(res, stats.t, fit=True)
fig = pplot.ppplot(line="45")
h = plt.title("Normal Probability Plot of Residuals")
plt.show()
```



In [135]:
```
plt.scatter(fulldata[:,0],res, c='b')
plt.plot([0.375,0.485],[0,0],c='r')
plt.xlim(0.375,0.485)
plt.xlabel('y')
plt.ylabel('Residual')
h = plt.title("Residuals as a Function of y")
plt.show()
```