Assignment3_py

November 16, 2024

1 Assignment 3

1.1 GEOS 300

Term 1 (Autumn 2024)

University of British Columbia

1.1.1 Instructor:

Marysa Laguë, Assistant Professor, Department of Geography

1.1.2 Preamble:

In this exercise you will use a 30-min data-set measured above an extensively flat cotton field near Kettleman City, CA, USA[1]. You will use two datasets:

df1 = wind200008021530.xls
df2 = turbulence200008021530.xls

The "wind" dataframe lists horizontal wind speeds u measured with cup-anemometers installed at six heights on a profile tower averaged over 30 minutes. Screen-level air temperature is also provided.

The "turbulence" dataframe contains longitudinal wind u, lateral wind v and vertical wind w measured every second over the same 30 minutes by a fast-response anemometer located at 6.4 m height.

For all questions assume neutral conditions and $z_d = 0$. Assume a pressure of 100 kPa.

Instructions: Please return your answers including all calculations, graphs and discussions in a well-structured report (PDF, either your jupyter notebook or a word/google doc saved as a pdf).

Label the report document with your name, your student number, the course and year.

Marks are indicated in square brackets. This assignment is worth 10% of your final grade.

[1]http://www.eol.ucar.edu/rtf/projects/ebex2000/

1.2 Import relevant packages:

[18]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import datetime
from datetime import datetime as dt
import time
import math
```

2 Load the datasets:

[19]:
```
### First, open the dataset:

# Import the data  - upload this file from Canvas and put it in the same folder␣
↪as your assignment.
data_file = 'wind200008021530.xls'

df1 = pd.read_excel(data_file,skiprows = 6,)

# a few post processing steps:

# 1. Remove units from variable names & rename variables
names_and_units = list(df1.columns.values)
just_names = [x.split(" ",1)[0] for x in names_and_units]

# 2. replace the column names with our new names that don't have units
df1.columns = just_names
df1 = df1.rename(columns={"Horizontal": "U"})

# 3. add a variable called logHeight:

df1['logHeight'] = np.log(df1['Height'])

df1.head()
```

[19]:

	Height	U	logHeight
0	0.95	1.54	-0.051293
1	1.55	1.83	0.438255
2	2.35	2.00	0.854415
3	3.72	2.22	1.313724
4	6.15	2.50	1.816452

[20]:
```
data_file = 'turbulence200008021530.xls'
```

```
dateparse = lambda x: dt.strptime(x, '%Y-%m-%d %H:%M:%S')

df2 = pd.read_excel(data_file,
                    # header = 0,
                    parse_dates=['Date'],
                    date_format='%Y-%m-%d %H:%M:%S',
                    index_col='Date')


# a few post processing steps:

# 1. add time from Date

df2['Time'] = df2.index.time

# 2. Remove units from variable names & rename variables
names_and_units = list(df2.columns.values)
just_names = [x.split(" ",1)[0] for x in names_and_units]

# 3. replace the column names with our new names that don't have units
df2.columns = just_names


# print out the top of the dataframe

df2.head()
```

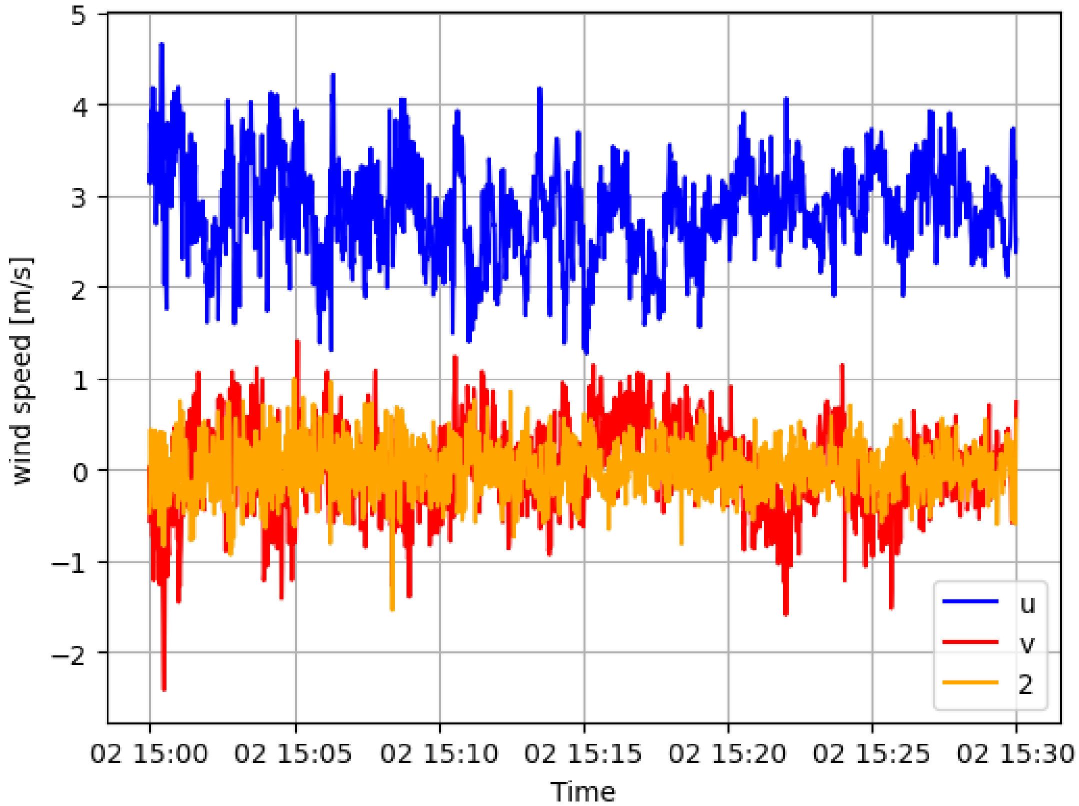
[20]:

```
                          u       v       w      Time
Date
2000-08-02 15:00:00  3.2174 -0.5631 -0.4740  15:00:00
2000-08-02 15:00:01  3.1313  0.0400  0.0050  15:00:01
2000-08-02 15:00:02  3.7852 -0.4075  0.4388  15:00:02
2000-08-02 15:00:03  3.6670 -0.2518  0.0259  15:00:03
2000-08-02 15:00:04  3.9281 -0.1403 -0.2484  15:00:04
```

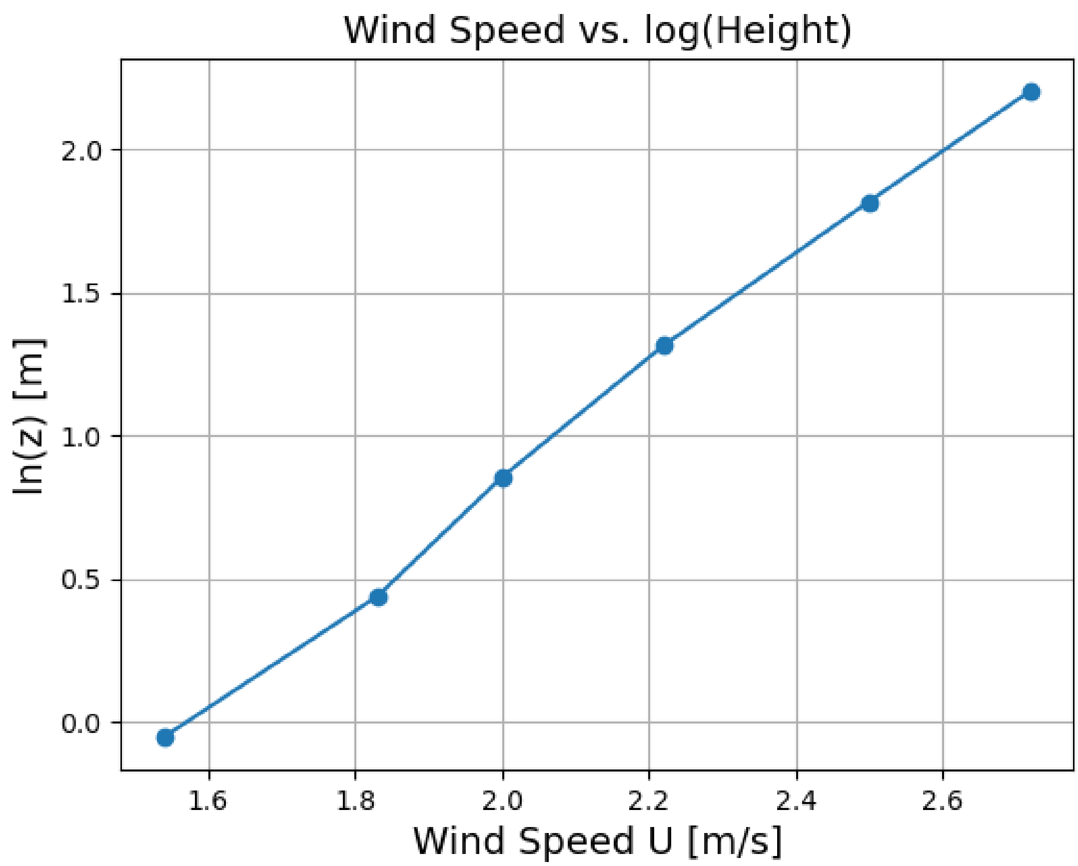
[21]:

```
# visualize the data:


plt.plot(df2.index,df2.u,color="blue",label='u')
plt.plot(df2.index,df2.v,color="red",label='v')
plt.plot(df2.index,df2.w,color="orange",label='2')
plt.legend()
plt.grid(':')
plt.xlabel('Time')
plt.ylabel('wind speed [m/s]')

plt.show()
```

```
plt.close()
```



[22]:
```
plt.plot(df1.U, df1.logHeight,marker='o')
plt.xlabel('Wind Speed U [m/s]',fontsize=14)
plt.ylabel('ln(z) [m]',fontsize=14)
plt.grid()
plt.title('Wind Speed vs. log(Height)',fontsize=14)

plt.show()
plt.close()
```



[ ]:

To estimate z_0, we need to calculate the y-intercept of the linear fit line that approximates the above. For a line $y = mx + b$, we have $x = u$, $y = ln(z)$, where m is the slope of the log-normal plot above, and $b = ln(z_0)$ is the intercept we're looking for. You can estimate this by hand by drawing the line back through the y axis on log-normal paper, or you can calculate the linear fit to the data. Both methods are fine for this assignment, and should give approximately similar answers. Answers within ± 0.01 m are acceptable.

Solving for the y-intercept via linear regression gives us $ln(z_0)$. To get z_0 itself, we have: $b = ln(z_0) \Rightarrow z_0 = e^b$

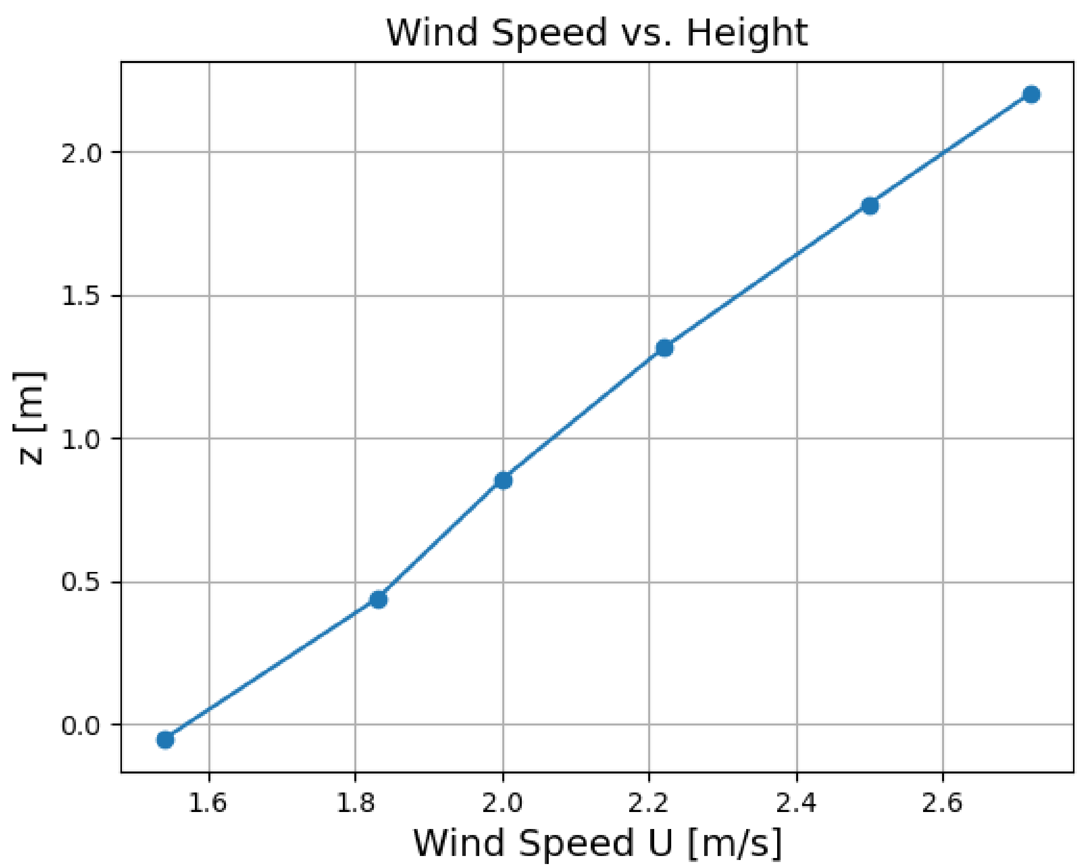
[23]:
```
# In python, we can do a simple linear fit using a 1-degree polynomial
# using the numpy package. You can also use fancier statistical modeling
# packages; they should all give approximately the same answer for this dataset.
↪

m,b = np.polyfit(df1.U, df1.logHeight, 1)

z0 = np.exp(b)
```

```
print('The y-intercept z0 = %1.3f'%z0)
```

The y-intercept z0 = 0.047

Solving for z_0 requires the use of $log(z)$ on the y axis. For demonstrational purposes, I'm also including the plot of u vs z, where you can see wind speed approaching zero near the surface (no-slip boundary condition), and wind speed increasing exponentially with height.

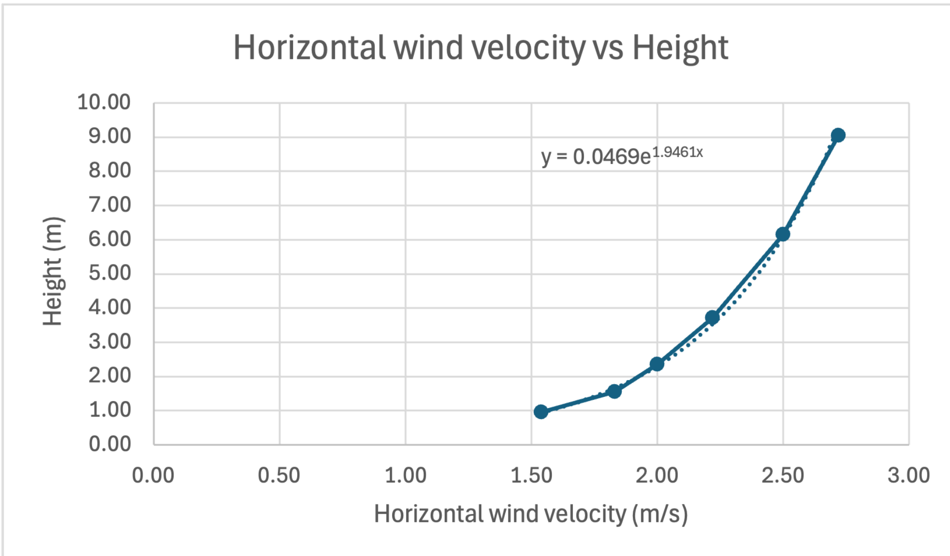
[24]:
```
plt.plot(df1.U, df1.logHeight,marker='o')
plt.xlabel('Wind Speed U [m/s]',fontsize=14)
plt.ylabel('z [m]',fontsize=14)
plt.grid()
plt.title('Wind Speed vs. Height',fontsize=14)

plt.show()
plt.close()
```



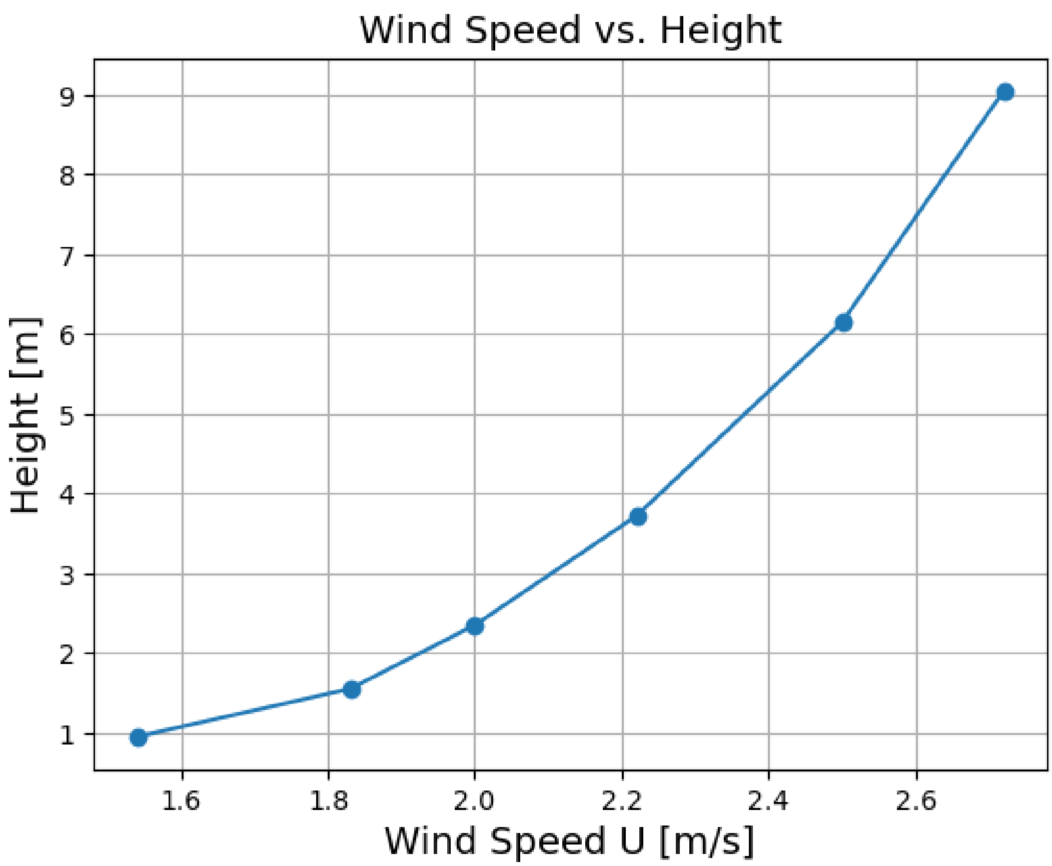
1. Did you use AI for this assignment? If so, how? [1] Rubric: [1] for any answer

No

2. z0 is the roughness length - the height at which a wind speed theoretically becomes zero in neutral atmospheric conditions. EsVmate z 0 graphically from all measured values of the wind profile in df1 (the wind data frame). You can either use a spreadsheet/software (e.g. R, python, excel) or the semi-logarithmic paper provided on Canvas. Note: If you solve this question using a semi-logarithmic paper, use a ruler and your graphical judgment (subjective) to create the best fit through the points. [6] Rubric: [2] approach [3] graph [1] answer for z 0



[25]:
```
plt.plot(df1.U, df1.Height,marker='o')
plt.xlabel('Wind Speed U [m/s]',fontsize=14)
plt.ylabel('Height [m]',fontsize=14)
plt.grid()
plt.title('Wind Speed vs. Height',fontsize=14)

plt.show()
plt.close()

print('z0 = ' +str(z0))
```




```
z0 = 0.046926188702080936
```

3. Based on the slope of the curve in QuesVon 2, calculate the friction velocity u∗. How would the wind profile in Q2 look different if u* were larger than what you calculated? [6] Rubric: [2] approach [1] u* value [3] discussion

The equation of the line is y = 0.0469e^1.9461x meaning the slope is 1.9461. Slope is equal to k/u*.

[26]:
```
Slope = 1.9461
k = 0.41
ustar = k/Slope

print('u* is ' + str(ustar))
```

```
u* is 0.21067776578798622
```

If u* was larger than the slope would be more flat meaning that the wind speed will be greater at lower height. This is because the slope is calculated by k/u*. Dividing by a bigger number will result in smaller y values (height).

4. Estimate the eddy diffusivities for momentum KM using the wind gradients u in the data

frame df1, between each layer:

(a) z= 0.95 and 1.55 m,
(b) z= 1.55 and 2.35 m,
(c) z = 2.35 and 3.72 m,
(d) z = 3.72 and 6.15 m, and
(e) z = 6.15 and 9.05 m. Assume an air density of ! = 1.131 kg/m 3. Hint: you need to calculate the Reynolds stress 0 Discuss how KM changes with height and explain why this happens. [6] rubric: [1.5] approach [1] values [1.5] how KM changes with height [2] discussion -1 for missing units.

$$K_M = \frac{\tau_0}{\rho_a}\frac{\Delta z}{\Delta \overline{u}} = u_*^2 \frac{\Delta z}{\Delta \overline{u}} \qquad \tau \approx \rho u_*^2$$

!

[27]:
```
z1 = 0.95
u1 = 1.54
z2 = 1.55
u2 = 1.83
z3 = 2.35
u3 = 2.00
z4 = 3.72
u4 = 2.22
z5 = 6.15
u5 = 2.5
z6 = 9.05
u6 = 2.72
pa = 1.131

KMa = (ustar**2)*((z2-z1)/(u2-u1))
KMb = (ustar**2)*((z3-z2)/(u3-u2))
KMc = (ustar**2)*((z4-z3)/(u4-u3))
KMd = (ustar**2)*((z5-z4)/(u5-u4))
KMe = (ustar**2)*((z6-z5)/(u6-u5))

print('Between the layers 0.95m and 1.55m the KM is '+str(KMa)+' m^2/s.')
print('Between the layers 1.55m and 2.35m the KM is '+str(KMb)+' m^2/s.')
print('Between the layers 2.35m and 3.72m the KM is '+str(KMc)+' m^2/s.')
print('Between the layers 3.72m and 6.15m the KM is '+str(KMd)+' m^2/s.')
print('Between the layers 6.15m and 9.05m the KM is '+str(KMe)+' m^2/s.')
```

```
print('KM changes with height because it relates the stress to the gradient in␣
↪wind speed.')
print('As you get higher in elevation the wind speed increases so there is␣
↪larger KM value which corresponds to larger eddies.' )
print('This happens because changing heights will increase the denominator and␣
↪the wind speed stays relatively same.')
```

```
Between the layers 0.95m and 1.55m the KM is 0.09183128482224327 m^2/s.
Between the layers 1.55m and 2.35m the KM is 0.20887115763490635 m^2/s.
Between the layers 2.35m and 3.72m the KM is 0.27639825348391833 m^2/s.
Between the layers 3.72m and 6.15m the KM is 0.3851994429418743 m^2/s.
Between the layers 6.15m and 9.05m the KM is 0.5850765949659585 m^2/s.
KM changes with height because it relates the stress to the gradient in wind
speed.
As you get higher in elevation the wind speed increases so there is larger KM
value which corresponds to larger eddies.
This happens because changing heights will increase the denominator and the wind
speed stays relatively same.
```

5. From the values in df1, calculate the aerodynamic resistance of the momentum flux raM for the layer from the surface to 9.05 m. How would an increased aerodynamic resistance alter the momentum flux? [4] rubric: [1] approach [1] value [2] discussion

$$r_{a_M} = \rho_a \frac{\Delta \overline{u}}{\tau_0}$$

[28]:
```
changeinu = 9.05
t = pa*(ustar**2)
ram = pa*(changeinu/t)

print('From the surface to 9.05 m, raM is '+str(ram)+ ' s/m.')
print('Larger raM value equals more aerodynamically rough surfaces, which are␣
↪less efficient at mixing momentum and creates smaller eddies.')
```

```
From the surface to 9.05 m, raM is 203.8971573497918 s/m.
Larger raM value equals more aerodynamically rough surfaces, which are less
efficient at mixing momentum and creates smaller eddies.
```

6. From the turbulence data provided in the turbulence dataframe df2, calculate , , and . . You can use a built-in averaging funcVon to check your answer, but write down and use the

equaVon to calculate the mean that involves a sum (from the lecture slides). The answers should match. [4] rubric: [2] approach [1] values [1] for equaVon. -1 for missing units.

$$\overline{a} = \frac{1}{N}\sum_{i=0}^{N-1} a(t_i, x_0)$$

[29]:
```
umean = (1/1800)*(df2['u'].sum())
vmean = (1/1800)*(df2['v'].sum())
wmean = (1/1800)*(df2['w'].sum())
umean1 = df2['u'].mean()
vmean1 = df2['v'].mean()
wmean1 = df2['w'].mean()

print('The temporal average of u is ' +str(umean) +' m/s.')
print('The temporal average of v is ' +str(vmean) +' m/s. This is close to zero␣
↪so negligible mean.')
print('The temporal average of w is ' +str(wmean) +' m/s. This is close to zero␣
↪so negligible mean.')
```

```
The temporal average of u is 2.8193847222222224 m/s.
The temporal average of v is -4.444444444532773e-07 m/s. This is close to zero
so negligible mean.
The temporal average of w is 2.777777777758968e-07 m/s. This is close to zero so
negligible mean.
```

7. From the data in the turbulence dataframe df2, calculate , , and . Name and briefly define/describe these parameters and state what they are used to calculate. [4]. rubric: [1.5] values [2.5] discussion -1 for missing units

$$\overline{a'^2} = \frac{1}{N}\sum_{i=0}^{N-1} a'^2(t_i, x_0)$$

[30]:
```
varianceu = ((df2['u']-umean)**2).sum()/1800
variancev = ((df2['v']-vmean)**2).sum()/1800
variancew = ((df2['w']-wmean)**2).sum()/1800
l = (df2['u']-umean)**2
```

```python
print('The variance of u is ' +str(varianceu)+' m^2/s^2.')
print('The variance of v is ' +str(variancev)+' m^2/s^2.')
print('The variance of w is ' +str(variancew)+' m^2/s^2.')

print('Variance describes the square of the anomalies from the mean.')
print('Each one of the numbers describes the mean of the variance in each velocity direction.')
print('They are used to calculate he level of variability in turbulent flow.  Using the root mean square of the instantaneous variation in velocity,')
print('the strength of turbulence is characterized. (which is the standard deviation)')
```

```
The variance of u is 0.26546658831658954 m^2/s^2.
The variance of v is 0.2035095967220247 m^2/s^2.
The variance of w is 0.07512605743881172 m^2/s^2.
Variance describes the square of the anomalies from the mean.
Each one of the numbers describes the mean of the variance in each velocity
direction.
They are used to calculate he level of variability in turbulent flow.  Using the
root mean square of the instantaneous variation in velocity,
the strength of turbulence is characterized. (which is the standard deviation)
```

8. From the data in the turbulence dataframe df2, calculate the turbulence intensiVes I u , Iv , and I w . Briefly discuss what these values tell you. [4] rubric: [1.5] values [2.5] discussion

$$I_u = \sigma_u / M$$

$$I_v = \sigma_v / M$$

$$I_w = \sigma_w / M$$

$$M = \sqrt{\overline{u}^2 + \overline{v}^2 + \overline{w}^2}$$

```
[31]: M = math.sqrt((umean**2)+(vmean**2)+(wmean**2))
Iu = (math.sqrt(varianceu))/M
Iv = (math.sqrt(variancev))/M
Iw = (math.sqrt(variancew))/M

print('The turbulence intensity for u is '+str(Iu)+'.')
print('The turbulence intensity for v is '+str(Iv)+'.')
print('The turbulence intensity for w is '+str(Iw)+'.')

print('These values tell us how strong the turbulence is in a turbulence field.
 ↪')
print('It is a dimensionless ratio between the standard deviation and the␣
 ↪length of the mean wind vector M.')
print('Turbulence intensity changes with the mean wind speed, with the surface␣
 ↪roughness, with the atmospheric stability and with the topographic features.
 ↪')
```

```
The turbulence intensity for u is 0.18274714135845666.
The turbulence intensity for v is 0.16000667658358786.
The turbulence intensity for w is 0.09721671835334078.
These values tell us how strong the turbulence is in a turbulence field.
It is a dimensionless ratio between the standard deviation and the length of the
mean wind vector M.
Turbulence intensity changes with the mean wind speed, with the surface
roughness, with the atmospheric stability and with the topographic features.
```

9. Turbulent kinetic energy:

a. Define turbulent kinetic energy and write the equation used to calculate it. [2]
b. From the data provided, calculate the mean turbulent kinetic energy per unit mass [2].
c. What is the raVo of to the mean kinetic energy per unit mass? [1] Rubric: a: [1] for equaVon, [1] for definiVon b: [1] for approach, [1] for value c: [1] for value

Turbulent kinetic energy the quantitative measure of the intensity of turbulence for a given flow.

$$\overline{e} = \frac{1}{2}\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right)$$

The equation to calculate it is:

```
[32]: emean = (varianceu + variancev + variancew)/2
print('The mean turbulent kinetic energy (TKE) is ' +str(emean)+' m^2/s^2.')
meankineticperunitmass = ((umean**2)+(vmean**2)+(wmean**2))/2
print('The ratio of  to the mean kinetic energy per unit mass is␣
 ↪'+str(emean)+' : '+str(meankineticperunitmass)+'.')
```

```
The mean turbulent kinetic energy (TKE) is 0.272051121238713 m^2/s^2.
The ratio of  to the mean kinetic energy per unit mass is 0.272051121238713 :
```

`3.974465105950176.`

10. Which of the three wind components, u, v or w, contains most turbulent kinetic energy per unit mass (in the dataframe df2)? Speculate about the shape of the eddies [2]. Rubric: [1] for answer [1] for discussion

The u wind component contains the most turbulent kinetic energy. The shape of the eddies would be long in the u direction as the wind velocity pushes the eddies in that u direction. With a higher u wind velocity the eddies are more stable.