Measuring pressure drop over a venturi or orifice plate to determine the flow rate of the system.

Course code:	ENME314
Lab title:	LabA: Flow Rate measurement
Author:	Justin France
Student number:	75349339
Student email:	Jfr125@uclive.ac.nz
Date:	05/03/2025
Lab partners:	Lauren Horlick and Callum Douglas

Introduction:

Test were carried out to determine if pressure drop over a venturi or orifice plate would correlate, to the flow rate measurements recorded from the electronic flow meter. Using a rearranged form of Bernoulli equation and a discharge coefficient to theoretically predict the relationship between pressure drops and flow rates. To verify these results the rotameter reading will also be compared against the electronic flow meter where the calibration curve was theoretically determined by linear regression.

Experimental data collected from the piezometers $[h_1 , h_2]$ allowed for the difference in head to be calculated and therefore the difference in pressure $[\Delta P]$ across the venturi and orifice plate to be obtained. Reynolds number was calculated using the electronic flow mete flow rates, which was converted into a velocity $[U]$, as well as using the kinematic velocity for water $[\mu]$ and diameter of the contractions$[D]$.

The discharge coefficient was required to account for vicious effects and followed the ISO standard 5167 and corresponding tables for discharge coefficients. The venturi discharge coefficient was determined by comparing the function of throat Reynalds number, provided by the ISO standard 5167 for venturi, agents the experimental obtained Reynolds number for the venturi. The plate orifice discharged coefficient was selected in a similar manner but using an additional diameter ratio$\left[\frac{D_2}{D_1}\right]$ to select the discharge coefficient [Refer to lab manual].

From the calculated pipe and contraction areas $[A_1, A_2]$, density $[\rho]$, gravity$[g]$, discharge coefficient $[c_d]$ and rearrangement of the Bernoulli along with the continuity equation in conjunction with the following assumptions: inviscid flow, incompressible, steady flow, density remains constant, no heat or work done. Results in the following formulars.

$$\Delta \boldsymbol{P} = (\boldsymbol{p_1} - \boldsymbol{p_2}) = \boldsymbol{\rho g h_1} - \boldsymbol{\rho g h_2} \quad \textbf{Eqn1}$$

$$\boldsymbol{R_e} = \frac{\boldsymbol{UD}}{\boldsymbol{\mu}} \quad \textbf{Eqn2}$$

$$\boldsymbol{Q} = \boldsymbol{c_d A_2} \sqrt{\frac{2(\Delta \boldsymbol{P})}{\boldsymbol{\rho}(1 - \frac{\boldsymbol{A_2^2}}{\boldsymbol{A_1^2}})}} \quad \textbf{Eqn3}$$

The calculations and data handling of the above formulars including uncertainties and was carried out in code see appendix.

Methodology / apparatus:

The rotameter uses fluids drag force of a specifically sized float to counteract gravitational forces and settles at an equilibrium position, where the flow rate is proportional to the height of the float. The rotameter readings were used to set the determined flow rates between each sample. Where the Piezometer readings were taken from tapings on or over the venturi inlet, venturi throat, both sides of the diffuser and either side of the orifice plate to collect the experiment data. Flow testing was carried out in accordance with the lab manual procedures for setting up the equipment and conducting the flow rate test.

Where the lab manual states the measured dimensions for the pipes, contractions and associated uncertainties values relating to the measuring equipment. It was assumed that the TecQuiptment H1F digital hydraulic bench flow meter reads with zero error and the system has no leaks.

The error for density, was from the accuracy of thermometer which varied by $\pm \mathbf{1^oC}$ and this corresponds to an error for density of $\boldsymbol{E_\rho = \pm 0.25}$ **[kg/m³].** This value was obtained from a density vs temperature table from the lab manual and the equation $[\frac{\rho_{(Temp=30)} - \rho_{(Temp=20)}}{10}]$.

Uncertainties for diameter measurements reading measurements, was $\boldsymbol{E_d = \pm 0.0005}$ **[m]** for the throats and pipe diameters along with the following equations the uncertainty for area was calculated.

$$\boldsymbol{E_{A1(venturi)} = \frac{\pi d_1}{2} E_d = \frac{\pi * 0.026}{2} * 0.0005 = \pm 0.0000204\ [m^2]}$$

$$\boldsymbol{E_{A2(venturi)} = \frac{\pi d_2}{2} E_d = \frac{\pi * 0.016}{2} * 0.0005 = \pm 0.0000126\ [m^2]}$$

Uncertainties from the piezometer readings errors $\boldsymbol{E_{h_1}, E_{h_2} = \pm 0.0005\ [m]}$ where the following equations the uncertainty for change in pressure was calculated for test 2 sample 1 for the venturi.

$$\boldsymbol{E_{\Delta p} = \pm\sqrt{\left(g(h_1 - h_2)E_\rho\right)^2 + \left(\rho g E_{h_1}\right)^2 + \left(\rho g E_{h_2}\right)^2}}$$

$$\boldsymbol{E_{\Delta p} = \pm\sqrt{(9.81(0.312 - 0.304)0.25)^2 + (998.2 * 9.81 * 0.0005)^2 + (998.2 * 9.81 * 0.0005)^2}}$$

$$\boldsymbol{E_{\Delta p} = \pm 6.9\ [Pa]}$$

Solving was carried out for subcomponents of the flow rate equation for: density, area and pressure with respect to flow rate which was then combined to form the flow rate uncertainty equation which assumes c_d error was negligible. Below shows an example for the venturi test 2 sample 1.

Density uncertainty:	$\boldsymbol{\left\lvert\frac{\partial Q}{\partial p}\right\rvert = \frac{0.5}{\rho} Q}$	$\frac{0.5}{998.2} * 8.35 * 10^{-5}$	$= 4.18 * 10^{-9}$
Area1 uncertainty:	$\boldsymbol{\left\lvert\frac{\partial Q}{\partial A_1}\right\rvert = \frac{1}{\left(\frac{1}{A_2^2} - \frac{1}{A_1^2}\right) A_2^3} Q}$	$\frac{1}{\left(\frac{1}{0.0002^2} - \frac{1}{0.0005^2}\right) * 0.0002^3} * 8 * 10^{-5}$	$= 0.497$
Area2 uncertainty:	$\boldsymbol{\left\lvert\frac{\partial Q}{\partial A_2}\right\rvert = \frac{1}{\left(\frac{1}{A_1^2} - \frac{1}{A_2^2}\right) A_1^3} Q}$	$\frac{1}{\left(\frac{1}{0.0005^2} - \frac{1}{0.0002^2}\right) * 0.0005^3} * 8 * 10^{-5}$	$= -0.032$
Pressure uncertainty:	$\boldsymbol{\left\lvert\frac{\partial Q}{\partial \Delta P}\right\rvert = \frac{0.5}{\Delta P} Q}$	$\frac{0.5}{78.34} * 8.35 * 10^{-5}$	$= 532.9 * 10^{-9}$

Flow rate uncertainty:

$$\boldsymbol{E_Q = \sqrt{\left(\left\lvert\frac{\partial Q}{\partial \Delta P}\right\rvert E_{\Delta p}\right)^2 + \left(\left\lvert\frac{\partial Q}{\partial \rho}\right\rvert E_\rho\right)^2 + \left(\left\lvert\frac{\partial Q}{\partial A_1}\right\rvert E_{A1}\right)^2 + \left(\left\lvert\frac{\partial Q}{\partial A_2}\right\rvert E_{A2}\right)^2}}$$

$$E_Q = \sqrt{(532.9 * 10^{-9} * 6.9)^2 + (4.18 * 10^{-9} * 0.25)^2 + (0.497 * 0.0000204)^2 + (-0.032 * 0.0000126)^2}$$

$$E_Q = 101.5 * 10^{-6}\ [\text{m}^3/\text{s}]$$

Results:

Table 1: Experiment 1 piezometer, rotameter and electronic flow meter raw data at a temperature of 20°C

Rotameter ±0.5[mm³/s]	Piezometer A ±0.5 [mm]	Piezometer B ±0.5 [mm]	Piezometer C ±0.5 [mm]	Piezometer D ±0.5[mm]	Piezometer E ±0.5[mm]	Piezometer F ±0.5 [mm]	Digital flow rate [l/m]
20	314	304	312	312	312	302	5.1
40	316	296	312	312	312	290	7.7
60	318	284	312	312	314	276	10.4
80	324	270	314	316	318	256	12.8
100	330	252	318	320	324	234	15.6
120	340	232	324	326	332	202	18.4
140	352	204	330	336	342	166	21.5
160	368	172	340	346	354	122	24.7
180	382	138	348	346	368	74	27.5

Table 2: Experiment 2 piezometer, rotameter and electronic flow meter raw data at a temperature of 21°C

Rotameter ±0.5[mm³/s]	Piezometer A ±0.5 [mm]	Piezometer B ±0.5 [mm]	Piezometer C ±0.5 [mm]	Piezometer D ±0.5 [mm]	Piezometer E ±0.5 [mm]	Piezometer F ±0.5 [mm]	Digital flow rate [l/m]
20	312	304	310	312	312	302	5
40	314	294	308	310	310	290	7.7
60	316	282	308	308	310	272	10.3
80	318	266	308	310	312	252	12.7
100	324	244	310	312	316	222	15.6
120	328	220	310	314	320	190	18.5
140	334	188	312	316	324	148	21.4
160	344	148	312	320	328	98	24.8
180	348	102	312	324	332	38	27.5

Table 3: Experiment 3 piezometer, rotameter and electronic flow meter raw data at a temperature of 22°C

Rotameter ±0.5[mm³/s]	Piezometer A ±0.5 [mm]	Piezometer B ±0.5 [mm]	Piezometer C ±0.5 [mm]	Piezometer D ±0.5 [mm]	Piezometer E ±0.5 [mm]	Piezometer F ± 0.5[mm]	Digital flow rate [l/m]
20	312	304	310	310	312	302	5.1
40	314	294	310	310	310	282	7.8
60	314	280	308	310	310	272	10
80	318	264	308	310	312	250	12.8
100	322	244	310	310	314	222	15.6
120	328	218	310	314	318	198	18.3
140	336	186	312	316	324	146	21.5
160	340	146	312	318	328	96	24.4
180	344	100	312	320	330	36	27.7

Table 4: Experiment 1 processed data at a temperature of 20°C

Venturi Q [m³/s]	Venturi Q error ± [m³/s]	Orifice Q [m³/s]	Orifice Q error ± [m³/s]	Electronic flow rate [m³/s]	Venturi ΔP [Pa]	Error venturi ±ΔP [Pa]	Orifice ΔP [Pa]	Error orifice ±ΔP [Pa]	Reynalds number venturi	Reynalds number orifice
9.3E-05	7.0E-06	8.6E-05	5.3E-06	8.50E-05	97.92	6.92	97.92	6.92	6717	5374
1.3E-04	9.7E-06	1.3E-04	6.9E-06	1.28E-04	195.85	6.92	215.43	6.92	10115	8092
1.7E-04	1.3E-05	1.7E-04	9.0E-06	1.73E-04	332.94	6.92	372.11	6.92	13671	10937
2.2E-04	1.6E-05	2.1E-04	1.1E-05	2.13E-04	528.79	6.93	607.13	6.93	16832	13466
2.6E-04	1.9E-05	2.6E-04	1.3E-05	2.60E-04	763.80	6.93	881.31	6.93	20546	16437
3.1E-04	2.3E-05	3.1E-04	1.6E-05	3.07E-04	1057.57	6.93	1273.00	6.93	24260	19408
3.6E-04	2.6E-05	3.6E-04	1.8E-05	3.58E-04	1449.27	6.93	1723.45	6.94	28291	22633
4.1E-04	3.0E-05	4.1E-04	2.1E-05	4.12E-04	1919.30	6.94	2271.82	6.95	32558	26046
4.6E-04	3.4E-05	4.7E-04	2.3E-05	4.58E-04	2389.33	6.95	2878.95	6.96	36193	28955

Table 5: Experiment 2 processed data at a temperature of 21°C

Venturi Q [m³/s]	Venturi Q error ± [m³/s]	Orifice Q [m³/s]	Orifice Q error ± [m³/s]	Electronic flow rate [m³/s]	Venturi ΔP [Pa]	Error venturi ±ΔP [Pa]	Orifice ΔP [Pa]	Error orifice ±ΔP [Pa]	Reynalds number venturi	Reynalds number orifice
8.3E-05	7.1E-06	8.6E-05	5.2E-06	8.30E-05	78.36	6.93	97.95	6.93	6559	5247
1.3E-04	9.7E-06	1.2E-04	6.9E-06	1.28E-04	195.90	6.93	195.90	6.93	10115	8092
1.7E-04	1.3E-05	1.7E-04	8.9E-06	1.72E-04	333.02	6.93	372.20	6.93	13592	10874
2.1E-04	1.6E-05	2.1E-04	1.1E-05	2.12E-04	509.33	6.93	587.69	6.93	16753	13403
2.6E-04	1.9E-05	2.6E-04	1.3E-05	2.60E-04	783.58	6.93	920.71	6.93	20546	16437
3.1E-04	2.3E-05	3.1E-04	1.6E-05	3.08E-04	1057.84	6.93	1273.32	6.93	24339	19472
3.6E-04	2.6E-05	3.6E-04	1.8E-05	3.57E-04	1430.04	6.94	1723.88	6.94	28212	22569
4.1E-04	3.0E-05	4.1E-04	2.1E-05	4.13E-04	1919.78	6.94	2252.80	6.95	32637	26110
4.6E-04	3.4E-05	4.7E-04	5.2E-06	4.58E-04	2409.52	6.95	2879.67	6.96	36193	28955

Table 6: Experiment 3 processed data at a temperature of 22°C

Venturi Q [m³/s]	Venturi Q error ± [m³/s]	Orifice Q [m³/s]	Orifice Q error ± [m³/s]	Electronic flow rate [m³/s]	Venturi ΔP [Pa]	Error venturi ±ΔP [Pa]	Orifice ΔP [Pa]	Error orifice ±ΔP [Pa]	Reynalds number venturi	Reynalds number orifice
8.3E-05	7.2E-06	8.6E-05	5.3E-06	8.50E-05	78.37	6.93	97.96	6.93	6717	5374
1.3E-04	9.8E-06	1.4E-04	6.9E-06	1.30E-04	195.92	6.93	274.29	6.93	10273	8219
1.7E-04	1.2E-05	1.7E-04	8.7E-06	1.67E-04	333.06	6.93	372.25	6.93	13197	10558
2.2E-04	1.6E-05	2.1E-04	1.1E-05	2.13E-04	528.99	6.93	607.35	6.93	16832	13466
2.6E-04	1.9E-05	2.6E-04	1.3E-05	2.60E-04	764.09	6.93	901.23	6.93	20546	16437
3.1E-04	2.2E-05	3.0E-04	1.6E-05	3.05E-04	1077.56	6.93	1175.52	6.93	24102	19282
3.6E-04	2.6E-05	3.6E-04	1.8E-05	3.58E-04	1469.40	6.94	1743.69	6.94	28291	22633
4.1E-04	3.0E-05	4.1E-04	2.1E-05	4.07E-04	1900.43	6.94	2272.68	6.95	32163	25730
4.6E-04	3.4E-05	4.7E-04	2.4E-05	4.62E-04	2390.23	6.95	2880.03	6.96	36509	29207

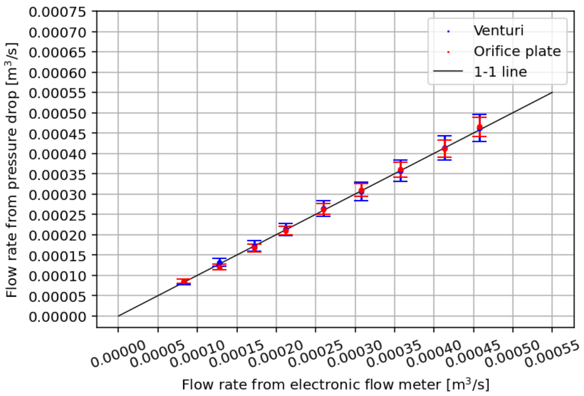

Figure 2: Experiment 2 comparing flow rates from pressure drop over a venturi and orifice plate to an electronic flow meter.

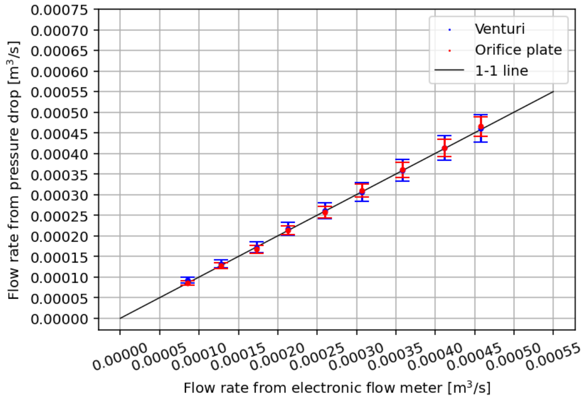

Figure 1: Experiment 1 comparing flow rates from pressure drop over a venturi and orifice plate to an electronic flow meter.

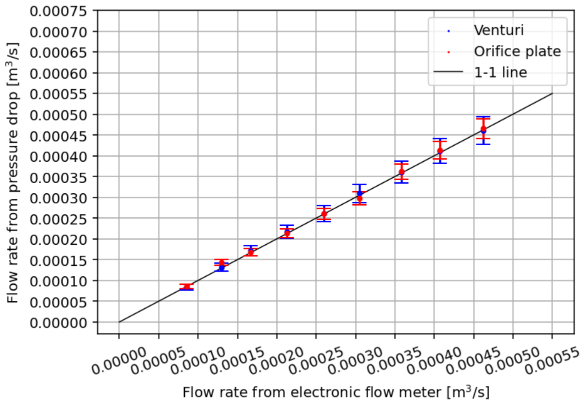

Figure 3: Experiment 3 comparing flow rates from pressure drop over a venturi and orifice plate to an electronic flow meter.

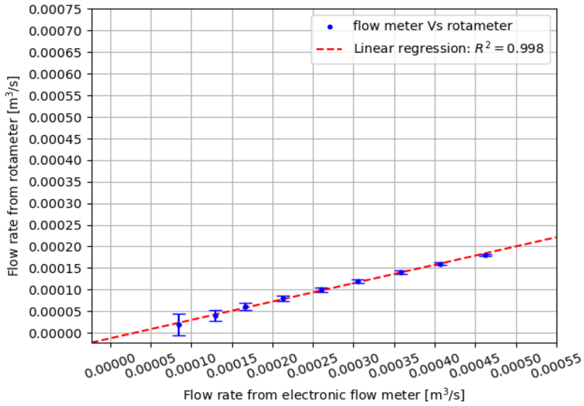

Figure 4: comparing the Rotameter values against the electronic flow meter to determine the calibration curve.

Discussion:

Venturis or orifice plates are known as obstruction meters as each configuration causes some restriction to the system and thus a pressure loss which was measured from piezometers attached to tapings across the apparatus components. From the difference in pressure along with the coefficient discharge, continuity and Bernoulli's equation the flow rate can be determined so long as the assumptions are meet. Sine all the mass flow in the system must past through the venturi and orifice plate the flow rate of the system can be accurately measured and compared agenst the electronic flow meter readings which agrees with the plotted data.

Considering for vicious drag forces over the venturi or orifice plate, the corresponding Reynolds numbers was obtained which was used to correctly select the coefficient discharge values for the venturi and orifice plate. With the coefficient of discharges values of $\boldsymbol{c_d = 0.97}$ for the venturi and $\boldsymbol{c_d = 0.61}$ the orifice plates the vicious losses was included in the flow rate equation to accurately model the system flow rate.

From the plot and calculation for the linear regression $\boldsymbol{R^2 = 0.998}$ shows the rotameter is calibrated correctly calibrated against the electronic flow meter when used in water at $\approx \boldsymbol{20^o C}$.

The piezometer uncertainties that were taken are relatively accurate, when comparing data points across the plots. These inaccuracies most likely propagated from the interpretation of the piezometer readings during testing as the precision of the measuring instruments were relatively negligible in comparison. To improve the precision, it would be recommended to measure the diameters of the venturi and orifice with a measuring tool of higher precision.

The assumption that no energy transfer occurs is invalid as energy is transferred into heat as flow separation occurs at the orifice plate as high shear flow stress and eddies to form after the contraction. The turbulent flow converges and creates heat energy which should be included in the model for flow rate as work done from the system.

During testing some of the piezometer readings were fluctuating thus indicating the flow, is not steady when the apparatus is operating at higher flow rates, this was particularly noticeable on the orifice where the tapping is in a region of high shear flow and turbulence. I would be advised to ensure the system is manufactured and configured to the exact specifications as per the ISO standard 5167. To further decrease internal energy losses of the system the number of bends should be reduced as this would lessen the associated effects of deans' vortices.

Porting and polishing of the pipes and components of the apparatus, so that only smooth streams lines along the internal walls of components would reduce the amount of viscous drag hence increasing the accuracy of the readings. Since energy is added to the system it would be recommended to install a cooler to ensure the apparatus fluid is maintained to a specific temperature and therefore the water would remain at a constant density.

Conclusion:

The experimental approach used to determine the relationship between pressure drops and flow rates, is valid when using a rearranged form of Bernoulli equation, with an appropriately selected coefficient of discharge as per the ISO standard 5167. Although inaccuracies from, interpretation of the piezometer's readings and the measurement precision from throat diameters allowed for some deviation as shown by the plots, this experiment including the calibration of the rotameter reasonably agrees with the electronic flow meter readings.

Appendix:

```python
@author: Justin France
"""


import matplotlib.pyplot as plt
import numpy as np
from scipy import stats
import pandas as pd



def load_data(data_set1, data_set2, data_set3):
    """ Load in data from exel sheet """
    data_1 = np.loadtxt(data_set1, delimiter=",", skiprows=1)

    data_2 = np.loadtxt(data_set2, delimiter=",", skiprows=1)

    data_3 = np.loadtxt(data_set3, delimiter=",", skiprows=1)
    # print("data_set1")
    # print(data_1)
    # print('\n')

    # print("data_set2")
    # print(data_2)
    # print('\n')

    # print("data_set3")
    # print (data_3)
    # print("\n")
    return(data_1, data_2, data_3)



def process_data(data, cf_m, fist_pizo, second_pizo, cf_ms):
    """ slect data set for pizeometer  for the venturi, diffuser , orifice plate
    and convert values to SI units and caculate the diffrence in head pressure"""
    pizo_data = np.array(data[:,[fist_pizo,second_pizo]])
    pizodata = pizo_data * cf_m
    head = np.array(pizodata[:,[0]])-np.array(pizodata[:,[1]])

    rotometer_flow_rate = np.array(data[:,[0]])


    elc_flow_data = (np.array(data[:,[7]])) * cf_ms
    elc_flow_rate = np.round(elc_flow_data, 6)



    # print(elc_flow_rate)
    #print(data)
    # print("Head pizometer",str(fist_pizo),"-pizometer",str(second_pizo))
    # print(pizodata)
    # print("\n")
    # print(head)
    # print('\n')
    return rotometer_flow_rate, head, elc_flow_rate


```

```python
def cal_pressure_diff(g, density, head):
    """ Caculates te change in pressure in [Pa g] from: density of water, gravity and the change in head"""
    pressure_diff = g * density * head
    return pressure_diff


def cal_reynolds_number(Q, D, v, A):
    """ caculate reynolds number: Re = uD/v to determine the flow characteristic."""
    U = Q / A
    #print(U)
    Re = np.round(((U*D)/v) ,2)
    return Re


def cal_cd_cofficient(Re_cal, Re_low_lim, Re_up_lim, c_d):
    """ compaires the caculated Reynalds number agenst the provided Reynalds
    numbers an assigns the approprate Cd cofficient"""
    # print("Re_low_lim", Re_low_lim)
    # print("Re_up_lim", Re_low_lim)
    # print("c_d", c_d, "\n")

    i = 0
    j = 0
    cd_cofficients = []
    for i in range(len(Re_cal)):
        for j in range(len(Re_low_lim)):
            if (Re_cal[i] > Re_low_lim[j]) and (Re_cal[i] <= Re_up_lim[j]):
                #print("Re", Re_cal[i], ">", Re_low_lim[j],  "&  Re", Re_cal[i], "<", Re_up_lim[j]  )
                cd_cofficients.append(c_d[j])
                break
    return cd_cofficients


def cal_flow_rate(cd_vals, A_1, A_2, density, p_diff):
    """ caculates the flow rate from fuge factor, area profiles, density and change in pressure to caculate
    the flow rate across the component  U = cd *A_2 *sqrt( (2 *p_diff) / (1- A_2**2 / A_1**2"""
    # print(cd_vals)
    # print(A_1)
    # print(A_2)
    # print(density)
    # print(p_diff)
    for cd_val in cd_vals:
        flow_rate =  cd_val * A_2 *np.sqrt((2 * np.array(p_diff)) /(density * ( 1 - ((A_2**2) / (A_1**2)))))
    #print(flow_rate)
    return flow_rate


def uncertainties_pressure_diff(venturi_head, piezo_error,density_error, gravity, density):
     """ caculate the uncertanties for the diffrence in pressure"""
     pressure_error = np.sqrt( (gravity * venturi_head * density_error)**2 + (density * gravity * piezo_error)**2 + (density * gravity * piezo_error)**2 )
     return pressure_error

```

```python

def Uncertanties_area(diameter, diameter_error):
    """ caculate unxcedrtanties for area"""
    area_error = (np.pi * diameter) / 2 * diameter_error
    return area_error

def Uncertanties_density(density_water, flow_rate):
        """ caculate the error of density"""
        density_uncertantity = (0.5 / density_water) * flow_rate
        return density_uncertantity

def Uncertanties_flow_area_1(A_1, A_2, flow_rate):
    """ caculate the area_1 uncertanty with respect to flow"""
    error_flow_area_1 = (1) / ( (1 / A_1**2 - 1 / A_2**2) * A_1**3) * flow_rate
    return error_flow_area_1


def Uncertanties_flow_area_2(A_1, A_2, flow_rate):
     """ caculate the area_2 uncertanty with respect to flow"""
     error_flow_area_2 = (1) / ( (1 / A_2**2 - 1 / A_1**2) * A_2**3) * flow_rate
     # print(flow_rate)
     # print(error_flow_area_2)
     return error_flow_area_2

def uncertanties__flow_calutation(flow_rate, pressure_diff, pressure_err, density_uncertantity, density_err, A_1_flow, A_2_flow, Area_1_err, Area_2_err):
    """caculate the error for the flow rate caculation"""
    error_flow_upper = np.sqrt( (((0.5 / pressure_diff) * flow_rate) * pressure_err)**2 + (density_uncertantity * density_err)**2 + ( A_1_flow * Area_1_err )**2 + (A_2_flow * Area_2_err )**2)
    error_flow_lower = np.sqrt( (((0.5 / pressure_diff) * flow_rate) * pressure_err)**2 + (density_uncertantity * density_err)**2 + ( A_1_flow * Area_1_err )**2 + (A_2_flow * Area_2_err )**2)
    return error_flow_upper, error_flow_lower


def error_rotameter_caculation(rotometer_flow_rate, rotameter_error):
    """ caculate the uncertanties for the rotameter readings"""
    rotameter_uncertanty = rotameter_error / rotometer_flow_rate
    return rotameter_uncertanty


''' plot_flow_rates(venturi_flow_rate, orifice_flow_rate, flow_rate, error_flow_venturi_upper, error_flow_orifice_upper)'''
def plot_flow_rates(venturi_flow_rate, orifice_flow_rate, flow_rate, error_flow_venturi, error_flow_orifice):
    """ plot the ventri and orifice flow rates agenst the electronic flow meter flow rate mesaruments
    with an additional 1 to 1 line)"""
    elc_flow_rate = np.array(flow_rate)

    axes = plt.axes()
    axes.scatter(elc_flow_rate, venturi_flow_rate, s=1, marker=".", color="blue", label="Venturi")
    axes.scatter(elc_flow_rate, orifice_flow_rate, s=1,  marker=".", color="red", label="Orifice plate")
    axes.plot(np.arange(0, 0.0006, 0.00005),np.arange(0, 0.0006, 0.00005), color="black", linewidth=0.75, label="1-1 line")
    axes.legend()

    # print("error_flow_venturi\n", error_flow_venturi)
    # print(error_flow_venturi[:, 0])

    # # print(venturi_flow_rate)

    axes.errorbar(elc_flow_rate, venturi_flow_rate[:, 0], yerr = (error_flow_venturi[:, 0]), fmt='.', color="blue", label="Venturi", capsize=5)
    axes.errorbar(elc_flow_rate,  orifice_flow_rate[:, 0], yerr = (error_flow_orifice[:, 0]), fmt='.', color="red", label="Orifice", capsize=5)
```

```python

    x_labels = np.arange(0, 0.0006, 0.00005)
    x_labels_5_sf = [f"{val:.5f}" for val in x_labels]
    axes.set_xticks(np.arange(0, 0.0006, 0.00005))
    axes.set_xticklabels(x_labels_5_sf, rotation=(20), fontsize=10)
    axes.set_xlabel(r"Flow rate from electronic flow meter [m$^3$/s]", fontsize=10)

    axes.set_yticks(np.arange(0, 0.0008, 0.00005))
    y_labels = np.arange(0, 0.0008, 0.00005)
    y_labels_5_sf = [f"{val:.5f}" for val in y_labels]
    axes.set_yticklabels(y_labels_5_sf, fontsize=10)
    axes.set_ylabel(r"Flow rate from pressure drop [m$^3$/s]", fontsize=10)
    axes.grid()
    plt.show()


def plot_rotameter_elc_flow_rates(rotometer_flow_rate,flow_rate, cf_m3s, rotameter_uncertanty):
    """ plot electronic flow rates agenst the rotameter flow rates"""
    # print(rotometer_flow_rate)
    rotameter_SI = np.array(rotometer_flow_rate) * cf_m3s
    # print(rotameter_SI)
    # print(flow_rate)
    flow_rate = np.ravel(flow_rate)
    rotameter = np.ravel(rotameter_SI)

    axes = plt.axes()
    axes.scatter(flow_rate, rotameter, marker=".", color="blue", label="flow meter Vs rotameter")

    axes.errorbar(flow_rate, rotameter_SI[:, 0], yerr = (rotameter_uncertanty[:, 0]), fmt='.', color="blue", capsize=5)

    slope, intercept, r_value, p_value, std_err = stats.linregress(flow_rate, rotameter)
    r_squared = r_value**2


    axes.axline(xy1=(0, intercept), slope=slope, color="red", linestyle="--", label=f"Linear regression: $R^2 = {r_squared:.3f}$")
    axes.legend()

    x_labels = np.arange(0, 0.0006, 0.00005)
    x_labels_5_sf = [f"{val:.5f}" for val in x_labels]
    axes.set_xticks(np.arange(0, 0.0006, 0.00005))
    axes.set_xticklabels(x_labels_5_sf, rotation=(20), fontsize=10)
    axes.set_xlabel(r"Flow rate from electronic flow meter [m$^3$/s]", fontsize=10)

    axes.set_yticks(np.arange(0, 0.0008, 0.00005))
    y_labels = np.arange(0, 0.0008, 0.00005)
    y_labels_5_sf = [f"{val:.5f}" for val in y_labels]
    axes.set_yticklabels(y_labels_5_sf, fontsize=10)
    axes.set_ylabel(r"Flow rate from rotameter [m$^3$/s]", fontsize=10)

    axes.grid()
    plt.show()
```

```python

def format_processed_data(venturi_flow_rate,
                          error_flow_venturi,
                          pressure_diff_venturi,
                          error_pressure_diff_venturi,
                          Re_venturi,orifice_flow_rate,
                          error_flow_orifice,
                          pressure_diff_orifice,
                          error_pressure_diff_orifice,
                          Re_orifice, flow_rate):
    """create a dictionaryfor processed data for the venturi and orifice plate:
        flow rate, flowrate uncertainty, pressure, pressure uncertainty,  Reynalds number"""
    #delta_P = "\u0394P"
    #plus_minus = "\u00B1"
    processed_data = {'Venturi Q [m\u00b3/s]':venturi_flow_rate[:, 0],
                      'Venturi Q error \u00B1 [m\u00b3/s]':error_flow_venturi[:, 0],
                      'Orifice Q [m\u00b3/s]':orifice_flow_rate[:, 0],
                      'Orifice Q error \u00B1 [m\u00b3/s]':error_flow_orifice[:, 0],
                      'Electronic flow rate [m\u00b3/s]':flow_rate[:, 0],
                      'Venturi \u0394P [Pa]':pressure_diff_venturi[:, 0],
                      'Error venturi \u00B1\u0394P [Pa]':error_pressure_diff_venturi[:, 0],
                      'Orifice \u0394P [Pa]':pressure_diff_orifice[:, 0],
                      'Error orifice \u00B1\u0394P [Pa]':error_pressure_diff_orifice[:, 0],
                      'Reynalds number venturi':Re_venturi[:, 0],
                      'Reynalds number orifice':Re_orifice[:, 0]}

    #print(processed_data)
    return processed_data

def export_data_to_exel(data):
    """ export dictionary as to make an exel spread sheet"""
    df = pd.DataFrame(data)
    # Export the DataFrame to an Excel file
    df.to_excel("processed_data.xlsx", index=False)  # Save to file without including row indices

    print("Data exported successfully to 'processed_data.xlsx'")

```

```python
def main():
    data_set1 = "LabA_data1.csv"
    data_set2 = "labA_data2.csv"
    data_set3 = "LabA_data3.csv"

    cf_M = 0.001 # conversion factor mm to M
    cf_ms = 1.6666667*10**-5 # conversion factor from liters per minute to meters per second
    cf_m3s = 1*10**-6 # conversion factor from mm/min to m^3 / s

    gravity = 9.81  # gravity in m/s^2
    kinematic_v_h20 = 1.007*10**-6 # kinematic velocity of water at 20 degrees celcius

    d_1_venturi = 0.026     # diameter in meters for the pipe upstream of venturi throat
    d_2_venturi = 0.016     # diameter in meters for the venturi throat
    A_1_venturi = (np.pi * d_1_venturi**2) / 4
    A_2_venturi = (np.pi * d_2_venturi**2) / 4

    d_1_orifice = 0.0519 #diameter in meters for the pipe down stream of orifice
    d_2_orifice =0.02 #diameter in meters of the orifice plate hole
    A_1_orifice = (np.pi * d_1_orifice ** 2) / 4
    A_2_orifice = (np.pi * d_2_orifice ** 2) / 4


    Re_low_lim = [1, 75000, 150000, 250000, 400000, 1000000, 2000000]  # lower threshold values of reynalds number
    Re_up_lim = [75000, 150000, 250000, 400000, 1000000, 2000000, 100000000] # upper threshold values of reynalds number
    c_d  = [0.97, 0.977, 0.992, 0.998, 0.995, 1.00, 1.01]   #fuge factor cofficient which corospond to the a specifice range of reynald numbers

    c_d_orifice = np.full(9, 0.61) # from (D_2 / D_1)  and ( renalds number < 10^4 ) determines a c_d orifice value of 0.61 from 314lab manual


    pizo_1 = 1  # venturi pizometer
    pizo_2 = 2  # venturi pizometer
    pizo_3 = 3  # diffuser pizometer
    pizo_4 = 4  # diffuser pizometer
    pizo_5 = 5  # orifice plate pizometer
    pizo_6 = 6  # orifice plate pizometer


    piezo_error = 0.0005 # mesurment uncertainty for pizometers [m]
    density_error = 0.25  # change in temperature of 1 degree celcius changes density of water.
    diameter_error = 0.0005 # error in mesurment from mesuring diameters.
    rotameter_error = 0.0005 # error in mesuring rotameter


    data_1, data_2, data_3 = load_data(data_set1, data_set2, data_set3)

```

```python

    """-----------------------------------------------------------------------------"""
    """ uncomment the following blocks of code to see the results of test1, test2, test3"""

    # rotometer_flow_rate, venturi_head, flow_rate =  process_data(data_1, cf_M, pizo_1, pizo_2, cf_ms)
    # rotometer_flow_rate, diffuser_head, flow_rate = process_data(data_1, cf_M, pizo_3, pizo_4, cf_ms)
    # rotometer_flow_rate, orifice_head, flow_rate = process_data(data_1, cf_M, pizo_5, pizo_6, cf_ms)
    # density_water = 998.2 # density of water at 20 degrees celcius [kg/m^3]


    # rotometer_flow_rate ,venturi_head, flow_rate =  process_data(data_2, cf_M, pizo_1, pizo_2, cf_ms)
    # rotometer_flow_rate, diffuser_head, flow_rate = process_data(data_2, cf_M, pizo_3, pizo_4, cf_ms)
    # rotometer_flow_rate, orifice_head, flow_rate = process_data(data_2, cf_M, pizo_5, pizo_6, cf_ms)
    # density_water = 998.45 # density of water at 21 degrees celcius [kg/m^3]


    rotometer_flow_rate, venturi_head, flow_rate =  process_data(data_3, cf_M, pizo_1, pizo_2, cf_ms)
    rotometer_flow_rate, diffuser_head, flow_rate = process_data(data_3, cf_M, pizo_3, pizo_4, cf_ms)
    rotometer_flow_rate, orifice_head, flow_rate = process_data(data_3, cf_M, pizo_5, pizo_6, cf_ms)
    density_water = 998.575 # density of water at 22 degrees celcius [kg/m^3]
    """-----------------------------------------------------------------------------"""

    pressure_diff_venturi = cal_pressure_diff(gravity, density_water, venturi_head)
    pressure_diff_diffuser = cal_pressure_diff(gravity, density_water, diffuser_head)
    pressure_diff_orifice  = cal_pressure_diff(gravity, density_water, orifice_head)


    Re_venturi = cal_reynolds_number(flow_rate, d_2_venturi, kinematic_v_h20, A_2_venturi)
    Re_orifice = cal_reynolds_number(flow_rate, d_2_orifice, kinematic_v_h20, A_2_orifice)


    venturi_cd = cal_cd_cofficient(Re_venturi, Re_low_lim, Re_up_lim, c_d)
    orifice_cd = cal_cd_cofficient(Re_orifice, Re_low_lim, Re_up_lim, c_d)


    venturi_flow_rate = cal_flow_rate(venturi_cd, A_1_venturi, A_2_venturi, density_water, pressure_diff_venturi)
    orifice_flow_rate = cal_flow_rate( c_d_orifice , A_1_orifice, A_2_orifice , density_water, pressure_diff_orifice)


    density_uncertantity = Uncertanties_density(density_water, flow_rate)

    error_pressure_diff_venturi = uncertainties_pressure_diff(venturi_head, piezo_error, density_error, gravity, density_water)
    error_area_1_venturi = Uncertanties_area(d_1_venturi, diameter_error)
    error_area_2_venturi = Uncertanties_area(d_2_venturi, diameter_error)

    error_pressure_diff_orifice = uncertainties_pressure_diff(orifice_head, piezo_error, density_error, gravity, density_water)
    error_area_1_orifice = Uncertanties_area( d_1_orifice, diameter_error)
    error_area_2_orifice = Uncertanties_area(d_2_orifice , diameter_error)

    flow_area_1_error_venturi = Uncertanties_flow_area_1(A_1_venturi, A_2_venturi, flow_rate)
    flow_area_1_error_orifice = Uncertanties_flow_area_1(A_1_orifice, A_2_orifice, flow_rate)

    flow_area_2_error_venturi = Uncertanties_flow_area_2(A_1_venturi, A_2_venturi, flow_rate)
    flow_area_2_error_orifice = Uncertanties_flow_area_2(A_1_orifice, A_2_orifice, flow_rate)

```

```python

    """uncertanties__flow_calutation(flow_rate, pressure_diff, pressure_err, density_uncertantity, density_err, A_1, A_2, Area_1_err, Area_2_err):"""
    error_flow_venturi_upper, error_flow_venturi_lower = uncertanties__flow_calutation(venturi_flow_rate, pressure_diff_venturi,
                                                                                        error_pressure_diff_venturi,
                                                                                        density_uncertantity,
                                                                                        density_error, flow_area_1_error_venturi,
                                                                                        flow_area_2_error_venturi,
                                                                                        error_area_1_venturi,
                                                                                        error_area_2_venturi)
    error_flow_orifice_upper, error_flow_orifice_lower = uncertanties__flow_calutation(orifice_flow_rate,
                                                                                        pressure_diff_orifice,
                                                                                        error_pressure_diff_orifice,
                                                                                        density_uncertantity, density_error,
                                                                                        flow_area_1_error_orifice,
                                                                                        flow_area_2_error_orifice,
                                                                                        error_area_1_orifice,
                                                                                        Serror_area_2_orifice)

    rotameter_uncertanty = error_rotameter_caculation(rotometer_flow_rate, rotameter_error)

    plot_flow_rates(venturi_flow_rate, orifice_flow_rate, flow_rate, error_flow_venturi_upper, error_flow_orifice_upper, )
    plot_rotameter_elc_flow_rates(rotometer_flow_rate, flow_rate, cf_m3s, rotameter_uncertanty)


    data_to_export = format_processed_data(venturi_flow_rate, error_flow_venturi_upper,
                                           pressure_diff_venturi,
                                           error_pressure_diff_venturi,
                                           Re_venturi,
                                           orifice_flow_rate,
                                           error_flow_orifice_upper,
                                           pressure_diff_orifice,
                                           error_pressure_diff_orifice,
                                           Re_orifice, flow_rate)

    export_data_to_exel(data_to_export)

```