Write-Up:

The structures are very different from each other, although the highest 3 scores are very close to each other. This means that sampling has not yet converged. These structures are still quite far from the native structure. The score is still not very low yet, and it is visibility noticeable that the structures are not super similar.

The total energies of the sampled decoys are very far off from the relaxed natives. Although most of the Rosetta score terms are fairly different, the fa_rep, fa_elec weights, omega, and rama_prepro are the most far apart. In regards to the constraints, the atom_pair_constraint's are the most far off from each other.

The energies of the new structures are a little better, but overall, they still look pretty different from the native structures. Even structures with energies in the 100's do not look very similar to the native structures.

For my improvements, I generated protein fragments from the Robetta server (http://robetta.bakerlab.org/fragmentsubmit.jsp) from the fasta sequences. I then made a PyRosetta Fragment mover which, each time the mover is applied, selects a random 3-mer window and inserts the backbone torsion angles from a random matching fragment in the fragment set. Then, I added the constraints and relaxed the pose incrementally as in Part 2. This method results in structures with very low energies (<100). The structures start looking very similar to the native conformations.

Code:

```
import nglview as nv
import pyrosetta
pyrosetta.init('-mute all')


import os

FILES_PATH = "/Users/tgoel/Downloads/Classes/GENOME/GENOME541/hw2/"
def get_pose_from_file(file_name):
    file_n = os.path.join(FILES_PATH, file_name)
    return pyrosetta.pose_from_file(file_n)

def get_pose_from_fasta(file_name):
    file_n = os.path.join(FILES_PATH, file_name)
    with open(file_n) as f:
        sequence = f.read().split("\n")[1]
    return pyrosetta.pose_from_sequence(sequence)
```

```
import numpy as np

def set_backbone_torsions(pose):
    for i in range(pose.total_residue()):
        pose.set_phi(i+1, np.random.randint(-180, 180))
        pose.set_psi(i+1, np.random.randint(-180, 180))


def set_constraints(pose, res1, res2): # set constraints on the target pose between the Cbeta of
residue res1 and residue res2
    id_i = pyrosetta.rosetta.core.id.AtomID(pose.residue(res1).atom_index("CB"), res1)
    id_j = pyrosetta.rosetta.core.id.AtomID(pose.residue(res2).atom_index("CB"), res2)
    ijfunc = pyrosetta.rosetta.core.scoring.constraints.BoundFunc(0.0, 6.0, 1.0, 'cst1');
    cst_ij = pyrosetta.rosetta.core.scoring.constraints.AtomPairConstraint(id_i, id_j, ijfunc)
    pose.add_constraint(cst_ij)

def set_constraints_with_file(pose, file_name):
    with open(FILES_PATH + file_name, "r") as f:
        constraints = f.read().strip().split("\n")
    for constraint in constraints:
        res1, res2 = constraint.split()
        set_constraints(pose, int(res1), int(res2))


scorefxn =
pyrosetta.rosetta.core.scoring.ScoreFunctionFactory.create_score_function("ref2015_cst.wts")
def relax_pose(pose, macrocycles=5):
    fastrelax = pyrosetta.rosetta.protocols.relax.FastRelax(scorefxn, macrocycles)
    fastrelax.apply(pose)
    return scorefxn.score(pose)


def fold_protein(protein, show=False):
    display(nv.show_file(FILES_PATH + protein + ".pdb")) if show else None #
nv.get_pose_from_file(FILES_PATH + protein + ".pdb")

    pose = get_pose_from_fasta(FILES_PATH + protein + ".fasta")
    set_backbone_torsions(pose)
    display(nv.show_rosetta(pose)) if show else None

    set_constraints_with_file(pose, protein + ".contacts")
    score = relax_pose(pose)
    display(nv.show_rosetta(pose)) if show else None
    return (pose, score) if not show else score

PROTEIN_ID = "6qfj_A"
# fold_protein(PROTEIN_ID, show=False)


import time

OUT_PATH = "/Users/tgoel/Downloads/"
def fold(prot, fold_fn, folds=10, show=False, save=False):

    prot_scores = []
    for i in range(folds):
        start_time = time.time()
        pose, score = fold_fn(prot)
        prot_scores.append((pose, score))
        end_time = time.time()
        print(prot, "fold", i+1, "completed --> score:", f"{score:3f}", "time:",
f"{end_time-start_time:2f}", "seconds")

    display(nv.show_file(FILES_PATH + prot + ".pdb")) if show else None

    prot_scores.sort(key=lambda x: x[1])
    for i, pose_score in enumerate(prot_scores[:3]):
        pose, score = pose_score
        if save:
            pose.dump_pdb(OUT_PATH + prot + "_" + str(i) + ".pdb")
        if show:
            display(nv.show_rosetta(pose))


# fold("5gua_A", fold_fn=fold_protein, show=True, save=False)
# fold("5h9h_C", fold_fn=fold_protein, show=True, save=False)
# fold("6ipy_A", fold_fn=fold_protein, show=True, save=False)
# fold("6qfj_A", fold_fn=fold_protein, show=True, save=False)


def relax_native(protein, show=False):
    pose = get_pose_from_file(FILES_PATH + protein + ".pdb")
    set_constraints_with_file(pose, protein + ".contacts")
    score = relax_pose(pose)
    print(scorefxn.show(pose))
    display(nv.show_rosetta(pose)) if show else None
    return score


# relax_native("5gua_A", show=True)
# relax_native("5h9h_C", show=True)
# relax_native("6ipy_A", show=True)
# relax_native("6qfj_A", show=True)


# https://pubmed.ncbi.nlm.nih.gov/11276088

residues_data = {
    'G': [(87, 7), (-66, -35), (70, -144), (105, 170), (-171, 177), (-87, 163)],
    'D': [(-140, 165), (-78, 141), (-108, 103), (-97, 5), (-64, -39), (57, 39)],
    'N': [(-140, 165), (-78, 141), (-108, 103), (-97, 5), (-64, -39), (57, 39)],
    'I': [(-132, 153), (-86, 127), (-118, 125), (-91, -9), (-63, -42), (57, 39)],
    'V': [(-132, 153), (-86, 127), (-118, 125), (-91, -9), (-63, -42), (57, 39)],
    'P': [(-64, 145), (-60, -29), (-60, -29), (-77, 161), (-77, 161), (-84, -2)],
    '.': [(-136, 153), (-76, 143), (-112, 119), (-91, -9), (-63, -42), (57, 39)]
}

def set_backbone_torsionsV2(pose):
    angle = np.random.randint(6)
    for i, residue in enumerate(list(pose.residues)):
        amino = residue.name1()
        amino = '.' if amino not in residues_data else amino
        phi, psi = residues_data[amino][angle]
        pose.set_phi(i+1, phi)
        pose.set_psi(i+1, psi)


def get_constraints_with_file(file_name):
    with open(FILES_PATH + file_name, "r") as f:
        constraints = f.read().strip().split("\n")
        constraints = [x.split() for x in constraints]
        constraints = [(int(x[0]), int(x[1])) for x in constraints]
        distances = [abs(x[0] - x[1]) for x in constraints]
        combined = list(zip(constraints, distances))
        combined.sort(key=lambda x: x[1])
    sorted_constraints = [x[0] for x in combined]
    return sorted_constraints


def fold_proteinV2(protein, show=False):
    display(nv.show_file(FILES_PATH + protein + ".pdb")) if show else None #
nv.get_pose_from_file(FILES_PATH + protein + ".pdb")

    pose = get_pose_from_fasta(FILES_PATH + protein + ".fasta")
    set_backbone_torsionsV2(pose)
    display(nv.show_rosetta(pose)) if show else None

    constraints = get_constraints_with_file(protein + ".contacts")
    threshold = 5
    for constraint in constraints:
        if abs(constraint[1] - constraint[0]) > threshold:
            relax_pose(pose, macrocycles=1)
            threshold += 5
        set_constraints(pose, constraint[0], constraint[1])
    score = relax_pose(pose, macrocycles=1)
    display(nv.show_rosetta(pose)) if show else None

    return (pose, score) if not show else score

# fold_proteinV2(PROTEIN_ID, show=False)


# fold("5gua_A", fold_fn=fold_proteinV2, show=True, save=False)
# fold("5h9h_C", fold_fn=fold_proteinV2, show=True, save=False)
# fold("6ipy_A", fold_fn=fold_proteinV2, show=True, save=False)
# fold("6qfj_A", fold_fn=fold_proteinV2, show=True, save=False)


with open(FILES_PATH + PROTEIN_ID + ".fasta", "r") as f:
    print(f.read().strip().split("\n")[1])
print(get_pose_from_fasta(FILES_PATH + PROTEIN_ID + ".fasta").sequence())
print("".join([i.name1() for i in get_pose_from_fasta(FILES_PATH + PROTEIN_ID +
".fasta").residues]))
# http://bioinf.cs.ucl.ac.uk/psipred/&psipred_uuid=57f6d54a-d58b-11ed-b4fb-00163e100d53


coil_helix_strand = {1: (-140, 130), # helix
                     2: (-60, -50)} # strand

# http://bioinf.cs.ucl.ac.uk/psipred/&uuid=3be4ad06-d58f-11ed-b4fb-00163e100d53
psipred = [0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 2,
           2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 2, 2, 2, 2, 2, 2, 2,
           2, 2, 2, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
           1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0]
```

```
def set_backbone_torsionsV3(pose, psipreds):
    for i, chs in enumerate(psipreds):
        phi, psi = coil_helix_strand[chs] if chs != 0 else (np.random.randint(-180, 180),
np.random.randint(-180, 180))
        pose.set_phi(i+1, phi)
        pose.set_psi(i+1, psi)


def fold_proteinV3(protein, show=False):
    display(nv.show_file(FILES_PATH + protein + ".pdb")) if show else None #
nv.get_pose_from_file(FILES_PATH + protein + ".pdb")

    pose = get_pose_from_fasta(FILES_PATH + protein + ".fasta")
    set_backbone_torsionsV3(pose, psipred)
    display(nv.show_rosetta(pose)) if show else None

    constraints = get_constraints_with_file(protein + ".contacts")
    threshold = 5
    for constraint in constraints:
        if abs(constraint[1] - constraint[0]) > threshold:
            relax_pose(pose, macrocycles=1)
            threshold += 5
        set_constraints(pose, constraint[0], constraint[1])
    score = relax_pose(pose, macrocycles=1)
    display(nv.show_rosetta(pose)) if show else None

    return (pose, score) if not show else score

# fold_proteinV3(PROTEIN_ID, show=False)


# http://robetta.bakerlab.org/fragmentsubmit.jsp
# http://old.robetta.org/downloads/fragments/79542/
# save http://old.robetta.org/downloads/fragments/79542/aat000_09_05.200_v1_3 to
fragments.txt

fragset = pyrosetta.rosetta.core.fragment.ConstantLengthFragSet(3)
fragset.read_fragment_file(OUT_PATH + "fragments.txt")


def fold_proteinV4(protein, show=False):
    display(nv.show_file(FILES_PATH + protein + ".pdb")) if show else None #
nv.get_pose_from_file(FILES_PATH + protein + ".pdb")
```

```
    pose = get_pose_from_fasta(FILES_PATH + protein + ".fasta")
    movemap = pyrosetta.MoveMap()
    movemap.set_bb(True)
    mover_3mer = pyrosetta.rosetta.protocols.simple_moves.ClassicFragmentMover(fragset,
movemap)
    pmm = pyrosetta.PyMOLMover()
    pmm.send_movemap(pose, movemap)
    for i in range(500):
        mover_3mer.apply(pose)
        pmm.apply(pose)
    display(nv.show_rosetta(pose)) if show else None

    constraints = get_constraints_with_file(protein + ".contacts")
    threshold = 5
    for constraint in constraints:
        if abs(constraint[1] - constraint[0]) > threshold:
            relax_pose(pose, macrocycles=1)
            threshold += 5
        set_constraints(pose, constraint[0], constraint[1])
    score = relax_pose(pose, macrocycles=1)
    display(nv.show_rosetta(pose)) if show else None

    return (pose, score) if not show else score

# fold_proteinV4(PROTEIN_ID, show=True)
```