DIRECTIONS: *Please write in complete sentences and use professional, scientific language. If calculations are required, show and/or explain how they are done. Hand-written submissions are not the expected format. Please plan to type and format your responses, including mathematics formulations and plots, figures, diagrams, etc. Your submission should appear professional and polished. A LaTeX template (like the one used to create this document) is provided on canvas if desired. Be aware of student integrity policies, but you are welcome to seek help and discuss these assignments with your classmates. The solutions, however, MUST be your own, including your code. You are expected to include your code as part of your homework submission.*

PROBLEM 1:

Diurnal evolution of the boundary layer and the different zones that can be identified within it were covered in the early parts of this course. In general, materials covered in class focused on idealized conditions. It is not always as straightforward to identify characteristics of the boundary layer in observed datasets.

For this problem, you will be analyzing radiosonde data from a weather balloon launched by NOAA-NWS Norman on 8 August 2020, valid for 0000UTC on 9 August. Write code to visualize these data in a way that allows you to identify any of the labeled features from the 'Stull' diagram (see Fig 1). Include support and reasoning for the labels you choose. You are free to seek out additional resources or datasets to support your claims; these must be documented. Minimum requirement: note/listing in your solution set.

The data in netCDF format and a basic 'starter' script are available on the HW 1 assignment page on canvas.

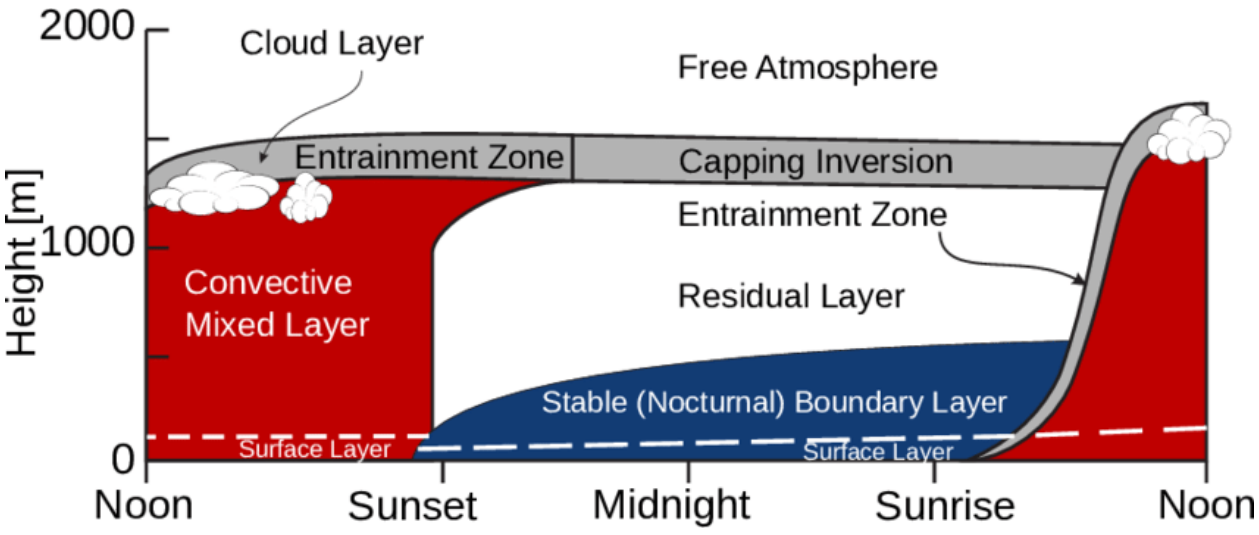

Figure 1: Conceptual diagram showing the temporal evolution of the boundary layer, adapted from Stull (1988).

PROBLEM 2:

After completing a solution for problem 1, explore publicly available datasets to find at least one other example where observations allow you to identify boundary-layer features like those we have covered in class materials. You could also choose to describe features related to the principles covered in the introduction to turbulence and thermodynamics sections.

You can use radiosonde data, but you are not limited to radiosonde data. Using the same code you started in problem 1, visualize the data you find. You can use similar or different techniques. As in problem 1, include support and reasoning for why you chose these data and what boundary-layer features or characteristics you identified.

For radiosondes, University of Wyoming (https://weather.uwyo.edu/upperair/sounding.html) and NOAA IGRA (https://www.ncei.noaa.gov/products/weather-balloon/integrated-global-radiosonde-archive) are easy to access archives. There are many tools available online to pull sounding data. Elizabeth Smith has a simple one developed as part of a boundary-layer height detection project here: https://github.com/eeeeelizzzzz/SondePull/blob/main/SondePull.py

Doing some research for datasets and tools can be a great way to expand your toolbox and develop your skill sets. This type of task is aligned with this course's 'self-guided learning' outcomes.

PROBLEM 3:

Reflect on the importance of the boundary layer to your research/interests/aspirations as a meteorologist/scientist. How does it intersect with other parts of the meteorology or general science community? Why does studying and understanding the lower-atmosphere matter? Provide examples NOT discussed in lecture.

PROBLEM 4:

Under different boundary layer conditions, plumes emitted from stacks (e.g., cooling towers, smoke stacks, etc.) can exhibit varying behavior. Figure 2 shows three plume dispersion patterns in panels (a)-(c). Note that these dispersion plumes are shown in x-z cross-sections and could have a y-direction component as well!

(a) Determine the type of plume behavior shown in each panel.

(b) Describe the atmospheric stability conditions that could support each type of plume behavior. Why does this condition promote the matched plume behavior?

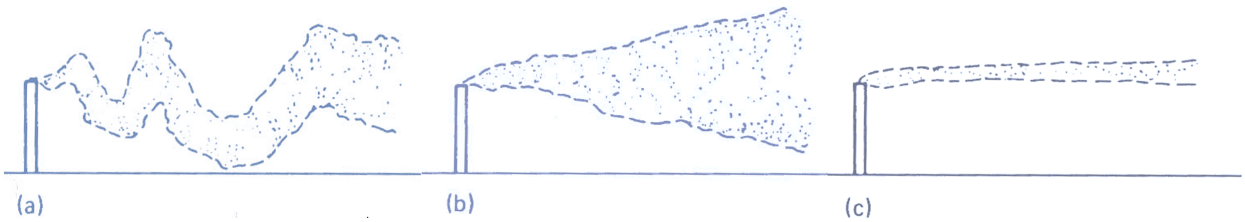

Figure 2: Various plume behaviors in panels a), b), and c).

PROBLEM 5:

In the Boundary-layer Thermodynamics section, we discussed stability in the atmosphere. Briefly describe in your own words the ideas or concepts behind local and non-local stability. What makes these two stability frameworks different? Fig. 3 shows an example of an idealized profile of virtual potential temperature in Earth's atmosphere. Based on your understanding and the concepts you described, classify the atmospheric stability in the environment represented by this profile. Is it the same in local and non-local terms?

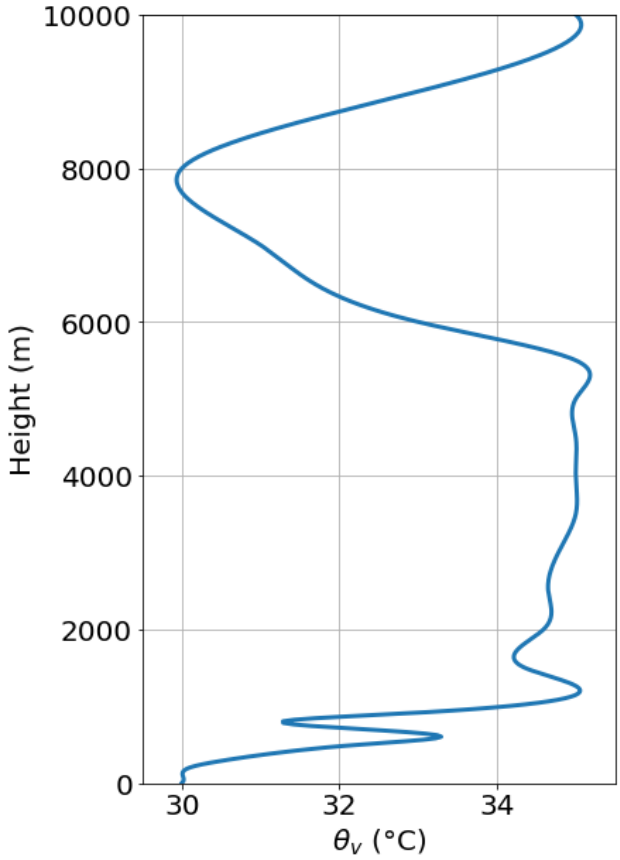

Figure 3: A profile diagram shows an idealized profile of virtual potential temperature

APPENDIX - Python code

```
###################
# Elizabeth Smith Code
# Example for Problem 5 Plot
# 1 Sept 2024
###################
from netCDF4 import Dataset
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d
import itertools
# A figure setting to make fonts big enough to read
plt.rcParams['font.size'] = 20

def average_duplicates(heights, temperatures):
    unique_heights = []
    averaged_temperatures = []
    for height, group in itertools.groupby(zip(heights, temperatures), key=lambda
        x: x[0]):
        unique_heights.append(height)
        temp_list = [temp for _, temp in group]
        averaged_temperatures.append(np.mean(temp_list))
    return np.array(unique_heights), np.array(averaged_temperatures)
    # we use this function to handle duplicate height values by averaging the
    # corresponding temperature values. If there are multiple temperature
    # values for the same height, this script averages these temperatures. This
    # ddress issues with interpolation functions like interp1d or
        make_interp_spline
    # that raise a ValueError when duplicate x-values are present, we need to
    # ensure the x-values are unique before passing them to these functions.
    # One approach is to average the duplicate temperatures/heights, which is
    # implemented in this function

# this is a manually generated list of heights and temperatures to build a
    profile
# feel free to play with the values and build your own in addition to this
    example from the homework
heights = [0,100,200,300,400,500,600,700,800,900,1000,1500,2000,2500,3000,3500,
    4000,4500,5000,5500,6000,7000,8000,9000,10000]
temperatures = [303-273, 303.01-273 ,303.1-273, 303.578-273, 304.278-273,
    305.278-273, 306.291-273, 305.278-273, 304.278-273, 305.75-273, 307.15-273,
    307.35-273, 307.593-273, 307.65-273, 307.78-273, 308-273, 308-273, 308-273,
    308-273, 308-273, 306-273, 304-273, 303-273, 306-273, 308-273 ]
```

```
# Average out duplicates with function above
heights, temperatures = average_duplicates(heights, temperatures)

# Sort heights and corresponding temperatures to ensure proper interpolation
sorted_indices = np.argsort(heights)
heights = heights[sorted_indices]
temperatures = temperatures[sorted_indices]

    # Interpolating to create smooth curves instead of a blocky one -- think about
    # if this is something you would want to do with REAL data vs idealized data
try:
    f = interp1d(heights, temperatures, kind='cubic', fill_value="extrapolate")
except ValueError:
    print("Error with cubic interpolation; trying linear interpolation instead.")
    f = interp1d(heights, temperatures, kind='linear', fill_value="extrapolate")

# Creating smooth height values
smooth_heights = np.linspace(min(heights), max(heights), 500)

# Plotting
 # Plot
plt.figure(figsize=(6, 10))
plt.plot(f(smooth_heights), smooth_heights, label="Temperature Profile", lw=3)

# the commented out lines below will plot the actual values (and legend) so you
# can see how the interpolation connects them
#plt.scatter(temperatures, heights, color='red', label="Input Points")
#plt.legend()

# check out how we include symbols like theta and a subscript in labels
plt.xlabel(r'$\theta _{v}$ (C)')
plt.ylabel('Height (m)')
plt.ylim(0,10000)
plt.xlim(29.5,35.5)
plt.grid(True)
plt.show()
```