lab1

February 16, 2026

1 Lab 1: Characterizing the Atmosphere

[312]:
```
import numpy as np
import matplotlib.pyplot as plt
```

1.0.1 Importing Data

[313]:
```
daksh = { "name": "Daksh",
        "f0": "Data/floor0/daksh_floor0.csv",
        "f1": "Data/floor1/daksh_floor1.csv",
        "f2": "Data/floor2/daksh_floor2.csv",
        "f3": "Data/floor3/daksh_floor3.csv",
        "f4": "Data/floor4/daksh_floor4.csv",
        "task1": "Data/table/pressure_daksh.csv",}

clarisse = { "name": "Clarisse",
        "f0": "Data/floor0/clarisse_floor0.csv",
        "f1": "Data/floor1/clarisse_floor1.csv",
        "f2": "Data/floor2/clarisse_floor2.csv",
        "f3": "Data/floor3/clarisse_floor3.csv",
        "f4": "Data/floor4/clarisse_floor4.csv",
        "sensor_stability": "Data/table/pressure_clarisse.csv",
        "task1": "Data/table/pressure_clarisse.csv"}

tim = { "name": "Tim",
        "f0": "Data/floor0/tim_floor0.csv",
        "f1": "Data/floor1/tim_floor1.csv",
        "f2": "Data/floor2/tim_floor2.csv",
        "f3": "Data/floor3/tim_floor3.csv",
        "f4": "Data/floor4/tim_floor4.csv",
        "task1": "Data/table/pressure_tim.csv"}

joel = { "name": "Joel",
        "f0": "Data/floor0/joel_floor0.csv",
        "f1": "Data/floor1/joel_floor1.csv",
        "f2": "Data/floor2/joel_floor2.csv",
        "f3": "Data/floor3/joel_floor3.csv",
```

```
        "f4": "Data/floor4/joel_floor4.csv",
        "task1": "Data/table/pressure_joel.csv"}

names = [daksh, clarisse, tim, joel]
heights = 395*np.array([0, 1, 2, 3, 4])

def get_data(filePath):
    data = np.loadtxt(filePath, delimiter=",", skiprows=1)
    pp = data[:, 1]
    return pp

dakshData0 = get_data(daksh["f0"])
clarisseData0 = get_data(clarisse["f0"])
timData0 = get_data(tim["f0"])
joelData0 = get_data(joel["f0"])

dakshData1 = get_data(daksh["f1"])
clarisseData1 = get_data(clarisse["f1"])
timData1 = get_data(tim["f1"])
joelData1 = get_data(joel["f1"])

dakshData2 = get_data(daksh["f2"])
clarisseData2 = get_data(clarisse["f2"])
timData2 = get_data(tim["f2"])
joelData2 = get_data(joel["f2"])

dakshData3 = get_data(daksh["f3"])
clarisseData3 = get_data(clarisse["f3"])
timData3 = get_data(tim["f3"])
joelData3 = get_data(joel["f3"])

dakshData4 = get_data(daksh["f4"])
clarisseData4 = get_data(clarisse["f4"])
timData4 = get_data(tim["f4"])
joelData4 = get_data(joel["f4"])

dakshTask1 = get_data(daksh["task1"])
clarisseTask1 = get_data(clarisse["task1"])
timTask1 = get_data(tim["task1"])
joelTask1 = get_data(joel["task1"])
```

1.1 Task 1: Determining Accuracy and Precision of Pressure Sensors

-
-

1.1.1 Part 1: Accuracy of Barometers

[333]:
```
task1Data = [dakshTask1, clarisseTask1, timTask1, joelTask1]

dakshTask1Mean = np.mean(dakshTask1)
clarisseTask1Mean = np.mean(clarisseTask1)
timTask1Mean = np.mean(timTask1)
joelTask1Mean = np.mean(joelTask1)

dakshTask1Std = np.std(dakshTask1)
clarisseTask1Std = np.std(clarisseTask1)
timTask1Std = np.std(timTask1)
joelTask1Std = np.std(joelTask1)

groupTask1Means = [dakshTask1Mean, clarisseTask1Mean, timTask1Mean,␣
 ↪joelTask1Mean]
groupTask1Mean = np.mean(groupTask1Means)
groupTask1Stds = [dakshTask1Std, clarisseTask1Std, timTask1Std, joelTask1Std]

dakshRelativeAccuracy = abs(dakshTask1Mean - groupTask1Mean)
clarisseRelativeAccuracy = abs(clarisseTask1Mean - groupTask1Mean)
timRelativeAccuracy = abs(timTask1Mean - groupTask1Mean)
joelRelativeAccuracy = abs(joelTask1Mean - groupTask1Mean)

groupRelativeAccuracies = [dakshRelativeAccuracy, clarisseRelativeAccuracy,␣
 ↪timRelativeAccuracy, joelRelativeAccuracy]
groupTask1Stds = [dakshTask1Std, clarisseTask1Std, timTask1Std, joelTask1Std]

print("Relative accuracies of each group member:")
i = 0
for rel in groupRelativeAccuracies:
    print(f"{names[i]['name']}: {rel:.4f} hPa")
    i += 1

print("\nStandard deviations of each group member:")
j = 0
for std in groupTask1Stds:
    print(f"{names[j]['name']}: {std:.4f} hPa")
    j += 1

print("\nVsariation of difference between measurements of two group members:")
for i in range(len(groupTask1Means)):
    for j in range(i+1, len(groupTask1Means)):
        diff = np.std(task1Data[i] - task1Data[j])
```

```
        print(f"Difference between {names[i]['name']} and {names[j]['name']}:␣
↪{diff:.4f} hPa")
```

```
Relative accuracies of each group member:
Daksh: 0.0578 hPa
Clarisse: 0.0622 hPa
Tim: 0.0535 hPa
Joel: 0.0578 hPa

Standard deviations of each group member:
Daksh: 0.0017 hPa
Clarisse: 0.0039 hPa
Tim: 0.0067 hPa
Joel: 0.0017 hPa

Vsariation of difference between measurements of two group members:
Difference between Daksh and Clarisse: 0.0055 hPa
Difference between Daksh and Tim: 0.0054 hPa
Difference between Daksh and Joel: 0.0000 hPa
Difference between Clarisse and Tim: 0.0104 hPa
Difference between Clarisse and Joel: 0.0055 hPa
Difference between Tim and Joel: 0.0054 hPa
```

1.2 Main Experiment: Measuring Pressure vs Height

1.2.1 Floor 0

Get means

[315]:
```
dakshMean0 = np.mean(dakshData0)
clarisseMean0 = np.mean(clarisseData0)
timMean0 = np.mean(timData0)
joelMean0 = np.mean(joelData0)

groupMean0 = np.mean([dakshMean0, clarisseMean0, timMean0, joelMean0])

print (f"Group Mean for floor 0: {groupMean0:.4f} hPa")
```

```
Group Mean for floor 0: 1012.2793 hPa
```

Get sample standard deviations

[316]:
```
dakshStd0 = np.std(dakshData0, ddof=1)
clarisseStd0 = np.std(clarisseData0, ddof=1)
timStd0 = np.std(timData0, ddof=1)
```

```
joelStd0 = np.std(joelData0, ddof=1)

groupStd0 = np.sqrt(np.mean([dakshStd0**2, clarisseStd0**2, timStd0**2,␣
  ↪joelStd0**2]))

print (f"Group Std Dev for floor 0: {groupStd0:.4f} hPa")
```

```
Group Std Dev for floor 0: 0.0086 hPa
```

1.2.2 Floor 1

Get means

[317]:
```
dakshMean1 = np.mean(dakshData1)
clarisseMean1 = np.mean(clarisseData1)
timMean1 = np.mean(timData1)
joelMean1 = np.mean(joelData1)

groupMean1 = np.mean([dakshMean1, clarisseMean1, timMean1, joelMean1])
print (f"Group Mean for floor 1: {groupMean1:.4f} hPa")
```

```
Group Mean for floor 1: 1011.5892 hPa
```

Get sample standard deviations

[318]:
```
dakshStd1 = np.std(dakshData1, ddof=1)
clarisseStd1 = np.std(clarisseData1, ddof=1)
timStd1 = np.std(timData1, ddof=1)
joelStd1 = np.std(joelData1, ddof=1)

groupStd1 = np.sqrt(np.mean([dakshStd1**2, clarisseStd1**2, timStd1**2,␣
  ↪joelStd1**2]))
print (f"Group Std Dev for floor 1: {groupStd1:.4f} hPa")
```

```
Group Std Dev for floor 1: 0.0101 hPa
```

1.2.3 Floor 2

Get means

[319]:
```
dakshMean2 = np.mean(dakshData2)
clarisseMean2 = np.mean(clarisseData2)
timMean2 = np.mean(timData2)
joelMean2 = np.mean(joelData2)

groupMean2 = np.mean([dakshMean2, clarisseMean2, timMean2, joelMean2])
print (f"Group Mean for floor 2: {groupMean2:.4f} hPa")
```

```
Group Mean for floor 2: 1010.9693 hPa
```

Get sample standard deviations

[320]:
```
dakshStd2 = np.std(dakshData2, ddof=1)
clarisseStd2 = np.std(clarisseData2, ddof=1)
timStd2 = np.std(timData2, ddof=1)
joelStd2 = np.std(joelData2, ddof=1)

groupStd2 = np.sqrt(np.mean([dakshStd2**2, clarisseStd2**2, timStd2**2,␣
 ↪joelStd2**2]))
print (f"Group Std Dev for floor 2: {groupStd2:.4f} hPa")
```

```
Group Std Dev for floor 2: 0.0137 hPa
```

1.2.4 Floor 3

Get means

[321]:
```
dakshMean3 = np.mean(dakshData3)
clarisseMean3 = np.mean(clarisseData3)
timMean3 = np.mean(timData3)
joelMean3 = np.mean(joelData3)

groupMean3 = np.mean([dakshMean3, clarisseMean3, timMean3, joelMean3])
print (f"Group Mean for floor 3: {groupMean3:.4f} hPa")
```

```
Group Mean for floor 3: 1010.4235 hPa
```

Get sample standard deviations

[322]:
```
dakshStd3 = np.std(dakshData3, ddof=1)
clarisseStd3 = np.std(clarisseData3, ddof=1)
timStd3 = np.std(timData3, ddof=1)
joelStd3 = np.std(joelData3, ddof=1)

groupStd3 = np.sqrt(np.mean([dakshStd3**2, clarisseStd3**2, timStd3**2,␣
 ↪joelStd3**2]))
print (f"Group Std Dev for floor 3: {groupStd3:.4f} hPa")
```

```
Group Std Dev for floor 3: 0.0113 hPa
```

1.2.5 Floor 4

Get mean

[323]:
```
dakshMean4 = np.mean(dakshData4)
clarisseMean4 = np.mean(clarisseData4)
timMean4 = np.mean(timData4)
joelMean4 = np.mean(joelData4)

groupMean4 = np.mean([dakshMean4, clarisseMean4, timMean4, joelMean4])
print (f"Group Mean for floor 4: {groupMean4:.4f} hPa")
```

```
Group Mean for floor 4: 1009.7764 hPa
```

Get sample standard deviations

[324]:
```
dakshStd4 = np.std(dakshData4, ddof=1)
clarisseStd4 = np.std(clarisseData4, ddof=1)
timStd4 = np.std(timData4, ddof=1)
joelStd4 = np.std(joelData4, ddof=1)

groupStd4 = np.sqrt(np.mean([dakshStd4**2, clarisseStd4**2, timStd4**2,␣
 ↪joelStd4**2]))
print (f"Group Std Dev for floor 4: {groupStd4:.4f} hPa")
```

```
Group Std Dev for floor 4: 0.0400 hPa
```

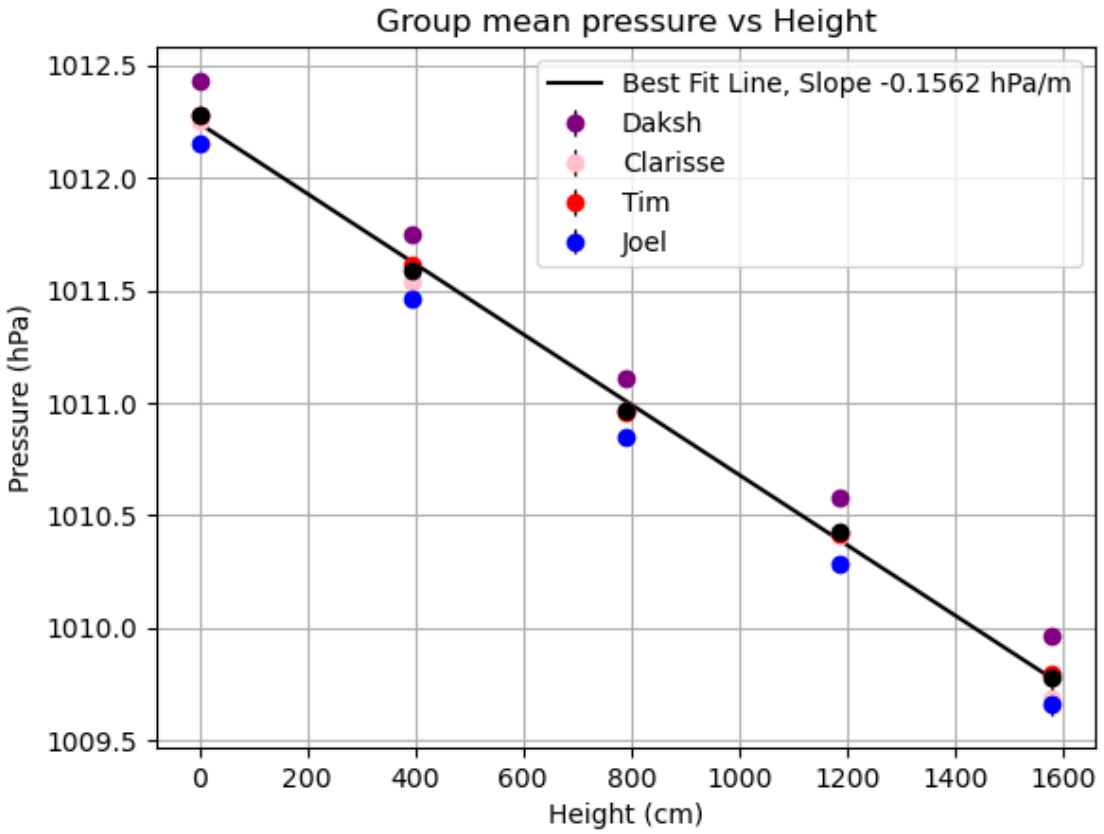
1.3 Graphing Group Pressure vs Height

[325]:
```
groupMeans = [groupMean0, groupMean1, groupMean2, groupMean3, groupMean4]
groupStds = [groupStd0, groupStd1, groupStd2, groupStd3, groupStd4]



dakshMeans = [dakshMean0, dakshMean1, dakshMean2, dakshMean3, dakshMean4]
clarisseMeans = [clarisseMean0, clarisseMean1, clarisseMean2, clarisseMean3,␣
 ↪clarisseMean4]
timMeans = [timMean0, timMean1, timMean2, timMean3, timMean4]
joelMeans = [joelMean0, joelMean1, joelMean2, joelMean3, joelMean4]

dakshStds = [dakshStd0, dakshStd1, dakshStd2, dakshStd3, dakshStd4]
clarisseStds = [clarisseStd0, clarisseStd1, clarisseStd2, clarisseStd3,␣
 ↪clarisseStd4]
timStds = [timStd0, timStd1, timStd2, timStd3, timStd4]
joelStds = [joelStd0, joelStd1, joelStd2, joelStd3, joelStd4]

daksh["means"] = dakshMeans
clarisse["means"] = clarisseMeans
tim["means"] = timMeans
joel["means"] = joelMeans

daksh["stds"] = dakshStds
clarisse["stds"] = clarisseStds
tim["stds"] = timStds
joel["stds"] = joelStds

daksh['color'] = 'purple'
clarisse['color'] = 'pink'
tim['color'] = 'red'
joel['color'] = 'blue'
```

[326]:
```
# 1. Extract Slope from Group Data
heights_m = heights / 100  # Convert cm to m
```

```
slope, intercept = np.polyfit(heights_m, groupMeans, 1)
g = 9.81
rho_experimental = - (slope * 100) / g  # Convert hPa/m to Pa/m


# 2. Theoretical Density
T_kelvin = 293.15
P0_pa = groupMeans[0] * 100
M = 0.02897 # kg/mol
R = 8.314
rho_theoretical = (P0_pa * M) / (R * T_kelvin)


for person in names:
    plt.errorbar(
        heights,
        person["means"],
        yerr=person["stds"],
        fmt='o',
        color=person['color'],
        ecolor='black',
        elinewidth=1,
        label=person["name"]
    )

plt.errorbar(
    heights,
    groupMeans,
    yerr=groupStds,
    fmt='o',
    color='black',
    ecolor='red',
    elinewidth=1,
)

t_fit = np.linspace(min(heights), max(heights), 100)
p_fit = np.polyval(np.polyfit(heights, groupMeans, 1), t_fit)


plt.title("Group mean pressure vs Height")
plt.xlabel("Height (cm)")
plt.ylabel("Pressure (hPa)")
plt.plot(t_fit, p_fit, color='black', label=f'Best Fit Line, Slope {slope:.4f}␣
 ↪hPa/m')
plt.legend()
plt.grid()
plt.show()
```

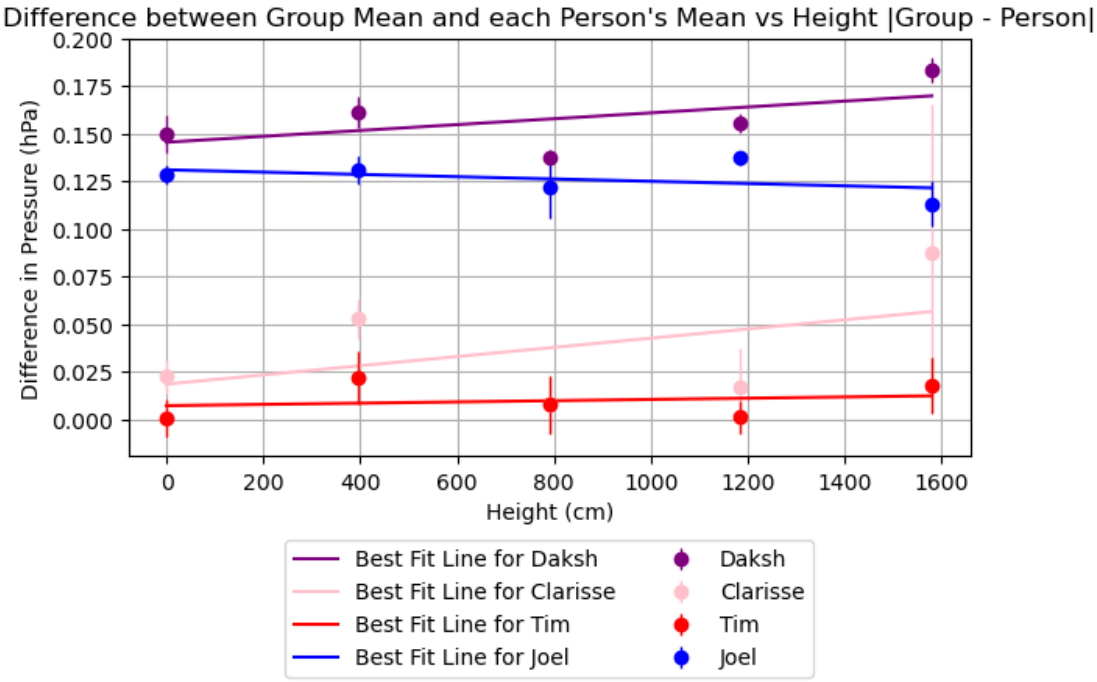
```
print(f"Experimental Slope: {slope:.4f} hPa/m")
print(f"Experimental Air Density (|slope| in kg/m^3): {rho_experimental:.4f} kg/
 ↪m^3")
print(f"Theoretical Air Density: {rho_theoretical:.4f} kg/m^3")
print(f"Percent Error: {abs((rho_experimental - rho_theoretical)/
 ↪rho_theoretical)*100:.2f}%")
```




```
Experimental Slope: -0.1562 hPa/m
Experimental Air Density (|slope| in kg/m^3): 1.5927 kg/m^3
Theoretical Air Density: 1.2032 kg/m^3
Percent Error: 32.37%
```

[327]:
```
for person in names:
    groupMinusPerson =  np.abs([groupMean0 - person["means"][0], groupMean1 -␣
 ↪person["means"][1], groupMean2 - person["means"][2], groupMean3 -␣
 ↪person["means"][3], groupMean4 - person["means"][4]])
    plt.errorbar(
        heights,
```

```
        groupMinusPerson,
        yerr=person["stds"],
        fmt='o',
        color=person['color'],
        ecolor=person['color'],
        elinewidth=1,
        label=person['name']
    )
    t_fit = np.linspace(min(heights), max(heights), 100)
    p_fit = np.polyval(np.polyfit(heights, groupMinusPerson, 1), t_fit)
    plt.plot(t_fit, p_fit, color=person['color'], label=f'Best Fit Line for␣
 ↪{person["name"]}')


plt.title(f"Difference between Group Mean and each Person's Mean vs Height␣
 ↪|Group - Person|")
plt.xlabel("Height (cm)")
plt.ylabel("Difference in Pressure (hPa)")
plt.grid()
plt.legend(bbox_to_anchor=(0.5, -0.2), ncol=2, loc='upper center',␣
 ↪borderaxespad=0.)
plt.tight_layout()
plt.show()
```



[ ]:
```
# 1. Calculate the average pressure drop per floor
total_pressure_drop = groupMeans[0] - groupMeans[-1] # Positive value
avg_drop_per_floor = total_pressure_drop / 4

print(f"Average Pressure Drop per Floor: {avg_drop_per_floor:.4f} hPa")

# 2. Calculate the theoretical slope (Standard Atmosphere)
rho_standard = 1.204 # kg/m^3 at 20C
g = 9.81
theoretical_slope = (rho_standard * g) / 100 # Convert Pa/m to hPa/m
print(f"Theoretical Slope: {theoretical_slope:.4f} hPa/m")

# 3. Calculate the Real Height
calculated_height_per_floor = avg_drop_per_floor / theoretical_slope
print(f"Calculated Height per Floor: {calculated_height_per_floor:.2f} meters")
```

```
Average Pressure Drop per Floor: 0.6257 hPa
Theoretical Slope: 0.1181 hPa/m
Calculated Height per Floor: 5.30 meters
```

[329]:
```
rho_standard = 1.204  # kg/m^3 (Standard air density at 20C)
g = 9.81              # m/s^2

# 1. Calculate the expected pressure drop per meter
# P = rho * g * h  =>  dP/dh = rho * g
expected_gradient_Pa_m = rho_standard * g
expected_gradient_hPa_m = expected_gradient_Pa_m / 100  # Convert to hPa/m

print(f"Standard Gradient: {expected_gradient_hPa_m:.4f} hPa/m")

# 2. Get Measured Pressure Drop per Floor

total_drop = groupMeans[0] - groupMeans[-1] # Pressure at bottom - top
avg_drop_per_floor = total_drop / 4

print(f"Measured Drop per Floor: {avg_drop_per_floor:.4f} hPa")

# 3. Solve for the Real Height
# Drop_per_Floor = Gradient_per_Meter * Height_Meters
real_floor_height = avg_drop_per_floor / expected_gradient_hPa_m

print(f"\nConclusion:")
print(f"The pressure data indicates the average floor height is:␣
 ↪{real_floor_height:.2f} meters.")
```

```
Standard Gradient: 0.1181 hPa/m
Measured Drop per Floor: 0.6257 hPa
```

```
Conclusion:
The pressure data indicates the average floor height is: 5.30 meters.
```

Although we initially estimated the floor height to be 4m, the barometric data indicates a vertical distance of 5.3 meters per floor.

The barometric data indicates an actual average floor height of 5.30 meters. Although the floors were initially estimated at 4m, the measured pressure drop per floor (0.6257 hPa) indicates a significantly greater vertical distance. While factors such as people walking past the sensors (creating dynamic pressure noise) or building ventilation (creating non-static gradients) could influence the results, the most robust physical explanation for the consistent pressure drop is that the architectural height of each floor is approximately 5.30 meters.

[ ]: