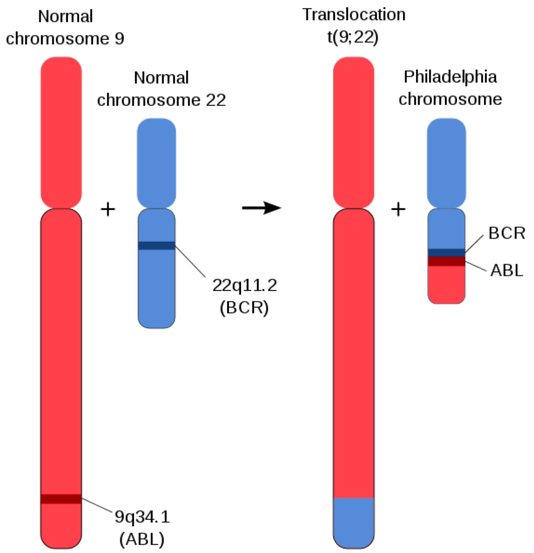
# Chapter 5. Biophysical Machine Learning

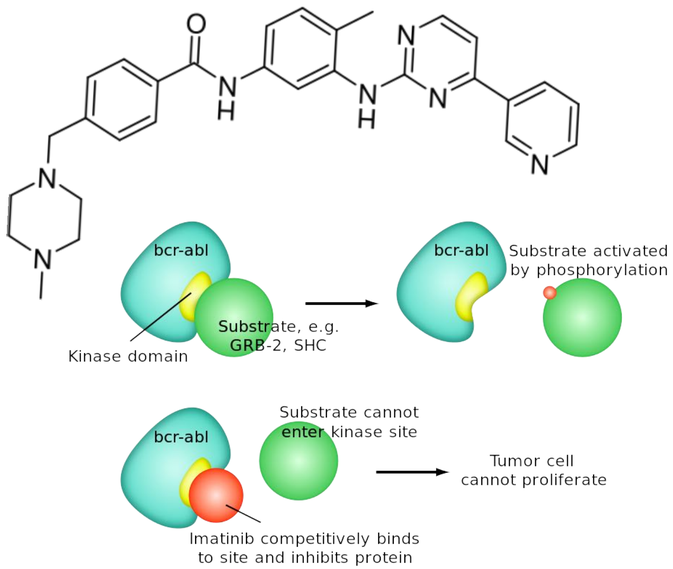
Junsu Ko
(2019-08-02)

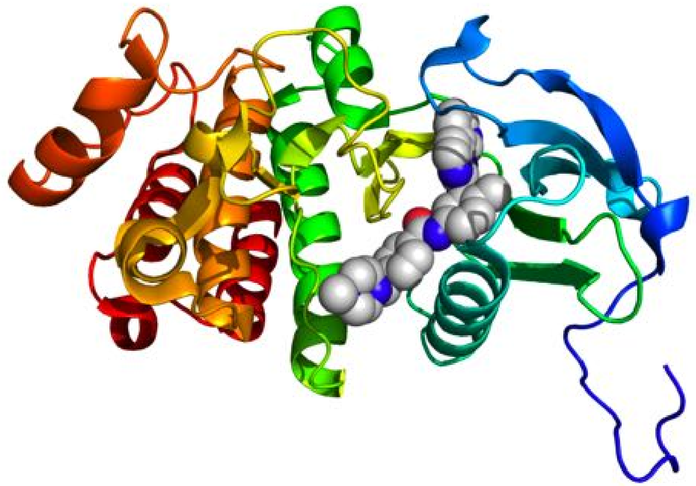
# 차례

# In this chapter

- Biophysical System을 이해하기 위해서 deep learning을 사용하는 방법
  - 저분자 약물 유사 물질(small drug-like molecules)이 대상 단백질에 결합하는지를 예측하는 방법
- 신약 개발에 있어서 아주 중요한 단계
  - The breakthrough cancer drug Imatinib tightly binds with BCR-ABL, for example, which is part of the reason for its efficacy.



https://en.wikipedia.org/wiki/Imatinib http://www.rcsb.org/structure/1IEP

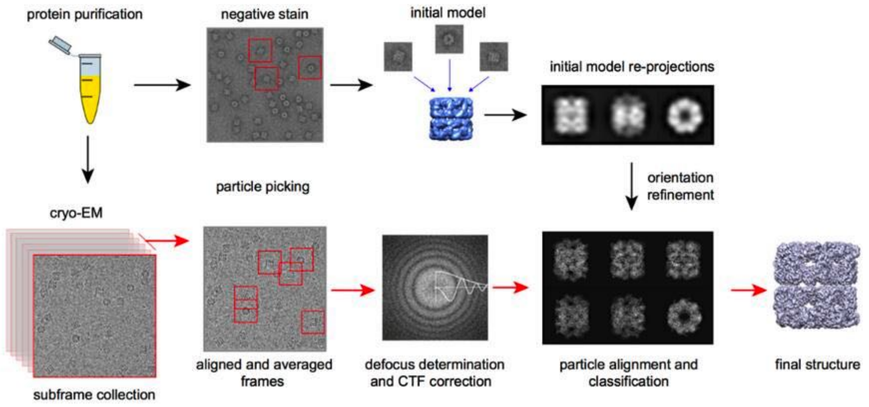


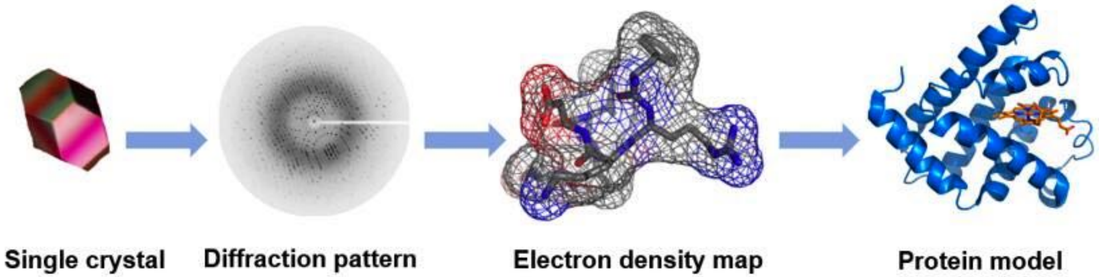
https://gtexportal.org/home/gene/ABL

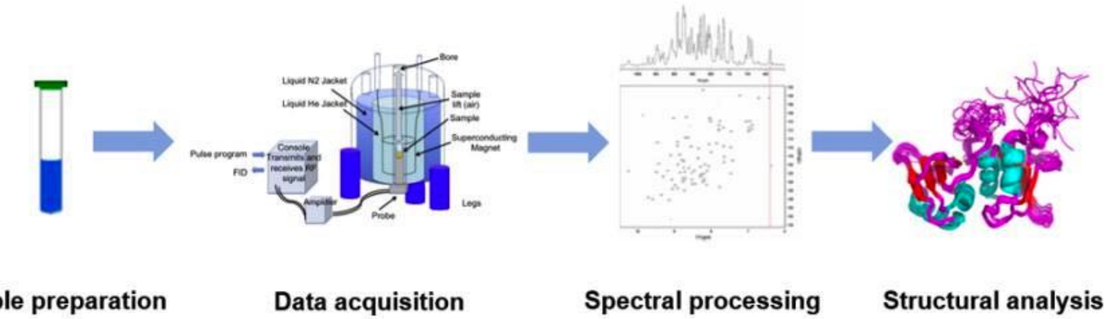
ARONTIER

# Protein Structures

- 단백질은 세포내에서 작동하는 가장 작은 기능의 단위이며, 수천개의 원자로 이루어져 있고, 아주 복잡한 과정을 거쳐서 작동을 함.
- 단백질이 다른 물질들과 어떻게 상호작용하는지 알아야 함.
  - 이를 위해서 3차원 구조를 알고 있어야 함.
- 구조를 확인하기 위한 실험적 방법
  - X-ray crystallography
  - nuclear magnetic resonance (NMR for short),
  - cryo-electron microscopy (cryo-EM for short).



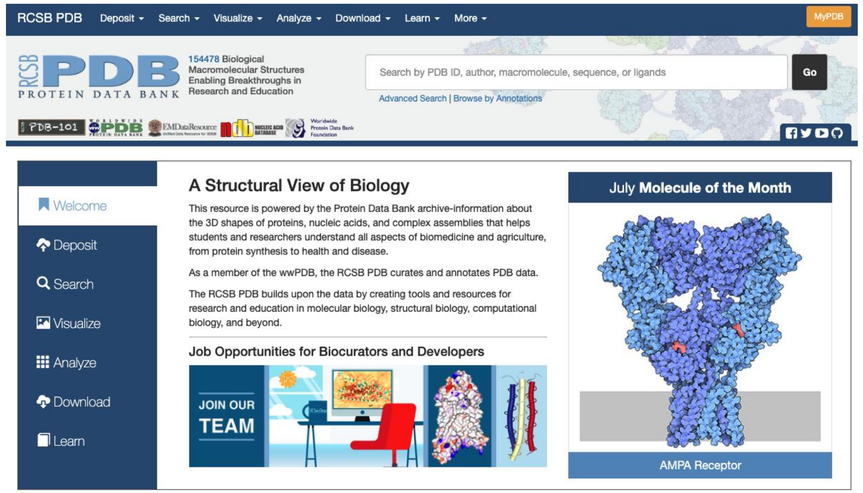




https://www.creative-biostructure.com/comparison-of-crystallography-nmr-and-em_6.htm

# PDB (Protein Data Bank)

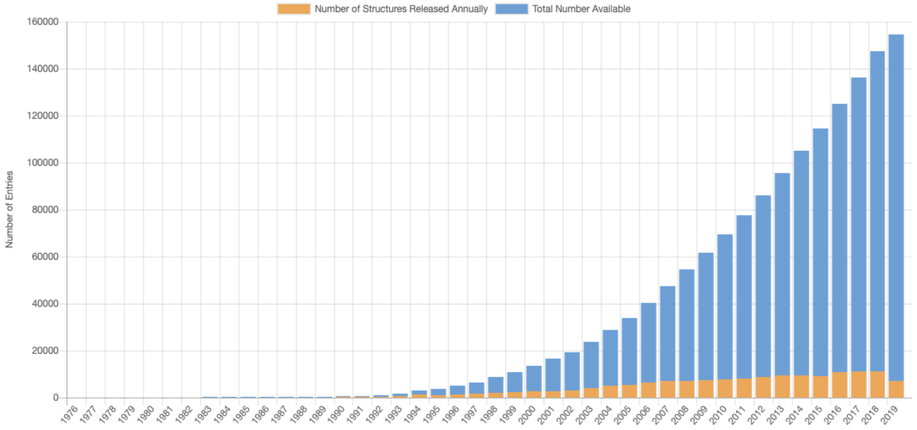
https://www.rcsb.org/



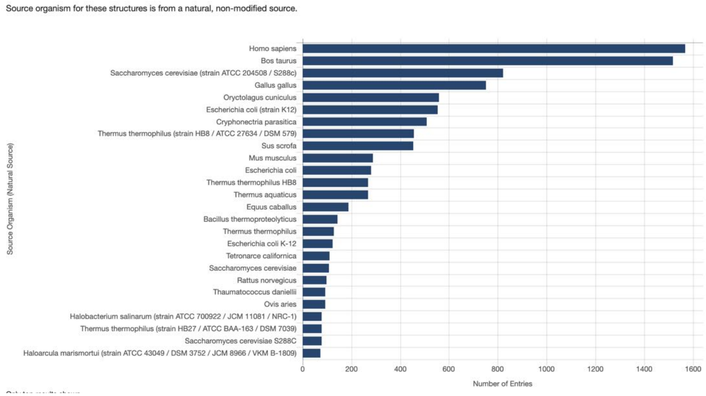

- PDB
  - 단백질 구조 저장소
  - 알려져있는 대부분의 단백질 구조가 등록되어 있음.
  - 154,478 Biological Macromolecular Structures

- Many proteins can exist in multiple functionally different states (for example, "active" and “inactive" states), so you want to know the structure of each state.
- The PDB is a fantastic resource, but the field as a whole is still in its "low data" stage

## 연도별 구조 증가 수

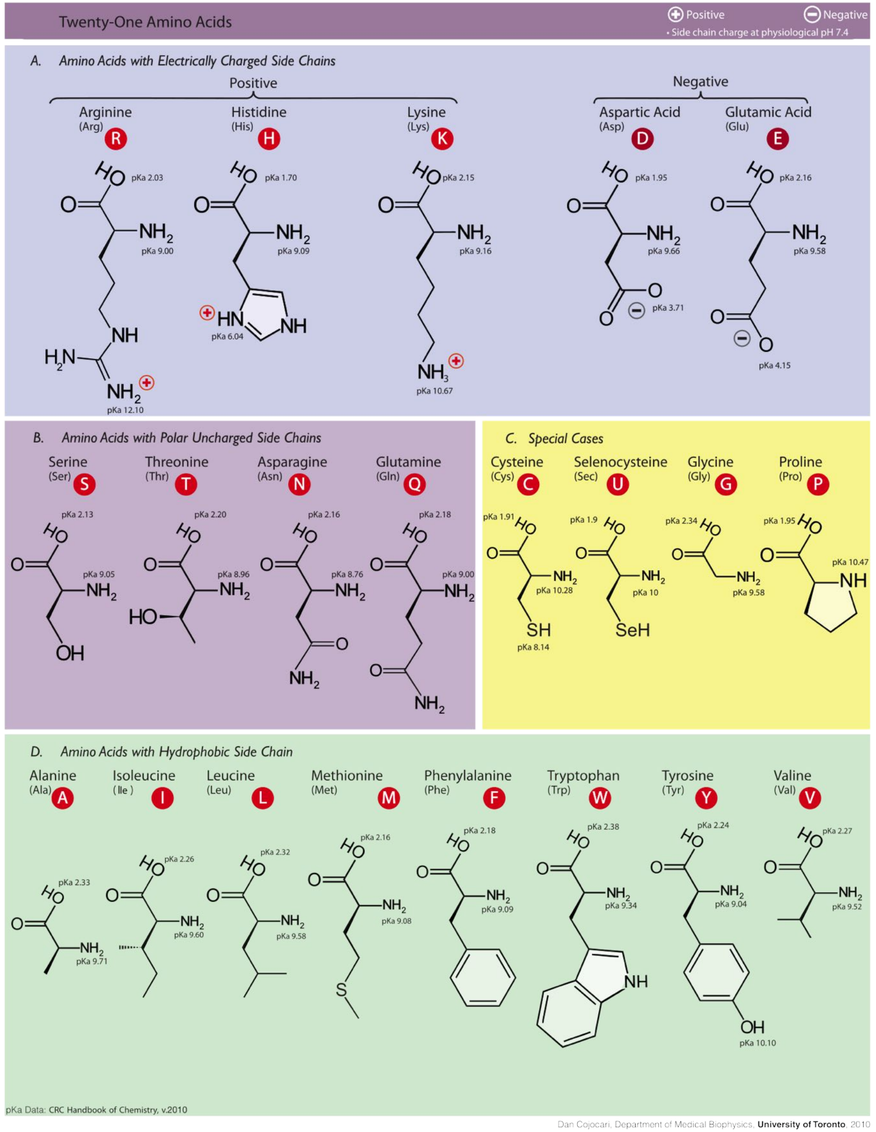


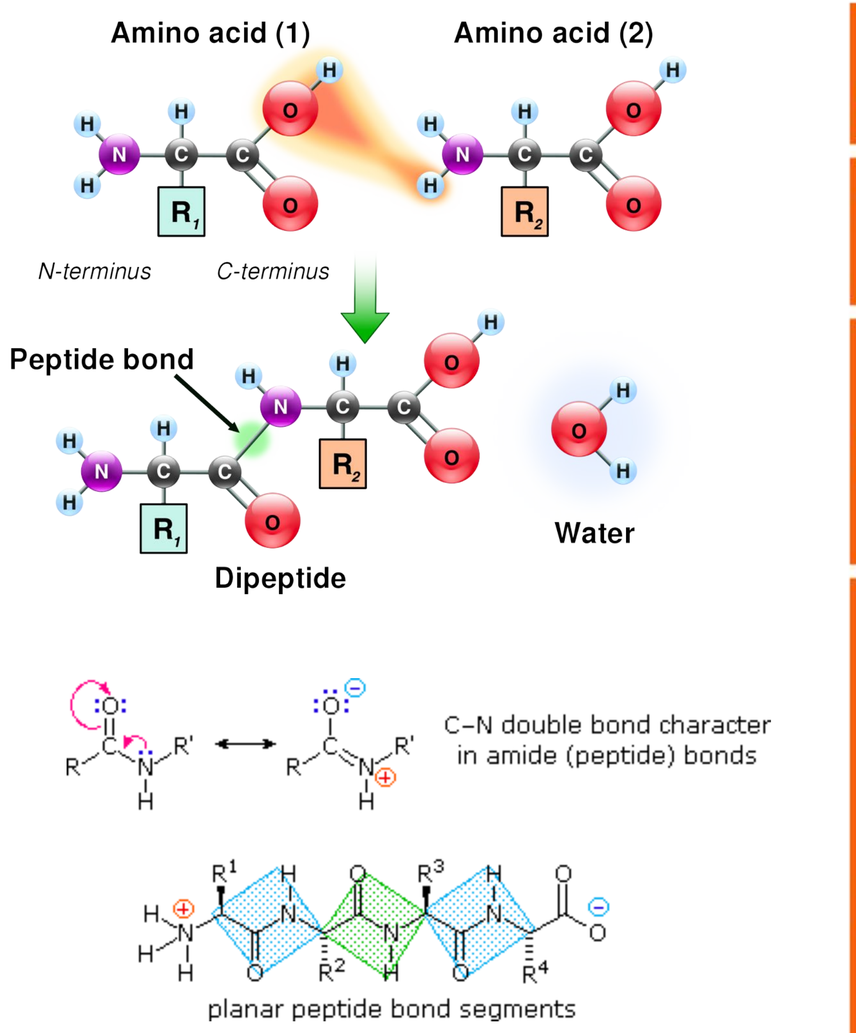

## 종별 구조 수

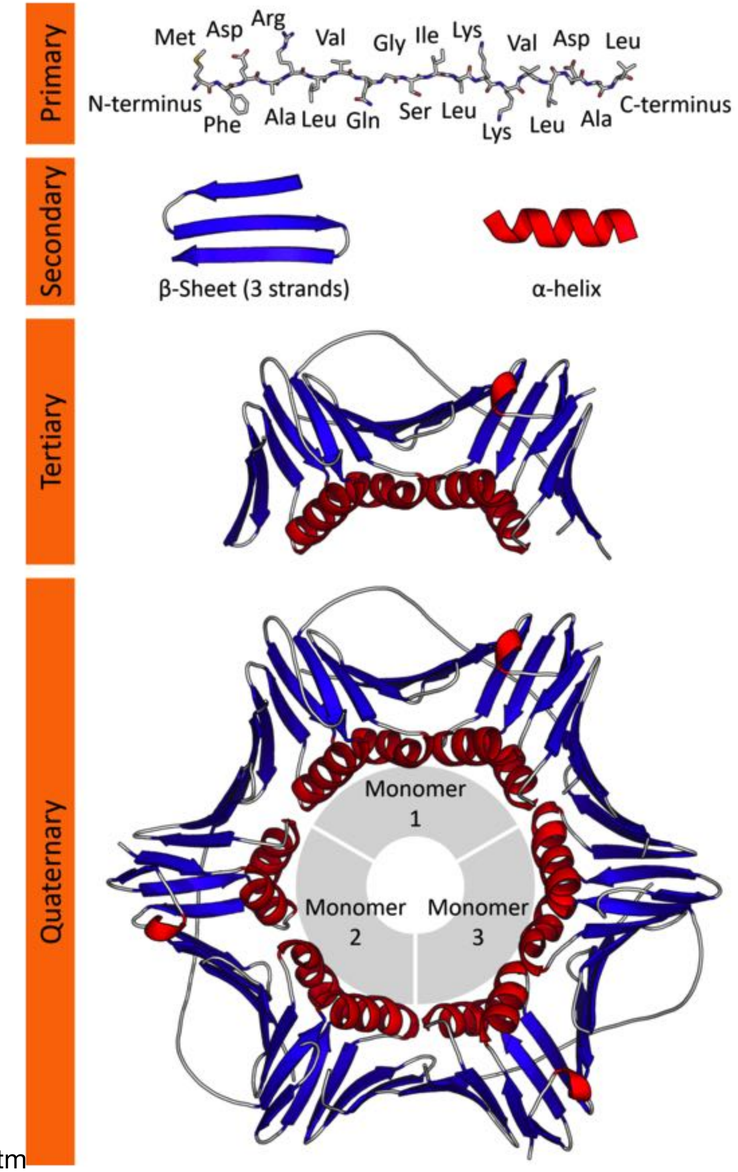



ARONTIER

# Protein Sequences



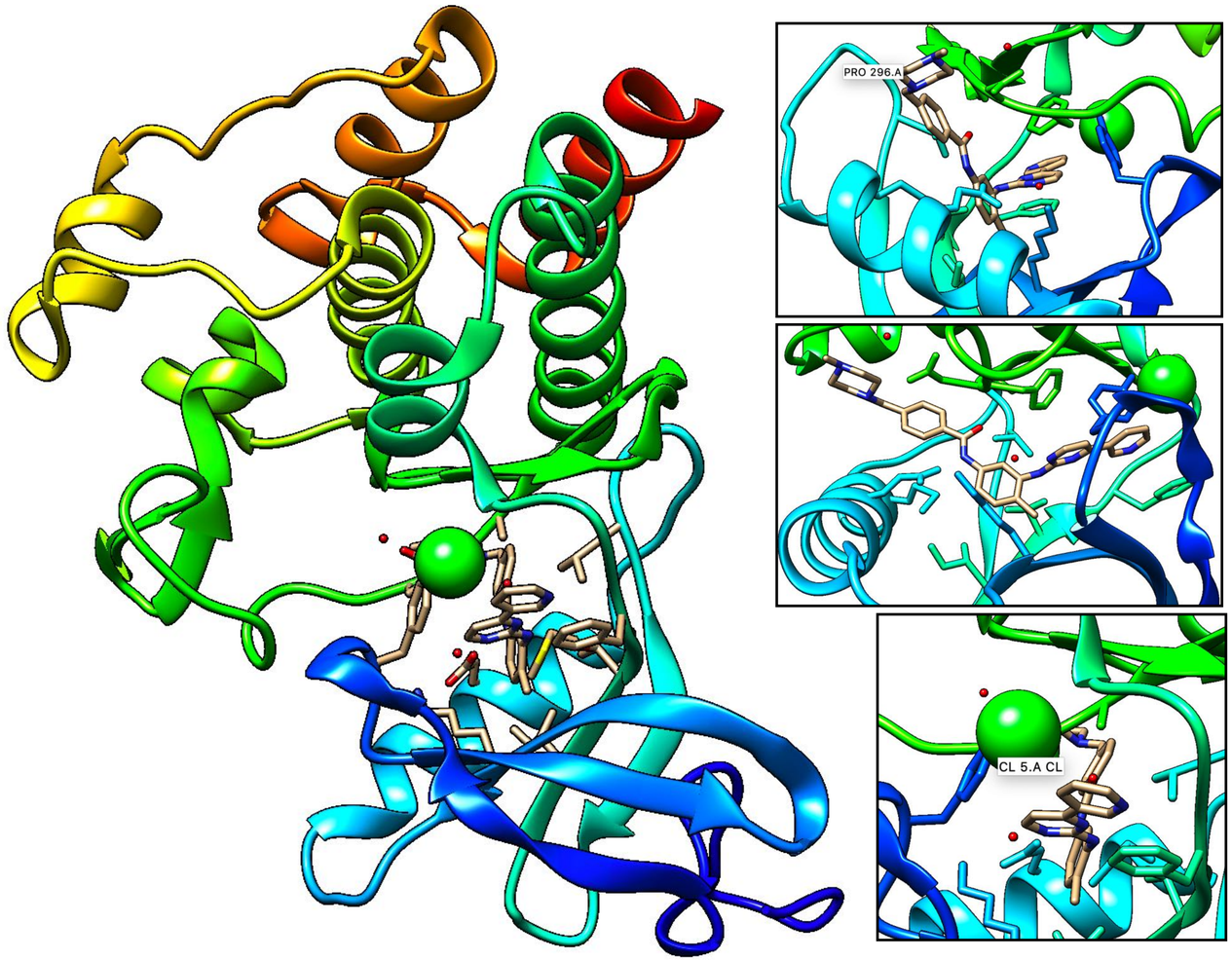



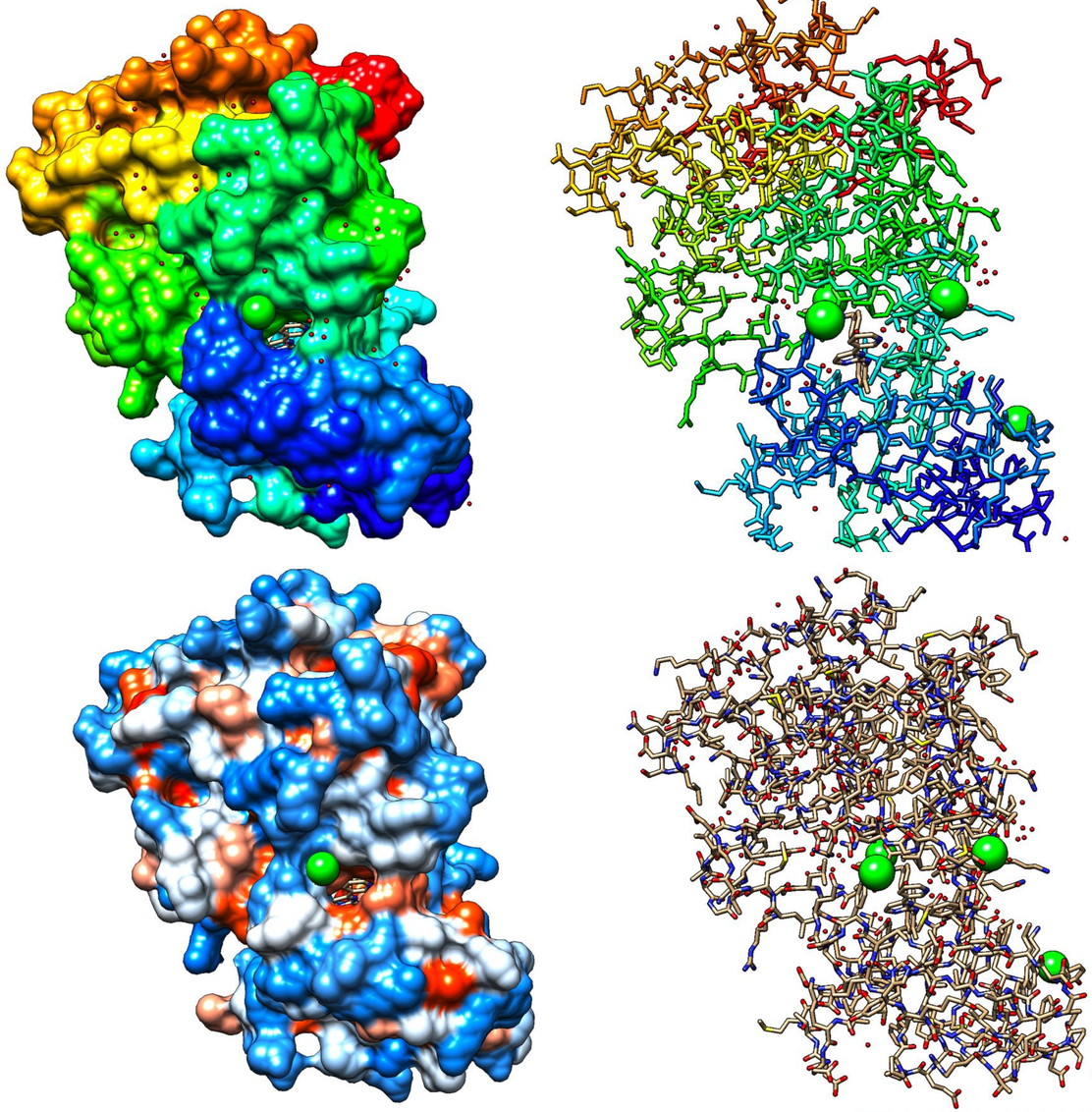




https://en.wikipedia.org/wiki/Peptide_bond

https://www2.chemistry.msu.edu/faculty/reusch/VirtTxtJml/protein2.htm

# Crystal structures of the kinase domain of c-Abl in complex with the small molecule inhibitors PD173955 and imatinib (STI-571)



https://www.rcsb.org/structure/1iep

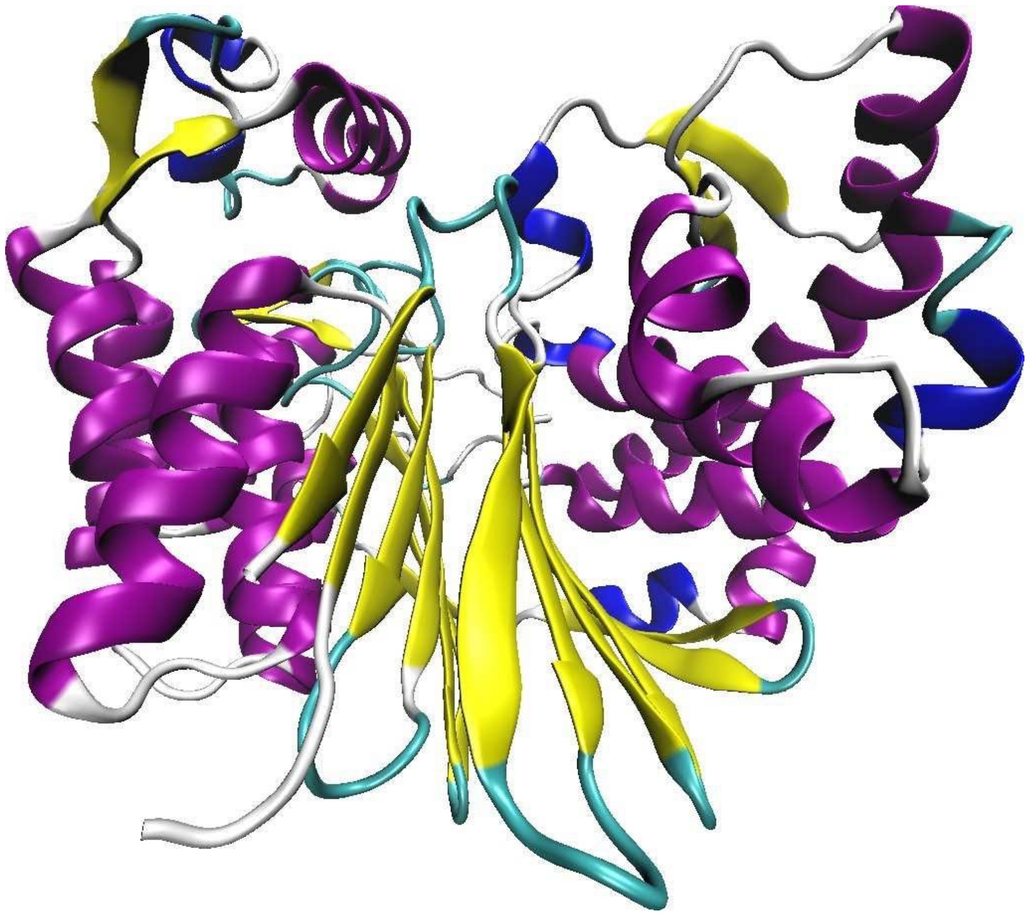
# Crystal structures of the kinase domain of c-Abl in complex with the small molecule inhibitors PD173955 and imatinib (STI-571)

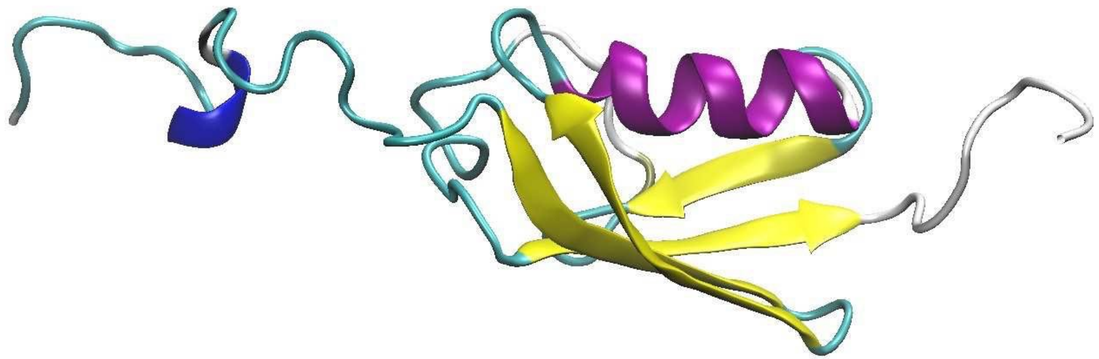
```
HEADER    TRANSFERASE                             10-APR-01   1IEP
TITLE     CRYSTAL STRUCTURE OF THE C-ABL KINASE DOMAIN IN COMPLEX
TITLE    2 WITH STI-571.
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: PROTO-ONCOGENE TYROSINE-PROTEIN KINASE ABL;
COMPND   3 CHAIN: A, B;
COMPND   4 FRAGMENT: KINASE DOMAIN;
COMPND   5 SYNONYM: P150, C-ABL;
COMPND   6 EC: 2.7.1.112;
COMPND   7 ENGINEERED: YES
SOURCE    MOL_ID: 1;
SOURCE   2 ORGANISM_SCIENTIFIC: MUS MUSCULUS;
SOURCE   3 ORGANISM_COMMON: HOUSE MOUSE;
SOURCE   4 ORGANISM_TAXID: 10090;
SOURCE   5 EXPRESSION_SYSTEM: SPODOPTERA FRUGIPERDA;
SOURCE   6 EXPRESSION_SYSTEM_COMMON: FALL ARMYWORM;
SOURCE   7 EXPRESSION_SYSTEM_TAXID: 7108;
SOURCE   8 EXPRESSION_SYSTEM_VECTOR_TYPE: BACULOVIRUS;
SOURCE   9 EXPRESSION_SYSTEM_PLASMID: PFASTBAC
KEYWDS    KINASE, KINASE INHIBITOR, STI-571, ACTIVATION LOOP,
KEYWDS   2 TRANSFERASE
EXPDTA    X-RAY DIFFRACTION
AUTHOR    B.NAGAR,W.BORNMANN,T.SCHINDLER,B.CLARKSON,J.KURIYAN
REVDAT   3   24-FEB-09 1IEP    1       VERSN
REVDAT   2   01-JUL-03 1IEP    1       JRNL
REVDAT   1   18-APR-01 1IEP    0
JRNL        AUTH   B.NAGAR,W.BORNMANN,P.PELLICENA,T.SCHINDLER,
JRNL        AUTH 2 D.R.VEACH,W.T.MILLER,B.CLARKSON,J.KURIYAN
JRNL        TITL   CRYSTAL STRUCTURES OF THE KINASE DOMAIN OF C-ABL
JRNL        TITL 2 IN COMPLEX WITH THE SMALL MOLECULE INHIBITORS
JRNL        TITL 3 PD173955 AND IMATINIB (STI-571)
JRNL        REF    CANCER RES.                   V.  62  4236 2002
JRNL        REFN                   ISSN 0008-5472
JRNL        PMID   12154025
REMARK   1
REMARK   2
REMARK   2 RESOLUTION.    2.10 ANGSTROMS.
REMARK   3
```

```
ATOM     25  CA  SER A 228      20.130  31.969  -5.773  1.00 62.95           C
ATOM     26  C   SER A 228      21.464  32.040  -5.031  1.00 65.13           C
ATOM     27  O   SER A 228      22.045  31.008  -4.688  1.00 66.16           O
ATOM     28  CB  SER A 228      20.364  31.447  -7.191  1.00 62.28           C
ATOM     29  OG  SER A 228      21.180  32.341  -7.928  1.00 62.06           O
ATOM     30  N   SER A 229      21.950  33.253  -4.788  1.00 64.53           N
ATOM     31  CA  SER A 229      23.216  33.437  -4.085  1.00 65.76           C
ATOM     32  C   SER A 229      23.144  32.931  -2.643  1.00 66.52           C
ATOM     33  O   SER A 229      22.121  33.077  -1.970  1.00 64.82           O
ATOM     34  CB  SER A 229      23.619  34.914  -4.089  1.00 66.77           C
ATOM     35  OG  SER A 229      24.716  35.143  -3.220  1.00 65.78           O
ATOM     36  N   PRO A 230      24.239  32.327  -2.153  1.00 66.91           N
ATOM     37  CA  PRO A 230      24.317  31.793  -0.789  1.00 67.68           C
ATOM     38  C   PRO A 230      24.165  32.880   0.276  1.00 67.83           C
ATOM     39  O   PRO A 230      23.752  32.609   1.406  1.00 68.00           O
ATOM     40  CB  PRO A 230      25.702  31.145  -0.751  1.00 67.73           C
ATOM     41  CG  PRO A 230      25.937  30.756  -2.179  1.00 67.84           C
ATOM     42  CD  PRO A 230      25.446  31.972  -2.919  1.00 66.34           C
ATOM     43  N   ASN A 231      24.500  34.111  -0.097  1.00 66.49           N
ATOM     44  CA  ASN A 231      24.419  35.239   0.819  1.00 66.05           C
ATOM     45  C   ASN A 231      23.237  36.164   0.516  1.00 61.13           C
ATOM     46  O   ASN A 231      23.340  37.382   0.668  1.00 58.71           O
ATOM     47  CB  ASN A 231      25.726  36.040   0.765  1.00 70.48           C
ATOM     48  CG  ASN A 231      26.937  35.211   1.171  1.00 74.28           C
ATOM     49  OD1 ASN A 231      26.811  34.220   1.891  1.00 78.33           O
ATOM     50  ND2 ASN A 231      28.117  35.619   0.716  1.00 74.69           N
ATOM     51  N   TYR A 232      22.114  35.591   0.092  1.00 55.17           N
ATOM     52  CA  TYR A 232      20.945  36.405  -0.222  1.00 50.20           C
ATOM     53  C   TYR A 232      20.279  36.969   1.023  1.00 48.25           C
ATOM     54  O   TYR A 232      19.972  36.238   1.966  1.00 52.46           O
ATOM     55  CB  TYR A 232      19.891  35.609  -0.999  1.00 46.52           C
ATOM     56  CG  TYR A 232      18.676  36.451  -1.350  1.00 43.11           C
ATOM     57  CD1 TYR A 232      18.715  37.355  -2.411  1.00 40.63           C
ATOM     58  CD2 TYR A 232      17.508  36.387  -0.583  1.00 41.66           C
ATOM     59  CE1 TYR A 232      17.625  38.180  -2.700  1.00 42.95           C
ATOM     60  CE2 TYR A 232      16.408  37.210  -0.866  1.00 37.96           C
ATOM     61  CZ  TYR A 232      16.478  38.102  -1.921  1.00 39.57           C
ATOM     62  OH  TYR A 232      15.420  38.937  -2.194  1.00 40.75           O
```

| COLUMNS | DATA TYPE | FIELD | DEFINITION |
|---|---|---|---|
| 1 - 6 | Record name | ATOM | |
| 7 - 11 | Integer | serial | Atom serial number |
| 13 - 16 | Atom | name | Atom name. |
| 17 | Character | altLoc | Alternate location indicator |
| 18 - 20 | Residue name | resName | Residue name. |
| 22 | Character | chainID | Chain identifier |
| 23 - 26 | Integer | resSeq | Residue sequence number. |
| 27 | AChar | iCode | Code for insertion of residues. |
| 31 - 38 | Real(8.3) | x | Orthogonal coordinates for X in Angstroms. |
| 39 - 46 | Real(8.3) | y | Orthogonal coordinates for Y in Angstroms. |
| 47 - 54 | Real(8.3) | z | Orthogonal coordinates for Z in Angstroms. |
| 55 - 60 | Real(6.2) | Occupancy | Occupancy. |
| 61 - 66 | Real(6.2) | tempFactor | Temperature factor. |
| 77 - 78 | LString(2) | element | Element symbol, right-justified. |
| 79 - 80 | LString(2) | charge | Charge on the atom. |

# Disordered Protein





ARONTIER

# Can't We Predict 3D Protein Structure Computationally?

- There are two main approaches to predicting protein structures.
- Homology modeling
  - Protein sequences and structures are the product of billions of years of evolution. If two proteins are near relatives (the technical term is "homologs") that only recently diverged from each other, they probably have similar structures.
  - To predict a protein's structure by homology modeling, you first look for a homolog whose structure is already known, then try to adjust it based on differences between the sequences of the two proteins.
  - Homology modeling works reasonably well for determining the overall shape of a protein, but it often gets details wrong.
- Physical modeling.
  - Using knowledge of the laws of physics, you try to explore many different conformations the protein might take on and predict which one will be most stable.
  - This method requires enormous amounts of computing time.
  - Until about a decade ago, it simply was impossible. Even today it is only practical for small, fast-folding proteins. Furthermore, it requires physical approximations to speed up the calculation, and those reduce the accuracy of the result. Physical modeling will often predict the right structure, but not always.

# A Short Primer on Protein Binding

- Proteins often bind to small molecules. Sometimes that binding behavior is central to the protein's function.
- Understanding the details of how, where, and when molecules bind to proteins is critical to understanding their functions and developing drugs.
- Protein binding involves lots of very specific interactions, which makes it hard to predict computationally.
  - A tiny change in the positions of just a few atoms can determine whether or not a molecule binds to a protein.
  - Many proteins are flexible and constantly moving. A protein might be able to bind a molecule when it's in certain conformations, but not when it's in others.
  - Binding in turn may cause further changes to a protein's conformation, and thus to its function.

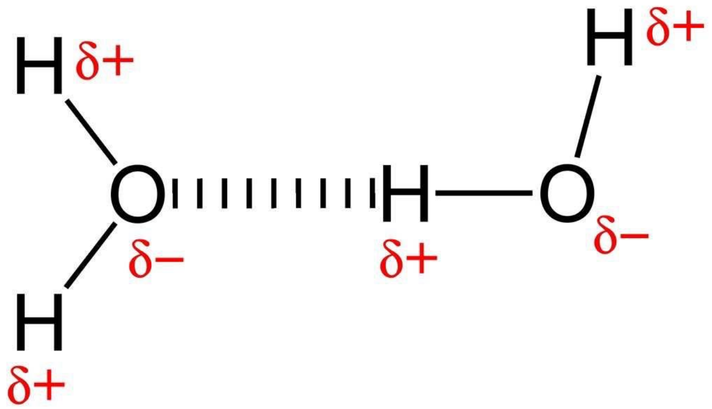
# Biophysical Featurizations

- The behaviors of biophysical systems are critically constrained by their 3D structures, so the 2D techniques from previous chapters miss crucial information.
- As a result, we will discuss a pair of new featurization techniques in this chapter.
  - Grid featurization
    - explicitly searches a 3D structure for the presence of critical physical interactions such as hydrogen bonds and salt bridges (more on these later), which are known to play an important role in determining protein structure.
    - The advantage of this technique is that we can rely upon a wealth of known facts about protein physics.
    - The weakness, of course, is that we are bound by known physics and lessen the chance that our algorithms will be able to detect new physics.
  - atomic featurization
    - which simply provides a processed representation of the 3D positions and identities of all atoms in the system.
    - This makes the challenge for the learning algorithm considerably harder, since it must learn to identify critical physical interactions, but it also makes it feasible for learning algorithms to detect new patterns of interesting behavior.

# Grid Featurization

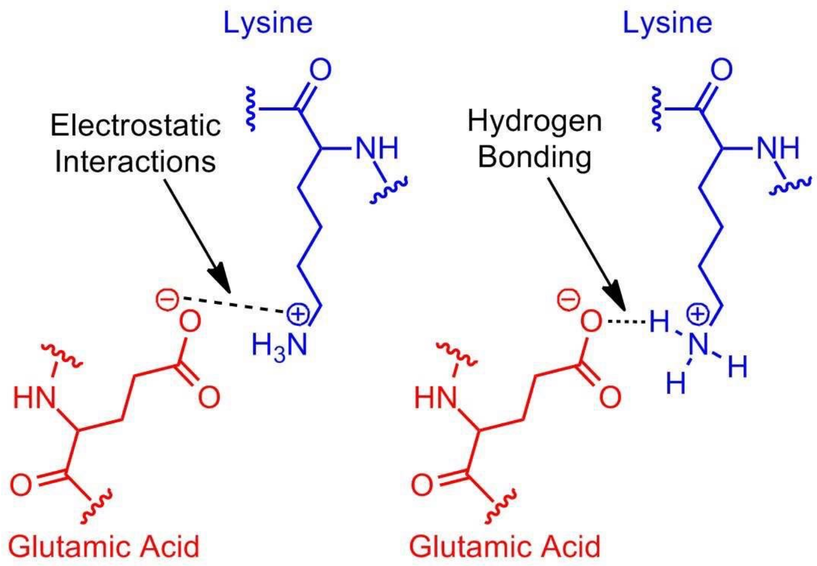
- By converting biophysical structures into vectors, we can use machine learning algorithms to make predictions about them.
- A featurization technique would need to have significant knowledge about the chemistry of such systems baked into it by design, so it could pull out useful features.
  - These features might include counts of noncovalent bonds between the protein and ligand, such as hydrogen bonds or other interactions.
- DeepChem has such a featurizer available.
  - RdkitGridFeaturizer summarizes a set of relevant chemical information into a brief vector for use in learning algorithms.
  - While it's not necessary to understand the underlying science in depth to use the featurizer, it will still be useful to have a basic understanding of the underlying physics.
  - The grid featurizer searches for the presence of such chemical interactions within a given structure and constructs a feature vector that contains counts of these interactions.

# Grid Featurization : HYDROGEN BONDS



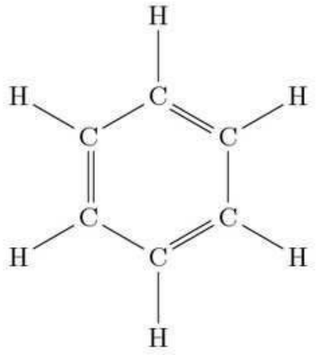

- Because hydrogen atoms are so small, they can get very close to other atoms, leading to a strong electrostatic attraction.
- This makes hydrogen bonds one of the strongest noncovalent interactions.
  - They are a critical form of interaction that often stabilizes molecular systems.
  - For example, water's unique properties are due in large part to the network of hydrogen bonds that form between water molecules.
- The RdkitGridFeaturizer attempts to count the hydrogen bonds present in a structure by checking for pairs of protein/ligand atoms of the right types that are suitably close to one another.
  - This requires applying a cutoff to the distance, which is somewhat arbitrary.
  - In reality there is not a sharp division between atoms being bonded and not bonded. This may lead to some misidentified interactions, but empirically, a simple cutoff tends to work reasonably well.

# Grid Featurization : SALT BRIDGES

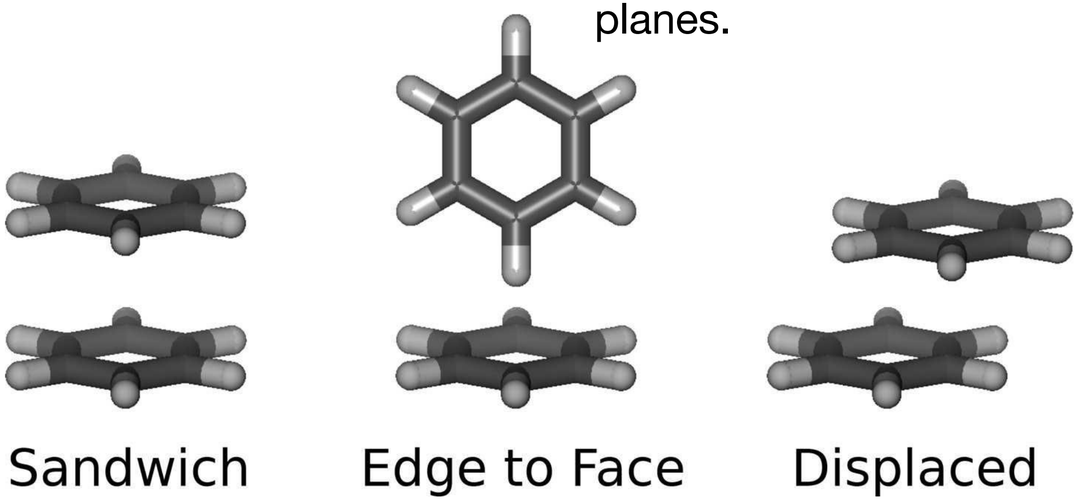



- A salt bridge is a noncovalent attraction between two amino acids, where one has a positive charge and the other has a negative charge.
- It combines both ionic bonding and hydrogen bonding.
- Although these bonds are relatively weak, they can help stabilize the structure of a protein by providing an interaction between distant amino acids in the protein's sequence.
- The grid featurizer attempts to detect salt bridges by explicitly checking for pairs of amino acids (such as glutamic acid and lysine) that are known to form such interactions, and that are in close physical proximity in the 3D structure of the protein.

# Grid Featurization : PI-STACKING INTERACTIONS



- Pi-stacking interactions are a form of noncovalent interaction between aromatic rings.
- These are flat, ring-shaped structures that appear in many biological molecules.
- Roughly speaking, pi-stacking interactions occur when two aromatic rings "stack" on top of each other.
  - Such stacking interactions, like salt bridges, can help stabilize various macromolecular structures.
- Importantly, pi-stacking interactions can be found in ligand-protein interactions.
  - The grid featurizer counts these interactions by detecting the presence of aromatic rings and checking for the distances between their centroids and the angles between their two planes.




ARONTIER

# Grid Featurization : FINGERPRINTS

- These fingerprints count the number of fragments of a given type in the molecule, then use a hash function to fit these fragment counts into a fixed-length vector.
- Such fragment counts can be used for 3D molecular complexes as well.
- Although merely counting the fragments is often insufficient to compute the geometry of the system, the knowledge of present fragments can nevertheless be useful for machine learning systems.
- This might perhaps be due to the fact that the presence of certain fragments can be strongly indicative of some molecular events.

# Grid Featurization : SOME IMPLEMENTATION DETAILS

- To search for chemical features such as hydrogen bonds, the dc.feat.RdkitGridFeaturizer needs to be able to effectively work with the geometry of the molecule.
  - DeepChem uses the RDKit library to load each molecule, protein, and ligand, into a common in-memory object.
  - These molecules are then transformed into NumPy arrays that contain the positions of all the atoms in space.
- performing a (crude) detection of a hydrogen bond simply requires looking at all pairs of atoms that could conceivably form a hydrogen bond (such as oxygen and hydrogen) that are sufficiently close to one another.
  - The same computational strategy is used for detecting other kinds of bonds.
  - For handling aromatic structures, there's a bit of special code to detect the presence of aromatic rings in the structure and compute their centroids.

# Atomic Featurization

- At the end of the previous section, we gave a brief overview of how features such as hydrogen bonds are computed by the RdkitGridFeaturizer.
  - Most operations transform a molecule with N atoms into a NumPy array of shape (N, 3) and then perform a variety of extra computations starting from these arrays.
- You can easily imagine that featurization for a given molecule could simply involve computing this (N, 3) array and passing it to a suitable machine learning algorithm.
  - The model could then learn for itself what features were important, rather than relying on a human to select them and code them by hand.
- The (N, 3) position array doesn't distinguish atom types, so you also need to provide another array that lists the atomic number of each atom.
  - As a second implementation-driven note, computing pairwise distances between two position arrays of shape (N, 3) can be very computationally expensive. It's useful to create "neighbor lists" in a preprocessing step, where the neighbor list maintains a list of neighboring atoms close to any given atom.
- DeepChem provides a **dc.feat.ComplexNeighborListFragmentAtomicCoordinates** featurizer that handles much of this for you.

# The PDBBind Case Study

- We will start by introducing the PDBBind dataset and the problem of binding free energy prediction.
- We will then provide code examples of how to featurize the PDBBind dataset and demonstrate how to build machine learning models for it.
- We will end the case study with a discussion of how to evaluate the results.

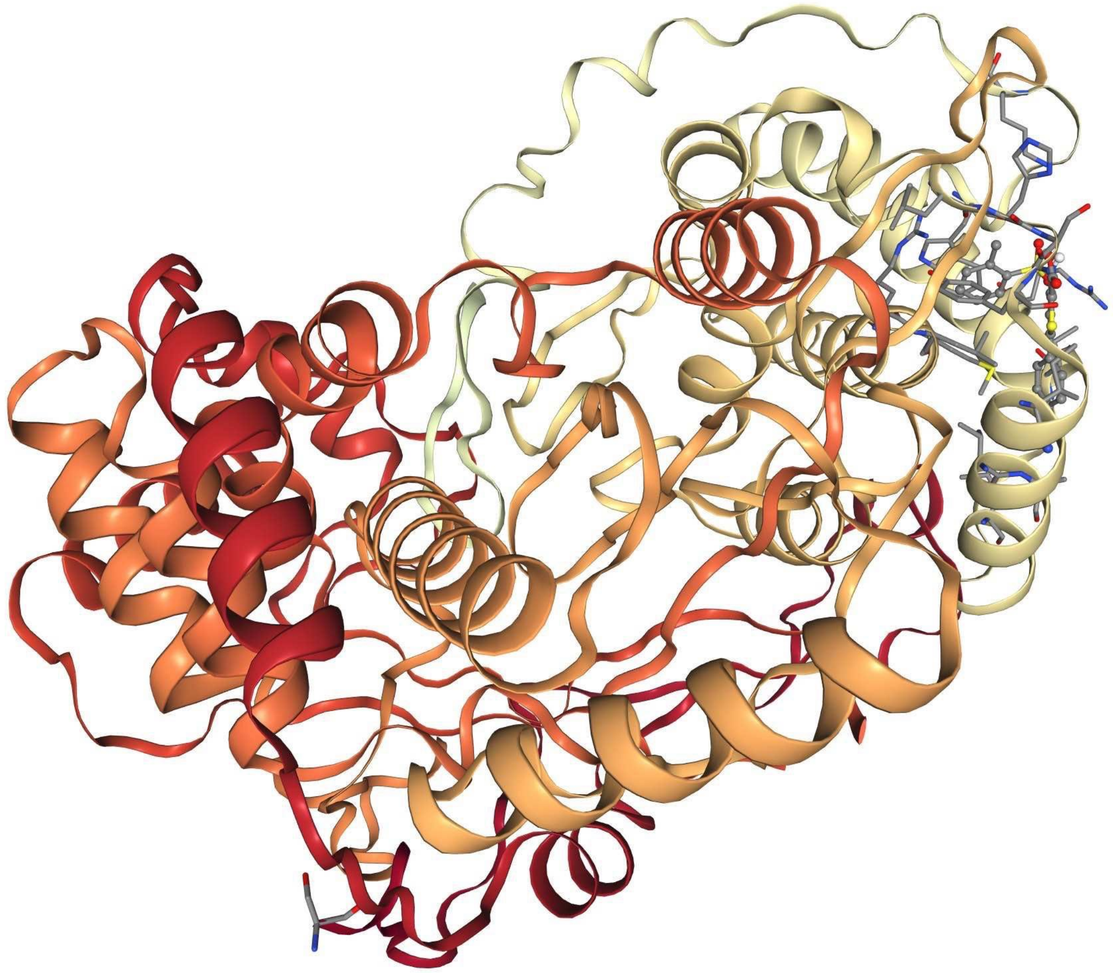

ARONTIER

# PDBBind Dataset

- The PDBBind dataset contains a large number of biomolecular crystal structures and their binding affinities.
  - A biomolecule is any molecule of biological interest. That includes not just proteins, but also nucleic acids (such as DNA and RNA), lipids, and smaller drug-like molecules.
  - A binding affinity is the experimentally measured affinity of two molecules to form a complex, with the two molecules interacting. If it is energetically favorable to form such a complex, the molecules will spend more time in that configuration as opposed to another one.
- The PDBBind dataset has gathered structures of a number of biomolecular complexes.
  - The large majority of these are protein–ligand complexes, but the dataset also contains protein–protein, protein–nucleic acid, and nucleic acid–ligand complexes.
  - The full dataset contains close to 15,000 such complexes, with the "refined" and "core" sets containing smaller but cleaner subsets of complexes.
  - Each complex is annotated with an experimental measurement of the binding affinity for the complex.
  - The learning challenge for the PDBBind dataset is to predict the binding affinity for a complex given the protein–ligand structure.
- The data for PDBBind is gathered from the Protein Data Bank. Note that the data in the PDB is highly heterogeneous!
  - For this reason, we will primarily use the filtered refined subset of the PDBBind dataset for doing our experimental work.

# PDBBind Dataset

- It can be very hard to understand the contents of a PDB file, so let's visualize a protein. We will use the NGLview visualization package, which integrates well with Jupyter notebooks.
- In the notebook associated with this chapter in the code repository, you will be able to manipulate and interact with the visualized protein. For now, Figure 5-10 shows a visualization of a protein–ligand complex (2D3U) generated within the Jupyter notebook.



# Featuring the PDBBind Dataset : Step 1

```
3 import deepchem as dc
4
5 grid_featurizer=dc.feat.RdkitGridFeaturizer(voxel_width=2.0,\
6         feature_types=['hbond','salt_bridge','pi_stack','cation_pi','ecfp','splif'],\
7         sanitize=True, flatten=True)
```

- sanitize=True : the featurizer to try to clean up any structures it is given.
  - Recall from our earlier discussion that structures are often malformed. The sanitization step will attempt to fix any obvious errors that it detects.
- flatten=True : the featurizer to output a one-dimensional feature vector for each input structure.
- feature_types : sets the types of biophysical and chemical features that the RdkitGridFeaturizer will attempt to detect in input structures.
  - The RdkitGridFeaturizer computes two different types of fingerprints, the ECFP and SPLIF fingerprints.
- voxel_width=2.0 : sets the size of the voxels making up the grid to 2 angstroms.
  - The RdkitGridFeaturizer converts a protein to a voxelized representation for use in extracting useful features.
  - For each spatial voxel, it counts biophysical features and also computes a local fingerprint vector.

# Featuring the PDBBind Dataset : Step 2

```
10 tasks, datasets, transformers = dc.molnet.load_pdbbind(featurizer="grid",split="random",subset="core")
11 train_dataset, valid_dataset, test_dataset = datasets
```

- 파라메터 설명
  - featurizer=“grid"
    - it will perform grid featurization automatically.
  - subset=“core”
    - we've loaded and featurized the core subset of PDBBind.

- 작동되지 않아서 다음과 같아 수정해서 실행

```
$ mkdir -p pdbbind/data
$ mkdir -p pdbbind/save
```

```
13 tasks, datasets, transformers = dc.molnet.load_pdbbind(featurizer="grid",split="random",subset="core",\
14         data_dir="pdbbind/data", save_dir="pdbbind/save",reload=False)
15 train_dataset, valid_dataset, test_dataset = datasets
```

# Featuring the PDBBind Dataset : Step 3 model fit

- train a classical model called a random forest

```
18 from sklearn.ensemble import RandomForestRegressor
19 sklearn_model = RandomForestRegressor(n_estimators=100)
20 model = dc.models.SklearnModel(sklearn_model)
21 model.fit(train_dataset)
```

- Building a neural network for predicting protein–ligand binding.

```
24 n_features = train_dataset.X.shape[1]
25 model = dc.models.MultitaskRegressor(n_tasks=len(pdbbind_tasks),n_features=n_features,\
26         layer_sizes=[2000, 1000],dropouts=0.5,learning_rate=0.0003)
27 model.fit(train_dataset, nb_epoch=250)
```

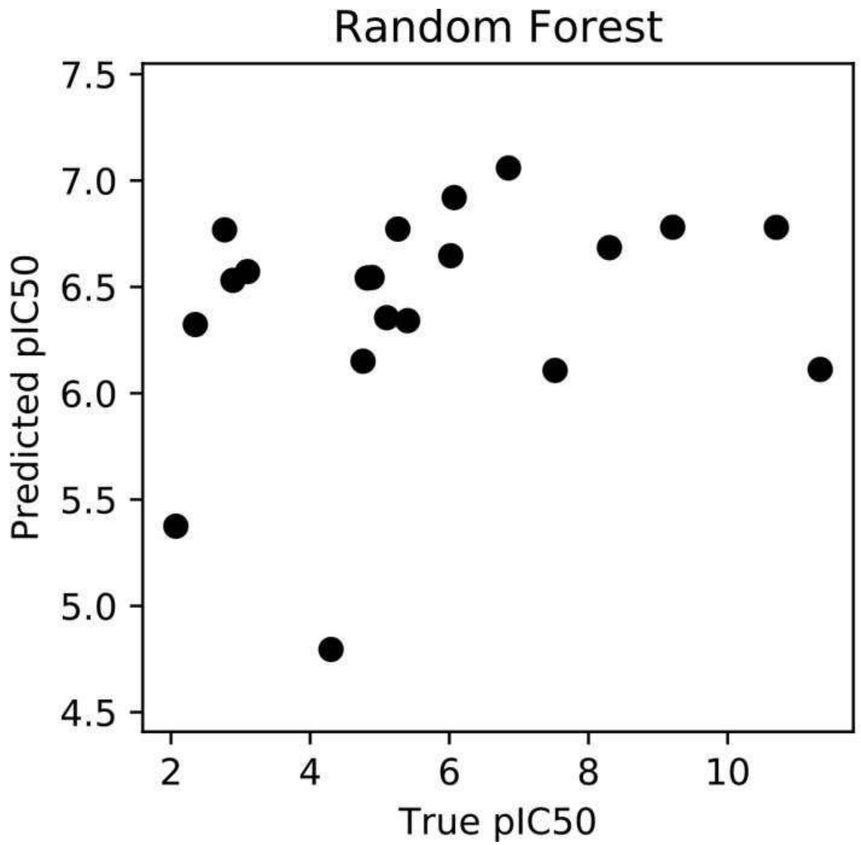
# Featuring the PDBBind Dataset : Step 4

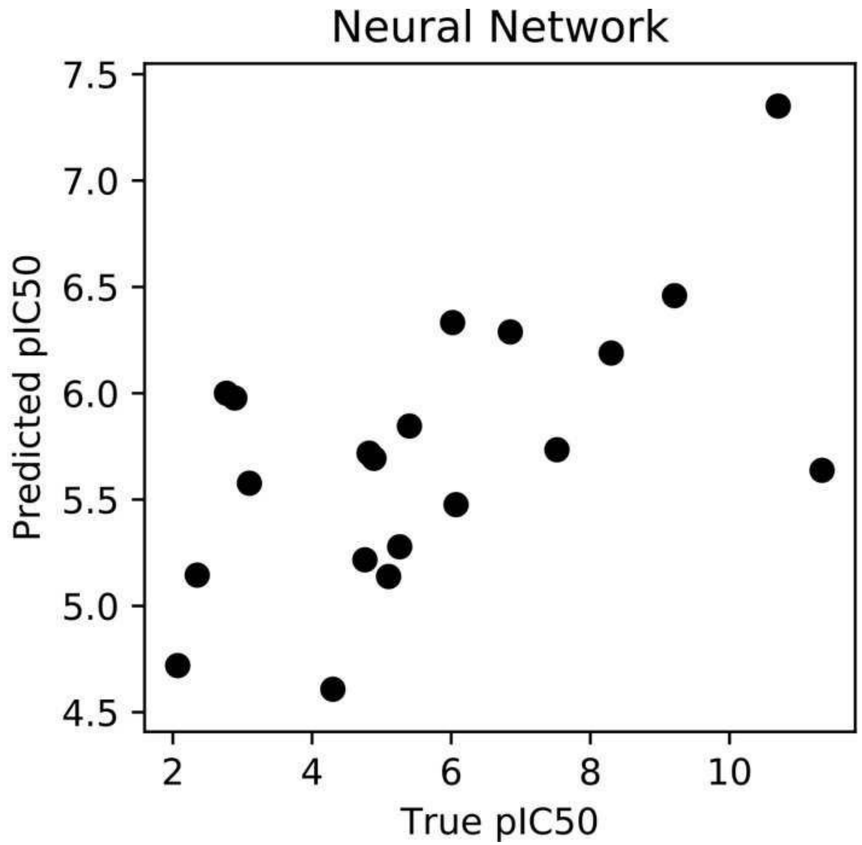
```
29
30 metric = dc.metrics.Metric(dc.metrics.pearson_r2_score)
31
32 print("Evaluating model")
33 train_scores = model.evaluate(train_dataset, [metric], transformers)
34 test_scores = model.evaluate(test_dataset, [metric], transformers)
35 print("Train scores") print(train_scores)
36 print("Test scores") print(test_scores)
37
```

- In order to evaluate the accuracy of the model, we have to first define a suitable metric.
  - Let's use the Pearson R2 score

# Featuring the PDBBind Dataset : Step 5. Results

- random forest :  a training set score of 0.979 but a test set score of only 0.133
- neural network : a training set score of 0.990 and a test set score of 0.359.



# Conclusion

- you've learned about applying deep learning to biophysical systems
  - in particular to the problem of predicting the binding affinity of protein–ligand systems.
- There remains a significant problem of scale. The atomic convolutional models are quite slow to train and require a great deal of memory.
- Antibody–antigen interactions are another form of critical biophysical interaction.

# END

# 환경설정

- Ubuntu 18.04

```
$ sudo apt install libxrender-dev

$ cd ~
$ wget https://repo.anaconda.com/archive/Anaconda3-2019.07-Linux-x86_64.sh
$ sh Anaconda3-2019.07-Linux-x86_64.sh
$ ~/anaconda3/bin/conda init bash

logout & login

$ conda create -n deepchem python=3.6
$ conda activate deepchem
$ conda install -c rdkit rdkit
$ conda install -c omnia pdbfixer
$ conda install -c deepchem deepchem
```