

Data_sorting.py

```
1 import pandas as pd
2 import numpy as np
3 import time
4 pd.set_option("display.max_rows", None, "display.max_columns", None)
5 desired_width = 320
6 pd.set_option('display.width', desired_width)
7
8 startTime = time.time()
9 date = "2023-02-07" # Here the desired date is entered as "YYYY-MM-DD"
10 file_type = 'f0000' # The 9th of november has file 0001, all others have 0000
11 print("1/5: Date which has been loaded: " + date)
12
13 def df_maker(filename, n=None):
14     """This function creates a dataframe from a .csv-file or .txt-file, with tabs as delimiter"""
15     with open(filename) as file:
16         df = pd.DataFrame(pd.read_csv(file, delimiter='\t', skiprows=n))
17     return(df)
18
19 def df_maker2(filename, n=None):
20     """This function creates a dataframe from a .csv-file or .txt-file, with comma as delimiter"""
21     with open(filename) as file:
22         df = pd.DataFrame(pd.read_csv(file, delimiter=',', skiprows=n))
23     return(df)
24
25 def df_maker3(filename, n=None):
26     """This function creates a dataframe from a .csv-file or .txt-file, with semi-colon as delimiter"""
27     with open(filename) as file:
28         df = pd.DataFrame(pd.read_csv(file, delimiter=';', skiprows=n))
29     return(df)
30
31 """Following is an example on how to read Excel files"""
32 path_to_excel_folder = "../Speciale/Metadata_files/"
33 excel_name = "Time_for_python.xlsx"
34 excel_file = path_to_excel_folder+excel_name
35 time_df = pd.read_excel(excel_file)
36 time_df = time_df.dropna()
37 list_of_columns = list(time_df.columns)
38 list_of_columns.remove('Date')
39
40 for c in list_of_columns: # This for-loop makes the time-dataframe nice to read and makes
    computing easier
41     time_df[c] = pd.to_datetime(time_df['Date'].astype(str) + " " + time_df[c].astype(str), format='%Y-
    %m-%d %H:%M:%S')
42 time_df['Date'] = time_df['Date'].dt.date
43
44 print("2/5: Time frames have been loaded")
45
46 """Predefined values"""
47 volume_of_glasses_l = 2
48 area_of_soil_cylinder_cm2 = (7/2)**2 * np.pi
49 height_of_cylinder_cm = 12
```

```

50 area_of_total_cylinder_cm2 = (7.5/2)**2 * np.pi
51 volume_of_total_cylinder = area_of_total_cylinder_cm2 * height_of_cylinder_cm
52 volume_of_tubes_to_machine_l = ((np.pi * 180 * 0.635 * 0.635)/1000)*2
53 volume_of_glass_minus_cylinder_l = ((2*1000)-volume_of_total_cylinder)/1000 +
volume_of_tubes_to_machine_l
54 temp_of_glasses_K = 7+273.15
55 pressure_atm = 1
56 molar_weight_of_N2O_g_mol = 44.01 # Molar weight of N2O
57 ideal_gas_constant = 0.0821 # L*atm*K-1*mol-1
58 n_mol =
(pressure_atm*volume_of_glass_minus_cylinder_l)/(ideal_gas_constant*temp_of_glasses_K) # Number
of moles within the glass bottle
59 standard_molar_volume_L_mol = volume_of_glass_minus_cylinder_l/n_mol # The standard molar
volume
60 print("3/5: Fixed values have been defined")
61
62 def filter_dataframe(file, name, lines_to_ignore, keep_columns, renames):
63     """This function takes in the raw files from the machine and
64     returns a new file only with important columns and saves this in the working data folder"""
65     df = df_maker2(file, n=lines_to_ignore) # This loads the file to a dataframe and the n is the
number of lines we wish to ignore
66     df = df[keep_columns] # Here we filter out the columns we do not wish to keep to reduce
computing time
67     df = df.rename(columns = renames) # We rename the columns to give nicer names
68     df['Time'] = pd.to_datetime(df['Time'], errors= 'coerce') # We remove lines where there is no date,
sometimes there are strange lines at the bottom of the file
69     df = df.dropna(subset='Time', axis = 0)
70     df.to_csv('./Speciale/First_WD_columns_removed/' + name, index=False)
71     return (df)
72
73 name_of_file = date
74 columns_to_keep = ['Time', ' [NN15O]_ppm', ' [N15NO]_ppm',
75 ' [NNO18]_ppm', ' d15NA', ' d15NB',
76 ' d15N', ' [N2O]_ppm', ' SP']
77 new_column_names = {'Time': 'Time', ' [NN15O]_ppm': 'NN15O_ppm',
78 ' [N15NO]_ppm': 'N15NO_ppm', ' [NNO18]_ppm': 'NNO18_ppm', ' d15NA':
'd15NA',
79 ' d15NB': 'd15NB', ' d15N': 'd15N', ' [N2O]_ppm': 'N2O_ppm', ' SP':
'SP'}
80 sorting = filter_dataframe('./Speciale/Raw_data_files/'+date+'/N2Oiso_'+date+'_' + file_type + '.txt',
name_of_file, 1, columns_to_keep, new_column_names)
81
82 print("4/5: Unecessary columns have been removed")
83
84 def apply_treatment(file, seconds):
85     """This function sorts the total data file into which bottle was measured at which time"""
86     df = df_maker2('./Speciale/First_WD_columns_removed/'+file) # We load
87     df['Time'] = pd.to_datetime(df['Time']) # We convert the time column to datetime, making sorting
easier
88     time_column = df[['Time']].copy() # We copy the time column
89     time_column['Time'] = pd.to_datetime(time_column['Time'])
90     time_df['Date'] = time_df['Date'].astype(str)
91     time_frames = time_df[time_df['Date'] == file] # We load the time frame file and select the date we
are interested in
92     time_frames = time_frames.drop(columns='Date') # We drop the date column due to it making
computing easier

```

```

93 list_to_use = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46,
48, 50, 52, 54] # This was necessary to make the labelling easier
94 new_df = pd.DataFrame() # This is the dataframe we will fill
95 for c in list_to_use:
96     bottle = c/2 + 1
97     start = time_frames.iloc[0][list_of_columns[c]] # The start time is loaded
98     end = time_frames.iloc[0][list_of_columns[c+1]] # The end time is loaded
99     working_df = df[(df['Time'] > start) & (df['Time'] <= end)].copy() # This selects all the lines
between the start and the end point for each measurement
100     working_df = working_df.iloc[seconds:] # Here we remove the first 200 seconds of the file
101     working_df = working_df.reset_index(drop=True) # Here we reset the index
102     working_df['Bottle'] = int(bottle) # We label the time series with the bottle measured
103     new_df = pd.concat([new_df, working_df]) # We add it to the new dataframe
104     new_df = new_df.sort_values('Time')
105     new_df['N2O_umol_L-1'] = (new_df['N2O_ppm'] * 10 ** (-6))/standard_molar_volume_L_mol *
1000000 # Here we estimate the amount of N2O in the air in terms of micromol, pr liter
106     new_df['N2O_umol'] = new_df['N2O_umol_L-1'] * volume_of_glass_minus_cylinder_l # Here we
estimate the amount of N2O in the air in terms of micromol by multiplying with the volume
107     new_df.to_csv('../Speciale/Second_WD_sorted/' + file, index=False)
108     return(new_df)
109
110 sorting2 = apply_treatment(date, 200)
111
112 print("5/5: File has been sorted and given proper bottle label")
113 executionTime = (time.time() - startTime)
114 print('Execution time in seconds: ' + str(round(executionTime, 2)))
115

```

Regression.py

```
1 import pandas as pd
2 import numpy as np
3 from scipy import stats
4 import matplotlib.pyplot as plt
5 import time
6 import warnings
7 warnings.filterwarnings(action='ignore', category=RuntimeWarning)
8 pd.set_option("display.max_rows", None, "display.max_columns", None)
9 desired_width = 320
10 pd.set_option('display.width', desired_width)
11 startTime = time.time()
12
13 def df_maker(filename, n=None):
14     """This function creates a dataframe from a .csv-file or .txt-file, with tabs as delimiter"""
15     with open(filename) as file:
16         df = pd.DataFrame(pd.read_csv(file, delimiter='\t', skiprows=n))
17     return(df)
18
19 def df_maker2(filename, n=None):
20     """This function creates a dataframe from a .csv-file or .txt-file, with comma as delimiter"""
21     with open(filename) as file:
22         df = pd.DataFrame(pd.read_csv(file, delimiter=',', skiprows=n))
23     return(df)
24
25 def df_maker3(filename, n=None):
26     """This function creates a dataframe from a .csv-file or .txt-file, with semi-colon as delimiter"""
27     with open(filename) as file:
28         df = pd.DataFrame(pd.read_csv(file, delimiter=';', skiprows=n))
29     return(df)
30
31 def polynomial_func(x, *params):
32     return np.polyval(params, x)
33
34 def regression(x, y, degree):
35     params = np.polyfit(x, y, degree)
36     p_value = stats.linregress(x, y).pvalue
37     return params, p_value
38
39 date = '2023-02-07' # Put in the date to be used in YYYY-MM-DD format
40 print("1/6: Date which has been loaded: " + date)
41
42 """Predefined values"""
43 volume_of_glasses_l = 2
44 area_of_soil_cylinder_cm2 = (7/2)**2 * np.pi
45 height_of_cylinder_cm = 12
46 area_of_total_cylinder_cm2 = (7.5/2)**2 * np.pi
47 volume_of_total_cylinder = area_of_total_cylinder_cm2 * height_of_cylinder_cm
48 volume_of_tubes_to_machine_l = ((np.pi * 180 * 0.635 * 0.635)/1000)*2
49 volume_of_glass_minus_cylinder_l = ((2*1000)-volume_of_total_cylinder)/1000 +
volume_of_tubes_to_machine_l
50 temp_of_glasses_K = 7+273.15
51 pressure_atm = 1
52 molar_weight_of_N2O_g_mol = 44.01
```

```

53 ideal_gas_constant = 0.0821 # L*atm*K-1*mol-1
54 n_mol =
(pressure_atm*volume_of_glass_minus_cylinder_l)/(ideal_gas_constant*temp_of_glasses_K)
55 standard_molar_volume_L_mol = volume_of_glass_minus_cylinder_l/n_mol
56
57 print("2/6: Fixed values have been defined")
58
59 def rate_calculator(file):
60     """This function calculates the rate for each measurement and returns one line
61     with measurement number, rates of change and p-value. The function returns one dataframe
62     with one rate pr replicate"""
63     df = df_maker2(file) # We open the file as a dataframe
64     df['Time'] = pd.to_datetime(df['Time']) # We convert the time column to datetime format
65     list_of_dicts = [] # We create a list where we will fill in dictionaries, these are quick to compute and
requires less computer power
66     list_of_bottles = sorted(list(df.Bottle.unique())) # We create a list of all the bottles used
67     for i in list_of_bottles:
68         working_df = df[df['Bottle'] == i].copy() # We create a data frame only with the time for each
bottle
69         working_df['Time_step_s'] = (working_df['Time'] -
working_df.iloc[0]['Time']).astype('timedelta64[s]') # We create a column with the amount of time since
the first line. This line has unit seconds
70         x = working_df['Time_step_s'].to_numpy() # We decide upon an x-column to perform linear
regression on, and convert this to a numpy array
71         ppm = working_df['N2O_ppm'].to_numpy() # We convert the N2O ppm column to a numpy
array, for easier regression
72         umol = working_df['N2O_umol'].to_numpy() # We convert the N2O umol column to a numpy
array, for easier regression
73         ppm_slope, ppm_intercept, ppm_r_value, ppm_p_value, ppm_std_err = stats.linregress(x, ppm)
# This in the linear regression
74         umol_slope, umol_intercept, umol_r_value, umol_p_value, umol_std_err = stats.linregress(x,
umol) # This in the linear regression
75         params, p_value = regression(x, umol, 2)
76         coefficients = np.polyder(params)
77         poly_slope = np.polyval(coefficients, 2)
78         x_fit = np.linspace(x.min(), x.max(), 100)
79         y_fit = polynomial_func(x_fit, *params)
80         # r_value, p_value, _, _, _ = stats.linregress(x, y_fit)
81
82         # y_fit_linear = np.polyval([umol_slope, umol_intercept], x_fit) #Uncomment this section to see
fits
83         # plt.scatter(x, umol, marker="x", color="black", label="Data points")
84         # plt.plot(x_fit, y_fit, label="Polynomial fit")
85         # plt.plot(x_fit, y_fit_linear, label="Linear fit")
86         # plt.legend()
87         # plt.show()
88
89         if ppm_p_value < 0.05: # This loop tests significance. Will be used to remove rates if they are
not significant
90             p = 'significant'
91         else:
92             p = 'not significant'
93         if poly_slope < 0:
94             flux = poly_slope/area_of_soil_cylinder_cm2
95         else:
96             flux = poly_slope/area_of_soil_cylinder_cm2

```

```

97     dictionary = {'Bottle': i, 'Date': date, 'PPM/s': ppm_slope, 'umol/s': umol_slope, 'umol_cm2_s':
flux, 'p': ppm_p_value, 'significance': p} # We create a dictionary only of important numbers
98     list_of_dicts.append(dictionary)
99     flux_df = pd.DataFrame(list_of_dicts)
100    flux_df.to_csv('./Speciale/Third_WD_rate_pr_bottle/'+date, index=False)
101    return(flux_df)
102
103 calculate_flux = rate_calculator('./Speciale/Second_WD_sorted/'+date)
104 print("3/6: Rates of change have been calculated")
105
106 def rate_combiner(file, date):
107     df = df_maker2(file) # We load the file into a dataframe
108     list_of_dicts = [] # We create a list where we will fill in dictionaries, these are quick to compute
and requires less computer power
109     list_of_treatments = [[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12], [13, 14, 15], [16, 17, 18], [19, 20, 21],
[22, 23, 24], [25, 26, 27], [28]] # This is a list, where each element is a list of bottles with the same
treatment
110     for i in list_of_treatments:
111         working_df = df[df['Bottle'].isin(i)] # We select all rows which have bottle-number in the same
treatment
112         avg_ppm = working_df['PPM/s'].mean() # We average the ppm/s
113         std_ppm = working_df['PPM/s'].std() # We estimate std of ppm/s
114         avg_umol = working_df['umol/s'].mean()
115         std_umol = working_df['umol/s'].std()
116         avg_flux = working_df['umol_cm2_s'].mean() # We estimate the average flux for each
treatment
117         std_flux = working_df['umol_cm2_s'].std() # We estimate the standard deviation for the
average flux for each treatment
118         if 1 in i: # All below is only labelling the treatments
119             treatment = "HW"
120         elif 4 in i:
121             treatment = "HD"
122         elif 7 in i:
123             treatment = "HA"
124         elif 10 in i:
125             treatment = "LW"
126         elif 13 in i:
127             treatment = "LD"
128         elif 16 in i:
129             treatment = "AA"
130         elif 19 in i:
131             treatment = "AW"
132         elif 22 in i:
133             treatment = "AD"
134         elif 25 in i:
135             treatment = "LA"
136         else:
137             treatment = "W"
138         dictionary = {"Date": date, 'Treatment': treatment, "avg_PPM/s": avg_ppm, "std_PPM/s":
std_ppm, "avg_umol/s": avg_umol, "std_umol/s": std_umol,
139                     "avg_flux_umol_cm2_s": avg_flux, "std_flux": std_flux}
140         list_of_dicts.append(dictionary)
141     combined_flux = pd.DataFrame(list_of_dicts)
142     combined_flux.to_csv('./Speciale/Fourth_WD_flux_pr_day/" + date, index=False)
143     return(combined_flux)
144

```

```

145 combine = rate_combiner('../Speciale/Third_WD_rate_pr_bottle/'+date, date)
146
147 print("4/6: Mean flux for each treatment have been calculated")
148
149 def combine_all_rates(list_of_files):
150     final_df = pd.DataFrame()
151     for f in list_of_files:
152         df = df_maker2("../Speciale/Fourth_WD_flux_pr_day/" + f[0])
153         df['24h'] = f[1]
154         final_df = pd.concat([final_df, df])
155     final_df.to_csv("../Speciale/Output_files/Combined", index=False)
156     return(final_df)
157
158 list_of_dates = [['2022-11-09', 1], ['2022-11-10', 1], ['2022-11-11', 1], ['2022-11-12', 1], ['2022-11-13',
159 1], ['2022-11-14', 1],
160 ['2022-11-15', 1], ['2022-11-16', 1], ['2022-11-17', 1], ['2022-11-18', 1], ['2022-11-19', 1],
161 ['2022-11-20', 1],
162 ['2022-11-21', 1], ['2022-11-22', 1], ['2022-11-23', 1], ['2022-11-24', 1], ['2022-11-25', 1.5],
163 ['2022-11-27', 2],
164 ['2022-11-29', 1.5], ['2022-11-30', 2], ['2022-12-03', 3], ['2022-12-07', 3], ['2022-12-10', 3],
165 ['2022-12-13', 3],
166 ['2022-12-16', 3], ['2022-12-19', 3], ['2022-12-22', 3.5], ['2022-12-26', 5.5], ['2023-01-02',
167 7], ['2023-01-09', 7],
168 ['2023-01-16', 6.5], ['2023-01-22', 6.5], ['2023-01-22', 11], ['2023-02-07', 9.5]]
169 combine_all = combine_all_rates(list_of_dates)
170
171 print("5/6: All average fluxes have been combined to one file")
172
173 def combine_bottles(list_of_files):
174     final_df = pd.DataFrame()
175     for l in list_of_files:
176         df = df_maker2("../Speciale/Third_WD_rate_pr_bottle/" + l[0])
177         df['24h'] = l[1]
178         final_df = pd.concat([final_df, df])
179     final_df = final_df[['Date', 'Bottle', 'PPM/s', 'umol/s', 'umol_cm2_s', 'p', 'significance', '24h']]
180     final_df.to_csv("../Speciale/Output_files/Combined_bottles", index=False)
181     return(final_df)
182
183 bottles = combine_bottles(list_of_dates)
184
185 print("6/6: All fluxes for each bottle have been combined to one file")
186 executionTime = (time.time() - startTime)
187 print('Execution time in seconds: ' + str(round(executionTime, 2)))
188

```

Calculation_bottles.py

```
1 import pandas as pd
2 import warnings
3 import time
4 warnings.filterwarnings(action='ignore', category=RuntimeWarning)
5 pd.set_option("display.max_rows", None, "display.max_columns", None)
6 desired_width = 320
7 pd.set_option('display.width', desired_width)
8 startTime = time.time()
9 def df_maker(filename, n=None):
10     """This function creates a dataframe from a .csv-file or .txt-file, with tabs as delimiter"""
11     with open(filename) as file:
12         df = pd.DataFrame(pd.read_csv(file, delimiter='\t', skiprows=n))
13     return(df)
14
15 def df_maker2(filename, n=None):
16     """This function creates a dataframe from a .csv-file or .txt-file, with comma as delimiter"""
17     with open(filename) as file:
18         df = pd.DataFrame(pd.read_csv(file, delimiter=',', skiprows=n))
19     return(df)
20
21 def df_maker3(filename, n=None):
22     """This function creates a dataframe from a .csv-file or .txt-file, with semi-colon as delimiter"""
23     with open(filename) as file:
24         df = pd.DataFrame(pd.read_csv(file, delimiter=';', skiprows=n))
25     return(df)
26
27 bottle_df = df_maker2('./Speciale/Output_files/Computed_bottles.csv')
28 print(bottle_df.columns)
29 list_of_dates = list(bottle_df.Date.unique())
30 list_of_bottle_pairs = [[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12], [13, 14, 15],
31                        [16, 17, 18], [19, 20, 21], [22, 23, 24], [25, 26, 27]]
32 list_of_dicts = []
33
34 for d in list_of_dates:
35     working_df = bottle_df[bottle_df["Date"] == d]
36     for i in list_of_bottle_pairs:
37         working_df_2 = working_df[working_df["Bottle"].isin(i)]
38         X24h = working_df_2.iloc[0]["X24h"]
39         avg_umol_N2O_cm2_s = working_df_2["umol_cm2_s"].mean()
40         std_umol_N2O_cm2_s = working_df_2["umol_cm2_s"].std()
41         avg_mg_N2O_cm2_h = working_df_2["mg_N2O_cm2_h"].mean()
42         std_mg_N2O_cm2_h = working_df_2["mg_N2O_cm2_h"].std()
43         avg_g_N2O_m2_d = working_df_2["g_N2O_m2_d"].mean()
44         std_g_N2O_m2_d = working_df_2["g_N2O_m2_d"].std()
45         avg_mg_N2O_m2_d = working_df_2["mg_N2O_m2_d"].mean()
46         std_mg_N2O_m2_d = working_df_2["mg_N2O_m2_d"].std()
47         avg_kg_N2O_ha_d = working_df_2["kg_N2O_ha_d"].mean()
48         std_kg_N2O_ha_d = working_df_2["kg_N2O_ha_d"].std()
49         avg_mg_N2O_N_cm2_h = working_df_2["mg_N2O_N_cm2_h"].mean()
50         std_mg_N2O_N_cm2_h = working_df_2["mg_N2O_N_cm2_h"].std()
51         avg_mg_N2O_N_m2_d = working_df_2["mg_N2O_N_m2_d"].mean()
52         std_mg_N2O_N_m2_d = working_df_2["mg_N2O_N_m2_d"].std()
53         avg_g_N2O_N_m2_d = working_df_2["g_N2O_N_m2_d"].mean()
```

```

54     std_g_N2O_N_m2_d = working_df_2['g_N2O_N_m2_d'].std()
55     avg_kg_N2O_N_ha_d = working_df_2['kg_N2O_N_ha_d'].mean()
56     std_kg_N2O_N_ha_d = working_df_2['kg_N2O_N_ha_d'].std()
57     avg_mg_N2O_N_d = working_df_2['mg_N2O_N_d'].mean()
58     std_mg_N2O_N_d = working_df_2['mg_N2O_N_d'].std()
59     avg_cumulative_mg_N2O_N = working_df_2['cumulative'].mean()
60     std_cumulative_mg_N2O_N = working_df_2['cumulative'].std()
61     avg_cumulative_mg_N2O_N_m2 = working_df_2['cumulative_m2'].mean()
62     std_cumulative_mg_N2O_N_m2 = working_df_2['cumulative_m2'].std()
63     if 1 in i:         # All below is only labelling the treatments
64         treatment = "HW"
65     elif 4 in i:
66         treatment = "HD"
67     elif 7 in i:
68         treatment = "HA"
69     elif 10 in i:
70         treatment = "LW"
71     elif 13 in i:
72         treatment = "LD"
73     elif 16 in i:
74         treatment = "AA"
75     elif 19 in i:
76         treatment = "AW"
77     elif 22 in i:
78         treatment = "AD"
79     elif 25 in i:
80         treatment = "LA"
81     else:
82         treatment = "W"
83     dictionary = {'Date': d, 'Treatment': treatment, 'X24h': X24h,
84                 'avg_umol_N2O_cm2_s': avg_umol_N2O_cm2_s, 'std_umol_N2O_cm2_s':
std_umol_N2O_cm2_s,
85                 'avg_mg_N2O_cm2_s': avg_mg_N2O_cm2_h, 'std_mg_N2O_cm2_s':
std_mg_N2O_cm2_h,
86                 'avg_g_N2O_m2_d': avg_g_N2O_m2_d, 'std_g_N2O_m2_d': std_g_N2O_m2_d,
87                 'avg_mg_N2O_m2_d': avg_mg_N2O_m2_d, 'std_mg_N2O_m2_d':
std_mg_N2O_m2_d,
88                 'avg_kg_N2O_ha_d': avg_kg_N2O_ha_d, 'std_kg_N2O_ha_d': std_kg_N2O_ha_d,
89                 'avg_mg_N2O_N_cm2_h': avg_mg_N2O_N_cm2_h, 'std_mg_N2O_N_cm2_h':
std_mg_N2O_N_cm2_h,
90                 'avg_mg_N2O_N_m2_d': avg_mg_N2O_N_m2_d, 'std_mg_N2O_N_m2_d':
std_mg_N2O_N_m2_d,
91                 'avg_g_N2O_N_m2_d': avg_g_N2O_N_m2_d, 'std_g_N2O_N_m2_d':
std_g_N2O_N_m2_d,
92                 'avg_kg_N2O_N_ha_d': avg_kg_N2O_N_ha_d, 'std_kg_N2O_N_ha_d':
std_kg_N2O_N_ha_d,
93                 'avg_mg_N2O_N_d': avg_mg_N2O_N_d, 'std_mg_N2O_N_d': std_mg_N2O_N_d,
94                 'avg_cumulative_mg_N2O_N': avg_cumulative_mg_N2O_N,
'avg_cumulative_mg_N2O_N_m2': avg_cumulative_mg_N2O_N_m2,
'std_cumulative_mg_N2O_N': std_cumulative_mg_N2O_N,
'std_cumulative_mg_N2O_N_m2': std_cumulative_mg_N2O_N_m2}
95     list_of_dicts.append(dictionary)
96
97
98     combined_final = pd.DataFrame(list_of_dicts)
99     combined_final.to_csv("../Speciale/Output_files/Combined_final", index=False)
100

```

```
101 print("1/1: Fluxes for each bottle have been estimated and statistics are now possible")
102 executionTime = (time.time() - startTime)
103 print('Execution time in seconds: ' + str(round(executionTime, 2)))
104
```

Plotting_lab.py

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import os
5 pd.set_option("display.max_rows", None, "display.max_columns", None)
6 desired_width = 320
7 pd.set_option('display.width', desired_width)
8
9 def df_maker(filename, n=None):
10     """This function creates a dataframe from a .csv-file or .txt-file, with tabs as delimiter"""
11     with open(filename) as file:
12         df = pd.DataFrame(pd.read_csv(file, delimiter='\t', skiprows=n))
13     return(df)
14
15 def df_maker2(filename, n=None):
16     """This function creates a dataframe from a .csv-file or .txt-file, with comma as delimiter"""
17     with open(filename) as file:
18         df = pd.DataFrame(pd.read_csv(file, delimiter=',', skiprows=n))
19     return(df)
20
21 def df_maker3(filename, n=None):
22     """This function creates a dataframe from a .csv-file or .txt-file, with semi-colon as delimiter"""
23     with open(filename) as file:
24         df = pd.DataFrame(pd.read_csv(file, delimiter=';', skiprows=n))
25     return(df)
26
27 list_of_dates = sorted(os.listdir('../Speciale/Second_WD_sorted/'))
28 for d in list_of_dates:
29     date = d
30     df = df_maker2('../Speciale/Second_WD_sorted/' + date, 0)
31     list_of_bottles = sorted(list(df['Bottle'].unique()))
32     df_1 = df[df['Bottle'] == 1]
33     df_2 = df[df['Bottle'] == 2]
34     df_3 = df[df['Bottle'] == 3]
35     df_4 = df[df['Bottle'] == 4]
36     df_5 = df[df['Bottle'] == 5]
37     df_6 = df[df['Bottle'] == 6]
38     df_7 = df[df['Bottle'] == 7]
39     df_8 = df[df['Bottle'] == 8]
40     df_9 = df[df['Bottle'] == 9]
41     df_10 = df[df['Bottle'] == 10]
42     df_11 = df[df['Bottle'] == 11]
43     df_12 = df[df['Bottle'] == 12]
44     df_13 = df[df['Bottle'] == 13]
45     df_14 = df[df['Bottle'] == 14]
46     df_15 = df[df['Bottle'] == 15]
47     df_16 = df[df['Bottle'] == 16]
48     df_17 = df[df['Bottle'] == 17]
49     df_18 = df[df['Bottle'] == 18]
50     df_19 = df[df['Bottle'] == 19]
51     df_20 = df[df['Bottle'] == 20]
52     df_21 = df[df['Bottle'] == 21]
53     df_22 = df[df['Bottle'] == 22]
```

```
54 df_23 = df[df['Bottle'] == 23]
55 df_24 = df[df['Bottle'] == 24]
56 df_25 = df[df['Bottle'] == 25]
57 df_26 = df[df['Bottle'] == 26]
58 df_27 = df[df['Bottle'] == 27]
59
60 x1 = np.array(df_1.index.values.tolist())
61 y1 = np.array(df_1['N2O_ppm'])
62 x2 = np.array(df_2.index.values.tolist())
63 y2 = np.array(df_2['N2O_ppm'])
64 x3 = np.array(df_3.index.values.tolist())
65 y3 = np.array(df_3['N2O_ppm'])
66 x4 = np.array(df_4.index.values.tolist())
67 y4 = np.array(df_4['N2O_ppm'])
68 x5 = np.array(df_5.index.values.tolist())
69 y5 = np.array(df_5['N2O_ppm'])
70 x6 = np.array(df_6.index.values.tolist())
71 y6 = np.array(df_6['N2O_ppm'])
72 x7 = np.array(df_7.index.values.tolist())
73 y7 = np.array(df_7['N2O_ppm'])
74 x8 = np.array(df_8.index.values.tolist())
75 y8 = np.array(df_8['N2O_ppm'])
76 x9 = np.array(df_9.index.values.tolist())
77 y9 = np.array(df_9['N2O_ppm'])
78 x10 = np.array(df_10.index.values.tolist())
79 y10 = np.array(df_10['N2O_ppm'])
80 x11 = np.array(df_11.index.values.tolist())
81 y11 = np.array(df_11['N2O_ppm'])
82 x12 = np.array(df_12.index.values.tolist())
83 y12 = np.array(df_12['N2O_ppm'])
84 x13 = np.array(df_13.index.values.tolist())
85 y13 = np.array(df_13['N2O_ppm'])
86 x14 = np.array(df_14.index.values.tolist())
87 y14 = np.array(df_14['N2O_ppm'])
88 x15 = np.array(df_15.index.values.tolist())
89 y15 = np.array(df_15['N2O_ppm'])
90 x16 = np.array(df_16.index.values.tolist())
91 y16 = np.array(df_16['N2O_ppm'])
92 x17 = np.array(df_17.index.values.tolist())
93 y17 = np.array(df_17['N2O_ppm'])
94 x18 = np.array(df_18.index.values.tolist())
95 y18 = np.array(df_18['N2O_ppm'])
96 x19 = np.array(df_19.index.values.tolist())
97 y19 = np.array(df_19['N2O_ppm'])
98 x20 = np.array(df_20.index.values.tolist())
99 y20 = np.array(df_20['N2O_ppm'])
100 x21 = np.array(df_21.index.values.tolist())
101 y21 = np.array(df_21['N2O_ppm'])
102 x22 = np.array(df_22.index.values.tolist())
103 y22 = np.array(df_22['N2O_ppm'])
104 x23 = np.array(df_23.index.values.tolist())
105 y23 = np.array(df_23['N2O_ppm'])
106 x24 = np.array(df_24.index.values.tolist())
107 y24 = np.array(df_24['N2O_ppm'])
108 x25 = np.array(df_25.index.values.tolist())
109 y25 = np.array(df_25['N2O_ppm'])
```

```
110 x26 = np.array(df_26.index.values.tolist())
111 y26 = np.array(df_26['N2O_ppm'])
112 x27 = np.array(df_27.index.values.tolist())
113 y27 = np.array(df_27['N2O_ppm'])
114
115 fig, axs = plt.subplots(3, 9, figsize=(30, 7))
116 fig.tight_layout(pad=5.0)
117 axs[0, 0].plot(x1, y1)
118 axs[0, 0].title.set_text('df_1')
119 axs[1, 0].plot(x2, y2)
120 axs[1, 0].title.set_text('df_2')
121 axs[2, 0].plot(x3, y3)
122 axs[2, 0].title.set_text('df_3')
123 axs[0, 1].plot(x4, y4)
124 axs[0, 1].title.set_text('df_4')
125 axs[1, 1].plot(x5, y5)
126 axs[1, 1].title.set_text('df_5')
127 axs[2, 1].plot(x6, y6)
128 axs[2, 1].title.set_text('df_6')
129 axs[0, 2].plot(x7, y7)
130 axs[0, 2].title.set_text('df_7')
131 axs[1, 2].plot(x8, y8)
132 axs[1, 2].title.set_text('df_8')
133 axs[2, 2].plot(x9, y9)
134 axs[2, 2].title.set_text('df_9')
135 axs[0, 3].plot(x10, y10)
136 axs[0, 3].title.set_text('df_10')
137 axs[1, 3].plot(x11, y11)
138 axs[1, 3].title.set_text('df_11')
139 axs[2, 3].plot(x12, y12)
140 axs[2, 3].title.set_text('df_12')
141 axs[0, 4].plot(x13, y13)
142 axs[0, 4].title.set_text('df_13')
143 axs[1, 4].plot(x14, y14)
144 axs[1, 4].title.set_text('df_14')
145 axs[2, 4].plot(x15, y15)
146 axs[2, 4].title.set_text('df_15')
147 axs[0, 5].plot(x16, y16)
148 axs[0, 5].title.set_text('df_16')
149 axs[1, 5].plot(x17, y17)
150 axs[1, 5].title.set_text('df_17')
151 axs[2, 5].plot(x18, y18)
152 axs[2, 5].title.set_text('df_18')
153 axs[0, 6].plot(x19, y19)
154 axs[0, 6].title.set_text('df_19')
155 axs[1, 6].plot(x20, y20)
156 axs[1, 6].title.set_text('df_20')
157 axs[2, 6].plot(x21, y21)
158 axs[2, 6].title.set_text('df_21')
159 axs[0, 7].plot(x22, y22)
160 axs[0, 7].title.set_text('df_22')
161 axs[1, 7].plot(x23, y23)
162 axs[1, 7].title.set_text('df_23')
163 axs[2, 7].plot(x24, y24)
164 axs[2, 7].title.set_text('df_24')
165 axs[0, 8].plot(x25, y25)
```

```
166  axs[0, 8].title.set_text('df_25')
167  axs[1, 8].plot(x26, y26)
168  axs[1, 8].title.set_text('df_26')
169  axs[2, 8].plot(x27, y27)
170  axs[2, 8].title.set_text('df_27')
171  fig.suptitle(str(date))
172  plt.show()
173
```

Bottles.R

```
1 install.packages("dplyr")
2 install.packages("tidyr")
3 install.packages("ggplot2")
4 install.packages("readxl")
5 install.packages("tidypaleo")
6 install.packages("tidyverse")
7 install.packages("gasfluxes")
8 install.packages("VGAM")
9 install.packages("pracma")
10 install.packages("egg")
11 install.packages("ggpubr")
12 install.packages("reshape2")
13 install.packages("patchwork")
14 install.packages("cowplot")
15 install.packages("Hmisc")
16 install.packages("mmtable2")
17 install.packages("knitr")
18
19 library(gridExtra)
20 library(dplyr)
21 library(tidyr)
22 library(ggplot2)
23 library(readxl)
24 library(gasfluxes)
25 library(tidyverse)
26 library(multcompView)
27 library(multcomp)
28 library(multcompLetters)
29 library(tidypaleo)
30 library(pracma)
31 library(egg)
32 library(ggpubr)
33 library(reshape2)
34 library(data.table)
35 library(patchwork)
36 library(cowplot)
37 library(emmeans)
38 library(extrafont)
39 my_theme <- theme_article() +
40   theme(text = element_text(family = "EB Garamond"))
41
42 # This is for loading the data and converting columns to the right format
43 calculation_string <- "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/Python/Speciale/Output_files/"
44 calculation_name <- "Combined_bottles"
45 calculation_file_location <- paste(calculation_string, calculation_name, sep="")
46 calculation_data_from_python <- read.csv(calculation_file_location, header=TRUE)
47 calculation_data_from_python <- na.omit(calculation_data_from_python)
48 calculation_data_from_python$Date <- as.Date(calculation_data_from_python$Date, format = "%Y-
%m-%d")
49
50 # Below is the conversion to the right units
```

```

51 calculation_data_from_python$umol_cm2_s <- ifelse(calculation_data_from_python$umol_cm2_s
== 0, 3.391683e-11, calculation_data_from_python$umol_cm2_s)
52 calculation_data_from_python['X24h_2'] <- calculation_data_from_python['X24h']
53 calculation_data_from_python$X24h_2[calculation_data_from_python$Date == '2023-01-09'] <- 3.5
54 calculation_data_from_python['mg_N2O_cm2_h'] <- (((calculation_data_from_python[['umol_cm2_s']]
* 3600)/1000000) * 44.013) * 1000
55 calculation_data_from_python['g_N2O_m2_d'] <-
(calculation_data_from_python['mg_N2O_cm2_h']/1000) * 10000 * 24
56 calculation_data_from_python['mg_N2O_m2_d'] <- calculation_data_from_python['g_N2O_m2_d'] *
1000
57 calculation_data_from_python['kg_N2O_ha_d'] <-
(calculation_data_from_python['g_N2O_m2_d']/1000) * 10000
58 calculation_data_from_python['mg_N2O_N_cm2_h'] <-
calculation_data_from_python['mg_N2O_cm2_h'] * (28.014/44.013)
59 calculation_data_from_python['g_N2O_N_m2_d'] <- calculation_data_from_python['g_N2O_m2_d'] *
(28.014/44.013)
60 calculation_data_from_python['mg_N2O_N_m2_d'] <-
calculation_data_from_python['g_N2O_N_m2_d'] * 1000
61 calculation_data_from_python['kg_N2O_N_ha_d'] <- calculation_data_from_python['kg_N2O_ha_d'] *
(28.014/44.013)
62 calculation_data_from_python['mg_N2O_N_d'] <-
calculation_data_from_python['mg_N2O_N_cm2_h'] * 24 * calculation_data_from_python['X24h'] *
(7/2)**2 * pi
63 calculation_data_from_python['mg_N2O_N_m2_d_total'] <-
calculation_data_from_python['mg_N2O_N_cm2_h'] * 24 * calculation_data_from_python['X24h_2'] *
10000
64
65 # Below is the calculation of the cumulated data
66 calculation_data_from_python <- calculation_data_from_python %>%
67   group_by(Bottle) %>%
68   mutate(cumulative = cumsum(mg_N2O_N_d))
69 calculation_data_from_python <- calculation_data_from_python %>%
70   group_by(Bottle) %>%
71   mutate(cumulative_m2 = cumsum(mg_N2O_N_m2_d_total))
72
73 # This saves the entire data file to a nice .csv file which can be used for many things
74 write.csv(calculation_data_from_python, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/Python/Speciale/Output_files/Computed_bottles.csv", row.names=FALSE)
75
76 # The proper treatment names are added
77 calculation_data_from_python['Treatment'] = ""
78 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 1 | Bottle
== 2 | Bottle == 3] <- 'HW')
79 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 4 | Bottle
== 5 | Bottle == 6] <- 'HD')
80 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 7 | Bottle
== 8 | Bottle == 9] <- 'HA')
81 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 10 |
Bottle == 11 | Bottle == 12] <- 'LW')
82 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 13 |
Bottle == 14 | Bottle == 15] <- 'LD')
83 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 16 |
Bottle == 17 | Bottle == 18] <- 'AA')
84 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 19 |
Bottle == 20 | Bottle == 21] <- 'AW')

```

```

85 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 22 |
Bottle == 23 | Bottle == 24] <- 'AD')
86 calculation_data_from_python <- within(calculation_data_from_python, Treatment[Bottle == 25 |
Bottle == 26 | Bottle == 27] <- 'LA')
87 calculation_data_from_python <-
calculation_data_from_python[!(calculation_data_from_python$Bottle == 28), ]
88 date_to_examine <- '2022-11-16'
89 stat_df <- calculation_data_from_python[calculation_data_from_python$Date == date_to_examine, ]
90 stat_df <- select(stat_df, Treatment, Bottle, Date, cumulative)
91
92 # We summarise the cumulative emissions
93 group_by(stat_df, Treatment) %>%
94   summarise(
95     count = n(),
96     mean = mean(cumulative, na.rm = TRUE),
97     sd = sd(cumulative, na.rm = TRUE)
98   )
99
100 # Here I pairwise use t-test to evaluate the difference between the emission factors of each
treatment
101 list_of_treatments <- list('AD', 'AA', 'AW', 'HD', 'HA', 'HW', 'LD', 'LA', 'LW')
102 p_list <- data.frame(Treatment1 = c(), Treatment2 = c(), p = c())
103 for (t in list_of_treatments){
104   working_df0 <- calculation_data_from_python[calculation_data_from_python$Date ==
date_to_examine,]
105   working_df1 <- working_df0[working_df0$Treatment == t,]
106   for (j in list_of_treatments){
107     working_df2 <- working_df0[working_df0$Treatment == j,]
108     if (t == j){
109       } else {
110         t1_cumulative <- working_df1[, "cumulative_m2"]
111         t2_cumulative <- working_df2[, "cumulative_m2"]
112         results <- t.test(t1_cumulative, t2_cumulative, var.equal = FALSE)
113         p_value <- results$p.value
114         tuple <- c(t, j, p_value)
115         p_list <- rbind(p_list, tuple)
116       }
117     }
118   }
119
120 t1 <- c(132, 438, 126)
121 t2 <- c(0.32, 0.97, 3.79)
122
123 t.test(t1, t2, var.equal = FALSE)
124
125 #Examine specific period, not used
126 folder_name <- "~/Documents/MacBook 2020/Geografi KU/Speiale/Data/R/Week_results/"
127 period_start <- "2022-11-09"
128 period_end <- "2023-01-09" # Enter last date to get entire period
129 period_df <- subset(calculation_data_from_python, Date >= period_start & Date <= period_end)
130 length_period <- round(sum(period_df[period_df$Bottle == 1,]$X24h_2))
131 period_df['mg_N2O_N_lost_m2'] <- period_df['cumulative_m2']
132 keeps <- c("Date", "Bottle", "X24h", "mg_N2O_N_m2_d", "mg_N2O_N_lost_m2", "Treatment")
133 period_df <- period_df[keeps]
134 max_df <- subset(period_df, Date == period_end)
135

```

```

136 mean_loss <- aggregate(max_df[mg_N2O_N_lost_m2], list(max_df$Treatment), mean)
137 mean_loss <- mean_loss %>%
138   rename(mg_N2O_N_lost_mean_m2 = mg_N2O_N_lost_m2)
139 std_loss <- aggregate(max_df[mg_N2O_N_lost_m2], list(max_df$Treatment), std)
140 std_loss <- std_loss %>%
141   rename(mg_N2O_N_lost_std_m2 = mg_N2O_N_lost_m2)
142 obs_loss <- aggregate(max_df[mg_N2O_N_lost_m2], list(max_df$Treatment), length)
143 obs_loss <- obs_loss %>%
144   rename(mg_N2O_N_lost_obs_m2 = mg_N2O_N_lost_m2)
145
146 period_summary <- merge(mean_loss, std_loss, by = "Group.1")
147 period_summary <- merge(period_summary, obs_loss, by = "Group.1")
148 period_summary <- period_summary %>%
149   rename("Treatment" = "Group.1")
150
151 period_summary2 <- period_df %>% group_by(Treatment) %>%
152   summarise(mg_N2O_N_m2_d_mean = weighted.mean(mg_N2O_N_m2_d, X24h),
153     mg_N2O_N_m2_d_std = sqrt(Hmisc::wtd.var(mg_N2O_N_m2_d, X24h)),
154     mg_N2O_N_m2_d_obs = length(mg_N2O_N_m2_d))
155
156 period_summary <- merge(period_summary, period_summary2, by = "Treatment")
157 period_summary$Treatment <- factor(period_summary$Treatment, levels = c("LD", "LA", "LW",
"AD", "AA", "AW", "HD", "HA", "HW"))
158 period_summary$input[period_summary$Treatment=="AA"] <- 19.2
159 period_summary$input[period_summary$Treatment=="AD"] <- 19.2
160 period_summary$input[period_summary$Treatment=="AW"] <- 19.2
161 period_summary$input[period_summary$Treatment=="HA"] <- 57.7
162 period_summary$input[period_summary$Treatment=="HD"] <- 57.7
163 period_summary$input[period_summary$Treatment=="HW"] <- 57.7
164 period_summary$input[period_summary$Treatment=="LA"] <- 3.8
165 period_summary$input[period_summary$Treatment=="LD"] <- 3.8
166 period_summary$input[period_summary$Treatment=="LW"] <- 3.8
167 period_summary$percent_loss <-
round(period_summary$mg_N2O_N_lost_mean/period_summary$input * 100, 2)
168
169 period_summary = subset(period_summary, select = -c(mg_N2O_N_lost_obs,
mg_N2O_N_m2_d_obs))
170
171 max_loss <- summarise(period_summary, max(mg_N2O_N_lost_mean))
172 max_std <- summarise(period_summary, max(mg_N2O_N_lost_std))
173 position <- as.numeric(max_loss+max_std)
174 upper_limit <- ceiling(position)
175
176 #plots of total loss and mean flux over the examined period
177 period_loss <- ggplot(aes(x = Treatment, y = mg_N2O_N_lost_mean), data = period_summary) +
178   geom_bar(stat = 'identity') + theme_article() +
179   geom_errorbar(aes(ymin = ifelse(mg_N2O_N_lost_mean-
mg_N2O_N_lost_std<0,0,mg_N2O_N_lost_mean-mg_N2O_N_lost_std), ymax =
mg_N2O_N_lost_mean+mg_N2O_N_lost_std), width = 0.2)+
180   geom_text(aes(label = paste(percent_loss, "%"), y = ifelse(mg_N2O_N_lost_mean +
(mg_N2O_N_lost_std/1) > 5, mg_N2O_N_lost_mean - (mg_N2O_N_lost_std/0.75),
mg_N2O_N_lost_mean + (mg_N2O_N_lost_std/1)), vjust=-1))+
181   ylab(~Total ~N ~released ~as ~N[2]*O ~(mg))+
182   annotate("text", x = 2, y = upper_limit, label = paste("From",period_start,"to", period_end,".",
period_length, "days"))+
183   ylim(0, upper_limit)

```

```

184
185 max_flux <- ceiling(as.numeric(summarise(period_summary,
max(mg_N2O_N_m2_d_mean+mg_N2O_N_m2_d_std))))
186 period_flux <- ggplot(data = period_summary, aes(x = Treatment, y = mg_N2O_N_m2_d_mean)) +
187   geom_point(shape = 15, size = 2) + theme_article() +
188   geom_errorbar(aes(ymin = ifelse(mg_N2O_N_m2_d_mean-
mg_N2O_N_m2_d_std<0,0,mg_N2O_N_m2_d_mean-mg_N2O_N_m2_d_std), ymax =
mg_N2O_N_m2_d_mean+mg_N2O_N_m2_d_std), width = 0.2))+
189   ylab(~Mean ~flux ~over ~the ~duration ~(mg ~N[2]*O ~-N ~m^{~2} ~d^{~1})))+
190   annotate("text", x = 2, y = max_flux, label = paste("From",period_start, "to", period_end,": ",
period_length, "days"))
191
192 #Creating tables for the examined period
193 period_summary_table <- period_summary
194 period_summary_table <- period_summary_table %>%
195   rename("input (mg N)" = "input", "loss (%)" = "percent_loss", "mg N2O-N lost
(avg)"="mg_N2O_N_lost_mean",
196     "mg N2O-N lost (std)"="mg_N2O_N_lost_std", "mg N2O-N lost (obs)"="mg_N2O_N_lost_obs",
197     "mg N2O-N m2 d1 (avg)" = "mg_N2O_N_m2_d_mean", "mg N2O-N m2 d1 (std)" =
"mg_N2O_N_m2_d_std", "mg N2O-N m2 d1 (obs)" = "mg_N2O_N_m2_d_obs")
198 period_summary_table <- as.data.frame(t(period_summary_table))
199 colnames(period_summary_table) <- period_summary$Treatment
200 period_summary_table <- subset(period_summary_table,! (AA=="AA"))
201 period_summary_table <- data.frame(period_summary_table)
202 period_summary_table <- mutate_all(period_summary_table, function(x)
as.numeric(as.character(x)))
203 period_summary_table <- period_summary_table %>%
204   mutate_if(is.numeric, round, digits = 4)
205 period_summary_table
206
207 period_table <- tableGrob(period_summary_table)
208
209 # Cumulative
210 df <- calculation_data_from_python[calculation_data_from_python$Date == '2023-01-02']
211 flux_copy <- calculation_data_from_python
212 flux_copy['max_cumulative'] <- "
213 flux_copy <- within(flux_copy, max_cumulative[Bottle == 1] <- 1749.104)
214 flux_copy <- within(flux_copy, max_cumulative[Bottle == 2] <- 2139.130)
215 flux_copy <- within(flux_copy, max_cumulative[Bottle == 3] <- 808.6125)
216 flux_copy <- within(flux_copy, max_cumulative[Bottle == 4] <- 2.952519)
217 flux_copy <- within(flux_copy, max_cumulative[Bottle == 5] <- 6.466581)
218 flux_copy <- within(flux_copy, max_cumulative[Bottle == 6] <- 5.427279)
219 flux_copy <- within(flux_copy, max_cumulative[Bottle == 7] <- 52.48218)
220 flux_copy <- within(flux_copy, max_cumulative[Bottle == 8] <- 7.363181)
221 flux_copy <- within(flux_copy, max_cumulative[Bottle == 9] <- 100.0195)
222 flux_copy <- within(flux_copy, max_cumulative[Bottle == 10] <- 45.40896)
223 flux_copy <- within(flux_copy, max_cumulative[Bottle == 11] <- 171.9291)
224 flux_copy <- within(flux_copy, max_cumulative[Bottle == 12] <- 66.04378)
225 flux_copy <- within(flux_copy, max_cumulative[Bottle == 13] <- 4.387771)
226 flux_copy <- within(flux_copy, max_cumulative[Bottle == 14] <- 3.566830)
227 flux_copy <- within(flux_copy, max_cumulative[Bottle == 15] <- 1.973524)
228 flux_copy <- within(flux_copy, max_cumulative[Bottle == 16] <- 35.34258)
229 flux_copy <- within(flux_copy, max_cumulative[Bottle == 17] <- 3.934895)
230 flux_copy <- within(flux_copy, max_cumulative[Bottle == 18] <- 68.49925)
231 flux_copy <- within(flux_copy, max_cumulative[Bottle == 19] <- 654.5374)
232 flux_copy <- within(flux_copy, max_cumulative[Bottle == 20] <- 730.0350)

```

```

233 flux_copy <- within(flux_copy, max_cumulative[Bottle == 21] <- 399.0687)
234 flux_copy <- within(flux_copy, max_cumulative[Bottle == 22] <- 2.187777)
235 flux_copy <- within(flux_copy, max_cumulative[Bottle == 23] <- 1.952786)
236 flux_copy <- within(flux_copy, max_cumulative[Bottle == 24] <- 2.215709)
237 flux_copy <- within(flux_copy, max_cumulative[Bottle == 25] <- 2.318660)
238 flux_copy <- within(flux_copy, max_cumulative[Bottle == 26] <- 0.3907410)
239 flux_copy <- within(flux_copy, max_cumulative[Bottle == 27] <- 3.187841)
240
241 flux_copy$max_cumulative <- as.numeric(flux_copy$max_cumulative)
242 flux_copy['percent'] <- flux_copy['cumulative_m2']/flux_copy['max_cumulative'] * 100
243 flux_copy <- flux_copy[flux_copy$Date <= as.Date('2023-01-04'),]
244
245 flux_copy_date <- flux_copy[flux_copy$Date == as.Date('2022-11-22'),]
246
247 comb_flux <-< flux_copy %>%
248   group_by(Date, Treatment) %>%
249   summarise(avg_percent = mean(percent),
250             std_percent = sd(percent))
251
252
253 comb_flux['N_amount'] <- ""
254 comb_flux <- within(comb_flux, N_amount[Treatment == 'LD'|Treatment == 'LA'|Treatment == 'LW'] <-
- 'low N')
255 comb_flux <- within(comb_flux, N_amount[Treatment == 'AD'|Treatment == 'AA'|Treatment == 'AW']
<- 'common N')
256 comb_flux <- within(comb_flux, N_amount[Treatment == 'HD'|Treatment == 'HA'|Treatment == 'HW']
<- 'high N')
257 comb_flux['Wetness'] <- ""
258 comb_flux <- within(comb_flux, Wetness[Treatment == 'LD'|Treatment == 'AD'|Treatment == 'HD'] <-
'Drained,')
259 comb_flux <- within(comb_flux, Wetness[Treatment == 'LA'|Treatment == 'AA'|Treatment == 'HA'] <-
'Field conditions,')
260 comb_flux <- within(comb_flux, Wetness[Treatment == 'LW'|Treatment == 'AW'|Treatment == 'HW']
<- 'Saturated,')
261
262 ggplot(aes(x = Date, y = avg_percent), data = comb_flux) +
263   geom_line() + geom_point(shape = 1) + my_theme + facet_grid(N_amount ~ Wetness)
264
265 unique(comb_flux$Date)
266
267 examining_date <- comb_flux[comb_flux$Date == as.Date("2022-11-22"),]
268
269 list_of_treatments <- c("HW", "HD", "HA", "LW", "LD", "LA", "AW", "AD", "AA")
270 p_df <- data.frame(t1 = character(), t2 = character(), p_val = numeric(), stringsAsFactors = FALSE)
271
272 for (t in list_of_treatments) {
273   working_df <- flux_copy_date[flux_copy_date$Treatment == t,]
274   for (j in list_of_treatments) {
275     if (t == j) {
276       next # Skip to the next iteration if t is equal to j
277     } else {
278       working_df2 <- flux_copy_date[flux_copy_date$Treatment == j,]
279       p <- t.test(working_df$percent, working_df2$percent, var.equal = FALSE)$p.value
280       p_df <- rbind(p_df, data.frame(t1 = t, t2 = j, p_val = p))
281     }
282   }

```

```
283 }  
284  
285 p_df <- p_df[!duplicated(p_df$p_val), ]  
286 letters <- multcomp::cld(p_df$p_val, lower = TRUE, upper = TRUE, Letters = letters)  
287  
288  
289 flux_copy_date <- subset(flux_copy_date, select = c(Treatment, percent))  
290
```

R_plotting.R

```
1 install.packages("dplyr")
2 install.packages("tidyr")
3 install.packages("ggplot2")
4 install.packages("readxl")
5 install.packages("tidypaleo")
6 install.packages("tidyverse")
7 install.packages("gasfluxes")
8 install.packages("VGAM")
9 install.packages("pracma")
10 install.packages("egg")
11 install.packages("ggpubr")
12 install.packages("reshape2")
13 install.packages("patchwork")
14 install.packages("cowplot")
15 install.packages("extrafont")
16 install.packages("RColorBrewer")
17
18 library(dplyr)
19 library(tidyr)
20 library(ggplot2)
21 library(readxl)
22 library(gasfluxes)
23 library(tidyverse)
24 library(tidypaleo)
25 library(pracma)
26 library(egg)
27 library(ggpubr)
28 library(reshape2)
29 library(data.table)
30 library(patchwork)
31 library(cowplot)
32 library(extrafont)
33 library(RColorBrewer)
34
35 my_theme <- theme_article() +
36   theme(text = element_text(family = "EB Garamond"))
37
38 # This is for loading the data and converting columns to the right format
39 python_string <- "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/Python/Speciale/Output_files/"
40 name <- "Combined_final"
41 file_location <- paste(python_string, name, sep="")
42 data_from_python <- read.csv(file_location, header=TRUE)
43 data_from_python <- na.omit(data_from_python)
44 data_from_python$Date <- as.Date(data_from_python$Date, format = "%Y-%m-%d")
45 ref_date <- as.Date("2022-11-09")
46 data_from_python$Day_number <- difftime(data_from_python$Date, ref_date, units = 'days')
47 data_from_python$Day_number <- as.integer(data_from_python$Day_number)
48 pd <- position_dodge(width = 0.4)
49
50 # Here I add the proper labels, such that legends will have proper layout
51 data_from_python['N_amount'] <- ""
```

```

52 data_from_python <- within(data_from_python, N_amount[Treatment == 'LD'|Treatment ==
'LA'|Treatment == 'LW'] <- 'low N')
53 data_from_python <- within(data_from_python, N_amount[Treatment == 'AD'|Treatment ==
'AA'|Treatment == 'AW'] <- 'common N')
54 data_from_python <- within(data_from_python, N_amount[Treatment == 'HD'|Treatment ==
'HA'|Treatment == 'HW'] <- 'high N')
55 data_from_python[Wetness] <- ""
56 data_from_python <- within(data_from_python, Wetness[Treatment == 'LD'|Treatment ==
'AD'|Treatment == 'HD'] <- 'Drained,')
57 data_from_python <- within(data_from_python, Wetness[Treatment == 'LA'|Treatment ==
'AA'|Treatment == 'HA'] <- 'Field conditions,')
58 data_from_python <- within(data_from_python, Wetness[Treatment == 'LW'|Treatment ==
'AW'|Treatment == 'HW'] <- 'Saturated,')
59 data_from_python[True_treatment] <- paste(data_from_python$Wetness,
data_from_python$N_amount)
60
61 legend_order <- c("Drained, low N", "Drained, common N", "Drained, high N",
62 "Field conditions, low N", "Field conditions, common N", "Field conditions, high N",
63 "Saturated, low N", "Saturated, common N", "Saturated, high N")
64 data_from_python$True_treatment <- factor(data_from_python$True_treatment, levels =
legend_order)
65
66 # Below are all the plots used in the thesis/paper
67 end_date <- as.Date('2022-11-16') # These dates are to filter out, so that we get the dataframe
that ends in a specific date
68 #end_date <- as.Date('2022-11-23')
69 #end_date <- as.Date('2023-01-04')
70
71 # Here I create new data frames so that I can inspect each water content more easily, also
used for plotting
72 wet_df <- data_from_python[!(data_from_python$Treatment == "HA" | data_from_python$Treatment
== "HD" | data_from_python$Treatment == "AA" | data_from_python$Treatment == "AD" |
data_from_python$Treatment == "LA" | data_from_python$Treatment == "LD"), ]
73 field_df <- data_from_python[!(data_from_python$Treatment == "HW" | data_from_python$Treatment
== "HD" | data_from_python$Treatment == "AW" | data_from_python$Treatment == "AD" |
data_from_python$Treatment == "LW" | data_from_python$Treatment == "LD"), ]
74 drained_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HW" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AW" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LW"), ]
75 wet_df <- wet_df[wet_df$Date <= end_date, ]
76 field_df <- field_df[field_df$Date <= end_date, ]
77 drained_df <- drained_df[drained_df$Date <= end_date, ]
78
79 # These are the plots used in the thesis/paper, fluxes and cumulated values
80 #summary_df <- aggregate(data_from_python$mg_cm2_h, by =
list(data_from_python$Treatment), FUN = "mean")
81 #summary_df['kg_n2o_ha'] <- summary_df['x'] * 100000000 / 1000
82 #summary_df['kg_n2o-n_ha'] <- summary_df['kg_n2o_ha'] * 28/44
83
84 # Figures, combined
85 p1 <- ggplot(data = wet_df, aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group =
True_treatment, color = True_treatment)) +
86 geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d < 0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
87 ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,

```

```

88     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
89   geom_line(position = pd, linewidth = 0.4) +
90   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) +
91   scale_color_manual(values = c("black", "red", "blue")) +
92   my_theme +
93   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
94     axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank(),
legend.title=element_blank(),
95     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
96   annotate("text", x = -Inf, y = Inf, label = "a", size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
97   theme(legend.justification = "left", legend.position = "right")
98
99   p2 <- ggplot(data = field_df, aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group =
True_treatment, color = True_treatment)) +
100   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d < 0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
101     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
102     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
103   geom_line(position = pd, linewidth = 0.4) +
104   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) +
105   scale_color_manual(values = c("black", "red", "blue")) +
106   my_theme +
107   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
108     axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank(),
legend.title=element_blank(),
109     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
110   annotate("text", x = -Inf, y = Inf, label = "b", size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
111   theme(legend.justification = "left", legend.position = "right")
112
113   p3 <- ggplot(data = drained_df, aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group =
True_treatment, color = True_treatment)) +
114   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d < 0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
115     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
116     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
117   geom_line(position = pd, linewidth = 0.4) +
118   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) + xlab(" Time since start (days)") +
119   scale_color_manual(values = c("black", "red", "blue")) +
120   my_theme +
121   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
legend.title=element_blank(),
122     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
123   scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
124   annotate("text", x = -Inf, y = Inf, label = "c", size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
125   theme(legend.justification = "left", legend.position = "right")
126
127   p4 <- ggplot(data = wet_df, aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group =
True_treatment, color = True_treatment)) +
128   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2 < 0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
129     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
130     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
131   geom_line(position = pd, linewidth = 0.4) +
132   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) + xlab(" Time since start (days)") +

```

```

133 scale_color_manual(values = c("black", "red", "blue")) +
134 my_theme +
135 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
136       axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank(),
legend.title=element_blank(),
137       axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
138 scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
139 annotate("text", x = -Inf, y = Inf, label = "a"), size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
140 theme(legend.justification = "left", legend.position = "right")
141
142 p5 <- ggplot(data = field_df, aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group =
True_treatment, color = True_treatment)) +
143   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
144     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
145     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
146   geom_line(position = pd, linewidth = 0.4) +
147   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) +
148   scale_color_manual(values = c("black", "red", "blue")) +
149   my_theme +
150   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
151     axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank(),
legend.title=element_blank(),
152     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
153   scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
154   annotate("text", x = -Inf, y = Inf, label = "b"), size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
155   theme(legend.justification = "left", legend.position = "right")
156
157 p6 <- ggplot(data = drained_df, aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group =
True_treatment, color = True_treatment)) +
158   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
159     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
160     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
161   geom_line(position = pd, linewidth = 0.4) +
162   ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) + xlab("Time since start (days)") +
163   scale_color_manual(values = c("black", "red", "blue")) +
164   my_theme +
165   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
legend.title=element_blank(),
166     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
167   scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
168   annotate("text", x = -Inf, y = Inf, label = "c"), size = 3, hjust = -1, vjust = 2, family = 'EB Garamond') +
169   theme(legend.justification = "left", legend.position = "right")
170
171 plot_grid(p1, p2, p3, ncol = 1, align = 'v', rel_heights =c(1, 1, 1.15))
172 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/lab_fluxes_one_week.png", width = 12, height = 12, units = "cm",
dpi = 300)
173
174 plot_grid(p4, p5, p6, ncol = 1, align = 'v', rel_heights =c(1, 1, 1.15))

```

```

175 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/lab_cumulated_one_week.png", width = 12, height = 12, units =
"cm", dpi = 300)
176
177
178 wet_1_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2022-11-16'), ]
179 wet_2_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2022-11-23'), ]
180 wet_8_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2023-01-04'), ]
181
182 field_1_df <- data_from_python[!(data_from_python$Treatment == "HW" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AW" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LW" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2022-11-16'), ]
183 field_2_df <- data_from_python[!(data_from_python$Treatment == "HW" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AW" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LW" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2022-11-23'), ]
184 field_8_df <- data_from_python[!(data_from_python$Treatment == "HW" |
data_from_python$Treatment == "HD" | data_from_python$Treatment == "AW" |
data_from_python$Treatment == "AD" | data_from_python$Treatment == "LW" |
data_from_python$Treatment == "LD") & data_from_python$Date <= as.Date('2023-01-04'), ]
185
186 dry_1_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HW" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AW" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LW") & data_from_python$Date <= as.Date('2022-11-16'), ]
187 dry_2_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HW" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AW" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LW") & data_from_python$Date <= as.Date('2022-11-23'), ]
188 dry_8_df <- data_from_python[!(data_from_python$Treatment == "HA" |
data_from_python$Treatment == "HW" | data_from_python$Treatment == "AA" |
data_from_python$Treatment == "AW" | data_from_python$Treatment == "LA" |
data_from_python$Treatment == "LW") & data_from_python$Date <= as.Date('2023-01-04'), ]
189
190 # Create the flux plots
191 a <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = dry_1_df) + my_theme +
192   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d < 0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
193     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
194     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
195   geom_line(position = pd, linewidth = 0.4) +
196   xlab("") + ylab(~ mg ~N[2]*O-N ~m^{2} ~d^{-1}) +
197   annotate("text", x = -Inf, y = Inf, label = "a) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +

```

```

198 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
199   axis.title.x=element_blank(),
200   axis.text.x=element_blank(),
201   axis.ticks.x=element_blank(), axis.text=element_text(size=8, color="black"),
axis.title=element_text(size=8, color="black")) +
202   scale_color_manual(values = c("black", "red", "blue"))
203
204 b <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = field_1_df) + my_theme +
205   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
206     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
207     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
208   geom_line(position = pd, linewidth = 0.4) +
209   xlab("") + ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) +
210   annotate("text", x = -Inf, y = Inf, label = "b) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
211 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
212   axis.title.x=element_blank(),
213   axis.text.x=element_blank(),
214   axis.ticks.x=element_blank(), axis.text=element_text(size=8, color="black"),
axis.title=element_text(size=8, color="black")) +
215   scale_color_manual(values = c("black", "red", "blue"))
216
217 c <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = wet_1_df) + my_theme +
218   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
219     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
220     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
221   geom_line(position = pd, linewidth = 0.4) +
222   xlab("Time since start (days)") + ylab(~ mg ~N[2]*O-N ~m^{-2} ~d^{-1}) +
223   annotate("text", x = -Inf, y = Inf, label = "c) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
224 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
225   axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
226   scale_color_manual(values = c("black", "red", "blue"))+
227   scale_x_continuous(breaks = scales::pretty_breaks(n = 10))
228
229 d <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = dry_2_df) + my_theme +
230   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
231     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
232     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
233   geom_line(position = pd, linewidth = 0.4) +
234   xlab("") +
235   annotate("text", x = -Inf, y = Inf, label = "d) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
236 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
237   axis.title.x=element_blank(),
238   axis.text.x=element_blank(),

```

```

239   axis.ticks.x=element_blank(),
240   axis.text.y=element_blank(),
241   axis.ticks.y=element_blank(),
242   axis.title.y = element_blank()) +
243   scale_color_manual(values = c("black", "red", "blue"))
244
245 e <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = field_2_df) + my_theme +
246   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
247     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
248     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
249   geom_line(position = pd, linewidth = 0.4) +
250   xlab("") +
251   annotate("text", x = -Inf, y = Inf, label = "e) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
252   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
253     axis.title.x=element_blank(),
254     axis.text.x=element_blank(),
255     axis.ticks.x=element_blank(),
256     axis.text.y=element_blank(),
257     axis.ticks.y=element_blank(),
258     axis.title.y = element_blank()) +
259   scale_color_manual(values = c("black", "red", "blue"))
260
261 f <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = wet_2_df) + my_theme +
262   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
263     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
264     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
265   geom_line(position = pd, linewidth = 0.4) +
266   xlab("Time since start (days)") +
267   annotate("text", x = -Inf, y = Inf, label = "f) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
268   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
269     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"),
axis.text.y=element_blank(),
270     axis.ticks.y=element_blank(),
271     axis.title.y = element_blank()) +
272   scale_color_manual(values = c("black", "red", "blue")) +
273   scale_x_continuous(breaks = scales::pretty_breaks(n = 7))
274
275 g <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = dry_8_df) + my_theme +
276   geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
277     ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
278     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
279   geom_line(position = pd, linewidth = 0.4) +
280   xlab("") +
281   annotate("text", x = -Inf, y = Inf, label = "g) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +

```

```

282 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
283 axis.title.x=element_blank(),
284 axis.text.x=element_blank(),
285 axis.ticks.x=element_blank(),
286 axis.text.y=element_blank(),
287 axis.ticks.y=element_blank(),
288 axis.title.y = element_blank()) +
289 scale_color_manual(values = c("black", "red", "blue"))
290
291 h <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = field_8_df) + my_theme +
292 geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
293 ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
294 width = .2, linetype = 1, color = "gray", alpha = 0.5) +
295 geom_line(position = pd, linewidth = 0.4) +
296 xlab("") +
297 annotate("text", x = -Inf, y = Inf, label = "h) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
298 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
299 axis.title.x=element_blank(),
300 axis.text.x=element_blank(),
301 axis.ticks.x=element_blank(),
302 axis.text.y=element_blank(),
303 axis.ticks.y=element_blank(),
304 axis.title.y = element_blank()) +
305 scale_color_manual(values = c("black", "red", "blue"))
306
307 i <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = wet_8_df) + my_theme +
308 geom_errorbar(aes(ymin = ifelse(avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d<0, 0,
avg_mg_N2O_N_m2_d - std_mg_N2O_N_m2_d),
309 ymax = avg_mg_N2O_N_m2_d + std_mg_N2O_N_m2_d), position = pd,
310 width = .2, linetype = 1, color = "gray", alpha = 0.5) +
311 geom_line(position = pd, linewidth = 0.4) +
312 xlab("Time since start (days)") +
313 annotate("text", x = -Inf, y = Inf, label = "i) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
314 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
315 axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"),
316 axis.text.y=element_blank(),
317 axis.ticks.y=element_blank(),
318 axis.title.y = element_blank()) +
319 scale_color_manual(values = c("black", "red", "blue")) +
320 scale_x_continuous(breaks = scales::pretty_breaks(n = 7))
321
322 # I create a copy only to get the legend
323 wet_copy <- wet_1_df
324 wet_copy$True_treatment <- gsub("Saturated, high N", "High N ", wet_copy$True_treatment)
325 wet_copy$True_treatment <- gsub("Saturated, low N", "Low N ", wet_copy$True_treatment)
326 wet_copy$True_treatment <- gsub("Saturated, common N", "Common N ",
wet_copy$True_treatment)

```

```

327 i_copy <- ggplot(aes(x = Day_number, y = avg_mg_N2O_N_m2_d, group = True_treatment, color =
True_treatment), data = wet_copy) + my_theme +
328   geom_line(position = pd, linewidth = 0.4) + scale_color_manual(values = c("black", "red", "blue"),
labels = c("Low N ", "Common N ", "High N ")) +
329   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="bottom",
legend.title=element_blank(),
330     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"),
331     axis.text.y=element_blank(),
332     axis.ticks.y=element_blank())
333
334 # Combination of flux-plots
335 legend_t <- get_legend(i_copy)
336 grid <- plot_grid(a, d, g,
337   b, e, h,
338   c, f, i,
339   ncol = 3,
340   rel_widths = c(0.9, 1, 1.2),
341   rel_heights = c(1, 1, 1.15),
342   align="v",
343   axis = "lbt")
344 grid2 <- plot_grid(grid, legend_t, ncol = 1,
345   rel_heights = c(1, 0.1))
346 grid3 <- grid2 + theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position
= 'bottom')
347 ggsave("~/Documents/MacBook 2020/Geografi
KU/Special/Data/R/Figures_for_thesis/lab_fluxes_all.png", width = 15, height = 12, units = "cm", dpi =
300)
348
349 # Cumulated fluxes
350 ac <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = dry_1_df) + my_theme +
351   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
352     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
353     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
354   geom_line(position = pd, linewidth = 0.4) +
355   xlab("") + ylab(~ mg ~N[2]*O-N ~m^{2} ~d^{1}) +
356   annotate("text", x = -Inf, y = Inf, label = "a) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
357   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
358     axis.title.x=element_blank(),
359     axis.text.x=element_blank(),
360     axis.ticks.x=element_blank(), axis.text=element_text(size=8, color="black"),
axis.title=element_text(size=8, color="black")) +
361   scale_color_manual(values = c("black", "red", "blue"))
362
363 bc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = field_1_df) + my_theme +
364   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
365     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,

```

```

366       width = .2, linetype = 1, color = "gray", alpha = 0.5) +
367 geom_line(position = pd, linewidth = 0.4) +
368 xlab("") + ylab(~ mg ~N[2]*O-N ~m^{2} ~d^{-1}) +
369 annotate("text", x = -Inf, y = Inf, label = "b) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
370 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
371       axis.title.x=element_blank(),
372       axis.text.x=element_blank(),
373       axis.ticks.x=element_blank(), axis.text=element_text(size=8, color="black"),
axis.title=element_text(size=8, color="black")) +
374 scale_color_manual(values = c("black", "red", "blue"))
375
376 cc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = wet_1_df) + my_theme +
377 geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
378               ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
379             width = .2, linetype = 1, color = "gray", alpha = 0.5) +
380 geom_line(position = pd, linewidth = 0.4) +
381 xlab("Time since start (days)") + ylab(~ mg ~N[2]*O-N ~m^{2} ~d^{-1}) +
382 annotate("text", x = -Inf, y = Inf, label = "c) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
383 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
384       axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black")) +
385 scale_color_manual(values = c("black", "red", "blue"))+
386 scale_x_continuous(breaks = scales::pretty_breaks(n = 10))
387
388 dc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = dry_2_df) + my_theme +
389 geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
390               ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
391             width = .2, linetype = 1, color = "gray", alpha = 0.5) +
392 geom_line(position = pd, linewidth = 0.4) +
393 xlab("") +
394 annotate("text", x = -Inf, y = Inf, label = "d) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
395 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
396       axis.title.x=element_blank(),
397       axis.text.x=element_blank(),
398       axis.ticks.x=element_blank(),
399       axis.text.y=element_blank(),
400       axis.ticks.y=element_blank(),
401       axis.title.y = element_blank()) +
402 scale_color_manual(values = c("black", "red", "blue"))
403
404 ec <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = field_2_df) + my_theme +

```

```

405 geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
406         ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
407         width = .2, linetype = 1, color = "gray", alpha = 0.5) +
408 geom_line(position = pd, linewidth = 0.4) +
409 xlab("") +
410 annotate("text", x = -Inf, y = Inf, label = "e) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
411 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
412         axis.title.x=element_blank(),
413         axis.text.x=element_blank(),
414         axis.ticks.x=element_blank(),
415         axis.text.y=element_blank(),
416         axis.ticks.y=element_blank(),
417         axis.title.y = element_blank()) +
418 scale_color_manual(values = c("black", "red", "blue"))
419
420 fc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = wet_2_df) + my_theme +
421 geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
422         ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
423         width = .2, linetype = 1, color = "gray", alpha = 0.5) +
424 geom_line(position = pd, linewidth = 0.4) +
425 xlab("Time since start (days)") +
426 annotate("text", x = -Inf, y = Inf, label = "f) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
427 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
428         axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"),
axis.text.y=element_blank(),
429         axis.ticks.y=element_blank(),
430         axis.title.y = element_blank()) +
431 scale_color_manual(values = c("black", "red", "blue")) +
432 scale_x_continuous(breaks = scales::pretty_breaks(n = 7))
433
434 gc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = dry_8_df) + my_theme +
435 geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
436         ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
437         width = .2, linetype = 1, color = "gray", alpha = 0.5) +
438 geom_line(position = pd, linewidth = 0.4) +
439 xlab("") +
440 annotate("text", x = -Inf, y = Inf, label = "g) drained", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
441 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
442         axis.title.x=element_blank(),

```

```

443   axis.text.x=element_blank(),
444   axis.ticks.x=element_blank(),
445   axis.text.y=element_blank(),
446   axis.ticks.y=element_blank(),
447   axis.title.y = element_blank()) +
448   scale_color_manual(values = c("black", "red", "blue"))
449
450 hc <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = field_8_df) + my_theme +
451   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
452     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
453     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
454   geom_line(position = pd, linewidth = 0.4) +
455   xlab("") +
456   annotate("text", x = -Inf, y = Inf, label = "h) field conditions", size = 3, hjust = -0.1, vjust = 2, family =
'EB Garamond') +
457   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
458     axis.title.x=element_blank(),
459     axis.text.x=element_blank(),
460     axis.ticks.x=element_blank(),
461     axis.text.y=element_blank(),
462     axis.ticks.y=element_blank(),
463     axis.title.y = element_blank()) +
464   scale_color_manual(values = c("black", "red", "blue"))
465
466 ic <- ggplot(aes(x = Day_number, y = avg_cumulative_mg_N2O_N_m2, group = True_treatment,
color = True_treatment), data = wet_8_df) + my_theme +
467   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2<0, 0, avg_cumulative_mg_N2O_N_m2 -
std_cumulative_mg_N2O_N_m2),
468     ymax = avg_cumulative_mg_N2O_N_m2 + std_cumulative_mg_N2O_N_m2), position =
pd,
469     width = .2, linetype = 1, color = "gray", alpha = 0.5) +
470   geom_line(position = pd, linewidth = 0.4) +
471   xlab("Time since start (days)") +
472   annotate("text", x = -Inf, y = Inf, label = "i) saturated", size = 3, hjust = -0.2, vjust = 2, family = 'EB
Garamond') +
473   theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="none",
legend.title=element_blank(),
474     axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"),
475     axis.text.y=element_blank(),
476     axis.ticks.y=element_blank(),
477     axis.title.y = element_blank()) +
478   scale_color_manual(values = c("black", "red", "blue")) +
479   scale_x_continuous(breaks = scales::pretty_breaks(n = 7))
480
481 gridc <- plot_grid(ac, dc, gc,
482   bc, ec, hc,
483   cc, fc, ic,
484   ncol = 3,
485   rel_widths = c(0.9, 1, 1.2),
486   rel_heights = c(1, 1, 1.15),

```

```

487         align="v",
488         axis = "lbt")
489 grid2c <- plot_grid(gridc, legend_t, ncol = 1,
490         rel_heights = c(1, 0.1))
491 grid3c <- grid2c + theme(plot.background = element_rect(fill = 'white', colour = 'white'),
legend.position = 'bottom')
492 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/lab_cumulated_fluxes_all.png", width = 15, height = 12, units =
"cm", dpi = 300)
493
494 # Here I inspect the total lost of N2O-N, below is just to make calculation easier
495 last_date <- '2022-11-29'
496 total_loss <- data_from_python[data_from_python$Date == last_date,]
497 total_loss <- select(total_loss, Treatment, Date, avg_cumulative_mg_N2O_N,
std_cumulative_mg_N2O_N)
498 total_loss$input[total_loss$Treatment=='AA'] <- 19.2
499 total_loss$input[total_loss$Treatment=='AD'] <- 19.2
500 total_loss$input[total_loss$Treatment=='AW'] <- 19.2
501 total_loss$input[total_loss$Treatment=='HA'] <- 57.7
502 total_loss$input[total_loss$Treatment=='HD'] <- 57.7
503 total_loss$input[total_loss$Treatment=='HW'] <- 57.7
504 total_loss$input[total_loss$Treatment=='LA'] <- 3.8
505 total_loss$input[total_loss$Treatment=='LD'] <- 3.8
506 total_loss$input[total_loss$Treatment=='LW'] <- 3.8
507 total_loss$samples <- 3
508 total_loss$percent_loss <- total_loss$avg_cumulative_mg_N2O_N/total_loss$input * 100 # Here we
calculate the emission factor for N2O
509 total_loss$Treatment <- factor(total_loss$Treatment, levels = c("LD", "LA", "LW", "AD", "AA", "AW",
"HD", "HA", "HW"))
510 total_loss$percent_loss <- round(total_loss$percent_loss ,digit=2)
511
512 start <- as.Date(min(data_from_python$Date))
513 end <- as.Date(last_date)
514 duration <- length(seq(from =start, to = end, by = "day"))
515
516 ggplot(total_loss, aes(y = avg_cumulative_mg_N2O_N, x = Treatment, label=percent_loss)) +
theme_article() +
517   geom_point(shape = 15, size = 2) +
518   geom_text(aes(label = paste(percent_loss, "%"), y = ifelse(avg_cumulative_mg_N2O_N +
(std_cumulative_mg_N2O_N/1) > 5, avg_cumulative_mg_N2O_N - (std_cumulative_mg_N2O_N/0.75),
avg_cumulative_mg_N2O_N + (std_cumulative_mg_N2O_N/1)), vjust=-1))+
519   geom_errorbar(aes(ymin = ifelse(avg_cumulative_mg_N2O_N - std_cumulative_mg_N2O_N<0, 0,
avg_cumulative_mg_N2O_N - std_cumulative_mg_N2O_N), ymax = avg_cumulative_mg_N2O_N +
std_cumulative_mg_N2O_N), color = 'black', width = 0.2))+
520   annotate("text", x = 2.5, y = 5, label = paste("From",start, "to", end))+annotate("text", x = 1.42, y =
4.5, label = paste("Duration =", duration))+
521   ylab(~Total ~N ~released ~as ~N[2]*O ~(mg))
522 ggsave("~/Documents/MacBook 2020/Geografi KU/Speciale/Presentations/total_loss.png", width
=15, height = 10.5, units = "cm", dpi = 300)
523
524
525 # Plotting til Bo
526 plotting_data <- data_from_python
527 plotting_data <- plotting_data[plotting_data$Date <= as.Date('2023-01-04'),]
528 plotting_data['avg_ug_N2O_N_m2_h'] <- (plotting_data['avg_g_N2O_N_m2_d'] * 1000 * 1000)/24
529 plotting_data['std_ug_N2O_N_m2_h'] <- (plotting_data['std_g_N2O_N_m2_d'] * 1000 * 1000)/24

```

```

530 plotting_data['se_ug_N2O_N_m2_h'] <- plotting_data['std_ug_N2O_N_m2_h']/sqrt(3)
531 keeps <- c('Date', 'Treatment', 'avg_ug_N2O_N_m2_h', 'se_ug_N2O_N_m2_h')
532 plotting_data <- plotting_data[keeps]
533 ref_date <- as.Date('2022-11-09')
534 plotting_data$Day_number <- difftime(plotting_data$Date, ref_date, units = 'days')
535 plotting_data$Day_number <- as.integer(plotting_data$Day_number)
536
537 wet_ug <- plotting_data[!(plotting_data$Treatment == "HA" | plotting_data$Treatment == "HD" |
plotting_data$Treatment == "AA" | plotting_data$Treatment == "AD" | plotting_data$Treatment == "LA" |
plotting_data$Treatment == "LD"), ]
538 wet_ug$Treatment <- factor(wet_ug$Treatment, levels = c('LW', 'AW', 'HW'))
539 not_wet_ug <- plotting_data[!(plotting_data$Treatment == "HW" | plotting_data$Treatment == "AW" |
plotting_data$Treatment == "LW"), ]
540 not_wet_ug$Treatment <- factor(not_wet_ug$Treatment, levels = c('LD', 'LA', 'AD', 'AA', 'HD', 'HA'))
541 drained_ug <- plotting_data[!(plotting_data$Treatment == "HW" | plotting_data$Treatment == "AW" |
plotting_data$Treatment == "LW" | plotting_data$Treatment == "HA" | plotting_data$Treatment == "AA" |
plotting_data$Treatment == "LA"), ]
542 drained_ug$Treatment <- factor(drained_ug$Treatment, levels = c('LD', 'AD', 'HD'))
543 ambient_ug <- plotting_data[!(plotting_data$Treatment == "HW" | plotting_data$Treatment == "AW" |
plotting_data$Treatment == "LW" | plotting_data$Treatment == "HD" | plotting_data$Treatment == "AD" |
plotting_data$Treatment == "LD"), ]
544 ambient_ug$Treatment <- factor(ambient_ug$Treatment, levels = c('LA', 'AA', 'HA'))
545
546 wet <- ggplot(aes(x = Day_number, y = avg_ug_N2O_N_m2_h, group = Treatment, color =
Treatment), data = wet_ug) +
547   geom_line(position = pd) + theme_article() +
548   geom_errorbar(aes(ymin = ifelse(avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h < 0, 0,
avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h), ymax = avg_ug_N2O_N_m2_h +
se_ug_N2O_N_m2_h), position = pd, width = .2, color = 'grey', alpha = 0.5) +
549   xlab(expression("Time since start (days)")) + ylab(~ mu~g ~N[2]*O~N ~m^{2} ~h^{-1})+
550   theme(legend.position = 'bottom') + theme(legend.title = element_blank())+
551   scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
552   scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
553   theme(plot.background = element_rect(fill = 'white', colour = 'white'))
554   ggsave("~/Documents/MacBook 2020/Geografi KU/Speciale/Presentations/wet_mug.png", width =
20, height = 10.5, units = "cm", dpi = 300)
555
556 not_wet <- ggplot(aes(x = Day_number, y = avg_ug_N2O_N_m2_h, group = Treatment, color =
Treatment), data = not_wet_ug) +
557   geom_line(position = pd) + theme_article() +
558   geom_errorbar(aes(ymin = ifelse(avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h < 0, 0,
avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h), ymax = avg_ug_N2O_N_m2_h +
se_ug_N2O_N_m2_h), position = pd, width = .2, color = 'grey', alpha = 0.5) +
559   xlab(expression("Time since start (days)")) + ylab(~ mu~g ~N[2]*O~N ~m^{2} ~h^{-1})+
560   theme(legend.position = 'bottom') + theme(legend.title = element_blank())+
561   scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
562   scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
563   theme(plot.background = element_rect(fill = 'white', colour = 'white'))
564   ggsave("~/Documents/MacBook 2020/Geografi KU/Speciale/Presentations/not_wet_mug.png", width =
20, height = 10.5, units = "cm", dpi = 300)
565
566 drained <- ggplot(aes(x = Day_number, y = avg_ug_N2O_N_m2_h, group = Treatment, color =
Treatment), data = drained_ug) +
567   geom_line(position = pd) + theme_article() +
568   geom_errorbar(aes(ymin = ifelse(avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h < 0, 0,
avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h), ymax = avg_ug_N2O_N_m2_h +

```

```

se_ug_N2O_N_m2_h), position = pd, width = .2, color = 'grey', alpha = 0.5) +
569 xlab(expression('Time since start (days)')) + ylab(~ mu~g ~N[2]*O-N ~m^{-2} ~h^{-1})+
570 theme(legend.position = 'bottom') + theme(legend.title = element_blank()+
571 scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
572 scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
573 theme(plot.background = element_rect(fill = 'white', colour = 'white'))
574 ggsave("~/Documents/MacBook 2020/Geografi KU/Speciale/Presentations/drained_mug.png", width
= 20, height = 10.5, units = "cm", dpi = 300)
575
576 ambient <- ggplot(aes(x = Day_number, y = avg_ug_N2O_N_m2_h, group = Treatment, color =
Treatment), data = ambient_ug) +
577 geom_line(position = pd) + theme_article() +
578 geom_errorbar(aes(ymin = ifelse(avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h < 0, 0,
avg_ug_N2O_N_m2_h - se_ug_N2O_N_m2_h), ymax = avg_ug_N2O_N_m2_h +
se_ug_N2O_N_m2_h), position = pd, width = .2, color = 'grey', alpha = 0.5) +
579 xlab(expression('Time since start (days)')) + ylab(~ mu~g ~N[2]*O-N ~m^{-2} ~h^{-1})+
580 theme(legend.position = 'bottom') + theme(legend.title = element_blank()+
581 scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
582 scale_y_continuous(breaks = scales::pretty_breaks(n = 5)) +
583 theme(plot.background = element_rect(fill = 'white', colour = 'white'))
584 ggsave("~/Documents/MacBook 2020/Geografi KU/Speciale/Presentations/ambient_mug.png", width
= 20, height = 10.5, units = "cm", dpi = 300)
585

```

R_Imer.R

```
1 install.packages("dplyr")
2 install.packages("tidyr")
3 install.packages("ggplot2")
4 install.packages("tidyverse")
5 install.packages("VGAM")
6 install.packages("pracma")
7 install.packages("egg")
8 install.packages("ggpubr")
9 install.packages("lme4")
10 install.packages("broom")
11 install.packages("multcomp")
12 install.packages("multcompView")
13 install.packages("emmeans")
14
15 library(dplyr)
16 library(tidyr)
17 library(ggplot2)
18 library(tidyverse)
19 library(VGAM)
20 library(pracma)
21 library(egg)
22 library(ggpubr)
23 library(lme4)
24 library(broom)
25 library(multcomp)
26 library(multcompView)
27 library(emmeans)
28
29 # This is for loading the data and converting columns to the right format
30 loc_string <- "~/Documents/MacBook 2020/Geografi KU/Speciale/Data/Python/Speciale/Output_files/"
31 loc_name <- "Computed_bottles.csv"
32 file_location <- paste(loc_string, loc_name, sep="")
33 flux_data <- read.csv(file_location, header=TRUE)
34 flux_data <- na.omit(flux_data)
35 flux_data$Date <- as.Date(flux_data$Date, format = "%Y-%m-%d")
36 keeps <- c('Date', 'Bottle', 'X24h_2', 'mg_N2O_N_m2_d', 'cumulative_m2')
37 flux_data <- flux_data[keeps]
38 flux_data <- subset(flux_data, Date <= '2023-01-09')
39
40 # Here I add the proper labels, such that legends will have proper layout
41 flux_data['Treatment'] <- "
42 flux_data <- within(flux_data, Treatment[Bottle == 1 | Bottle == 2 | Bottle == 3] <- 'HW')
43 flux_data <- within(flux_data, Treatment[Bottle == 4 | Bottle == 5 | Bottle == 6] <- 'HD')
44 flux_data <- within(flux_data, Treatment[Bottle == 7 | Bottle == 8 | Bottle == 9] <- 'HA')
45 flux_data <- within(flux_data, Treatment[Bottle == 10 | Bottle == 11 | Bottle == 12] <- 'LW')
46 flux_data <- within(flux_data, Treatment[Bottle == 13 | Bottle == 14 | Bottle == 15] <- 'LD')
47 flux_data <- within(flux_data, Treatment[Bottle == 16 | Bottle == 17 | Bottle == 18] <- 'AA')
48 flux_data <- within(flux_data, Treatment[Bottle == 19 | Bottle == 20 | Bottle == 21] <- 'AW')
49 flux_data <- within(flux_data, Treatment[Bottle == 22 | Bottle == 23 | Bottle == 24] <- 'AD')
50 flux_data <- within(flux_data, Treatment[Bottle == 25 | Bottle == 26 | Bottle == 27] <- 'LA')
51 flux_data <- flux_data[!(flux_data$Bottle == 28), ]
52 flux_data <- flux_data %>%
53   group_by(Bottle) %>%
```

```

54 mutate(days = cumsum(X24h_2))
55 flux_data['Day_fac'] <- factor(flux_data$days)
56
57 log_flux <- flux_data[c('Date', 'Bottle', 'Treatment', 'Day_fac', 'mg_N2O_N_m2_d', 'cumulative_m2')]
58 log_flux['Input_mg_m2'] <- "
59 log_flux <- within(log_flux, Input_mg_m2[Treatment == 'HW'|Treatment == 'HA'|Treatment == 'HD'] <-
15000)
60 log_flux <- within(log_flux, Input_mg_m2[Treatment == 'AW'|Treatment == 'AA'|Treatment == 'AD'] <-
5000)
61 log_flux <- within(log_flux, Input_mg_m2[Treatment == 'LW'|Treatment == 'LA'|Treatment == 'LD'] <-
1000)
62 log_flux$Input_mg_m2 <- as.integer(log_flux$Input_mg_m2)
63 log_flux['Emission_factor'] <- log_flux['cumulative_m2']/log_flux['Input_mg_m2']*100
64
65 test <- log_flux#[log_flux$Treatment == 'AD' | log_flux$Treatment == 'AW' | log_flux$Treatment
== 'AA',]
66 test <- test[test$Day_fac == 57,]
67 summm <- test %>%
68   group_by(Treatment) %>%
69   summarise(mean_ef = mean(Emission_factor),
70             sd_ef = sd(Emission_factor),
71             mean_loss = mean(cumulative_m2))
72 summm
73
74 test$cumulative_m2
75
76 # The functions below are very slow, only run if necessary
77 # The functions below are the linear mixed models, which is used for repeated measures
analysis.
78
79 # This line of code says that I should evaluate the difference in cumulated emissions between
treatments by each day, and use the bottles as the random parameter
80 mixed <- lmer(log(cumulative_m2 + 1) ~ Treatment*Day_fac + (1|Bottle), data = log_flux)
81 emm <- emmeans(mixed, pairwise ~ Treatment|Day_fac, adjust = 'tukey') # This takes in the
mixed model and performs the test, using tukey test
82 cld_pairs <- cld(object = emm, Letters = letters, adjust = 'tukey', alpha = 0.05) # This adds
significance groups, which makes significant differences much easier to read
83 sink("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/cumulated_signi_groups.txt")
84 print(cld_pairs)
85 sink()
86
87 emm2 <- emmeans(mixed, pairwise ~ Treatment|Day_fac, adust = 'tukey')
88 sink("~/Documents/MacBook 2020/Geografi KU/Speciale/Data/R/Week_results/cumulated.txt")
89 summary(emm2)
90 sink()
91
92 emission_model <- lmer(log(Emission_factor + 1) ~ Treatment*Day_fac + (1|Bottle), data = log_flux)
93 emm3 <- emmeans(emission_model, pairwise ~ Treatment|Day_fac, adjust = 'tukey')
94 cld_pairs <- cld(object = emm3, Letters = letters, adjust = 'tukey', alpha = 0.05)
95 sink("~/Documents/MacBook 2020/Geografi KU/Speciale/Data/R/Week_results/emission_groups.txt")
96 print(cld_pairs)
97 sink()
98
99 emm4 <- emmeans(emission_model, pairwise ~ Treatment|Day_fac, adust = 'tukey')
100 sink("~/Documents/MacBook 2020/Geografi KU/Speciale/Data/R/Week_results/emission.txt")

```

```
101 summary(emm4)
102 sink()
103
104 flux_model <- lmer(log(mg_N2O_N_m2_d + 1) ~ Treatment*Day_fac + (1|Bottle), data = log_flux)
105 emm5 <- emmeans(flux_model, pairwise ~ Treatment|Day_fac, adjust = 'tukey')
106 cld_pairs <- cld(object = emm5, Letters = letters, adjust = 'tukey', alpha = 0.05)
107 sink("~/Documents/MacBook 2020/Geografi KU/Speciale/Data/R/Week_results/flux_groups.txt")
108 print(cld_pairs)
109 sink()
110
111 emm6 <- emmeans(flux_model, pairwise ~ Treatment|Day_fac, adjust = 'tukey')
112 sink("~/Documents/MacBook 2020/Geografi KU/Speciale/Data/R/Week_results/flux.txt")
113 summary(emm6)
114 sink()
115
```

R_1_2_8_weeks.R

```
1 install.packages("dplyr")
2 install.packages("tidyr")
3 install.packages("ggplot2")
4 install.packages("readxl")
5 install.packages("tidypaleo")
6 install.packages("tidyverse")
7 install.packages("gasfluxes")
8 install.packages("VGAM")
9 install.packages("pracma")
10 install.packages("egg")
11 install.packages("ggpubr")
12 install.packages("reshape2")
13 install.packages("patchwork")
14 install.packages("cowplot")
15 install.packages("Hmisc")
16 install.packages("mmtable2")
17 install.packages("gridExtra")
18
19 library(gridExtra)
20 library(dplyr)
21 library(tidyr)
22 library(ggplot2)
23 library(readxl)
24 library(gasfluxes)
25 library(tidyverse)
26 library(tidypaleo)
27 library(pracma)
28 library(egg)
29 library(ggpubr)
30 library(reshape2)
31 library(data.table)
32 library(patchwork)
33 library(cowplot)
34 library(Hmisc)
35 library(gridExtra)
36 library(grid)
37 library(forcats)
38
39 # This is for loading the data and converting columns to the right format
40 loc_string <- "~/Documents/MacBook 2020/Geografi KU/Speciala/Data/Python/Speciala/Output_files/"
41 loc_name <- "Computed_bottles.csv"
42 file_location <- paste(loc_string, loc_name, sep="")
43 flux_data <- read.csv(file_location, header=TRUE)
44 flux_data <- na.omit(flux_data)
45 flux_data$Date <- as.Date(flux_data$Date, format = "%Y-%m-%d")
46
47 # Here I add the proper labels, such that legends will have proper layout
48 flux_data["Treatment"] <- "
49 flux_data <- within(flux_data, Treatment[Bottle == 1 | Bottle == 2 | Bottle == 3] <- 'HW')
50 flux_data <- within(flux_data, Treatment[Bottle == 4 | Bottle == 5 | Bottle == 6] <- 'HD')
51 flux_data <- within(flux_data, Treatment[Bottle == 7 | Bottle == 8 | Bottle == 9] <- 'HA')
52 flux_data <- within(flux_data, Treatment[Bottle == 10 | Bottle == 11 | Bottle == 12] <- 'LW')
53 flux_data <- within(flux_data, Treatment[Bottle == 13 | Bottle == 14 | Bottle == 15] <- 'LD')
```

```

54 flux_data <- within(flux_data, Treatment[Bottle == 16 | Bottle == 17 | Bottle == 18] <- 'LA')
55 flux_data <- within(flux_data, Treatment[Bottle == 19 | Bottle == 20 | Bottle == 21] <- 'AW')
56 flux_data <- within(flux_data, Treatment[Bottle == 22 | Bottle == 23 | Bottle == 24] <- 'AD')
57 flux_data <- within(flux_data, Treatment[Bottle == 25 | Bottle == 26 | Bottle == 27] <- 'AA')
58 flux_data <- flux_data[!(flux_data$Bottle == 28), ]
59 flux_data[Input_N_mg] <- "
60 flux_data <- within(flux_data, Input_N_mg[Treatment == 'HW' | Treatment == 'HD' | Treatment ==
'HA'] <- 15000)
61 flux_data <- within(flux_data, Input_N_mg[Treatment == 'LW' | Treatment == 'LD' | Treatment == 'LA']
<- 1000)
62 flux_data <- within(flux_data, Input_N_mg[Treatment == 'AW' | Treatment == 'AD' | Treatment ==
'AA'] <- 5000)
63 flux_data$Input_N_mg <- as.numeric(flux_data$Input_N_mg)
64
65 # This is for making the dataframe nicer to work with
66 flux_data[date_weight] <- flux_data[X24h]
67 flux_data[ug_N2O_N_m2_h] <- (flux_data[g_N2O_N_m2_d] * 1000*1000)/24
68 flux_data[cumulative_ug_N2O_N_m2] <- flux_data[cumulative_m2] * 1000
69 keeps <- c('Date', 'Bottle', 'Treatment', 'date_weight', 'ug_N2O_N_m2_h',
'cumulative_ug_N2O_N_m2', 'Input_N_mg')
70 flux_data <- flux_data[keeps]
71
72 # I make subsets which are good to work with, so I can create summary tables
73 one_week <- subset(flux_data, Date <= '2022-11-16')
74 two_weeks <- subset(flux_data, Date <= '2022-11-23')
75 eight_weeks <- subset(flux_data, Date <= '2023-01-04')
76 eight_weeks$date_weight[eight_weeks$Date == '2023-01-02'] <- 6
77
78 # Below I create summary tables for each period. This is used for the large table in the results
section,
79 # all lines until next comment is the same but for different parameters
80 one_week_results <- one_week %>%
81   group_by(Treatment) %>%
82   summarise(avg_flux = weighted.mean(ug_N2O_N_m2_h, date_weight),
83             weighted_std = sqrt(wtd.var(ug_N2O_N_m2_h, date_weight)),
84             count = n())
85 one_week_results[Duration] <- 'One week'
86
87 two_weeks_results <- two_weeks %>%
88   group_by(Treatment) %>%
89   summarise(avg_flux = weighted.mean(ug_N2O_N_m2_h, date_weight),
90             weighted_std = sqrt(wtd.var(ug_N2O_N_m2_h, date_weight)),
91             count = n())
92 two_weeks_results[Duration] <- 'Two weeks'
93
94 eight_weeks_results <- eight_weeks %>%
95   group_by(Treatment) %>%
96   summarise(avg_flux = weighted.mean(ug_N2O_N_m2_h, date_weight),
97             weighted_std = sqrt(wtd.var(ug_N2O_N_m2_h, date_weight)),
98             count = n())
99 eight_weeks_results[Duration] <- 'Eight weeks'
100
101 results <- rbind(one_week_results, two_weeks_results, eight_weeks_results)
102 write.csv(results, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_week_results.csv", row.names=FALSE)
103

```

```

104 one_week_max <- one_week %>%
105   group_by(Treatment) %>%
106   summarise(max_flux = max(ug_N2O_N_m2_h))
107 one_week_max['Duration'] <- 'One week'
108 two_weeks_max <- two_weeks %>%
109   group_by(Treatment) %>%
110   summarise(max_flux = max(ug_N2O_N_m2_h))
111 two_weeks_max['Duration'] <- 'Two weeks'
112 eight_weeks_max <- eight_weeks %>%
113   group_by(Treatment) %>%
114   summarise(max_flux = max(ug_N2O_N_m2_h))
115 eight_weeks_max['Duration'] <- 'Eight weeks'
116 max_results <- rbind(one_week_max, two_weeks_max, eight_weeks_max)
117 write.csv(max_results, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_week_results_max.csv", row.names=FALSE)
118
119 one_week_min <- one_week %>%
120   group_by(Treatment) %>%
121   summarise(min_flux = min(ug_N2O_N_m2_h))
122 one_week_min['Duration'] <- 'One week'
123 two_weeks_min <- two_weeks %>%
124   group_by(Treatment) %>%
125   summarise(min_flux = min(ug_N2O_N_m2_h))
126 two_weeks_min['Duration'] <- 'Two weeks'
127 eight_weeks_min <- eight_weeks %>%
128   group_by(Treatment) %>%
129   summarise(min_flux = min(ug_N2O_N_m2_h))
130 eight_weeks_min['Duration'] <- 'Eight weeks'
131 min_results <- rbind(one_week_min, two_weeks_min, eight_weeks_min)
132 write.csv(min_results, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_week_results_min.csv", row.names=FALSE)
133
134 eight_weeks <- eight_weeks[eight_weeks$Date == '2023-01-02',]
135 eight_weeks['cumulative_mg_N2O_N_m2'] <- eight_weeks['cumulative_ug_N2O_N_m2']/1000
136 eight_weeks['percent_loss'] <-
eight_weeks['cumulative_mg_N2O_N_m2']/eight_weeks['Input_N_mg'] * 100
137 two_weeks <- two_weeks[two_weeks$Date == '2022-11-23',]
138 two_weeks['cumulative_mg_N2O_N_m2'] <- two_weeks['cumulative_ug_N2O_N_m2']/1000
139 two_weeks['percent_loss'] <- two_weeks['cumulative_mg_N2O_N_m2']/two_weeks['Input_N_mg'] *
100
140 one_week <- one_week[one_week$Date == '2022-11-16',]
141 one_week['cumulative_mg_N2O_N_m2'] <- one_week['cumulative_ug_N2O_N_m2']/1000
142 one_week['percent_loss'] <- one_week['cumulative_mg_N2O_N_m2']/one_week['Input_N_mg'] * 100
143
144 eight_weeks_loss <- eight_weeks %>%
145   group_by(Treatment) %>%
146   summarise(avg_loss = mean(percent_loss),
147             std_loss = std(percent_loss))
148 eight_weeks_loss['Duration'] <- 'Eight weeks'
149
150 two_weeks_loss <- two_weeks %>%
151   group_by(Treatment) %>%
152   summarise(avg_loss = mean(percent_loss),
153             std_loss = std(percent_loss))
154 two_weeks_loss['Duration'] <- 'Two weeks'
155

```

```

156 one_week_loss <- one_week %>%
157   group_by(Treatment) %>%
158   summarise(avg_loss = mean(percent_loss),
159             std_loss = std(percent_loss))
160 one_week_loss[Duration] <- 'One week'
161
162 loss_results <- rbind(one_week_loss, two_weeks_loss, eight_weeks_loss)
163 write.csv(loss_results, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_week_loss.csv", row.names=FALSE)
164
165 eight_weeks_loss_mg <- eight_weeks %>%
166   group_by(Treatment) %>%
167   summarise(avg_loss = mean(cumulative_mg_N2O_N_m2),
168             std_loss = std(cumulative_mg_N2O_N_m2))
169 eight_weeks_loss_mg[Duration] <- 'Eight weeks'
170
171 two_weeks_loss_mg <- two_weeks %>%
172   group_by(Treatment) %>%
173   summarise(avg_loss = mean(cumulative_mg_N2O_N_m2),
174             std_loss = std(cumulative_mg_N2O_N_m2))
175 two_weeks_loss_mg[Duration] <- 'Two weeks'
176
177 one_week_loss_mg <- one_week %>%
178   group_by(Treatment) %>%
179   summarise(avg_loss = mean(cumulative_mg_N2O_N_m2),
180             std_loss = std(cumulative_mg_N2O_N_m2))
181 one_week_loss_mg[Duration] <- 'One week'
182 loss_mg <- rbind(one_week_loss_mg, two_weeks_loss_mg, eight_weeks_loss_mg)
183 write.csv(loss_mg, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_week_loss_mg.csv", row.names=FALSE)
184
185 total_df <- merge(results, max_results, by = c('Treatment', 'Duration'))
186 total_df <- merge(total_df, min_results, by = c('Treatment', 'Duration'))
187 total_df <- merge(total_df, loss_mg, by = c('Treatment', 'Duration'))
188 total_df <- merge(total_df, loss_results, by = c('Treatment', 'Duration'))
189
190 write.csv(total_df, "~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Week_results/1_2_8_all.csv", row.names=FALSE)
191
192 #plots, only emission factors, this is not used in the paper, but the values are reported in a
table
193
194 #This is to add significance groups to each treatment
195 one_week_signi <- data.frame(
196   Treatment = c('LD', 'LA', 'LW', 'AD', 'AA', 'AW', 'HD', 'HA', 'HW'),
197   signi = c('a', 'ab', 'b', 'a', 'a', 'ab', 'a', 'a', 'ab'))
198 two_weeks_signi <- data.frame(
199   Treatment = c('LD', 'LA', 'LW', 'AD', 'AA', 'AW', 'HD', 'HA', 'HW'),
200   signi = c('ab', 'bc', 'c', 'a', 'a', 'c', 'a', 'ab', 'c'))
201 eight_weeks_signi <- data.frame(
202   Treatment = c('LD', 'LA', 'LW', 'AD', 'AA', 'AW', 'HD', 'HA', 'HW'),
203   signi = c('ab', 'bc', 'cd', 'a', 'a', 'd', 'a', 'ab', 'd'))
204
205 # Here I merge with the summary table calculated above
206 one_week_signi <- merge(one_week_signi, one_week_loss, by = 'Treatment')
207 two_weeks_signi <- merge(two_weeks_signi, two_weeks_loss, by = 'Treatment')

```

```

208 eight_weeks_signi <- merge(eight_weeks_signi, eight_weeks_loss, by = 'Treatment')
209
210 # Add proper labels
211 one_week_signi['N_amount'] <- ""
212 one_week_signi <- within(one_week_signi, N_amount[Treatment == 'LD'|Treatment ==
'LA'|Treatment == 'LW'] <- 'Low N')
213 one_week_signi <- within(one_week_signi, N_amount[Treatment == 'AD'|Treatment ==
'AA'|Treatment == 'AW'] <- 'Normal N')
214 one_week_signi <- within(one_week_signi, N_amount[Treatment == 'HD'|Treatment ==
'HA'|Treatment == 'HW'] <- 'High N')
215 one_week_signi['Wetness'] <- ""
216 one_week_signi <- within(one_week_signi, Wetness[Treatment == 'LD'|Treatment == 'AD'|Treatment
== 'HD'] <- 'Drained')
217 one_week_signi <- within(one_week_signi, Wetness[Treatment == 'LA'|Treatment == 'AA'|Treatment
== 'HA'] <- 'Field conditions')
218 one_week_signi <- within(one_week_signi, Wetness[Treatment == 'LW'|Treatment ==
'AW'|Treatment == 'HW'] <- 'Saturated')
219 one_week_signi <- one_week_signi %>%
220   arrange(factor(N_amount, levels = c("Low N", "Normal N", "High N")),
221     Wetness) %>%
222   mutate(N_amount = fct_relevel(N_amount, "Low N", "Normal N", "High N"))
223
224 two_weeks_signi['N_amount'] <- ""
225 two_weeks_signi <- within(two_weeks_signi, N_amount[Treatment == 'LD'|Treatment ==
'LA'|Treatment == 'LW'] <- 'Low N')
226 two_weeks_signi <- within(two_weeks_signi, N_amount[Treatment == 'AD'|Treatment ==
'AA'|Treatment == 'AW'] <- 'Normal N')
227 two_weeks_signi <- within(two_weeks_signi, N_amount[Treatment == 'HD'|Treatment ==
'HA'|Treatment == 'HW'] <- 'High N')
228 two_weeks_signi['Wetness'] <- ""
229 two_weeks_signi <- within(two_weeks_signi, Wetness[Treatment == 'LD'|Treatment ==
'AD'|Treatment == 'HD'] <- 'Drained')
230 two_weeks_signi <- within(two_weeks_signi, Wetness[Treatment == 'LA'|Treatment ==
'AA'|Treatment == 'HA'] <- 'Field conditions')
231 two_weeks_signi <- within(two_weeks_signi, Wetness[Treatment == 'LW'|Treatment ==
'AW'|Treatment == 'HW'] <- 'Saturated')
232 two_weeks_signi <- two_weeks_signi %>%
233   arrange(factor(N_amount, levels = c("Low N", "Normal N", "High N")),
234     Wetness) %>%
235   mutate(N_amount = fct_relevel(N_amount, "Low N", "Normal N", "High N"))
236
237 eight_weeks_signi['N_amount'] <- ""
238 eight_weeks_signi <- within(eight_weeks_signi, N_amount[Treatment == 'LD'|Treatment ==
'LA'|Treatment == 'LW'] <- 'Low N')
239 eight_weeks_signi <- within(eight_weeks_signi, N_amount[Treatment == 'AD'|Treatment ==
'AA'|Treatment == 'AW'] <- 'Normal N')
240 eight_weeks_signi <- within(eight_weeks_signi, N_amount[Treatment == 'HD'|Treatment ==
'HA'|Treatment == 'HW'] <- 'High N')
241 eight_weeks_signi['Wetness'] <- ""
242 eight_weeks_signi <- within(eight_weeks_signi, Wetness[Treatment == 'LD'|Treatment ==
'AD'|Treatment == 'HD'] <- 'Drained')
243 eight_weeks_signi <- within(eight_weeks_signi, Wetness[Treatment == 'LA'|Treatment ==
'AA'|Treatment == 'HA'] <- 'Field conditions')
244 eight_weeks_signi <- within(eight_weeks_signi, Wetness[Treatment == 'LW'|Treatment ==
'AW'|Treatment == 'HW'] <- 'Saturated')
245 eight_weeks_signi <- eight_weeks_signi %>%

```

```

246 arrange(factor(N_amount, levels = c("Low N", "Normal N", "High N")),
247           Wetness) %>%
248 mutate(N_amount = fct_relevel(N_amount, "Low N", "Normal N", "High N"))
249
250 # Plots
251 ggplot(one_week_signi, aes(x = N_amount, y = avg_loss)) +
252   geom_point() +
253   geom_errorbar(aes(ymin = avg_loss - std_loss, ymax = avg_loss + std_loss), width = 0.1) +
254   geom_text(aes(label = signi, y = avg_loss + std_loss), vjust = -0.5) +
255   facet_grid(. ~ Wetness, scales = "free_y") +
256   theme_article() +
257   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
258         strip.background = element_rect(colour = "white", fill = "white"),
259         axis.text.x = element_text(margin = margin(t = 10)),
260         panel.border = element_rect(colour = "NA"),
261         axis.line = element_line(colour = "black"),
262         axis.title = element_text(size = 10),
263         strip.text = element_text(size = 10),
264         plot.background = element_rect(fill = 'white', colour = 'white')) +
265   labs(x = "", y = "Emission factor (%)") +
266   geom_text(data = subset(one_week_signi, Wetness == "Drained"), x = -Inf, y = Inf, label = "a"),
267   size = 5, hjust = -1, vjust = 2) +
268   ylim(0, 9)
269 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/emission_1.png", width = 15, height = 10, units = "cm", dpi = 300)
270
271 ggplot(two_weeks_signi, aes(x = N_amount, y = avg_loss)) +
272   geom_point() +
273   geom_errorbar(aes(ymin = avg_loss - std_loss, ymax = avg_loss + std_loss), width = 0.1) +
274   geom_text(aes(label = signi, y = avg_loss + std_loss), vjust = -0.5) +
275   facet_grid(. ~ Wetness, scales = "free_y") +
276   theme_article() +
277   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
278         strip.background = element_rect(colour = "white", fill = "white"),
279         axis.text.x = element_text(margin = margin(t = 10)),
280         panel.border = element_rect(colour = "NA"),
281         axis.line = element_line(colour = "black"),
282         axis.title = element_text(size = 10),
283         strip.text = element_text(size = 10),
284         plot.background = element_rect(fill = 'white', colour = 'white')) +
285   labs(x = "", y = "Emission factor (%)") +
286   geom_text(data = subset(two_weeks_signi, Wetness == "Drained"), x = -Inf, y = Inf, label = "b"),
287   size = 5, hjust = -1, vjust = 2) +
288   ylim(0, 15)
289 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/emission_2.png", width = 15, height = 10, units = "cm", dpi = 300)
290
291 ggplot(eight_weeks_signi, aes(x = N_amount, y = avg_loss)) +
292   geom_point() +
293   geom_errorbar(aes(ymin = avg_loss - std_loss, ymax = avg_loss + std_loss), width = 0.1) +
294   geom_text(aes(label = signi, y = avg_loss + std_loss), vjust = -0.5) +
295   facet_grid(. ~ Wetness, scales = "free_y") +
296   theme_article() +
297   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
298         strip.background = element_rect(colour = NA, fill = NA),
299         axis.text.x = element_text(margin = margin(t = 10)),

```

```
298   panel.border = element_rect(colour = "NA"),
299   axis.line = element_line(colour = "black"),
300   axis.title = element_text(size = 10),
301   strip.text = element_text(size = 10),
302   plot.background = element_rect(fill = 'white', colour = 'white')) +
303   labs(x = "", y = "Emission factor (%)") +
304   geom_text(data = subset(eight_weeks_signi, Wetness == "Drained"), x = -Inf , y = Inf, label = "c"),
size = 5, hjust = -1, vjust =2)+
305   ylim(0, 18)
306   ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/emission_8.png", width = 15, height = 10, units = "cm", dpi = 300)
307
```

Climate.R

```
1  install.packages("dplyr")
2  install.packages("tidyr")
3  install.packages("ggplot2")
4  install.packages("readxl")
5  install.packages("tidypaleo")
6  install.packages("tidyverse")
7  install.packages("gasfluxes")
8  install.packages("VGAM")
9  install.packages("pracma")
10 install.packages("egg")
11 install.packages("ggpubr")
12
13 library(gridExtra)
14 library(dplyr)
15 library(tidyr)
16 library(ggplot2)
17 library(readxl)
18 library(gasfluxes)
19 library(tidyverse)
20 library(tidypaleo)
21 library(pracma)
22 library(egg)
23 library(ggpubr)
24 library(reshape2)
25 library(data.table)
26 library(patchwork)
27 library(cowplot)
28 library(Hmisc)
29 library(gridExtra)
30 library(grid)
31 library(forcats)
32 library(splines)
33 my_theme <- theme_article() +
34   theme(text = element_text(family = "EB Garamond"))
35
36 # This is for loading the data and converting columns to the right format
37 loc_string <- "~/Documents/MacBook 2020/Geografi KU/Speciale/Data/Excel/"
38 loc_name <- "climate_soro.csv"
39 file_location <- paste(loc_string, loc_name, sep="")
40 climate_data <- read.csv(file_location, header=TRUE)
41 climate_data$label <- factor(climate_data$label, levels = unique(climate_data$label), ordered =
TRUE)
42
43 # Here I add the proper labels, such that legends will have proper layout
44 climate_data[Season] <- ""
45 climate_data <- within(climate_data, Season[label == 'Jan'|label == 'Feb'|label == 'Dec'] <- 'Winter')
46 climate_data <- within(climate_data, Season[label == 'Mar'|label == 'Apr'|label == 'May'] <- 'Spring')
47 climate_data <- within(climate_data, Season[label == 'Jun'|label == 'Jul'|label == 'Aug'] <- 'Summer')
```

```

48 climate_data <- within(climate_data, Season[label == 'Sep'|label == 'Oct'|label == 'Nov'] <- 'Autumn')
49
50 # I make a summary of the most important data, this is used for plotting
51 summary_climate <- climate_data %>%
52   group_by(Season) %>%
53   summarise(Mean_temp = mean(avg_t),
54             Std_temp = std(avg_t),
55             Sum_p = sum(avg_p))
56
57 climate_data %>%
58   summarise(mean_t = mean(avg_t),
59             std_t = std(avg_t),
60             sum_p = sum(avg_p))
61
62 climate_data[order(climate_data$avg_t),]
63
64 # Below is the hydrotherm for the area
65 f = 4
66 ggplot(aes(x = label, y = avg_p), data = climate_data) + my_theme +
67   geom_bar(stat = 'identity') +
68   geom_line(aes(x = label, y = avg_t * f, group = 1)) +
69   scale_y_continuous(name = "Monthly precipitation (mm)",
70                     sec.axis = sec_axis(~ . / f, name = "Daily temperature (°C)"))+
71   theme(plot.background = element_rect(fill = 'white', colour = 'white'),
72         axis.text=element_text(size=9, color="black"), axis.title=element_text(size=9, color="black"),
73         axis.text.x = element_text(angle = 45, hjust=1), axis.title.x=element_blank())
74
75 ggsave("~/Documents/MacBook 2020/Geografi
KU/Speciale/Data/R/Figures_for_thesis/hydrotherm.png", width = 9, height = 7, units = "cm", dpi = 300)
76
77 theme(plot.background = element_rect(fill = 'white', colour = 'white'), legend.position="right",
78       axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank(),
legend.title=element_blank(),
79       axis.text=element_text(size=8, color="black"), axis.title=element_text(size=8, color="black"))
80

```