Assignment2_solutions

October 30, 2024

1 Assignment 2

1.1 GEOS 300

Term 1 (Autumn 2024)

University of British Columbia

1.1.1 Instructor:

Marysa Laguë, Assistant Professor, Department of Geography

1.2 Instructions:

It is strongly recommended that you complete this assignment in Python or R. Templates (this document) are provided for the Python programming language. You may choose to complete this assignment using other software, such as Excel, Sheets, or Numbers.

Please upload your completed assignment as a .pdf file to the course Canvas page as a single, well structured report. Include all figures, tables, graphs, code/calculations, and written answers. We recommend completing the assignment within a Jupyter Notebook document (like this one); you can add "cells" for written answers using the Markdown format for the cell. The completed notebook can then be downloaded as a .pdf file (File -> Save and Export Notebook As -> pdf), which you can upload to canvas. If you do not complete the assignment in a JupyterNotebook, please upload a separate file containing your code.

You can choose to instead write your answers in some other document processor (e.g. Word) and paste your figures, code/calculations etc., however, if you choose to do this, please ensure all your code and calculations are legible, and ensure it is clear what language/program was used to perform the calculations. Label the report document with your name, your student number, the course and year. Upload your report to Canvas by the Assignment deadline on the Canvas page. Do not attach a spreadsheet (except as a supplemental code document if you complete the assignment in Sheets/Excel/Numbers etc).

Include **correct units on all plots and all answers**, where applicable. Label all axes with the appropriate variables and units.

Points per question are indicated in square brackets. This assignment is worth 10% of the final course grade.

1.2.1 Site/data details:

dataset 1 (data20100710): This dataset contains radiation data measured on the UBC Vancovuer campus at Totem Field (49.2 ◦ N, 123.2 ◦ W). This data can be found in the file data20100710_py.csv on the Canvas webpage for this assignment. The .csv file contains the following variables: incoming and reflected short-wave radiation (K↓, K↑), incoming and outgoing longwave radiation (L↓, L↑), air temperature (Ta) and relative humidity (RH). Use this dataset to answer the remaining questions. Place the .csv file in the same folder as this .ipynb folder in your JupyterOpen file system.

dataset 2 (data20090324): The soil at the climate station has been analyzed in the lab and the following values were determined: porosity is P = 0.57, bulk density of the dry soil is s = 1.13 Mg m−3. The soil organic mass fraction was determined 3.77 % (of total dry soil mass). Assume that those values apply to the entire vertical profile.

Getting started: enter your name and student number

[1]:
```
Student_Name = 'Marysa Lague'
Student_Number = 123456789
print(f'GEOS 300 Assignment 1 Submission for {Student_Name}: {Student_Number}')
```

```
GEOS 300 Assignment 1 Submission for Marysa Lague: 123456789
```

We need to import python "packages" that contain useful functions for the kind of data analysis covered in this assignment.

[2]:
```
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
```

1.3 For the first YY questions, we'll continue using the same dataset as in assignment 1.

[3]:
```
### First, open the dataset:

# Import the data  - upload this file from Canvas and put it in the same folder␣
 ↪as your assignment.
# data_file = 'GEOS300/Fall2024/Assignment1/FromSara/data/data20100710.csv'
data_file = 'data20100710_py.csv'
```

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

# Pandas (pd here) allows us to set a timestamp as an index which lets us␣
 ↪easily parse time series data
# It opens the csv file into a data format called a "data frame", so we're␣
 ↪going to call it "df" for short
df1 = pd.read_csv(data_file,parse_dates=['Date'],date_format='%Y-%m-%d %H:
 ↪%M',index_col=['Date'])

# df contains all the variables that were column headers of the .csv file.
# the "Date" dimension is the "index" dimension for the dataframe. The dates␣
 ↪are saved as "datetime" objects which know
# lots of useful things about time, like how to interpret minutes and hours,␣
 ↪etc.

# We can get a extra variables (DOY & HOUR) that will be helpful later
df1['HOUR'] = df1.index.hour
df1['DOY'] = df1.index.dayofyear
df1['TIME'] = df1.index.time

# Take a quick look at the first few entries - the pandas "head()" command␣
 ↪prints out the top of the dataframe that you just opened:
df1.head()

# "NaN" stands for "not a number", and is used in datasets to show where there␣
 ↪is no value for the variable at that time.
```

[3]:

Date	K_in	K_out	L_in	L_out	AirT	RH	HOUR	DOY	TIME
2010-07-10 00:10:00	0.0	0.0	383.0	403.2	NaN	NaN	0	191	00:10:00
2010-07-10 00:20:00	0.0	0.0	370.9	400.8	NaN	NaN	0	191	00:20:00
2010-07-10 00:30:00	0.0	0.0	363.1	399.1	19.3	70.2	0	191	00:30:00
2010-07-10 00:40:00	0.0	0.0	355.6	397.5	NaN	NaN	0	191	00:40:00
2010-07-10 00:50:00	0.0	0.0	357.3	397.4	NaN	NaN	0	191	00:50:00

[ ]:

1.4 Question 1:

[7]

(a) [3] Estimate the approximate surface albedo of the grass surface for that day. Justify your calculation of the albedo.

(b) [4] Plot the variation in surface albedo over the course of the day. Label all axes and lines, and include correct units if/where appropriate. Do you observe any diurnal variation of the albedo? **HINT: you can copy-paste the plotting code from Assignment 1 and modify it to plot albedo instead of radiative fluxes.**

[ ]:

[ ]:

Rubric:

(a) [1] for right approach / justification (albedo = reflected / incident), [1] for code, [1] for right answer

(b) [1] for generating a plot, [1] for labeling axes, [1] for the correct line, [1] for commeting on the diurnal variation

[ ]:

Answer:

(a) [3]

[1] for right approach / justification (albedo = reflected / incident)

[1] for code

[1] for right answer

[4]:
```
avg_albedo = sum(df1['K_out'])/sum(df1['K_in'])

print('The approximate surface albedo is %1.2f'%avg_albedo)


avg_albedo = np.mean(df1['K_out'])/np.mean(df1['K_in'])

print('The approximate surface albedo is %1.2f'%avg_albedo)

# note to TA if htey calculate albedo at each time step and average that, that␣
 ↪is also fine

avg_albedo = np.mean((df1['K_out'])/(df1['K_in']))

print('The approximate surface albedo is %1.2f'%avg_albedo)
```

```
The approximate surface albedo is 0.24
The approximate surface albedo is 0.24
The approximate surface albedo is 0.26
```

[5]:
```
avg_albedo = df1['K_out'].mean()/df1['K_in'].mean()

print('Alternative Calculation: The approximate surface albedo is %1.
  ↪2f'%avg_albedo)
```

```
Alternative Calculation: The approximate surface albedo is 0.24
```

(b) [4]

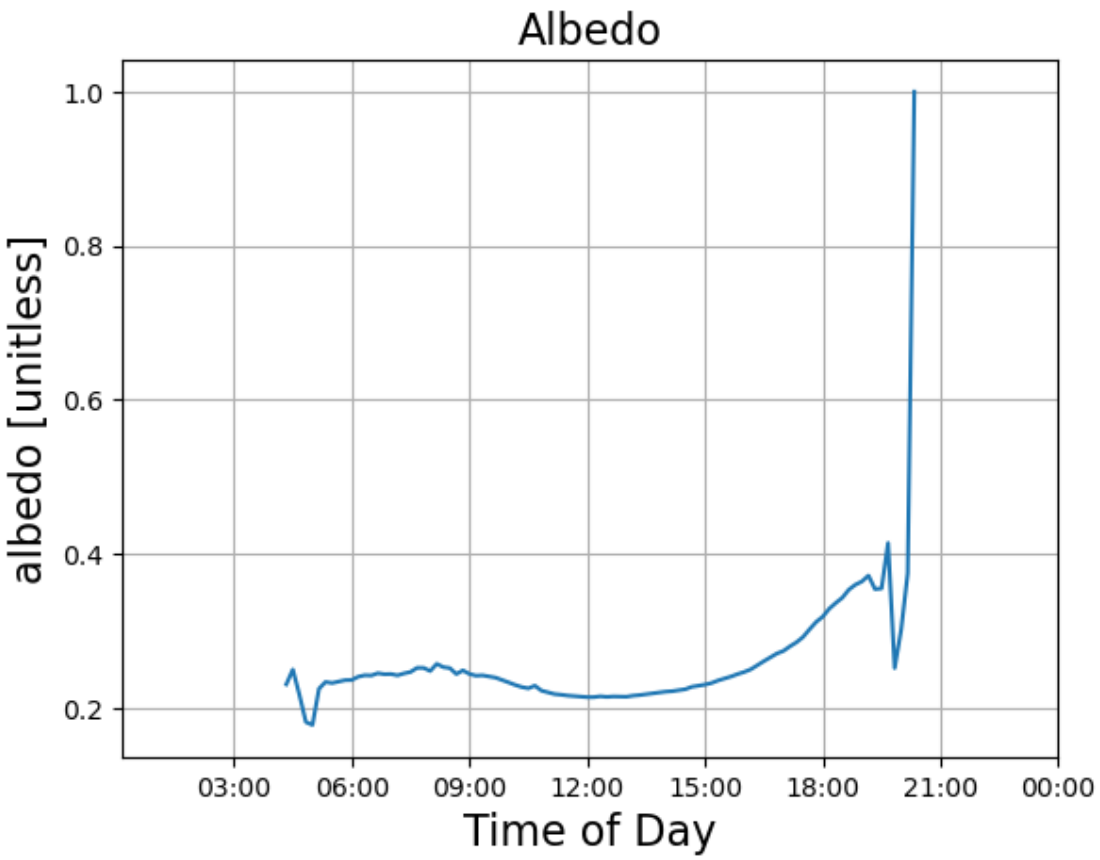
[1] for generating a plot

[1] for labeling axes

[1] for the correct line

[1] for commeting on the diurnal variation

[6]:
```
x_data = df1.index
y_data = df1['K_out']/df1['K_in']

plt.plot(x_data,y_data)
plt.xlabel('Time of Day',fontsize=16)
plt.ylabel('albedo [unitless]',fontsize=16)

plt.xlim([x_data[0],x_data[-1]])

plt.title('Albedo',fontsize=16)

plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))


plt.grid()

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



Albedo cannot be calculated when there is no incoming (and thus no reflected) shortwave radiation. There is a small variation in grass albedo throughout the day, until just before sunset when the grass appears to become extremely reflective.

This drastic increase in evening albedo doesn't mean grass is a mirror in the evening, but rather that as incoming shortwave radiation approaches zero, small measurement errors can lead to the answer "blowing up".

[ ]:

1.5 Question 2

[7]

Look-up in a text book or scientific paper an estimate of the surface emissivity 0 of a short grass-surface; cite your source in your answer.

Using this value, calculate the true surface temperature T0 in degrees Celcius (i.e. considering that the surface is a grey body and reflects) at noon for the given day.

Hint: typical emissivities of vegetation are between 0.9-1. Always include units.

rubric:

[7] - 1 for plausible 0 value, 1 for citing your source, 2 for correct approach, 2 for showing work, 1 for correct answer; -0.5 for missing units.

[ ]:

[1] for a plausible value (between 0.9-1); [1] for citing the source; [3] for correct approach to calculating surface temperature (apply stephan-boltzmann law [1], do calculation[2]); [1] for correct answer; [1] for units.

$$L_{up} = \epsilon_0 \sigma T_s^4 + (1 - \zeta_0) L_{down}$$
$$L_{up} = \epsilon_0 \sigma T_s^4 + (1 - \epsilon_0) L_{down}$$
$$T_s = \left(\frac{L_{up} - (1 - \epsilon_0) L_{down}}{\sigma \epsilon_0}\right)^{1/4}$$

Here I use an emissivity of 0.97 (appropriate for green grass, source: French et al. 2000) https://www.sciencedirect.com/science/article/pii/S0034425700001152?casa_token=rGMykw85tG8AAAAA:EOI

[7]:
```
eps0 = 0.97
sigma = 5.67e-8


df_noon = df1.loc[df1['TIME']==datetime.time(hour=12, minute=0, second=0)]

Tsfc = (( df_noon['L_out'] - ((1-eps0)*df_noon['L_in']) )/(eps0*sigma))**(1/4)

# print Tsfc in Kelvin:
Tsfc

Tsfc_C = Tsfc - 273.15

print("The true surface temperature is: %1.2f"%Tsfc_C + " " + chr(176) + "C")
```

```
The true surface temperature is: 31.91 °C

/tmp/ipykernel_753/4250567362.py:14: FutureWarning: Calling float on a single
element Series is deprecated and will raise a TypeError in the future. Use
float(ser.iloc[0]) instead
  print("The true surface temperature is: %1.2f"%Tsfc_C + " " + chr(176) + "C")
```

[ ]:

[ ]:

1.6 Question 3

[8]

Find a way to calculate the 'apparent' radiative sky temperature (T_{sky}) in °C from the measured L↓ at noon ('apparent' means you should assume $\varepsilon_a = 1.0$). How would you interpret T_{sky}?

rubric:

[8] - 2 for approach, 2 for showing work, 2 for numerical answer (-0.5 per missing unit in answers), 2 for discussion

[ ]:

[ ]:

$$L_{in} = \sigma T_{sky}^4$$
$$\Rightarrow$$
$$T_{sky}^4 = \frac{L_{in}}{\sigma}$$
$$\Rightarrow$$
$$T_{sky} = \left(\frac{L_{in}}{\sigma}\right)^{\frac{1}{4}}$$

Where T_{sky} is in KELVIN. To get to degrees Celcius, subtract 273.15:

$$T_{sky,C} = T_{sky} - 273.15$$

[8]:
```
# Tsky_app <- (data$L_in[ind]/(1*sigma))^(1/4)-273.15
df_noon = df1.loc[df1['TIME']==datetime.time(hour=12, minute=0, second=0)]
sigma = 5.67e-8


T_sky_K = (df_noon['L_in']/sigma)**(1/4)
T_sky_C = T_sky_K - 273.15


print("The apparent sky temperature is: %1.2f"%T_sky_C.values[0] + " " +␣
 ↪chr(176) + "C")
```

```
The apparent sky temperature is: 12.69 °C
```

Discussion: T_{sky} is the temperature the sky would be if all the downwards longwave radiation were coming from a single "slab" of atmosphere with a temperature of 12.7 degrees C. The actual temperature of the atmosphere varies with height, and the surface is receiving downwards longwave radiation from parts of the atmosphere with different temperatures.

[ ]:

1.7 The remainder of the problem set uses a dataset of soil temperature observations

The data-set consists of 15-min averages of the following variables: four soil temperatures (T1, T2, T3, and T4) measured at depths of 5 cm, 10 cm, 20 cm and 50 cm, respectively, soil heat flux density Q_G from a soil heat flux plate installed at a depth of 7.5 cm, soil volumetric water content $_w$ measured using TDR at -7.5 cm, net all-wave radiation Q^* measured 2 m above the surface, and sensible heat flux density in the atmosphere Q_H measured 2m above the surface. Use this data-set to answer all the following questions.

The soil at the climate station has been analyzed in the lab and the following values were determined: porosity is P = 0.57, bulk density of the dry soil is s = 1.13 Mg m−3. The soil organic mass fraction was determined 3.77 % (of total dry soil mass). Assume that those values apply to the entire vertical profile.

dataset name:

data20090324.xls

[ ]:

[9]:
```
# Import the data  - upload this file from Canvas and put it in the same folder␣
 ↪as your assignment.
data_file = 'data20090324.xls'


df2 = pd.read_excel(data_file)

# a few post processing steps:

# 1. 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]

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

df2.head()
```

[9]:

	Date	T_1	T_2	T_3	T_4	QG	qw	Q*	QH
0	2009-03-24 00:15:00	0.844	1.110	1.383	1.313	-1.88	0.366	-1.2	NaN
1	2009-03-24 00:30:00	0.822	1.112	1.372	1.307	-1.86	0.367	-1.5	-40.7
2	2009-03-24 00:45:00	0.817	1.089	1.358	1.316	-1.89	0.367	-2.1	NaN
3	2009-03-24 01:00:00	0.797	1.093	1.381	1.330	-1.98	0.367	-2.1	-16.3
4	2009-03-24 01:15:00	0.788	1.060	1.382	1.333	-2.12	0.368	-1.8	NaN

1.8 Question 4

[8]

Calculate the net warming/cooling of the soil over the 24 hours separately for the 5 cm, 10 cm, 20 cm and the 50 cm depth (i.e. the temperature change from midnight to midnight). Speculate what causes the warming or cooling.

Always include units.

rubric:

[8] - 2 for approach, 2 for showing work, 1ea for numerical answer, -0.5 per missing unit in answers

[ ]:

Solution:

[10]:
```
# Note to calculate the net warming/cooling you can calculate T at 24:00 (last␣
 ↪value in column) - T at 00:10 (first value in colum)
# for each depth (e.g. to TS_1 that would be tail(data$T_1, n=1)-data$T_1[1])
dT1 = df2['T_1'].values[-1] - df2['T_1'].values[0]
dT2 = df2['T_2'].values[-1] - df2['T_2'].values[0]
dT3 = df2['T_3'].values[-1] - df2['T_3'].values[0]
dT4 = df2['T_4'].values[-1] - df2['T_4'].values[0]

print("the net warming/cooling rates are:\n 5 cm: %1.3f K/day\n 10 cm: %1.3f K/
 ↪day\n 20 cm: %1.3f K/day\n 50 cm: %1.3f K/day\n  "
      %(dT1,dT2,dT3,dT4) )
```

```
the net warming/cooling rates are:
 5 cm: 1.270 K/day
 10 cm: 1.406 K/day
 20 cm: 0.968 K/day
 50 cm: 0.004 K/day
```

[ ]:

[ ]:

Each of the soil layers is experiencing warming. The least warming happens at the deepest (50 cm) soil layer. The most warming doesn't actually happen right at the surface (5 cm soil layer), but rather at the 10 cm soil layer.

Reasons for this could include variations in heat capacity, thermal conductivity, and/or porosity. More heating would occur if the layer had a lower heat capacity (high heat capacity = more energy needs to be added to get the same change in temperature). Other discussion points also fine (eg argue for thermal conductivity, differneces in porosity, differences in water content etc.)

[ ]:

1.9 Question 5:

[14]

(a) [8] Calculate the daily average soil temperature for each of the four depths where temperatures are provided (T1 to T4).

(b) [6]Using those, determine the direction of the daily total QG in the soil layers from 5 - 10 cm, 10 - 20cm and 20 - 50cm?

Always include units.

rubric:

(a) [8] - 2 for approach, 2 for showing work, 1ea for numerical answer

(b) [6] - 2 for approach, 1 for showing work, 1ea for 3x direction of flux answer

-0.5 per missing unit.

[ ]:

[ ]:

Solution:

[33]:
```
# Note to calculate the net warming/cooling you can calculate T at 24:00 (last␣
 ↪value in column) - T at 00:10 (first value in colum)
# for each depth (e.g. to TS_1 that would be tail(data$T_1, n=1)-data$T_1[1])
T1_avg = df2['T_1'].mean()
T2_avg = df2['T_2'].mean()
T3_avg = df2['T_3'].mean()
T4_avg = df2['T_4'].mean()

print("the average daily temperatures at each layer are:\n 5 cm: %1.3f degrees␣
 ↪C\n 10 cm: %1.3f degrees C\n 20 cm: %1.3f degrees C\n 50 cm: %1.3f degrees␣
 ↪C\n  "
      %(T1_avg,T2_avg,T3_avg,T4_avg) )
```

```
the average daily temperatures at each layer are:
 5 cm: 2.696 degrees C
 10 cm: 2.145 degrees C
 20 cm: 1.577 degrees C
 50 cm: 1.279 degrees C
```

Because the upper soil layers are always warmer than the lower soil layers, the daily total QG is downwards - that is, the upper layers are heating the lower layers, and the soil column as a whole is absorbing (rather than losing) energy.

[ ]:

[ ]:

[ ]:

1.10 Question 6:

[4]

Calculate the daily total of Q_G at 7.5 cm depth in MJ m−2 day−1 using the measured values from the soil heat flux plate. Compare the direction of Q_G to the direction of the heat flux obtained for the 5-10 cm layer in question 5.

Rubric: [1] for approach, [1] for showing your work, [1] for numerical answer, [1] for discussion. -0.5 per missing unit.

[ ]:

Solution:

[34]:
```
df2
```

[34]:

	Date	T_1	T_2	T_3	T_4	QG	qw	Q*	QH
0	2009-03-24 00:15:00	0.844	1.110	1.383	1.313	-1.88	0.366	-1.2	NaN
1	2009-03-24 00:30:00	0.822	1.112	1.372	1.307	-1.86	0.367	-1.5	-40.7
2	2009-03-24 00:45:00	0.817	1.089	1.358	1.316	-1.89	0.367	-2.1	NaN
3	2009-03-24 01:00:00	0.797	1.093	1.381	1.330	-1.98	0.367	-2.1	-16.3
4	2009-03-24 01:15:00	0.788	1.060	1.382	1.333	-2.12	0.368	-1.8	NaN
..	…	…	…	…	…	…	…	…	…
91	2009-03-24 23:00:00	2.308	2.730	2.399	1.280	-2.05	0.360	-25.9	-27.4
92	2009-03-24 23:15:00	2.273	2.668	2.382	1.314	-2.15	0.361	-26.7	NaN
93	2009-03-24 23:30:00	2.207	2.610	2.384	1.300	-2.26	0.361	-28.8	-27.9
94	2009-03-24 23:45:00	2.159	2.556	2.345	1.318	-2.38	0.361	-32.9	NaN
95	2009-03-25 00:00:00	2.114	2.516	2.351	1.317	-2.53	0.361	-31.0	-24.5

[96 rows x 9 columns]

[35]:
```
# Calculate the daily total of QG at 7.5 cm depth in MJ m-2 day-1
# calculate the mean then convert to MJ and days
QG_daily = df2['QG'].mean()*(60*60*24)/(10**6)

print("The daily total of QG at 7.5 cm depth is %1.2f MJ m-2 day-1"%QG_daily)
```

```
The daily total of QG at 7.5 cm depth is 0.41 MJ m-2 day-1
```

Discussion: The value of 0.41 MJ/m2/day found here at 7.5 cm depth suggests positive downwards heat flux, which is consistent with the expected flux between layer 1 (5m) and layer 2 (10 cm) because layer 1 is warmer than layer 2, so there should be ground heatflux downwards (positive) from layer 1 to layer 2.

[36]:
```
# Calculate the daily total of QG at 7.5 cm depth in MJ m-2 day-1
# calculate the mean then convert to MJ and days
QG_daily = df2['QG'].sum()*(60*15)/(10**6)

print("The daily total of QG at 7.5 cm depth is %1.2f MJ m-2 day-1"%QG_daily)
```

```
The daily total of QG at 7.5 cm depth is 0.41 MJ m-2 day-1
```

1.11 Question 7:

[7]

Estimate the thermal conductivity of the soil k at noon that day. Is k constant throughout the day?

Hint: use the temperature gradient between the 5 and 10 cm soil layers.

Rubric:

[2] for approach, [2] for showing work, [1] for numerical answer, [2] for discussion of variation in k throughout day. -0.5 for missing units.

[ ]:
```

```

[ ]:
```

```

Solution:

[37]:
```
# find noon - can use a fancy method like we did with df1, or just brute-force␣
↪find noon:
noon = df2.Date[47]
print(noon)

df_noon = df2.iloc[47]
df_noon
```

```
2009-03-24 12:00:00
```

[37]:
```
Date    2009-03-24 12:00:00
T_1                   4.032
T_2                    1.86
T_3                    1.01
T_4                   1.229
QG                    16.17
qw                    0.353
Q*                    269.5
QH                     49.3
Name: 47, dtype: object
```

From fourier's law in the slides,

$$Q_G = -k\frac{dT}{dz}$$

so

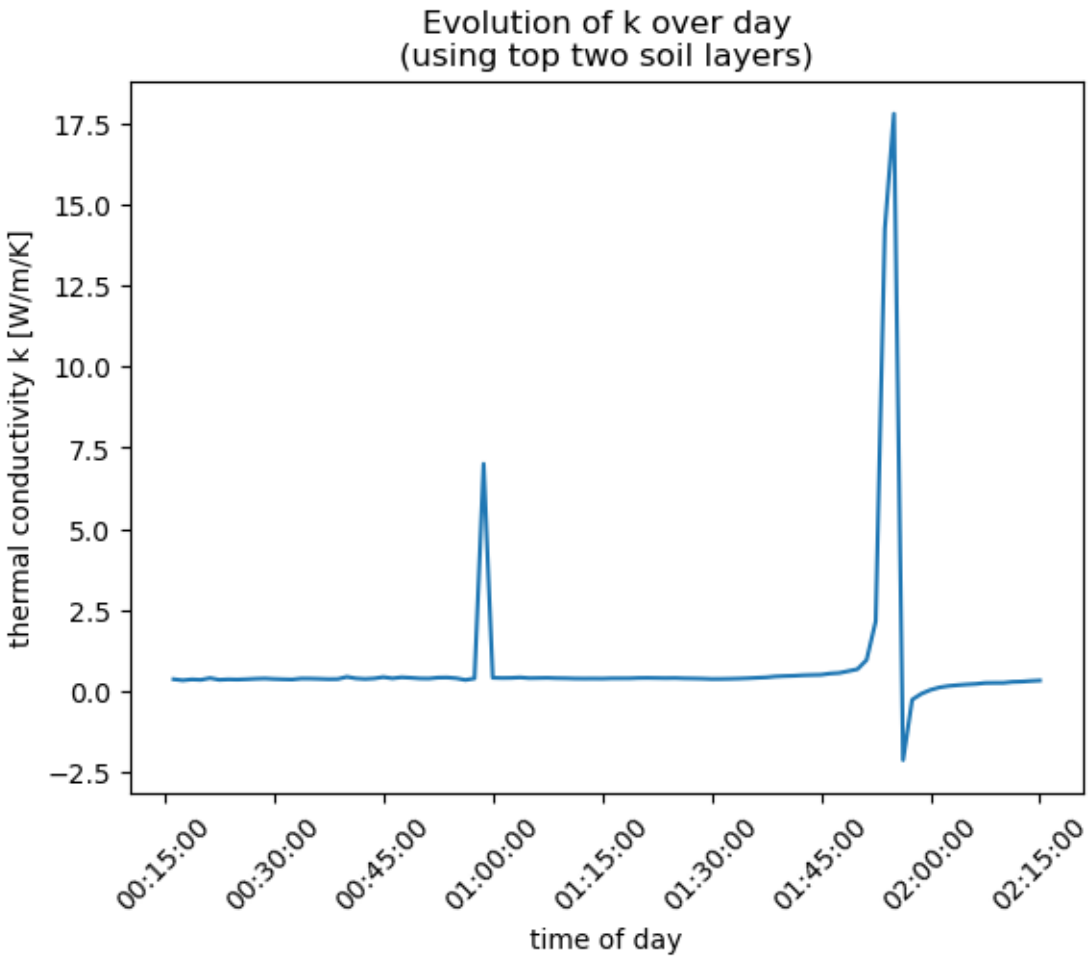
$$k = -\frac{(Q_G)}{(\frac{dT}{dz})}$$

[38]:
```
k_noon = -df_noon.QG / ( (df_noon.T_2 - df_noon.T_1  )/( 0.05 ) )

print("The thermal conductivity k at noon is %1.2f W m-1 K-1 "%k_noon)
```

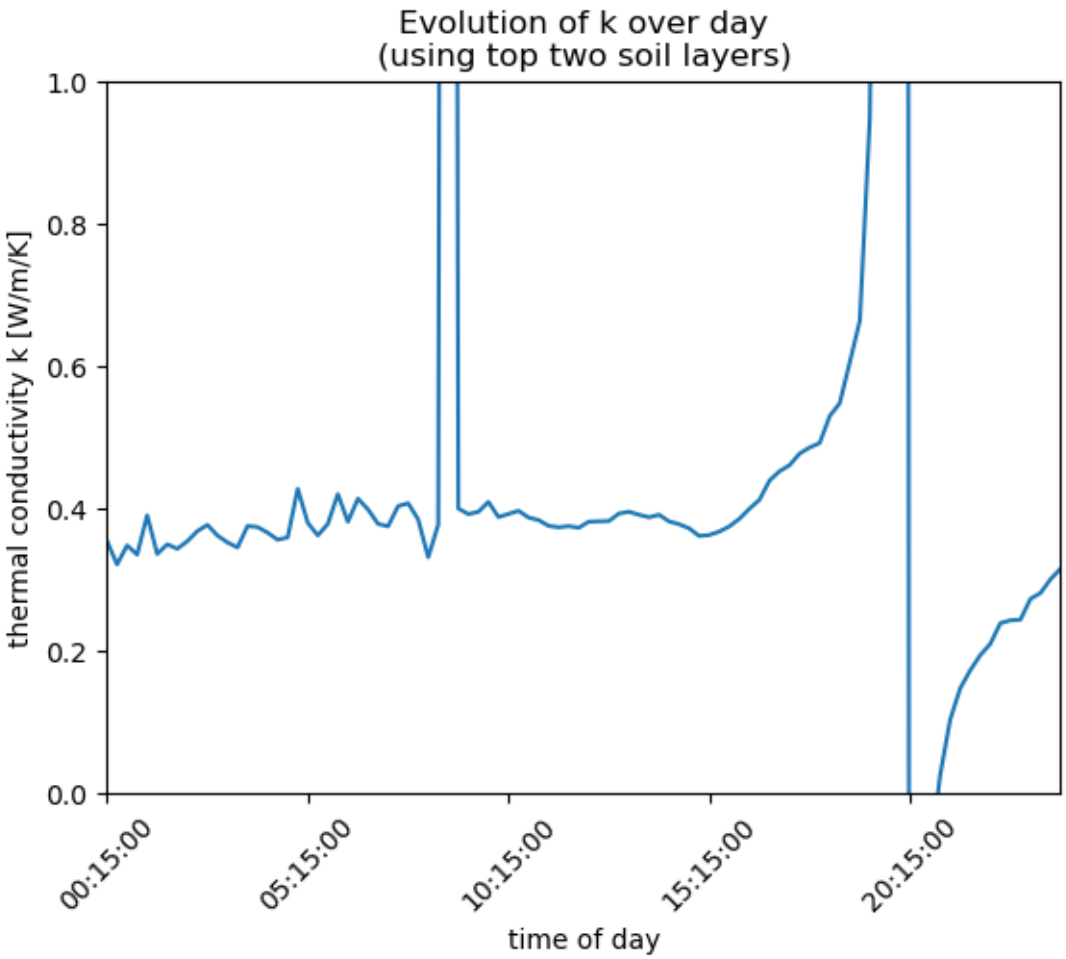
```
The thermal conductivity k at noon is 0.37 W m-1 K-1
```

[39]:
```
# to estimate of k changes over the course of the day, you can either
# calculate it at several times during the day, or plot it:

k_day = -df2.QG / ( (df2.T_2 - df2.T_1  )/( 0.05 ) )

plt.plot(df2.Date,k_day)
plt.xlabel('time of day')
plt.ylabel('thermal conductivity k [W/m/K]')
plt.title('Evolution of k over day\n(using top two soil layers)')
plt.gca().set_xticklabels(df2.Date.dt.time,rotation=45)

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

```
/tmp/ipykernel_753/1470654979.py:10: UserWarning: set_ticklabels() should only
be used with a fixed number of ticks, i.e. after set_ticks() or using a
FixedLocator.
  plt.gca().set_xticklabels(df2.Date.dt.time,rotation=45)
```

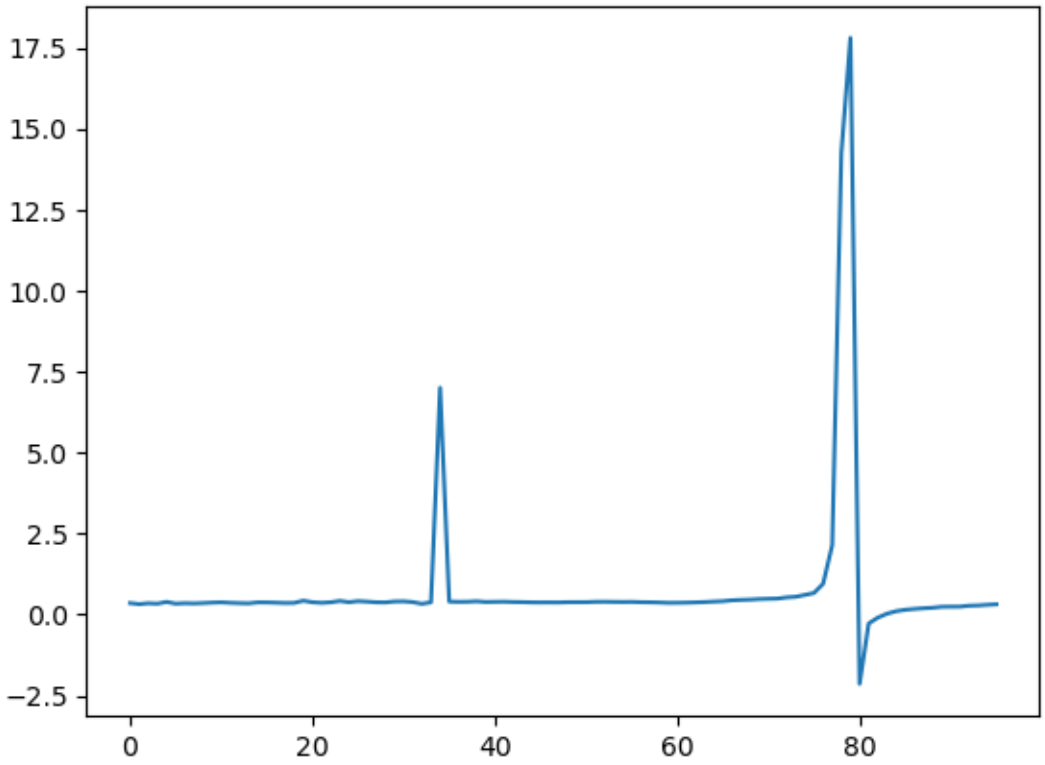


[40]:
```
# to estimate of k changes over the course of the day, you can either
# calculate it at several times during the day, or plot it:

k_day = -df2.QG / ( (df2.T_2 - df2.T_1  )/( 0.05 ) )

plt.plot(df2.Date,k_day)
plt.xlabel('time of day')
plt.ylabel('thermal conductivity k [W/m/K]')
plt.title('Evolution of k over day\n(using top two soil layers)')
plt.gca().set_xticks(df2.Date[::20])
plt.gca().set_xticklabels(df2.Date.dt.time[::20],rotation=45)
plt.ylim([0,1])
plt.xlim([df2.Date.min(),df2.Date.max()])
plt.show()
plt.close()
```



[41]:
```
plt.plot(k_day.values)
```

[41]:
```
[<matplotlib.lines.Line2D at 0x7f821ed88950>]
```



[42]:
```
print(df2.Date.min())
print(df2.Date.max())
```

```
2009-03-24 00:15:00
2009-03-25 00:00:00
```

[43]:
```
df2.Date
```

[43]:
```
0    2009-03-24 00:15:00
1    2009-03-24 00:30:00
2    2009-03-24 00:45:00
3    2009-03-24 01:00:00
4    2009-03-24 01:15:00
             …
91   2009-03-24 23:00:00
92   2009-03-24 23:15:00
93   2009-03-24 23:30:00
94   2009-03-24 23:45:00
95   2009-03-25 00:00:00
Name: Date, Length: 96, dtype: datetime64[ns]
```

Discussion: the thermal diffusivity k is approximately constant throughout the day, though there are some spurious extreme values that show up when the temperature difference between the 5 and

10 cm soil layers is almost identical, resulting in us dividing by zero.

[44]:
```
df2
```

[44]:

	Date	T_1	T_2	T_3	T_4	QG	qw	Q*	QH
0	2009-03-24 00:15:00	0.844	1.110	1.383	1.313	-1.88	0.366	-1.2	NaN
1	2009-03-24 00:30:00	0.822	1.112	1.372	1.307	-1.86	0.367	-1.5	-40.7
2	2009-03-24 00:45:00	0.817	1.089	1.358	1.316	-1.89	0.367	-2.1	NaN
3	2009-03-24 01:00:00	0.797	1.093	1.381	1.330	-1.98	0.367	-2.1	-16.3
4	2009-03-24 01:15:00	0.788	1.060	1.382	1.333	-2.12	0.368	-1.8	NaN
..	…	…	…	…	…	…	…	…	…
91	2009-03-24 23:00:00	2.308	2.730	2.399	1.280	-2.05	0.360	-25.9	-27.4
92	2009-03-24 23:15:00	2.273	2.668	2.382	1.314	-2.15	0.361	-26.7	NaN
93	2009-03-24 23:30:00	2.207	2.610	2.384	1.300	-2.26	0.361	-28.8	-27.9
94	2009-03-24 23:45:00	2.159	2.556	2.345	1.318	-2.38	0.361	-32.9	NaN
95	2009-03-25 00:00:00	2.114	2.516	2.351	1.317	-2.53	0.361	-31.0	-24.5

[96 rows x 9 columns]

[52]:
```
kappa = k_noon / (Cs*10**6)
print(kappa)
print(k_noon)
```

```
1.5440410003445616e-07
0.3722375690607736
```

[53]:
```
k_mean = (-df2.QG / ( (df2.T_2 - df2.T_1  )/( 0.05 ) )).mean()
kappa_mean = k_mean / (Cs*10**6)
print(kappa_mean)
print(k_mean)
```

```
3.1108201737557654e-07
0.7499568593098583
```

1.12 Question 8:

[5]

Calculate the heat capacity C of the soil using the lab analysis results (see text where the dataset is described above) and measured soil water content w.

Rubric: [2] for approach, [2] for showing work, [1] for answer; -0.5 for missing units.

[ ]:
```

```

solution:

From the slides,

$$C_s = C_m\theta_m + C_o\theta_o + C_w\theta_w + C_a\theta_a$$

C_a is small compared to the rest, so we ignore this term.

Use values of C_m, C_o, and C_w from slides:

[46]:
```
Cm = 2.1
Co = 2.5
Cw = 4.18
```

From the slides, porosity P = 1 - θ_g (the dry soil grains), so we can get θ_g from the measured posority. Once we have θ_g, we can get θ_m and θ_o from the measured organic fraction.

[47]:
```
theta_g = 1-0.57
print(theta_g)
theta_o = 0.0377 * theta_g
print(theta_o)
theta_m = (1-0.0377) * theta_g
print(theta_m)
```

```
0.43000000000000005
0.016211
0.4137890000000001
```

We can get θ_w from the dataset (use the average value over the measurement period):

[48]:
```
theta_w = df2.qw.mean()
```

[49]:
```
print(theta_o + theta_m + theta_w)
```

```
0.7891666666666667
```

[50]:
```
# now calculate:
Cs = Cm*theta_m + Co*theta_o + Cw*theta_w

print("The average heat capacity of the soil using the lab analysis results is␣
 ↪%1.2f MJ m^-3 K^-1 "%Cs)
```

```
The average heat capacity of the soil using the lab analysis results is 2.41 MJ
m^-3 K^-1
```

[ ]:
```

```

[51]:
```
pwd
```

[51]:
```
'/home/jovyan/GEOS300/Fall2024/Assignment2/A2_python'
```

1.13 Question 9:

[7]

With C from question 8, calculate the depths where you expect the amplitude of the diurnal and yearly waves to drop below 5% of the amplitude of the sinusoidal surface temperature wave.

rubric: [2] approach, [2] showing work, [2] numerical answer ([1] each diurnal/yearly); -0.5 per missing units.

[54]:
```
kappa = k_noon / (Cs*10**6)
print(kappa)
print(k_noon)
```

```
1.5440410003445616e-07
0.3722375690607736
```

[55]:
```
k_mean = (-df2.QG / ( (df2.T_2 - df2.T_1  )/( 0.05 ) )).mean()
kappa_mean = k_mean / (Cs*10**6)
print(kappa_mean)
print(k_mean)
```

```
3.1108201737557654e-07
0.7499568593098583
```

[56]:
```
p_diurnal = 60*60*24 #s
p_annual =  60*60*24*365 #s

D_diurnal = (kappa*p_diurnal/np.pi)**(1/2)
D_annual = (kappa*p_annual/np.pi)**(1/2)

# at 3*D the amplitude drops below 5% (from slides):
D3_diurnal = 3*D_diurnal
D3_annual = 3*D_annual


print("depth for the diurnal wave =  %1.2f m "%D3_diurnal)
print("depth for the annual wave =  %1.2f m "%D3_annual)
```

```
depth for the diurnal wave =  0.20 m
depth for the annual wave =  3.73 m
```

[57]:
```
p_diurnal = 60*60*24 #s
p_annual =  60*60*24*365 #s

D_diurnal = (kappa_mean*p_diurnal/np.pi)**(1/2)
D_annual = (kappa_mean*p_annual/np.pi)**(1/2)

# at 3*D the amplitude drops below 5% (from slides):
D3_diurnal = 3*D_diurnal
D3_annual = 3*D_annual
```

```
print("depth for the diurnal wave =  %1.2f m "%D3_diurnal)
print("depth for the annual wave =  %1.2f m "%D3_annual)
```

```
depth for the diurnal wave =  0.28 m
depth for the annual wave =  5.30 m
```

[ ]:

[ ]:

[ ]:

[26]:
```
# # First calculate for 10:00

# # Find indices of interest
# ind1 <- which(Time == '10:00')
# ind2 <- which(Time == '10:15')

# dZ <- 0.075
# dt <- 15*60
# dT <- (data$T_1[ind2]-data$T_1[ind1])

# Qs <- (C*10^6)*dT/dt*dZ

# QG <- data$QG[ind1]+Qs
# cat("the value at the surface QG(0) at 10:00: ", round(QG,2), "W m-2", "\n")

# # Then calculate for 19:00

# # Find indices of interest
# ind1 <- which(Time == '19:00')
# ind2 <- which(Time == '19:15')

# dZ <- 0.075
# dt <- 15*60
# dT <- (data$T_1[ind2]-data$T_1[ind1])

# Qs <- (C*10^6)*dT/dt*dZ
# Qs
# C
# QG <- data$QG[ind1]+Qs
# cat("the value at the surface QG(0) at 19:00: ", round(QG,2), "W m-2", "\n")
```

[ ]:

[ ]:

[ ]:

[ ]:

Solution:

[ ]:

1.14 Question 10

[4]

The Bowen ratio describes the ratio between the sensible and latent heat flux densities directed into the atmosphere, i.e. = QH/QE. Calculate from the available data for noon that day. Neglect the energy use for photosynthesis.

rubric: [2] for approach, [1] for showing work, [1] for answer

[27]:
```
QH = df_noon.QH
QE = df_noon['Q*'] - df_noon['QH'] - df_noon['QG']

beta = QH/QE

print('the bowen ratio at noon is %1.2f'%beta)
```

```
the bowen ratio at noon is 0.24
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: