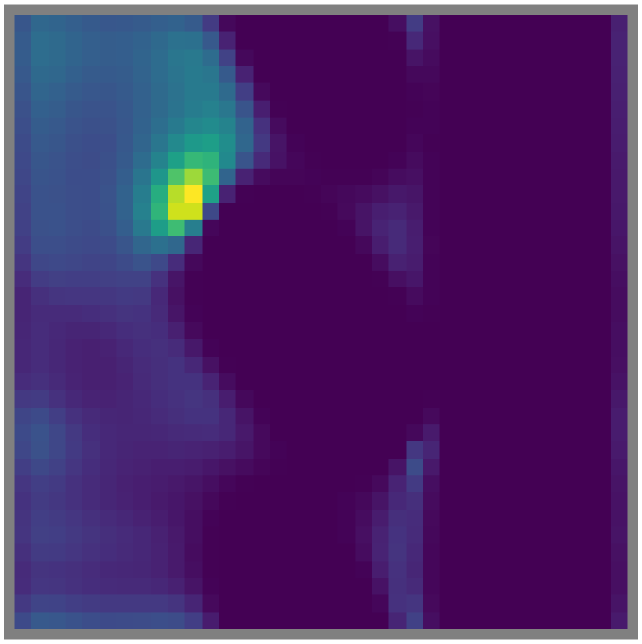
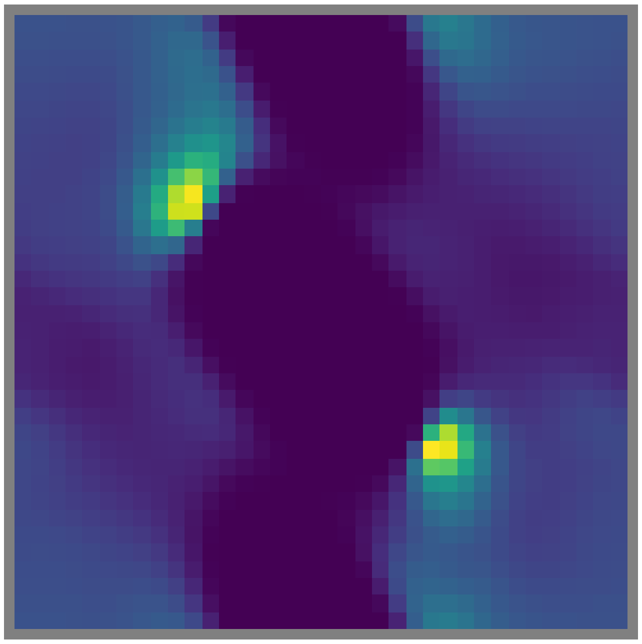
Plots:

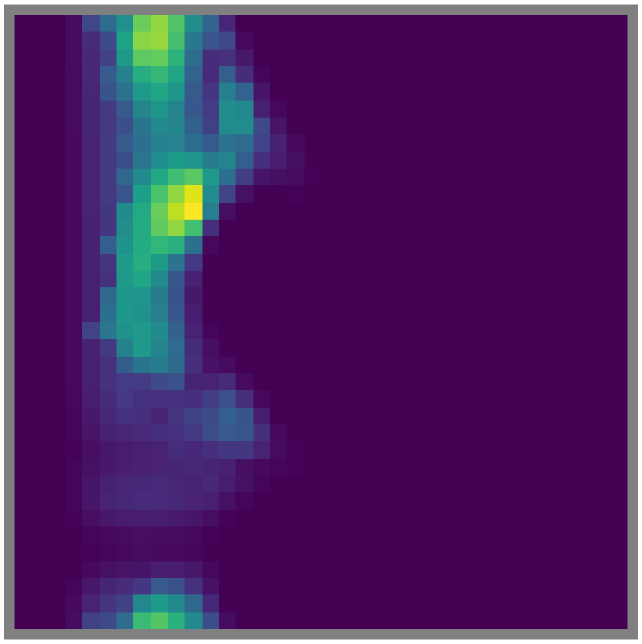
energy map “A”



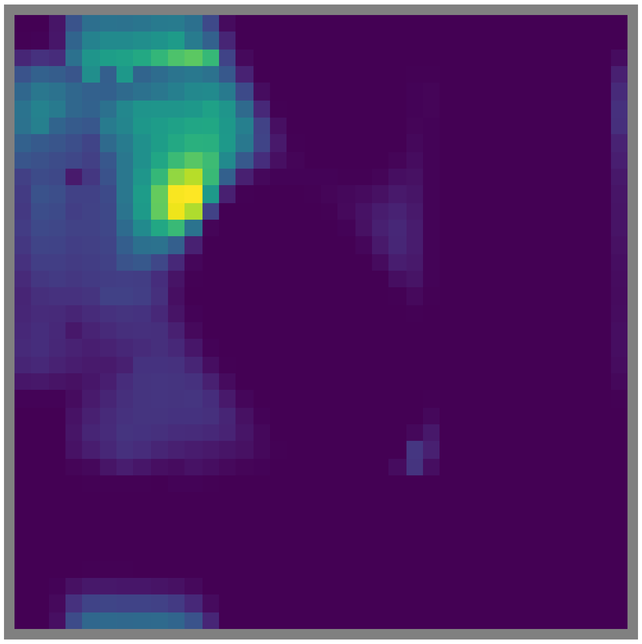
energy map “G”



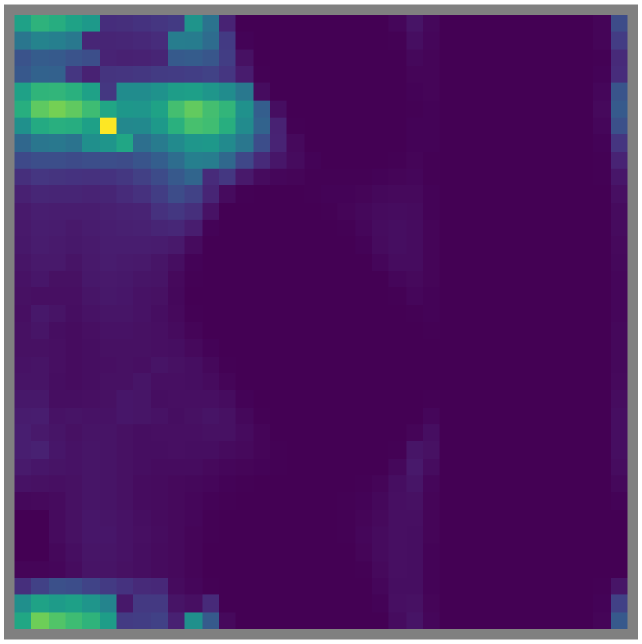
energy map “P”



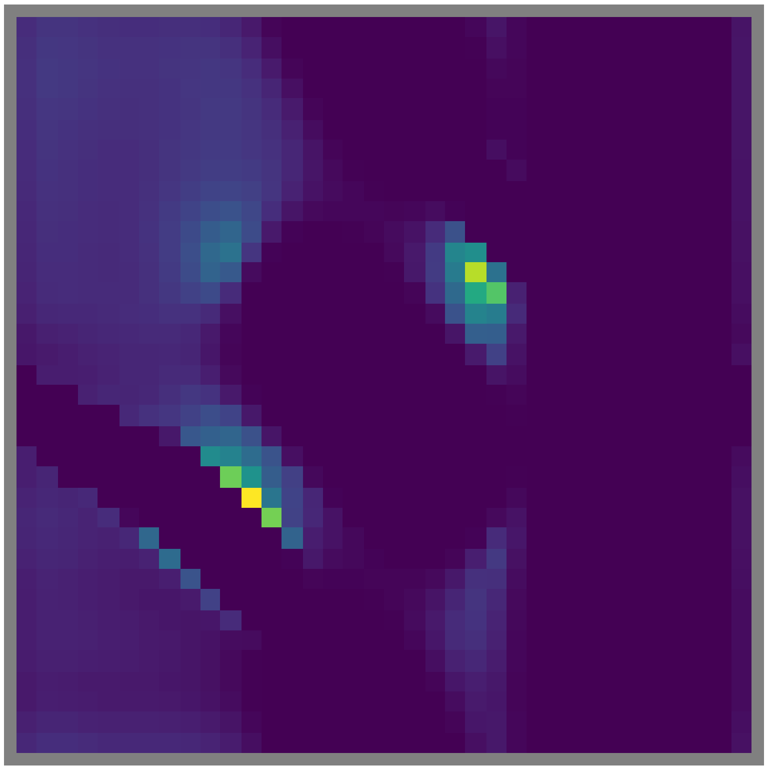
energy map “I”



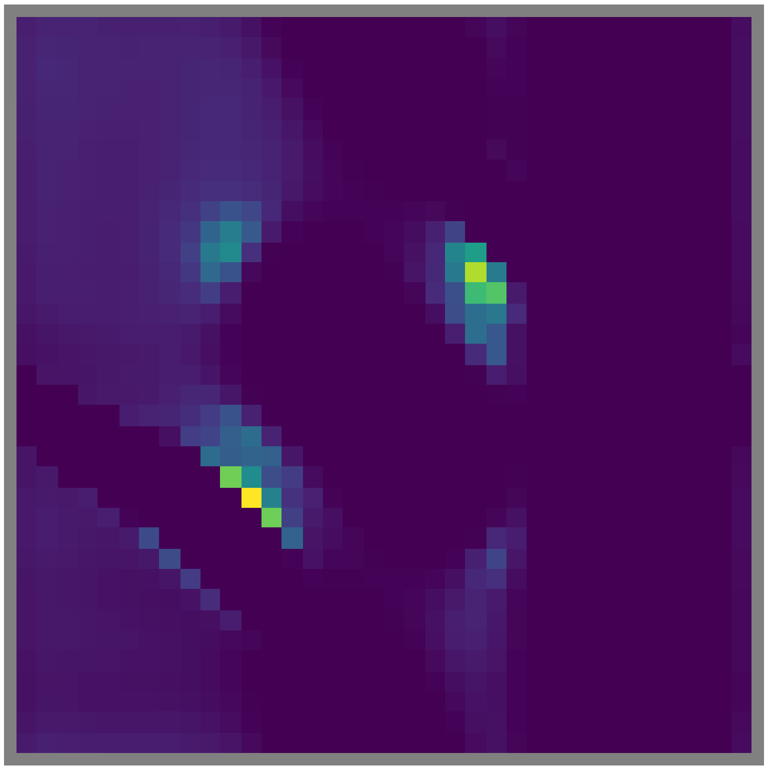
energy map “K”



alanine 9-mer without hydrogen bonding



alanine 9-mer with hydrogen bonding



Questions:

This plot is very similar to the "general case" Ramachandran plot. This is weird because the Ramachandran plot is based on the distribution of angles observed in a large number of different protein structures, while this plot is based on the energy landscape of a very small sequence of alanines. Maybe alanine, being a small and relatively simple amino acid, has energetically allowed regions in the Ramachandran plot that are similar to those of other amino acids. In fact, alanine is often found in energetically favorable regions of the Ramachandran plot.

The plot is slightly different than before. The areas are wider, especially on the right side. I think this is because glycine has more conformational flexibility than other amino acids due to its small side chain. This is very similar to the Ramachandran plot of glycine.

The proline graph is very weird. There is no left-handed helical region in the plot. I think this is because proline has a very unique conformational preference due to its rigid pyrrolidine ring structure. The ring restricts the backbone flexibility of proline, which stops it from forming a hydrogen bond in the backbone and limits its ability to have certain phi and psi angles, making angles from the left-handed helical region not energetically allowed.

The isoleucine graph also differs slightly to the A and G graphs. Isoleucine has two chiral centers in its side chain. In proteins, isoleucine almost always adopts the L configuration at both chiral centers, seen in the very highlighted region in the graph, as this orientation can affect the conformational preferences of the backbone and influence the overall energy landscape of isoleucine in the Ramachandran plot.

I chose to analyze lysine. Lysine has a long, flexible side chain with an amine group at its end, which can form hydrogen bonds (although we did not include these in our scoring function yet) and electrostatic interactions with nearby residues. These interactions can affect the conformational preferences of the backbone and influence the overall energy landscape. However, the size and flexibility of the lysine side chain may also cause steric clashes with neighboring residues, restricting the range of allowed phi and psi angles. Additionally, the positive charge of the amine group can also affect the local electrostatic environment and influence the conformational preferences of nearby residues.

The plot of the energies of a 9-mer of alanines all with the same set of phi psi torsion angles is pretty similar to the "AAA" case in part 2, except that some of the active regions in the first plot are less active here. The larger number of alanine residues in the 9-mer makes more chances for steric clashes or other unfavorable interactions that restrict the range of allowed phi and psi angles.

The alpha helix from the Ramachandran plot is clearly noticeable in this plot. And for the most part, I do seem to find low-energy states at these repeats, although there are some minor differences. This could be due to many different factors (ie. steric hindrance or interactions with neighboring residues) that would influence the energetics of these repeats.

The observed low- and high-energy points in the plot do make physical sense. The lowest-energy point with a phi of -60° and a psi of 50° indicates that the 9-mer is in a stable conformation characterized by a perfect spiral. On the other hand, the highest-energy point with a phi of 120° and a psi of 40° has overlapping atoms, resulting in an energetically unfavorable circular conformation. The simulation could be improved by using a more accurate energy function that better captures the underlying physics and chemistry of the system. Also, more extensive sampling of the conformational space could give a more comprehensive picture of the energy landscape.

Enabling hydrogen bonding terms in the energy function does slightly change the analysis. The regions of the Ramachandran plot that are energetically favored are still about the same, but there is slightly more emphasis on regions that promote hydrogen bonding, such as the alpha-helical region. The structures with the largest relative change in energies have significant hydrogen bonding interactions since they are alpha-helices (which are stabilized by hydrogen bonds).

Code:

```
import pyrosetta
pyrosetta.init('-mute all')


def generate_tripeptide(amino, phi, psi):

    # create a modified energy function
    scorefxn = pyrosetta.ScoreFunction()
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_atr, 1.0)
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_rep, 1.0)
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_elec, 1.0)

    # generate a tripeptide of the sequence ‘AXA’ where X may be any one of the twenty amino
acids
    sequence = f"A{amino}A"
    pose = pyrosetta.pose_from_sequence(sequence)

    # set the phi and psi angles of the center amino acid to the input values
    pose.set_phi(2, phi)
    pose.set_psi(2, psi)

    # set the phi/psi values of the first and third residues to “extended,” phi=-120 and psi=120
    pose.set_phi(1, -120)
    pose.set_psi(1, 120)
    pose.set_phi(3, -120)
    pose.set_psi(3, 120)

    # “repack” the center residue, finding the sidechain conformation that optimizes total energy
    task_pack = pyrosetta.standard_packer_task(pose)
    task_pack.restrict_to_repacking()
    task_pack.temporarily_fix_everything()
    task_pack.temporarily_set_pack_residue(2, True)
    pack_mover =
pyrosetta.rosetta.protocols.minimization_packing.PackRotamersMover(scorefxn, task_pack)
    pack_mover.apply(pose)

    # evaluate the Rosetta energy of the tripeptide
    energy = scorefxn(pose)
    return energy

generate_tripeptide("W", -60, -50)


import numpy as np

def get_energies_tripeptide(amino):
    energy_map = np.zeros(shape=(36, 36))
    for phi in range(36):
        for psi in range(36):
            energy_map[phi][psi] = generate_tripeptide(amino, (phi-17)*10, (psi-17)*10)
    return energy_map

energies = get_energies_tripeptide("A")


import matplotlib.pyplot as plt

def plot_energies(energy_map, temperature=1, ax='off'):

    beta = 1 / temperature
    partition_function = np.sum(np.exp(-beta * energy_map))
    boltzmann_probabilities = np.exp(-beta * energy_map) / partition_function

    fig = plt.figure(figsize=(7.5, 7.5))
    fig.set_facecolor('gray')
    if ax=='off':
        plt.axis('off')
    plt.imshow(np.flip(boltzmann_probabilities.T, axis=0), cmap='viridis')
    plt.show()

plot_energies(energies)


energies = get_energies_tripeptide("G")
plot_energies(energies)
```

```
energies = get_energies_tripeptide("P")
plot_energies(energies, temperature=1.5)


energies = get_energies_tripeptide("I")
plot_energies(energies)


energies = get_energies_tripeptide("K")
plot_energies(energies)


OUT_PATH = "/Users/tgoel/Downloads/outpath.pdb"

def generate_9mer(phi, psi, output_file=None, hb=False):

    # create a modified energy function
    scorefxn = pyrosetta.ScoreFunction()
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_atr, 1.0)
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_rep, 1.0)
    scorefxn.set_weight(pyrosetta.rosetta.core.scoring.fa_elec, 1.0)

    if hb:
        # take into account hydrogen bonding
        # hydrogen bonds are a major component in both helices and sheets
        scorefxn.set_weight(pyrosetta.rosetta.core.scoring.hbond_sr_bb, 1.0)
        scorefxn.set_weight(pyrosetta.rosetta.core.scoring.hbond_lr_bb, 1.0)

    # generate a 9-mer of alanine
    sequence = "A" * 9
    pose = pyrosetta.pose_from_sequence(sequence)

    # set the phi and psi angles of all amino acids to the input values
    for i in range(9):
        pose.set_phi(i+1, phi)
    for i in range(9):
        pose.set_psi(i+1, psi)

    # “repack” the sidechains of all residues, finding the sidechain conformations that optimize the
total energy
```

```
    task_pack = pyrosetta.standard_packer_task(pose)
    task_pack.restrict_to_repacking()
    task_pack.temporarily_fix_everything()
    for i in range(9):
        task_pack.temporarily_set_pack_residue(i+1, True)
    pack_mover =
pyrosetta.rosetta.protocols.minimization_packing.PackRotamersMover(scorefxn, task_pack)
    pack_mover.apply(pose)

    # evaluate the Rosetta energy of the 9-mer
    energy = scorefxn(pose)

    # write the output to a PDB file if requested
    if output_file:
        pose.dump_pdb(output_file)

    return energy


def get_energies_9mer(hb=False):
    energy_map = np.zeros(shape=(36, 36))
    for phi in range(36):
        for psi in range(36):
            energy_map[phi][psi] = generate_9mer((phi-17)*10, (psi-17)*10, hb=hb)
    return energy_map

energies = get_energies_9mer()
plot_energies(energies, temperature=10)


# lowest energy
x, y = np.unravel_index(np.argmin(energies), energies.shape)
generate_9mer((x-17)*10, (y-17)*10, OUT_PATH)


import nglview as nv

nv.show_file(OUT_PATH)
```

```
# highest energy
x, y = np.unravel_index(np.argmax(energies), energies.shape)
generate_9mer((x-17)*10, (y-17)*10, OUT_PATH)


nv.show_file(OUT_PATH)


energies_hb = get_energies_9mer(hb=True)
plot_energies(energies_hb, temperature=10)


# largest relative change in energies
diff = np.abs(energies_hb - energies)
x, y = np.unravel_index(diff.argmax(), diff.shape)

generate_9mer((x-17)*10, (y-17)*10, OUT_PATH)
nv.show_file(OUT_PATH)


for idx in np.argsort(diff.ravel())[::-1][:10]:
    x, y = idx // diff.shape[1], idx % diff.shape[1]
    generate_9mer((x-17)*10, (y-17)*10, OUT_PATH)
    display(nv.show_file(OUT_PATH))
# all alpha helices
```