FiberForge: enabling high-throughput simulations of the mechanical properties of helical fibrils

Kieran Nehil-Puleo[1], Zhongyue Yang[2-6]

[1]Interdisciplinary Material Science Program, Vanderbilt University, 2301 Vanderbilt Pl, Nashville, TN 37235, United States

[2]Department of Chemistry, Vanderbilt University, Nashville, Tennessee 37235, United States

[3]Center for Structural Biology, Vanderbilt University, Nashville, Tennessee 37235, United States

[4]Vanderbilt Institute of Chemical Biology, Vanderbilt University, Nashville, Tennessee 37235, United States[5]*Data Science Institute, Vanderbilt University, Nashville, Tennessee 37235, United States*

[6]Department of Chemical and Biomolecular Engineering, Vanderbilt University, Nashville, Tennessee 37235, United States

Contents

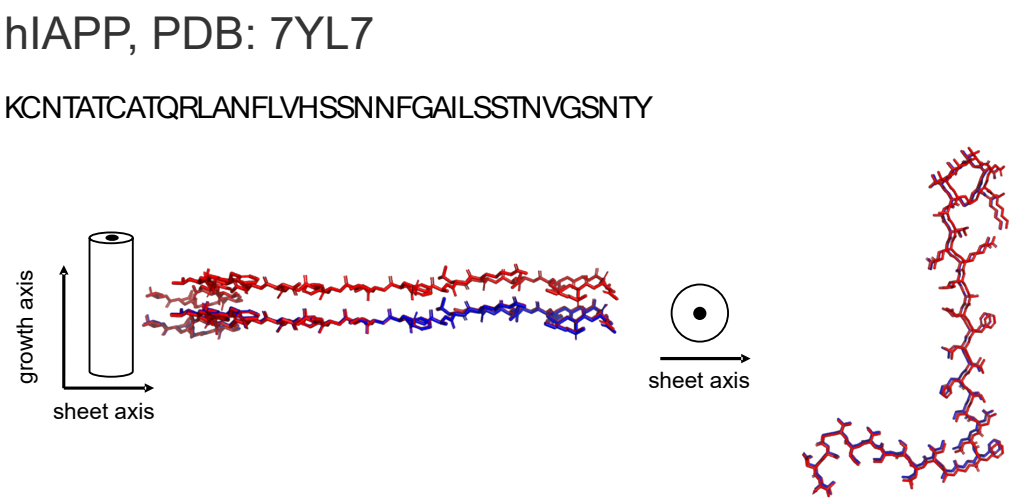

Figure S1: One of the best performing reconstructions (0.07 Å). Note that there is almost no difference between the crystal and the predicted structure.

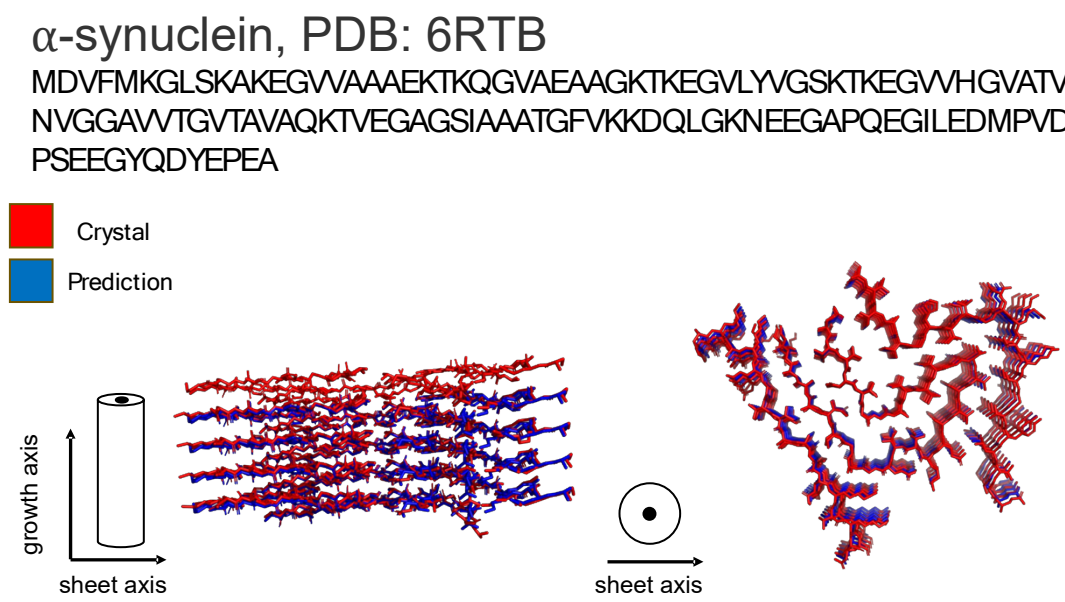

Figure S2: The worst performing reconstruction (3.6 Å) from the dataset. Note the divergence of the outer right region of the predicted structure from the crystal structure (image on Left).

```
from FiberForge.Build import (
  clean_structure
  identify_protofibrils,
  calculate_average_helical_parameters,
  build_fibril,
  remove_chains,
  solvate_fibril
)

# The path to your amyloid structure in PDB format
pdb_file = "amyloid.pdb"

# Locate the protofibrils, returns List[dict[str, np.array(3)]]
protofibrils = identify_protofibrils(pdb_file)

# If multiple protofibril structure, remove other chains
remove_chains(
    pdb_file,
    "protofibril.pdb", # Saves to protofibril.pdb
    chains_to_remove = [] # for example we assume fibril made of 1 protofibril
)

# Calculate the best theta and t
theta, t, growth_axis = calculate_average_helical_parameters("protofibril.pdb")

build_fibril(
    "protofibril.pdb",
    theta,
    t,
    n_units=20,
    output_file = "extended_protofibril.pdb"
)

solvate_fibril(
    input_structure="extended_protofibril.gro",
    output_gro="solvated.gro",
    topol_file="topol.top",
    box=box,
    water_model='spc216.gro',
    gmx_path='module load gromacs;gmx',
    expand_box=False
)
```

Figure S3: Example of using the FiberForge.build module. In this example we identifiy the protofibrils from an experimental PDB file, remove chains so a single protofibril remains, calculate the helical parameters characterizing the structure, elongate the fibril to 20 units, then solvate the fibril. Refer to https://github.com/ChemBioHTP/FiberForge/blob/main/README.md for further details.

Alter the fiberverse database location for your machine init.py, create the list of pdb_ids you want to simulate, specify n_replicates, specify pulling conditions, specify the chains you wish to pull.

```
python init.py
```

After you have initialized your jobs with python init.py perform the following operations to run simulations on your system

```
python project.py run -o preprocess_pdb
python project.py run -o create_eq_submission
python project.py run -o run_equilibration
```

After minimization and equilibration have finished run:

```
python project.py run -o create_pull_mdp
python project.py run -o create_pull_submission
python project.py run -o preprocess_pull
python project.py run -o run_pull
```

Finally, after the pulling simulation has finished run:

```
python project.py run -o run_analysis
```

Figure S4: Example of using the FiberForge.simulate module to perform a tensile test of a fibril system. Refer to https://github.com/ChemBioHTP/FiberForge/blob/main/README.md for further details.

```
from FiberForge.analyze import calculate_cross_sectional_area

pdb_file = 'extended_protofibril.pdb'
project_path = 'path/to/files'

cross_sectional_area = calculate_cross_sectional_area(pdb_file)
cross_sectional_area = cross_sectional_area * 1e-20 # A^2 to m^2

# Calculate the strain in units of nm
time_length = calculate_variable_over_time(project_path + '/4_smd/pull_pullx.xvg')
length_over_time = np.array([l for (t, l) in time_length])
strain = (length_over_time - length_over_time[0]) / length_over_time[0]

# Calculate the stress
time_force = calculate_variable_over_time(project_path + '/4_smd/pull_pullf.xvg')
force_over_time = np.array([f for (t, f) in time_force])*(1e-9)*(1/1000) # kJ/mol/nm to N
stress = force_over_time / cross_sectional_area

# Calculate the ultimate tensile strength
ultimate_tensile_strength = np.max(stress)

# Calculate the elastic modulus, assuming there is no plastic deformation
E, yield_point = estimate_elastic_modulus(stress, strain)
```

Figure S5: Example of using the FiberForge.analyze module to calculate the stress and strain induced from a tensile test simulation. Refer to https://github.com/ChemBioHTP/FiberForge/blob/main/README.md for further details.

Figure S6: Spearman rank correlation matrix (left) and corresponding p-value matrix (right) computed for key structural and sequence-derived features across the amyloid fiber dataset. Features include elastic modulus, ultimate tensile strength, rotational and translational helical parameters, mean hydrophobicity (Kyte–Doolittle scale), and interchain hydrogen-bond density. Only the lower triangular portion of the matrices is shown for clarity. Color intensity reflects the magnitude of the correlation (left) or statistical significance (right).

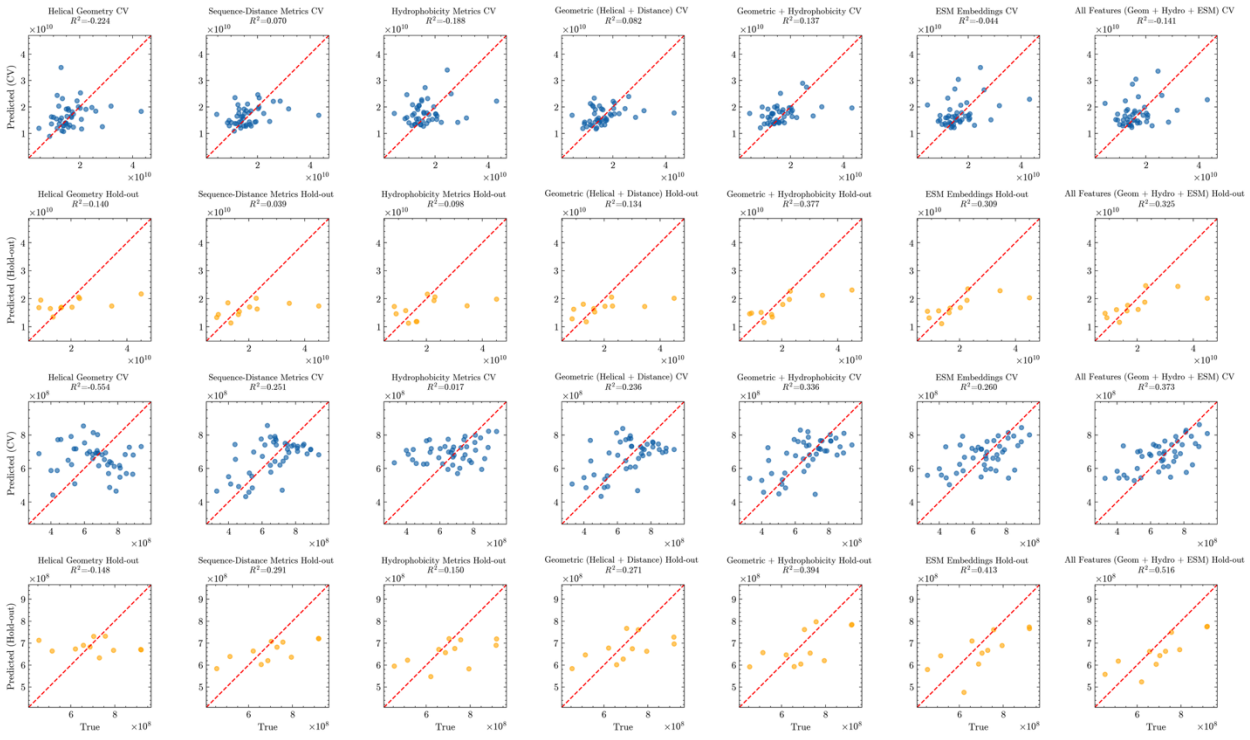

Figure S7: Parity plots showing predicted versus true values for each target and feature group. Top row (blue points) for each target shows 10-fold cross-validation predictions on the training set, and the bottom row shows hold-out predictions from the final model trained on all training data. First 2 rows are for prediction of elastic modulus, last 2 rows are for UTS prediction. Feature groups are: “Helical Geometry” (rotation, translation), “Sequence-Distance Metrics” (H-bond count, cross-sectional area, H-bond density), “Hydrophobicity Metrics” (mean, standard deviation, and fraction of hydrophobic residues), “Geometric (Helical + Distance)” (rotation, translation, H-bond count, cross-sectional area, H-bond density), “Geometric + Hydrophobicity” (all geometric and

hydrophobicity metrics), “ESM Embeddings” (sequence embeddings), and “All Features (Geom + Hydro + ESM)” (all features). Dashed lines indicate perfect parity, and R^2 values are reported for each panel.