Chapter 1

De-novo design of metal binding moieties using machine learning

1.1 Inspiration from bioinformatics and computational modeling

Advances in software engineering normally found in data science, robotics, and telecommunication have been increasingly adopted in the biological sciences for -omic level analysis. The field of biology, traditionally thought of as a 'wet' science where discoveries are empirically derived, is now benefiting from software engineering to automate and process the now numerous datasets coming from genomics, proteomics, metabolomics, and so on. This new field, known as bioinformatics, uses data science techniques to analyze datasets from the life sciences to make conclusions which were typically reserved for empirically conducted experiments. In particular, statistical and machine learning methods are routinely employed in biological datasets to either uncover insights or train algorithms. The intention is to make conclusions, create trends, derive robust machine learned models to analyze future or unknown datasets. Many bioinformatic applications are directed towards diagnostics, or developing new drugs with predicted antibody-ligand binding interactions [**larranaga2006machine**].

There are two factors that contribute to the success of bioinformatic analysis.

The first is the quality and size of the dataset. The onus is primarily on the experimental work, such as preparing, processing, and measuring samples. For genomics, this is typically whole-genome sequencing. The rise in new sequencing technologies and knowledge of new culturing and cell isolation methods have contributed to the widespread success of genomic-based bioinformatics. The second factor is the efficiency and accuracy of the algorithm being used to analyze such datasets. New software tools and pipelines are constantly being made and improved upon[1], but the real crux of all bioinformatic tools is the number of biological assumptions that are made and whether or not these assumptions or models accurately describe the dataset. With poor assumptions, such as fitting a linear curve to a trend that is non-polynomial, bioinformatic methods would add little value to data interpretation. In the worst case, the results would be coherent but entirely inaccurate, causing a cascade of poor decisions inspired by errors.

These types of errors are a result of either underfitting, or overfitting. Underfitting a dataset fails occurs when a model fails to sufficiently predict or align with the trends or patterns of the dataset. This may be caused by poor model selection (e.g. linear versus non-linear models), or too few or too general variables. What may be worse than failing to construct a model is to construct one that is wrong. This type of error is overfitting, and may occur when there are too many parameters (i.e. assumptions) that are tweaked to force a fit that is not inherent in the dataset. What happens is a propagation of inaccurate conclusions for subsequent datasets, as an overfitted model will most likely not fit other datasets with its own nuances.

For the purpose of this work, the question is whether or not current datasets and bioinformatic tools are capable of modeling protein-metal binding interactions. The goal is to determine which configurations of amino acids confer the greatest metal-binding affinity. In other words, which sequence of amino acids is capable of capturing heavy metals from the environment with high specificity for bioremediation purposes. Rather than using well-known metal binding moieties for metal capture (examples in

[1] one out of many curated list of bioinformatic software tools: `https://github.com/danielecook/Awesome-Bioinformatics`

Chapter 4), is it instead possible to create de-novo peptide sequences tailored for a specific metal of interest? Already, many scientists in the protein engineering space know that the 6xHis tag or cysteine rich domains are good metal binders. Given that humans have learned some basic intuition on metal-peptide binding patterns, can a machine also learn these patterns?

To begin, a dataset of known metal-binding proteins should be derived or curated. Such dataset exists crystallographically thanks to the efforts of the Protein Data Bank (PDB) where solved protein crystal structures are publically deposited in an online server[2] [**berman2008**] (Figure **??**). In the PDB, more than 30% of all proteins contain some metal-binding domain [**waldron2009**], and these 3-dimensional interactions could be used to generate datasets for statistical analysis and machine learning on protein-metal binding interactions. A filtered PDB only containing structures with metal ligands is called the metal PDB (mPDB)[3] [**andreini2013**]. The mPDB provides additional metadata on the metal-protein structure such as metal location, possible amino acid binding partners, and parameters relevant to the quality of the 3D crystal structure (Figure **??**).

[2]PDB url: `https://www.rcsb.org/`
[3]metalPDB url: `http://metalweb.cerm.unifi.it/`

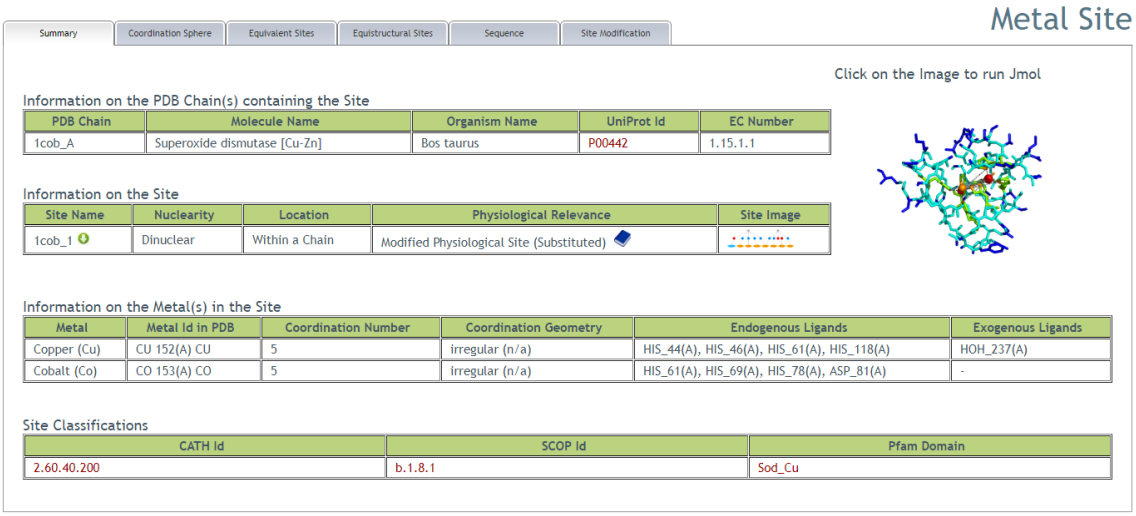


Figure 1.1 | Overview of protein database resources. The Protein Data Bank (PDB) contains all published protein crystal structures and downloadable files. File formats are text readable .mmCIF or .pdb formats which contain experimental metadata and atomic locations. The metal PDB (mPDB) only contain entries with metal ligands. Entires also provide additional data derived from the PDB files such as metal location, metal site environment, and metadata of the protein origin using other database servers such as Uniprot and Pfam.

The goal is to somehow reorganize the 3-dimensional information of each protein structure into a processable dataset for statistical analysis. Once an adequate dataset is derived, the next step would be to apply statistical or machine learning framework to discover patterns between metals and their preferred (or most occuring) protein-binding environment. It may not be necessary to deconstruct the entire protein structure, but to only analyze the vicinity around the metal binding site. Within these binding sites, assumptions around the relevance of each parameter (i.e. features) can be made, such as the importance of the distances between amino acids, identity of the amino acid, types of bonding, bonding geometry, and so on. With the appropriate machine learning tools it may be possible to train a model to understand these features and how it may impact metal binding affinity. The potential results can help answer two specific questions: 1) given an unknown binding site, how likely is it to be a metal binder, and if so, which metal? and 2) given a desired metal, what is the optimal sequence and arrangement of amino acids that confer the greated binding affinity?

1.2 Methods, algorithms, & data processing

The interface between biological data and designing an analytical pipeline to consume, transform, and interpret such data is possibly one of the most difficult steps in bioinformatics. The question arises as to how biological data, which is typically unordered, noisy, and sometimes uninterpretable by both machine and human, can be extracted of its fundamental features and then analyzed as a sequence of numbers or categories. Throughout these transformations, a caution is to maintain as much biological relevance without introducing contrived data that is not present in the original dataset, or eliminating/ignoring relevant data for the sake of efficiency or simplicity.

1.2.1 Extracting data from the metal PDB

List of protein structures with metal ligands was extracted from the *m*PDB by automating a webscraper that fetched all protein files that contained either Ag, Al, Au, Ba, Ca, Cd, Co, Cr, Cs, Cu, Fe, Ga, Hg, In, K, Li, Mg, Mn, Na, Ni, Pb, Pd, Pt, Rb, Sr, W, and Zn (there were no files that containted As, Si)[4].

The data for each file contained a PDB ID (a unique 4 digit alphanumeric code representing the protein structure), a number next to the PDB ID representing the metal instance (one protein can contain multiple metal binding sites, hence suffixing the ID code with incrementing numbers); the metal or metals found in that binding pocket; a string denoting the molecule which the metal belongs to (often it is just the metal, but for example a heme group would be included because of its Fe center) the numeric location of the metal (all atoms and molecules in a PDB file format are numbered), the chain letter, and the metal identity; and finally the ligands determined by the *m*PDB to be significant binding partners. These metal binding partners were delimited by a 3 amino acid or molecular code, followed by the numeric location of the binding partner, followed by a chain letter in which the binding partner was part

[4]Rare earth metals for Ce, Er, Eu, Gd, Ho, La, Lu, Os, Pa, Pr, Re, Sm, Ta, Tb, U, Yb were also extracted but not used in this analysis.

of in the overall protein structure. The data structure for the raw data taken from the *m*PDB can be found in Table **??**.

	SiteName	Metal(s)	Metal(s) in pdb file	Ligand(s)
format	PDBID_no.	$metal_x$	molecule_loc(chain)_metal	residue_loc(chain)
example	1apq_2	Cu	CU_125(A)_CU	TYR_76(A),
				HIS_105(A)
example	1arm_1	Hg, Cu	CU_315(A)_CU	GLU_270(A),
				HOH_320(A)
			HG_310(A)_HG	HIS_69(A),
				GLU_72(A),
				HIS_196(A),
				TRS_319(A),
				HOH_320(A)

Table 1.1 | Data format of protein files from the *m*PDB. Data from the *m*PDB of each metal instance contained the PDB ID, the identity of the metal for that instance, the molecular location of that metal, and the molecular location of the ligands that bind to that metal.

1.2.2 Filtering and cleaning data

Although the data from the *m*PDB contained valuable information, it does contain missing entries, redundancies, and is of a format which is difficult to parse for basic analytical pipelines that expect numerical values or consistent categorical strings. Therefore, a filtering step was performed to remove any erroneous entries, and the data was transformed to fit a particular format more amenable for downstream analysis (Figure **??**). *m*PDB entries were further validated by ping'ing the original PDB database to double check the existance of the protein structure.

The first filter was to remove any entries with multiple metals per metal binding site, as this would confuse the analysis as to which metal was more significant in the metal binding pocket. The second filter was whether or not the PDB file exists. Unfortunately some PDB files were either removed, archived, or did not exist when checked on `https://www.rcsb.org/`. The third filter was whether or not the

metal PDB data contained valid annotations of where the metal resides, both in the chain lettering and metal numbering. In some cases, some annotations in the *m*PDB were incorrect, and some metals did not exist in a small fraction of provided protein structures.

During the filtering and cleaning step, more features were extracted from the *m*PDB and PDB database. Useful features such as metal valency, metal binding geometry, the geometry quality (distorted or regular) were extracted. Additional metadata was extracted to paint a better picture of the protein (although not necesary for the analytical framework) such as UNIPROT ID, organism, and enzyme commission annotations (example of the new dataset structure is tabulated in Table **??**).

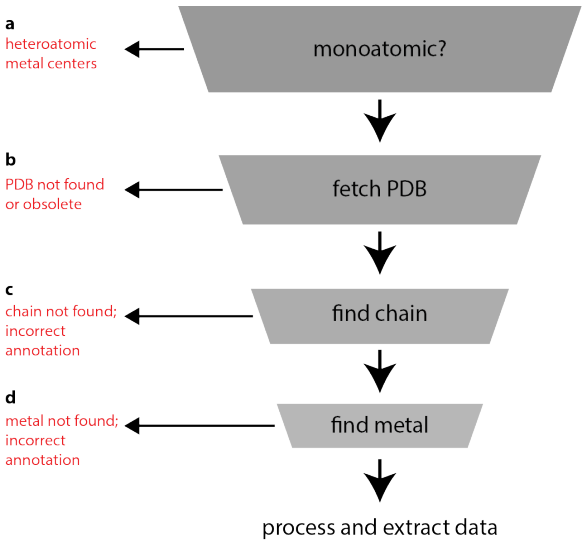


Figure 1.2 | Pipeline for parsing and extracting PDB information from the metal PDB. (**a**) Entries that contained multiple metal atoms per metal site were removed to avoid confusion during analysis. (**b**) PDB files that were unfetchable (e.g. obsolete, archived, or non-existent) were ignored. (**c, d**) Annotations provided by the metal PDB were sometimes incorrect. Often this would occur with wrong chain numberings or non-existent metal locations. This filtering step was performed after retrieving the protein structure from the PDB (`https://www.rcsb.org/`).

(a)

	ID	instance	chain	location id	metal id
format	PDBID	metal number	A, B, ...	loc(chain)	metal atom location
example	1q06	1	B	300(B)	24985

(b)

	protein	organism	Uniprot	EC
format	UNIPROT protein name	UNIPROT protein name	Uniprot accession number	enzyme commission number
example	HTH-type transcriptional regulator CueR	Escherichia coli	P0A9G4	-

(c)

	valency	geometry	idealized	ligands
format	0–9	linear, trigonal, ...	-, distored, regular	residue_loc(chain)
example	2	linear	regular	CYS_112(B), ...

Table 1.2 | Tranformed dataset structure of protein entires filtered from the *m*PDB. (**a**) The original metal PDB entries were cleaned to individully represent the PDB ID, metal instance for that PDB file, chain location, molecular location of the metal, and atomic location of the metal. (**b**) Additional metadata was extracted such as the UNIPROT ID, organism name, accession number, etc. (**c**) Categorical data such as metal valency, metal binding geometry, and ligands were also taken from the metal PDB.

1.2.3 Creating datasets processable by machine learning algorithms

The filtered and cleaned data from the *m*PDB had yet to undergo another transformation in order to be inputted into a statistical or machine learning framework. To create such an input–output pipeline the data structure was converted to numerical or categorical features. More so, the previously processed features were not feature rich

(Table **??**), as in, they did not quantitatively explain the metal-protein structure relationship in detail. An example would be the sparse 1–5 amino acid binding partner description per metal coordination sphere. Therefore, a new algorithm was developed to extract 3-dimensional data from the protein crystal structure which bared more fine-grained atomic data. The algorithm developed was a variation of the nearest-neighbor algorithm [**arya1998optimal**] in which an imaginary radii stretched from the metal center is modelled to extract the closest molecule/amino acid from that metal center (Figure **??**). The discovered neighbors were further processed to determine which atom from that molecule were closest, and the distance between the atom and the molecule from the metal center were also calculated[5].

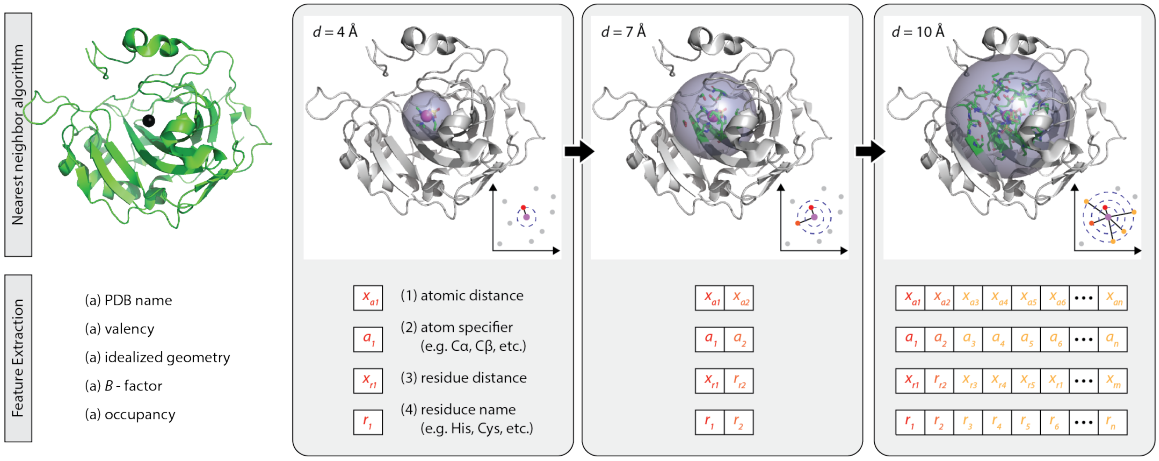


Figure 1.3 | Variation of the nearest-neighbor algorithm to extract molecular and atomic neighbors from the metal center of a protein structure. Molecular and atomic neighbors at incremental radii away from the metal center were identified using the nearest-neighbor algorithm and stored in arrays. These arrays were later concatenated to create either a 2D dataset or 1D row of features. Additional parameters were extracted during the algorithm. These parameters were the valency and geometry of the metal-binding environment, the metal's β-factor and its percent occupancy.

The features collected were the atoms closest to the metal center, and their distances away from the metal center. The same process was performed for nearest neighbor molecules and their distances. These features were ordered from closest to

[5]Bio.PDB package in python was used to handle PDB structural data: `https://biopython.org/wiki/The_Biopython_Structural_Bioinformatics_FAQ`

furthest, the closest being 1 Å and furthest being 50 Å away. All features were concatenated to create a 1D aray with each analyzed protein structure representing a row of a larger 2-dimensional data structure. These datasets were segregated by metal's analyzed. For example, there existsed separate datasets of acquired nearest-neighbor information for Li, Na...Pb, where each for of each dataset represents the features extracted from a single protein structure.

In addition, several metadata parameters were included in the feature list. They were the metal's β-factor and percent occupancy. These values helped score the confidence of the metal location in the protein structure, and future work would use these values to under-weigh or ignore potential outliers or bad instances.

(a)

	metal	ID	valency	geometry	idealized	βfactor	occupancy	anisotropy
example	Ag	1q06	2	linear	regular	26.6	1	-1

(b)

	atom name	atom distance (Å)	molecule name	molecule distance (Å)
example	SG, CB, CB, CB, CA, O	2.35, 3.2, 3.37, 3.5, 3.51, 3.6	CYS, CYS, CYS, SER, CYS, SER	4.17, 3.51, 4.17, 3.88, 3.51, 3.88

Table 1.3 | Training set data structure as input to machine learning frameworks. (**a**) Several metal-protein specific parameters such as the metal's valency and geometry, as well as its β-factor and percent occupancy were collected. (**b**) Arrays of atom names, atom distances, residue names, and residue distances sorted from closet to furthest from the metal center were concatenated into a single array.

1.3 Results

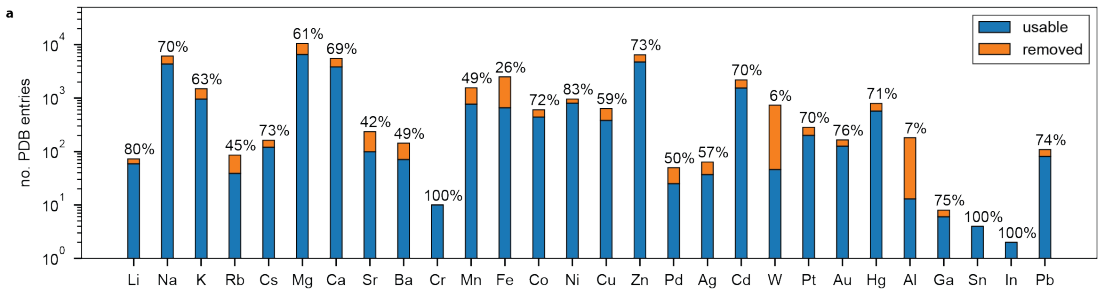
1.3.1 Analysis of protein-metal data curated by the *m*PDB

A significant portion of the curated protein structures from the *m*PDB were not usable, primarily because there were multiple metals per metal binding site, and some of the chain and metal location annotations were incorrect. For metals that

had more than 10 PDB entires, the filtering step removed almost 25–40% of entires (Figure **??**a).

The metals with the most PDB entries were Na, Mg, Ca, and Zn. However, alkaline and alkaline-earth metals like Na and Mg may be over-represented because these metals are frequently present in buffers during crystallization.More so, during PDB submission man of the crystal structure solute environment remains annotated, scoring hundreds to thousands of spectator ions. Unfortunately, the *m*PDB does not filter these entries, and in this work these entries propogated through the analysis. To remove these false-positive metal-bound proteins would require differentiating metals in the buffer from metals bound to the protein. To do so would require accessing the PDB structure directly and querying every metal. Alternatively, a threshold could be set that if a protein contains more than X number of metals, specifically Na, Mg, etc., then it should be eliminated. However, this assumption is crude, and may falsely eliminate good protein structures.

The fact that many metals in the protein structure entries were solutes rather than bound metals help explains the high β-factors and low percent occupancy for most of the alkaline and alkaline-earth metals. Also, many metals which are rarely found in proteins such as the metalloid and noble metals like Pt and Hg had poor β-factors and percent occupancies (Figure **??**b). When looking at the metal-bound protein structures holistically, on average each structure contains 2 or less binding sites (Figure **??**c). In other words, it is likely that a protein structure containing a particular metal will have one or two binding pockets for that metal.



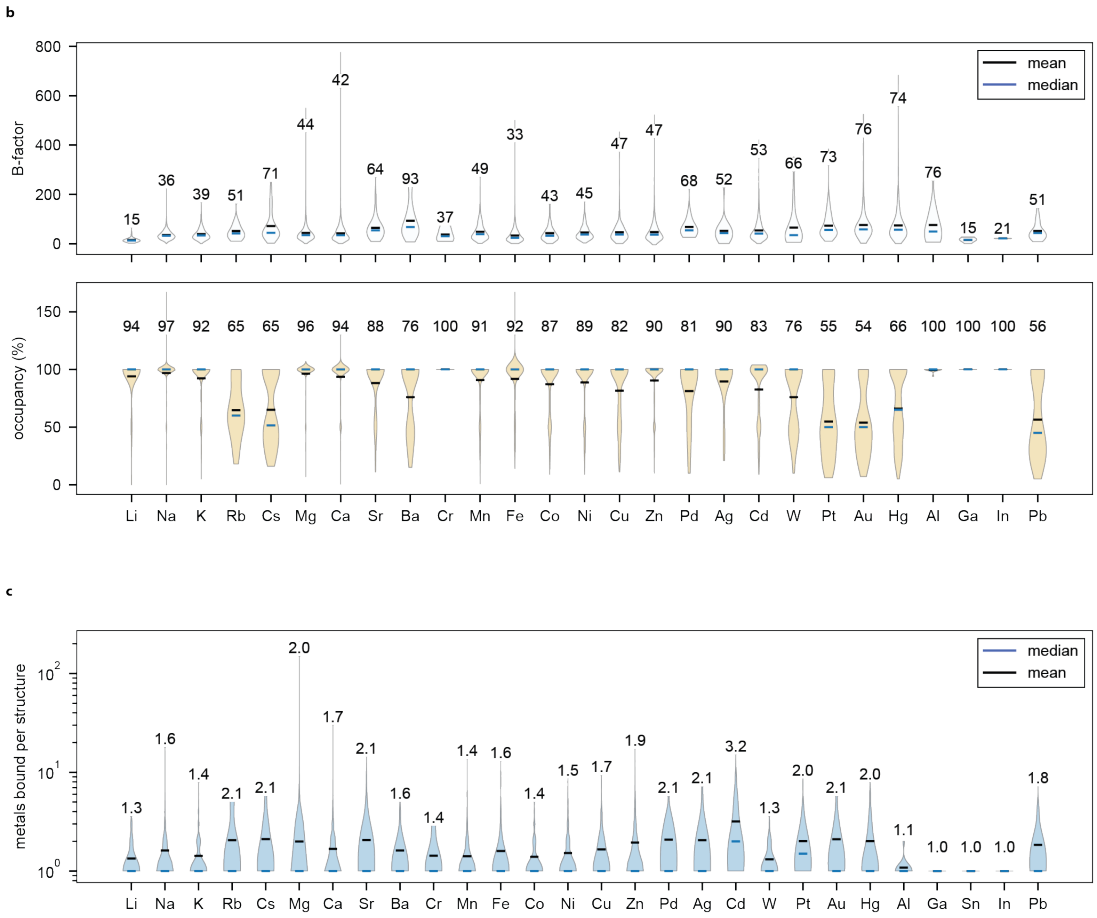


Figure 1.4 | Number of usable metal PDB entries and their statistics. (**a**) Several protein structures provided by the *m*PDB were not usable and had to be filtered (see Figure **??** on filtering criteria). Note the y-axis is in log scale, and that bar heights are not linearly proportional. Some metals such as Ga, Sn, and In, particularly from the semi-metals contained very few entries (less than 10). Alkaline and alkaline-earth metals were highly represented, most likely because these metals are often present as dissolved solutes during protein crystallization. (**b**) Plot showing β-factors and percent occupancy of each metal. Low β-factors suggest more accurate metal position, and higher percent occupancy suggest greater confidence in metal localization. Metals that are not often found in proteins such as Rb, Cs, Hg, etc. have high β-factors and low percent occupancies, which may suggest that these are not natural or favorable binding environments. (**c**) Plot showing the average number of metal binding sites per protein for a given metal. Overall, the average metal-binding proteins usually contain less than 2 metal binding sites.

In most metal-binding environments many metals, especially the alkaline and alkaline-earth metals, have valencies of 1 or 2. For transition metals they may be

found in several valent states ranging from 1–7 because of the electron donating d-orbitals; however, still many transition metal co-factors are found in the divalent state such as Mn^{2+}, Fe^{2+}, Zn^{2+} etc. So it was surpising to see that the valencies calculated by the *m*PDB were not between 1–2, but rather varying dramatically.

no.	val.	abbrv.	full geometry name	instances	percent
1	0	-	-	4086	15%
2	0	IRR	irregular	7530	28%
3	2	TRV	trigonal plane with a vacancy	2209	8%
4	2	LIN	linear	230	1%
5	3	TRI	trigonal plane	214	1%
6	3	SPV	square plane with a vacancy	634	2%
7	3	TEV	tetrahedron with a vacancy	1047	4%
8	4	BVP	trigonal bipyramid with a vacancy (equatorial)	234	1%
9	4	BVA	trigonal bipyramid with a vacancy (axial)	505	2%
10	4	SPL	square plane	537	2%
11	4	PYV	square pyramid with a vacancy (equatorial)	1124	4%
12	4	TET	tetrahedron	1405	5%
13	5	SPY	square pyramid	1247	5%
14	5	TBP	trigonal bipyramid	202	1%
15	5	TPV	trigonal prism with a vacancy	87	0%
16	6	OCT	octahedron	3219	12%
17	6	TPR	trigonal prism	49	0%
18	6	PVP	pentagonal bipyramid with a vacancy (equatorial)	491	2%
19	6	CTF	trigonal prism, square-face monocapped with a vacancy (capped face)	40	0%
20	6	CTN	trigonal prism, square-face monocapped with a vacancy (non-capped edge)	95	0%
21	6	PVA	pentagonal bipyramid with a vacancy (axial)	129	0%
22	6	CON	octahedron face monocapped with a vacancy (non-capped face)	77	0%
23	6	COF	octahedron face monocapped with a vacancy (capped face)	71	0%
24	7	HVP	hexagonal bipyramid with a vacancy (equatorial)	49	0%
25	7	CUV	cube with a vacancy	4	0%
26	7	CTP	trigonal prism square-face monocapped	99	0%
27	7	PBP	pentagonal bipyramid	544	2%
28	7	HVA	hexagonal bipyramid with a vacancy (axial)	2	0%
29	7	COC	octahedron face monocapped	148	1%
30	7	SAV	square antiprism with a vacancy	83	0%
31	8	BTT	trigonal prism triangular-face bicapped	0	0%
32	8	BOC	octahedron trans-bicapped	0	0%
33	8	BTS	trigonal prism square-face bicapped	56	0%
34	8	SQA	square antiprism	79	0%
35	8	CUB	cube	3	0%
36	8	HBP	hexagonal bipyramid	5	0%
37	9	CSA	square antiprism square-face monocapped	0	0%
38	9	TTP	trigonal prism square-face tricapped	0	0%

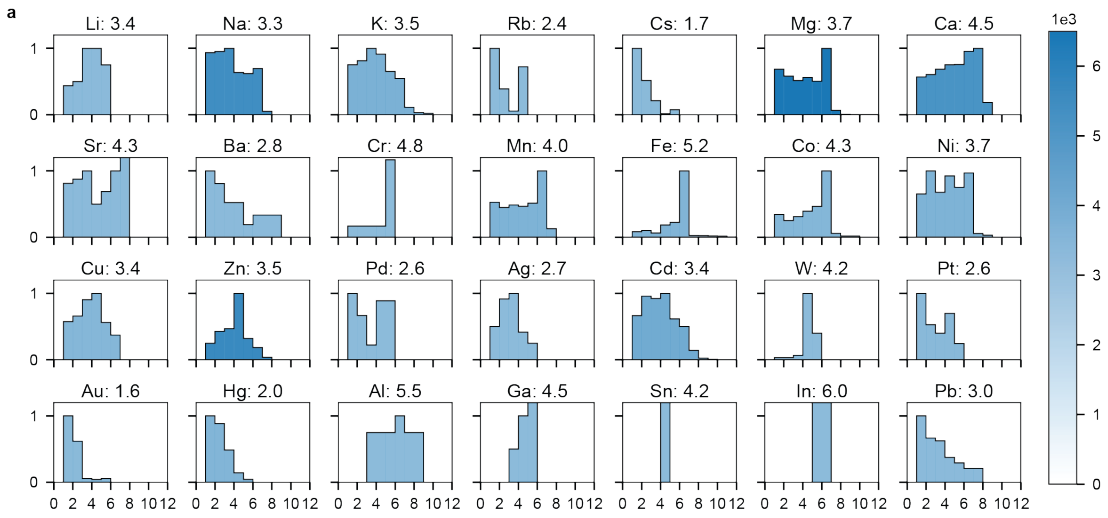
Table 1.4 | Table of metal binding geometries for all metals examined from the metal PDB. Many metal geometries could not be identified, or were irregular (>40%). The most prominent geometries were trigonal planar with a vacancy (valency of 2), and octahedral (valency of 8).

Elements such as Li, Na, and K had valencies above 3, and many of the metalloids had valencies above 4 (Figure **??**a). A hypothesis is that these valencies were calculated indirectly by the number of binding partners found in the protein structure. The number of metal-binders predicted by the *m*PDB may have simply counted the number of ligands and summed them to generate a valency value (Table **??**; "Ligand(s)" column).

The overall representation of metal geometries were either irregular (28%), not identifiable (15%), octahedral (12%), or trigonal planar with a vacancy (8%) (Table **??**). What this data suggest is that valency could be a poor identifier to distinguish different types of metal binding environments.

1.3.2 Differentiating protein-metal interactions by clustering steric and ligand data

Data strictly derived from the *m*PDB were used as input datasets for statistical and clustering analysis. The intention was to use features that were filtered and cleaned (Section **??**) from the *m*PDB to help elucidate patterns that could help differentiate metals and their metal-protein binding interactions from one another. The most basic objective was whether or not the curated data from the *m*PDB could discern between alkaline, transition, metalloids, and noble metals from one another.



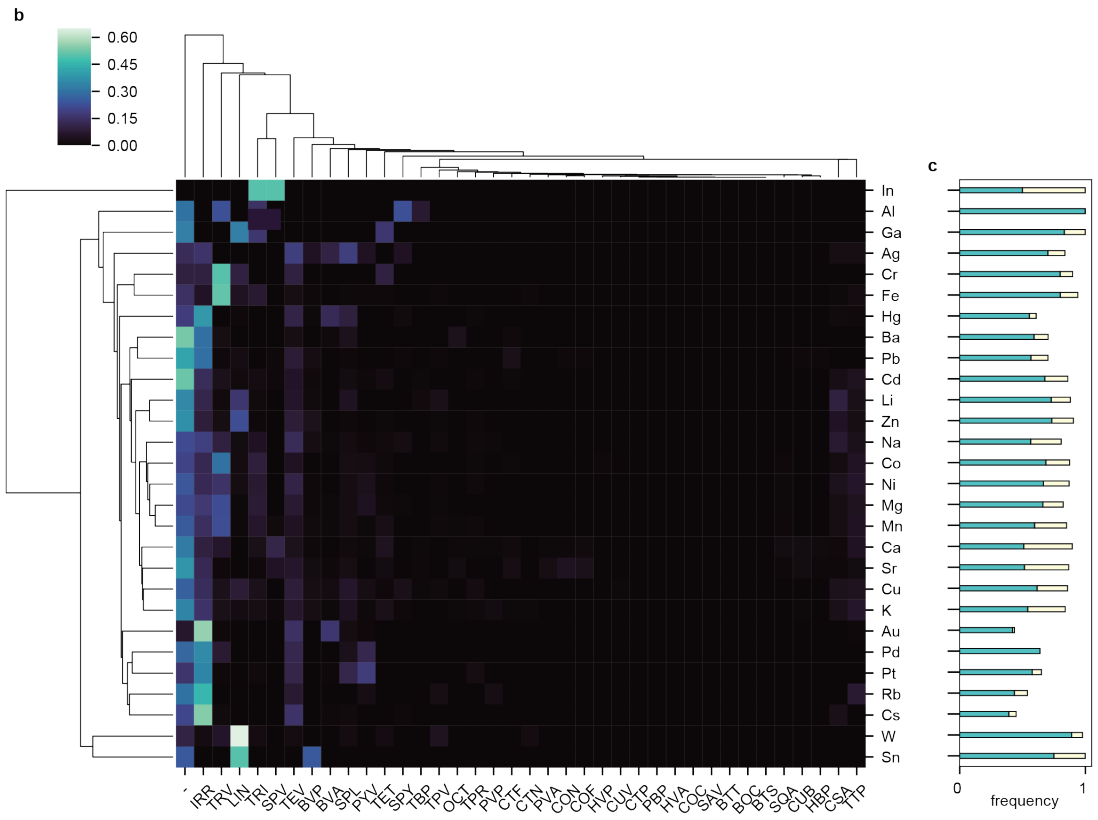


Figure 1.5 | Analyzing metal valency and geometry data to determine differences between metal groups (**a**) Histogram of valencies for each metal. Typically, most metals found biologically have valencies between 1–2; however, data from the *m*PDB suggest valencies above 2–, on average3. (**b**) Binned geometry occurances for each metal were clustered to identify any discrimination between metal groups. There were no distinct groupings observed. (**c**) Each geometry was either deemed regular, distorted, or unknown. For each metal, the annotation of each geometry was summed and plotted. The horizontal bar chart represent the frequency of regular (turquoise bar), and frequency of distored (beige bar) for each metal presented in the heatmap b). The remainder up to 1 represents the frequency of unknown.

Data on the metal's sterics, such as the metal's valency and geometry, were used for clustering analysis. Clusters were generated by using a single linkage method and euclidian distance as the distance metric. Unfortunately, there were no discernable groupings of alkaline metals from transition metals from metalloids, etc. (Figure **??**b). Overall, there was no consistent clustering pattern that could differentiate a metal-protein binding interaction using observations based on the steric environment alone.

Instead of analyzing the metal's sterics, the metal-binding ligands (i.e. amino acid residues) represented in each *m*PDB entry (Table **??** were used instead for clustering analysis. Although the ligand data lacked statistical power (many residue entries were zero), the clustering did show discrimination between certain metal groups (Figure **??**). In particular, the majority of alkaline and alkaline-earth metals were clustered together, and some of the transition, metalloid, and noble metals were segregated with statistical significance.

These preliminary findings suggest that it may be possible to systematically differentiate metals based on their amino acid-binding environment. Simply, a straightforward counting and binning of nearby binding residues was enough to superficially differentiate metals based on their periodic grouping (Figure **??**). To further this investigation, the data provided by the *m*PDB could be more feature rich if counts of neighboring residues in the metal binding environment were accounted for at varing distances away from the metal, and the same goes for nearby atoms. From here it would be possible to either construct a supervised or unsupervised machine learning model to predict the most common amino acid configuration away from the metal. With these models it may be possible to eventually develop autoencoders or generative models [**goodfellow2014generative**] to create de-novo metal-protein binding sites given robust predictions of the metal's nearest-neighbor data.

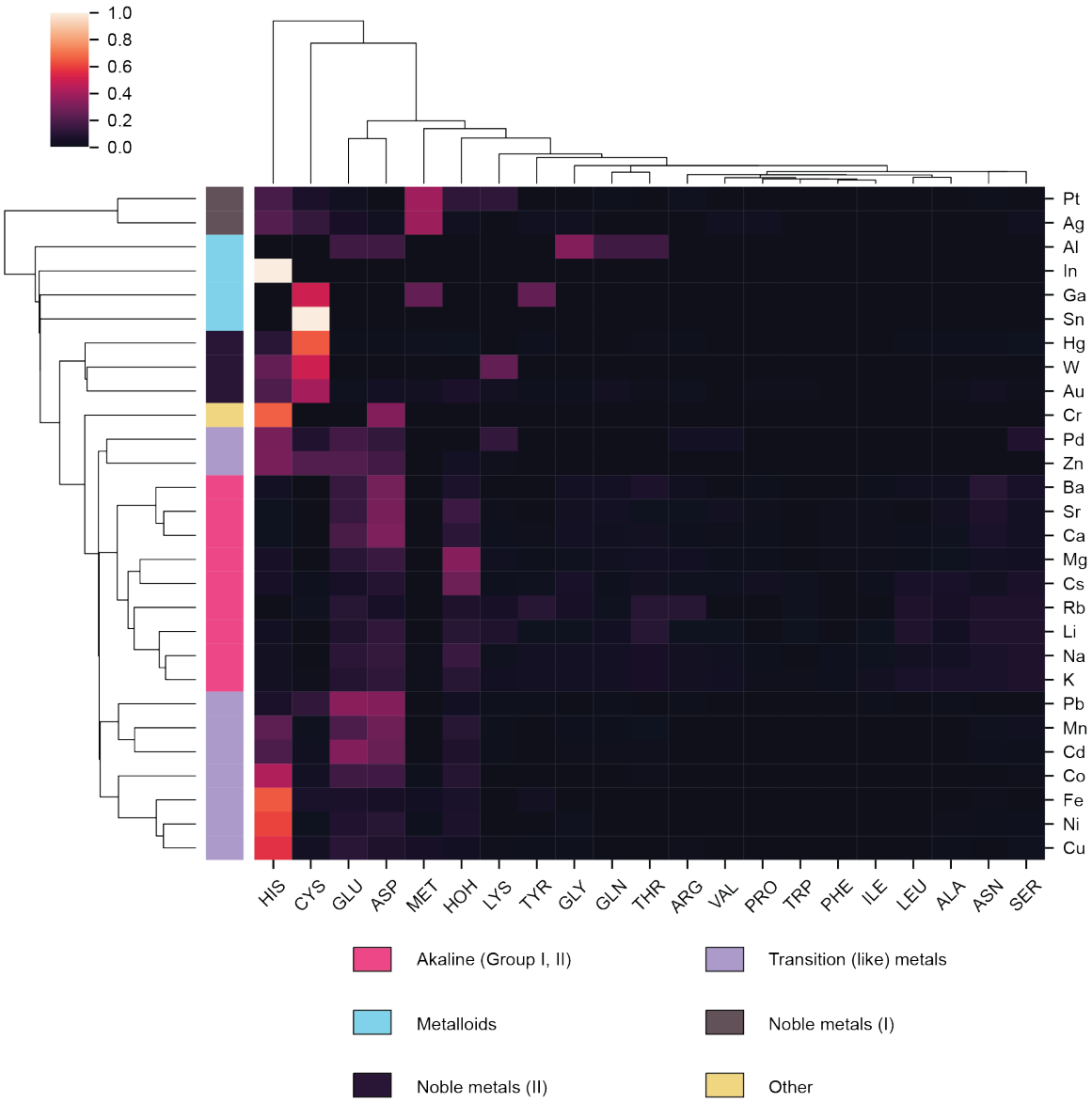


Figure 1.6 | Clustering metals based on their metal-binding ligands given data from the *m*PDB. Alkaline and alkaline-earth metals clustered together (pink rows), whereas some of the transition metals (light purple rows) were separate from the metalloid (turquoise rows) and noble metals (dark pink and brown rows).

1.3.3 Extracting and engineering features for better protein-metal discrimination

Preliminary results taken only from the *m*PDB annotations provided somewhat of a glimpse as to what featueres were important to differentiate metals from their protein structure. Information on sterics such as valency and binding geometry could not adequately cluster metals (Figure **??**); however, straightforward binning the frequency of occurance of residues nearest the metal did product discernible clusters. The next step was to make the binned nearest-neighbor residue data more feature rich. This meant to re-process the filtered data from the *m*PDB by fetching the entire protein structure from the PDB and using the modified nearest-neighbor algorithm (Figure **??**) to extract more granular data on the metal-binding environment.

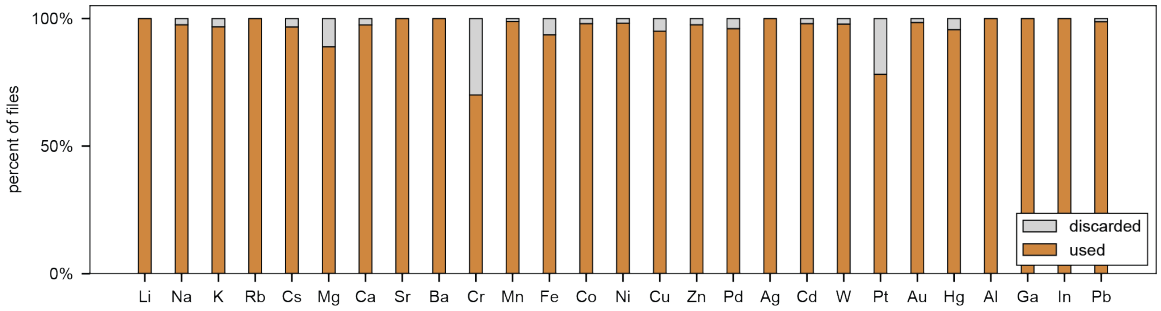
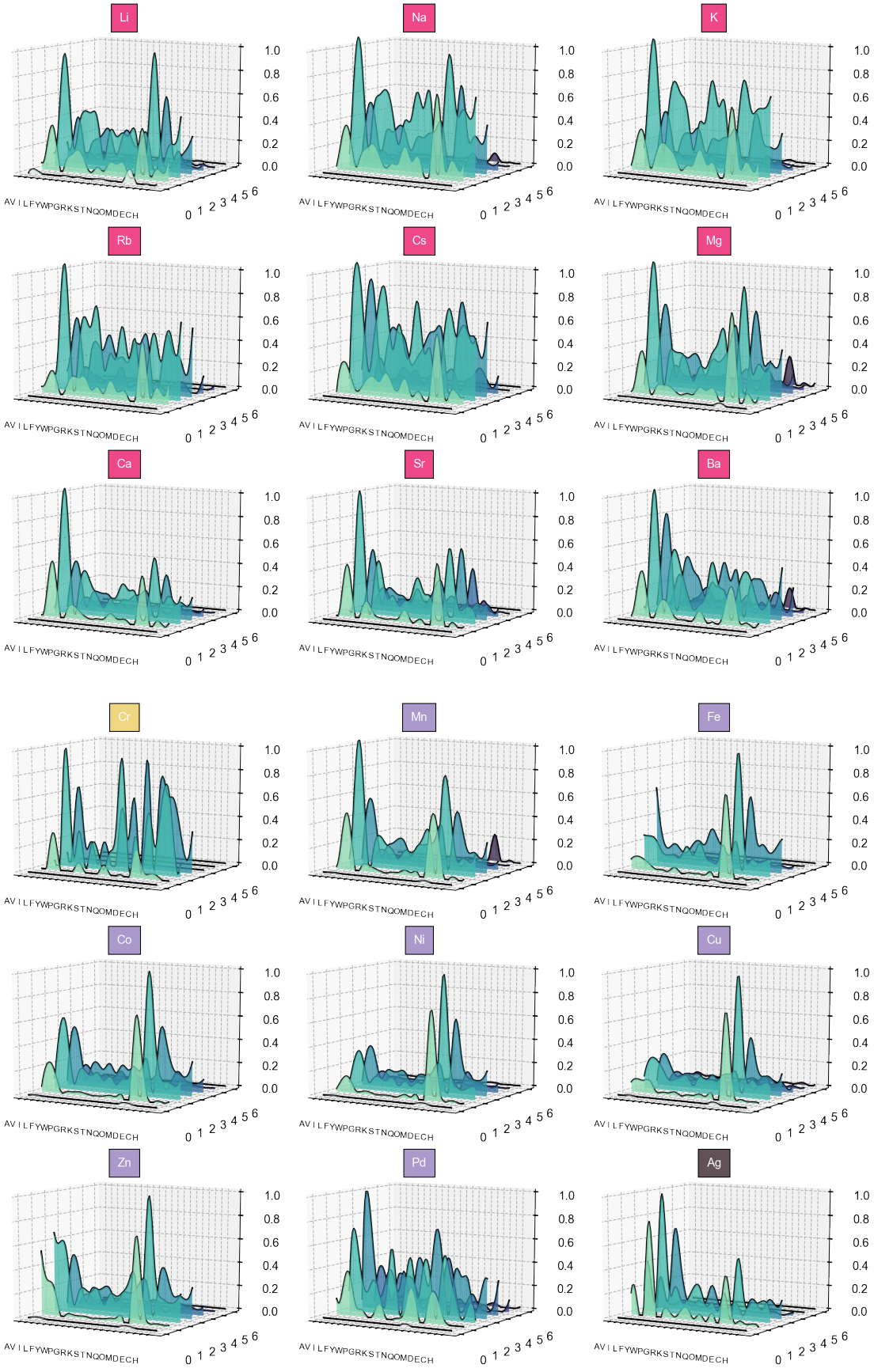


Figure 1.7 | Filtering metal PDB entries for incorrectly annotated or removed PDB entries. The chain letter and metal location were used to locate the metal center from the PDB entry. However, some annotations provided by the *m*PDB were incorrect and led to wrongly assigned metal IDs or non-existent locations.

Unfortunately, this additional step did eliminate more protein entries, as some of the annotations provided by the *m*PDB were incorrect, or some protein structures were out of date (additional filtering step was discussed in Figure **??**c,d). On average, approximately 2–10% of files were discarded when attempting to retreive protein structures from the PDB using file information taken from the *m*PDB (Figure **??**).

Figure 1.8 | Histogram of binned residue counts at 10 Åaway from the metal center. The frequency of occurance of residues encountered 10 Å away from the metal center. The frequency plot represents a "residue profile" of the most common amino acids present in the vicinity of the metal-binding environment. Metal coloring for each plot title corresponds to the cluster they belong to, analyzed in Figure **??**

From the usable files, the metal center of each PDB entry was located, and the nearest-neighbors were tabulated for a given radii away from the metal. At each radii, an array was constructed by tabulating the nearest atom, distance, and the same for residues and their distances away from the metal (the data structure format was shown in Table **??**).


Li
Na
K
Rb
Cs
Mg
Ca
Sr
Ba
Cr
Mn
Fe
Co
Ni
Cu
Zn
Pd
Ag
1.0
0.8
0.6
0.4
0.2
0.0
AV I L F Y W P G R K S T N Q O M D E C H
0 1 2 3 4 5 6

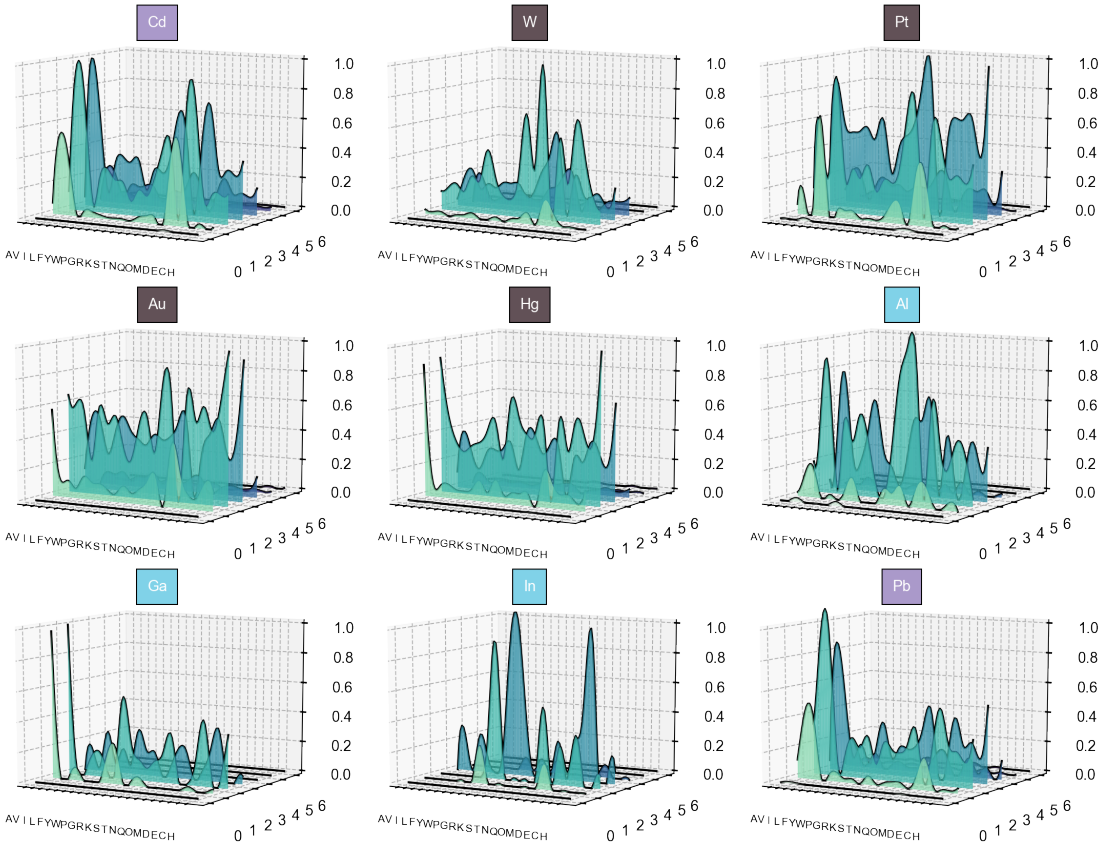


Figure 1.9 | 3-dimensional histogram of residue counts as a function of distance away from the metal center. x-axis (left most axis) are single code-letterings of amino acids. y-axis (right most axis) represent the distance in Å away from the metal center. z-axis (vertical axis) represent the percent occurance of the amino acids at the given distance away, noramlized to 1.

Each row represents a particular metal-protein interaction, and features were concatenations of atoms, residues, and their distances (in ascending order) away from the metal. Similar to counting the representation of ligands annotated by the *m*PDB, the occurances of residues were counted and binned at each radii to create stacks of 2-dimensional histograms (Figure **??**). These histograms represent a 'residue profile', in other words the statistical representation of residues encountered as one moves away from the metal center.

Histograms of each radii slice can be stacked together to generate a 3-dimensional plot of residues encountered (x-axis) versus distance (y-axis) versus frequency of encountered residue (z-axis) (Figure **??**).

Similarly, these same plots can be flattened to 2-dimensions by color encoding the z-axis as a heatmap (Figure **??**).

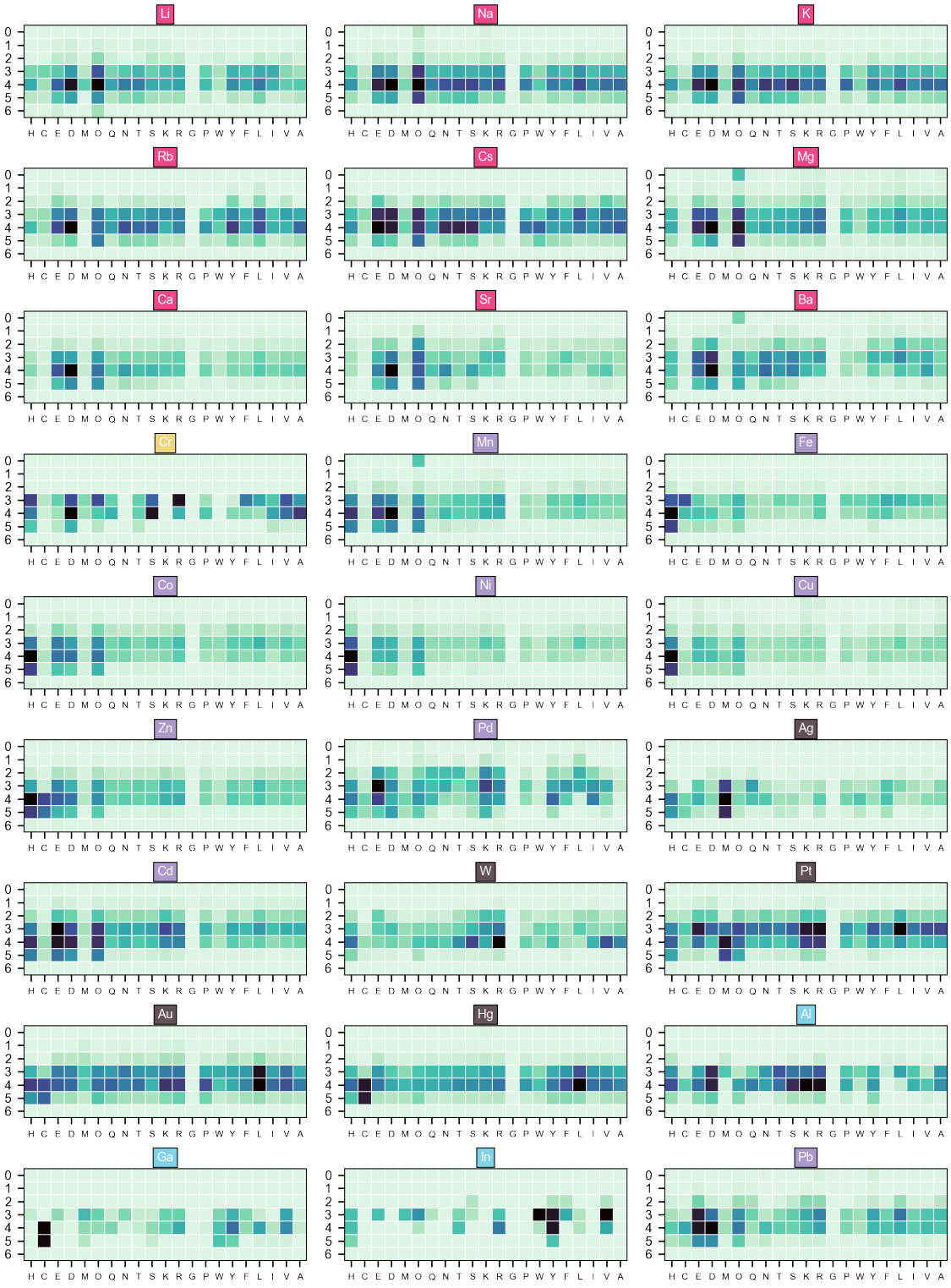


Figure 1.10 | 2-dimensional heatmap of residue counts as a function of distance away from the metal center

The 2-dimensional dataset for each metal can be further flattened to a 1-dimensional array with each row concatenated into a single vector.

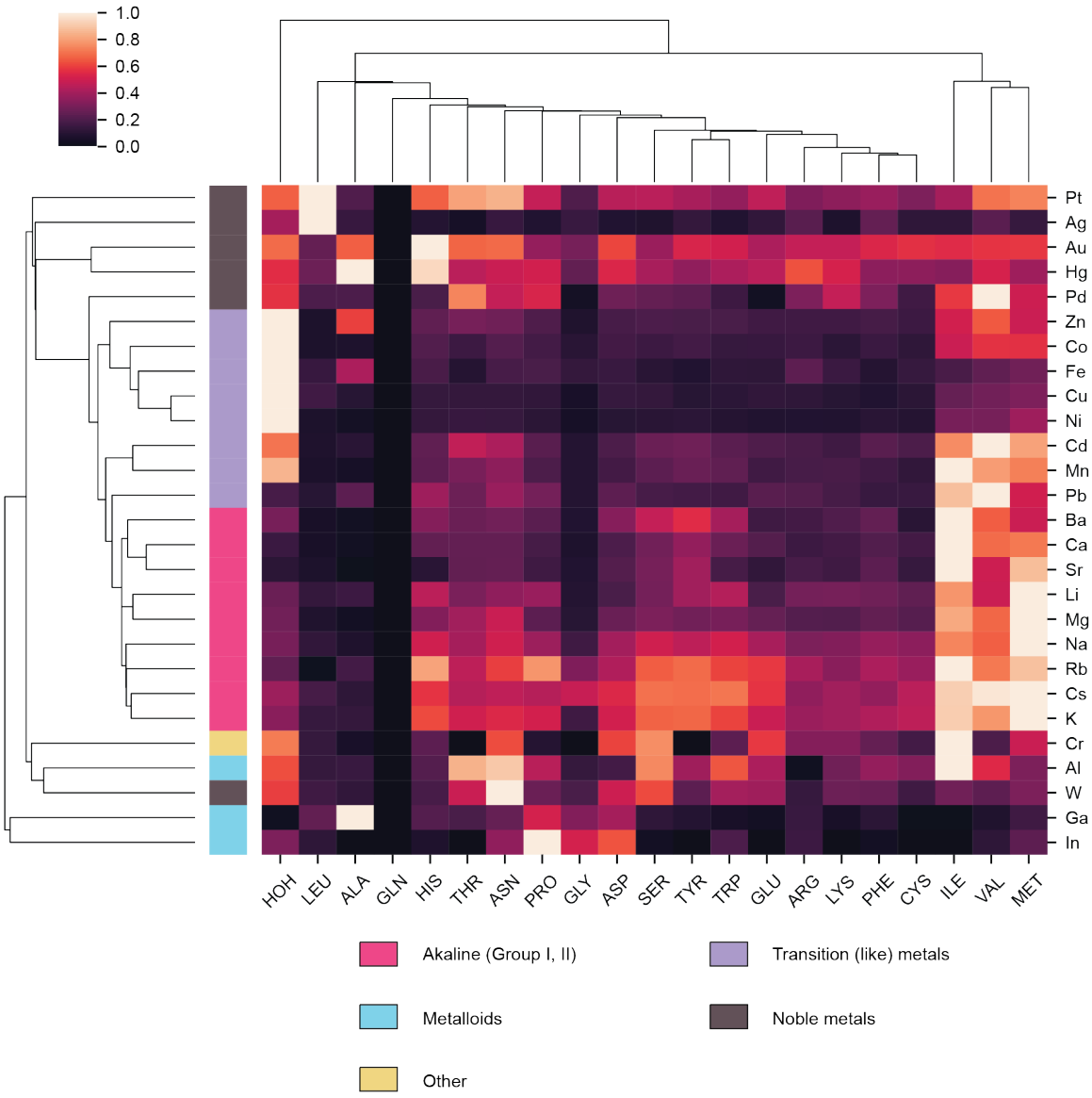


Figure 1.11 | Improved metal clustering using new datasets derived from the modified nearest-neighbor algorithm. Data on amino acid occurances as a function of distance away from the metal center provided a more data rich analysis compared to data provided from the *m*PDB (Figure **??**). In addition, the segregation of metal groups was more distinct, as each periodic grouping: alkaline/alkaline-earth metals (pink rows), transition metals (light purple rows), noble metals (brown rows), and metalloids (turquoise rows), were segregated from one another with statistical significance. The only outliers were W (tungsten), which grouped into the metalloid cluster, and Cr (chromium), which was distinguished as its own group.

With this, the same clustering analysis performed for Figure ?? could be done. The result is an improved segregation of metal groups given the more feature rich dataset (Figure ??).

1.4 Statistical analysis and machine learning; future work

Future work needs to be done to further investigate different feature extraction strategies and appropriate usage of statistical and machine learning models. This work is under current investigation and is under active research. The goal of this future work is to take the insights from the previous sections and machine learning models for predicitve or generative purposes. Examples include predicting the metal identity in unknown metal-binding pockets, or generating de-novo protein-metal binding environments.

Future steps to take include:

- **Further feature extraction and engineering**

  Only residue counts as a function of distance were used as features for the clustering analysis (Figure ??). Yet there are many features not used, such as the distance between metal and residues, and the same for the nearest atoms. Other features can be calculated such as the euclidean angle between metal and residues, or trigonometric parameters between residues in the same binding sphere. More sophisticated 3-dimensional analytics can be performed to solve for the sphere of hydration between the metal and resiudes, or the accessible surface topology of the metal binding pocket. In addition, metadata such as β-factor, percent occupancy, and other intrinsic data to the PDB structure can be used to weigh specific entries such that outliers are supressed during analysis.

- **Dimensionality reduction**

  Too many features add noise or unnecessarily increase computational time. In

most cases too many features may contribute to overfitting. Therefore, dimensional reduction techniques such as principal component analysis should be used to determine the most fundamental linear combinations of features that still fully represent the dataset. Dimensionality reduction can also project the dataset onto another plane that better discriminates instances; for example, projecting a cone from the top to form can separate the heights of the cone on a 2-dimensional plane. However, a poor projection from the side would overlap the surfaces of the cone into a triangle. So testing a variety of dimensional reduction techniques should be checked before proceeding to training machine models .

- **Experiments with a variety of classifier models**

  There exist numerous machine learning models, many with their own benefits and drawbacks. Each model should be tested empirically to determine which performs the best. For this purpose classifiers would be appropriate, as the model should be able to classify which arrangement of residues have the highest probability of containing a certain metal. Examples of classifiers are decision trees or random forrests.

- **Hyperparameter tuning using gridsearch**

  Mostly all models do not work "out of the box" and need fine tuning to adjust for the nuances of a particular dataset. These parameters, such as rate of learning, variable weights, and so on, are called hyperparameters. Often, these hyperparameters are brute-forced optimized through grid searching, where several hyperparameters are permutated for a single training run, and performed repeatedly over the entire combinatorial space of hyperparameters until a value is selected that optimizes the training output, or reduces the error.

- **Creating pipelines for ensemble learning**

  Several algorithms can work in conjunction to provide better modeling and decision power. Examples include large or convulational neural networks, algorithms sub-classified as deep learning algorithms. With these new algorithms,

the pipeline of feature engineering, dimensionality reduction, and hyperparameter tuning may need to be further optimized.

One overaching concern is whether the quantity and quality of PDB entries is enough to robustly train a model to accurately predict metal-binding interactions. So far there are 151,754 protein structures in the PDB[6]. However, some are redundant, and the quality varies dramatically between entires. It actually may be too early, or ambitious, to study every protein structure as the data is not yet sufficient to train a suitable machine model. Another setback may be the difficulty to extract useful features from a complex 3-dimensional crystal structure, and that algorithms have not yet been developed. However, the work so far suggest that it may be possible to cluster groups of similar metals together, and that may be enough to help create custom peptide/proteins with affinities for alkaline and alkaline-earth metal from the transition, noble, and metalloids.

[6]statistics on the PDB can be found here: `https://www.rcsb.org/stats`