CSCI-GA.3033-111: Protein Design HW1

Christine Wu (email:cw4459@nyu.edu)

Oct 6 2024

1 Learn how to download protein data from PDB. Become familiar with common protein tools.

1.1 Look up Trypsin on Uniprot, explore its function. Extract its PDB ID and explore on PDB.

Solution: According to the web information from UniProt Trypsin Entry, the functionality of Trypsin can be considered as:

- Catalytic activity: preferential cleavage at Arg | Xaa, Lys | Xaa, meaning the enzyme cleaves peptide bonds on the C-terminal side of arginine (Arg) or lysine (Lys), regardless of the following amino acid (Xaa).
- Cofactor: the enzyme requires Ca^{2+} as a cofactor, binding one calcium ion per subunit. This is supported by databases like UniProtKB, Rhea, and CHEBI (CHEBI:29108).
- Activity regulation: it is inhibited by scorpion cyclotide trypsin inhibitor TopI1.
- Features: the enzyme has several active and binding sites. The active sites at positions 63, 107, and 200 are involved in the charge relay system, essential for catalysis. It binds Ca^{2+} at positions 75, 77, 80, and 85, while positions 194-195, 197-198, and 200 are responsible for substrate binding.
- Cellular Component: it is located in the extracellular space and part of the serine protease inhibitor complex.
- Molecular Function: it exhibits serine-type endopeptidase activity, endopeptidase activity, metal ion binding, and serpin family protein binding.
- Biological Process: it is involved in digestion and proteolysis.
- Keywords: Hydrolase, protease, serine protease, calcium binding, metal-binding.

Besides the general functionality of Trypsin, we also learned that Trypsin is secreted as an inactive precursor, trypsinogen, by the pancreas. It is activated in the small intestine by enteropeptidase, which cleaves trypsinogen to form active trypsin. Active trypsin also plays a role in activating other digestive enzymes, such as chymotrypsinogen and procarboxypeptidase, driving a cascade of enzyme activations that contribute to protein digestion. It also uses a serine residue in its active site to perform hydrolysis of peptide bonds. This mechanism is characteristic of serine proteases, etc.

For its PDB ID, we can move to the structure section of UniProt Trypsin Entry, and we want to choose a structure with a high resolution (preferably below 2.5 Å, as higher-resolution structures are typically more accurate, as well as a common form. Hence we picked **2PTN**.

1.2 Using PDB's website built-in-search, download two sets of similar proteins. One found using a sequence similarity search, and one using a structure similarity search.

Solution: Now, we're using PDB 2PTN to do the sequence and structure searches.

- **Sequence similarity search:** For its sequence similarity, we found a set of PDB IDS that can be found from here: 2PTN sequence similarity search. This includes **1AQ7**, **1AUJ**, **1AZ8**, etc.
- **Structure similarity search:** For its structure similarity, we found a set of PDB IDS from here: 2PTN structure similarity search. This includes **3PTB**, **5MOR**, **1TNL**, etc.

1.3 Explore how the sequence search set's structures look with pymol.

Solution: Here, we choose the sequences of **2PTN**, **1AQ7**, and **1AUJ** to do the comparisons (Note: I'm using PyMol open source for this assignment).

1.3.1 Align structures on top of each other

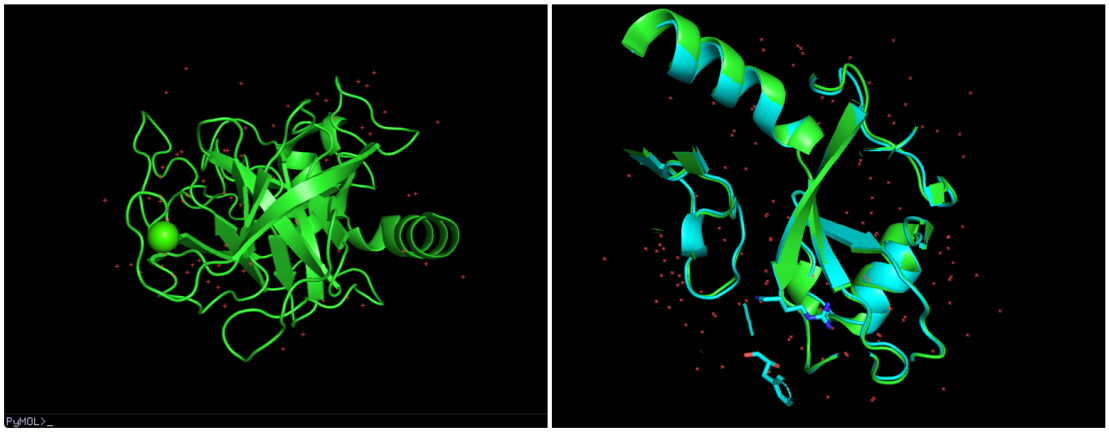


Figure 1: left: 2PTN; right: 2PTN and 1AQ7 aligning on top of each other, where 2PTN is the green one and 1AQ7 is the blue one.

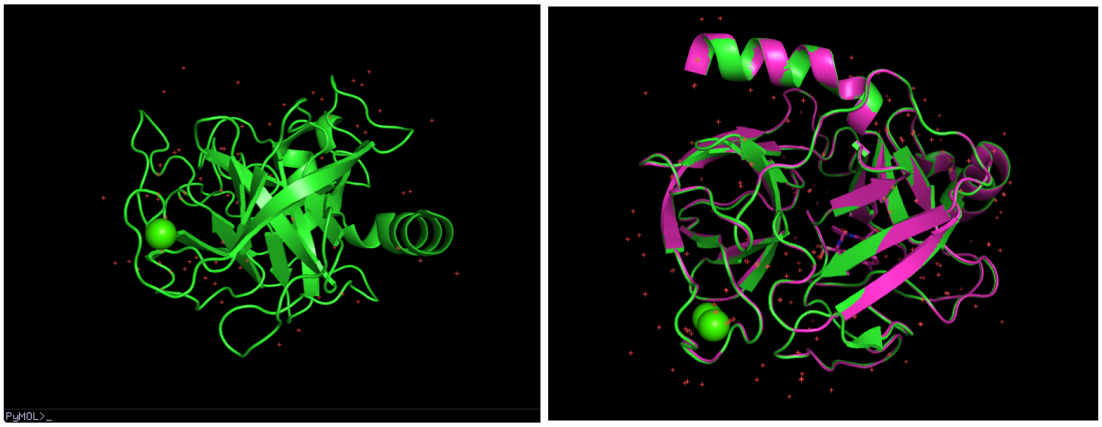


Figure 2: left: 2PTN; right: 2PTN and 1AUJ aligning on top of each other, where 2PTN is the green one and 1AUJ is the pink one.

1.3.2 Try different visualization methods (surface, cartoon, etc.)

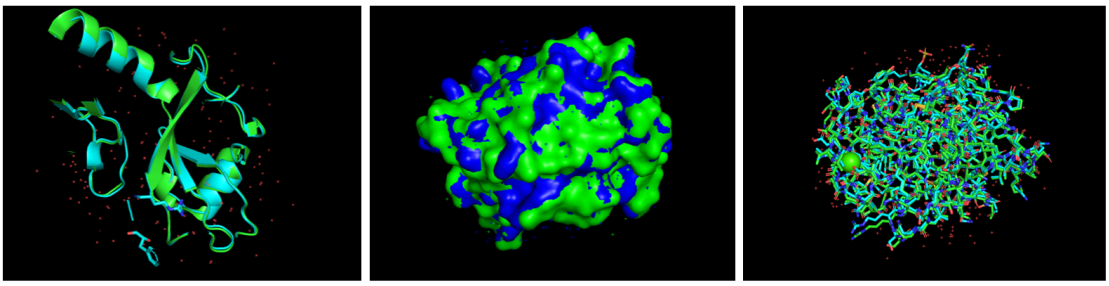


Figure 3: 2PTN and 1AQ7 aligning on top of each other. left: carton method; middle: surface method; right: stick method

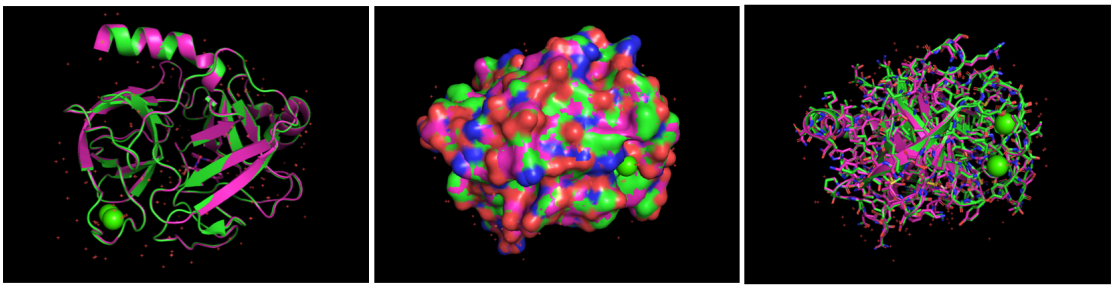


Figure 4: 2PTN and 1AUJ aligning on top of each other. left: carton method; middle: surface method; right: stick method

1.4 Using Blast and Align, create an MSA for Trypsin. Try to include proteins from at least 10 different organisms in the MSA.

Solutions: From the **Blast**, we included at least 10 different organisms as the following (with top 10 organisms selected):

BLAST 250 results found in UniProtKB

Overview | Taxonomy | Hit Distribution | Text Output | Input Parameters | API Request

Tools | Download | Add | Customize columns | Resubmit

Entry	Entry Name	Protein Names	Gene Names	Organism	Length
P00760	TRY1_BOVIN	**Serine protease 1** [...]	**PRSS1**, TRP1, TRY1, TRYP1	Bos taurus (Bovine)	246 AA
A0A4W2FD21	A0A4W2FD21_BOBOX	Cationic trypsin	**LOC113890875**	Bos indicus x Bos taurus (Hybrid cattle)	246 AA
A0A6P5BKV6	A0A6P5BKV6_BOSIN	Cationic trypsin	**LOC109557710**	Bos indicus (Zebu)	246 AA
A0A452EML0	A0A452EML0_CAPHI	Cationic trypsin	**LOC102178719**, LOC102179184	Capra hircus (Goat)	246 AA
A0A6P3IFR1	A0A6P3IFR1_BISBB	Cationic trypsin isoform X1	**LOC104999401**	Bison bison bison (North American plains bison)	246 AA
A0A8B9W8U3	A0A8B9W8U3_BOSMU	**Peptidase S1 domain-containing protein**		Bos mutus grunniens (Wild yak) (Bos grunniens)	246 AA
A0A452FA14	A0A452FA14_CAPHI	**Peptidase S1 domain-containing protein**	**LOC102179184**	Capra hircus (Goat)	246 AA
A0A6P3IN88	A0A6P3IN88_BISBB	Cationic trypsin isoform X2	**LOC104999401**	Bison bison bison (North American plains bison)	246 AA
W5Q1Y9	W5Q1Y9_SHEEP	**Peptidase S1 domain-containing protein**	**LOC101112049**	Ovis aries (Sheep)	246 AA
A0A5N3XT99	A0A5N3XT99_MUNRE	**Peptidase S1 domain-containing protein**	**FD755_012394**	Muntiacus reevesi (Reeves' muntjac) (Cervus reevesi)	246 AA

Figure 5: BLAST Result

For the **Align** result, we have the following:

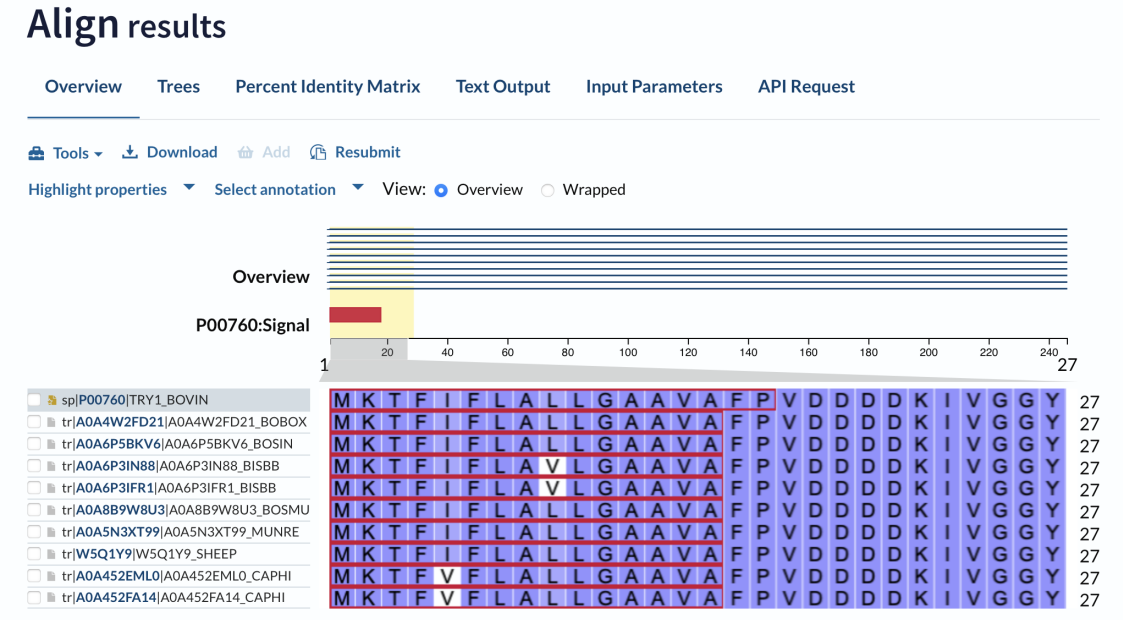


Figure 6: The overview of the aligned result for MSA

(**Note:** the more detailed version of the wrapped result and the output MSA are attached at the end of the document).

1.5 Use the MSA to identify adjacent conserved residues within Trypsin. Using PyMol select these resides and use them to align to the other proteins you obtained in B).

Solution: The adjacent conserved residues we will be using are: **Positions 1-4:** MKTF, **Positions 37-46:** GGYTCGANT, **Positions 65-72:** CGGSLINS, and **Positions 222-229:** KPGVYTKV. Then, the results of using PyMol are the following:

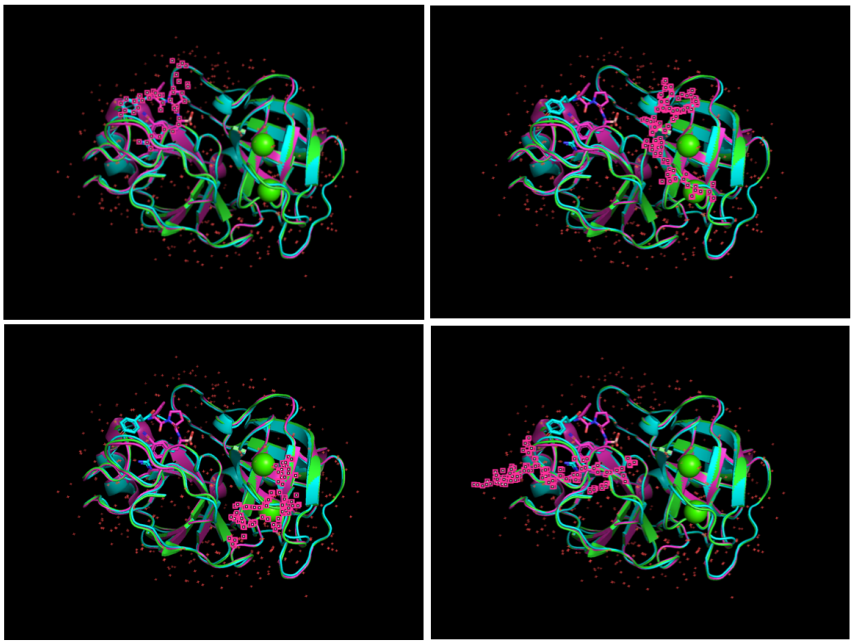


Figure 7: Sequence similarity proteins: green-2PTN, blue-1AQ7, pink-1AUJ, with adjacent conserved residues **Positions 1-4:** MKTF, **Positions 37-46:** GGYTCGANT, **Positions 65-72:** CGGSLINS, and **Positions 222-229:** KPGVYTKV (highlighted in red dots).

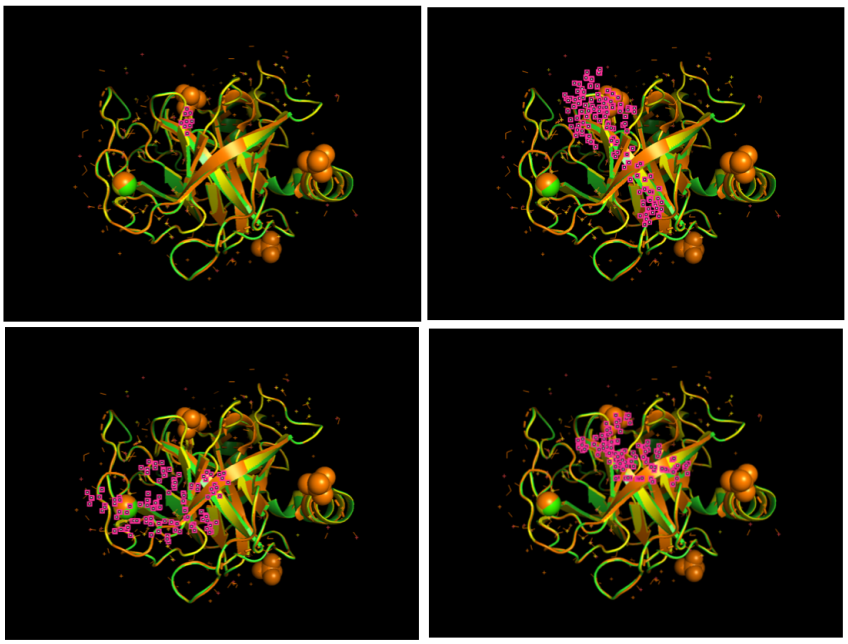


Figure 8: Structure similarity proteins: green-2PTN, yellow-3PTB, orange-5MOR, with adjacent conserved residues **Positions 1-4:** MKTF, **Positions 37-46:** GGYTCGANT, **Positions 65-72:** CGGSLINS, and **Positions 222-229:** KPGVYTKV (highlighted in red dots).

1.6 In Python, using the examples obtained in B), implement two distance functions: one that is based on the proteins' sequence, and another that relies on its structure. You may use an existing library or create this yourself, depending on the level of challenge you want to attempt.

Solution: Here, we used the built-in libraries to implement the functions:

a) The distance function that calculates based on protein's sequences:

```python
import os
import warnings
from Bio import PDB
from Bio import pairwise2
from Bio.pairwise2 import format_alignment
from prody import *
import time

# Suppress PDB warnings generically
warnings.filterwarnings("ignore", message="WARNING:.*discontinuous.*")

#------------------------------------------------------------------------------
def get_pdb_id(pdb_file):
    """
    Extracts and returns the PDB ID (filename without the extension) from a PDB file path.
    """
    return os.path.splitext(os.path.basename(pdb_file))[0]

#------------------------------------------------------------------------------
def extract_sequence_from_pdb(pdb_file):
    """
    Extracts the sequence of the first chain from the given PDB file using PPBuilder.
    """
    parser = PDB.PDBParser()
    structure = parser.get_structure("protein", pdb_file)

    # Use the PPBuilder to extract the sequence
    ppb = PDB.PPBuilder()

    # Get the sequence from the first polypeptide found in the first model
    for pp in ppb.build_peptides(structure):
        sequence = pp.get_sequence()
        return str(sequence)

#------------------------------------------------------------------------------
def sequence_distance(pdb1, pdb2):
    """
    Calculates sequence distance (based on percentage of non-identical residues)
    between two proteins based on their PDB files.
    """
    seq1 = extract_sequence_from_pdb(pdb1)
    seq2 = extract_sequence_from_pdb(pdb2)

    # Use global pairwise alignment (Needleman-Wunsch) to align sequences
    alignments = pairwise2.align.globalxx(seq1, seq2)

    # Get the alignment with the highest score (optimal alignment)
    best_alignment = alignments[0]

    # Calculate the percentage of identical residues in the alignment
    identical_residues = sum(1 for a, b in zip(best_alignment[0], best_alignment[1]) if a == b)
    alignment_length = len(best_alignment[0])

    # Calculate percentage identity
    percentage_identity = (identical_residues / alignment_length) * 100
    return 100 - percentage_identity
```

b) The distance function that calculates based on protein's structures:

```python
#-------------------------------------------------------------------------------
def calculate_rmsd(pdb1, pdb2):
    """
    Calculates the RMSD (Root Mean Square Deviation) between two PDB files based on their backbone
        alignment.
    """
    # Parse the PDB files
    structure1 = parsePDB(pdb1)
    structure2 = parsePDB(pdb2)

    # Select backbone atoms for alignment
    backbone1 = structure1.select('backbone')
    backbone2 = structure2.select('backbone')

    if backbone1 is None or backbone2 is None:
        print(f"Error: Backbone atoms not found in {pdb1} or {pdb2}")
        return None

    # Check if both backbones have the same number of atoms
    if backbone1.numAtoms() != backbone2.numAtoms():
        print(f"Skipping RMSD calculation for {pdb1} and {pdb2} due to mismatched atom counts.")
        return None

    # Superimpose the two structures
    transformation = calcTransformation(backbone2, backbone1)
    transformation.apply(backbone2)

    # Calculate RMSD
    rmsd_value = calcRMSD(backbone1, backbone2)
    return rmsd_value
```

1.7 Benchmark both the sequence-similar set and the structure-similar set with both criterions. Compare their respective values. Compare them to a set of randomly chosen proteins from PDB. Write a short conclusion on why those values make sense.

Solution: We first have the benchmark function as the following:

```python
#-------------------------------------------------------------------------------
def benchmark_proteins(pdb_set_1, pdb_set_2):
    """
    Benchmarks protein pairs for both sequence distance and structure distance (RMSD).
    """
    for pdb1 in pdb_set_1:
        for pdb2 in pdb_set_2:
            start_time = time.time()

            try:
                seq_distance = sequence_distance(pdb1, pdb2)
                struct_distance = calculate_rmsd(pdb1, pdb2)

                pdb_id1 = get_pdb_id(pdb1)
                pdb_id2 = get_pdb_id(pdb2)

                print(f"Comparing {pdb_id1} and {pdb_id2}:")
                print(f" Sequence distance: {seq_distance}")

                if struct_distance is not None:
                    print(f" Structure distance (RMSD): {struct_distance}")
                else:
                    print(f" Structure distance (RMSD): Skipped due to mismatched atom counts or missing
                        backbone.")

            except Exception as e:
```

```
                print(f"Error comparing {pdb1} and {pdb2}: {e}")

            finally:
                print(f"Time taken: {time.time() - start_time:.2f} seconds\n")


#--------------------------------------------------------------------------------
# load the files:
sequence_similar_set = [
    "2ptn.pdb",
    "1aq7.pdb",
    "1auj.pdb"
]

structure_similar_set = [
    "2ptn.pdb",
    "3ptb.pdb",
    "5mor.pdb"
]

random_pdb_ids = [
    "1xyz.pdb",
    "4hhb.pdb",
    "3def.pdb"
]

#--------------------------------------------------------------------------------
print("------------------- START --------------------")
# Perform the benchmark for sequence-similar, structure-similar, and random sets
print("Benchmarking Sequence-Similar Set:")
benchmark_proteins(sequence_similar_set, sequence_similar_set)
print("-----------------------------------------------")

print("\nBenchmarking Structure-Similar Set:")
benchmark_proteins(structure_similar_set, structure_similar_set)
print("-----------------------------------------------")

print("\nBenchmarking Random Proteins Set:")
benchmark_proteins(random_pdb_ids, random_pdb_ids)
print("------------------- END ----------------------")
```

Now, we have the outputs as:

```
------------------- START --------------------
Benchmarking Sequence-Similar Set:
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 2ptn and 2ptn:
  Sequence distance: 0.0
  Structure distance (RMSD): 4.976419911094366e-15
Time taken: 0.08 seconds

@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 2ptn and 1aq7:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.4472085649964476
Time taken: 0.09 seconds

@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 2ptn and 1auj:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.18055592175961002
Time taken: 0.07 seconds

@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
```

```
Comparing 1aq7 and 2ptn:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.44720856499644845
Time taken: 0.07 seconds

@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 1aq7 and 1aq7:
  Sequence distance: 0.0
  Structure distance (RMSD): 7.235314718896078e-15
Time taken: 0.08 seconds

@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 1aq7 and 1auj:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.450915874320609
Time taken: 0.10 seconds

@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 1auj and 2ptn:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.18055592175960897
Time taken: 0.07 seconds

@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1886 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 1auj and 1aq7:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.4509158743206105
Time taken: 0.07 seconds

@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1802 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 1auj and 1auj:
  Sequence distance: 0.0
  Structure distance (RMSD): 7.535431622541843e-15
Time taken: 0.07 seconds

----------------------------------------------

Benchmarking Structure-Similar Set:
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 2ptn and 2ptn:
  Sequence distance: 0.0
  Structure distance (RMSD): 4.976419911094366e-15
Time taken: 0.10 seconds

@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 2ptn and 3ptb:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.10683739877958595
Time taken: 0.07 seconds

@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
Comparing 2ptn and 5mor:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.13068966615174452
Time taken: 0.11 seconds

@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 3ptb and 2ptn:
  Sequence distance: 0.0
```

```
  Structure distance (RMSD): 0.10683739877958587
Time taken: 0.07 seconds

@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 3ptb and 3ptb:
  Sequence distance: 0.0
  Structure distance (RMSD): 1.0887956813044752e-14
Time taken: 0.10 seconds

@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
Comparing 3ptb and 5mor:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.12969343118978552
Time taken: 0.11 seconds

@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
@> 1712 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 5mor and 2ptn:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.13068966615174468
Time taken: 0.11 seconds

@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
@> 1701 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 5mor and 3ptb:
  Sequence distance: 0.0
  Structure distance (RMSD): 0.12969343118978627
Time taken: 0.14 seconds

@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
@> 3367 atoms and 1 coordinate set(s) were parsed in 0.04s.
Comparing 5mor and 5mor:
  Sequence distance: 0.0
  Structure distance (RMSD): 5.6429828945950354e-14
Time taken: 0.15 seconds

----------------------------------------------

Benchmarking Random Proteins Set:
@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
Comparing 1xyz and 1xyz:
  Sequence distance: 0.0
  Structure distance (RMSD): 1.947462900819195e-14
Time taken: 0.24 seconds

@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
Skipping RMSD calculation for 1xyz.pdb and 4hhb.pdb due to mismatched atom counts.
Comparing 1xyz and 4hhb:
  Sequence distance: 82.39795918367346
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.23 seconds

@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
Skipping RMSD calculation for 1xyz.pdb and 3def.pdb due to mismatched atom counts.
Comparing 1xyz and 3def:
  Sequence distance: 87.05882352941177
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.15 seconds

@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
Skipping RMSD calculation for 4hhb.pdb and 1xyz.pdb due to mismatched atom counts.
Comparing 4hhb and 1xyz:
```

```
  Sequence distance: 82.39795918367346
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.23 seconds

@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
Comparing 4hhb and 4hhb:
  Sequence distance: 0.0
  Structure distance (RMSD): 6.146794224938476e-15
Time taken: 0.21 seconds

@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
Skipping RMSD calculation for 4hhb.pdb and 3def.pdb due to mismatched atom counts.
Comparing 4hhb and 3def:
  Sequence distance: 82.183908045977
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.14 seconds

@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 5621 atoms and 1 coordinate set(s) were parsed in 0.06s.
Skipping RMSD calculation for 3def.pdb and 1xyz.pdb due to mismatched atom counts.
Comparing 3def and 1xyz:
  Sequence distance: 87.05882352941177
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.18 seconds

@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 4779 atoms and 1 coordinate set(s) were parsed in 0.05s.
Skipping RMSD calculation for 3def.pdb and 4hhb.pdb due to mismatched atom counts.
Comparing 3def and 4hhb:
  Sequence distance: 82.183908045977
  Structure distance (RMSD): Skipped due to mismatched atom counts or missing backbone.
Time taken: 0.14 seconds

@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
@> 2128 atoms and 1 coordinate set(s) were parsed in 0.02s.
Comparing 3def and 3def:
  Sequence distance: 0.0
  Structure distance (RMSD): 6.093927363022499e-14
Time taken: 0.08 seconds

-------------------- END ----------------------
```

Summary of Benchmarking Results:
The benchmarking results of sequence-similar, structure-similar, and randomly selected proteins provide insights into the relationship between protein sequences and structures.

- **Sequence-Similar Set:** The sequence distance for all protein pairs was consistently 0.0, indicating nearly identical sequences. The RMSD values were small but varied slightly, with 0.447 Å for 2ptn vs. 1aq7 and 0.181 Å for 2ptn vs. 1auj, which is expected due to minor structural deviations. Comparisons of identical proteins (e.g., 2ptn vs. 2ptn) produced RMSD values close to zero, confirming no structural differences.

- **Structure-Similar Set:** Low RMSD values (0.1-0.13 Å) confirmed that these proteins share very similar structures. However, the sequence distance was also 0.0 for all pairs, suggesting that these proteins are also highly conserved in sequence, which may require re-evaluation to focus on structurally similar but sequence-divergent proteins.

- **Random Proteins Set:** The random proteins showed high sequence divergence (82-87 percent non-identity)as expected. Many RMSD calculations were skipped due to mismatched atom counts, reflecting the structural dissimilarity between the proteins. RMSD values were close to zero for comparisons of identical proteins, as anticipated.

1.8 Compute an intra-protein residue-wise distance map for Trypsin.

Solution: We have the map as below:

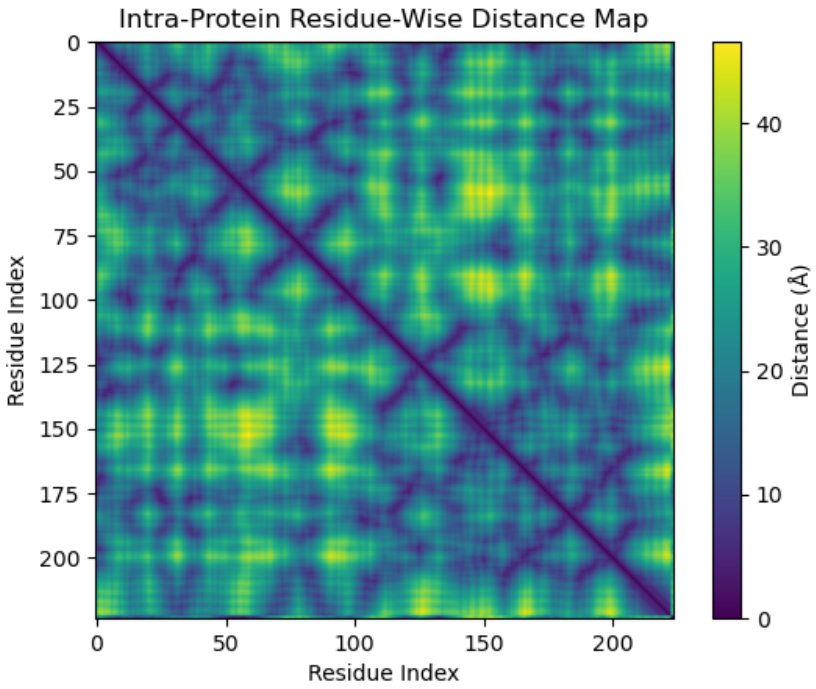


Figure 9: Intra-protein residue-wise distance map for Trypsin

The code is:

```
#-------------------------------------------------------------------------------
def compute_distance_map(pdb_file):
    """
    Computes the intra-protein residue-wise distance map for the given PDB file.
    The distances are calculated between the alpha carbon atoms of all residues.
    """
    parser = PDBParser()
    structure = parser.get_structure("protein", pdb_file)

    # Extract alpha carbon atoms
    ca_atoms = []
    for model in structure:
        for chain in model:
            for residue in chain:
                if 'CA' in residue: # Check if residue has an alpha carbon
                    ca_atoms.append(residue['CA'].get_coord())

    # Convert the list of atom coordinates into a NumPy array
    ca_coords = np.array(ca_atoms)

    # Initialize an empty distance matrix
    n = len(ca_coords)
    distance_map = np.zeros((n, n))

    # Compute pairwise distances
    for i in range(n):
        for j in range(i, n):
            distance = np.linalg.norm(ca_coords[i] - ca_coords[j])
```

```
            distance_map[i, j] = distance_map[j, i] = distance

    return distance_map

def plot_distance_map(distance_map, title="Intra-Protein Residue-Wise Distance Map"):
    """
    Plots the distance map using Matplotlib.
    """
    plt.imshow(distance_map, cmap='viridis')
    plt.colorbar(label='Distance ()')
    plt.title(title)
    plt.xlabel("Residue Index")
    plt.ylabel("Residue Index")
    plt.show()


pdb_file = "2ptn.pdb"
distance_map = compute_distance_map(pdb_file)
plot_distance_map(distance_map)
```

>sp|P00760|TRY1_BOVIN
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGTSYPDVLKCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGSGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A4W2FD21|A0A4W2FD21_BOBOX
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGTSYPDVLKCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGSGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A6P5BKV6|A0A6P5BKV6_BOSIN
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGTSYPDVLXCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGSGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A6P3IN88|A0A6P3IN88_BISBB
MKTFIFLAVLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVAEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGSNYPDVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A6P3IFR1|A0A6P3IFR1_BISBB
MKTFIFLAVLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVAEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGTSYPDVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A8B9W8U3|A0A8B9W8U3_BOSMU
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVAEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASISLPTSCASAGTQCLISGWGNTKSSGTSYPNVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIK
QTIASN
>tr|A0A5N3XT99|A0A5N3XT99_MUNRE
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNQNTLNNDIMLIKLKSAASLN
SRVASVSLPTSCASAGTQCLISGWGNTKSSGTSYPDVLQCLKAPILSDSSCSSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCNGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIQ
QTIASN
>tr|W5Q1Y9|W5Q1Y9_SHEEP
MKTFIFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSRWVVS
AAHCYKSGIQVRLGEDNINVAEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASVSLPTSCASAGTQCLISGWGNTKSSGTSYPDVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCNGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIQ
QTIASN
>tr|A0A452EML0|A0A452EML0_CAPHI
MKTFVFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASVSLPTSCASAGTQCLISGWGNTKSSGTSYPDVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIQ
QTIASN
>tr|A0A452FA14|A0A452FA14_CAPHI
MKTFVFLALLGAAVAFPVDDDDKIVGGYTCGANTVPYQVSLNSGYHFCGGSLINSQWVVS
AAHCYKSGIQVRLGEDNINVVEGNEQFISASKSIVHPSYNSNTLNNDIMLIKLKSAASLN
SRVASVSLPTSCASAGTQCLISGWGNTKSSGSNYPDVLQCLKAPILSDSSCKSAYPGQIT
SNMFCAGYLEGGKDSCQGDSGGPVVCSGKLQGIVSWGYGCAQKNKPGVYTKVCNYVSWIQ
QTIASN

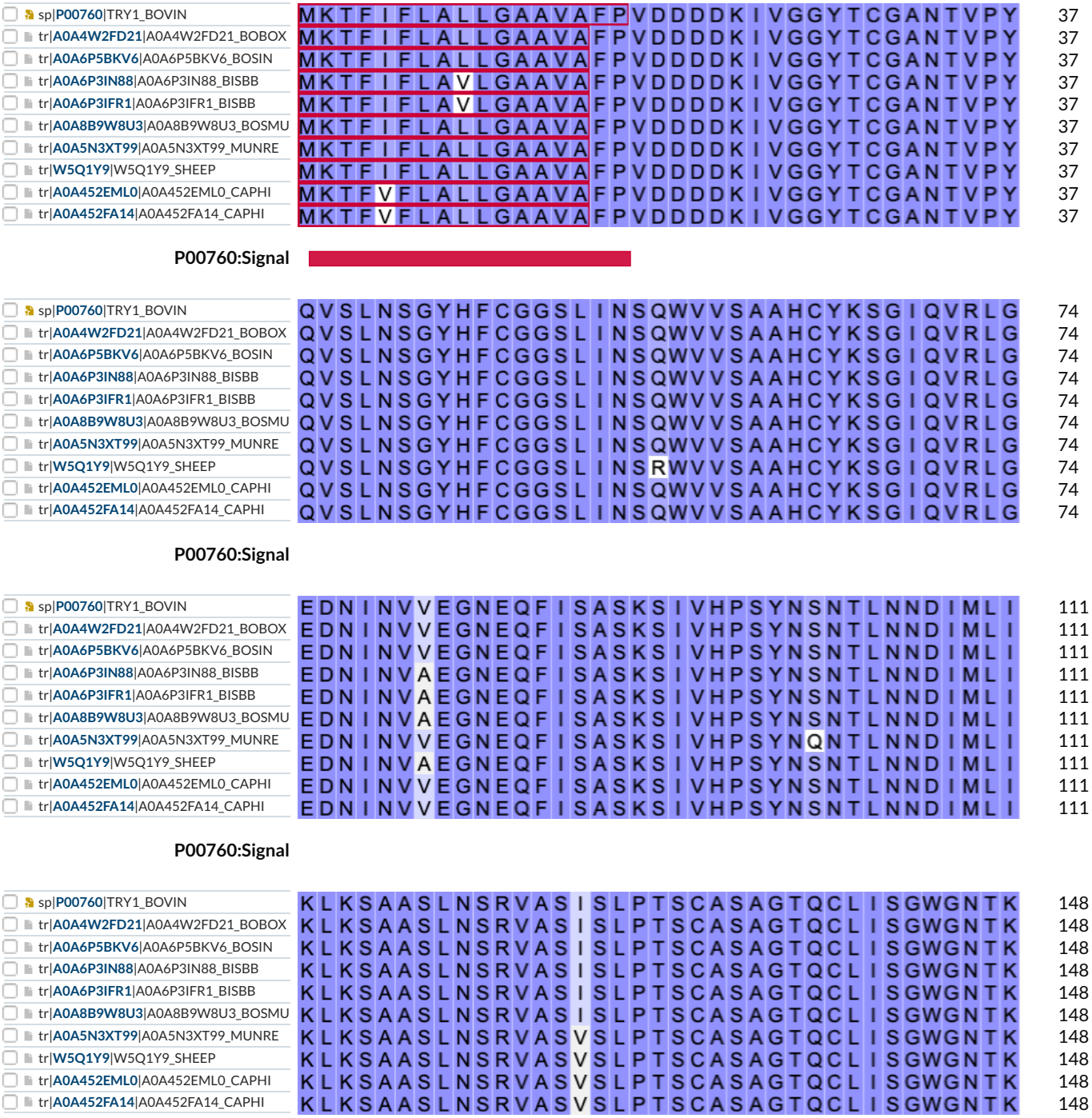
Tools Tool results Advanced | List Search Help

Align results

Overview Trees Percent Identity Matrix Text Output Input Parameters API Request

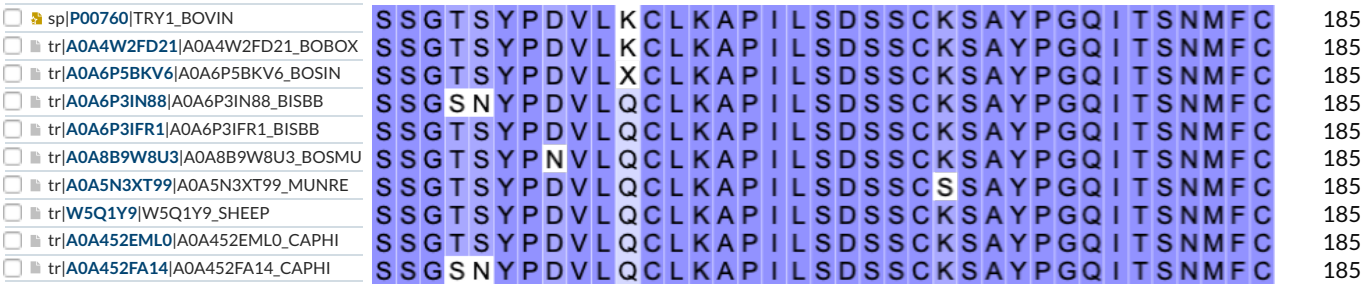
Tools Download Add Resubmit

Highlight properties Select annotation View: Overview Wrapped



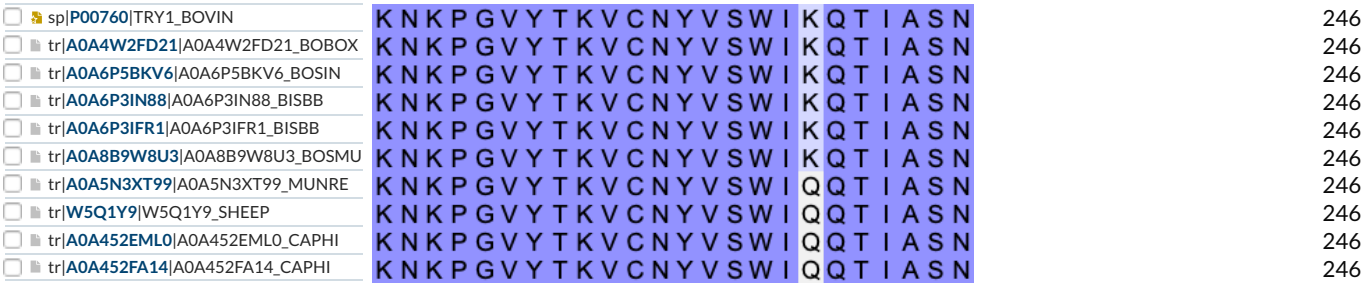
Feedback

P00760:Signal



P00760:Signal

P00760:Signal



P00760:Signal