CC03 Electricity and Magnetism Practicals

RC Circuit. (Investigation of Capacitance)

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy as sp
from scipy.optimize import curve_fit
import sympy as smp
```

In [2]:
```
x_data = np.array([0.24,
0.33,
0.49,
0.56,
0.67,
0.81,
0.93,
1.16,
1.28,
1.4])
y_data = np.array([0.0005757575758,
0.001393939394,
0.001909090909,
0.002303030303,
0.002666666667,
0.003181818182,
0.003636363636,
0.004090909091,
0.004454545455,
0.00496969697])*1000  # in mA
R = 330   # in ohms
f1 = 1000  # in Hz
```

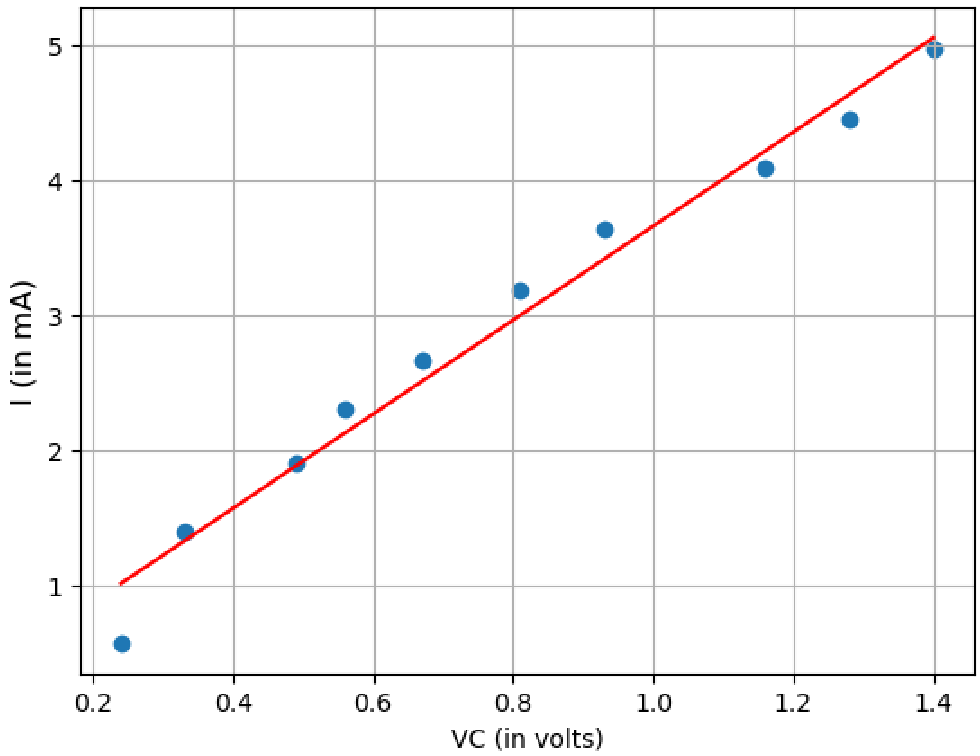
In [3]:
```python
def model_f(x, a, b):
    return a*x + b

popt, pcov = curve_fit(model_f, x_data, y_data, p0=[1,0])
a_opt, b_opt = popt
x_model = np.linspace(min(x_data), max(x_data), 100)
y_model = model_f(x_model, a_opt, b_opt)

plt.scatter(x_data,y_data)
plt.plot(x_model,y_model, color='r')
plt.xlabel('VC (in volts)')
plt.ylabel('I (in mA)', fontsize=12)
plt.grid()
plt.show()

x, y, lam = smp.symbols('x y \lambda', real=True, positive=True)
y = a_opt*x + b_opt
C1 = a_opt/(2*np.pi*f1)*1000    # in micro F

print('Slope of the graph, I/VC  =', a_opt, '\n')
print('Capacitance is, C1 =', C1, 'micro F.')
```




```
Slope of the graph, I/VC  = 3.476682670553934

Capacitance is, C1 = 0.5533312325805901 micro F.
```

In [4]:
```
x_data = np.array([0.08,
0.18,
0.24,
0.35,
0.46,
0.59,
0.7,
0.77,
0.88,
0.99])
y_data = np.array([0.0003939393939,
0.0008787878788,
0.001181818182,
0.001666666667,
0.002151515152,
0.002727272727,
0.003181818182,
0.003545454545,
0.004060606061,
0.004545454545])*1000  # in mA
R = 330   # in ohms
f2 = 2000  # in Hz
```

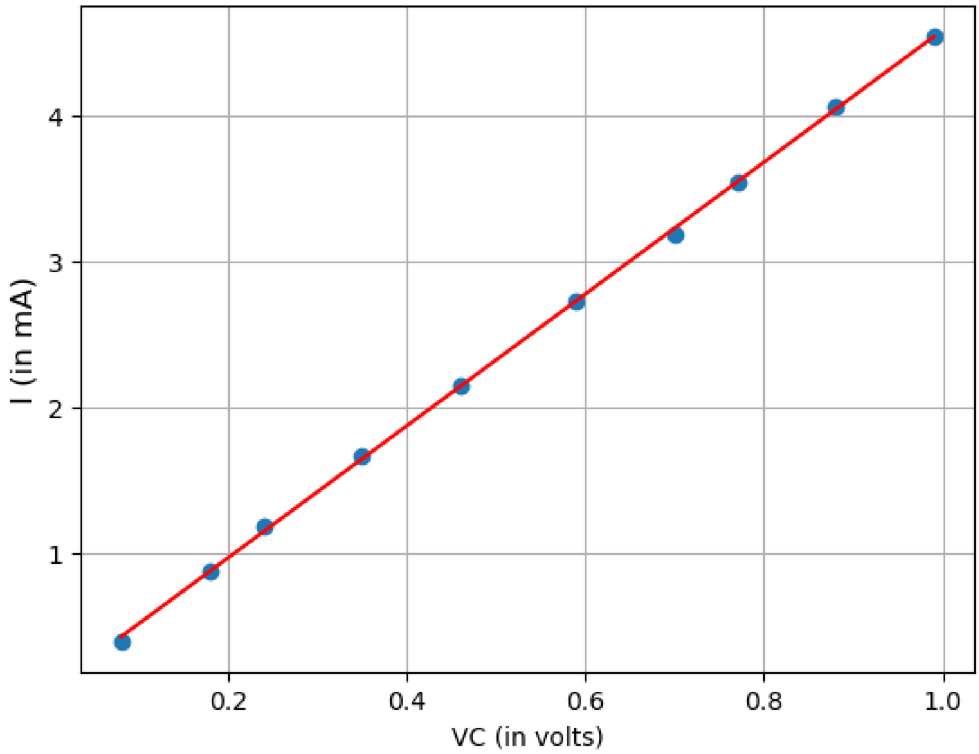
In [5]:
```
def model_f(x, a, b):
    return a*x + b

popt, pcov = curve_fit(model_f, x_data, y_data, p0=[1,0])
a_opt, b_opt = popt
x_model = np.linspace(min(x_data), max(x_data), 100)
y_model = model_f(x_model, a_opt, b_opt)

plt.scatter(x_data,y_data)
plt.plot(x_model,y_model, color='r')
plt.xlabel('VC (in volts)')
plt.ylabel('I (in mA)', fontsize=12)
plt.grid()
plt.show()

x, y, lam = smp.symbols('x y \lambda', real=True, positive=True)
y = a_opt*x + b_opt
C2 = a_opt/(2*np.pi*f2)*1000    # in micro F

print('Slope of the graph, I/VC  =', a_opt, '\n')
print('Capacitance is, C2 =', C2, 'micro F.')
```




```
Slope of the graph, I/VC  = 4.522193806258143

Capacitance is, C2 = 0.35986474894276815 micro F.
```

In [ ]:

In [6]:
```
x_data = np.array([100,
200,
300,
400,
500,
600,
700,
800,
900])
y_data = np.array([0.0001818181818,
0.0003979185797,
0.0006560449859,
0.0008612440191,
0.001096067054,
0.00128458498,
0.001498127341,
0.001706722396,
0.001935012778])
R = 330   # in ohms
Vs = 1    # supply voltage
```

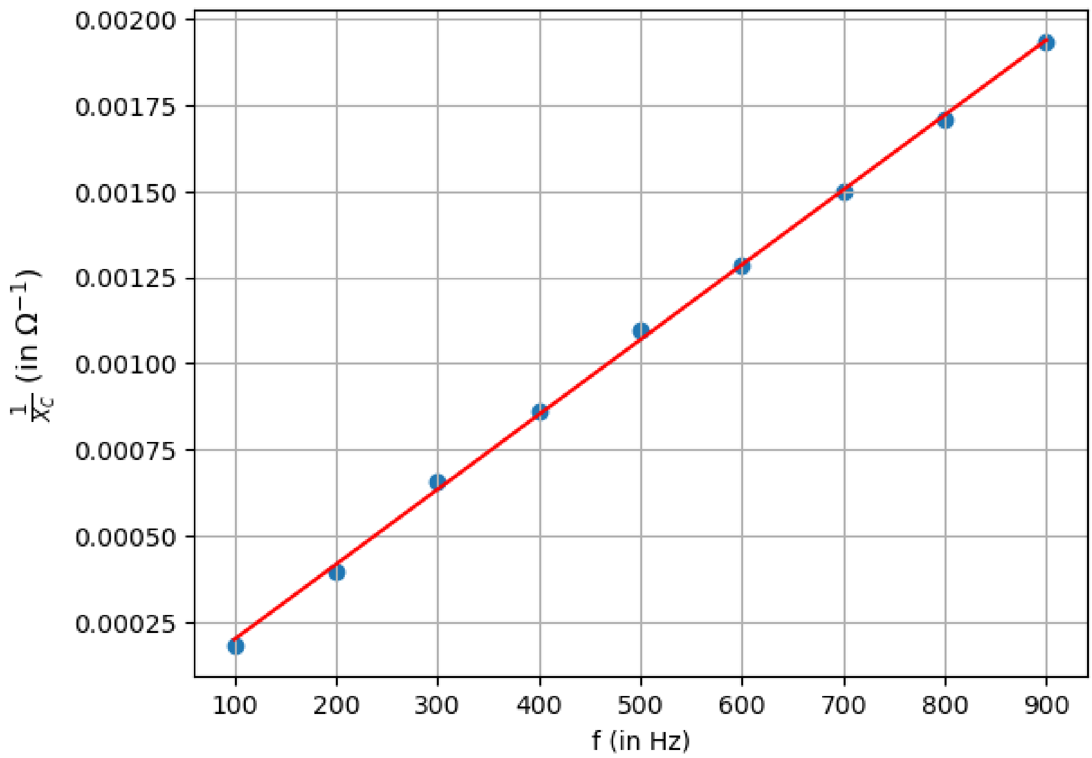
In [7]:
```
def model_f(x, a, b):
    return a*x + b

popt, pcov = curve_fit(model_f, x_data, y_data, p0=[1,0])
a_opt, b_opt = popt
x_model = np.linspace(min(x_data), max(x_data), 100)
y_model = model_f(x_model, a_opt, b_opt)

plt.scatter(x_data,y_data)
plt.plot(x_model,y_model, color='r')
plt.xlabel('f (in Hz)')
plt.ylabel(r'$\frac{1}{X_C}$ (in $\Omega^{-1}$)', fontsize=12)
plt.grid()
plt.show()

x, y, lam = smp.symbols('x y \lambda', real=True, positive=True)
y = a_opt*x + b_opt
C2 = a_opt/(2*np.pi)* 10**6   # in micro F

print('Slope of the graph, I/V  =', a_opt, '\n')
print('Capacitance is, C2 =', C2, 'micro F.')
```




```
Slope of the graph, I/V  = 2.1744492553196062e-06

Capacitance is, C2 = 0.34607434748660615 micro F.
```

In [ ]:

LR Circuit. (Investigation of Inductance)

In [ ]:

In [8]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy as sp
from scipy.optimize import curve_fit
import sympy as smp
```

In [9]:
```
x_data = np.array([0.9,
1.29,
1.51,
1.63,
2.22])
y_data = np.array([0.0022,
0.003,
0.0037,
0.004,
0.0054])   # in A
R = 100   # in ohms
f1 = 2000  # in Hz
```

In [10]:

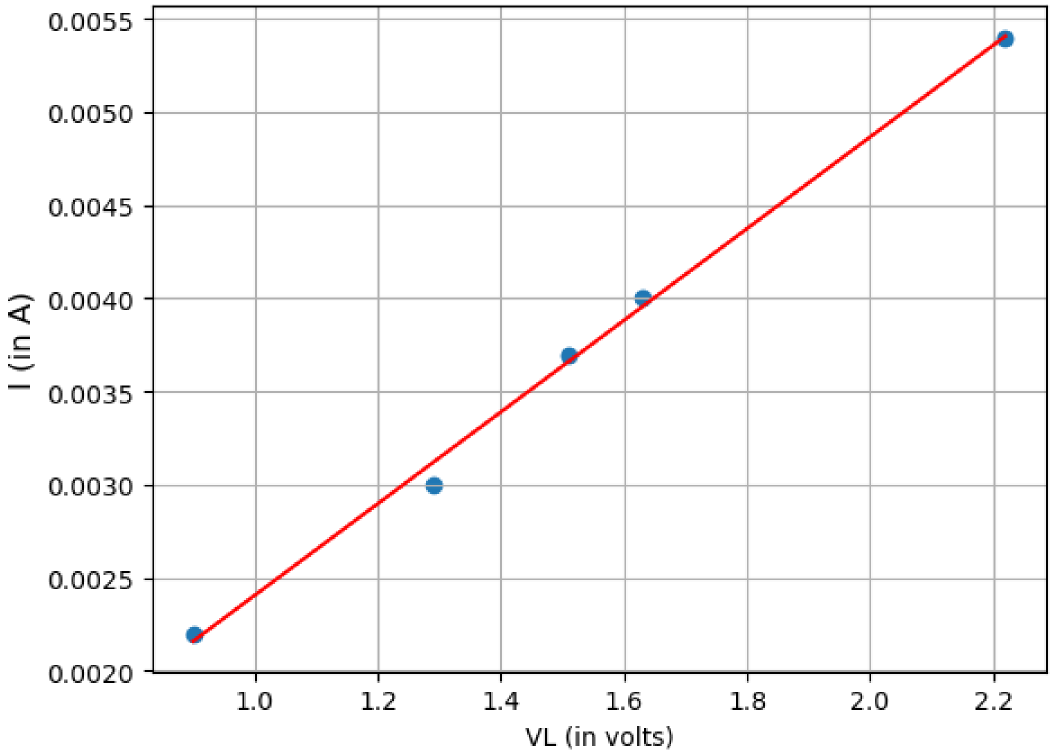
```
def model_f(x, a, b):
    return a*x + b

popt, pcov = curve_fit(model_f, x_data, y_data, p0=[1,0])
a_opt, b_opt = popt
x_model = np.linspace(min(x_data), max(x_data), 100)
y_model = model_f(x_model, a_opt, b_opt)

plt.scatter(x_data,y_data)
plt.plot(x_model,y_model, color='r')
plt.xlabel('VL (in volts)')
plt.ylabel('I (in A)', fontsize=12)
plt.grid()
plt.show()

x, y, lam = smp.symbols('x y \lambda', real=True, positive=True)
y = a_opt*x + b_opt
L1 = 1/(2*np.pi*f1*a_opt)*1000    # in mH

print('Slope of the graph, I/VL  =', a_opt, '\n')
print('Inductance is, L1 =', L1, 'mH.')
```




```
Slope of the graph, I/VL  = 0.0024621938239033853

Inductance is, L1 = 32.31974297612008 mH.
```

In [ ]:

In [11]:
```
x_data = np.array([2.78,
2.27,
1.84,
1.35,
0.92])
y_data = np.array([0.0036,
0.0029,
0.0024,
0.0017,
0.0011])   # in A
R = 100   # in ohms
f2 = 5000  # in Hz
```

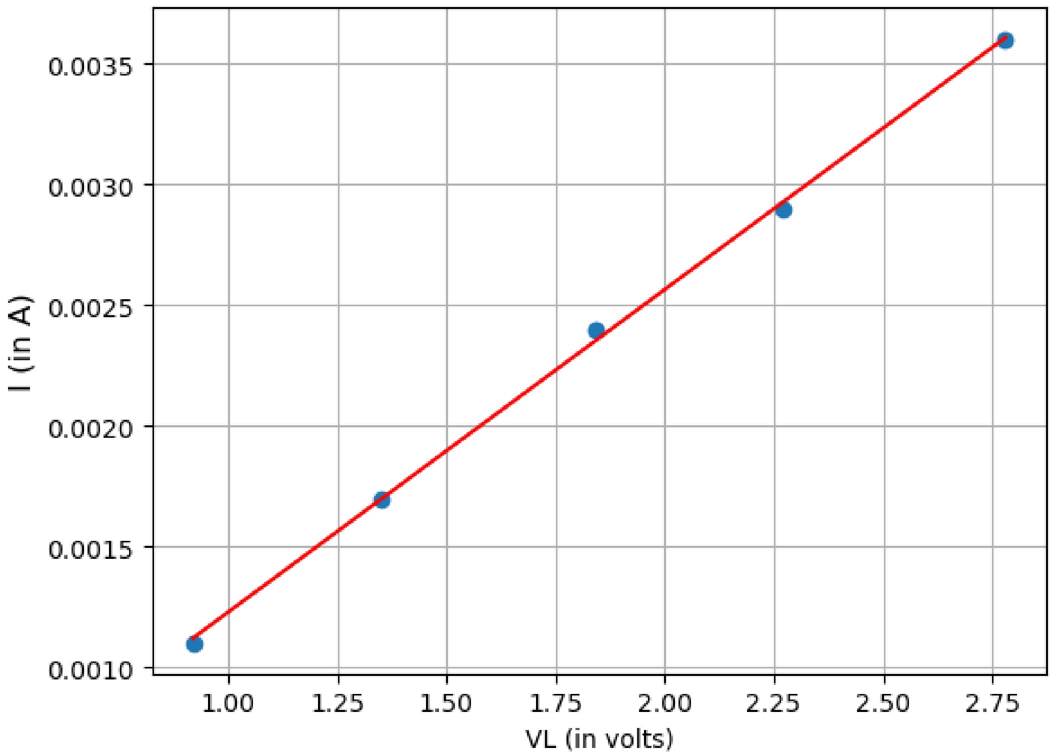
In [12]:
```
def model_f(x, a, b):
    return a*x + b

popt, pcov = curve_fit(model_f, x_data, y_data, p0=[1,0])
a_opt, b_opt = popt
x_model = np.linspace(min(x_data), max(x_data), 100)
y_model = model_f(x_model, a_opt, b_opt)

plt.scatter(x_data,y_data)
plt.plot(x_model,y_model, color='r')
plt.xlabel('VL (in volts)')
plt.ylabel('I (in A)', fontsize=12)
plt.grid()
plt.show()

x, y, lam = smp.symbols('x y \lambda', real=True, positive=True)
y = a_opt*x + b_opt
L2 = 1/(2*np.pi*f2*a_opt)*1000   # in mH

print('Slope of the graph, I/VL  =', a_opt, '\n')
print('Inductance is, L2 =', L2, 'mH.')
```




```
Slope of the graph, I/VL  = 0.0013364397497342398

Inductance is, L2 = 23.81775057551893 mH.
```

Try rest of the graphs.

```
In [ ]:
```