import **os**

import **pandas** as **pd**

import **sys**

import **string**

import **shutil**

import **sys**

import **matplotlib**.**pyplot** as **plt**

import **datetime**

import **getpass**

output\_folder = **sys**.argv[1]

alphabet = **list**(**string**.ascii\_uppercase)

**print**("Pythonskript gestartet")

def **gen\_colnames**(n\_frames):

    alphabet = **string**.ascii\_uppercase

    names = []

    i = 0

    while **len**(names) < n\_frames:

        name = ""

        num = i

        while True:

            name = alphabet[num % 26] + name

            num = num // 26 - 1

            if num < 0:

                break

        names.**append**(f"frame{name}")

        i += 1

    return names

def **df\_combined**(dir\_path, sorted\_list):

    df\_all\_forces = None

    df\_all\_distances = None

    for filename in sorted\_list:

        file\_path = **os**.path.**join**(dir\_path, filename)

*# print(filename)*

        df = **pd**.**read\_csv**(file\_path, sep='\t', comment='#', header=None, names=["Time (ps)", "Value"]) *# comment übersprint lines mit '#' am Anfang*

        col\_name = **os**.path.**basename**(filename).replace('.xvg', '')

        df = df.**rename**(columns={"Value": col\_name})

*# print(col\_name)*

        if "pullf" in filename:

            if df\_all\_forces is None:

                df\_all\_forces = df

            else:

                df\_all\_forces = **pd**.**merge**(df\_all\_forces, df, on="Time (ps)")

        elif "pullx" in filename:

            if df\_all\_distances is None:

                df\_all\_distances = df

            else:

                df\_all\_distances = **pd**.**merge**(df\_all\_distances, df, on="Time (ps)")

        else:

**print**("Kein geeigneter Datentyp identifiziert")

*# Vorbereitungen für die Bennenung der neuen Spaltennamen hier nur für einen der Beiden Dataframes berechnet, da die Dateien immer komplementär vorliegen, also eine Force datei zu einer Distance Datei*

    anzahl\_cols = **len**(df\_all\_forces.columns) - 1  *# -1 weil Time (ps)*

    new\_colnames = ["Time (ps)"] + **gen\_colnames**(anzahl\_cols)

*# Neue Splatennamen umbenennen in den jeweiligen Dataframes*

    df\_all\_forces.columns = new\_colnames

    df\_all\_distances.columns = new\_colnames

    return df\_all\_forces, df\_all\_distances

def **sorted\_filelist**(dirname):

    xvg\_file\_names = []

    files = **os**.**listdir**(dirname)

    for file in files:

        if file.endswith(".xvg") and "combined" not in file: *# combined muss nur rein, dass wenn man das Skrpt zwei mal runt, dass die sammeldateien nicht mit ausgelesen werden*

            xvg\_file\_names.**append**(file)

    xvg\_file\_names\_sort = **sorted**(xvg\_file\_names, key=lambda x: **int**(x.split('\_')[1][5:]))

    return xvg\_file\_names\_sort

def **move\_basedata\_files**(dirname, dir\_path):

*# Zielordner erstellen, falls noch nicht vorhanden*

    basedata\_folder = **os**.path.**join**(dir\_path, "basedata")

**os**.**makedirs**(basedata\_folder, exist\_ok=True)

*# Alle Dateien im Ordner durchgehen*

    for filename in **os**.**listdir**(dir\_path):

*# vollständiger Pfad der Datei*

        full\_path = **os**.path.**join**(dir\_path, filename)

*# Nur Dateien bewegen (keine Unterordner)*

        if **os**.path.**isfile**(full\_path):

*# Prüfen, ob es NICHT die gewünschten kombinierten Dateien sind*

            if filename in [f"forces\_combined\_{dirname}.xvg", f"distances\_combined\_{dirname}.xvg"]:

                continue

            else:

*# Datei in basedata verschieben*

**shutil**.**move**(full\_path, basedata\_folder)

def **create\_mean\_stbw\_files**(dir\_path: **str**) -> None:

    run\_name = **os**.path.**basename**(dir\_path)

    file\_names = **os**.**listdir**(dir\_path)

    for file\_name in file\_names:

        if file\_name.**endswith**(".xvg"):

            file\_path = **os**.path.**join**(dir\_path, file\_name)

            curr\_df = **pd**.**read\_csv**(file\_path, sep='\t')

            mean\_val = curr\_df.**drop**(columns="Time (ps)").**mean**()

            std\_val = curr\_df.**drop**(columns="Time (ps)").**std**()

*# Forces*

            if "forces" in file\_name:

                df\_force\_mean\_std = **pd**.**DataFrame**({

                    "Frame": mean\_val.index,

                    "Force (kJ/mol/nm)": mean\_val.values,

                    "STD\_F": std\_val.values

                })

*# Distances*

            if "distances" in file\_name:

                df\_distance\_mean\_std = **pd**.**DataFrame**({

                    "Frame": mean\_val.index,

                    "Distance (nm)": mean\_val.values,

                    "STD\_D": std\_val.values

                })

*# Erst nachdem alle Files ausgelesen wurden solle dann das Mergen beginnen*

*# Merge*

    df\_merged = **pd**.**merge**(df\_distance\_mean\_std, df\_force\_mean\_std, on="Frame")

    output\_path = **os**.path.**join**(dir\_path, f"summary\_file\_{run\_name}.xvg")

    df\_merged.**to\_csv**(output\_path, sep='\t', index=False)

def **plot\_distance\_force**(dir\_path: **str**) -> None:

    run\_name = **os**.path.**basename**(dir\_path)

    file\_path = **os**.path.**join**(dir\_path, f"summary\_file\_{run\_name}.xvg")

*# Lade die Zusammenfassungsdatei*

    df\_summary = **pd**.**read\_csv**(file\_path, sep='\t')

*# Plot*

**plt**.**errorbar**(

        df\_summary["Distance (nm)"], df\_summary["Force (kJ/mol/nm)"],

        xerr=df\_summary["STD\_D"], yerr=df\_summary["STD\_F"],

        fmt='o', ecolor='gray', capsize=3, markersize=4

    )

**plt**.**title**(f"{run\_name}")

**plt**.**xlabel**("Distance (nm)")

**plt**.**ylabel**("Force (kJ/mol/nm)")

**plt**.**grid**(True)

**plt**.**tight\_layout**()

*# Speichern*

    output\_path = **os**.path.**join**(dir\_path, f"distance\_force\_plot\_{run\_name}.png")

**plt**.**savefig**(output\_path)

*# plt.show()*

**plt**.**close**()

**print**(f"Plot gespeichert: {output\_path}")

def **plot\_histogram**(dir\_path: **str**) -> None:

    run\_name = **os**.path.**basename**(dir\_path)

    files = **os**.**listdir**(dir\_path)

*# Iteriert durch die beiden combined Files aus den vorherigen Schritten*

    for file\_name in files:

        if file\_name.**endswith**(".xvg") and "combined" in file\_name:

            if "forces" in file\_name:

                prefix = "force"

                title = "Histogramm der Kräfte"

                xlabel = "Force (kJ/mol/nm)"

            elif "distances" in file\_name:

                prefix = "distance"

                title = "Histogramm der Distanzen"

                xlabel = "Distance (nm)"

*# Daten aus der Datei laden*

            file\_path = **os**.path.**join**(dir\_path, file\_name)

            df = **pd**.**read\_csv**(file\_path, sep="\t")

            df = df.**drop**(columns=["Time (ps)"])  *# Zeitspalte entfernen da nicht benötigt*

*# Transformation der Daten in ein 1D Array/ Liste*

            all\_values = df.values.**flatten**()

*# Histogramm plotten*

**plt**.**figure**(figsize=(8, 5))

**plt**.**hist**(all\_values, bins=30, density=True, edgecolor='black')  *# density=True → relative Häufigkeit*

**plt**.**title**(title)

**plt**.**xlabel**(xlabel)

**plt**.**ylabel**("Relative Häufigkeit")

**plt**.**grid**(True)

**plt**.**tight\_layout**()

*# plt.show()*

*# Speichern*

            output\_path = **os**.path.**join**(dir\_path, f"Histogramm\_{prefix}\_{run\_name}.png")

**plt**.**savefig**(output\_path)

*# plt.show()*

**plt**.**close**()

**print**(f"Plot gespeichert: {output\_path}")

def **create\_report**(output\_folder: **str**) -> None:

    output\_path = **os**.path.**join**("output", "summary\_report.txt")

    report\_lines = []

    username = **getpass**.**getuser**()

**print**(username)

*# Header*

    report\_lines.**append**("# REPORT")

    report\_lines.**append**(f"# Created on {**datetime**.**date**.**today**()} by {username}")

    report\_lines.**append**("#")

    report\_lines.**append**("# Run\tForce variation(kJ/mol/nm)\tDistance variation(nm)")

    for item\_name in **os**.**listdir**(output\_folder): *# Geht jeden Run ordner durch*

        if "run" in item\_name:

            dir\_path = **os**.path.**join**(output\_folder, item\_name)

            run\_name = **os**.path.**basename**(dir\_path) *# Hier könnte man auch einfach das Item selst benutzen*

*# Pfade zu den bereits erstellten Zusammenfassungsdateien*

            summary\_file = **os**.path.**join**(dir\_path, f"summary\_file\_{run\_name}.xvg")

            if not **os**.path.**exists**(summary\_file):

**print**(f"summary\_file{run\_name} existiert nicht.")

                continue

            df\_summary = **pd**.**read\_csv**(summary\_file, sep="\t") *# Einlesen des Files*

*# Mittelwert der Standardabweichungen für Force und Distance berechnen*

            std\_f = df\_summary["STD\_F"].**mean**()

            std\_d = df\_summary["STD\_D"].**mean**()

            report\_lines.**append**(f"{run\_name}\t{std\_f:.4f}\t{std\_d:.4f}")

*# Report schreiben*

    with **open**(output\_path, "w") as f:

        f.**write**("\n".**join**(report\_lines))

*#!!!!!!!!!!!!!!!!!!!!!!!*

*#!!! Executive Part !!!#*

*#!!!!!!!!!!!!!!!!!!!!!!!*

for dirname in **os**.**listdir**(output\_folder): *# Iteriert durch jeden Inhalt in dem Ordner/Verzeichnis "output"*

    dir\_path = **os**.path.**join**(output\_folder, dirname) *# Fügt die Namen der Inhalte mit dem Ursprungspfad zu einem gemeinsamen Pfad zusammen*

    if "run" in dirname:

*# create\_mean\_stbw\_files(dir\_path)*

*# plot\_distance\_force(dir\_path)*

*# plot\_histogram(dir\_path)*

        if "basedata" in **os**.**listdir**(dir\_path): *# Notwendig um beim erneuten Runnen des Files keine Fehler zu erzeugen*

            continue

        xvg\_file\_names\_sort = **sorted\_filelist**(dir\_path)

        df\_all\_forces, df\_all\_distances = **df\_combined**(dir\_path, xvg\_file\_names\_sort)

*# Erstellte Dateien outputen*

        output\_path = **os**.path.**join**(dir\_path, f"forces\_combined\_{dirname}.xvg")

        df\_all\_forces.**to\_csv**(output\_path, sep='\t', index=False)

        output\_path = **os**.path.**join**(dir\_path, f"distances\_combined\_{dirname}.xvg")

        df\_all\_distances.**to\_csv**(output\_path, sep='\t', index=False)

*# Ordner aufräumen*

**move\_basedata\_files**(dirname, dir\_path)

*# Erstellen des Files mit den jeweiligen Durchschnittswerten und zugehöriger STBW*

**create\_mean\_stbw\_files**(dir\_path)

*# Erstellt die Plots mit Force über Distance*

**plot\_distance\_force**(dir\_path)

*# Erstellt die Histogramme mit den Häufigkeiten auf der y-Achse und den möglichen Werten auf der x-Achse*

**plot\_histogram**(dir\_path)

    else:

        continue

*# Erstellt den Abschlussreport*

**create\_report**(output\_folder)

*# TODO Dataframes werden in run1 erst ab Sekunden 1.6 gemerged, da die xvg Dateien 0 jeweils bei 1.6 starten sollen die alle bei 0 starten oder die sollen herausgelassen werden?*

"""

#TODO

Ich solle nun folgendes tuen, ich habe ja in vorherigen Schritten die kombinierten files (distance\_combined\_run1.xvg, force\_combined\_run1.xvg)

erstellt. nun möchte ich auf grundlage dessen ein histogram erstellen, welche mit die realtive häufigkeit eines wertes auf der y achse anzeigt und auf der x-achse den zugehörigen wert

wie kann ich das umsetzen?

ich habe ja folgenden aufbau des dataframes

Time (ps)   frameA  frameB  frameC

1.6 2.46268 2.49449 2.5302  2.51886

1.7 2.46264 2.49544 2.52615 2.51938

... ... ... ... ...

Das ganze dann jeweils für das Force File und das Distance File

"""

"""

#TODO

Danach solle ich noch eine reportdatei erstellen, welche folgenden Aufbau hat.

# REPORT

# Created on <DATE> by <NAME>

#

# Run TAB Force variation(kJ/mol/nm) TAB Distance variation(nm)

Run\_1 ... ...

Run\_2

Run\_3

das TAB steht dabei für die trennung über TAB also \t

"""

*# ######## für eine Datei*

*# # Ließt die Dateien ein und wandelt diese in einen Dataframe um*

*# df = pd.read\_csv(file\_path, sep='\t', comment='#', header=None, names=["Time (ps)", "Value"]) # comment übersprint lines mit '#' am Anfang*

*# # Nimmt den aktuellen Dataframe und fügt diesen an den "Sammeldataframe" an*

*# df\_all\_distances = pd.merge(df\_all\_distances, df[['Time (ps)', 'Force']], on='Time (ps)')*

*# # Benennt die angehängte Spalte um*

*# df\_all\_distances = df\_all\_distances.rename(columns={'Distance': 'frameA'})*

*# for file in files:*

*#     # Datei einlesen*

*#     df = pd.read\_csv(file, sep='\s+', comment='#', header=None, names=["Time (ps)", "Value"])*

*#     # Eindeutigen Namen für die Wertespalte aus dem Dateinamen*

*#     col\_name = os.path.basename(file).replace('.xvg', '')*

*#     # Umbenennen der 'Value'-Spalte zu etwas Aussagekräftigem*

*#     df = df.rename(columns={"Value": col\_name})*

*# if combined\_df is None:*

*#     # Beim ersten Mal übernehmen wir den gesamten DataFrame*

*#     combined\_df = df*

*# else:*

*#     # Danach mergen wir über die 'Time (ps)'-Spalte*

*#     combined\_df = pd.merge(combined\_df, df, on="Time (ps)")*

"""

#TODO

Ich solle nun folgendes tuen, ich habe ja in vorherigen Schritten die kombinierten files (distance\_combined\_run1.xvg, force\_combined\_run1.xvg)

erstellt. nun möchte ich auf grundlage dessen ein histogram erstellen, welche mit die realtive häufigkeit eines wertes auf der y achse anzeigt und auf der x-achse den zugehörigen wert

wie kann ich das umsetzen?

ich habe ja folgenden aufbau des dataframes

Time (ps)   frameA  frameB  frameC

1.6 2.46268 2.49449 2.5302  2.51886

1.7 2.46264 2.49544 2.52615 2.51938

... ... ... ... ...

Das ganze dann jeweils für das Force File und das Distance File

"""

"""

#TODO

Danach solle ich noch eine reportdatei erstellen, welche folgenden Aufbau hat.

# REPORT

# Created on <DATE> by <NAME>

#

# Run TAB Force variation(kJ/mol/nm) TAB Distance variation(nm)

Run\_1 ... ...

Run\_2

Run\_3

das TAB steht dabei für die trennung über TAB also \t

"""