Osdag Screening Task: 2D and 3D Visualization of Shear Force and Bending Moment Diagrams

Name: Yash Kashyap

---

1. Objective

The objective of this task is to extract shear force and bending moment data from an Xarray dataset and visualize them using 2D and 3D plots. The visualizations are generated for bridge girders using actual node coordinates and element connectivity data, similar to MIDAS post-processing.

2. Dataset Description

The project utilizes a NetCDF dataset (xarray_data.nc) which contains structural analysis results. This data is processed using the Python xarray library.

- **Format:** Xarray Dataset (NetCDF).
- **Key Variable:** forces
- **Force Components:** The dataset stores forces at the start (i) and end (j) of each element:
    - **Mz_i, Mz_j:** Bending Moment about the Z-axis.
    - **Vy_i, Vy_j:** Shear Force along the Y-axis.
- **Geometry Data:** Node coordinates and element connectivity are imported from external Python modules (node.py and element.py) to map the forces to physical 3D space.

3. Task-1: 2D SFD & BMD

This task focused on visualizing the internal forces for a single **Central Longitudinal Girder** to verify the data extraction logic and general force distribution.

Implementation Details

- **Target Elements:** A specific list of central girder elements was isolated (IDs: 15, 24, 33, 42, 51, 60, 69, 78, 83).
- **Bending Moment Diagram (BMD):**
    - The Bending Moment (M_z) was plotted using a standard line plot.
    - To smooth the visualization for the 2D representation, the values were averaged between nodes i and j ($M_{avg} = (M_{z\_i} + M_{z\_j}) / 2$).
    - The area under the curve was filled (red) to highlight the magnitude.
- **Shear Force Diagram (SFD):**
    - The Shear Force (V_y) was plotted using a **step-wise** method (plt.step).
    - This accurately represents the constant shear behavior across finite elements typical in FEA results.
    - The diagram preserves the sign convention, showing positive and negative shear zones clearly.

Code: -

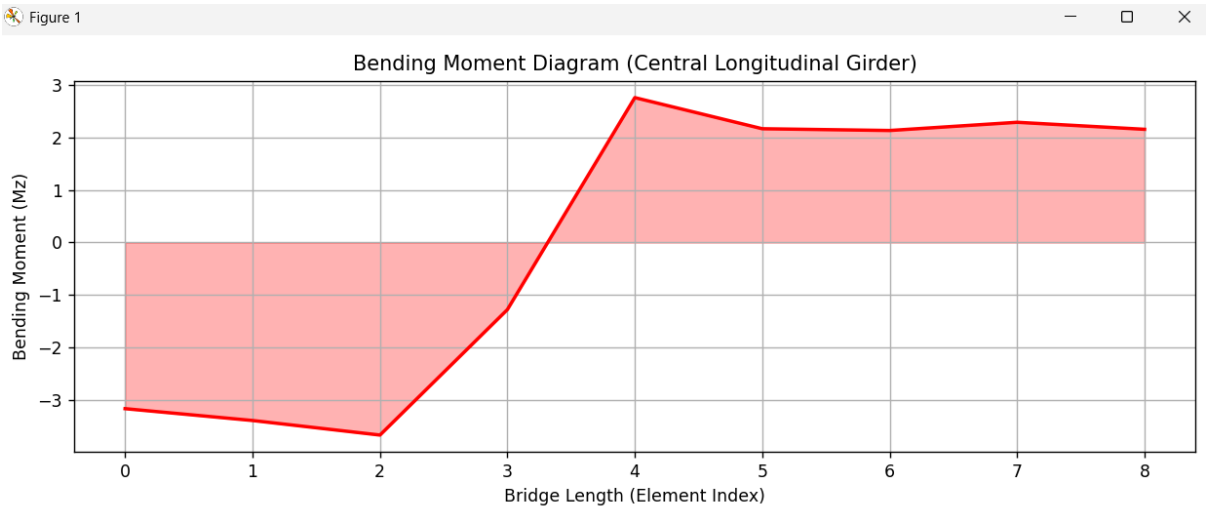
```
import xarray as xr
import matplotlib.pyplot as plt
ds = xr.open_dataset("../data/xarray_data.nc")
central_elements = [15, 24, 33, 42, 51, 60, 69, 78, 83]
x_bmd = []
bmd_vals = []
for i, elem in enumerate(central_elements):
    ef = ds["forces"].sel(Element=elem)
    Mz_i = float(ef.sel(Component="Mz_i"))
    Mz_j = float(ef.sel(Component="Mz_j"))
    Mz_avg = (Mz_i + Mz_j) / 2
    x_bmd.append(i)
    bmd_vals.append(Mz_avg)
plt.figure(figsize=(10, 4))
plt.plot(x_bmd, bmd_vals, color="red", linewidth=2)
plt.fill_between(x_bmd, bmd_vals, 0, color="red", alpha=0.3)
plt.title("Bending Moment Diagram (Central Longitudinal Girder)")
plt.xlabel("Bridge Length (Element Index)")
plt.ylabel("Bending Moment (Mz)")
plt.grid(True)
plt.tight_layout()
plt.show()
x_sfd = []
sfd_vals = []
pos = 0
for elem in central_elements:
    ef = ds["forces"].sel(Element=elem)
    Vy_i = float(ef.sel(Component="Vy_i"))
    Vy_j = float(ef.sel(Component="Vy_j"))
```

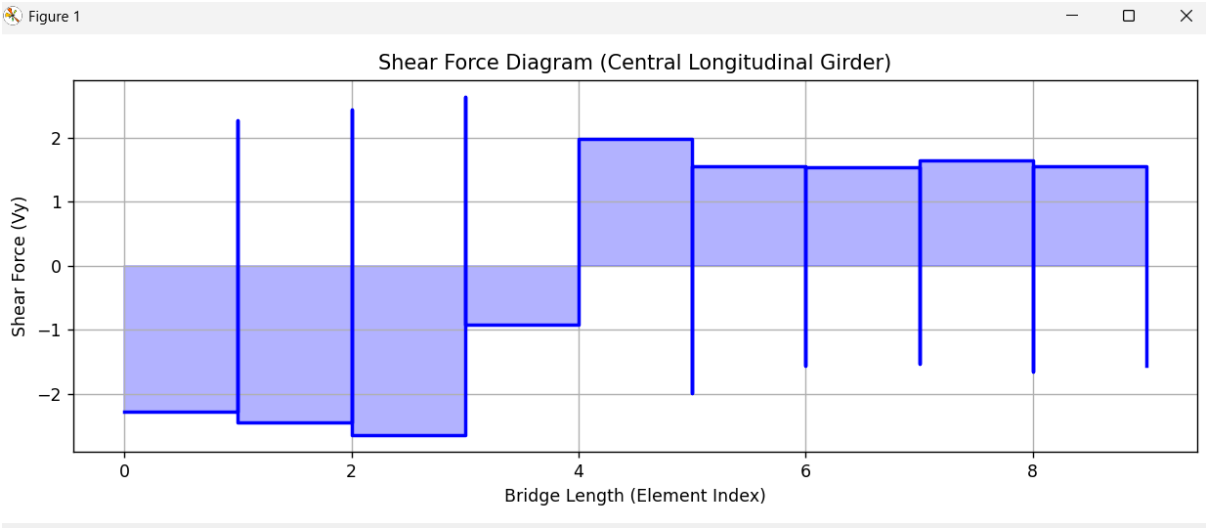
```
    x_sfd.extend([pos, pos + 1])
    sfd_vals.extend([Vy_i, Vy_j])
    pos += 1
plt.figure(figsize=(10, 4))
plt.step(x_sfd, sfd_vals, where="post", color="blue", linewidth=2)
plt.fill_between(x_sfd, sfd_vals, 0, step="post", color="blue", alpha=0.3)
plt.title("Shear Force Diagram (Central Longitudinal Girder)")
plt.xlabel("Bridge Length (Element Index)")
plt.ylabel("Shear Force (Vy)")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Visual Output

Two distinct figures were generated:

1. **BMD:** Red continuous plot showing the variation of moment along the bridge length.
2. **SFD:** Blue stepped plot showing the shear transitions at element boundaries.





4. Task-2: 3D SFD & BMD

This task expanded the visualization to the full bridge structure, plotting forces for all five longitudinal girders simultaneously in a 3D environment.

Implementation Details

- **Geometry Mapping:** The node.py and element.py files were utilized to retrieve (x, y, z) coordinates for every element end-point.
- **3D Extrusion (Y-Direction):**
    - To visualize the magnitude of forces in 3D, the force values were used as the vertical "height" (Y-axis) in the plot.
    - The X and Z coordinates corresponded to the physical length and width of the bridge, respectively.
- **Grouping & Coloring:**
    - Elements were grouped into five distinct girders ("Girder 1" through "Girder 5").
    - Each girder was assigned a unique color (Red, Blue, Green, Purple, Orange) to distinguish them visually in the 3D space.
- **Reference Frame:** A gray wireframe of the bridge structure was plotted as a baseline (alpha=0.3) to provide spatial context for the force diagrams.
- **Scaling:** Scaling factors (BMD_SCALE=0.3, SFD_SCALE=1.0) were applied to the force values to ensure the diagrams were legible and proportional to the bridge dimensions.

Code: -

```
import sys

import os

sys.path.append(os.path.abspath("../data"))
```

```
import xarray as xr
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from node import nodes
from element import members
ds = xr.open_dataset("../data/xarray_data.nc")
girders = {
    "Girder 1": [13, 22, 31, 40, 49, 58, 67, 76, 81],
    "Girder 2": [14, 23, 32, 41, 50, 59, 68, 77, 82],
    "Girder 3": [15, 24, 33, 42, 51, 60, 69, 78, 83],
    "Girder 4": [16, 25, 34, 43, 52, 61, 70, 79, 84],
    "Girder 5": [17, 26, 35, 44, 53, 62, 71, 80, 85],
}
girder_colors = {
    "Girder 1": "red",
    "Girder 2": "blue",
    "Girder 3": "green",
    "Girder 4": "purple",
    "Girder 5": "orange",
}
BMD_SCALE = 0.3
SFD_SCALE = 1.0
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection="3d")
for elem, (n1, n2) in members.items():
    x1, _, z1 = nodes[n1]
    x2, _, z2 = nodes[n2]
    ax.plot([x1, x2], [0, 0], [z1, z2], color="gray", alpha=0.3)
for gname, elements in girders.items():
    color = girder_colors[gname]
    for elem in elements:
```

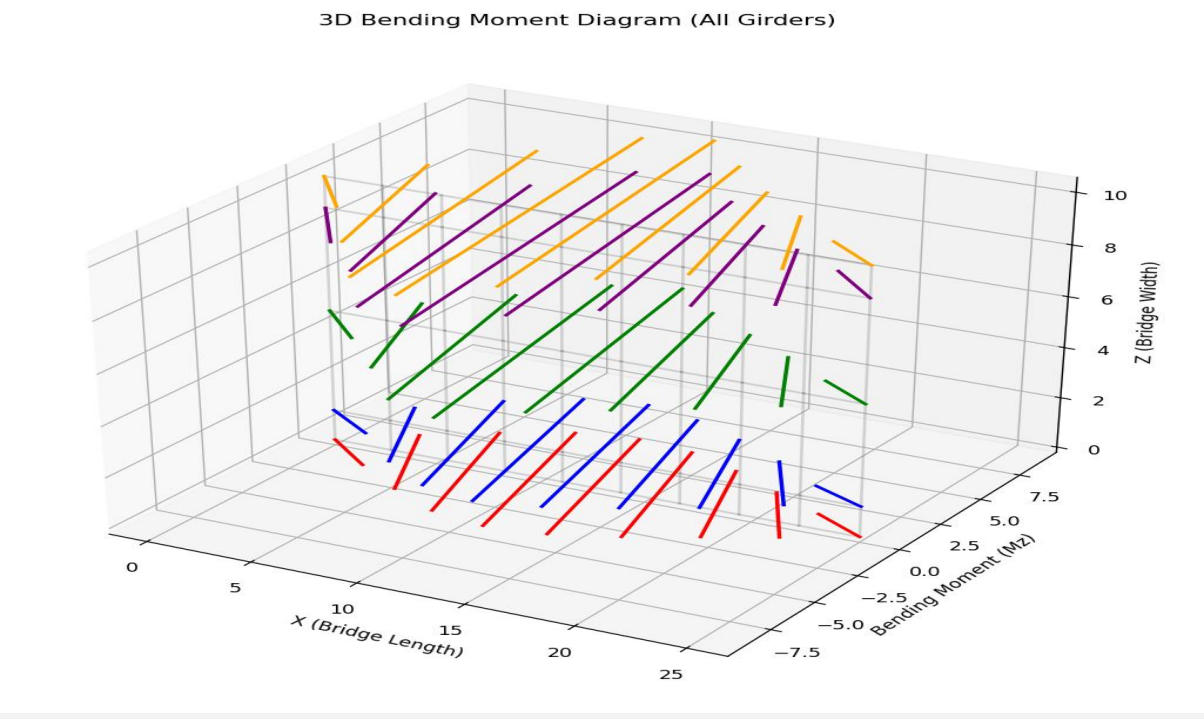
```
        n1, n2 = members[elem]
        x1, _, z1 = nodes[n1]
        x2, _, z2 = nodes[n2]
        ef = ds["forces"].sel(Element=elem)
        Mz_i = float(ef.sel(Component="Mz_i"))
        Mz_j = float(ef.sel(Component="Mz_j"))
        ax.plot(
            [x1, x2],
            [BMD_SCALE * Mz_i, BMD_SCALE * Mz_j],
            [z1, z2],
            color=color,
            linewidth=2
        )
ax.set_title("3D Bending Moment Diagram (All Girders)")
ax.set_xlabel("X (Bridge Length)")
ax.set_ylabel("Bending Moment (Mz)")
ax.set_zlabel("Z (Bridge Width)")
plt.tight_layout()
plt.show()
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection="3d")
for elem, (n1, n2) in members.items():
    x1, _, z1 = nodes[n1]
    x2, _, z2 = nodes[n2]
    ax.plot([x1, x2], [0, 0], [z1, z2], color="gray", alpha=0.3)
for gname, elements in girders.items():
    color = girder_colors[gname]
    for elem in elements:
        n1, n2 = members[elem]
        x1, _, z1 = nodes[n1]
        x2, _, z2 = nodes[n2]
```

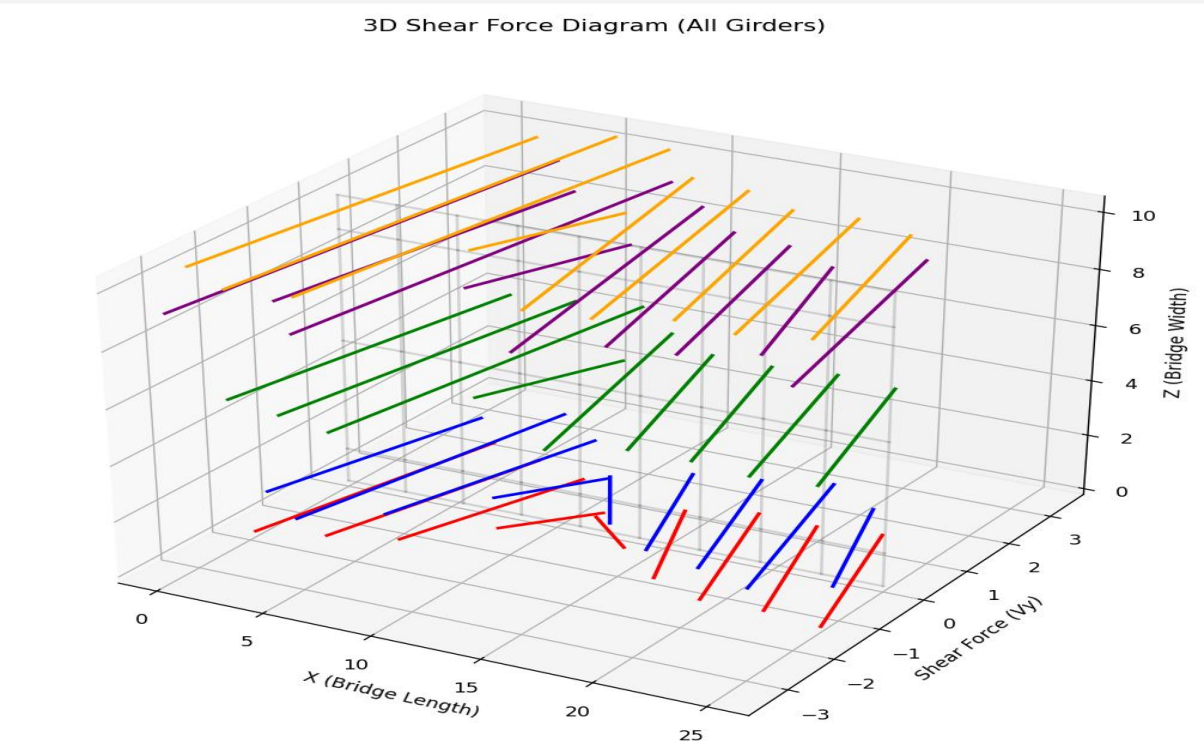
```
        ef = ds["forces"].sel(Element=elem)
        Vy_i = float(ef.sel(Component="Vy_i"))
        Vy_j = float(ef.sel(Component="Vy_j"))
        ax.plot(
            [x1, x2],
            [SFD_SCALE * Vy_i, SFD_SCALE * Vy_j],
            [z1, z2],
            color=color,
            linewidth=2
        )
ax.set_title("3D Shear Force Diagram (All Girders)")
ax.set_xlabel("X (Bridge Length)")
ax.set_ylabel("Shear Force (Vy)")
ax.set_zlabel("Z (Bridge Width)")
plt.tight_layout()
plt.show()
```

Visual Output

The script generates two interactive 3D plots:

1. **3D Bending Moment:** Visualizes the sagging/hogging behavior across the entire bridge deck width.
2. **3D Shear Force:** Visualizes the shear distribution, allowing for quick comparison of load paths between outer and inner girders.





5. Conclusion

This project successfully demonstrated the extraction and visualization of structural engineering data using Python. By leveraging xarray for data handling and matplotlib for rendering, we replicated standard post-processing capabilities found in commercial software like MIDAS. The 2D plots provided detailed verification for individual members, while the 3D plots offered a holistic view of the bridge's structural response, effectively highlighting critical zones across multiple girders.