processes

MDPI

Supporting Information

Comparison study on the influence of water to biomass ratio in hydrothermal carbonization of fresh seaweed

S.1 Analysis of Variance

To compare the effects of biomass type and pyrolysis temperature on the elemental composition of the hydrochar, a one-way analysis of variance (ANOVA) was performed. The results showed that both biomass type and pyrolysis temperature had a significant effect on the elemental composition of the hydrochar.

ANOVA was used to determine the statistical significance of differences between groups. The ANOVA model used was:

$$Y_{ij} = \mu + \tau_i + \varepsilon_{ij} \quad (1)$$

μ is overall average (mean); τ_j is the differential effect (response) associated with the j level of X; ε_{ij} = noise or error associated with the particular ij data value.

The F-statistic was calculated as the ratio of the variance between groups to the variance within groups:

$$F = \frac{MS_{within}}{MS_{between}} \quad (2)$$

where $MS_{between}$ is the mean square between groups and MS_{within} is the mean square within groups.

The p-value was then calculated (the p-value is calculated based on the F-statistic and the degrees of freedom associated with the F-distribution) to determine the probability of obtaining a test statistic as extreme as the observed F-statistic, assuming the null hypothesis is true. A p-value less than 0.05 was considered statistically significant.

In this study, ANOVA was used to compare the means of multiple groups.

Table S1 F-statistic and p-value

	N		C		H		S		O	
	F-statistic	p-value	F-statistic	p-value	F-statistic	p-value	F-statistic	p-value	F-statistic	p-value
UFS-0	152.96	0.08	269.73	0.03	42.98	0.13	0.43	0.37	272.00	0.02
UFS-3	99.07	0.11	123.60	0.05	15.58	0.34	38.41	0.25	9.15	0.20
FS-0	193.92	0.07	26.21	0.32	1.28	0.74	1.30	0.02	5.68	0.59
FS-3	22.53	0.35	29.19	0.22	2.42	0.71	35.52	0.07	19.85	0.28
Dry S	24.81	0.06	75453	0.03	2.29	0.72	25.11	0.11	988.81	0.03

Based on the provided results, it appears that the elemental composition of hydrochar varies depending on the type of biomass and pyrolysis conditions used. The table shows the F-statistic and p-value for each element analyzed (C, H, S, O) for each sample tested (UFS-0, UFS-3, FS-0, FS-3, and Dry S).

For example, for the N element, the F-statistic for UFS-0 is 152.96, which indicates a significant difference in the amount of carbon in the hydrochar produced from this biomass at the specified pyrolysis conditions. The p-value for this F-statistic is 0.08, which is above the typical threshold of 0.05 for statistical significance. On the other hand, the F-statistic for UFS-3 is 99.07, with a corresponding p-value of 0.11, indicating a lower degree of statistical significance for this sample.

Similar patterns can be seen for the other elements analyzed (H, S, and O) and for the other biomass and pyrolysis conditions tested. These results suggest that the type of biomass and pyrolysis conditions used can have a significant impact on the elemental composition of hydrochar, highlighting the importance of carefully selecting these factors in the production of hydrochar for different applications.

Python code:

```
import numpy as np
from scipy.stats import f_oneway
import pandas as pd
from statsmodels.stats.multicomp import pairwise_tukeyhsd

# Define the data as a dictionary of lists
data = {
    'N': {
        'UFS-0': [[4.2, 4.14], [4.02, 3.98], [4.53, 4.56]],
        'UFS-3': [[3.98, 3.99], [4.21, 4.19], [4.11, 4.08]],
        'FS-0': [[3.87, 3.93], [4.01, 4.02], [4.38, 4.37]],
        'FS-3': [[3.92, 3.88], [3.97, 3.89], [4.26, 4.17]],
        'Dry S': [[3.68, 3.56], [3.4, 3.5], [3.9, 3.97]]
    },
    'C': {
        'UFS-0': [[44.99, 44.21] , [47.67, 48.14] , [55.81, 56.69]],
        'UFS-3': [[43.03, 43.23] ,[51.34, 53.1] , [55.71, 54.85]],
        'FS-0': [[44.79, 45.98] ,[48.47,50.78] ,[53.08,54.02]],
        'FS-3': [[45.4, 43] , [50.53, 47.81] ,[55.34,56.06]],
        'Dry S': [[48.59,48.29] , [48.24, 47.64] , [58.58,58.96]]
    },

    'H': {
        'UFS-0': [[4.759,4.626],[4.72,4.732],[5.142,5.153]],
        'UFS-3': [[4.732,4.758],[4.926,5.057],[5.018,5.042]],
        'FS-0': [[4.755,4.825],[4.572,4.88],[4.913,4.944]],
        'FS-3': [[5.094, 4.846],[4.864,4.586],[5.054,5.037]],
        'Dry S': [[5.211,5.185],[5.319,5.181],[5.358,5.301]]
    },

    'S': {
        'UFS-0': [[0.374,0.414],[0.356,0.363],[0.408,0.306]],
        'UFS-3': [[0.277,0.299],[0.318,0.353],[0.185,0.192]],
        'FS-0': [[0.562,0.646],[0.668,0.677],[0.72,0.631]],
        'FS-3': [[0.446,0.502],[0.223,0.24],[0.322,0.279]],
        'Dry S': [[0.636,0.671],[1.077,1.255],[0.687,0.585]]
    },


    'O': {
        'UFS-0': [[34.117,35.05],[30.844,30.395],[22.68,21.861]],
```

```
        'UFS-3': [[23.471,23.213],[26.386,24.48],[26.579,27.438]],
        'FS-0': [[29.903,28.499],[27.01,24.373],[25.757,24.885]],
        'FS-3': [[32.4,35.032],[26.903,29.964],[23.514,22.944]],
        'Dry S': [[29.703,30.114],[30.954,31.414],[20.025,19.734]]
    },

}
# Define empty lists to store the results
f_statistics = {key: [] for key in data}
p_values = {key: [] for key in data}

# Iterate over each group of samples for each parameter
for param in data:
    for sample in data[param]:
        # Convert the list of lists to a numpy array
        sample_data = np.array(data[param][sample])
        # Calculate the mean and standard deviation for each parameter
        sample_means = np.mean(sample_data, axis=1)
        overall_mean = np.mean(sample_data)
        # Calculate the sum of squares between groups and within groups
        ss_between = np.sum((sample_means - overall_mean)**2) * sample_data.shape[1]
        ss_within = np.sum((sample_data - sample_means.reshape(-1, 1))**2)
        # Calculate the degrees of freedom for the ANOVA test
        df_between = sample_data.shape[0] - 1
        df_within = sample_data.shape[0] * (sample_data.shape[1] - 1)
        # Calculate the mean square values for the ANOVA test
        ms_between = ss_between / df_between
        ms_within = ss_within / df_within
        # Calculate the F-statistic and p-value for the ANOVA test
        f_statistic = ms_between / ms_within
        p_value = 1 - f_oneway(*sample_data.T)[1]
        # Add the F-statistic and p-value to the lists
        f_statistics[param].append(f_statistic)
        p_values[param].append(p_value)


# Print the results

# Print the results
for param in data:
    print('Parameter:', param)
    for i, sample in enumerate(data[param]):
```

```
    print('Sample:', sample)

print('F-statistic:', f_statistics[param])
print('P-value:', p_values[param])
```

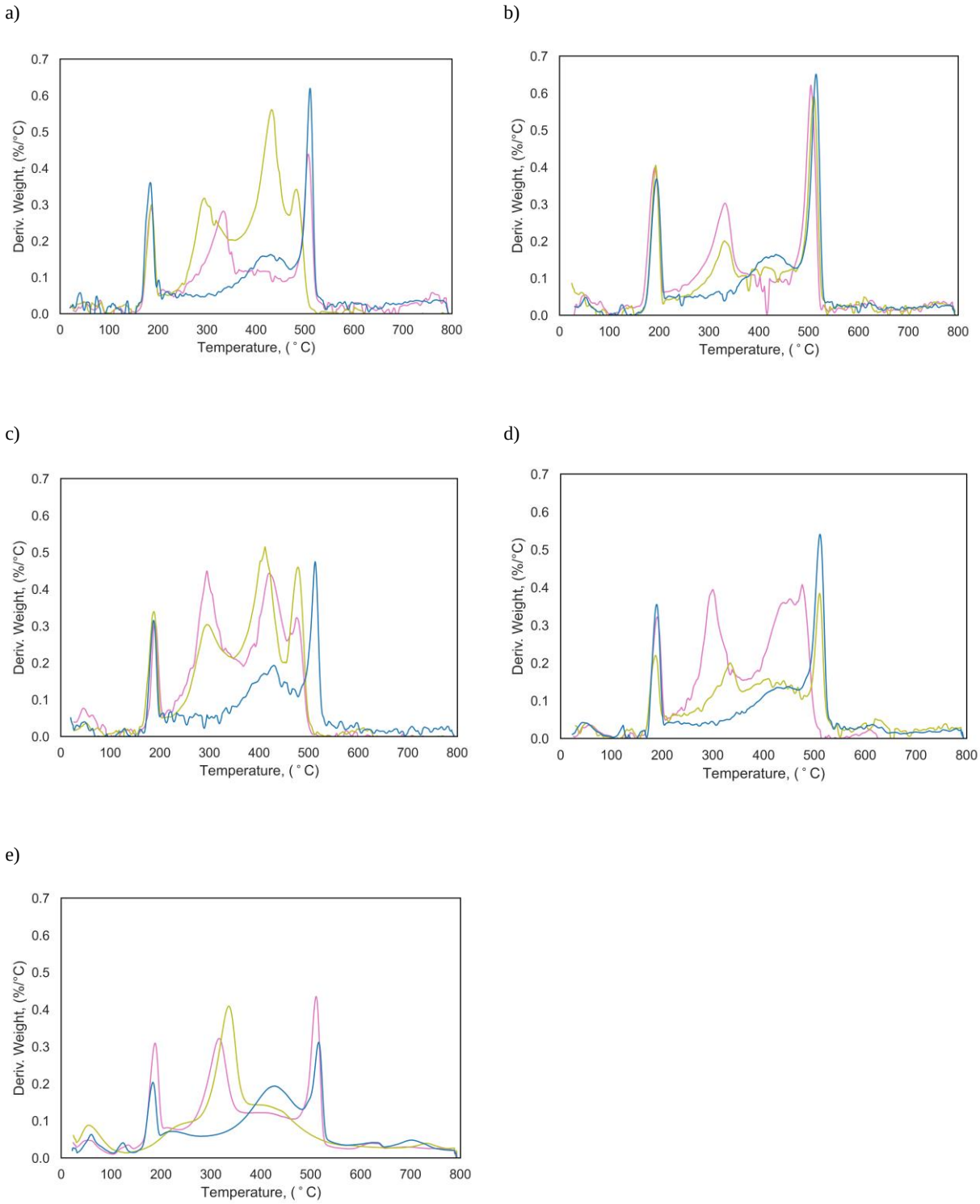

Figure S1: TGA curves of hydrochar, (a) FS-0; (b) FS-3; (c) UFS-0; (d) UFS-3; (e) dry seaweed. W: B=3; at temperature of (—) 180℃ (—) 210℃ (—) 250℃, Conditions: reaction temperature 210℃ and reaction time 4 h.

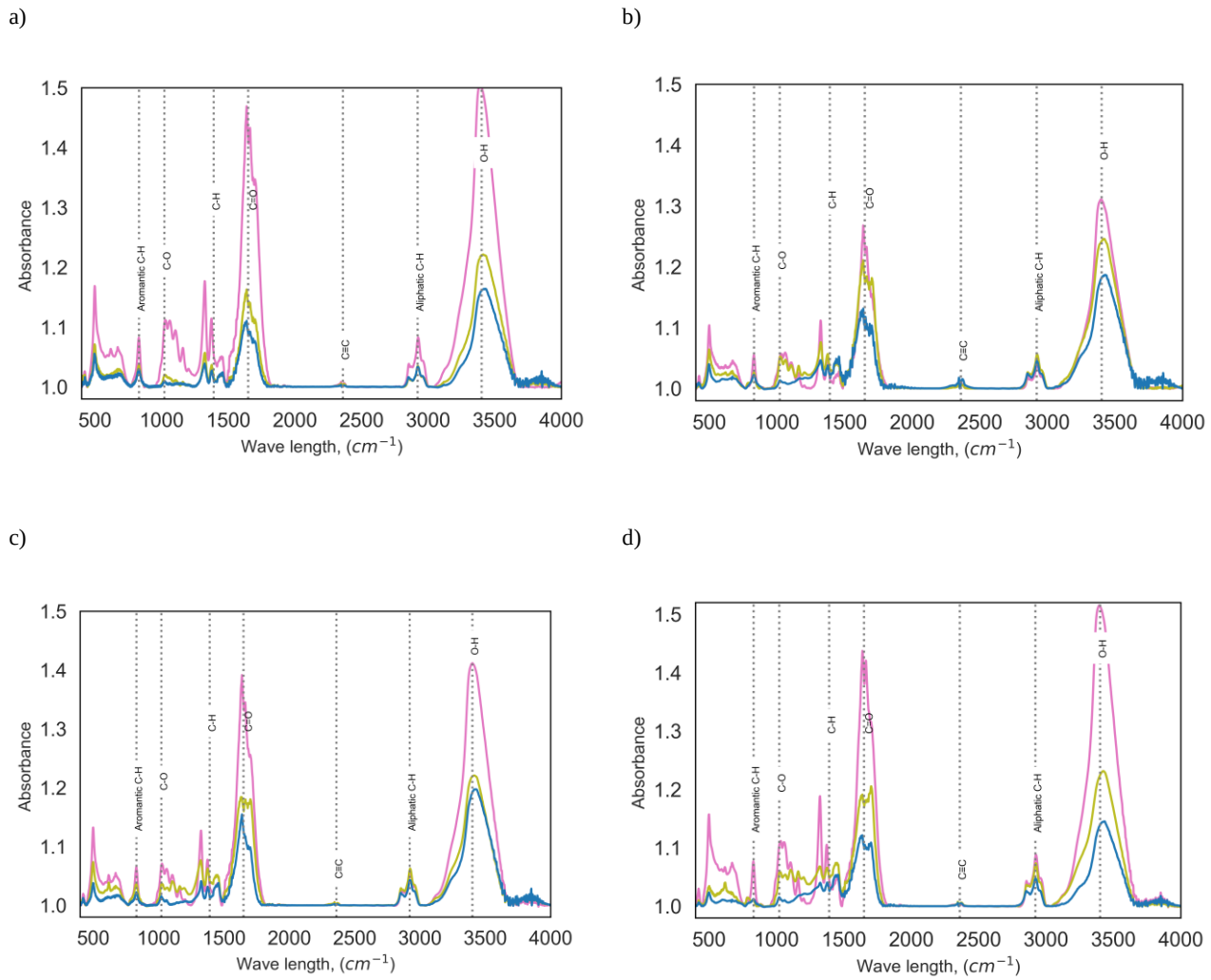

Figure S2: FT-IR analysis, (a) FS-0; (b) FS-3; (c) UFS-0; (d) UFS-3 at the temperature of (—) 180℃ (—) 210℃ (—) 250℃.

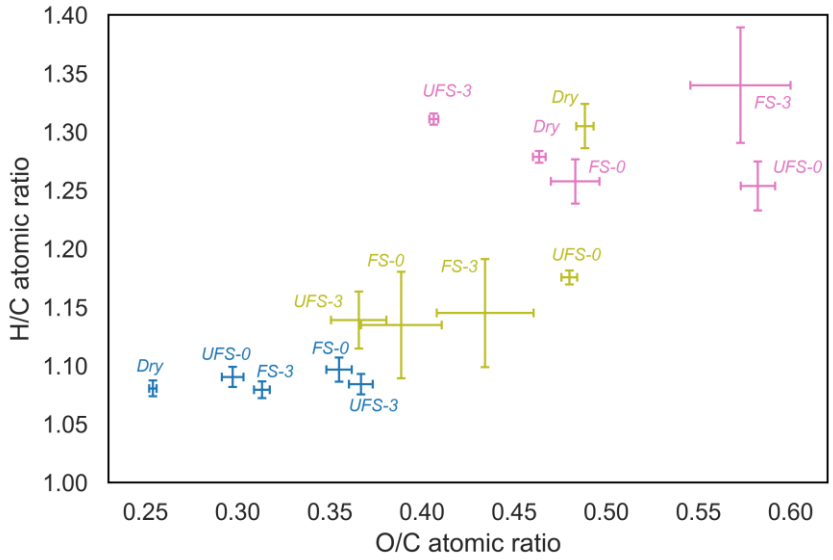

Figure S3: Van Krevelen diagram of hydrochar, at temperature of (—) 180°C (—) 210°C (—) 250°C.

S.1 Residual sum of squares

Within this context, the RSS (residual sum of squares) is computed using Eq (1) to measures the difference between the predicted values and the experimental dataset for the refined parameter set. A smaller value for the RSS indicates a better fit of the model to the experimental data.

$$RSS = \sum_{i=1}^{n} (q_i - f(x_i))^2 \quad (3)$$

The predicted value of qi using the adsorption models is represented by $f(x_i)$, whereas q_i refers to the experimentally measured MB concentration. The least squares method in the Python Scipy library was utilized to fit the adsorption models and RSS.

Python Code:

```
# -*- coding: utf-8 -*-
"""
Created on Mon Dec 19 16:10:00 2022

@author: Sepideh
"""

import csv
import scipy
import matplotlib.pyplot as plt
import numpy as np
from scipy.interpolate import interp1d
from scipy.optimize import curve_fit
import statistics
from numpy import log as ln
from sklearn.metrics import r2_score

#from scipy import interpolate1
plt.figure(dpi=500)

def Langmuir(C_liq, K, C_tot, q_m):
    return    q_m * K*C_liq / (1+K*C_liq)

V = 0.01 ## Volume of Methylene Blue (liter)-from sample No. 81 is 10 ml.
f_main_calib = 'C:/Users/GUGC/OneDrive - UGent/work/Data and Results/UV-Vis Calibration.csv'
f_main_UVVIS = 'C:/Users/GUGC/OneDrive - UGent/work/Data and Results/Data Experiment-UV-Vis-FCH data-MB-PH=7.csv'


f_name_calib = 'UV-Vis Calibration.csv'
```

```
M= np.genfromtxt(f_name_calib, delimiter=',')

#*********************************************************************
Sample_list = [117, 101, 124] #Those sample number I want to plot
#*********************************************************************

my_colors = ['blue', 'orange', 'fuchsia', 'brown', 'yellowgreen'] # gray: #b6b6b6

K_list = []
q_m_list = []
C_tot_list = []
K_SDV_list = []
C_tot_SDV_list = []
C_liq_mean = []
C_ADS_mean = []
q_m_SDV_list = []

for Sample_no, color in zip(Sample_list, my_colors):

    M2 = np.genfromtxt(f_main_UVVIS, delimiter=',', skip_header=1, usecols=range(0,5))
    M2_reduced = M2[M2[:,0]==Sample_no,:]

    Intp_f = interp1d(M[:,1],M[:,0])

    ABS_resampled = np.linspace(np.min(M[:,1]),np.max(M[:,1]),num=51)
    Con_predicted = Intp_f(ABS_resampled)

    C_liq_x1 = Intp_f(M2_reduced[:,3]) #experiment 1
    C_liq_x2 = Intp_f(M2_reduced[:,4]) #experiment 2
    C_liq = np.append(C_liq_x1, C_liq_x2)

    C_liq_mean = np.mean ([C_liq_x1 , C_liq_x2], axis=0)
    C_liq_std = np.std ([C_liq_x1 , C_liq_x2], axis=0)

    C_ADS_x1 = (M2_reduced[:,1] - C_liq_x1) * V / M2_reduced[:,2]
    C_ADS_x2 = (M2_reduced[:,1] - C_liq_x2) * V / M2_reduced[:,2]
    C_ADS = np.append(C_ADS_x1, C_ADS_x2)

    C_ADS_mean = np.mean ([C_ADS_x1 , C_ADS_x2], axis=0)
    C_ADS_std = np.std ([C_ADS_x1 , C_ADS_x2], axis=0)

    popt, pcov = curve_fit(Langmuir, C_liq, C_ADS, method='dogbox')
```

```
K = popt[0]
C_tot = popt[1]
q_m = popt[2]

perr = np.sqrt(np.diag(pcov))
K_SDV = perr[0]
C_tot_SDV = perr[1]
q_m_SDV = perr[2]

K_list = np.append(K_list, K)
C_tot_list = np.append(C_tot_list, C_tot)
K_SDV_list = np.append(K_SDV_list, K_SDV)
C_tot_SDV_list = np.append(C_tot_SDV_list, C_tot_SDV)
q_m_list = np.append(q_m_list, q_m)
q_m_SDV_list = np.append(q_m_SDV_list, q_m_SDV)

C_liq2 = np.linspace(min(C_liq), max(C_liq), num=51)
C_ADS2 = Langmuir(C_liq2, K, C_tot, q_m)
C_ADS2_U = Langmuir(C_liq2, K + K_SDV, C_tot+C_tot_SDV, q_m+q_m_SDV)

RSS = 0
for i in range(len(C_liq)):
    RSS += (C_ADS[i] - Langmuir(C_liq[i], K, C_tot, q_m))**2

print('RSS', RSS)

#Temperature finding
f_main = 'C:/Users/GUGC/OneDrive - UGent/work/Data and Results/Data Experiment Main.csv'

with open(f_main, newline='') as csvfile:
        data_main = list(csv.reader(csvfile))

temp = [i for i,x in enumerate(data_main[0]) if x==str(Sample_no)][0]

Temp_reduced_i = int(data_main[9][temp])

plt.errorbar(C_liq_mean, C_ADS_mean, xerr=C_liq_std, yerr=C_ADS_std,
            ls='', lw=0.6, marker='+', markersize=4, elinewidth =1,
            capsize=4, label= str(Temp_reduced_i) + '    (min)', color=color)
plt.plot(C_liq2,C_ADS2, color=color)

# plt.plot(Temp_reduced_i, SSA, 'o')
```

```
plt.plot(C_liq2,C_ADS2, color=color)
#plt.legend(loc='lower right', fontsize=10, facecolor='white')
plt.xlabel('$\mathregular{C_{liq}}$, (mg$\mathregular {L^{-1}}$)')
plt.ylabel('$\mathregular{q_e}$, (mg $\mathregular {g^{-1}}$)')
```