```
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 18 19:56:56 2020

@author: maxhu
collaborator: grant
"""
#=====================Importing Modules=====================================#
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
plt.style.use('classic')
#=====================Defining Constants====================================#
"""
39 uV/C
Where data is in mV
======> 39e-3 V/C
"""
VoltageToCelsiusConversion = 39e-3
PsiToAtm = 0.068046
LiteratureValues = [1.11, .27, -.062] #CO2, N2, He respectively\
Gasses = ['CO2', 'N2', 'He']
#=========================Importing Data====================================#
data = pd.read_excel (
        r'C:\Users\maxhu\Documents\PChem\PchemII\M1\Joule_Thompson\
        M1_Joule_thompson_Data.xlsx'
        , sheet_name='CO2')

df = pd.DataFrame(data, columns= ['psi'])
hold = df.values.tolist()
CO2Pressure  = np.array([val for sublist in hold for val in sublist])
CO2Pressure = CO2Pressure * PsiToAtm

df = pd.DataFrame(data, columns= ['mV'])
hold = df.values.tolist()
CO2Voltage  = np.array([val for sublist in hold for val in sublist])
CO2Temperature = CO2Voltage / VoltageToCelsiusConversion
#===========================================================#
data = pd.read_excel (
        r'C:\Users\maxhu\Documents\PChem\PchemII\M1\Joule_Thompson\
        M1_Joule_thompson_Data.xlsx'
        , sheet_name='N2')

df = pd.DataFrame(data, columns= ['psi'])
hold = df.values.tolist()
N2Pressure  = np.array([val for sublist in hold for val in sublist])
N2Pressure = N2Pressure * PsiToAtm

df = pd.DataFrame(data, columns= ['mV'])
hold = df.values.tolist()
N2Voltage  = np.array([val for sublist in hold for val in sublist])
N2Temperature = N2Voltage / VoltageToCelsiusConversion
#===========================================================#
data = pd.read_excel (
        r'C:\Users\maxhu\Documents\PChem\PchemII\M1\Joule_Thompson\
```

```
        M1_Joule_thompson_Data.xlsx'
        , sheet_name='He')

df = pd.DataFrame(data, columns= ['psi'])
hold = df.values.tolist()
HePressure  = np.array([val for sublist in hold for val in sublist])
HePressure = HePressure * PsiToAtm

df = pd.DataFrame(data, columns= ['mV'])
hold = df.values.tolist()
HeVoltage  = np.array([val for sublist in hold for val in sublist])
HeTemperature = HeVoltage / VoltageToCelsiusConversion
#=================My Linear Regression Function==========================#
def LinearRegression(x,y, GasName):
    m, b, r_value, p_value, std_err = stats.linregress(
            x,y)
    slope=round(m,6)
    if GasName == 'CO2':
        error = round((abs(LiteratureValues[0] - m) / LiteratureValues[0]
        ) * 100, 2)
    if GasName == 'N2':
        error = round((abs(LiteratureValues[1] - m) / LiteratureValues[1]
        ) * 100, 2)
    if GasName == 'He':
        error = round(abs((LiteratureValues[2] - m) / LiteratureValues[2]
        ) * 100, 2)
    print('The slope of {} is {} K/atm with an error of {}%'.format(
            GasName, slope, error))
    return m*x + b
#=======================Plotting=========================================#
size = 20
size_config = .8
fig = plt.figure(1,figsize=(10,6))
my_fig = fig.add_subplot(111)
fig.suptitle('$Temperature\ as\ a\ Function\ of\ Pressure$', fontsize=size)
plt.xlabel('$Pressure\ (atm)$', fontsize=size_config*size)
plt.ylabel('$Temperature\ (C)$', fontsize=size_config*size)
#========================================================#
plt.plot(CO2Pressure, LinearRegression(CO2Pressure,-CO2Temperature, 'CO2'),
         color='black', label='$CO_2\ Fit$', linestyle='dotted')
plt.scatter(CO2Pressure,-CO2Temperature,color='black', label='$CO_2\ Data$',
            marker ='+')
#========================================================#
plt.plot(N2Pressure, LinearRegression(N2Pressure,-N2Temperature, 'N2'),
         color='black', label='$N_2\ Fit$', linestyle='dashed')
plt.scatter(N2Pressure,-N2Temperature,color='black', label='$N_2\ Data$',
            marker ='H')
#========================================================#
plt.plot(HePressure, LinearRegression(HePressure,-HeTemperature, 'He'),
         color='black', label='$He\ Fit$', linestyle='dashdot')
plt.scatter(HePressure,-HeTemperature,color='black', label='$He\ Data$',
            marker ='x')
#========================================================#
plt.legend(loc='best')
plt.savefig('Joule_Thompson.png', dpi=600)
#============================================================================#
```