In [ ]:
```python
#python3
# Hans D. Escobar H. (hdescobarh@unal.edu.co)

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import spearmanr # type: ignore
from statsmodels.stats.multitest import multipletests
from itertools import combinations
```

Asociación estadística entre la altura, el diametro y el número de hojas de una aracéa

1. Datos

Los datos corresponden a observaciones de una población de plantas de la familia *araceae*, en la Reserva Nacional Forestal Bosque de Yotoco (fecha, 26-May-2023). Datos colectados por estudiantes del curso de Fundamentos de Ecología de Poblaciones 2023-S1, Departamento de Biología. Universidad Nacional de Colombia.

In [ ]:
```python
araceae_raw_df = pd.read_csv("./aracea_altura.csv", header= 1).astype({"grupo": str})
araceae_raw_df.drop("grupo", axis=1, inplace= True)
araceae_raw_df
```

Out[ ]:

	altura_m	diametro_cm	numero_hojas
0	0.70	1.27	12.0
1	0.74	1.62	16.0
2	1.01	1.24	10.0
3	1.00	1.34	9.0
4	0.83	1.46	10.0
5	0.76	1.27	12.0
6	1.36	1.43	17.0
7	0.87	1.21	4.0
8	1.25	1.27	20.0
9	1.01	1.59	15.0
10	1.20	1.00	18.0
11	1.16	1.10	8.0
12	1.31	1.20	16.0
13	1.04	1.10	12.0
14	0.85	1.10	15.0
15	0.84	1.00	15.0
16	1.33	2.30	13.0
17	0.80	1.00	12.0
18	0.81	0.90	10.0
19	1.08	1.30	14.0
20	1.10	1.27	16.0
21	0.74	1.27	15.0

	altura_m	diametro_cm	numero_hojas
22	1.75	1.27	14.0
23	1.80	2.07	31.0
24	1.24	1.72	18.0
25	1.52	1.27	15.0
26	0.90	1.11	15.0
27	1.80	1.59	14.0
28	1.20	1.27	3.4
29	1.66	1.59	9.0
30	3.00	1.21	25.0
31	2.46	1.60	9.0
32	1.30	0.30	15.0
33	1.80	1.10	18.0
34	1.88	1.00	15.0
35	1.50	1.20	12.0
36	1.94	1.01	17.0
37	1.64	1.40	10.0
38	2.15	1.50	19.0
39	1.60	1.20	5.0
40	3.00	3.80	25.0
41	2.46	1.60	9.0
42	1.30	0.30	15.0
43	1.80	1.10	18.0

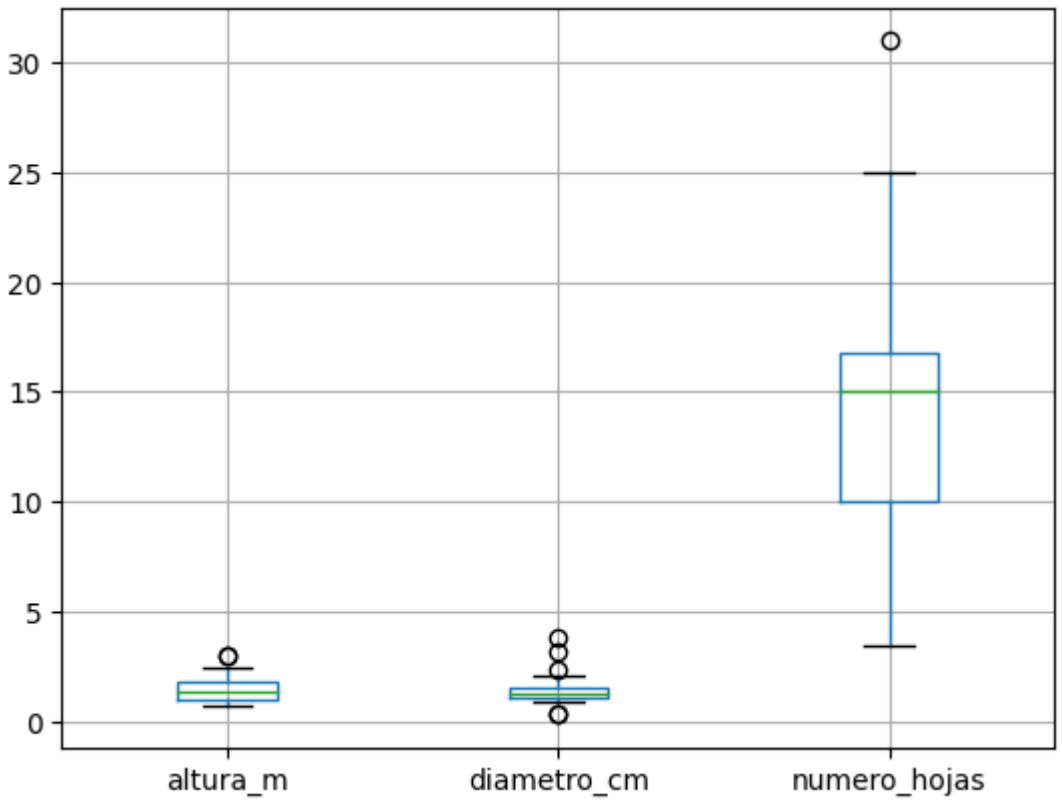
	altura_m	diametro_cm	numero_hojas
44	1.88	1.00	15.0
45	1.50	1.20	12.0
46	1.94	3.20	17.0
47	1.64	1.40	10.0
48	2.15	1.50	19.0
49	1.60	1.20	5.0

1.1. Limpieza de datos

En el boxplot se puede observar que, bajo el criterio del 1.5·IQR (intervalo interquartilico) hay algunos outliers.

In [ ]:
```
print("Dataset lenght:{}".format(len(araceae_raw_df)))
araceae_raw_df.boxplot()
plt.show()
```

```
Dataset lenght:50
```

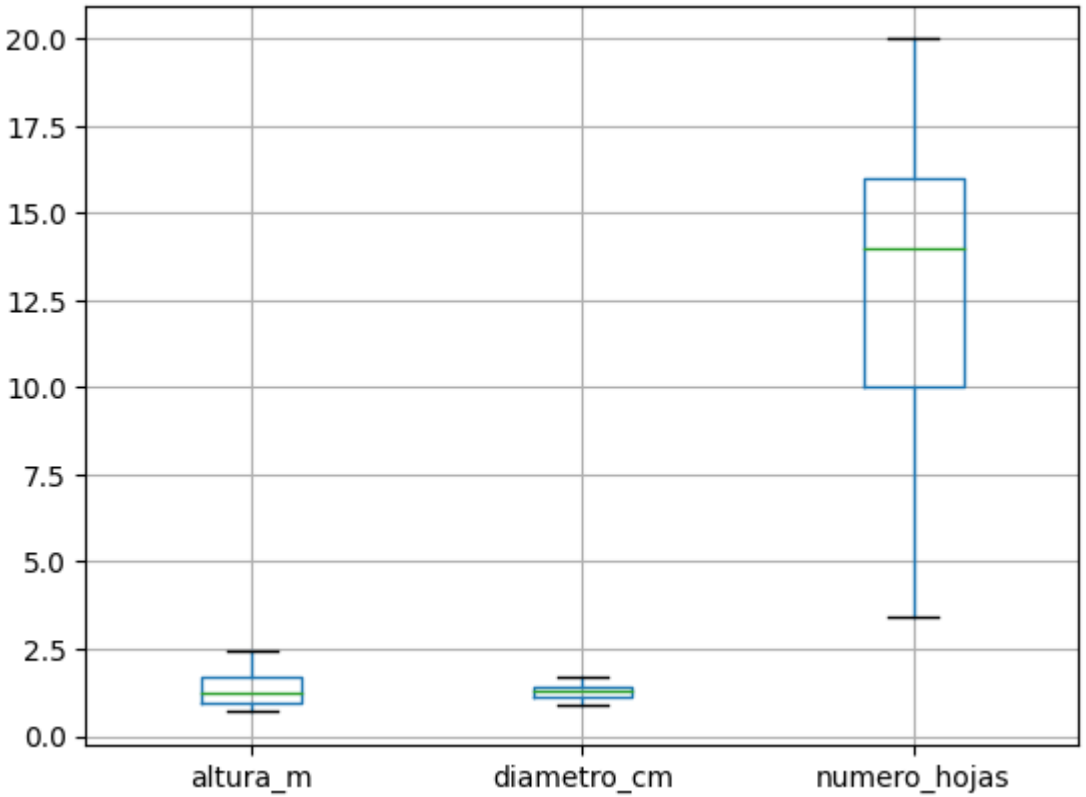


In [ ]:

```
# Calculate upper and lower limits of +-1.5IQR interval
descriptive_df = araceae_raw_df.describe()
limit_for_outlier: dict[str, tuple[float, float]] = {col_name: (0,0) for col_name in araceae_raw_df.columns}
for col_name in limit_for_outlier:
    iqr = descriptive_df[col_name]["75%"] - descriptive_df[col_name]["25%"]
    upper_limit = descriptive_df[col_name]["75%"] + 1.5 * iqr
    lower_limit = descriptive_df[col_name]["25%"] - 1.5 * iqr
    limit_for_outlier[col_name] = (lower_limit, upper_limit)
# make a query proposition from the limits dictionary
my_query = ""
tolerance = 1e-15
for col_name, (lower_limit, upper_limit) in limit_for_outlier.items():
    my_query += f"({col_name} - {lower_limit} >= -{tolerance} & {col_name} - {upper_limit} <= {tolerance})"
my_query = my_query.replace(")(", ") & (")
# Return cleaned dataset and shows boxplot
araceae_cleaned_df = araceae_raw_df.query(my_query)
```

```
print("Dataset lenght:{}".format(len(araceae_cleaned_df)))
araceae_cleaned_df.boxplot()
plt.show()
```

```
Dataset lenght:43
```



2. Análisis de correlación

In [ ]:
```
def spearman_correlation_analysis(dataframe: pd.DataFrame):
    # Realiza el análisis de correlación de Spearman, devuelve stadisticos y p-values
    rho, pvalue_uncorrected = spearmanr(dataframe, alternative="two-sided")

    # Extrae el triangulo superior sin la diagonal, y realiza corrección para multiples test
    correction_output = multipletests(pvalue_uncorrected[np.triu_indices(3, 1)], method="bonferroni")

    # Reconstruye una nueva matriz pero con los p-value corregidos
```

```
    pvalue_corrected = np.zeros((3,3))
    pvalue_corrected[np.triu_indices(3, 1)] = correction_output[1]
    pvalue_corrected[np.tril_indices(3, -1)] = correction_output[1]

    column_names = list(dataframe)
    rho_df = pd.DataFrame(rho, index = column_names, columns = column_names)
    pvalue_df = pd.DataFrame(pvalue_corrected, index = column_names, columns = column_names)

    return rho_df, pvalue_df
```

Ejemplo de como obtener el p-value mediante permutación. Recomendado cuando n es bajo.

```
from scipy.stats import permutation_test
from scipy.stats import spearmanr

r_spearman = lambda x: spearmanr(x,y).statistic
x = araceae_df['numero_hojas']
y = araceae_df['diametro_cm']
# Solo es necesario permutar una columna!
res_exact = permutation_test(
    (x, ), r_spearman, vectorized=False, permutation_type='pairings')
res_exact.statistic, res_exact.pvalue
...
PermutationTestResult(statistic=0.07548153631659311, pvalue=0.603, null_distribution=array([ 0.00678922,
0.38133573,  0.09674633, ..., -0.10993681,
        0.00434025,  0.15091458]))
```

In [ ]:

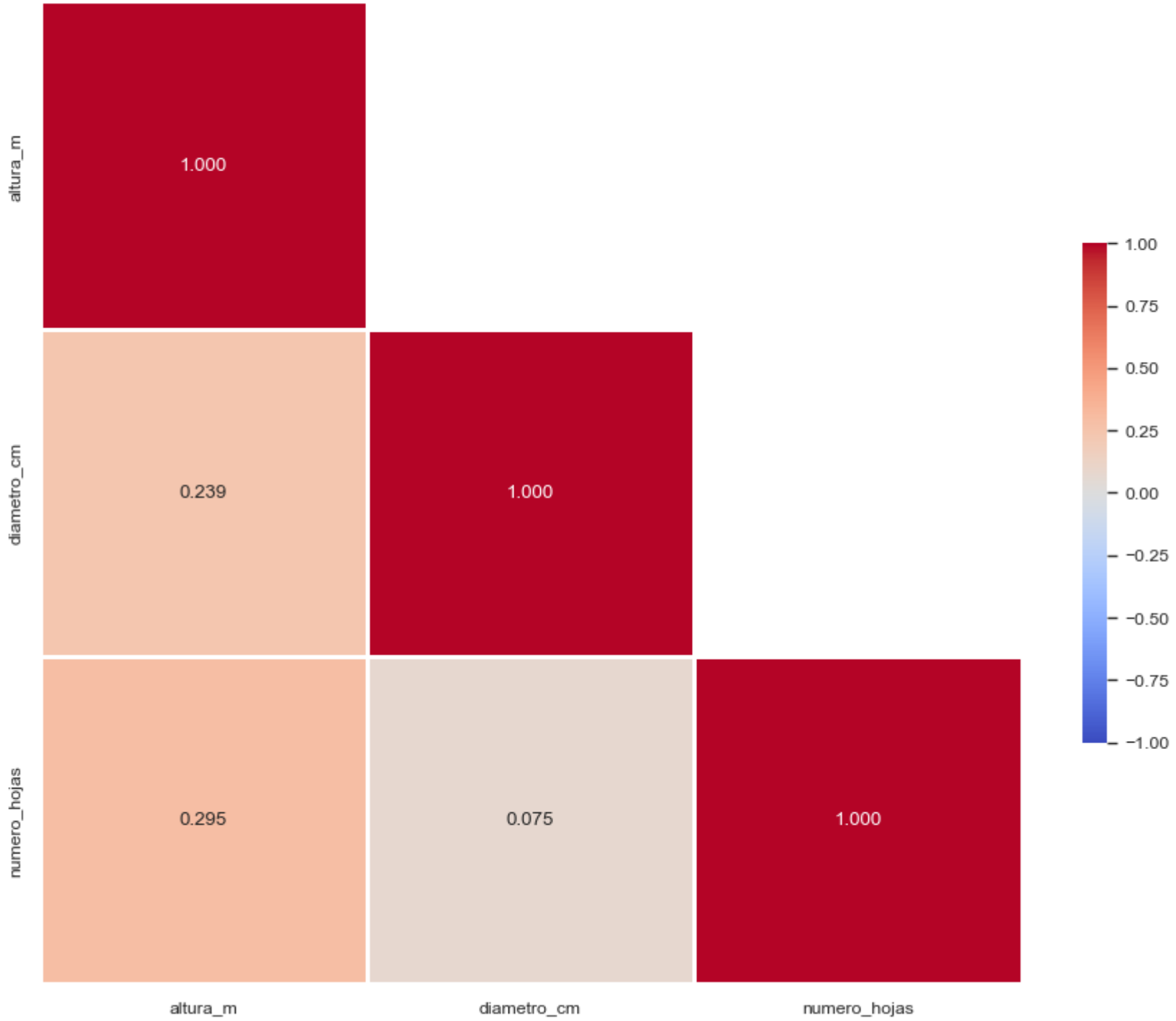
```
# Código para graficas
def make_heatmap(rho_df: pd.DataFrame):
    sns.set_theme(context= "paper", style="white", palette="bright")
    mask = np.triu(np.ones_like(rho_df, dtype=bool), 1)
    f, ax = plt.subplots(figsize=(11, 9))
    cmap = sns.color_palette("coolwarm", as_cmap=True) # sns.diverging_palette(h_neg= 15, h_pos=225, s=100, l=40, as_
    sns.heatmap(rho_df, mask=mask, annot= True, fmt=".3f",
                cmap=cmap, vmax=1, center=0, vmin=-1,
                square=True, linewidths=1, cbar_kws={"shrink": .5}, ax = ax)
    plt.show()

def make_scatterplot_histogram(araceae_df: pd.DataFrame, rho_df: pd.DataFrame, pvalue_df:pd.DataFrame):
    sns.set_theme(context= "paper", style="whitegrid", palette="muted")
```

```
    s = set_title = np.vectorize(lambda ax, rho, pval: ax.title.set_text(
        "ρ = {:.2f}\np-valor = {:.4f}".format(rho, pval)) if ax!=None else None)
    g = sns.PairGrid(araceae_df, diag_sharey=False, corner= True)
    palette = sns.color_palette("Set2")
    g.map_diag(plt.hist, color = palette[0]) # type: ignore
    g.map_lower(sns.scatterplot, color = palette[1], s=50) # type: ignore
    #g.map_upper(sns.kdeplot, lw=2)
    plt.subplots_adjust(hspace = 0.6)
    set_title(g.axes,rho_df, pvalue_df)
    plt.show()
```

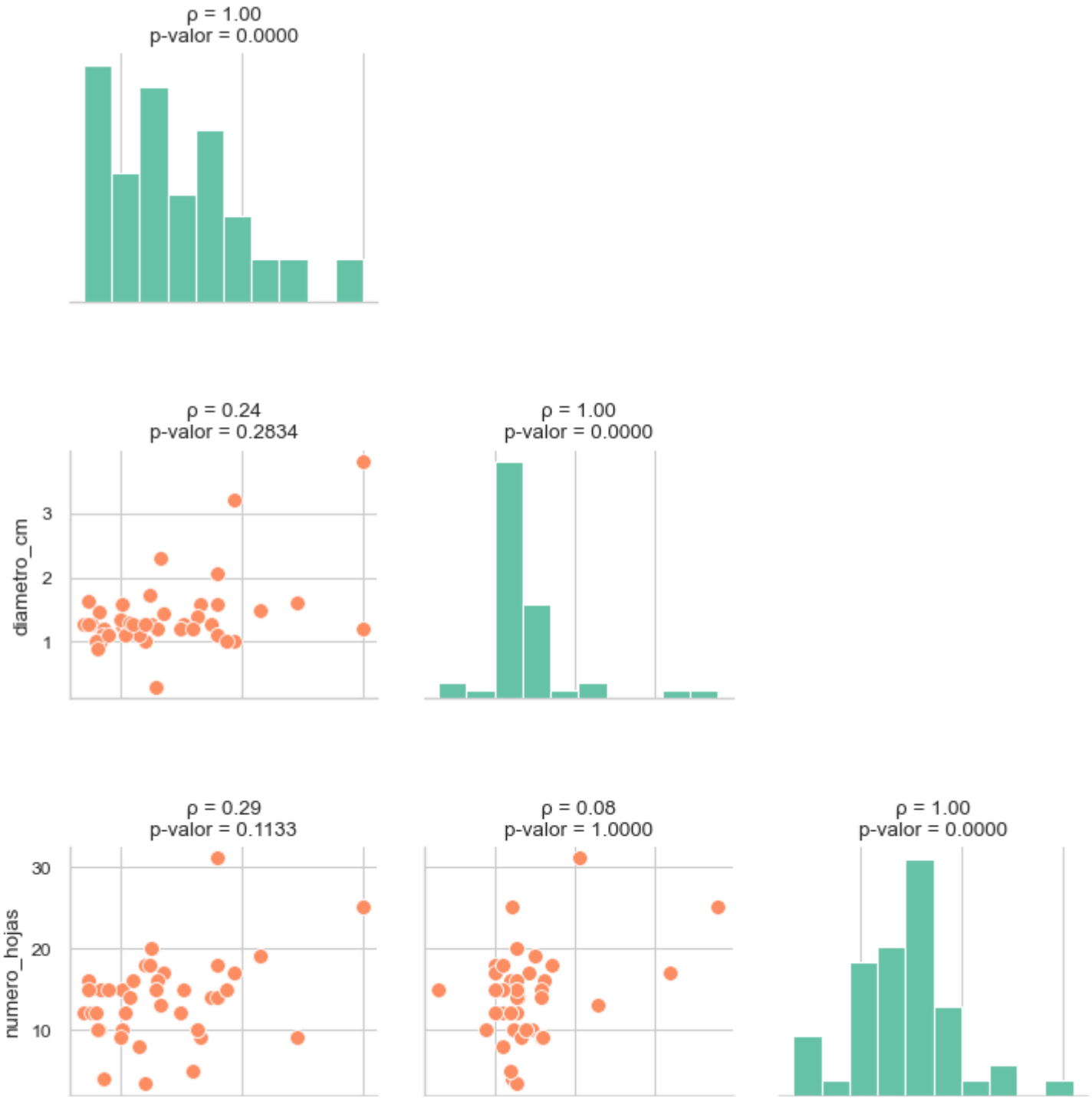
2.1. Datos sin limpiar

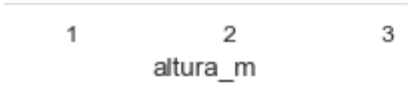
In [ ]:
```
rho_unclean, pval_corrected_unclean = spearman_correlation_analysis(araceae_raw_df)
```

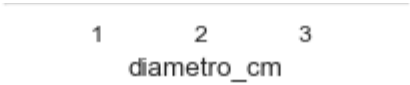
In [ ]:
```
make_heatmap(rho_unclean)
```

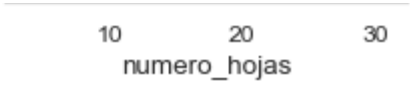

altura_m
diametro_cm
numero_hojas
1.000
0.239
1.000
0.295
0.075
1.000
altura_m
diametro_cm
numero_hojas
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

In [ ]:
```
make_scatterplot_histogram(araceae_raw_df, rho_unclean, pval_corrected_unclean)
```


ρ = 1.00
p-valor = 0.0000
ρ = 0.24
p-valor = 0.2834
ρ = 1.00
p-valor = 0.0000
diametro_cm
3
2
1
ρ = 0.29
p-valor = 0.1133
ρ = 0.08
p-valor = 1.0000
ρ = 1.00
p-valor = 0.0000
numero_hojas
30
20
10



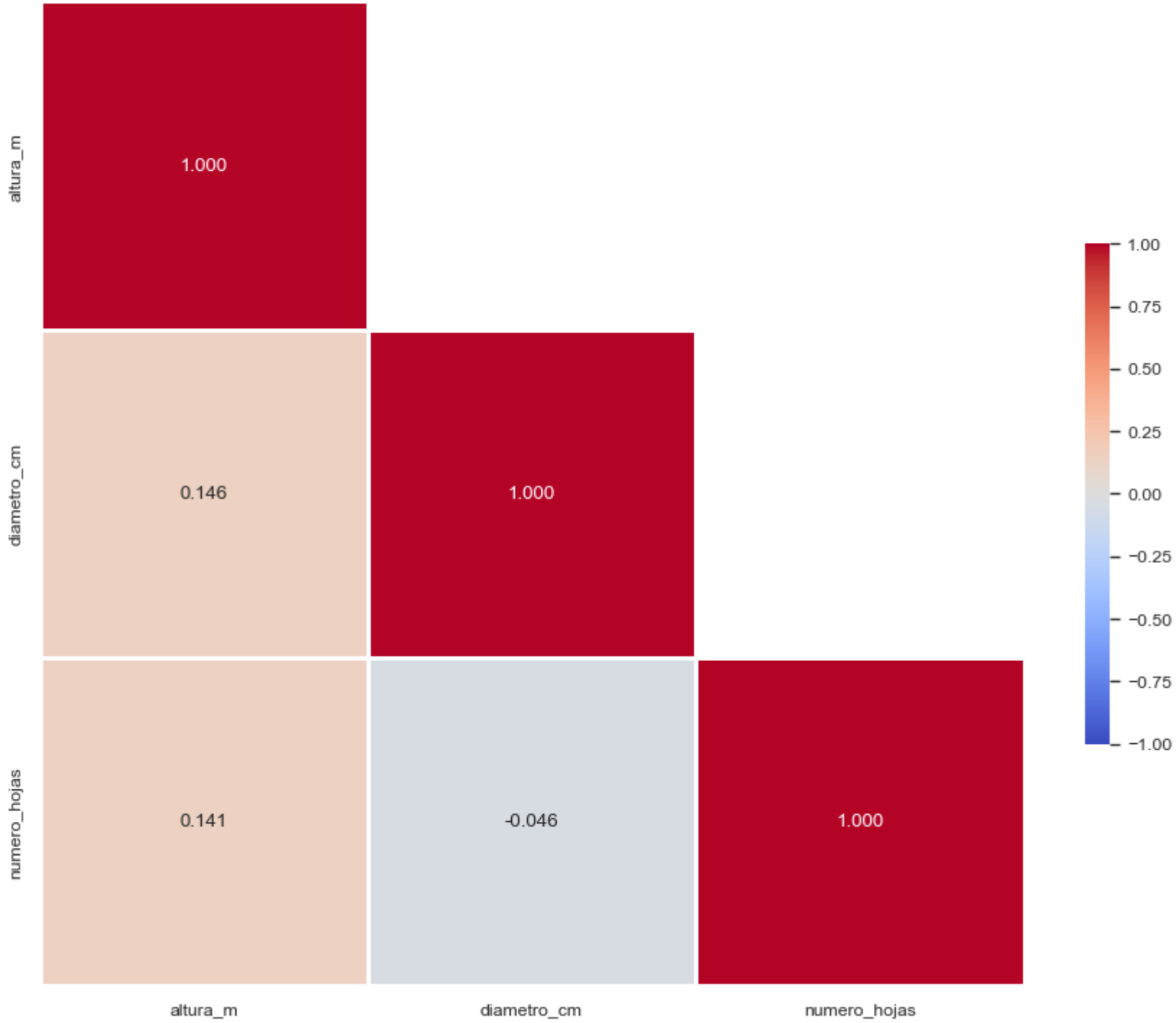




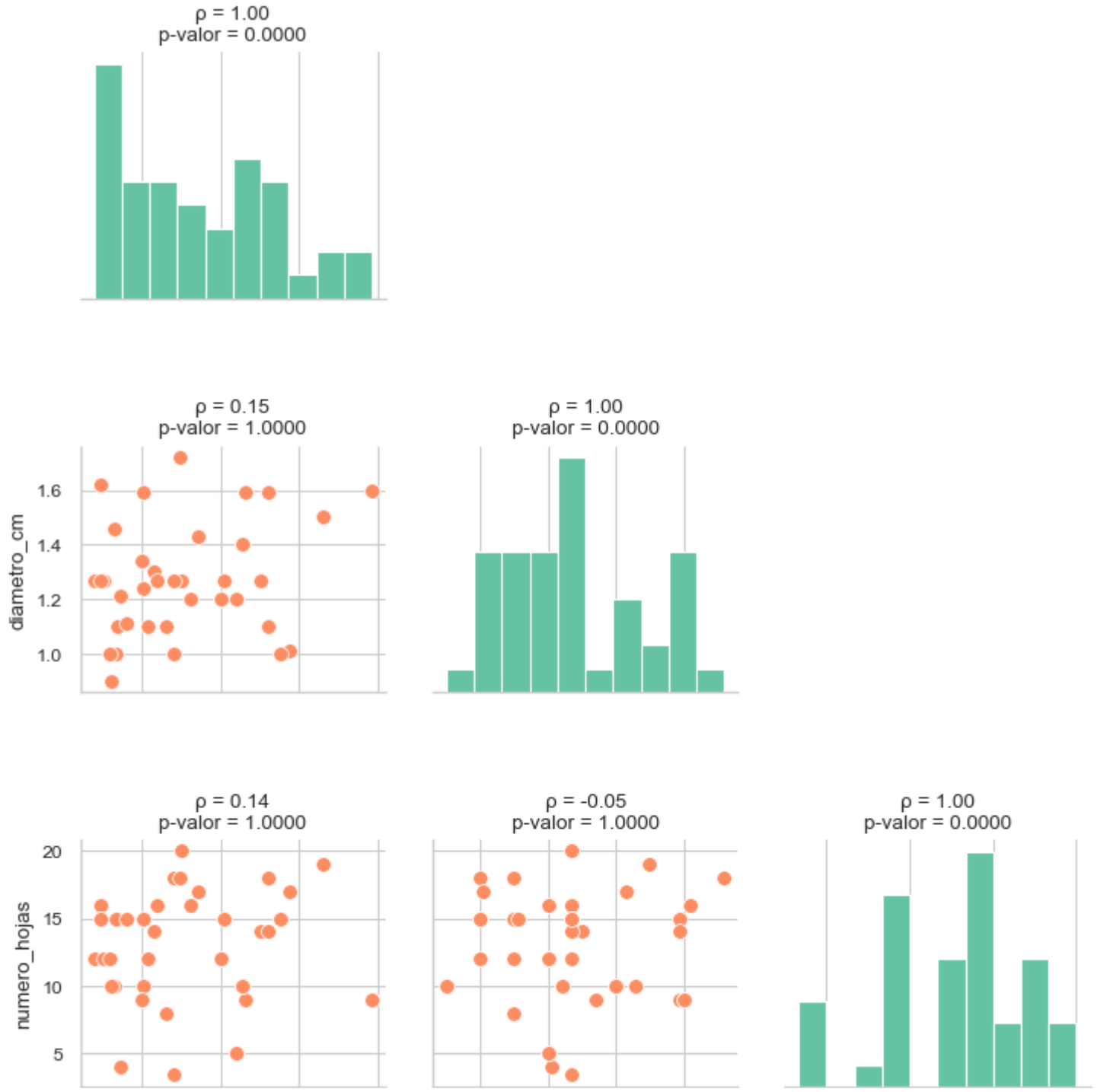
2.2. Datos sin outliers

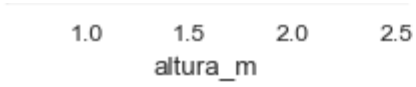
In [ ]:
```
rho_clean, pval_corrected_clean = spearman_correlation_analysis(araceae_cleaned_df)
```

In [ ]:
```
make_heatmap(rho_clean)
```

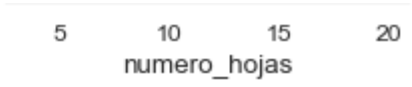

altura_m
diametro_cm
numero_hojas
1.000
0.146
1.000
0.141
-0.046
1.000
altura_m
diametro_cm
numero_hojas
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

In [ ]:
```
make_scatterplot_histogram(araceae_cleaned_df, rho_clean, pval_corrected_clean)
```


ρ = 1.00
p-valor = 0.0000
ρ = 0.15
p-valor = 1.0000
ρ = 1.00
p-valor = 0.0000
diametro_cm
1.6
1.4
1.2
1.0
ρ = 0.14
p-valor = 1.0000
ρ = -0.05
p-valor = 1.0000
ρ = 1.00
p-valor = 0.0000
numero_hojas
20
15
10
5


1.0
1.5
2.0
2.5
altura_m


1.0
1.2
1.4
1.6
diametro_cm


5
10
15
20
numero_hojas