Dataset Analysis of Protein Structures

Dataset Overview

In this notebook, we load a curated list of PDB identifiers, download their corresponding structures, and compute the following metrics for each protein:

- **Residue count**: total number of amino acid residues.
- **Chain count**: number of polypeptide chains.
- **Center of mass**: 3D coordinates of the center of mass (X, Y, Z).
- **Average backbone angles**: mean φ (phi) and ψ (psi) angles in degrees.

Dataset Description

The dataset contains 32 protein structures chosen to capture four broad functional classes:

Category	Examples	Purpose
Enzymes	1AKE, 1H2W, 4WNC	Classic catalytic models for RMSD benchmarking
Small single–domain proteins	1CRN, 1BRS	Compact folds for φ/ψ angle statistics
Membrane proteins / receptors	1A0S, 2RH1, 3MKT	Trans-membrane topology diversity
Large multichain complexes	4V4Q, 6VXX	Stress-testing performance on big assemblies

The table below lists every PDB identifier, its functional family, the total number of residues (computed on the fly), and the source (all structures were downloaded from the RCSB PDB archive).

In [1]:
```
%pip install -r ../requirements.txt
```

```
Requirement already satisfied: biopython>=1.79 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 1)) (1.85)
Requirement already satisfied: matplotlib>=3.5.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 2)) (3.10.3)
Requirement already satisfied: numpy>=1.23.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 3)) (2.2.6)
Requirement already satisfied: flask>=3.1.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 4)) (3.1.1)
Requirement already satisfied: gunicorn>=20.1.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 5)) (23.0.0)
Requirement already satisfied: pytest>=7.0.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 6)) (8.3.5)
Requirement already satisfied: requests>=2.28.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from -r ../requirements.txt (line 7)) (2.32.4)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (1.3.2)
Requirement already satisfied: cycler>=0.10 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (4.58.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (25.0)
Requirement already satisfied: pillow>=8 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (11.2.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (2.9.0.post0)
Requirement already satisfied: blinker>=1.9.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (1.9.0)
Requirement already satisfied: click>=8.1.3 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (8.2.1)
Requirement already satisfied: itsdangerous>=2.2.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (2.2.0)
Requirement already satisfied: jinja2>=3.1.2 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (3.1.6)
Requirement already satisfied: markupsafe>=2.1.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (3.0.2)
Requirement already satisfied: werkzeug>=3.1.0 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from flask>=3.1.1->-r ../requirements.txt (line 4)) (3.1.3)
Requirement already satisfied: colorama in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from pytest>=7.0.0->-r ../requirements.txt (line 6)) (0.4.6)
Requirement already satisfied: iniconfig in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from pytest>=7.0.0->-r ../requirements.txt (line 6)) (2.1.0)
Requirement already satisfied: pluggy<2,>=1.5 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from pytest>=7.0.0->-r ../requirements.txt (line 6)) (1.6.0)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from requests>=2.28.0->-r ../requirements.txt (line 7)) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from requests>=2.28.0->-r ../requirements.txt (line 7)) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from requests>=2.28.0->-r ../requirements.txt (line 7)) (2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from requests>=2.28.0->-r ../requirements.txt (line 7)) (2025.6.15)
Requirement already satisfied: six>=1.5 in c:\users\user\pycharmprojects\protein_explorer\.venv\lib\site-packages (from python-dateutil>=2.7->matplotlib>=3.5.0->-r ../requirements.txt (line 2)) (1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```python
import os
import pandas as pd
import numpy as np
import requests
from Bio.PDB import PDBList, PDBParser
from Bio.PDB.MMCIFParser import MMCIFParser
import plotly.express as px

pdb_list_file = '../data/pdb_list.txt'
```

In [3]:
```python
with open(pdb_list_file) as f:
    pdb_ids = [line.strip() for line in f if line.strip()]

pdb_ids
```

Out[3]:
```
['1AKE   # adenylate kinase',
 '2PTC   # trypsinвЋ“inhibitor complex',
 '1LYZ   # hen-egg-white lysozyme',
 '1D66   # dihydrofolate reductase',
 '1H2W   # HIV-1 protease',
 '1H4L   # cytochrome P450-cam',
 '4WNC   # human acetylcholinesterase',
 '2C7E   # alanine racemase',
 '5E8A   # catalytic antibody 48G7',
 '3G5U   # T7 RNA polymerase',
 '1CRN   # crambin (plant protein)',
 '1BRS   # barnase',
 '1PPT   # pancreatic trypsin inhibitor (BPTI)',
 '1TIT   # titin I27 domain',
 '2WXC   # WW domain of Pin1',
 '1A0S   # bacteriorhodopsin',
 '1A0I   # KcsA potassium channel',
 '2POR   # OmpF porin',
 '2RH1   # OI2-adrenergic GPCR',
 '3SN6   # Oj-opioid GPCR',
 '3MKT   # leucine transporter (LeuT)',
 '6CP6   # mitochondrial ATP-synthase F1 domain',
 '5I6X   # human GLUT1 glucose transporter',
 '4V4Q   # E. coli 70S ribosome (small subunit)',
 '3J3Q   # yeast mitochondrial ribosome',
 '7KOO   # human 80S ribosome (snapshot)',
 '6VXX   # SARS-CoV-2 spike trimer',
 '5L93   # human clathrin coat']
```

In [4]:
```python
from Bio.PDB.vectors import calc_dihedral
from Bio.PDB import PPBuilder

pdbl = PDBList()
parser = PDBParser(QUIET=True)

STRUCT_DIR = "structures"
os.makedirs(STRUCT_DIR, exist_ok=True)

def download_structure(pdb_id, out_dir=STRUCT_DIR):
    pdb_id = pdb_id.lower()
    pdb_path = os.path.join(out_dir, f"{pdb_id}.pdb")
    cif_path = os.path.join(out_dir, f"{pdb_id}.cif")

    # уже скачан?
    if os.path.isfile(pdb_path):
        return pdb_path
    if os.path.isfile(cif_path):
        return cif_path

    # пробуем PDB
    url_pdb = f"https://files.rcsb.org/download/{pdb_id}.pdb"
    r = requests.get(url_pdb)
    if r.ok and len(r.text) > 100:
        with open(pdb_path, "w") as f:
            f.write(r.text)
        print(f"Downloaded PDB '{pdb_id}'")
        return pdb_path

    # пробуем CIF
    url_cif = f"https://files.rcsb.org/download/{pdb_id}.cif"
    r = requests.get(url_cif)
    if r.ok and len(r.text) > 100:
        with open(cif_path, "w") as f:
            f.write(r.text)
        print(f"Downloaded CIF '{pdb_id}'")
        return cif_path

    raise FileNotFoundError(f"No PDB/CIF found for {pdb_id}")

def parse_structure(path):
    if path.endswith(".pdb"):
        parser = PDBParser(QUIET=True)
    elif path.endswith(".cif"):
        parser = MMCIFParser(QUIET=True)
    else:
        raise ValueError("Unsupported format")
    struct_id = os.path.basename(path).split('.')[0]
    return parser.get_structure(struct_id, path)
def compute_center_of_mass(struct):
    coords = []
    masses = []
    for atom in struct.get_atoms():
        if hasattr(atom, 'mass'):
```

```
            coords.append(atom.get_coord() * atom.mass)
            masses.append(atom.mass)
    if not masses:
        return (float('nan'), float('nan'), float('nan'))
    com = np.sum(coords, axis=0) / np.sum(masses)
    return tuple(com)

def compute_phi_psi(struct):
    angles = []
    for model in struct:
        for chain in model:
            for poly in PPBuilder().build_peptides(chain):
                for phi, psi in poly.get_phi_psi_list():
                    if phi and psi:
                        angles.append((phi, psi))
    return angles
```

In [5]:
```
results = []
for pdb_id in pdb_ids:
    pdb_id = pdb_id[0:4]
    path = download_structure(pdb_id)
    struct = parse_structure(path)
    residues = [res for res in struct.get_residues() if res.id[0] == ' ']
    residue_count = len(residues)
    chain_count = len({chain.id for chain in struct.get_chains()})
    center_of_mass = compute_center_of_mass(struct)
    phi_psi = compute_phi_psi(struct)
    if phi_psi:
        avg_phi = np.degrees(np.mean([a for a,_ in phi_psi]))
        avg_psi = np.degrees(np.mean([b for _,b in phi_psi]))
    else:
        avg_phi = float('nan')
        avg_psi = float('nan')
    results.append({'pdb_id': pdb_id, 'residue_count': residue_count, 'chain_count': chain_count,
                    'center_x': center_of_mass[0], 'center_y': center_of_mass[1], 'center_z': center_of_mass[2]
                    'avg_phi': avg_phi, 'avg_psi': avg_psi})

df = pd.DataFrame(results)
print(df.head())
print(df.describe())
```

```
Downloaded PDB '7koo'
Downloaded PDB '6vxx'
Downloaded PDB '5l93'
  pdb_id  residue_count  chain_count   center_x   center_y   center_z  \
0   1AKE            428            2  20.184478  25.612156  20.331380
1   2PTC            281            2  10.865600  70.390938  17.345945
2   1LYZ            129            1  -0.411743  20.632387  19.084470
3   1D66            152            4  27.459647  41.773087  28.397540
4   1H2W            710            1  36.327195  41.427252  88.999231

     avg_phi    avg_psi
0 -68.610796  25.445530
1 -78.083670  67.282326
2 -63.558812  18.431132
3 -70.433316  37.974446
4 -81.041085  58.647656
       residue_count  chain_count    center_x    center_y    center_z  \
count      28.000000    28.000000   28.000000   28.000000   28.000000
mean    13106.714286    56.928571   39.644866   51.374000   43.768366
std     58953.845281   255.362477  103.577866  121.870345  102.340069
min        36.000000     1.000000  -36.960645  -24.995074  -94.236221
25%       296.000000     1.000000   -0.726389   -0.236386    5.901621
50%       780.000000     2.500000   12.166653   15.203633   20.758394
75%      2327.250000     6.500000   35.824746   48.586068   40.989514
max    313236.000000  1356.000000  497.961359  610.332677  471.068158

         avg_phi    avg_psi
count  28.000000  28.000000
mean  -74.292477  32.384134
std    10.981704  30.891503
min   -95.869610 -36.424408
25%   -81.258475  17.027068
50%   -73.336133  24.618345
75%   -66.070764  51.120125
max   -55.379707  94.118098
```

Center of Mass Scatter (X vs Y)

This scatter plot displays the X and Y coordinates of each protein's center of mass.

- Each point represents one PDB structure.

- Color encodes the number of chains in that structure (warmer colors = more chains).

By examining this plot, we can see how protein mass is distributed spatially (in the XY plane) and whether multi-chain complexes tend to occupy different spatial regions than single-chain proteins.

Center of Mass Scatter (X vs Y)

This scatter plot displays the X and Y coordinates of each protein's center of mass.

- Each point represents one PDB structure.
- Color encodes the number of chains in that structure (warmer colors = more chains).

By examining this plot, we can see how protein mass is distributed spatially (in the XY plane) and whether multi-chain complexes tend to occupy different spatial regions than single-chain proteins.

Average Phi and Psi Angle Distributions

Here we compare the backbone dihedral angles across all proteins:

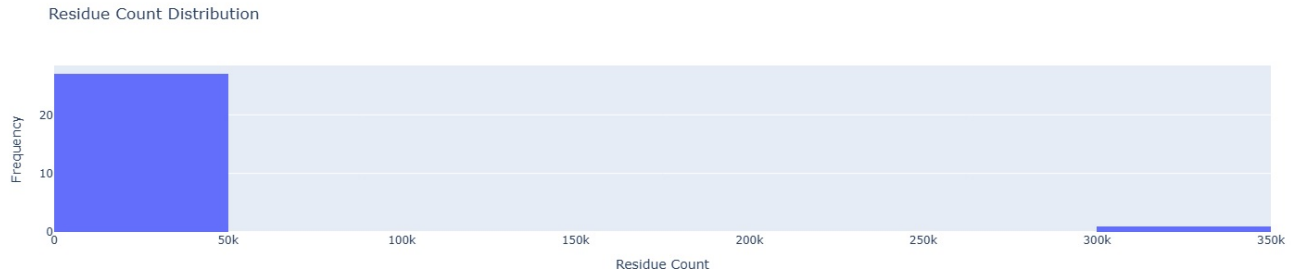
- **avg_phi** box shows the distribution of mean φ angles.
- **avg_psi** box shows the distribution of mean ψ angles.

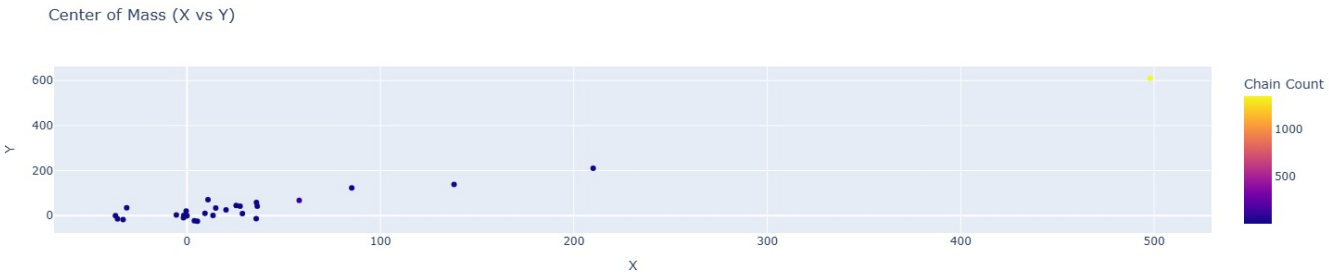
Box plots summarize the median, interquartile range, and outliers for each angle.
This analysis provides insight into the typical backbone conformations in our dataset.

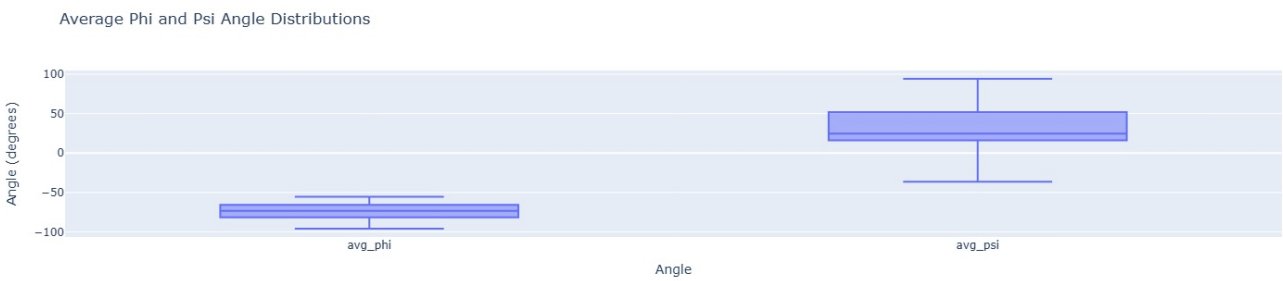
In [7]:
```
fig1 = px.histogram(df, x='residue_count', title='Residue Count Distribution')
fig1.update_xaxes(title='Residue Count')
fig1.update_yaxes(title='Frequency')
fig1.show()

fig2 = px.scatter(df, x='center_x', y='center_y', color='chain_count',
                  title='Center of Mass (X vs Y)', labels={'center_x':'X', 'center_y':'Y', 'chain_count':'Chain
fig2.show()

fig3 = px.box(df, y=['avg_phi', 'avg_psi'], title='Average Phi and Psi Angle Distributions',
              labels={'value':'Angle (degrees)', 'variable':'Angle'})
fig3.show()
```







Exporting Metrics to CSV

Finally, we save the compiled metrics into `results/dataset_metrics.csv`, which can be used for further downstream analysis or shared with collaborators.

In [8]:
```
os.makedirs('results', exist_ok=True)
df.to_csv('results/dataset_metrics.csv', index=False)
```

In [ ]:
```
```