**Protein Structure Analysis Training – Topics & Plan**

A screenshot of a computer

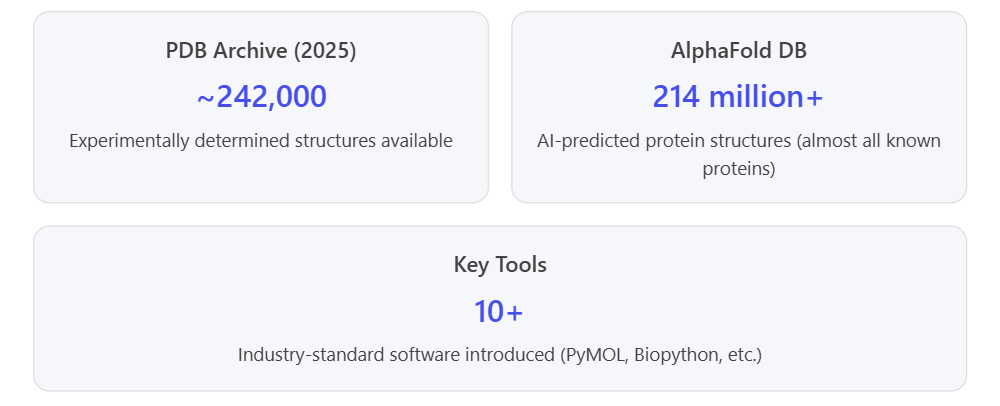
AI-generated content may be incorrect.

**Introduction**

Protein structure analysis is a cornerstone of bioinformatics, illuminating how a protein’s 3D shape determines its function and interactions. This training program will equip participants (beginner to intermediate) with both **theoretical knowledge and practical skills** in structural bioinformatics. **We will cover fundamental concepts** (like levels of protein structure and molecular interactions) **and practice using popular tools and workflows** to analyze protein structures in real-world scenarios. The focus includes working with PDB/mmCIF files, visualizing structures, and examining interactions such as protein–protein and protein–ligand binding. Each module pairs concepts with hands-on exercises, often via Python Jupyter notebooks for interactive learning. The goal is to enable trainees to not only understand the science but also perform common tasks like parsing structure files, identifying binding sites, and even predicting structures using AI tools like AlphaFold.

Importantly, this curriculum builds on existing knowledge resources. For example, an **internal Bioinformatics Workshop (July 2025)** collected training materials and emphasized using Jupyter notebooks for bioinformatics analyses[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d). While that workshop showcased data analysis (e.g. using pandas for RNA-seq data)[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d), the collaborative approach and use of notebooks will be mirrored here for structural topics. Furthermore, our organization’s Bioinformatics Community has held knowledge-sharing sessions on tools like AlphaFold (e.g. a recorded talk by Stefan Seemayer on deploying AlphaFold internally)[[2]](https://basf.sharepoint.com/teams/BioinformaticsCommunities/Shared%20Documents/Forms/DispForm.aspx?ID=95\&web=1). We will leverage such internal insights alongside external best practices to provide a comprehensive learning experience.

Below is a breakdown of **core training topics**, grouped by category (theory, tools, workflows, interaction types). Each topic is annotated with whether it lends itself to Python notebook demos or other practical components:



**Training Topics Overview**

| **Category** | **Topic** | **Python Notebook Suitability** |
| --- | --- | --- |
| **Theory** | **Protein Structure Basics** – Levels of structure (primary, secondary, tertiary, quaternary) and examples of each; how sequence determines 3D fold. | *Not code-driven* (conceptual background, diagrams). |
| **Theory** | **Molecular Interactions** – Types of non-covalent forces in structures (hydrogen bonds, hydrophobic effects, salt bridges, etc.) and their role in stability and binding. | *Partial*: Primarily conceptual, but can illustrate with a code demo (e.g. calculating distances to identify H-bonds). |
| **Theory** | **Structure Determination Methods** – Overview of how structures are obtained (X-ray crystallography, NMR, cryo-EM) and quality metrics (resolution, confidence scores). | *Not suited for notebooks* (informational). |
| **Theory** | **Protein–Protein vs Protein–Ligand Interfaces** – Characteristics of protein interaction interfaces vs small-molecule binding sites (surface area, specificity, typical bonding). | *Not code-driven* (analysis concepts, with examples). |

| **Tools** | **PDB and mmCIF Formats** – Understanding structure file contents. The mmCIF format (now primary in PDB) stores detailed data (e.g. unit cell, entities, sequences)[[3][3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). How to retrieve structures from the RCSB PDB database. | **Yes** ✅ (*Jupyter*): Use Biopython to read a .cif file and query its contents (e.g. list polymer chains, ligands)[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). | | **Tools** | **Biopython (Bio.PDB)** – Python library for parsing PDB/mmCIF files and manipulating structures. Loading a structure into a Structure object, iterating over atoms/residues, etc. | **Yes** ✅ (*Jupyter*): Hands-on parsing of a structure, extracting sequences or atom coordinates, etc.[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). | | **Tools** | **PyMOL** – Molecular visualization software to view 3D structures, create publication images, and analyze interactions. Features include displaying secondary structure, measuring distances, and finding contacts. | **Partial** ⚠️: PyMOL is external GUI, but we will demonstrate usage via prepared sessions or scripts. (E.g. run PyMOL to show a ligand binding site with H-bonds highlighted[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html).) Notebooks can include snapshots or use PyMOL’s API for automation. | | **Tools** | **Jupyter Visualization (py3Dmol/NGLview)** – Tools to embed interactive 3D views in notebooks. Enables rotation/zoom of structures in-browser. | **Yes** ✅: Use py3Dmol to load a PDB structure and allow trainees to explore it interactively in a notebook[[4]](https://www.blopig.com/blog/2024/06/interactive-visualization-of-protein-ligand-complexes-with-py3dmol/). | | **Tools** | **AlphaFold** – Deep-learning model for structure prediction. Using AlphaFold’s public database and Colab notebooks to obtain predicted structures for a given sequence. Discuss confidence metrics (pLDDT) and limitations. | **Yes** ✅: *Jupyter/Colab*: Show how to fetch a predicted structure from AlphaFold DB (via API or provided code). Trainees might run an AlphaFold Colab on a sample sequence (time permitting)[[5]](https://www.nature.com/articles/d41586-021-02050-3). | | **Tools** | **Other Key Tools** – Brief intro to other common structural bioinformatics tools: e.g. *ChimeraX* (advanced visualization), *LigPlot* (2D ligand interaction diagrams), *AutoDock Vina* (docking) – focusing on awareness. | **Partial**: Notebooks will not run these, but we mention their use. (E.g. show a LigPlot diagram of a protein–ligand complex to complement PyMOL view[[6]](https://learning.cloud.microsoft/detail/42c79580-7711-461e-b874-975d4ba32cda?context=%7b%22subEntityId%22:%7b%22source%22:%22M365Search%22%7d%7d).) |

| **Workflows** | **Structure Retrieval & Prep** – Workflow to go from a PDB ID to an analysis: downloading the file, examining contents, adding missing hydrogens, etc. | **Yes** ✅: Use Python (Biopython or requests) to fetch a structure by ID and parse it. Possibly call external services for adding hydrogens if needed. | | **Workflows** | **Basic 3D Visualization** – Viewing the protein’s overall fold and active sites. Changing representations (cartoon, surface, sticks) to highlight features. | **Yes** ✅: In notebooks, use an embedded viewer to toggle representations; offline, demonstrate via PyMOL for high-quality visuals. | | **Workflows** | **Analyzing a Binding Site** – Identify the ligand in a structure and the amino acids around it. Possibly visualize the pocket and list interactions. | **Yes** ✅: Write a notebook function to find all residues within X Å of a ligand. Use Biopython’s NeighborSearch to find contacts[[7]](https://education.molssi.org/python-scripting-biochemistry/chapters/biopython_mmcif.html), then visualize those residues in context. | | **Workflows** | **Protein–Protein Interface Analysis** – Given a complex (multichain structure), find which residues from each protein are at the interface. Calculate interface area or contacts. | **Yes** ✅: Use Python to iterate through atoms of two chains and find contacts within a distance cutoff. (Optionally introduce tools like PISA for interface area calculation – likely via demonstration). | | **Workflows** | **Sequence-to-Structure Modeling** – Demonstrate generating a model when no experimental structure exists (e.g. run a homology model or use AlphaFold). | **Yes** (guided): Show how to input a FASTA sequence into AlphaFold Colab or an API and obtain a model, then analyze that model’s quality. |

| **Interaction**  
**Types** | **Protein–Ligand Interactions** – Focus module on how proteins bind small molecules. Cover identifying key binding site residues, hydrogen bond donors/acceptors, hydrophobic pockets, and using tools to analyze ligand pose. | **Yes** ✅: Use an example PDB complex; notebook to automatically find and output ligand contacts (residues + distances). Visualization: generate an image with PyMOL showing ligand and nearby residues + H-bond dashed lines[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). | | **Interaction**  
**Types** | **Protein–Protein Interactions** – Study a protein complex (e.g. an enzyme with inhibitor protein or an antibody–antigen complex). Discuss interface properties (planar vs pocket, often large hydrophobic patches) and biologically relevant assemblies. | **Yes** ✅: Analyze an example dimer in a notebook – identify interface residues on both sides, and perhaps count intermolecular hydrogen bonds (with criteria). Possibly use a precomputed result from a tool (PISA or PDBe) to validate the manual approach. | | **Interaction**  
**Types** | **Nucleic Acid Interactions** (optional extension) – If relevant, cover proteins binding DNA/RNA (recognition motifs, binding-induced conformational changes). | **Yes** (if included): Demonstrate on a protein–DNA complex (e.g. a transcription factor bound to DNA). A notebook could list contacts between the protein and DNA (hydrogen bonds to bases, etc.). |

**Table: Training topics organized by category, with an indication of which ones will involve hands-on Python notebook exercises.** Topics marked “Yes” will have accompanying Jupyter notebooks or coding demos, whereas conceptual topics are delivered via lectures/slides.

**Theoretical Foundations**

**1. Protein Structure Basics:** We begin with a refresher on protein structural biology. **Proteins have hierarchical structure:**

* *Primary structure* – the amino acid sequence.
* *Secondary structure* – local motifs like α-helices and β-sheets stabilized by backbone hydrogen bonds.
* *Tertiary structure* – the full 3D fold of a single polypeptide chain.
* *Quaternary structure* – assembly of multiple protein subunits into a complex.

Trainees will learn how each level builds on the previous and how **amino acid properties affect folding** (e.g. hydrophobic residues buried inside, polar residues outside). For example, the difference between a **globular enzyme** vs a **fibrous protein** can be understood by their tertiary/quaternary arrangements. This theoretical module sets the stage for why we analyze structures at all – to link structure with function.

**2. Molecular Interactions in Structures:** Next, we cover the **non-covalent forces** that hold proteins in shape and allow them to bind other molecules. This includes:

* **Hydrogen bonds** – e.g. between backbone C=O and N–H in helices/sheets or between side chains and ligands.
* **Salt bridges (ionic interactions)** – e.g. between a glutamate (–COO⁻) and lysine (–NH₃⁺) side chain.
* **Hydrophobic interactions** – clustering of nonpolar residues away from water.
* **Van der Waals contacts** – how close packing of atoms stabilizes interfaces.

Understanding these interactions is crucial for analyzing binding sites and protein interfaces. We will illustrate examples (like how a drug forms hydrogen bonds with an enzyme’s active site). *This topic is largely conceptual*, but we will reinforce it later by actually identifying these interactions in example structures.

**3. Methods to Determine Protein Structures:** A brief overview of how experimental structures are obtained:

* **X-ray crystallography** – still the most common method in PDB. We’ll explain the notion of electron density and resolution (e.g. a 2.0 Å structure vs 4.0 Å – higher resolution means more detail).
* **NMR spectroscopy** – for smaller proteins in solution, yielding an ensemble of models.
* **Cryo-Electron Microscopy (cryo-EM)** – increasingly important for large complexes; discuss resolution revolution enabling near-atomic maps.

We won’t delve deeply into physics, but trainees should be aware of concepts like *resolution, R-factor*, and why some structures have missing loops (no clear density). We’ll also note that AlphaFold-predicted models come from computation, not experiment – and thus have **confidence scores instead of resolution**.

**4. Protein–Protein vs Protein–Ligand Interfaces:** We discuss the general differences between how proteins interact with other proteins versus small molecules:

* Protein–protein interfaces tend to be **larger and flatter** surfaces, often involving complementary shapes and multiple weak forces across a broad area (hundreds or thousands of square angstroms of contact area).
* Protein–ligand (small molecule) interfaces are typically **pockets or grooves** on the protein surface, where a small molecule fits with high specificity. Binding pockets often have a mix of polar groups positioned to H-bond with ligand polar atoms and hydrophobic regions to snugly fit the ligand’s nonpolar parts.

For example, in a hormone receptor dimer interface, one might see a broad hydrophobic patch and several salt bridges; whereas a kinase binding ATP has a well-defined nucleotide pocket with specific H-bonds for the adenine and phosphate groups. This conceptual understanding will guide the practical analysis: participants will know what to look for in each scenario (like key residues at a protein interface vs key pocket residues for ligand binding).

**Data Formats and Repositories**

**1. PDB and mmCIF File Formats:** The primary data source for structures is the Protein Data Bank (PDB). We cover the legacy **PDB format** (a columnar text format) and the now-standard **PDBx/mmCIF format**. The mmCIF (Macromolecular Crystallographic Information File) format is richer and more structured. It stores all details about the structure: atomic coordinates, but also metadata such as experimental conditions, biological source, and defined “entities” (proteins, ligands, ions, waters)[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html).

We will demonstrate reading an mmCIF file to show its content. For example, using Biopython, we can load an mmCIF and inspect data categories. In the case of PDB entry 1MBN (sperm whale myoglobin): the mmCIF lists three entities – one polymer (the myoglobin protein) and two non-polymers (a hydroxide ion and the heme prosthetic group)[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). This illustrates how **mmCIF explicitly catalogs each molecular component** of the structure, including ligands or cofactors. Trainees will learn to navigate these files, understanding key sections like \_atom\_site (atomic coordinates) and \_entity descriptors.

Additionally, we discuss how to **access the PDB database**:

* Using the RCSB PDB website for manual download (by PDB ID).
* Programmatically, via REST APIs or Python utilities. In a notebook, we’ll show how to fetch a structure by its ID (for instance, downloading the mmCIF for a given PDB code, which we will use in exercises).

By the end, users should know that around quarter-million structures are publicly available (the PDB reached ~242k entries in 2025[[7]](https://education.molssi.org/python-scripting-biochemistry/chapters/biopython_mmcif.html)), and how to get those data for analysis.

**2. Parsing Structures with Biopython:** We introduce Biopython’s Bio.PDB module as a key tool for handling structure files in Python. In the training, each participant will use a Jupyter notebook with Biopython to:

* Load an mmCIF/PDB file into a Structure object.
* Traverse the structure’s hierarchy: model → chain → residue → atom.
* Extract simple information (e.g., list all chains and their lengths, or print all ligands present).

For example, we will demonstrate reading a structure, then perhaps print out all residues of a ligand’s binding pocket by selecting atoms within a distance cutoff (an exercise using Biopython’s neighbor search functionality). This hands-on parsing is highly suitable for a notebook environment – the code is straightforward and immediate results can be seen (like how many chains, how many residues, etc. in a given structure)[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). By interacting with real data, trainees reinforce their understanding of file content and gain a skill (scripted data mining of structures) useful for research.

**3. Structure Visualization Tools: PyMOL and Friends:** Seeing the 3D structure is vital. **PyMOL** is a popular molecular graphics tool we’ll use for high-quality visualization. We plan to do live demos in PyMOL to show participants:

* How to load a structure and display it in different modes (cartoon for secondary structure, sticks for ligands, surfaces for pockets).
* How to color-code by property (e.g., highlight hydrophobic residues in a binding site).
* Measuring distances or angles (e.g., the distance between a ligand atom and a protein residue).
* Finding hydrogen bonds or contacts within PyMOL (Action → find → polar contacts can automatically show H-bonds for a selection).

As an example, we’ll generate an image of a protein–ligand complex highlighting key interactions: *the ligand and nearby residues are shown in sticks, and hydrogen bonds are displayed as dashed lines*[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). This visual encapsulation helps solidify the earlier theoretical points about binding. Trainees won’t necessarily run PyMOL via notebooks (since it’s a separate application), but they will learn to use it on their own machines. We will provide PyMOL session files or scripts (PML files) as needed so they can reproduce the views.

To integrate visualization into our interactive materials, we will also introduce **py3Dmol**, an open-source tool to embed 3D molecular viewers in Jupyter notebooks. This will allow participants to rotate and explore structures right within their browser[[4]](https://www.blopig.com/blog/2024/06/interactive-visualization-of-protein-ligand-complexes-with-py3dmol/). For instance, after parsing a PDB file with Biopython, we can feed it into py3Dmol to show the structure and let the user click-and-drag to inspect the binding pocket. This approach keeps the training self-contained and interactive, especially for remote or self-paced learners who can’t easily switch to an external PyMOL window.

**4. AlphaFold and Structure Prediction:** A modern protein analysis training is incomplete without discussing **AlphaFold**, the AI system that predicts protein structures from sequence. We’ll cover:

* What AlphaFold is and its significance (it solved the 50-year protein folding problem, achieving experimental-level accuracy in many cases).
* The AlphaFold Protein Structure Database: how in 2021–2022 it expanded from 350k to over 200 million predicted structures, essentially covering “almost every catalogued protein”[[5]](https://www.nature.com/articles/d41586-021-02050-3)[[8]](https://www.embl.org/news/science/alphafold-200-million/). (This incredible breadth means if trainees work with a protein that has no PDB entry, there’s a good chance an AlphaFold model is available.)
* How to interpret AlphaFold models: understanding the confidence score per residue (pLDDT) and that these are **predictions** (some regions may be unreliable, especially unstructured loops).

For the hands-on part, we plan a demonstration of retrieving an AlphaFold-predicted structure. For example, if a participant is interested in a human protein with no crystal structure, we can show how to go to the AlphaFold DB and download the model, or even use a small script to fetch it by UniProt ID. In a notebook, one could integrate with the AlphaFold database API to automate this. Additionally, we might walk through the use of the public Google Colab notebook for AlphaFold if time permits – so trainees learn the steps to run a prediction on a custom sequence (bearing in mind computational limits).

The inclusion of AlphaFold connects theory to cutting-edge practice. It highlights how computational methods supplement experimental data (a theme that might resonate, given our organization’s interest in AI for R\&D[[2]](https://basf.sharepoint.com/teams/BioinformaticsCommunities/Shared%20Documents/Forms/DispForm.aspx?ID=95\&web=1)). By the end, participants should feel comfortable obtaining and using AlphaFold models and be aware of their accuracy considerations.

**Practical Workflows and Case Studies**

This section of the training ties everything together in end-to-end workflows, each culminating in an analysis of real protein structure examples. All these workflows will be accompanied by **Python Jupyter notebooks or guided exercises**, ensuring participants actively apply what they’ve learned.

**1. Retrieving and Inspecting a Structure (PDB Data Pipeline):** We’ll start with a simple but important task: given a protein of interest, **retrieve its structure and examine it**. The workflow:

* **Find the structure:** Use the PDB ID or search by protein name on RCSB. (If multiple entries exist, choose a representative high-quality structure.)
* **Download the file:** In the notebook, we can include a snippet that uses Bio.PDB.PDBList or a direct URL fetch to get the mmCIF file. For example, retrieving PDB ID 1ABC and loading it into our environment.
* **Parse and summarize:** Using Biopython, parse the file and print a summary (e.g., “Structure 1ABC contains 2 chains: Chain A (enzyme, 250 aa) and Chain B (inhibitor peptide, 20 aa) plus 1 ligand (ATP)”). This step shows how to extract basic info programmatically.

By automating these steps, participants see how quickly they can get from a PDB code to having the data structures to work with in Python. It reinforces understanding of file content and provides a template they can reuse in their own work.

**2. Visualizing Overall Fold and Domains:** Once a structure is loaded, the next workflow is **visual analysis of the protein**:

* Use PyMOL or py3Dmol to display the protein’s overall fold. Emphasize identifying secondary structure elements (helices, sheets) and any domain organization. For instance, we might guide them to observe “this protein has two domains – an N-terminal β-barrel and a C-terminal helical domain”.
* Identify any obvious functional sites: e.g., a pocket with a bound ligand or a metal ion. We will show how to highlight that region (perhaps zooming in PyMOL or isolating the pocket in the viewer).

This part is largely about getting comfortable with looking at structures. We might have an exercise where participants use the viewer to answer a question like “Where do you think the active site might be?” or “Identify a Zn²⁺ ion and the residues coordinating it”. They can then verify with provided answers or by toggling on labeled views.

**3. Binding Site Analysis – Protein–Ligand:** We dedicate a significant practical module to analyzing a protein-ligand complex, since this encapsulates many skills: parsing, selecting, measuring, and interpreting interactions. We’ll use a concrete example (say, **HIV protease bound to a drug inhibitor** or **hemoglobin with heme and oxygen**).

Steps in this workflow:

* **Identify ligand and pocket:** In the notebook, find the ligand by scanning the structure for hetero-residues (non-standard residues). With Biopython, one can filter residues where .id[0] != " " (i.e., hetero-flag is set). Suppose the ligand is found to be “MK1” (just a placeholder name). We then collect residues of the protein within, say, 5 Å of any ligand atom using NeighborSearch[[7]](https://education.molssi.org/python-scripting-biochemistry/chapters/biopython_mmcif.html).
* **List interacting residues:** The code can output a list like: *“Ligand MK1 contacts 12 residues: Asp25, Gly27, Ile50, …”* each within 5 Å. We might further classify which are polar vs nonpolar or identify potential hydrogen bonds by checking donor-acceptor geometry (though full H-bond identification might require knowing hydrogen positions, which we could add or assume).
* **Visualization:** We will visualize the ligand in its pocket. Using PyMOL, for instance, we can demonstrate the script or commands to show the ligand and pocket residues, and draw dotted lines for hydrogen bonds. PyMOL’s dist command in mode=2 can find H-bonds if hydrogens are present or by simple distance criteria[[3]](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html). The outcome (which we might prepare as an image in slides or let them generate if they have PyMOL) is a clear picture of the ligand bound in the active site with key interactions labeled or highlighted. For example, showing that an Asp25 side chain is making a hydrogen bond to the ligand’s amine group.

This case study reinforces earlier content: trainees see *practically* which residues are important for binding and how those correspond to the interactions discussed in theory. Because they’ll perform the neighbor search themselves, it’s an active discovery (“Oh, these are the residues within 5 Å; I recognize some are polar – likely hydrogen bonding – and some are hydrophobic – likely forming a pocket shape”).

**4. Protein–Protein Interaction Analysis:** Another key workflow is examining a protein–protein complex. We might select an example such as an **antibody-antigen complex** or a **dimeric enzyme**. The aim is to show how to analyze interfaces:

* **Separating chains:** First, participants will separate the two chains or partners (the notebook can treat chain A vs chain B).
* **Contact analysis:** Using a similar approach as the ligand case, find all contacts between chain A and chain B (e.g., any atom of A within 5 Å of any atom of B). This yields a set of interface residues on each side.
* **Interface properties:** We will then discuss what we see: e.g., “Chain A and B bury ~1500 Å² of surface area upon binding” (we can provide this number from literature or use a tool like PDBe PISA to get it, since calculating surface area from scratch is complex). But from our contacts list we can infer, for instance, that several hydrophobic residues (maybe a Phe, Leu cluster) are in the interface core, and a couple of salt bridges (say between a Lys on A and an Asp on B) at the periphery.
* **Visualization:** We’ll show how to highlight the interface residues on the 3D structure (e.g., color them red on both subunits, or display just the interface region). This can be done in PyMOL by selecting those residues. The visual might be presented to emphasize how the two proteins fit together like puzzle pieces.

By performing this, trainees learn practical ways to identify which parts of a protein are involved in protein–protein recognition. This is also a good point to mention any known hotspots (like how certain residues, when mutated, disrupt the interaction – connecting to real-world applications like rational mutagenesis or protein engineering).

**5. From Sequence to Structure – Modeling Workflow:** To make the training holistic, we include a scenario where a protein has no known PDB structure. What does a researcher do? We introduce two approaches briefly:

* **Homology modeling** (mention tools like SWISS-Model or Modeller, without going in-depth).
* **AlphaFold-based prediction**, which we focus on as it's state-of-the-art.

The practical element here could be running an AlphaFold Colab notebook on a short protein sequence (or at least walking through the inputs/outputs because running a full AlphaFold might be time-consuming). Alternatively, we might retrieve a ready AlphaFold model to mimic this process (e.g., take a UniProt ID, go to AlphaFold DB, get the model).

Once a model is obtained, we treat it like an experimental structure: load it up, inspect confidence scores (AlphaFold .pdb files have b-factor column repurposed for confidence – we’ll show how low-confidence regions can be identified by color-coding the structure). We will also compare the predicted model to any partial experimental data if available, or at least discuss how one would validate it (Ramachandran plots, etc., though that might be beyond scope for beginners).

The goal is for participants to know there are avenues to get a structure even if none is experimentally known – empowering them to not stop their analysis due to lack of PDB entry. They also gain awareness of the limitations (e.g., “AlphaFold is great, but if your protein is disordered or has no homologs, the prediction might be unreliable in places”).

Throughout all these workflows, we will emphasize **reproducibility and practice**: the provided notebooks can serve as templates for the attendees’ future projects. For instance, the pocket analysis notebook could be adapted by them later to study any enzyme-inhibitor pair they are interested in.

Also, each workflow doubles as a case study reinforcing multiple topics: e.g., the ligand binding exercise brings together understanding file parsing, chemical interactions, and use of visualization tools; the protein interface exercise reinforces chain hierarchy, surfaces, and hydrophobic vs polar interactions concepts.

**Emphasis on Molecular Interactions**

Because the user especially mentioned interest in interaction types (protein–protein, protein–ligand, etc.), the training puts special focus on analyzing and interpreting these interactions:

**Protein–Ligand Interactions Module:** In this dedicated section, beyond the practical exercise described earlier, we teach general principles of how ligands bind:

* Common patterns: ligands often sit in pockets lined by complementary shapes; enzymes bind substrates often in a specific orientation, etc.
* We’ll highlight the concept of **key binding residues** vs others. Often a few residues contribute most of the binding energy (e.g., a bidentate salt bridge or a pair of hydrogen bonds might be critical). We can cite known examples, like the HIV protease inhibitors which rely on crucial Asp–ligand interactions.
* Tools: We briefly demonstrate **LigPlot** (a tool that generates 2D diagrams of protein-ligand interactions). For instance, for a given PDB, LigPlot will show which residues form H-bonds (with distances) and which ones make hydrophobic contacts with the ligand. We can show a LigPlot diagram to the class as another representation. (LigPlot was listed among the tools used in an advanced course[[6]](https://learning.cloud.microsoft/detail/42c79580-7711-461e-b874-975d4ba32cda?context=%7b%22subEntityId%22:%7b%22source%22:%22M365Search%22%7d%7d), confirming it as a standard utility in structural bioinformatics.)
* We also discuss water-mediated interactions if relevant (sometimes water molecules bridge protein and ligand – though that might be going a bit deep, it’s worth mentioning in passing).

By the end of this module, participants should be able to look at a protein-ligand structure and **identify the major interactions** – a skill useful for drug design, inhibitor optimization, etc.

**Protein–Protein Interactions Module:** Similarly, we delve into what makes protein interfaces special:

* The idea of *shape complementarity* – good interfaces have high shape and chemical complementarity.
* Often, interfaces have “hot spots”: e.g., a particular salt bridge or a stack of aromatic residues might contribute disproportionately. We can mention experimental findings (alanine scanning results from literature) to illustrate how not every contact is equal – some are critical.
* If time permits, we might introduce the PDBe PISA web tool, which can analyze a given PDB file and report interface area, solvation energy gain, etc. Even if participants don’t run it themselves, we can demonstrate a PISA report for an example complex to show how interfaces are quantified in pro terms. For example, “PISA calculates that the interface buries 1600 Å² and has 8 hydrogen bonds and 2 salt bridges – indicating a stable complex.”

Additionally, we can tie back to theory by discussing why interfaces are often rich in hydrophobic residues (the “hydrophobic effect” pushing nonpolars out of solvent, which stabilizes the complex) and have well-placed polar interactions to add specificity.

**Other Interaction Types:** While protein–protein and protein–ligand are the focus, we won’t ignore other biologically important interactions:

* **Protein–DNA/RNA:** If any trainees work in this area, we can discuss how DNA-binding proteins recognize specific sequences (via hydrogen bonds to bases in the major groove, etc.), and how one might analyze a protein–DNA complex similarly (identifying the DNA contact residues, which are often positively charged if binding DNA because DNA is negatively charged). We can showcase a structure like a transcription factor-DNA complex and highlight an arginine making specific contacts to guanine bases. The analysis approach (neighbor search, visualization) is analogous to protein–ligand.
* **Metal ion coordination:** Many proteins have metal cofactors (Zn in zinc fingers, Mg in enzymes, etc.). We mention how these are seen in structures (metal ions as separate “ligand” with usually several side chains around). PyMOL can easily show coordination distances. We might include a quick example (like calcium-binding in a calmodulin EF-hand motif, showing the coordinating residues). It’s a neat demonstration of geometry (often octahedral coordination, etc.).

By covering these scenarios, the training ensures breadth – attendees get a taste of analyzing all sorts of complexes they might encounter. The unifying skill is the ability to dissect a structure and understand who interacts with whom, and how.

**Tools and Platforms in Depth**

Here we compile notes on the specific tools introduced, ensuring trainees know how to use them and where to find more information or internal support:

* **PyMOL:** All participants will be encouraged to install PyMOL (an open-source version is available). We provide a quick-start guide (one page cheatsheet with common commands). During training, when we do the live PyMOL demo, we’ll also share that PyMOL session file. **Use cases in training:** viewing structures, highlighting residues, making pretty images. We note that PyMOL is scriptable (in fact, the .pml scripts we used can be modified – some advanced participants might try automating tasks). PyMOL is a staple in structural biology, so competency here is a takeaway goal.
* **Biopython & Scripting:** For many, this might be their first exposure to using Python for structural biology. We will ensure the notebooks are well-documented and beginner-friendly (with comments explaining each step). Since the training users are mix beginner/intermediate, some may not be expert programmers – but seeing how a 10-line Python script can carry out a complex analysis (like identifying an active site) can be illuminating. We’ll also point them to Biopython’s documentation and encourage experimentation (e.g., “try changing the distance cutoff and see how the contact list changes” as an exercise).
* **AlphaFold & Colab:** We will clarify that running AlphaFold locally requires heavy compute and isn’t trivial, which is why we demonstrate via the Colab (which uses Google’s resources). However, we’ll mention that internally, if there’s interest, one could use our HPC resources – in fact, our company has explored running AlphaFold on supercomputer clusters as noted by internal discussions[[2]](https://basf.sharepoint.com/teams/BioinformaticsCommunities/Shared%20Documents/Forms/DispForm.aspx?ID=95\&web=1). For training though, stick to using the freely available interfaces. Also mention related tools: e.g., **I-TASSER** for homology/modeling (which was listed in that external course alongside AlphaFold[[6]](https://learning.cloud.microsoft/detail/42c79580-7711-461e-b874-975d4ba32cda?context=%7b%22subEntityId%22:%7b%22source%22:%22M365Search%22%7d%7d)) as a classical method, though likely AlphaFold outperforms it now.
* **Visualization in Notebooks:** The introduction of py3Dmol and possibly **NGLview** (another Jupyter widget for molecules) is to empower participants to create shareable reports. We show that one can produce an interactive report of an analysis (for example, a notebook that not only lists interface residues but also includes a panel for viewing the interface). This might inspire them to create similar notebooks for presentations or publications. Technically, we’ll provide an environment file to install py3Dmol easily. The feedback from them seeing a protein spin in their browser is usually great – and it hammers home that modern bioinformatics is quite accessible.
* **Other Tools Awareness:** We briefly mention other tools without going deep:
  + **ChimeraX** – a newer visualization tool with powerful analysis capabilities and even some AlphaFold integration. Just so they know alternatives to PyMOL.
  + **MD Simulation Tools (e.g. GROMACS)** – since interactions can be dynamic, we mention that advanced analysis might involve running molecular dynamics. (Our external course reference shows MD simulation was taught alongside docking[[6]](https://learning.cloud.microsoft/detail/42c79580-7711-461e-b874-975d4ba32cda?context=%7b%22subEntityId%22:%7b%22source%22:%22M365Search%22%7d%7d), but for our scope we likely won’t do an MD demo; just note it exists if someone wants to simulate a protein-ligand complex stability).
  + **Docking Tools (AutoDock Vina)** – if time/interest, we might show how one would set up a simple docking experiment or at least explain the concept (predicting how a small molecule might fit into a protein when you don't have an experimental complex). We have internal interest in such predictive methods for ligand placement. However, doing a live docking might be too much; so this will remain informational.

By giving this overview, we ensure trainees have a map of the toolkit landscape. They’ll dive deep with PyMOL, Biopython, AlphaFold in practice, but also know what else is out there for future learning (e.g., if someone wants to pursue drug design, they know to explore docking and MD tools).

**Leveraging Internal Resources**

As part of the training design, we looked for **existing internal materials** that could enrich the program:

* The **July 2025 Bioinformatics Workshop** (led by [Kerr Wall](https://www.office.com/search?q=Kerr+Wall\&EntityRepresentationId=4aee818f-29fd-4c95-a458-584a6bf75ea9)) compiled numerous resources and even an email with links to materials for training purposes[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d). We will review those materials to reuse any relevant portions. For instance, if that workshop included general introductions to bioinformatics or data management tips (the notes mention using pandas for data analysis[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d) and setting up shared code workspaces[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d)), we can incorporate similar approaches here. The idea of a “shared workspace for code and data” is very applicable – we might set up a shared OneDrive or Teams folder where all training notebooks and data files are available to participants.
* Our organization’s **Bioinformatics Community Meetings** often touch on structural topics. For example, in August 2023 there was an AlphaFold session open to all of BASF[[2]](https://basf.sharepoint.com/teams/BioinformaticsCommunities/Shared%20Documents/Forms/DispForm.aspx?ID=95\&web=1). If the recording or slides from that session are accessible, we can provide them as supplementary material for interested attendees. This gives a real-world context of how BASF scientists are using AlphaFold on internal projects (e.g. Stefan Seemayer discussing pipeline integration).
* We also have internal experts: colleagues in the Protein Engineering team have developed pipelines (there’s mention of a “Protein Analysis Pipeline” using Snakemake on our HPC, which includes structure prediction and analysis[[9]](https://engage.cloud.microsoft/main/threads/eyJfdHlwZSI6IlRocmVhZCIsImlkIjoiMzI4MjczMDIwOTE5ODA4MSJ9)). We might invite one of them for a guest segment or at least cite their work as an example of enterprise-scale structural bioinformatics – demonstrating to trainees the relevance of what they’re learning to actual projects.

If any internal tutorial or documentation on tools like PyMOL or Biopython exists (perhaps in our internal knowledge bases), we’ll link those for further reading. The training will thus not be an isolated session but connect trainees with a network of resources and people in our company for continued learning.

**Conclusion**

By the end of this training, participants will have a **comprehensive foundation in basic protein structure analysis**. They will understand theoretical concepts (how proteins fold and interact) and have practical experience with:

* Reading and interpreting structure files (.cif/.pdb).
* Visualizing proteins and complexes to spot key structural features.
* Using Python to automate structure analysis – from simple tasks like listing ligand neighbors to more complex ones like mapping a protein–protein interface.
* Utilizing cutting-edge tools like AlphaFold to predict or obtain structures when needed.

We believe this combination of theory and hands-on practice is ideal for beginners to intermediates: it builds intuition and confidence. The inclusion of interactive Jupyter notebooks for many topics means attendees can **learn-by-doing**, tinkering with code and seeing immediate results, which research shows is effective for technical skill uptake. They’ll also have these notebooks as take-home **templates**.

Finally, tying into our organizational context, this training prepares scientists to leverage internal and external tools in their projects, and fosters an interest in structural bioinformatics that could lead to new ideas (for example, applying these analyses to design better catalysts or to understand protein targets in our pipeline). With the provided skills, attendees can, for instance, explain to their teams how a given protein’s structure underpins its function or how a prospective compound might bind – thereby elevating the scientific discussion and innovation in their respective roles.

All materials (slides, notebooks, example data, and references) will be shared on our internal SharePoint for future reference. We’ll also maintain a discussion channel (perhaps a Teams channel) for follow-up questions, as building a community of practice is one objective (like the one noted in the workshop follow-ups[[1]](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit\&mobileredirect=true\&wdorigin=Sharepoint\&DefaultItemOpen=1\&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d\&wd=target%28/Meetings.one/%29\&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d\&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d)).

In summary, this training module offers a well-rounded introduction to protein structural analysis, **grounded in theoretical knowledge and reinforced with practical skills**. Trainees will be equipped to tackle tasks ranging from simple structure browsing to conducting their own interaction analyses in Python, paving the way for more advanced explorations (like docking or molecular dynamics) should they choose to pursue them. The synergy of internal knowledge and external best practices will make it a rich learning journey.

**References**

[1] [Bioinf Workshop](https://basf-my.sharepoint.com/personal/wallke_basfad_basf_net/_layouts/15/Doc.aspx?action=edit&mobileredirect=true&wdorigin=Sharepoint&DefaultItemOpen=1&sourcedoc=%7bbe54a777-837c-4bdd-915f-99e96cb3f5aa%7d&wd=target(/Meetings.one/)&wdpartid=%7ba966a277-911a-481e-819b-99315f05dcec%7d%7b1%7d&wdsectionfileid=%7b834961bc-3d55-4bc6-ac25-0093ace038f1%7d)

[2] [AlphaFold2, Session 2 - Using the technologies at BASF - Priya Anand and Stefan Seemayer - CPEC-20210827\_021436](https://basf.sharepoint.com/teams/BioinformaticsCommunities/Shared%20Documents/Forms/DispForm.aspx?ID=95&web=1)

[3] [PyMOL tutorial: Generate ligand interaction images](https://research.uni-leipzig.de/straeter/pymol/pymol_tutorial_generate_ligand_interaction_images.html)

[4] [Interactive visualization of protein–ligand complexes with Py3Dmol ...](https://www.blopig.com/blog/2024/06/interactive-visualization-of-protein-ligand-complexes-with-py3dmol/)

[5] [Daily briefing: Inside the intricate glass skeleton of a deep ... - Nature](https://www.nature.com/articles/d41586-021-02050-3)

[6] [Bioinformatics; Learn Docking & Mol Dynamics Simulation](https://learning.cloud.microsoft/detail/42c79580-7711-461e-b874-975d4ba32cda?context=%7b%22subEntityId%22:%7b%22source%22:%22M365Search%22%7d%7d)

[7] [Analyzing MMCIF Files using Biopython - MolSSI Education](https://education.molssi.org/python-scripting-biochemistry/chapters/biopython_mmcif.html)

[8] [AlphaFold predicts structure of almost every catalogued protein known ...](https://www.embl.org/news/science/alphafold-200-million/)

[9] [Hello Community! We are happy to come back with our digest of projects and activities taking place in the Protein Engine](https://engage.cloud.microsoft/main/threads/eyJfdHlwZSI6IlRocmVhZCIsImlkIjoiMzI4MjczMDIwOTE5ODA4MSJ9)