**Diversity indices and other exploratory analysis**

import pandas as pd

# Load the data from the Excel file

file\_path = '/mnt/data/Diatoms.xlsx'

diatoms\_data = pd.read\_excel(file\_path)

# Display the first few rows of the dataframe to understand its structure

diatoms\_data.head()

# Define climate periods and their corresponding year ranges

climate\_periods = {

'RWP': (0, 200),

'DACP': (200, 550),

'S&TWP': (550, 700),

'MCA': (900, 1400),

'LIA': (1400, 1850),

'CWP': (1850, 2000)

}

# Add a new column to classify each row into a climate period

def classify\_period(age\_ce):

for period, (start, end) in climate\_periods.items():

if start <= age\_ce <= end:

return period

return 'Unknown'

diatoms\_data['Climate Period'] = diatoms\_data['Age CE'].apply(classify\_period)

# Calculate the average relative abundance of each species for each climate period

average\_abundance = diatoms\_data.groupby('Climate Period').mean().drop(columns=['Age cal yr BP', 'Age CE'])

import ace\_tools as tools; tools.display\_dataframe\_to\_user(name="Average Abundance of Diatom Species by Climate Period", dataframe=average\_abundance)

average\_abundance

import matplotlib.pyplot as plt

# Set up the plotting

fig, axes = plt.subplots(nrows=5, ncols=1, figsize=(12, 20), sharex=True)

# Plotting the relative abundance trends for some key species across different climate periods

species\_to\_plot = [

'Achnanthidium minutissimum',

'Staurosira construens',

'Amphora copulata',

'Staurosira mutabilis',

'Anomoeneis costata'

]

for ax, species in zip(axes, species\_to\_plot):

ax.plot(average\_abundance.index, average\_abundance[species], marker='o', label=species)

ax.set\_title(f'Relative Abundance of {species}')

ax.set\_ylabel('Relative Abundance (%)')

ax.legend()

ax.grid(True)

axes[-1].set\_xlabel('Climate Period')

plt.tight\_layout()

plt.show()

# Identify the dominant species in each climate period

dominant\_species = average\_abundance.idxmax(axis=1)

dominant\_abundance = average\_abundance.max(axis=1)

dominant\_species\_df = pd.DataFrame({

'Dominant Species': dominant\_species,

'Relative Abundance (%)': dominant\_abundance

})

import ace\_tools as tools; tools.display\_dataframe\_to\_user(name="Dominant Diatom Species by Climate Period", dataframe=dominant\_species\_df)

dominant\_species\_df

import seaborn as sns

plt.figure(figsize=(12, 8))

# Prepare data for seaborn violin plot

dominant\_species\_df\_reset = dominant\_species\_df.reset\_index()

# Create a violin plot for the relative abundance of the dominant species by climate period

sns.violinplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, inner=None)

# Overlay a strip plot to show individual data points

sns.stripplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, color='k', alpha=0.5)

plt.title('Relative Abundance of Dominant Diatom Species by Climate Period')

plt.ylabel('Relative Abundance (%)')

plt.xlabel('Climate Period')

plt.grid(True)

plt.show()

# Create a bar plot to visualize the dominant species and their relative abundance by climate period

plt.figure(figsize=(12, 8))

# Bar plot for the relative abundance of the dominant species by climate period

sns.barplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, palette='viridis')

# Adding value annotations on the bars

for index, row in dominant\_species\_df\_reset.iterrows():

plt.text(index, row['Relative Abundance (%)'] + 1, round(row['Relative Abundance (%)'], 2), color='black', ha="center")

plt.title('Relative Abundance of Dominant Diatom Species by Climate Period')

plt.ylabel('Relative Abundance (%)')

plt.xlabel('Climate Period')

plt.grid(axis='y')

plt.show()

plt.figure(figsize=(12, 8))

# Bar plot for the relative abundance of the dominant species by climate period

bar\_plot = sns.barplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, palette='viridis')

# Adding value and species name annotations on the bars

for index, row in dominant\_species\_df\_reset.iterrows():

plt.text(index, row['Relative Abundance (%)'] + 1,

f"{round(row['Relative Abundance (%)'], 2)}%\n{row['Dominant Species']}",

color='black', ha="center")

plt.title('Relative Abundance of Dominant Diatom Species by Climate Period')

plt.ylabel('Relative Abundance (%)')

plt.xlabel('Climate Period')

plt.grid(axis='y')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# Define the correct order of the climate periods from ancient to modern

period\_order = ['RWP', 'DACP', 'S&TWP', 'MCA', 'LIA', 'CWP', 'Unknown']

# Reorder the dataframe according to the specified period order

dominant\_species\_df\_reset['Climate Period'] = pd.Categorical(dominant\_species\_df\_reset['Climate Period'], categories=period\_order, ordered=True)

dominant\_species\_df\_reset = dominant\_species\_df\_reset.sort\_values('Climate Period')

plt.figure(figsize=(12, 8))

# Bar plot for the relative abundance of the dominant species by climate period

bar\_plot = sns.barplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, palette='viridis')

# Adding value and species name annotations on the bars

for index, row in dominant\_species\_df\_reset.iterrows():

plt.text(index, row['Relative Abundance (%)'] + 1,

f"{round(row['Relative Abundance (%)'], 2)}%\n{row['Dominant Species']}",

color='black', ha="center")

plt.title('Relative Abundance of Dominant Diatom Species by Climate Period')

plt.ylabel('Relative Abundance (%)')

plt.xlabel('Climate Period')

plt.grid(axis='y')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# Save the plot as an SVG file

svg\_file\_path = "/mnt/data/Dominant\_Diatom\_Species\_by\_Climate\_Period.svg"

plt.figure(figsize=(12, 8))

# Bar plot for the relative abundance of the dominant species by climate period

bar\_plot = sns.barplot(x='Climate Period', y='Relative Abundance (%)', data=dominant\_species\_df\_reset, palette='viridis')

# Adding value and species name annotations on the bars

for index, row in dominant\_species\_df\_reset.iterrows():

plt.text(index, row['Relative Abundance (%)'] + 1,

f"{round(row['Relative Abundance (%)'], 2)}%\n{row['Dominant Species']}",

color='black', ha="center")

plt.title('Relative Abundance of Dominant Diatom Species by Climate Period')

plt.ylabel('Relative Abundance (%)')

plt.xlabel('Climate Period')

plt.grid(axis='y')

plt.xticks(rotation=45)

plt.tight\_layout()

# Save the plot

plt.savefig(svg\_file\_path, format='svg')

# Display the path to the saved file

svg\_file\_path

# Select the most common species for the heatmap

most\_common\_species = average\_abundance.max().nlargest(15).index # Select top 15 most abundant species

# Prepare data for heatmap

heatmap\_data = average\_abundance[most\_common\_species].T

plt.figure(figsize=(15, 10))

# Create heatmap

sns.heatmap(heatmap\_data, annot=True, cmap='viridis', cbar\_kws={'label': 'Relative Abundance (%)'}, linewidths=.5)

plt.title('Temporal Distribution of Most Common Diatom Species Across Climate Periods')

plt.xlabel('Climate Period')

plt.ylabel('Diatom Species')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# Save the heatmap as an SVG file

heatmap\_svg\_file\_path = "/mnt/data/Temporal\_Distribution\_Diatom\_Species.svg"

plt.figure(figsize=(15, 10))

# Create heatmap

sns.heatmap(heatmap\_data, annot=True, cmap='viridis', cbar\_kws={'label': 'Relative Abundance (%)'}, linewidths=.5)

plt.title('Temporal Distribution of Most Common Diatom Species Across Climate Periods')

plt.xlabel('Climate Period')

plt.ylabel('Diatom Species')

plt.xticks(rotation=45)

plt.tight\_layout()

# Save the plot

plt.savefig(heatmap\_svg\_file\_path, format='svg')

# Display the path to the saved file

heatmap\_svg\_file\_path

# Define warm and cold periods

warm\_periods = ['RWP', 'S&TWP', 'MCA', 'CWP']

cold\_periods = ['DACP', 'LIA']

# Calculate average relative abundance for warm and cold periods

warm\_abundance = average\_abundance.loc[warm\_periods].mean()

cold\_abundance = average\_abundance.loc[cold\_periods].mean()

# Combine into a single dataframe for comparison

comparison\_df = pd.DataFrame({

'Warm Periods': warm\_abundance,

'Cold Periods': cold\_abundance

})

import ace\_tools as tools; tools.display\_dataframe\_to\_user(name="Comparison of Diatom Species Abundance in Warm and Cold Periods", dataframe=comparison\_df)

comparison\_df

**Comparison of Diatom Species Abundance in Warm and Cold Periods**

# Select species with significant differences

significant\_species = comparison\_df.loc[(comparison\_df['Warm Periods'] > 2) | (comparison\_df['Cold Periods'] > 2)].index

# Prepare data for plotting

comparison\_plot\_data = comparison\_df.loc[significant\_species].reset\_index().melt(id\_vars='index', var\_name='Period', value\_name='Relative Abundance (%)')

plt.figure(figsize=(14, 8))

# Create bar plot

sns.barplot(x='index', y='Relative Abundance (%)', hue='Period', data=comparison\_plot\_data, palette='viridis')

plt.title('Comparison of Diatom Species Abundance in Warm and Cold Periods')

plt.xlabel('Diatom Species')

plt.ylabel('Relative Abundance (%)')

plt.xticks(rotation=90)

plt.legend(title='Period')

plt.tight\_layout()

plt.show()

# Calculate species richness

species\_richness\_per\_period = diatoms\_data.drop(columns=['Age cal yr BP', 'Age CE']).groupby('Climate Period').apply(lambda x: (x > 0).sum(axis=1).mean())

# Calculate Shannon diversity index

shannon\_diversity\_per\_period = diatoms\_data.drop(columns=['Age cal yr BP', 'Age CE']).groupby('Climate Period').apply(lambda x: x.apply(lambda y: entropy(y, base=2), axis=1).mean())

# Combine into a single dataframe for comparison

biodiversity\_df = pd.DataFrame({

'Species Richness': species\_richness\_per\_period,

'Shannon Diversity Index': shannon\_diversity\_per\_period

})

import ace\_tools as tools; tools.display\_dataframe\_to\_user(name="Biodiversity Indices by Climate Period", dataframe=biodiversity\_df)

biodiversity\_df

# Plotting Species Richness and Shannon Diversity Index

fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(14, 12), sharex=True)

# Plot for Species Richness

sns.barplot(x=biodiversity\_df.index, y='Species Richness', data=biodiversity\_df, ax=axes[0], palette='viridis')

axes[0].set\_title('Species Richness by Climate Period')

axes[0].set\_ylabel('Species Richness')

axes[0].grid(True)

# Plot for Shannon Diversity Index

sns.barplot(x=biodiversity\_df.index, y='Shannon Diversity Index', data=biodiversity\_df, ax=axes[1], palette='viridis')

axes[1].set\_title('Shannon Diversity Index by Climate Period')

axes[1].set\_ylabel('Shannon Diversity Index')

axes[1].set\_xlabel('Climate Period')

axes[1].grid(True)

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

from scipy.stats import ttest\_ind

# Determine the minimum sample size across all climate periods

min\_sample\_size = diatoms\_data['Climate Period'].value\_counts().min()

# Function to calculate biodiversity indices

def calculate\_biodiversity\_indices(data):

# Shannon diversity index

shannon = data.apply(lambda x: entropy(x, base=2), axis=1).mean()

# Evenness index: Shannon index divided by the log of the number of species

species\_count = (data > 0).sum(axis=1)

evenness = (shannon / np.log2(species\_count + 1)).mean()

# Dominance index: 1 minus the Simpson's index

simpsons\_index = data.apply(lambda x: np.sum((x / np.sum(x))\*\*2), axis=1)

dominance = (1 - simpsons\_index).mean()

return shannon, evenness, dominance

# Initialize lists to store indices for each period

shannon\_indices = []

evenness\_indices = []

dominance\_indices = []

period\_labels = []

# Calculate indices for each climate period

for period in diatoms\_data['Climate Period'].unique():

period\_data = diatoms\_data[diatoms\_data['Climate Period'] == period]

# Randomly sample min\_sample\_size entries

sampled\_data = period\_data.sample(n=min\_sample\_size, random\_state=42)

# Drop non-species columns

sampled\_data = sampled\_data.drop(columns=['Age cal yr BP', 'Age CE', 'Climate Period'])

# Calculate indices

shannon, evenness, dominance = calculate\_biodiversity\_indices(sampled\_data)

# Append results

shannon\_indices.append(shannon)

evenness\_indices.append(evenness)

dominance\_indices.append(dominance)

period\_labels.append(period)

# Create a dataframe for the indices

indices\_df = pd.DataFrame({

'Climate Period': period\_labels,

'Shannon Index': shannon\_indices,

'Evenness Index': evenness\_indices,

'Dominance Index': dominance\_indices

})

import ace\_tools as tools; tools.display\_dataframe\_to\_user(name="Biodiversity Indices (Normalized Sample Size)", dataframe=indices\_df)

indices\_df

# Biodiversity indices normalized

# Plot the biodiversity indices

fig, axes = plt.subplots(nrows=3, ncols=1, figsize=(14, 18), sharex=True)

# Shannon Index Plot

sns.barplot(x='Climate Period', y='Shannon Index', data=indices\_df, ax=axes[0], palette='viridis')

axes[0].set\_title('Shannon Index by Climate Period')

axes[0].set\_ylabel('Shannon Index')

axes[0].grid(True)

# Evenness Index Plot

sns.barplot(x='Climate Period', y='Evenness Index', data=indices\_df, ax=axes[1], palette='viridis')

axes[1].set\_title('Evenness Index by Climate Period')

axes[1].set\_ylabel('Evenness Index')

axes[1].grid(True)

# Dominance Index Plot

sns.barplot(x='Climate Period', y='Dominance Index', data=indices\_df, ax=axes[2], palette='viridis')

axes[2].set\_title('Dominance Index by Climate Period')

axes[2].set\_ylabel('Dominance Index')

axes[2].set\_xlabel('Climate Period')

axes[2].grid(True)

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# Perform t-tests between warm and cold periods for each index

warm\_periods\_indices = indices\_df[indices\_df['Climate Period'].isin(warm\_periods)]

cold\_periods\_indices = indices\_df[indices\_df['Climate Period'].isin(cold\_periods)]

# Initialize dictionary to store p-values

p\_values = {}

# Perform t-tests for Shannon Index

t\_stat, p\_value = ttest\_ind(warm\_periods\_indices['Shannon Index'], cold\_periods\_indices['Shannon Index'])

p\_values['Shannon Index'] = p\_value

# Perform t-tests for Evenness Index

t\_stat, p\_value = ttest\_ind(warm\_periods\_indices['Evenness Index'], cold\_periods\_indices['Evenness Index'])

p\_values['Evenness Index'] = p\_value

# Perform t-tests for Dominance Index

t\_stat, p\_value = ttest\_ind(warm\_periods\_indices['Dominance Index'], cold\_periods\_indices['Dominance Index'])

p\_values['Dominance Index'] = p\_value

p\_values