Title
From proteins, to machines, to protons, to genes, and back again

Author
Fraga, Keith Jeffrey

Publication Date
2022

Peer reviewed|Thesis/dissertation

From proteins, to machines, to protons, to genes, and back again

By

KEITH JEFFREY FRAGA
DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Molecular and Cellular Biology

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

_____
Ian Korf, Chair

_____
Sharon Aviran

_____
Enoch Baldwin

Committee in Charge

2022

Acknowledgements and Dedication

There are many individuals I need to thank who have helped me in my journey through my PhD process. First, I want to acknowledge my wife, Treasure Warren, for supporting me through my education all these years. I also want to acknowledge my brother and sister-in-law, Chad Fraga and Sabina Simon for your love and support you have given me. Thank you to my mother and father, Judy and Jeff Fraga, for supporting me and growing with me.

Next, I want to acknowledge my committee members, Enoch Baldwin and Sharon Aviran, for their unwavering dedication and compassionate support of me as a student. Thank you. Thank you to Wolf Dietrich Heyer and Javier Arsuaga for their support and all I learned from them. Next, I want to acknowledge Sean Burgess and Chris Fraser for advising me in critical points of my graduate career. I want to acknowledge Krystof Fidelis for sponsoring my attendance at CASP13, which had a critical impact on my career. I also want to thank Gaetano Montelione for our productive collaboration in NMR data management. Much thanks to Vladimir Yarov-Yarovoy for very productive discussions and growing vibrant computational structural biology community on campus.

I have had the privilege to work with and learn from some amazing peers and colleagues throughout my time in the BMCDB graduate group. I want to acknowledge (in no particular order) Helen Lamb, Ryan Polischuk, Robert Stolz, Tamara Christani, Osman Sharifi, Hannah Lyman, Aiyana Emigh, Emily Cartwright, Anna Feitzinger, Natasha Mariano, and Amalia Karesh. There are certainly too many to list, but thank you to so many in the UC Davis graduate student community for helping me feel welcomed and that I can finish this dissertation.

Lastly, certainly not least, I want to acknowledge my PhD advisor Ian Korf for mentoring me and guiding me through my studies. Your support from quite literally the first time I visited UC Davis on recruitment to now, and I know beyond, is something I don’t take for granted and forever thankful.

I also want to dedicate my dissertation to the UC Davis Counseling Center and staff, without which I am not sure I would have completed my graduate studies at UC Davis.

Abstract

The success of data standards and public databases in biology is the foundation for the current and continued success of machine learning in biology and medicine. This dissertation explores the interactions between biology, computers, and people in order to develop novel machine learning methods to model complex biological problems. Data is one of the main resources to do machine learning, and Chapters 1, 2, 3 are explicitly about data organization and quality assurance in the protein Nuclear Magnetic Resonance (NMR) spectroscopy discipline. Chapters 4 and 5 present new machine learning architectures to address learning tasks in genomic site recognition and NMR chemical shift prediction. Chapter 1 investigates the manner protein NMR chemical shift data is deposited at the Biological Magnetic Resonance Bank (BMRB) in order to build simple table look-up models to estimate protein chemical shifts. In Chapter 1, we find there is low sequence diversity and data redundancy in the BMRB that was a challenge to locate and filter out. Without filtering out BMRB entries with the same sequence, and possibly the same chemical shifts, look-up models will be more accurate due to data contamination in training and testing sets. Chapter 2 examines approaches to curate a large protein sample production and NMR database to create an NMR time-domain dataset. Quality assurance tests in this NMR sample/FID database uncovered data collisions and redundancies among the database records, which motivated the development of new NMR database management tools. Chapter 3 presents a relational database schema to archive protein NMR samples and associated time-domain data called *SpecDB*. *SpecDB* is open source and available at https://github.rpi.edu/RPIBioinformatics/SpecDB.git. Chapter 4 explores how deep neural networks can recognize genomic splice acceptor and donor sites from sequence alone, achieving 97% accuracy for highly used splice donor sites. Chapter 4 also investigates neural networks for intron/exon sequence classification, maximally reaching 77% accuracy. Chapter 5 presents the application of marginalized graph kernels to prediction of NMR chemical shifts for small organic molecules. Incorporating chemical descriptors to graph kernels reaches a 3.501 ppm mean absolute error for Carbon chemical shifts. In total, the following five dissertation chapters explore work in data integrity, organization, and learning techniques from data for applications to structural biology problems.

Table of Contents

1. Toolkits in NMR chemical shift dataset generation for machine learning

1.1 Introduction

The chemical shift is the “milepost” in NMR and is a critical measurement for the structural and dynamic studies of proteins(Berjanskii and Wishart, 2017). The chemical shift is derived from an atom’s nuclear resonant frequency in an external magnetic field(Case, 2013). The chemical shift is a relative frequency measure, where an atom’s chemical shift is measured relative to an internal or external standard atomic resonant frequency(Case, 2013). In protein NMR, chemical shifts are often utilized as “ledger entries” to track NOEs (nuclear Overhauser effects) as distance restraints from measured NOEs are a major source of structural information to build NMR structural models(Berjanskii and Wishart, 2017). Chemical shifts also can elucidate other structural and dynamic properties of biomolecules(Berjanskii and Wishart, 2017), such as identifying protein second structure segments(Marsh et al., 2006; Shen and Bax, 2012; Wishart and Sykes, 1994), estimate torsion angles(Berjanskii et al., 2006; Cheung et al., 2010; Shen and Bax, 2013), determine solvent accessible surface area(Hafsa et al., 2015), and local chain flexibility(Berjanskii and Wishart, 2013, 2008). Chemical shifts can also be used to refine molecular models(Berjanskii et al., 2015).

The NMR community has a rich history of developing and applying machine learning at all stages in experimental and structure-generation pipelines(Cobas, 2020; Hoch, 2019). Specifically, prediction of NMR chemical shifts has been particularly successful with numerous methods developed to predict NMR chemical shifts, particularly for proteins. For protein chemical shift prediction, the task is typically to predict chemical shifts given an input atomic structure and/or from sequence information, like sequence similarity to proteins with known chemical shifts. There are three categories of protein chemical shift prediction methods, empirical(Han et al., 2011; Kohlhoff et al., 2009; Li et al., 2015, 2020; Meiler, 2003; Shen and Bax, 2010; Zeng et al., 2013), ab initio(A. Bratholm and H. Jensen, 2017; Berjanskii et al., 2015; Moon and Case, 2007; Vila et al., 2009), and molecular dynamics(Lehtivarjo et al., 2009; Markwick et al., 2010; Tian et al., 2012). It is generally considered that empirical methods produce more accurate shift predictions, with ab initio and MD methods generally less accurate but promising. Learning from available experimental chemical shifts, protein sequences and structures remains a significant goal and challenge in the NMR community.

The Biological Magnetic Resonance Bank (BMRB) is the central public repository archiving magnetic resonance data for a vast array of biomolecular studies(Romero et al., 2020). The organizing goals of the BMRB is to archive magnetic resonance data from scientific studies investigating molecular systems from proteins to complex metabolomic samples. The BMRB is also a source of molecular structures derived from magnetic resonance data.

The NMR chemical shift is the most abundant data type deposited and archived by the BMRB, with over 10 million chemical shifts deposited across over 13,000 BMRB entries. BMRB entries are typically studies that either resulted in a peer-reviewed study and/or a molecular structure determined. A BMRB study may have multiple proteins and samples with many sample conditions and many types of NMR data restraints collected. The BMRB utilizes a relational schema system to curate and archive NMR data, samples, and experiments called NMR-STAR (**S**elf-defining **T**ext **A**rchival and **R**etrieval)(Ulrich et al., 2019). NMR-STAR has nearly 6,500 tags/attributes that are organized to describe an NMR study, experiment, and data(Ulrich et al., 2019, p.). Chemical shift lists deposited at the BMRB are archived in NMR-STAR format, providing a human-readable and interchangeable format to store assigned chemical shifts.

Chemical shift data deposited at the BMRB has been crucial for dataset development for many empirical chemical shift prediction methods. The RefDB database is a dataset of protein structures and chemical shifts derived from the BMRB, and RefDB is a commonly used dataset for chemical shift prediction development(Zhang et al., 2003). The RefDB database's specific purpose is to address the issue of chemical shift referencing errors present in an estimated 20%-40% of NMR studies. There remains many more chemical shifts available, however, in the BMRB to use for learning. Data is an essential resource for machine learning, and for new advancements in biomolecular NMR chemical shift prediction new datasets are needed.

Datasets for chemical shift prediction have either relied on RefDB, or on individually curated datasets. Differences in datasets have been demonstrated to lead to differences in performance, particularly in the case between ShifX2 and UCBShift(Li et al., 2020). In other machine learning domains, technical and performance progress is partly a result of standard datasets(Goodfellow et al., 2016; Halevy et al., 2009; “Opportunities and obstacles for deep learning in biology and medicine | Journal of The Royal Society Interface,” n.d.). These trends of dataset construction are also being applied in the life sciences, particularly in protein structure prediction(AlQuraishi, 2019).

As the Biological Magnetic Resonance Bank (BMRB) represents the central resource of chemical shift data, the purpose of this study is to develop a pipeline to harvest molecule information and chemical shift assignments from studies deposited at the BMRB. As a proof-of-concept for our pipeline, we developed simple look-up models to demonstrate the usage of the pipeline and to test the performance of look-up models using filtered chemical shift data from the BMRB.

1.2 Methods

Our approach was to collect the chemical shift lists from BMRB entries into one file to work from. Figure 1.1 illustrates the funnel approach that was used to collect the BMRB entries for this study. We were able to collaborate with the BMRB to provide us with a list of BMRB entries that was used to generate the summary shift statistics on their website, and used those BMRB entries. The characteristics of these entries were of the following:

1. No aromatic or paramagnetic ligand(s)
2. No chemical shift outside of eight standard deviations of average calculated from full BMRB database
3. A chemical shift of a carbon bound proton smaller than -2.5 ppm or greater than 10 ppm

After the BMRB entries are collected and the chemical shifts are gathered, the data is organized into a single JSON file formatted as indicated in Figure 1.1. Each BMRB entry is a separate record in the JSON file, and the chemical shifts from the entry are split across the different atom types in the protein. Each atom type-specific chemical shift list is equal to the length of the protein sequence, and where no chemical shift existed for a particular atom type and residue, a None placeholder was used. In this way the complete chemical shift data of a protein can be captured.

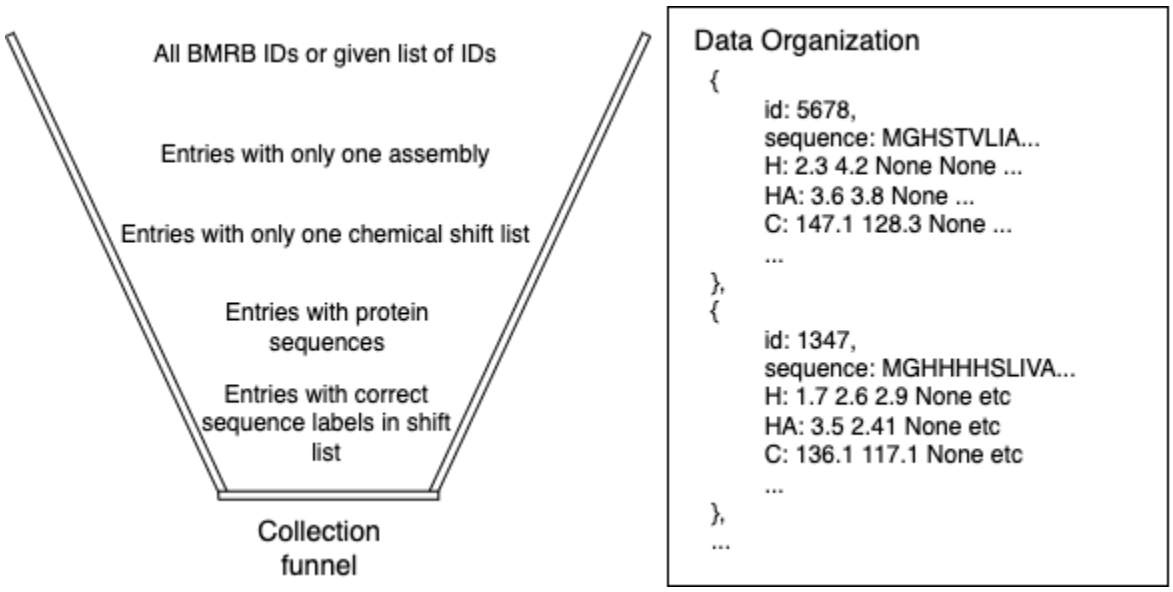

Fig. 1.1: Data collection funnel and data organization after collection

Funnel to collect and organize BMRB entries and their associated chemical shift lists into a usable data format. We were provided a list of BMRB entry IDs that were used to create the chemical shift statistics tables on the BMRB website to build our dataset. We only could process entries with one assembly, one chemical shift list, and entity per assembly. We threw out entries that were not proteins, and entries that had errors in their chemical shift lists.

We performed sequence identity filtering to further filter the BMRB entries collected. We used BLAST to align all protein sequences from the BMRB entries together, then used single linkage clustering to remove entries with percent sequence identity above a user-defined threshold. In the case where no sequence filtering is performed, we found cases where different BMRB entries had the same exact chemical shifts by simple visual inspection. We performed an exhaustive search that performed all pairwise BMRB entry comparisons to find pairs of entries with the same underlying chemical shifts. We built simple table look-up models for chemical shift prediction using the unfiltered and filtered data. Table look-up models are described in Figure 1.2.

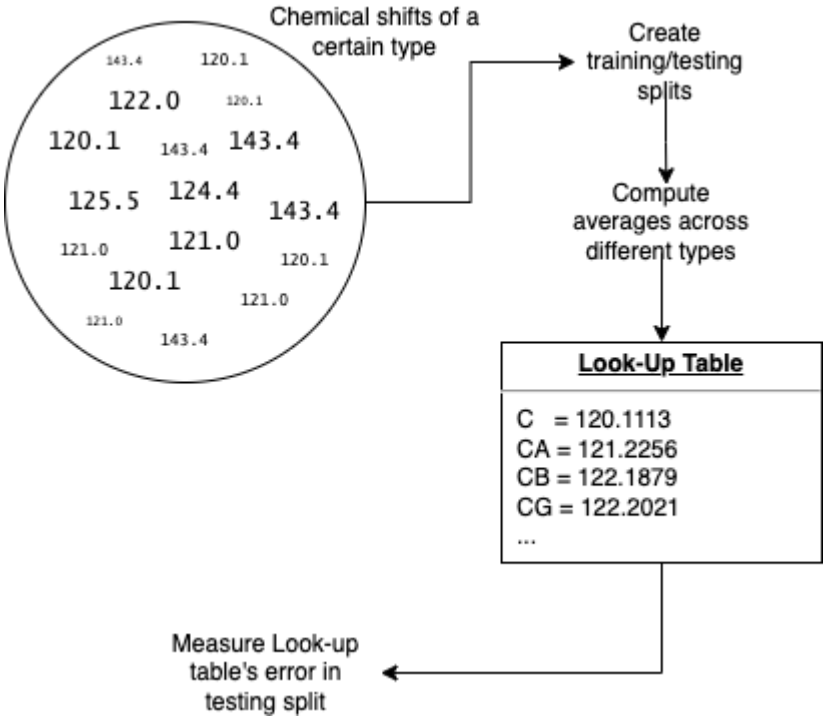

Fig. 1.2: Pipeline for building and testing chemical shift look-up tables

A set of chemical shifts are split into training and testing using cross validation. Then, the chemical shifts are broken into different classes/types. In the example shown in this figure, carbon chemical shifts are classified into atom types groups, for example CA, CB, etc. The average chemical shifts within each class constructs the look-up model.

1.3 Results

The first observation we made from analyzing the entries we used to create the chemical shift dataset was there is low sequence diversity in BMRB. Table 1.1 illustrates the low sequence diversity in this dataset. Out of an initial set of 9,458 entries, there are 7,923 unique sequences. Then performing a 90% sequence identity filtering drops the entries down to 5,646, a ~40% reduction.

Low sequence diversity will have an effect on chemical shift statistics and amino acid k-mer statistics as illustrated in Table 1.1. However, closer inspection of the BMRB entries with shared sequences illustrated that not only were the amino acid sequences the same, but also the chemical shifts themselves. We performed an exhaustive all-versus-all comparison for every entry and chemical shift to find cases where two entries had the same exact reported chemical shift values. Out of the 9,458 initial entries, there were 1,376 cases of two entries having the same chemical shift information for a particular amino acid atom type. This translates to 143 pairs of entries with at least one atom type having the same chemical shift data. These data collisions indicate BMRB entries that are copies with some unknown differences, but at the reported chemical shift level they are the same. Sequence identity filtering past 90% removes these collisions as they occur where entries share the same sequence.

Percent Identity	Total entries	Unique Sequences	Total shifts	Number of colliding dataset pairs	Number of Unique 3-mers	Number of Unique 5-mers
None	9,458	7,923	6,691,778	1,376	7,993	454,658
90%	5,646	N/A	4,567,284	0	7,993	421,521
80%	5,2866	N/A	4,331,099	0	7,993	408,866
50%	3,758	N/A	3,016,922	6	7,983	303,989
10%	2,105	N/A	1,354,777	6	7,902	145,226

Table 1.1: Protein sequence and data collisions statistics on the BMRB dataset

We collected 9,458 BMRB entries that passed our criteria. We then searched for entries with duplicate chemical shift shifts at different sequence identity thinnings of the original dataset. The pairwise sequence alignments were precomputed using BLAST. We also record the number of unique 3/5-mers in each slice of the dataset to indicate the effect of low sequence diversity on the observable k-mers in the dataset.

With the presence of data collisions in the data set, we wanted to test the performance of simple look-up models with/without sequence identity filtering. There are four levels of look-up models. (1) chemical shifts are organized by atomic element, i.e all nitrogen shifts grouped together, all carbon shifts grouped together, and all proton shifts grouped together. (2) chemical shifts are organized by the specific atom type they are from, i.e all backbone amide nitrogens grouped together, backbone amide protons grouped together, CA shifts grouped together, etc. (3) chemical shifts are grouped by the atom type and identity of the amino acid they are from. (4) chemical shifts are grouped by the atom-type and the amino acid 3-mer they are a part of, which takes into account local sequence content. We evaluate the look-up models using 10-fold cross validation with three different slices of the chemical shift dataset: no sequence identity filtering, 80%, and 50%. The performances are recorded in Table 1.2. In the vast majority of cases, if no sequence filtering is performed the look-up models perform better as measured by lower root-mean square errors (RMSEs) than if sequence filtering is used.

Model type	Unfiltered BMRB	80% sequence identity filtering	50% sequence identity filtering
element	N 5.2362 H 4.4809 HA 0.7272 CA 18.5593 CB 38.9796	N 5.2293 H 4.5779 HA 0.8060 CA 18.2486 CB 36.6648	N 5.2034 H 4.5922 HA 0.8183 CA 17.8782 CB 38.6072
Atom type	N 5.2256 H 0.6212 HA 0.4810 CA 4.8825 CB 12.8111	N 5.2170 H 0.6267 HA 0.4834 CA 4.8499 CB 12.7664	N 5.1908 H 0.6160 HA 0.4807 CA 4.8372 CB 12.7499
Atom type + residue	N 3.8523 H 0.6189 HA 0.4522 CA 2.2053 CB 1.9705	N 3.8839 H 0.6244 HA 0.4559 CA 2.2167 CB 1.9848	N 3.8433 H 0.6138 HA 0.4530 CA 2.2153 CB 1.9835
Atom type + residue + 1 flanking residues	N ***3.5614*** H ***0.6018*** HA ***0.4384*** CA ***2.0896*** CB ***1.9429***	N 3.6379 H 0.6158 HA 0.4483 CA 2.2167 CB 1.9843	N 3.6221 H 0.6097 HA 0.4499 CA 2.1399 CB 2.0043

Table 1.2: Results for look-models across different data set filtering strategies

We designed four different look-up models that model chemical shifts using different levels of information, from the element type all the way to local sequence identity. We report the 10-fold cross validation RMSE of look-up models on three different slices of the dataset, no sequence filtering (with data collisions), and 80%/50% sequence identity filtering (no data collisions). The model with atom type plus 1 flanking residue performed the best, and the data set with no sequence filtering had the best performance between the other two sequence filtering.

1.4 Discussion

The BMRB is the central repository for magnetic resonance experiments of biomolecules. The BMRB is a critical resource for data mining and machine learning efforts to improve the prediction and usage of NMR chemical shifts in biomolecular NMR experimental and structural studies. The BMRB reports statistics on chemical shifts deposited in the repository, and chemical statistics are used in many applications, like in the prediction/simulation of HSQC spectra, and knowledge-based approaches to NMR sequential assignment. In this study we attempted to organize the chemical shift data in the BMRB into a dataset to do machine learning. We found that there are challenges to using the chemical shift data deposited at the BMRB. A principle finding of this study is that the BMRB is a data resource/repository, not a ready-made dataset for data mining or machine learning. This is due to the large number of data collisions present in the BMRB, and without careful inspection, data redundancy can pollute training and testing datasets used in machine learning projects. Table 1.2 demonstrates how without filtering for data redundancy and sequence identity, look-up models will consistently perform better than look-up models with sequence identity filtering.

It is difficult to diagnose the source of the data collisions we encountered in this project. Perhaps the most prevalent reason why data collisions occur is that experimentalists will often deposit the same experimental data for a project twice (or more) into the BMRB with slight modifications. Sometimes researchers deposit assigned chemical shifts, or partial assignments ahead of a study's publication for sharing with colleagues. However, upon publication, the study is re-deposited as a separate BMRB entry. This

does not always happen, and further, the BMRB has processes to handle cases to update a previous entry submission. Yet, from the work presented here, these collisions happen often enough to make using the BMRB ‘out-of-the-box’ difficult. Sequence identity filtering below 90% removes these collisions obviously, and going forward we recommend this be a common practice for using protein chemical shift data from the BMRB.

The low sequence diversity of entries deposited at the BMRB also highlights the need and utility of structural genomic efforts that can grow the sequence diversity of publicly available databases. The Northeast Structural Genomics Consortium had this exact goal, with target selection partly influenced by a target's ability to serve as a homolog for other unknown protein structures in the human genome and other genomes(Wunderlich et al., 2004).

There are two limitations to the study presented in this chapter. First, no chemical shift re-referencing was implemented in the chemical shifts collected. It is estimated between 20%-40% of chemical shifts in the BMRB are mis-referenced(Zhang et al., 2003). Re-referencing is possible if it is known what the original reference was for a chemical shift dataset. In future efforts, we will explore how to incorporate chemical shift reference information from the BMRB entry into a broad-based re-referencing protocol. Second, we only considered easily parse-able BMRB entries, i.e entries with one assembly, one entity, and only protein. In the future, we will more fully use the NMR-STAR parser to incorporate more complex BMRB entries into our dataset pipelines.

Another limitation and future direction is correlation of chemical shift differences to experimental sample conditions. Sample conditions are recorded in the BMRB entry, and this information can be used to learn how changes to conditions change chemical shift measurements. This is an area where the low sequence diversity of BMRB entries is an advantage because there are different NMR experiments on the same protein, and these cases can be useful case studies for how differences (or lack thereof) in sample conditions influence changes in chemical shift measurements.

A final limitation and future direction is the addition of structural information into the chemical shift data set. A subset of BMRB entries with chemical shifts have deposited structures, or the BMRB entry is linked to a PDB ID. These atomic structural models can be harvested together to have a combined chemical shift and structural model data set.

1.5 Conclusion

In this study we collected chemical shifts from BMRB entries into a data format amenable for machine learning and data mining. Inspection of collected chemical shift lists found data collisions, where two different BMRB entries had the same chemical shifts. We implemented an all-versus-all comparison across all the BMRB entries we used for our study and found 143 pairs of BMRB entries with chemical shift data collisions. Filtering for sequence identity removes these collisions. Additionally, building simple look-up models with un-filtered data shows better RMSE performance than models built from sequence identity filtered datasets. In future directions we will work to

incorporate sample condition information into the dataset of chemical shifts, as well atomic structural models. Future use of chemical shift statistics from the BMRB should include pre-filtering for sequence identity to remove the data collisions observed in this study.

2. Quality Assessment in the SPINE database

2.1 Introduction

The success of machine learning in structural biology over the past 10 years, particularly deep learning in the field of protein structure prediction("Artificial intelligence in structural biology is here to stay," 2021; Baek et al., 2021; Jumper et al., 2021; Kryshtafovych et al., 2021), is highlighting new areas where machine learning can aid the discovery of biomolecular structures and motions. Protein **N**uclear **M**agnetic **R**esonance (NMR) spectroscopy is a central experimental tool in the structural biologists toolkit to measure structural and dynamic information of biomolecules. NMR spectroscopy has a rich history of developing and applying machine learning at all stages in the experimental pipeline(Cobas, 2020; Hoch, 2019), from predicting protein torsion angles(Shen and Bax, 2015), chemical shift prediction(Li et al., 2020), NMR spectral peak picking(Klukowski et al., 2018; Li et al., 2021), and reconstruction of non-uniformly sampled free induction decays (FIDs)(Karunanithy and Hansen, 2021; Luo et al., 2020; Qu et al., 2020).

The free induction decay collected from the NMR spectrometer is the raw data that all proceeding steps in NMR experimental pipelines rely and build on(Wuthrich, 1986). The terms *time domain data* and *free induction decay* (FID) data are used interchangeably in the community for these raw data. The prospect of automatic analysis of FIDs to produce NMR resonance assignments, dynamic information, or even molecular

structures, is a long-standing goal and challenge in the NMR spectroscopy field. A large set of curated time domain datasets is a critical first step to support such applications. Biomolecular NMR time domain datasets are archived in the **B**iological **M**agnetic **R**esonance **B**ank (BMRB)(Romero et al., 2020). However, only a small percentage of BMRB entries have associated time domain data. Yet, the NMR community as a whole has generated perhaps 100,000s of FIDs over the course of the field's history. In particular, there have been large structural genomic projects that generated large amounts of NMR data on diverse protein targets, like the **N**orth**E**ast **S**tructural **G**enomics (NESG) consortium that collected X-ray and NMR data for hundreds of novel protein targets(Wunderlich et al., 2004). The advantage with the NESG project was the development and use of a dedicated **L**aboratory **I**nformation **M**anagement **S**ystem (LIMS) to record and track the progress of structural genomic targets through screening and structure determination pipelines called SPINE (**S**tructural **P**roteomics **i**n the **N**orth**E**ast). The goal of this chapter is to assess the feasibility of harvesting HSQC screening FIDs into a dataset for future data mining and machine learning. Previous efforts to build deep neural networks for FID reconstruction, for example, relied on synthetically made FIDs for neural network training(Karunanithy and Hansen, 2021; Luo et al., 2020). Yet, real FIDs do exist in abundance, and we are making use of the database infrastructure in SPINE to explore how to bring real FIDs into a dataset.

2.2 Methods and Results

We were provided a nested hierarchy of folders from the SPINE database that represented all the HSQC screening data in SPINE. A major effort in SPINE was to track the progress of sample preparation for the structural genomic targets in the NESG. These targets had a specific history, starting with the protein/gene the target was of, then to the specific construct or domain of the target to be studied, then to the molecular sample which was given **P**rotein **S**ample **T**ube (PST) ID. The hierarchy of folders provided from SPINE reflected this target, construct, PST hierarchy.

We desired to collect all the saved FIDs from these PSTs and store them in a database that stored very minimal metadata about the sample and ultimately the FID. We computed the MD5 hashing value of the FIDs to be inserted to use as unique identifiers for the data being inserted into the new database. This was done to prevent insertions of the same FID into the database multiple times. Having duplicate FID records in our dataset would hinder the machine learning projects downstream. However, we found that most of the data from SPINE broke this constraint; a great deal of FIDs in the SPINE set we received had the same MD5 hash key, extremely rare to happen by chance, suggesting duplicates exits across the database. Figure 2.1 illustrates the plan and structure of the MD5 analysis we choose for this project.

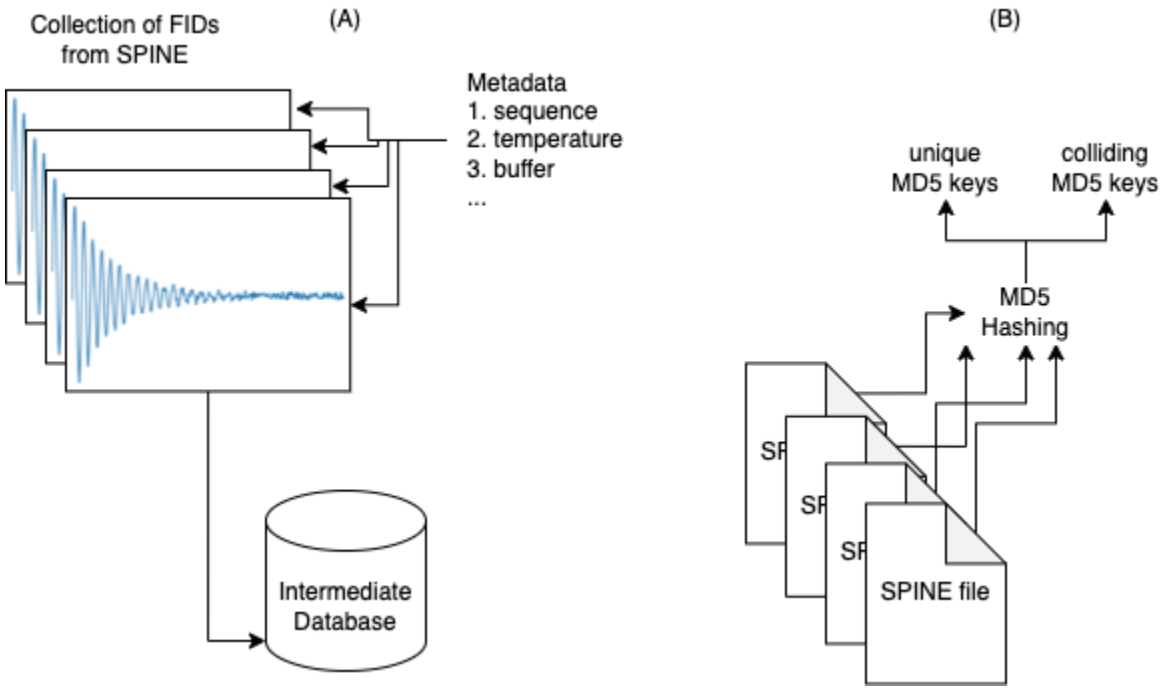

Fig 2.1: Overview of data collection and MD5 analysis in SPINE

(A) We took the collection of FIDs and their associated metadata and inserted it into an intermediate database we can use for downstream data mining and machine learning. Importantly, the intermediate database had a constraint that no two FIDs can be inserted with the same MD5 hash key. (B) In the MD5 analysis, we can take the MD5 key of every files from SPINE (including all the NMR experimental acquisition files), and sorted the MD5 keys into those that were unique and those that were not.

We decided to perform a complete MD5 analysis of every file we received to see where collisions occurred. Table 2.1 is a summary of that analysis. We found 97,603 unique MD5 keys out of an initial set of 342,546 files. There are two file types that hold time domain data, *fid* and *ser*. Usually, 1-dimensional NMR experiments are recorded in *fid* files, 2-dimensional experiments in *ser* files. There were a total of 9,238 *fid*s, and out of these 1,134 occurred once, and 1,666 occurred multiple times in the SPINE data set.

Number of files processed	342546
Number of unique MD5 keys	97603
Number of *fid* files present	9238
Unique/redundant *fid* files	1134/1666
Number of *ser* files present	7609
Unique/redundant *ser* files	1074/1396

Table 2.1: Summary of results of MD5 analysis of SPINE dataset

2.3 Discussion

Data management systems in biology and science more broadly are important features to an experimental set-up that many groups invest considerable time and resources into. SPINE helped track the progress of NESG targets through the sample production and structure determination pipeline. SPINE is a valuable resource for future data mining and machine learning, as there is much more than just structures and NMR screening data in SPINE, information that relates to sample production, protein purification protocols, and NMR spectral quality scores. The data duplication in time domain files across SPINE has unknown sources. Regardless of the source, the MD5 collisions recorded in this study highlight the need of more quality assurance and curation methods in SPINE. One possibility for the MD5 collisions observed is how easily NMR experimental directories may be copied and moved in a filesystem. Experimental directories may be over-written unknowingly. The data integrity challenges uncovered in this analysis motivated more advanced work in designing light-weight, robust databases for the biomolecular NMR research group. We developed SpecDB motivated in part from the experiences working with SPINE.

2.4 Conclusion

NMR time domain data is a readily measurable NMR observable for biomolecular samples. NMR spectroscopists can record a dozen or more FIDs in a single data collection session. The abundance of FIDs presents a novel and unused resource for sophisticated data mining and machine learning to aid the biomolecular sample optimization methods and NMR structure determination pipelines. We found data

integrity challenges in the SPINE database that supported the NESG where multiple FIDs were copied across the database. Future database development should have built in checks to prevent such redundancies from happening for future NMR time domain datasets.

3. SpecDB: A Relational Database for Archiving Biomolecular NMR Spectra Data

Submitted for review in Journal of Magnetic Resonance 01/25/2022

Keith J. Fraga[1], Yuanpeng J. Huang[2], Theresa A. Ramelot[2], G.V.T. Swapna[3], Arwin Lashawn Anak Kendary[1], Ethan Li[2], Ian Korf[1]*, and Gaetano T. Montelione[2]*

[1]Department of Molecular and Cellular Biology, University of California, Davis, California, 95616, USA

[2]Department of Chemistry and Chemical Biology, Center for Biotechnology and Interdisciplinary Sciences, Rensselaer Polytechnic Institute, Troy, New York, 12180 USA

[3]Department of Pharmacology, Robert Wood Johnson Medical School, Rutgers The State University of New Jersey, Piscataway, NJ 08854, USA

3.1 Abstract

NMR is a valuable experimental tool in the structural biologist's toolkit to elucidate the structures, functions, and motions of biomolecules. The progress of machine learning, particularly in structural biology, reveals the critical importance of large, diverse, and reliable datasets in developing new methods and understanding in structural biology and science more broadly. Protein NMR research groups produce large amounts of data, and there is renewed interest in organizing this data to train new, sophisticated machine learning architectures to improve biomolecular NMR analysis pipelines. The foundational data type in NMR is the free-induction decay (FID). There are opportunities to build sophisticated machine learning methods to tackle long-standing problems in NMR data processing, resonance assignment, dynamics analysis, and structure

determination using NMR FIDs. Our goal in this study is to provide a lightweight, broadly available tool for archiving FID data as it is generated at the spectrometer, and grow a new resource of FID data and associated metadata. This study presents a relational schema for storing and organizing the metadata items that describe an NMR sample and FID data, which we call **Spec**tra **D**ata**b**ase (SpecDB). SpecDB is implemented in SQLite and includes a Python software library providing a command-line application to create, organize, query, backup, share, and maintain the database. This set of software tools and database schema allow users to store, organize, share, and learn from NMR time domain data. SpecDB is freely available under an open source license at https://github.rpi.edu/RPIBioinformatics/SpecDB.

3.2 Introduction

The success of machine learning in biology over the past 10 years, particularly deep learning in the field of protein structure prediction(“Artificial intelligence in structural biology is here to stay,” 2021; Baek et al., 2021; Jumper et al., 2021; Kryshtafovych et al., 2021), is leading many communities in the biological and medical sciences to reevaluate their data ecosystems(“Opportunities and obstacles for deep learning in biology and medicine | Journal of The Royal Society Interface,” n.d.). Data is the key resource to train and deploy sophisticated machine learning models(Goodfellow et al., 2016), and the degree to which well-organized data is available can spur innovation across biology, chemistry, and medicine. Modern protein Nuclear Magnetic Resonance (NMR) spectroscopy laboratories also require an easy-to-use, lightweight data management system for managing NMR time domain data, archiving them locally, and

eventually moving these data into public repositories. The goal of this study is to advance the data infrastructure and practices for the scientific community to provide tools and protocols for archiving NMR time domain data for future data mining and machine learning.

NMR spectroscopy has a rich history of developing and applying machine learning at all stages in the experimental pipeline(Cobas, 2020; Hoch, 2019). High impact examples include predicting protein torsion angles(Shen and Bax, 2015), chemical shift prediction(Li et al., 2020), NMR spectral peak picking(Klukowski et al., 2018; Li et al., 2021), and reconstruction of non-uniformly sampled free induction decays (FIDs)(Karunanithy and Hansen, 2021; Luo et al., 2020; Qu et al., 2020). Additionally, there have been efforts to organize NMR data into datasets suitable for machine learning, like the RefDB dataset with re-referenced chemical shifts(Zhang et al., 2003). Designing deep neural network architectures and applications of existing deep learning methods to tasks across the NMR data analysis and structure determination pipeline is an active area of research. To further develop and engineer sophisticated machine learning methods for NMR data analysis requires an accessible data infrastructure to collect more and richer datasets.

The free induction decay collected from the NMR spectrometer is the raw data that all proceeding steps in NMR experimental pipelines rely and build on(Wuthrich, 1986). The terms *time domain data* and *free induction decay* (FID) data are used interchangeably in the community for these raw data. The prospect of automatic analysis of FIDs to

produce NMR resonance assignments, dynamic information, or even molecular structures, is a long-standing goal and challenge in the NMR spectroscopy field. A large set of curated time domain datasets is a critical first step to support such applications. Biomolecular NMR time domain datasets are archived in the **B**iological **M**agnetic **R**esonance **B**ank (BMRB)(Romero et al., 2020). However, only a small percentage of BMRB entries have associated time domain data. One way to address this data gap is to provide a simple tool to allow organization and archiving of FID data, and associated metadata, soon after they are generated at the NMR spectrometer, and to provide a simple process for moving these data into the BMRB. In this way, a data resource of FIDs will grow in time.

Our approach to addressing the challenges in archiving and distributing raw NMR time domain data is a data management tool called **Spec**tral **D**ata**B**ase. SpecDB is a simple data management system that individual NMR research groups can install and use to create their own archive of organized experimental NMR data. SpecDB also provides capabilities to share all, or selected sets, of these data between research groups, and to transfer these data to the BMRB. Furthermore, SpecDB should be easily maintained by any spectroscopist or NMR spectroscopy research group, without much relational database knowledge.

One important goal recommended by the wwPDB NMR Validation Task Force is to foster community practices of consistently depositing time domain NMR data into the BMRB(Montelione et al., 2013). The need for such large-scale efforts in preserving and

disseminating FID data is greatly appreciated in the NMR and structural biology communities(McAlpine et al., 2019; Morris, 2018). However, unless these time domain data are stored in an organized manner, together with appropriate metadata describing the sample and data collection parameters, deposition and retrieval of the underlying FID data for biomolecular NMR studies and machine learning is difficult and time consuming, and is often not even attempted. SpecDB provides a platform for organizing and storing NMR time domain data, together with metadata describing associated data collection parameters and samples, in a form suitable for future data mining and machine learning. SpecDB also addresses important issues of data reproducibility and validation of research results. This software platform is a step forward in developing a data infrastructure for learning on NMR time domain data, as well as promoting practices of regular deposition of FID data to the BMRB.

SpecDB is related to **L**aboratory **I**nformation **M**anagement **S**ystems, or LIMSs. There are, and have been, many LIMS developed by the NMR community, and across the chemical and biological disciplines. One successful LIMS is the **S**tructural **P**roteomics in the **N**orth**E**ast (SPINE) database(Bertone et al., 2001; Goh et al., 2003), built to support the protein sample production and structure determination efforts of the **N**orth**E**ast **S**tructural **G**enomics (NESG) Consortium (https://nesg.org/). The SPINE MySQL relational database tracks the progress of protein targets and projects through specific pipelines for protein sample production, characterization, and structure determination by NMR and X-ray crystallography. SPINE is associated with the OracleSQL relational database SPINS, **S**tandardized **P**rote**I**n **N**MR **S**torage(Baran et al., 2006, 2002), the

goal of which was to archive each step and associated data necessary to completely reproduce a specific protein NMR data analysis pipeline. Other successful LIMSs and/or software suites providing some of these same capabilities include ProteinTracker(Ponko and Bienvenue, 2012), Sesame(Haquin et al., 2008), PiMS(Morris, 2015), NMRFAM-SPARKY(Lee et al., 2015), NMRbox(Maciejewski et al., 2017), and CCPN(Vranken et al., 2005) to name a few. SPINE and SPINS are specialized to support the pipeline and infrastructure of a specific pipeline of a large-scale structural genomics project, and are not sufficiently general, light-weight, and portable to support the broader needs for data archiving across the biomolecular NMR community. However, they serve as motivations and guides for the design of SpecDB, which aims to address the specific data management problem of archiving NMR FID data and associated metadata by a small research group, needed to archive these FID data in the BMRB.

SpecDB, an FID database suitable for use by a single laboratory or a biomolecular NMR facility, was developed with five principal features. (i) The raw time domain data (FID) is the centrally tracked entity. (ii) Experimentalists can also archive metadata items needed to describe the FID data through text forms. (iii) The system supports interchange between database items in SpecDB to database items tracked by the BMRB, to allow for BMRB deposition. (iv) The database is searchable with structured queries. (v) Query outputs can write FID and associated metadata from the SpecDB database into a folder-based hierarchy, to allow users to interact with the FIDs and sample information in a filesystem format.

In this paper we discuss implementation and system requirements for the SpecDB software, the relational schema in SpecDB, the overall workflow for archiving NMR FIDs using SpecDB, and some useful query tools. The software is freely available for implementation by any laboratory on Linux computer systems through the following GitHub code repository: https://github.rpi.edu/RPIBioinformatics/SpecDB.

3.3 Methods

SpecDB is a software platform that can archive minimal sample and experimental descriptions of FID data obtained from an NMR spectrometer. The NMR spectroscopist can provide the appropriate information about the sample and NMR experiment in files or in text based forms. The information is then funneled into a relational database. With relational databases, data items are stored in tables, or spreadsheets, yet there are multiple tables where columns in one table connect or relate to columns in different tables. The relationships, or connections between columns in a SQL table is the relational aspect we are referring to. SpecDB is a database that NMR research groups can construct locally on their laboratory Unix or Linux computer systems. The SpecDB software has two overarching components: (i) the relational database that describes an NMR experiment data collection process and associated FID data implemented in SQLite, (ii) the Python software package of SpecDB that manages the insertion and querying of data from the database.

There are three key computational characteristics in SpecDB. First, the SpecDB schema and database is built using SQLite, a light-weight and fast implementation of

SQL. SQLite powers many websites and scientific applications, and is an important industry standard in IT and data science. With SQLite, the entire relational database is a single file, which makes managing database read/write/query permissions equivalent to managing file permissions in a file system. Sharing within group(s) can be easily set up with group permissions. Second, the SpecDB code base is developed in the Python language, which is one of the most widely used programming languages, particularly in the data science and bioinformatics field. Third, SpecDB utilizes the **J**ava**S**cript **O**bject **N**otation (JSON) text interchange format(“ECMA-404,” n.d.) to store key NMR experimental metadata items that describe an NMR experiment and FID data. JSON files are human readable, allowing investigators to easily work with them, and to update them interactively. Using JSON forms provides a general solution for representing metadata for biomolecular samples and NMR experiments. Various form filling tools can be developed and implemented in the future to produce the JSON files needed for SpecDB. In our current implementation of SpecDB, we use user-edited Google Sheets to create these JSON files. Fourth, SpecDB is developed for Linux operating systems as is common and standard in bioinformatics and structural biology.

3.4 Results

Figure 3.1 illustrates the data ecosystem for protein NMR and the challenges with archiving and organizing the raw time domain data. NMR research groups typically use laboratory or institute NMR facilities. Within each NMR research group are individual investigators working collaboratively on diverse molecular systems and questions. It is often the case that the storage and organization of the raw time domain data is left to

the individual scientist who collected the data, and this leads to many different practices and conventions, even within a single research group, for storing FIDs and the essential metadata that describe the experiment.

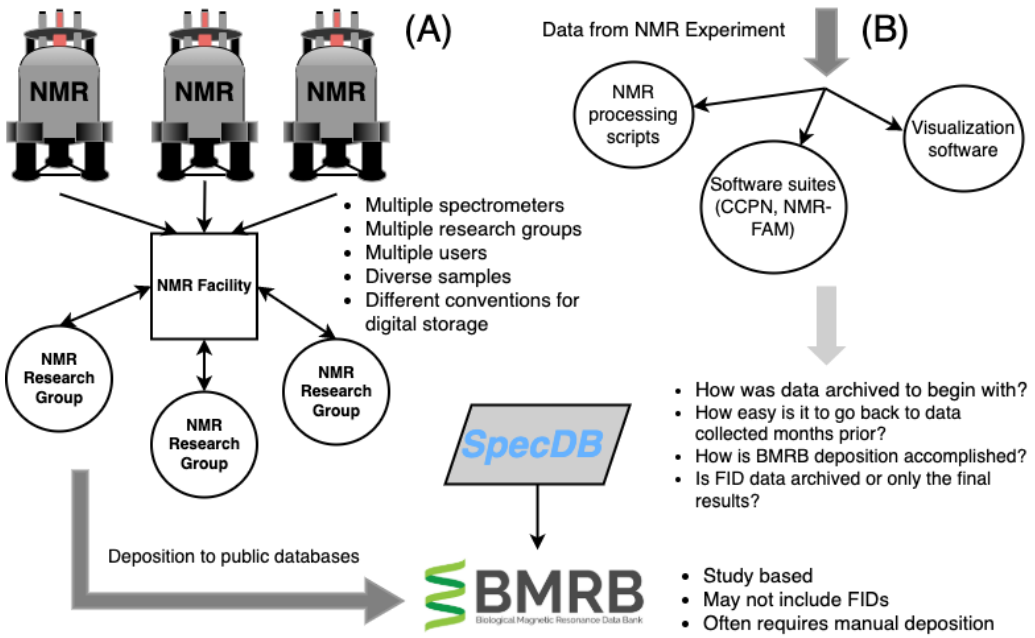

Fig. 3.1 : Data ecosystem for biomolecular NMR

(A) In general, biomolecular NMR research groups make use of shared NMR facilities where many NMR spectrometers are maintained and scheduled to specific users in specific research groups. After data is collected, a study is typically published and the experimental data to support the study is uploaded to the PDB and BMRB databases. The BMRB deposition is based on a specific study, and depositors are not required to submit time domain data. (B) Time domain data from NMR experiment is typically funneled into a processing phase of the data analysis, using specialized NMR processing tools, visualization tools, or other software suites for analysis and visualization, such as NMRPipe, SPARKY, CCPN, and NMR-FAM software suite. SpecDB provides solutions for some key questions including: how is the raw time domain data collected and stored?; how easy is it to find FIDs from a particular study or data range?; how do I retrieve the FID together with metadata and organize it for a BMRB deposition?

On commercial NMR spectrometer systems, the NMR FID data is included in a data collection directory that also includes many details of data collection, including the actual NMR pulse sequence code, spectrometer shim parameters, specific data collection parameters, pulse sequence waveforms, etc. In SpecDB, the “FID data” that is stored refers to this entire data collection directory. While most of the data items in these parameter files do not have specific representations in the SpecDB schema, they are still stored in the SpecDB database as a compressed directory. This allows for future development of the SpecDB schema to include specific data collection parameters, such as NOESY mixing time values or pulse widths, that are stored in these data collection directories. Hence, the initial focus of the SpecDB schema is to provide a platform for archiving these data, along with metadata about the NMR sample and other data collection parameters that are not included in these FID data directories.

3.4.1 Process of Developing the SpecDB Schema

The SpecDB schema developed to describe FID data (actually, the FID data directory), NMR sample, and associated metadata is designed to be compatible with both the SPINE database schema(Bertone et al., 2001; Goh et al., 2003), and with the NMR-STAR data ontology(Ulrich et al., 2019) used for archiving NMR data in the BMRB. Tables in the SPINE schema provide detailed information about protein samples, including information about the protein itself, the protein families it has been classified into, information about homologous proteins, disorder predictions, details of cloning, expression, crystallization data, and progress in structural characterization by NMR and X-ray crystallography. Most of these details are not required for SpecDB. We assessed each data item and data table in SPINE to identify a condensed subset that could

minimally and routinely describe an NMR FID data collection experiment and the corresponding NMR sample. In addition, by inspecting numerous representative NMR-STAR files from the BMRB, and in consultation with the BMRB developers, we identified additional data items that need to be provided for deposition an FID dataset into the BMRB, and hence need to be tracked through the SpecDB database. SpecDB thus provides direct translation from SpecDB data items to NMR-STAR tags. By using both SPINE and NMR-STAR, we were able to arrive at a minimal SQL schema to describe an NMR FID dataset sufficiently well to ensure its reproducibility, and to convey it into the BMRB.

The schema of SpecDB can be viewed as having two main parts, the database tables that describe the NMR sample, and the database tables that describe the FID data. Figure 3.2 depicts this two-wing structure of the SpecDB schema. Making a simple schema that is general enough for a wide range of applications is a significant challenge. Hence, the SpecDB schema is designed to be flexible enough to provide for significant modifications needed to support specific data pipelines and query requirements. Some examples of information not included in the SpecDB schema include details about the DNA cloning protocols used for making protein constructs, details of biomolecule purification procedures, detailed information about fermentation and expression, and bioinformatics, evolutionary, and gene-family metadata about the biomolecular target. These are not essential for the process of archiving the FID data and depositing it into the BMRB. However, the schema provides the flexibility for expansion to handle these additional data items in the future, which can be guided by,

for example, the SPINE schema which includes many of these additional sample preparation details.

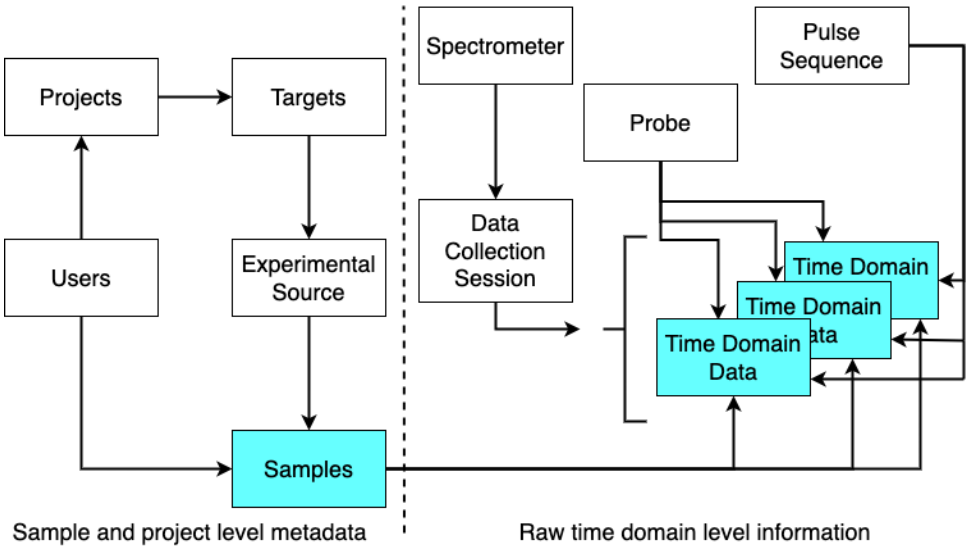

Fig. 3.2: The two wings of the SpecDB schema

The SQL schema for SpecDB can be considered in two parts, or wings. First is the description of the experimental Sample used for NMR data collection (left side). Users define Projects, Targets, Experimental Sources, and Samples. A Sample is part of a Project, defined by the group using SpecDB. Within Projects are Targets, biomolecules that are the subjects of the Project study. Experimental Sources describe aspects of the production of the Target. Samples (PSTs) are the actual samples that are analyzed at the spectrometer. The second wing of SpecDB relates information about the FID data (right side). SpecDB collects information about the Spectrometer, Probe, and Pulse Sequence used for collecting a specific FID. On some spectrometry systems, including Bruker systems, FIDs are collected in a “Session”, which is a series of related NMR experiments. This session hierarchy is preserved in the schema of SpecDB. The right-hand side of the figure indicates the many-to-one relationship between Sessions and FIDs, as there are multiple time domain datasets associated with the Sessions table.

3.4.2 SpecDB Tables that Provide Sample Information

The main table in SpecDB that describes the NMR sample is the **P**rotein **S**ample **T**ube (PST) table. The Protein Sample Tube refers to a physical tube holding a sample (which may be protein, nucleic acid, or other biomolecule or non-biological chemical). It includes sample tubes used in preparing a sample (e.g. Eppendorf tubes), or the actual NMR tube inserted into the NMR spectrometer. The set of relational tables that specify the sample and project are summarized in Figure 3.3. Each Protein Sample Tube is assigned a unique text identifier by the user called the *pst_id*. The *pst_id* is assigned by the user/research group. This identifier must be unique in the database. However, the actual id is determined by a lab-specific naming convention. This naming convention system is discussed below.

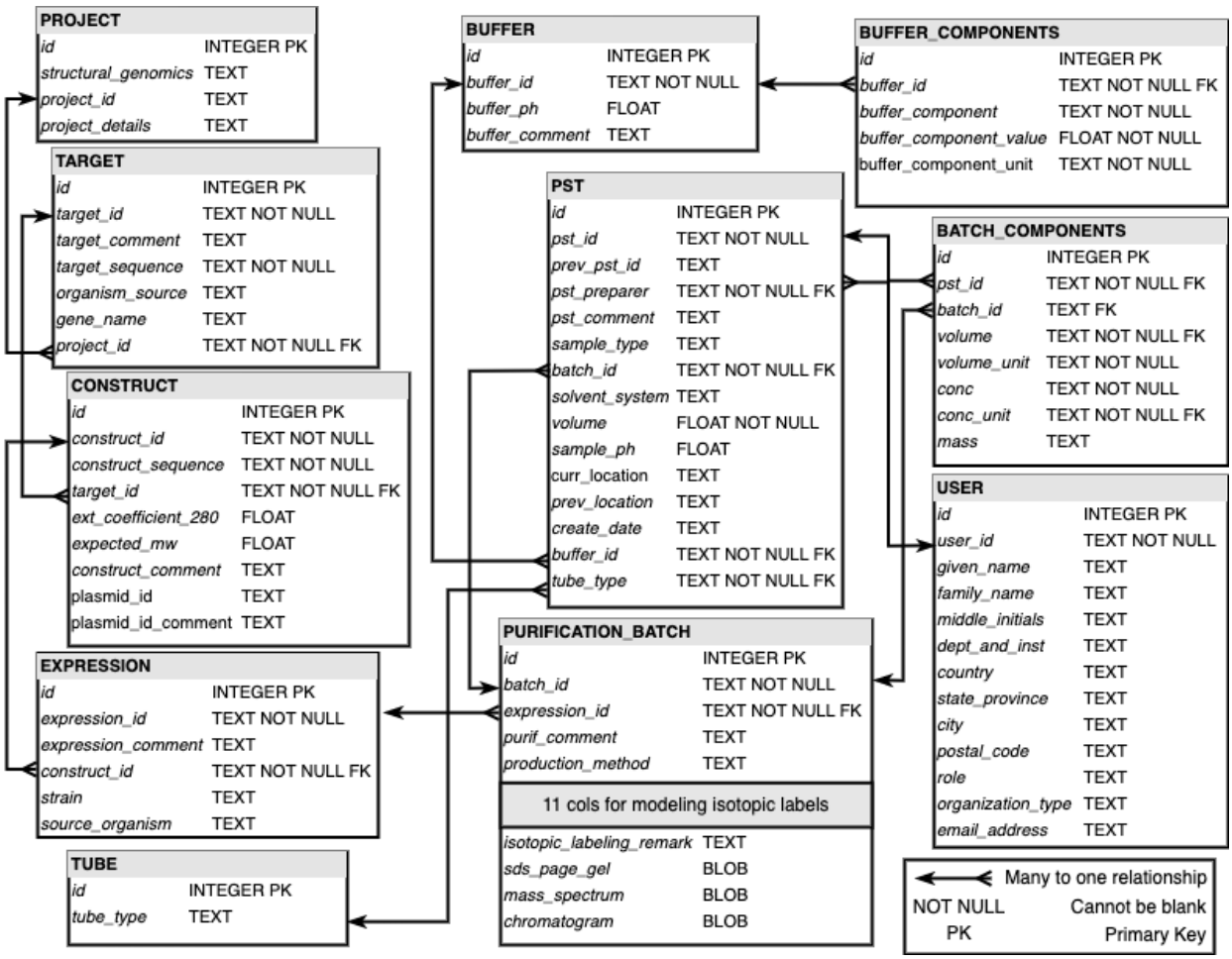

Fig. 3.3: Relational diagram for SpecDB tables

A view of the tables that describe an NMR sample and project. A hierarchy of meta-data information is depicted in the nested relationships between PROJECT, TARGET, CONSTRUCT, EXPRESSION, PURIFICATION_BATCH, and PST. Across the entire SpecDB schema there are 17 tables, 12 of which are displayed above for the description and modeling of NMR samples. Some data items (e.g. isotope-enrichment tags, shown in Supplementary Table 3.S1) are excluded for clarity. The connectors between tables indicate the relationships between tables. All the connectors in the diagram indicate a specific type of relationship, many-to-one relationships.

A key feature of the sample specific tables presented in Figure 3.3 is the nested nature of these tables. A sample description starts with the PROJECT table. Samples are part of a project, or cohesive study. The data items in the PROJECT table describe the research project, and provide a simple unique name for the project, the *project_id*. The hierarchical flow of information for describing a sample follows PROJECT, TARGET, CONSTRUCT, EXPRESSION, PURIFICATION_BATCH, BATCH_COMPONENTS, and PST. Multiple samples, each prepared as a “purification batch”, may be combined in a single PST to form complexes, as defined by the BATCH_COMPONENTS table. As a consequence of this hierarchy, every purification batch is associated with an expression experiment (also called a fermentation) run, every expression experiment is associated with a construct, every construct is associated with a target, and every target is part of a project. This nested hierarchy reflects the SPINE data schema, and in the future will allow for archiving NMR spectra from the NESG SPINE and SPINS databases into SpecDB for public distribution.

Inspection of the tables in Figure 3.3 illustrates that nearly every table has a text based identifier that is unique across the respective table. For example, the PST table has a *pst_id,* which provides a unique name for each protein (or nucleic acid) sample tube. SpecDB does not impose a specific convention or nomenclature on the data record identifiers (*project_id, target_id*, *pst_id*, etc), except that each data record must have a unique identifier. The naming convention for these unique identifiers should follow a convention set by the research group. For example, assignment of the unique textual identifier for a Protein Sample Tube, *pst_id*, may be chosen by the user who prepared

the sample tube. There is no internal SpecDB mechanism to generate identifiers other than preserving their uniqueness within their respective table. However, SpecDB checks identifiers at data input to prevent using an ID already in the database (unless a user specifies with a flag the need to update the associated record).

In the SPINE database, record id's follow a convention based on the *project_id*; e.g. HR for "human protein project at Rutgers". At each subsequent level in the organization hierarchy (targets, constructs, expressions, purification batches, and PSTs), there is a new delimiter that is added to the ID to make the ID unique and convey some information about the sample. Accordingly, *project_id* name HR defines the *target_id*'s; e.g. HR001A (the first domain of 1st protein HR001, in the project HR), which then defines the naming of *construct_id's*, *expression_id's, purification_batch_id's, and pst_id's.* In this example, the *purification_batch_id* HR001A.200_345_NTag.NiNTA.004 is the 4th batch of a construct of target HR001A that comprises residues 200 - 245 with an N-terminal hexaHis tag purified by NiNTA affinity purification. The corresponding NMR sample tube *pst-id*'s are assigned abbreviated names based on the *target_id* (e.g. HR001A.001, HR001A.002, etc.), which fit better on NMR tube labels. It should be noted that this naming convention is convenient, but does not replace accessing the corresponding PST record (and the associated hierarchy of records) to get complete and accurate information about the sample. Users of SpecDB may adopt this convention, or develop their own unique naming system for record ids.

Inspection of the schema for the PST table (see Figure 3.3) highlights the relational nature of SpecDB. The PST table links to other tables that describe the sample. For example, the user who generated the Protein Sample Tube is recorded in the PST table using the *pst_preparer* column. The value in the *pst_preparer* indicates the user who created the PST. In order to know the many attributes that describe a user, the value provided in the *pst_preparer* column is a key that links back to the USER table where the remaining items to describe the user are stored, rather than creating many columns within the PST table to record the user's first name, last name, email address, etc. All the user's information is stored in the USER table, and is linked to the necessary rows in the PST table through the *user_id* key. There are four tables that all connect to the PST table, as illustrated in Figure 3 through the barbed connectors. The barbed connectors indicate that the relationship between the two tables being connected is a many-to-one relationship. For instance, many sequence constructs can be made for a single protein target.

Next we will describe each table presented in Figure 3.3 and the role each plays in describing an NMR sample, following the hierarchy discussed above. A target is generally a biomolecule (protein, nucleic acid, polysaccharide, etc), although non-biological molecule samples can also be described with this schema. The biomolecule may come from a natural source, or be artificially designed or synthesized. For natural proteins, the protein is defined by the Uniprot(The UniProt Consortium, 2021) protein sequence of the full-length protein. A unique *target_id* is defined for each target, and linked back to the corresponding PROJECT table. Following the TARGET table is the

CONSTRUCT table. It is often the case that the biomolecules being studied with NMR have an amended primary sequence, for purification reasons (e.g., a purification tag), resulting from mutations introduced for functional studies, due to truncations to suppress aggregation, or for other reasons. Hence, the construct sequence studied by NMR is generally different from the target sequence. A construct is assigned a *construct_id* and a link to the *target_id* from which it was made. Associated with each construct are one or more expression (or fermentation) experiments. The EXPRESSIONS table, designated by a unique *expression_id* provides metadata on how the expression of the construct in a particular bacterial strain or other organism was accomplished.

Following expressions are protein purification batches, described in the PURIFICATION_ BATCH table. This table also provides a *sample_sequence*, which may be different from the construct_sequence if purification tags are removed in the process of purification. Here, SpecDB also allows users to store the absorbance extinction coefficient (e.g., at 280 nm for proteins) expected for the purified sample_sequence, which can be estimated relatively accurately from the protein or nucleic acid sequence(Gill and von Hippel, 1989; Nwokeoji et al., 2017), and the expected molecular weight. If the construct is isotope-enriched, this needs to be accounted for when retrieving the expected molecular weight from the sequence. Within the PURIFICATION_BATCHES table is a recording of the isotopic-labeling actually achieved for the biomolecule. The isotope-labeling may be that expected based on the isotope-enrichment strategy used, or that determined by experimental data such

as NMR or mass spectrometry. Not all the isotopic labeling schemes tracked in the SpecDB schema are listed Figure 3.3; additional schemes are listed and described in Table 3.S2. Currently, eleven common types of isotope labeling can be tracked in the PURIFICATION_BATCH table and more can be added as needed in future versions. The *isotope_labeling_remark* is a free-text field that allows the user to record labeling methods not captured by the isotope-enrichment strategies currently supported for the PURIFICATION_BATCH table. A PST may come from a single purification batch, or (in the case of complexes) multiple purification batches. The one or more batches combine to form a PST are tracked by the BATCH_COMPONENTS table.

The PST table also provides a description of the protein sample tube itself using a controlled vocabulary of common sample and NMR tubes, including conventional NMR tubes and Shegemi NMR of various diameters (i.e. 1-mm, 1.7-mm, 3-mm, 4-mm, 5-mm, 8-mm, 10-mm). In the case of solid-state NMR, a PST tube can be a rotor of various sizes. The PST also tracks the actual sample pH (or the expected pH based on the buffer used), who prepared the sample tube, and the physical location of the protein sample tube, the solvent, the buffer, as well as the the sample volume and concentration of the target molecule(s) in the sample tube.

Associated with the PST table is the BUFFER table. The BUFFER table records all the buffers used in the database, and each buffer is provided a *buffer_id*. A *buffer_ph* is recorded, which may be different from actual *sample_ph* recorded in the PST table. In order to describe the contents of a buffer, SpecDB also has a

BUFFER_COMPONENTS table. Each row of the BUFFER_COMPONENTS table is a different component used to make buffers, where the buffer is associated with this component through the *buffer_id*. A buffer component requires three items to complete its description: the name of the component, the concentration of that component, and the unit of concentration. Buffers can be very complex, and having a simple table structure to record all the buffer components of a particular buffer may be tedious in the short term, but highly valuable due to accuracy in archiving the sample, reproducibility, and for future data mining.

The last table to highlight in Figure 3.3 is the USER table. Here, the investigators in the research group are recorded, their names, emails, department and institution, etc. The USER table is important for many reasons. In particular it is helpful for trouble-shooting a project when it is known who made a particular sample, or recorded a specific spectrum. User information is also required for creating a BMRB deposition, and for documenting credit for publication.

Elements of the SpecDB schema not illustrated in Figure 3.3 are the controlled vocabularies on the SpecDB data items, or the text strings or values allowed to be inserted into the database. Not every data item has a controlled vocabulary, but several require controlled vocabulary to ensure consistency in what users input as information across the schema. Table 3.1 presents a representative sample of the data items that have a controlled vocabulary in the SpecDB schema. As an example, items such as *volume_unit* cannot take any text string, there are only certain text strings (i.e., units of

volume) allowable to be used for the *volume_unit* value. This helps maintain consistency in the database.

SpecDB data type	Description	Controlled Vocabulary Description
structural_genomics	Optional for BMRB deposition. Indicate if project is part of a structural genomics effort.	(‘yes’ OR ‘no’)
iso_13c_enrichment	Describe the carbon-13 isotopic enrichment.	Must have substring (“ % 13C “)
buffer_component_unit	Record the unit the buffer component exists in the buffer.	(‘mM’ OR ‘% (v/v)’ OR ‘mg/ml’)
sample_type	Record the type of PST sample.	(‘solution’ OR ‘solid’)
volume_unit	Unit of volume for protein component of PST.	(‘nL’ OR ‘µL’ OR ‘mL’ OR ‘L’)
conc_unit	Concentration unit for protein component of PST	(‘µg/ml’ OR ‘mg/mL’ OR ‘nM’ OR ‘µM’ OR ‘mM’)
nus	Indicate if non-linear sampling was employed	(‘yes’ OR ‘no’)

Table 3.1: Controlled vocabularies across the SpecDB relational schema

These are representative examples of the controlled vocabularies used by SpecDB. Indicated are the SpecDB data types where a controlled vocabulary exists, a description of the data modeled for each data item, and the exact expression that controls the allowable values for each SpecDB column. Not every column with a controlled vocabulary is presented in this table. The allowable tube type names are listed in Table S1, and controlled vocabularies for different isotopic labeling is described in Table 3.S2.

3.4.3 SpecDB SQL Tables to Archive FIDs

The second wing to the SpecDB schema are the tables that describe FID data sets (Figure 3.4). An FID is recorded on a spectrometer, so there is a SPECTROMETER table that records the names of the spectrometer used, the spectrometer model, and the field strength.

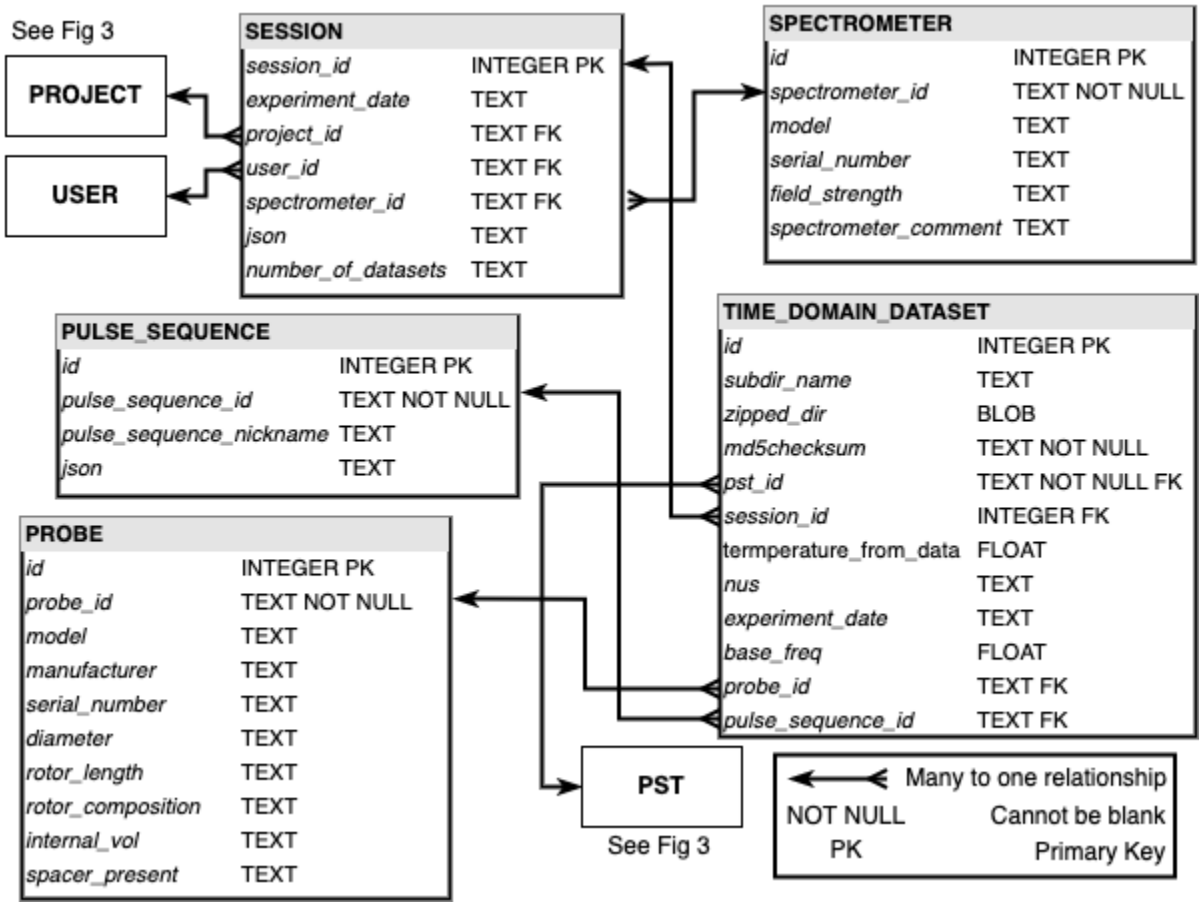

Fig. 3.4: Relational diagram for SpecDB tables that describe NMR FID data
The relationship diagram depicted in this figure are for the tables in the SpecDB schema that describe NMR experiments and the data collected at the NMR spectrometer. Inside the diagram are callbacks to the PROJECT, USER, and PST tables in Fig. 3. The complete FID subdirectories are stored as **B**inary **L**arge **Ob**ject**s** (BLOBS) in the *zipped_dir* column of the Time_Domain_Datasets table, allowing other auxiliary files such as acquisition/acquisition status files needed to reproduce the experiment to be archived along with the time domain NMR data.

The next level of this hierarchy is SESSIONS, which are sets of FIDs collected together in a data collection session. A session could be a single FID data set (as for example data collected on Varian spectrometers using VNMR software), or a directory containing subdirectories with a single FID in each (as is the case on Bruker spectrometer systems). The concept of a session stems from the management of FIDs on Bruker spectrometers using TopSpin software. Here, the NMR spectroscopist may queue up several pulse sequences to be run in succession at the spectrometer. The FIDs from these pulse sequences are placed into different subdirectories of a session directory. This subdirectory structure is reflected in the SpecDB SESSIONS table. In the SpecDB SESSIONS table, the spectrometer that is being used is recorded, the data collection dates, the number of FIDs to be collected in the session, the project associated with the session, along with the user running the session. Internally to SpecDB, each session is given a *session_id*, which is simply a row integer counter, and is an item that the user does not set. The session directory contains the *specdb.json* JSON file, with metadata provided by the user, as well as the sub-directories with the recorded FID data and spectrum-specific acquisition parameters. In this way all of the metadata describing all of the FID data collected in the session, including information about the user, sample, and other aspects of the data collection are all stored together in the JSON text file at the session directory level.

The SESSIONS table is a useful LIMS concept for NMR data management. Sessions reflect the fact that some FIDs are related to other FIDs. The SESSIONS table also highlights that SpecDB is more than a database of FIDs, it describes the samples the

FIDs are recorded from, and also maintains information about relationships between FIDs.

Ultimately, the recorded FIDs (i.e. FID data directories) themselves are stored in the *zipped_dir* column of the TIME_DOMAIN_DATASETS table. The *zipped_dir* data item is a compressed data directory containing the FID and the associated vendor-specific data collection metadata. The *zipped_dir* is stored in the database as a **B**inary **L**arge **Ob**ject (BLOB). To ensure that every stored FID in SpecDB is unique, we perform a MD5 hashing function on the raw FID data file to be inserted to SpecDB, and the hashed string is stored in the TIME_DOMAIN_DATASETS table in the *md5checksum* data item. Also included in the TIME_DOMAIN_DATASETS table is the *probe_id*, which links back to the PROBES table that describes the NMR probe used in the collection of the FID. The probe information is stored at the level of the FID instead of the session in the SpecDB schema because it is possible that within the same session, users may switch out probes for different applications. The probe information collected in the PROBES table displayed in Figure 3.4 contains items that relate to probes for both solution and solid-state NMR.

The TIME_DOMAIN_DATASETS table also includes a name of the pulse sequence used to collect the corresponding FID, and the *pst_id* for the sample being analyzed. The pulse sequence name is a nickname (e.g. 2D NOESY) used for queries or BMRB deposition; the actual pulse sequence miniprogram is included in the FID data directory associated with the TIME_DOMAIN_DATASETS table. Like the probe information

described above, the *pst_id* is modeled at the level of the FID instead of session because spectroscopists might change the sample, or use different samples over the course of a session. For instance, spectroscopists might perform a pH titration by adjusting the pH in the sample tube and recording a new FID on the adjusted sample tube. In this case, each time the pH is changed results in a new *pst_id*. The spectroscopist records the path of sample changes using the *prev_pst_id* data item in the PST table, allowing protein sample tubes to inherit and trace information from each other.

There are two remaining tables that are not represented in Figures 3 and 4 and do not necessarily fit the two-wing structure used to describe the SpecDB schema. The first is the STAR_CONVERSION table. This table is not intended to be modified by users as it contains a translation between the SpecDB data items to NMR-STAR tags. This helps the SpecDB applications for writing database contents into NMR-STAR formats. The second table, discussed below, is the SUMMARY table for queries, which is a subset of commonly searched data items in SpecDB in one flat table.

3.4.4 SpecDB Workflow

The intended workflow with SpecDB is illustrated in Figure 3.5, which depicts an NMR spectrometer with the associated computer workstation where the FID is initially recorded. Typically, these FIDs cannot be stored indefinitely at these NMR workstations, and are moved to a laboratory server where SpecDB is installed, using *rsync* or other mirroring operation. SpecDB is run and queried on this laboratory server.

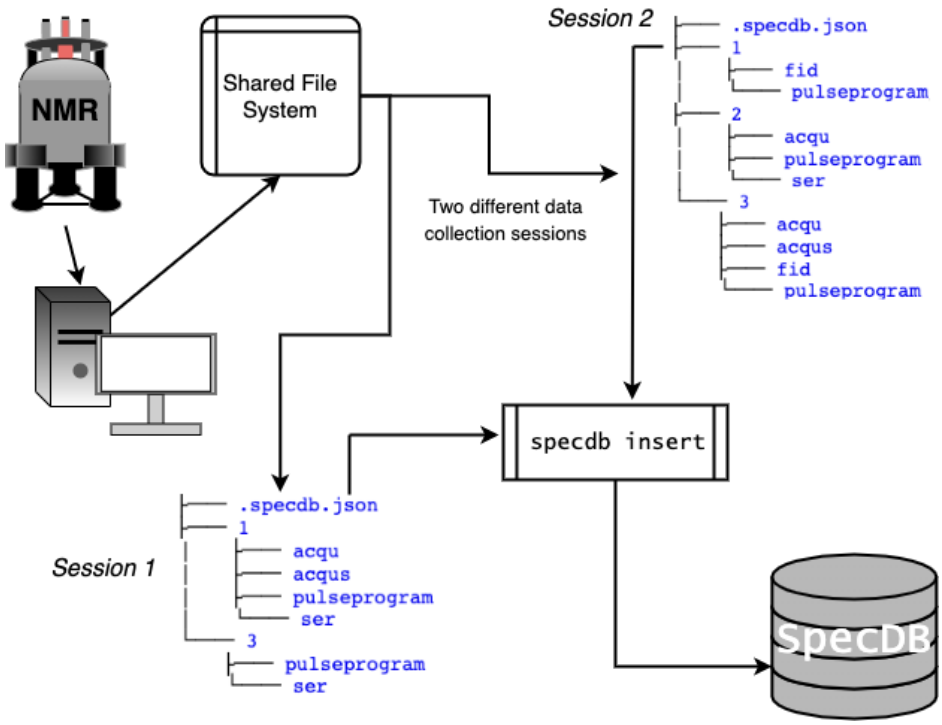

Fig. 3.5: Movement of NMR time domain data from NMR spectrometer to SpecDB

FIDs are generated at the spectrometer and stored on the associated computer workstation. Typically the collected data from the NMR spectrometer workstation is then transferred, either through an *rsync*, a mirror, or manual copying to a different, more stable filesystem. The NMR spectroscopist will typically store the FIDs they collect in a directory somewhere in the shared file system. These directories are highlighted with the indicated *Session 1* and *Session 2* directory structures. From both *Session 1* and *2*, there are two or more sub directories with FID data (denoted here as fid for 1D NMR data and ser for multidimensional NMR data). At the top level of these sessions sits a *specdb.json* JSON file. The *specdb.json* describes the data collection session. Once the spectroscopist enters the required metadata information into the JSON file, the *specdb insert* command is used to insert the *specdb.json* file into the database.

The JSON file (*specdb.json*) is located in the main directory of a data collection session, and contains information about each FID data set in the subdirectories under that session. The structure of the JSON file defines which sample, pulse sequence, etc. is associated with each subdirectory FID data set (Figure 3.5). The JSON file may be edited either before data collection (e.g, entering sample data prior to data collection), at the spectrometer (e.g. designating which NMR experiment is being collected in each subdirectory), or after moving the data to the database server (e.g. completing metadata information prior to submitting the data to the database). Once the JSON file metadata is complete, the SpecDB command line tool can be run by the user to insert the FID data sets and metadata for the session into the SpecDB database.

3.5 SpecDB Sub Commands

There are six subcommands in SpecDB: *create*, *backup*, *restore*, *insert*, *forms*, *summary*, and *query*. Table 3.2 lists each SpecDB subcommand, the arguments each takes, and an example command as potentially run at the command line. The SpecDB subcommands *create*, *backup*, and *restore* are designed to be used by a research group's SpecDB manager. A new SpecDB database is created with the command *specdb create*. The location where the SpecDB SQLite database resides, and the backup SQLite database file are command line arguments to the *create* subcommand. Together, *specdb backup* and *specdb restore* perform the incremental backup operations for SpecDB. The subcommands *insert*, *forms*, *summary*, and *query* are intended to be routinely used by individual researchers. The SpecDB command-line tool

allows users to interact with the database in a shell environment, but we are also developing graphical applications to make these commands more user-friendly.

Once the user has completed a *specdb.json* file for their data collection session, these data are inserted into SpecDB database using *specdb insert*. Inserts that would override data already present are not allowed by default: SpecDB warns the user and forces the user to confirm if editing of previous values is intentional. s*pecdb forms* can be used to generate template JSON forms for any data item/table in the database, to provide a guide to assist users in creating a JSON file of metadata.

SpecDB Command	Arguments		Example
create	*db*	Name and path where SpecDB database to be built	$ specdb create --db lab/data/lab.specdb.db --backup lab/backups/backup.db
	backup	Path and name where incremental backup will be maintained	
insert	*json*	Path to JSON file to process for insertion	$ specdb insert --json lab/data/new/lab.specdb.json --db lab/data/lab.specdb.db --ovewrite
	db	SpecDB database to insert into	
	overwrite	On conflicts between the JSON file and SpecDB, update the corresponding SpecDB row with data from the JSON file	
forms	*table*	SpecDB table to create a filled text form for	$ specdb forms --table user --num 3
	num	Number of forms to make for the requested table	
backup	*db*	SpecDB database to be backed up	$ specdb backup --db lab/data/lab.specdb.db --backup lab/backups/backup.db
	backup	Database to backup to	
summary	*env*	SpecDB environment file	$ specdb summary psts --env lab/data/lab.specdb.env
	table	SpecDB table to view a summary report of	
query	*sql*	Raw SQL query on the *summary* table	$ specdb query --sql “SELECT user_id FROM summary” --output dir --env lab/data/lab.specdb.env
	output	Process results into either a directory structure or STAR files	
	env	SpecDB environment file	

Table 3.2: Description of SpecDB subcommands

The above table lays out all the commands within the SpecDB library that are used to manage NMR FID data in a filesystem and a SQLite database. The left column provides each sub command name. The middle column provides documentation on the command line arguments for each sub command. The right most column provides illustrative examples of how each SpecDB sub command could be executed in a general shell environment.

The *specdb summary* subcommand provides a summary of any table in the subject SpecDB database instance. Using *specdb summary* the contents of the requested table is printed in a formatted table. For example, *specdb summary users* will display a formatted table in the terminal session with table columns *user_id*, *given_name*, *last_name*, etc, and rows be the users that have been entered into the database. This allows users to review the data items and their values already inserted into the database, which can help users complete their *specdb.json* files and to assess inconsistencies.

Lastly, *specdb query* command allows users to perform queries against a SpecDB database and retrieve the subset of FIDs data sets that satisfy the query. These data are output in one of two formats, either a directory hierarchy of the data or NMR-STAR files for each FID. Building a SQLite database for NMR FIDs and sample information allows researchers to utilize the SQL language to extract data from the database using diverse and complex queries using SQL. The *specdb query* tool is designed to give researchers a way to make queries against a SpecDB database without using a sophisticated SQL query. With *specdb query*, users submit a SQL *SELECT* statement to be run against a SpecDB database. The *specdb query* tool will return all FIDs captured in the provided SQL *SELECT* statement. However, *specdb query* will only accept queries of data items listed in the SUMMARY table. SUMMARY is a SQL view of the SpecDB database, where columns from different tables are stitched together into a 2-dimensional table that is compatible with spreadsheets. More complex queries can be accomplished by connecting directly to the SpecDB SQLite database file. Table 3.3 lists

out the exact terms incorporated into the SpecDB SUMMARY view, as well as examples of each data item.

Column Name	Description	Example data
id	Row counter that assigns a unique integer to every FID collected	12
experiment_date	Data FID was collected	2022-01-11
user_id	The user_id of the user that collected the FID	KJF
project_id	Project_id for the project the FID is a part of	SPIKE Project
structural_genomics	Whether the FID is part of a structural genomics project	no
temperature	Temperature FID was collected at	25 C
buffer_id	Buffer identifier that sample was in	NMR-Buffer-17
pst_id	Protein Sample Tube identifier for the sample the FID was recorded of	SPIKE.2022
batch_id	Identifier for purification batch	SPIKE.2022.b
expression_id	Identifier for expression run	SPIKE.2022.e
construct_id	Identifier for construct	SPIKE.102-450
target_id	Identifier for target	SPIKE
target_sequence	Protein sequence of target	MGHSSSTVLAM…
construct_sequence	Protein sequence of construct	HHHHHHLEMGHSSSTV…
pulse_sequence_id	Name of pulse sequence	1H-NOESY
spectrometer_id	Identifier of spectrometer FID was measured at	Hu800
field_strength	Field strength of spectrometer	800 MHz
probe_id	Identifier for the probe used in FID acquisition	Avance_2033478
tube_type	Type of tube PST was in	4-mm Shigemi tube
nus	Whether non-linear sampling was employed in FID acquisition	no
zipped_dir	Binary object of the zipped Bruker directory that contains the FID	(BINARY)

Table 3.3: Schema description of SpecDB *Summary* View

This table presents the specific items tracked in *Summary* View in the SpecDB schema. Users can make structured queries against these columns and elect to have the query results be formatted into a directory structure or into NMR-STAR files. Left column indicates the names of the columns in the SpecDB *summary* view. Middle column is a description of each column in the *summary* view. Right column provides an example of the data types stored in each of the columns.

Figure 3.6 illustrates a *specdb query* and shows condensed examples of the two output format types. NMR researchers are often expecting a directory structure when they are working with their data, so outputting query results as a directory hierarchy is a natural format option. One goal of SpecDB is to also generate FID data sets and metadata in NMR-STAR format by a query against the database. Using the STAR_CONVERSION table, every SpecDB data item can be translated to NMR-STAR save frames and tags. These NMR-STAR files can be used for deposition to the BMRB, or for sharing experiments between researchers and labs.

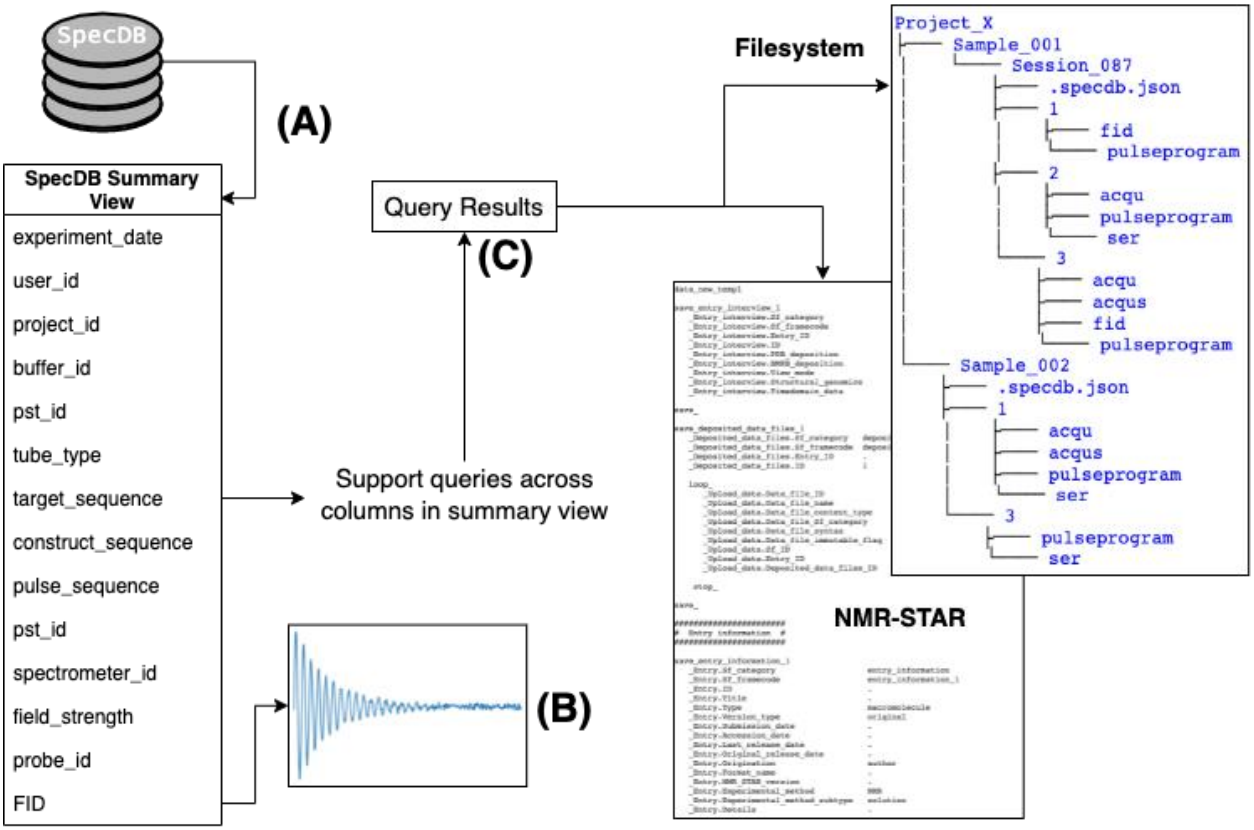

Fig. 3.6: Overview for the SpecDB query system

(A) The list on the left of the figure above is a condensed version of the SpecDB Summary table; the complete list of columns supported in the Summary view is provided inTable 3 Supplementary Table X.. (B) A link to the raw binary data for each free induction decay (FID) is included in the Summary view. (C) SpecDB restricts SQL queries to data items in the Summary view. More complex queries can be handled directly through *sqlite3*. Queries generate FID data collection directories, formatted either in a filesystem folder hierarchy or as a set of NMR-STAR files.

3.6 Discussion

SpecDB introduced in this study is a lightweight, flexible, robust LIMS for organizing and archiving NMR FID data generated in a small NMR research group or a large NMR facility center. In this first iteration of SpecDB, we had five goals: (i) Archive time domain FID data and key associated metadata, (ii) harvest user-supplied metadata that describes an FID experiment in human read-able JSON files, (iii) provide tools to allow queries of FID data sets in the database, (iv) allow records in SpecDB to be queried, organized, and formatted in NMR-STAR format for automatic deposition to the BMRB, (v) allow users to query, organize, and output SpecDB contents in a user-friendly hierarchical directory structure. All five goals are successfully implemented in version 1.0 of SpecDB. The SpecDB schema, based on the SPINE and NMR-Star schemas, was developed to describe an NMR sample and FID data set. Although focused on supporting descriptions of biomolecular (e.g. proteins and nucleic acids), the schema will also support non-biological NMR sample descriptions. SpecDB also includes command line tools that manage the insertion of new data into the SpecDB database, incremental backup of the database, and querying and retrieval of data from the database.

SpecDB falls under the general umbrella of a **L**aboratory **I**nformation **M**anagement **S**ystem (LIMS). There are several LIMS systems for NMR studies, and for many other domains of science. The diversity of LIMS systems is driven by the unique needs of a scientific discipline and community data standards. Dedicated LIMS have been

developed for individual research groups that have specific workflows. The challenge with any LIMS system is the balance between complete control over data tags/items to be collected from users, vs complete flexibility where software is intelligent enough to handle what a user is providing or requesting. Designing too much control makes the utility ”brittle” and incapable of handling slight deviations from the original data management pipeline, posing challenges to users who want to use the LIMS system but are frustrated by strict data management policies imposed by the structure of the system. On the other hand, designing a highly flexible system that is sufficiently light-weight for general distribution is very challenging.

LIMS or data curation software employed across the NMR data ecosystem can be organized into three main groups. First, there are LIMS that seek to archive and track sample production. Examples include SPINE(Bertone et al., 2001), ProteinTracker(Ponko and Bienvenue, 2012), Sesame(Haquin et al., 2008), and PiMS(Morris, 2015) to name a few. Across these sample production specific LIMS, the schemas are quite different from each other as they serve different needs, processes, and communities.

Second, are data/software communities and packages that organize the software needed to record and process NMR data, and to track intermediate and final results of a data analysis pipeline. Examples of these include SPINS(Baran et al., 2006, 2002), CCPN(Vranken et al., 2005), NMRFAM-SPARKY(Lee et al., 2015) and NMRbox(Maciejewski et al., 2017). The applications and packages that make up this

second set are not databases that store FIDs or processed NMR spectra in a relational database. They represent software suites where software conventions, versions, and data input and output formats are standardized.

The third group of data organization and curation software in the NMR field is the global community standards for making NMR data and structures publicly available. The BMRB is the main public data repository for magnetic resonance data types and molecular structures. The BMRB schema also organizes sample details, spectrometer and probe information, pulse sequence and experimental details, and is the international archive for many different NMR data types. The BMRB schema also has a textual-based archive format called NMR **S**tandard **T**ext **A**rchival and **R**etrieval (NMR-STAR) format(Ulrich et al., 2019). Using NMR-STAR, NMR experiments can be recorded in a text based, machine-readable format for deposition to the BMRB, as well as storage of NMR data and experiments in a standard, well-defined ontology. Alongside NMR-STAR is the **N**MR **E**xchange **F**ormat(Gutmanas et al., 2015) (NEF), a different textual ontology to describe NMR experiments and data. NEF has particular value as a light-weight NMR restraint exchange format. NMR-STAR and NEF are standard ontologies and schema to archive and/or share NMR data and experiment descriptions, but they are not databases designed to save reproducible descriptions of NMR experiments and the collected FIDs from an experiment. Researchers will typically utilize NEF only well into a NMR study (e.g. for structural modeling) and interact with the BMRB only at a late stage of a project, after most of the study has been completed.

SpecDB is designed to allow archiving of NMR FID data immediately after data collection at the spectrometer. It does not handle any stage of post-processing of NMR FIDs, so SpecDB does not fit into the second grouping of software packages for NMR analysis described above. NMR FID data are an important data resource that will serve as input into future data mining and machine learning efforts. SpecDB fulfills the timely need for a light-weight database that can reliably organize NMR experiments as they are being collected, where the raw FID data is the central data item in the database along with experiment and sample metadata. SpecDB also supports data interchange into other FID deposition formats, like NMR-STAR.

The FID as a data item in the SpecDB schema represents a significant shift in the understanding of LIMS for NMR data. Historically, FID binary files presented a challenge for digital storage due to their size and limits on available storage. Dedicated servers or archival media (e.g.removable disks, tapes) are usually used to store FIDs. Although a separate database might be available to organize the metadata for the experiment, in most cases the connection between the FID data and the sample metadata is provided only through a physical laboratory notebook. In some LIMS systems, FID datasets are accessed through a filesystem path designating where the FID is located on the filesystem or archival media. For instance, in SPINE and SPINS, NMR data was recorded and tracked, but the FIDs sit in hierarchical directories linked to these metadata via filesystem paths. SPINE stores a wide range of experimental data and valuable information, yet the raw NMR experimental data is outside the relational nature of SPINE, leaving it vulnerable to separation from the metadata, data loss, and

security issues. Presently, storage and memory resource limitations are not as much of a concern as they were a few decades ago, and relational databases can directly archive several hundred or thousands of binary files from multidimensional NMR experiments. Storing FIDs directly into the database also protects against data loss as the FIDs are internal to the database and associated with their metadata descriptions. SpecDB provides storage of FID data directly into a relational database as data items themselves.

SpecDB does not make an effort at this time to store processed frequency-domain NMR spectra. Since processed spectra files are much larger than the FID data from which they are generated, they present larger memory and storage challenges. However, it is possible to archive processing scripts in SpecDB (e.g. NMRPipe(Delaglio et al., 1995) processing scripts), allowing regeneration of specific frequency-domain processed spectra. It would also be useful to have a database of such processed spectra (or scripts), prepared by NMR processing experts, for machine learning applications, but this is beyond the scope of the current version of SpecDB.

Using the Structured Query Language (SQL) to construct a relational database allows for structured queries to be completed by the NMR experimentalist, and ultimately data scientists analyzing these data post data collection. In the biomolecular NMR community FID data (as well as processed spectra) are typically stored in a file system. These data are often left to be organized by the specific researcher for a particular project. The standards/conventions employed by one researcher to organize their data

collection may not be consistent across the community, or even within a research team. For example, if it were necessary to collect FID data generated in a specific date range without a database, relational or otherwise, custom software would be needed to accomplish the task. These issues are addressed by SpecDB, which provides a uniform and query-able platform for organizing NMR FID data within a research laboratory or NMR facility, and a path to sharing these data across the scientific community.

SpecDB uses JSON files to record metadata information about an NMR experiment. Data items in JSON files can be used in SQL queries. When an FID is collected at the NMR spectrometer system, other auxiliary files are also created by the data collection software, including data collection parameter files, the pulse sequence mini-program, and various spectrometer-specific acquisition files including waveform and shim files. These auxiliary files are critical to allow reproducibility of an NMR experiment. For this reason, the entire data collection directory needs to be captured and stored in the SpecDB database. To allow for queries on these data items in spectrometer files, some of them, such as the date(s) when the data were collected, and the temperature of data collection, are automatically pulled from the data collection files into JSON files, and then archived in tables of the relational database. Future query requirements can be supported by adding additional data items (e.g. NOESY mixing time) to the set of items pulled from the data collection parameter files and supported by the JSON files and SpecDB.

The command line tool of SpecDB has features similar to *git*, the command line tool to manage software projects involving many developers. In *git*, there are subcommands

like *status*, *add*, *commit*, etc that are all particular steps in the tracking and maintaining a software codebase with many collaborators. In *git*, new files are added and committed to the repository at the discretion of the developer. Similar to *git*, the NMR experimentalist inserts FIDs into a SpecDB database when they determine they have recorded a complete set of metadata items for the session in the JSON file. In essence, SpecDB command line tools track the status of JSON files that contain the same data items and tables as the relational schema, and insert all the corresponding information correctly into the database.

JSON files also allow for innovative approaches for harvesting metadata for SpecDB. The initial distribution of SpecDB includes tools for using Google Sheets for metadata entry and conversion to JSON files. In our own lab, we are also exploring Microsoft Excel files, WordPress forms, and the commercial LabArchive electronic laboratory notebook tools for this purpose. We intentionally reserve judgment on recommending the “best solution” to this data harvest problem, since this will be laboratory dependent. The input to SpecDB is, ultimately, the JSON files, and various approaches can be taken to create these files.

SpecDB provides a lightweight, flexible, and robust schema and tools to archive time domain data of NMR experiments. As mentioned in the Introduction, there is a community-wide effort to expand the manner and standards for NMR researchers to deposit the raw FIDs that support their studies. Deposition of time-domain data is a major recommendation and goal of the wwPDB NMR Validation Task force for rigor and

reproducibility in biomolecular NMR studies. The BMRB is collecting time-domain data, and using SpecDB able to produce NMR-STAR files for an FID is an important next step for wide-adoption of policies and practices for deposition of raw FID data.

3.7 Conclusion

The goal of SpecDB was to build a relational schema and software to collect and track data items about biomolecular NMR samples and FID data sets, with the primary purpose of archiving and sharing these NMR time domain data. Standardized approaches for archiving FID data in relational databases provides the opportunity to develop rich datasets needed to learn new approaches for NMR data analysis. Although developed primarily using solution NMR data for proteins and nucleic acids recorded on Bruker NMR spectrometer systems, SpecDB can be easily generalized for archiving also solid-state NMR data, NMR data for oligosaccharides or small molecules, and data obtained on Varian, Agilent, JOEL, or Q-One NMR spectrometer systems. Broad use of SpecDB has the potential to create a rich data resource for a wide range of machine learning applications for biomolecular NMR. SpecDB is publicly available under the MIT open source license at the following GitHub repository: https://github.rpi.edu/RPIBioinformatics/SpecDB. The repository comes with installation instructions and tutorials to get started with SpecDB.

3.8 Author Contributions

All authors contributed to the design of the database schema. KF wrote the SpecDB database code. The manuscript was written with contributions from all authors.

3.9 Funding Sources

This work was supported by grants from National Institutes of Health grants R01 GM120574 (to GTM) and R35 GM141818 (to GTM).

3.10 Conflict of Interest Statement

GTM is a founder of Nexomics Biosciences, Inc. This affiliation is not a competing interest with respect to this study. The remaining authors declare no competing interests.

3. 11 Acknowledgments

We thank Drs. S. Aviran, E. Baldwin, J. Hoch and J. Wedell for helpful discussions and advice.

3. 12 Supplementary Materials

Allowed NMR tube type names	“0.5 mL Eppendorf”
	“1.7 mL Eppendorf”
	“15 mL cent. tube”
	“50 mL cent. tube”
	“1 L cent. tube”
	“8-well PCR strip”
	“NMR tube”
	“plate well”
	“1-mm NMR tube”
	“1.7-mm NMR tube”
	“3-mm NMR tube”
	“3-mm Shigemi tube”
	“4-mm NMR tube”
	“4-mm Shigemi tube”
	“5-mm NMR tube”
	“5-mm Shigemi tube”
	“8-mm NMR tube”
	“8-mm Shigemi tube”
	“10-mm NMR tube”
	“10-mm Shigemi tube”

Table 3.S1: Controlled vocabulary for the allowed tube types in SpecDB

The table above lists the allowed tube types for SpecDB. If a user attempts to indicate a tube type different from the tube name in this list, then insertion into the database will be prevented and the problem logged into the SpecDB log file. Users can edit the controlled vocabulary for the tube names in their own SpecDB instances by amending the Tubes table.

SpecDB data type	Description	Controlled Vocabulary Description
iso_13c_enrichment	Describe the Carbon-13 isotopic enrichment	Must have substring (“ % 13C “)
iso_15n_enrichment	Describe the Nitrogen-15 isotopic enrichment	Must have substring (“ % 15N “)
iso_2h_enrichment	Describe the Deuterium isotopic enrichment	Must have substring (“ % 2H “)
iso_19f_Trp_enrichment	Describe the Fluorine-19 isotopic enrichment on Tryptophan amino acids	Must have substring (“ % 19F-Trp “)
iso_19f_Phe_enrichment	Describe the Fluorine-19 isotopic enrichment on Phenylalanine amino acids	Must have substring (“ % 19F-Phe “)
iso_1hd1_Leu_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Leu-HD1 protons	Must have substring (“ % 1HD1-Leu “)
iso_1hd2_Leu_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Leu-HD2 protons	Must have substring (“ % 1HD2-Leu “)
iso_1hd_Ile_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Ile-HD protons	Must have substring (“ % 1HD-Ile “)
iso_1hg1_Val_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Val-HG1 protons	Must have substring (“ % 1HG1-Val “)
iso_1hg2_Val_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Val-HG2 protons	Must have substring (“ % 1HG2-Val “)
iso_1hb_Ala_methyl_ enrichment	Describe the Deuterium isotopic enrichment for Ile-HD protons	Must have substring (“ % 1HD-Ile “)

Table 3.S2: Controlled vocabulary for different isotope labeling methods

The isotopic labeling terms are data types described in the Purification Batch table. Protein samples can contain various isotope labeling schemes, and the terms above captures several common isotope labeling schemes. For isotope labeling that does not fit in these isotope labeling terms listed above, there is a separate *isotope_labeling_remark* in the Purification Batch table supporting isotope labeling methods not modeled across these columns.

4. Predicting Genomic Signals from DNA Sequence Alone with Deep Neural Networks

4.1 Abstract

One of the foundations of modern biology is structural genome annotation, which labels genomic subsequences with categorical information such as repeat, exon, intron, promoter, or variant. Industries from health to agriculture to synthetic biology rely on the structural annotation, so it is important that these features are annotated as accurately as possible. Much of what has been labeled in the past has relied on classical models such as position weight matrices. Years ago, when sequence data was sparse, it was not possible to train more sophisticated models, but today, with the advent of high throughput sequencing, data is no longer scarce and we can now employ more sophisticated models. This chapter explores the use of deep neural networks for the recognition of genomic features.

4.2 Introduction

Identifying the exon-intron structure of protein-coding genes in genomes is an important challenge in bioinformatics(Salzberg, 2019). Improvement in sequencing technologies has led to higher-quality genome assemblies at a decreasing financial cost. The annotation of assembled genomes is thus a critical bottleneck for discovering new biological and evolutionary mechanisms(Ejigu and Jung, 2020). In general, there are

two levels of genome annotation: structural and functional annotation(Ejigu and Jung, 2020; König et al., 2018). Structural annotation refers to identifying the specific genomic structures, like promoters, protein coding regions, exons, introns, transposons, repetitive elements, transcription factor binding sites, etc. Functional annotation refers to describing the functions of the various structurally annotated elements in a genome, for example the function of a particular gene, or the phenotype of a specific single nucleotide polymorphism. Functional annotation and other downstream studies such as evolutionary comparison require structural annotation. For this reason, it is critically important for the structural annotation to be as accurate as possible.

Genome annotation often utilizes comparative approaches between evolutionarily-close pre-annotated genomes to annotate or re-annotate a genome(König et al., 2018). Using conservation and comparative approaches has advantages and disadvantages, and ultimately with the production of high-quality genomes at faster rates requires ab-initio methods that annotate genomes from the sequence content of genomes(Ejigu and Jung, 2020; Salzberg, 2019). Furthermore, comparative annotation approaches are vulnerable to error propagation between annotation versions, particularly when estimates for misannotation in genomes range from 5% to as larger as 80%(Jones et al., 2007; Schnoes et al., 2009).

Identifying genes and gene structure is a central problem in annotation. Scallzitti et al performed an assessment of the top five gene prediction methods using a curated benchmark set of 1793 genes across 147 species(Scalzitti et al., 2020). The top

performing gene predictor achieved an F1 measure of 0.52 for predicting whether each nucleotide in the benchmark genes are part of an exon or intron. Scalzitti and collaborators performed other evaluations across the five gene predictors and found that ultimately ab-initio gene prediction is a challenging problem with substantial room for improvement(Scalzitti et al., 2020).

The advance of artificial intelligence in many computational tasks(Goodfellow et al., 2016), from computer vision, natural language processing, protein structure prediction(Pearce and Zhang, 2021), and much more presents new opportunities to improve gene prediction algorithms. Specifically, the prospect of designing and training deep neural network architectures for biology and medicine is an active area of research(“Opportunities and obstacles for deep learning in biology and medicine | Journal of The Royal Society Interface,” n.d.). This chapter explores the extent to which deep neural networks can be trained to classify genomic sequences. Figure 1 illustrates the machine learning tasks tackled in this chapter.

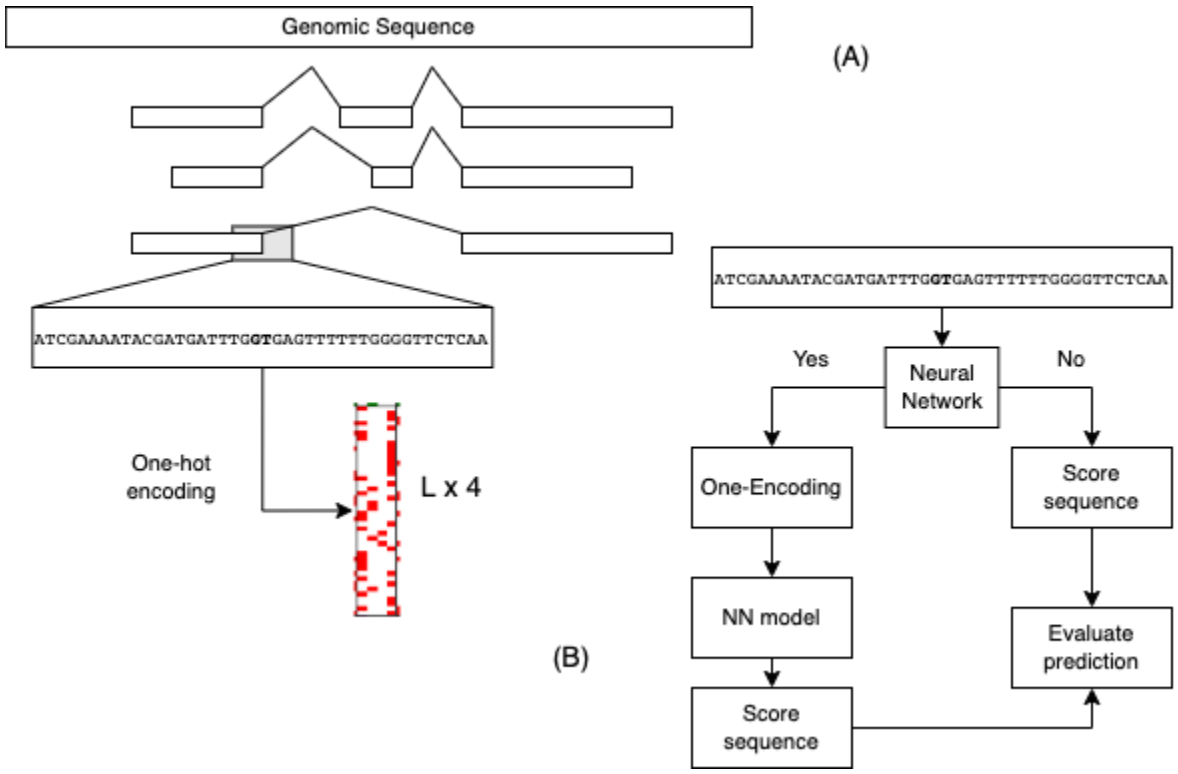

Fig. 4.1: Overall schematic of the learning task tackled in this study

(A) A genomic region is presented with multiple splicing isoforms. A specific splice donor site is highlighted in the gray box. This sequence can be one-hot encoded to represent the sequence numerically. (B) The general learning task pipeline is presented. First an input sequence is provided, then one-encoding is implemented if prediction is being performed with a neural network, the sequence is then scored, and the prediction is evaluated if the respective model produced the correct label for this input sequence.

Deep neural networks are computational architectures with nodes, or neurons, that are wired together in non-trivial ways to process input information in a hierarchical fashion(Goodfellow et al., 2016; LeCun et al., 2015). The wires, or connections, between nodes across different layers of a deep neural network have weights associated with them, indicating the strength of the connection between the pairs of nodes. The weights are learnable parameters, and the numerical value of the weights are optimized relative to a specific learning task. In our case, the task is to learn from the patterns present in genomic sequences what determines a sequence to be a splice donor or acceptor site. We also built and trained neural networks for exon and intron classification.

To date, there are three deep neural networks trained for splice site prediction: DeepSplice(Zhang et al., 2018), Splice2Deep(Albaradei et al., 2020), and DASSI(Moosa et al., 2021). Each architecture was trained on different datasets, but each presented >94% accuracy for splice site predictions. Each study utilized convolutional neural networks (CNNs), a specific type of deep neural network architecture that is well established on a variety of tasks(Sercu et al., 2015; Zhang et al., 2018). CNNs in the context of learning from DNA sequences in principle find local sequence motifs that correlate with the class label assigned to the sequence. To use CNNs, DNA sequences must be represented in a numeric fashion. Each method above employed one-hot encoding to represent each sequence in their training and testing datasets. In one-hot encoding, each nucleotide in a sequence is represented by a binary vector, where only entry is non-zero which indicates the letter type at that

position of the sequence. Examples of one-hot encoded sequences are presented in Figure 1A.

When crafting a CNN for a task such as splice site recognition, care must be taken in how the CNN is constructed (e.g number of layers, sizes of convolution filters, number of pooling layers, etc) and also how it is trained. The sequences in the training set must be high quality and free of errors.

Choosing the training set for modeling splice sites is much more difficult than it may appear. Most of the time, people treat splice sites as if they were Boolean values. E.g. a particular GT corresponds to a real splice donor site or it does not. However, in reality, some sites are used frequently while others are not. It wasn't until the advent of massively parallel sequencing that we were able to observe thousands or even millions of splicing events per gene. Today, RNA-seq data shows us how often various splice sites are used. The number of splice sites one observes is partly a function of how deeply one sequences and how highly expressed the gene is. It is a complex picture and decidedly not Boolean.

In our studies, we don't simply consider splice sites to be True or False. Instead, we categorize them as highly or poorly expressed. This is a significant break with other studies that attempt to model splice sites in a Boolean manner. We take the same approach to exons and introns. That is, we segment them into categories of highly and poorly expressed.

The overall goals of the project are as follows:

1. Evaluate if expression level has any bearing on splice site, exon, or intron recognition.
2. Compare deep neural networks to position weight matrices (PWMs) for splice site recognition. Neural networks should be able to find patterns that PWMs cannot.
3. Compare deep neural networks to Markov models for exon and intron recognition.

4.3 Materials and Methods

4.3.1 Data Collection

The data used in this study are described in Tables 1 and 2. Table 1 contains the datasets collected from real genomic sequences, the splice donor/acceptor sites and the exon/intron sequences. Table 2 describes the different fabricated sequences generated to test the effect of positional/compositional dependencies on neural network performance.

Genomic Signal	Example	Levels
Splice donor site	ATCGAAAATACGATGATTTG**GT**GAGTTTTTTGGGGTTCT CAA CTCGAATAAAACCACAGAAG**GT**TAATTAATTGTTGTGAC AAT TGGATGGGAAAACTGTTTCG**GT**GCGTTCACTATTTTTA GAGA TTTTCAATACCGAATCCAGG**GT**GAGTTTGAATTGTTTTT TTT AACAAGCACTACAATTGGGT**GT**AAGTTTCACTTTTGGTT ATT GATTAGTGATATGGAAATTG**GT**GAGTGGCTTCAGGAAA ACAG	hi 8215 lo 8708 fake 172559
Splice acceptor site	TCTAAATTGTGAAACTTTTC**AG**CTCCCCGGTCCACCTG CTCA AACAAAATTAATAAAATTTC**AG**AAATATTCCAAGCTCTG CGT TTTTAAAGTTTTTTTTTTTC**AG**AAAATTCTGGAATCCACA AC TAGAAATCCAAAAAAAATTC**AG**GTTGTGGTAGACTGCC TGGA TTTTTTATGTTTTTATATTC**AG**GTGCTATACGCGATGAA CCC AACTCATTAAAAAAATTTTC**AG**GTGGTGTGCTCATCCC AGCT	hi 8193 lo 9148 fake 186671
Exons	GAGTATCCCGACAGATATCGATTCAAAAGACACACCGTTGCTC GATATAT AAGCGATGGCCCGCAGCCGCACACGATTCTTCTAGAATTCCAG AAAAAGA TTTTCCGTCAAACACAAACATTCAACGAGCCCCAGGGATGGAC ATTTATC TCCCGATAAAAATATCACTAATTTCGACGACGAGGATTTTGCCA ATTTTA TCACCTGTAAACTGTCAGTTTTTGGAAATATTTGGTTTGTCTACT GGAGA TCAATCGGTCGGTCTCATGAAATACCTGCAAACAGTGCAGAAG AATCCAG	total 33578 hi 16781 lo 16797 total.fake 33578 hi.fake 16781 lo.fake 16797
Introns	AAAAATCGAAATTACTTCTTAAAAATCTCGTAAAAATCGAATTCT TTCAG TAAAGGAAAAACATGAATTTCTAGCTTTTTCAGAGGTTTTCTATT AAAAA TCCATTTTGTGGTGGGGCTTATTCCGAAAAATCGTTGTTTTTTTT TTCAA GGTTGACAAAAGTATTTATGCATCGTGACTGCTTTTTTAGTCGG TTCTAC TAATGCTGTCTTAGTTTTTAATAGAGTGTATTTAAATTTTTTAAAT ATTC CTGTTTCAAATCAACCTTACAAAAGTTAGAAAAAACCAAAAGAG TATGAA	total 19389 hi 9691 lo 9698 total.fake 19389 hi.fake 9691 lo.fake 9698

Table 4.1: Summary of collected sequence dataset sizes from the *C. elegans* genome (WS282)

We used the *C. elegans* reference genome (WS282) from WormBase to collect DNA sequences for training and testing. We collected the reference genomes annotation and found all genes in the dataset. We removed genomic regions where multiple genes overlapped each other, or genes had converging promoters, or genes that were not well validated/expressed/misannotated.

We constructed two separate classification tasks, one for classifying whether 42 bp segments is a splice donor site and one for classifying whether 42 bp segments is a splice acceptor site. In each donor/acceptor classification task, the 42 bp is the consensus site with 20 bp flanking up and down stream. The negative label in each of these tasks were 42 bp segments inside genes that have a central AG/GT but had no evidence of being used for splicing.

We also collected full length exons and introns from the set of “normal” genes. We removed exons and introns above/below the 90%/10% percentile for read depth to remove outliers. We then used the median expression depth to split exons/introns into highly expressed sequences and lowly expressed sequences. From these sets, we sampled a single random 50 bp window within the full length exon/intron. These 50 bp windows were used for training and testing. The null/fake exon/intron sequences were random sequences with the same base frequencies as the real exon/intron sequences. For classification tasks involving high/low read depth exons/introns, the fake sequences were made with corresponding high or low read depth exons/introns.

Name	Experiment
don.obs	NNNNNNNNNNNNNNNNNNNN**GTrrg**NNNNNNNNNNNNNNNNN
don.ex1	NNNNNNNNNNNNNNNNNNNN**GTAAg**NNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNNNNNN**GTGGg**NNNNNNNNNNNNNNNNN
don.ex2	NNNNNNNNNNNNNNNNNNNN**GTaag**NNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNNNNNN**GTggg**NNNNNNNNNNNNNNNNN
acc.obs	NNNNNNNNNNNNNNNN**tttcAG**NNNNNNNNNNNNNNNNNNNN
acc.ex3	NNNNNNNNNNNNNNNN**TTTCAG**NNNNNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNN**CTTTAG**NNNNNNNNNNNNNNNNNNNN
acc.ex4	NNNNNNNNNNNNNNNN**tttcAG**NNNNNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNN**ctttAG**NNNNNNNNNNNNNNNNNNNN
acc.ex5	NNNNNNNNNNNNNNNN**TTNCAG**NNNNNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNN**NTTCAG**NNNNNNNNNNNNNNNNNNNN
acc.ex6	NNNNNNNNNNNNNNNN**ttNcAG**NNNNNNNNNNNNNNNNNNNN NNNNNNNNNNNNNNNN**NttcAG**NNNNNNNNNNNNNNNNNNNN

Table 4.2: Descriptions of the eight fabricated sequence experiments

Capital letters for the four nucleotides indicate 100% frequency for that nucleotide to be generated at its particular position. Lower-case letters indicate an 85% frequency for the nucleotide to be produced at the position, and the remaining 15% equally split among the three remaining nucleotide types. Each fabricated sequence experiment was generated 10,000 times.

To test the ability of neural networks to detect dependence between positions flanking a donor/acceptor site, we designed eight sets of fabricated donors/acceptors. These eight fabricated sets are listed in Table 2. We implemented three fabricated donor experiments, (1) all random sequence except for the consensus GTRRG motif, (2) sequences where all position are except the consensus is either GTAAG or GTGGG, representing a case where the +1/+2 positions after the GT are dependent on each other, (3) same as experiment 2, but the +1/+2 positions emit A/G 90% of the time instead of 100% as in experiment (2) Experiment (3) is experiment (2), but with random exceptions.

The remaining five fabricated experiments are all variations on the TTTCAG acceptor consensus sequence. (1) The TTTCAG consensus sequence is produced, but each position has 85% chance of producing the canonical base, remaining 15% is random errors. (2) Two flavors of the consensus: TTTCAG versus CTTTAG. (3) Same as (2) but the TTTC/CTTT are produced correctly 85% of the time per position. (4) Two flavors of the consensus: TTNCAG or NTTCAG. This is a case where sequence dependence skips a position. (5) same as (4) but with the pattern produced imperfectly, like in experiment (3).

4.3.2 Model architectures

There are five different machine learning models we built and trained for our classification tasks. First are position weight matrices (PWMs). PWMs model DNA

sequences as a series of independent positions and record the base frequencies per position in the sequence. PWMs, markov models, and weight array matrices (WAMs) are all described in Figure 4.2. To score how likely a given sequence may be produced from a PWM is simply the product of the probabilities for observing each nucleotide in their respective positions. PWMs are used to model specific locations in a genome such as splice donor or acceptors sites and cannot be used to model continuous features such as exons or introns. Using PWMs in classification, we build two PWMs, one for the true labeled set and one for a fake/negative labeled set. To evaluate a PWM model then is to score each sequence in the test set against the true/fake PWMs, and if the score is higher for the true PWM, then the test sequence is labeled true, otherwise the test sequence is given the fake label.

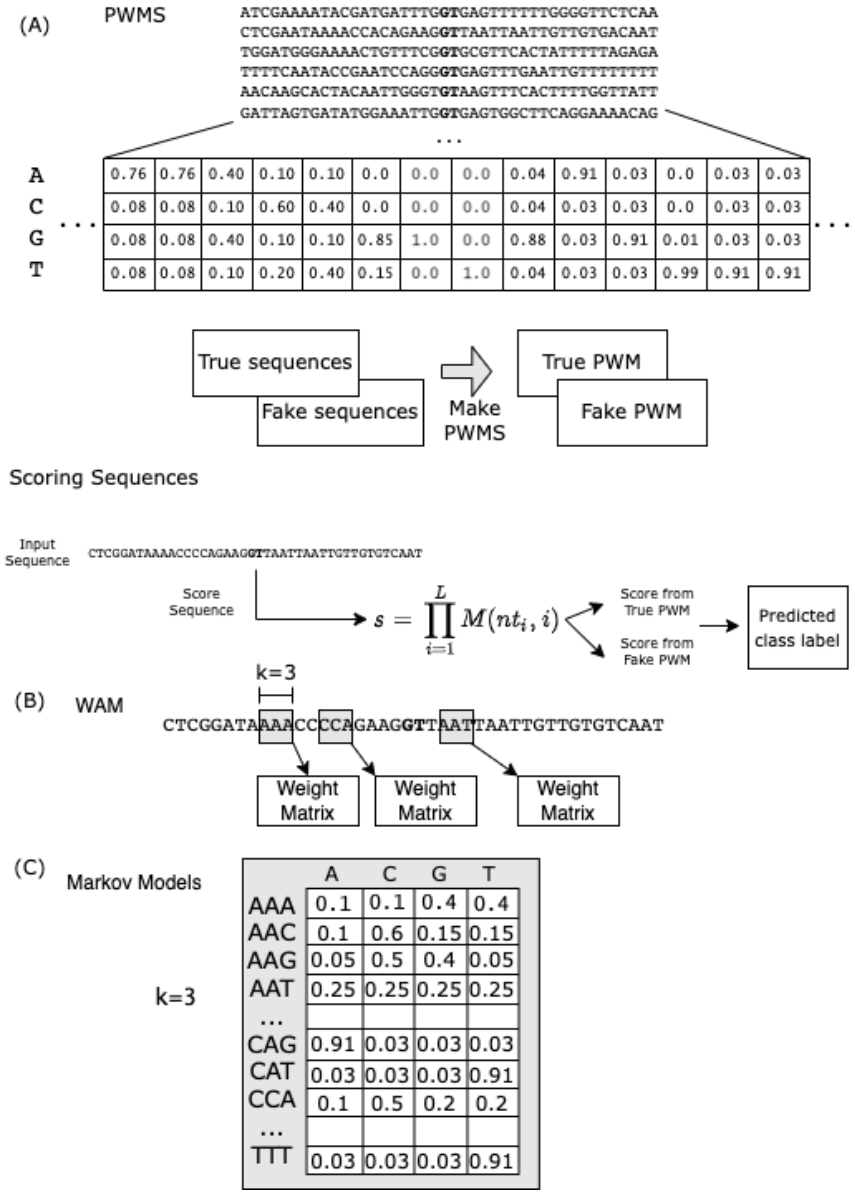

Fig. 4.2: Overview of PWMs, WAMs, and Markov Models for classification

(A) Overview of construction of PWMs and how to score sequences against a weight matrix. (B) Weight array matrices are PWMs but anchored to a specific position in the sequence. They relate the probability of observing each of the four nucleotides at a specific position given some level of context, represented by the *k*-mer in (B). (C) Markov models also model sequence context, but are not anchored to a specific position in the sequence. Rather they measure the frequency of emitting one of the four nucleotides given some previous *k* letters in the sequence.

The next layer of complexity above PWMs are Markov models. Instead of modeling sequences as a series of independent columns, a Markov model stores the conditional probabilities of each nucleotide given the previous *k* nucleotides. For example, in a 2nd order Markov model, the probability of generating an A is dependent on the previous 2 nucleotides. In this way, the base frequencies depend on the sequence context preceding each column. The value of *k* needs to be less than the total length of the sequence, but typical values for *k* range from 2-5. A *k* of 2 means base frequencies are dependent on the identity of the two preceding nucleotide identities. Markov models are used for modeling continuous features such as exons and introns rather than specific locations such as splice sites.

A slight variation of PWMs and Markov models are weight array matrices (WAMs). WAMs are used to model specific locations, like splice sites, not continuous features like exons. Each position in a WAM specifies a *k*th-orderMarkov model. This allows each position to model some local dependencies. Viewed this way, a PWM is simply a WAM employing 0th order Markov models.

There are two types of neural networks we built for the various classification tasks in this study. First are multi-layer perceptrons (MLPs). MLPs are networks of artificial neurons that are wired in a fully-connected fashion, illustrated in Figure 4.3. The phrase “fully-connected” refers to how every neuron/node in pairs of layers exhibits connections between all possible nodes, a subset of which is illustrated in Figure 4.3. MLPs have an input layer, multiple hidden layers that process information from the input, and ultimately

an output layer that is used for classification. MLPs are trained by computing an error/loss function between the network output and value the output should be given the input, which is known as supervised learning. The parameters of the MLP, all the weights between nodes and node biases, are updated to reduce this loss function using gradient descent and back propagation(Goodfellow et al., 2016; LeCun et al., 2015). The sequence information is converted into its one-hot representation prior to being passed through the network, also indicated in Figure 4.3.

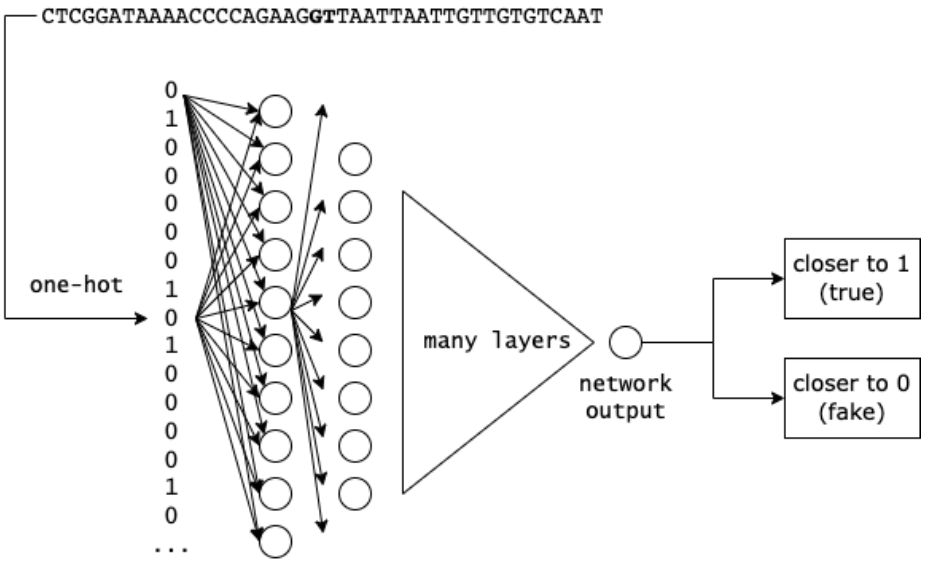

Fig. 4.3: Cartoon overview of multi-layer perceptrons used in classification of DNA sequences

The last type of machine learning architecture we trained for DNA sequence classification was convolutional neural networks (CNNs). Figure 4.4 illustrates a CNN and its use in classified DNA sequences. With MLPs, all combinations of nodes in each successive pairs of layers are wired together. CNNs take a different approach by convolving a filter across the input to find features that correlate with the sample's class label. CNNs do not only train one filter, but many filters as indicated by the *k* filters in Figure 4.4. Filters are convolved over the one-hot encoded sequence, each convolution produces a single number which is stored in a feature map. Each slice of the feature map is the convolution of a different filter. CNNs then perform another convolution on the resulting feature map, and can have many layers of convolutions. To ultimately produce the class label, the last feature map is flatten to a one-dimensional area, and optionally fed through a MLP to ultimately arrive at one output node for classification. The same program of gradient descent and back propagation is used to train CNNs as it was for MLPs.

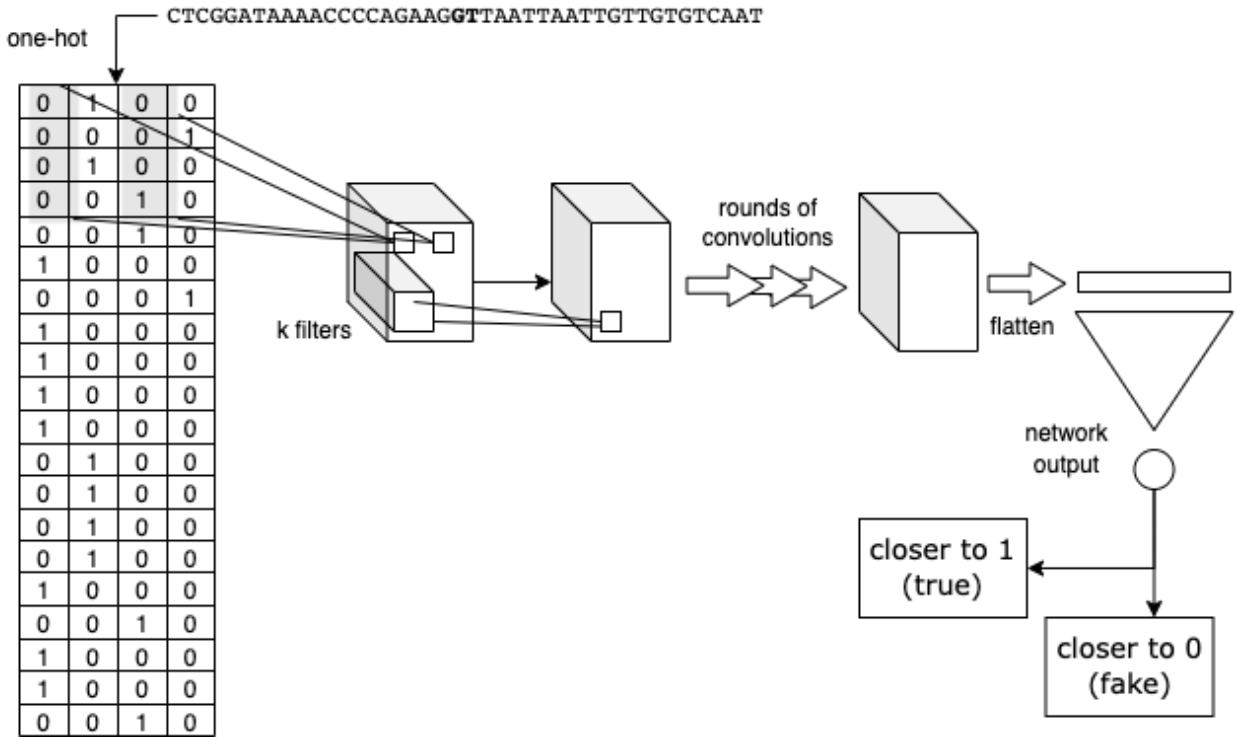

Fig. 4.4: Convolution neural networks in DNA sequence classification

Prior to the CNN, an input sequence must be one-hot encoded into a matrix Lx4, where L is the length of the sequence. Convolutional filters (grey) are convolved over the input one-hot sequence matrix, building up a feature map, or tensor indicated by the rectangular prism adjacent to the one-hot matrix. The depth of the rectangular prism indicated the number of convolutional filters used in the network. These feature maps then undergo multiple rounds of convolutions, then are flattened and processed into a single output node for classification.

In both neural network types, there are several hyperparameters that need to selected that are not optimized/trainable during the supervised training process. Examples of hyperparameters are the number of nodes/layers in a MLP. The architecture of the MLP/CNN is shown by the user, and the parameters of the designed network are then optimized to best reproduce the true labels of the sequences in the training set. Hyperparameter optimization in neural networks is a challenging task, and we explored many architectures and choices, but for simplicity and progress we chose a set of hyperparameters that produced consistently good performance. The hyperparameters for the MLP and CNN in this study are listed in Table 4.3.

Hyper-parameters	MLP	CNN
Layers	5	9
Nodes/layer	Splice [68, 42, 21, 10, 1] Intron/ Exon [100, 50, 25, 20, 1]	N/A
Learning rate	1e-2	1e-3
Optimizer	SGD	Adam
Kernel sizes	N/A	[(6,1), (4,2), (3,3), (5,4), (5,4)]
Residual connections	No	Yes
Batch Normalization	No	Yes
Final MLP layer	N/A	[50, 10, 1]
Batch size	32	512
L2 Regularization	None	5e-3
Epochs	50	25
Activation Function	Rectified Linear Unit (ReLU)	Rectified Linear Unit (ReLU)

Table 4.3: Hyperparameter description of MLP and CNN used for learning splice sites, exons, and introns

4.4 Results

Position weight matrices can achieve a ~90% accuracy on splice donor and acceptor sites, but only for highly used splice sites, which is indicated in Table 4.4. Building and testing PWMs for less used splice sites indicate ~73% accuracy on donor and acceptor sites. As expected, the entropy of a PWM is the indicator for better accuracy performance, with PWMs having greater entropy exhibiting better classification results.

Model	donor/acceptor	True	Fake	Accuracy	Entropy
PWM	don	don.hi	don.fake	0.9087	10.6647
PWM	don	don.lo	don.fake	0.7402	6.6211
PWM	don	don.hi	don.lo	0.7737	10.6647
PWM	acc	acc.hi	acc.fake	0.9314	13.0229
PWM	acc	acc.lo	acc.fake	0.7269	7.0464
PWM	acc	acc.hi	acc.lo	0.8233	13.0386
WAM (k=4)	don	don.hi	don.fake	0.9506	N/A
WAM (k=4)	don	don.lo	don.fake	0.8581	N/A
WAM (k=4)	don	don.hi	don.lo	0.5107	N/A
WAM (k=4)	acc	acc.hi	acc.fake	0.9650	N/A
WAM (k=4)	acc	acc.lo	acc.fake	0.8631	N/A
WAM (k=4)	acc	acc.hi	acc.lo	0.5538	N/A
MLP	don	don.hi	don.fake	**0.9744**	N/A
MLP	don	don.lo	don.fake	0.9380	N/A
MLP	don	don.hi	don.lo	0.7416	N/A
MLP	acc	acc.hi	acc.fake	**0.9776**	N/A
MLP	acc	acc.lo	acc.fake	0.9411	N/A
MLP	acc	acc.hi	acc.lo	0.7780	N/A
CNN	don	don.hi	don.fake	0.9390	N/A
CNN	don	don.lo	don.fake	0.7050	N/A
CNN	don	don.hi	don.lo	0.7695	N/A
CNN	acc	acc.hi	acc.fake	0.9513	N/A
CNN	acc	acc.lo	acc.fake	0.7271	N/A
CNN	acc	acc.hi	acc.lo	0.8180	N/A

Table 4.4: Accuracy for different models on real splice site data

The situation improves by about 5% when WAMs with four nucleotides of context are used. WAMs also perform better when built from highly used splice sites versus poorly used sites. Interestingly, WAMs reach ~50% accuracy when attempting to classify high versus low usage splice donor/acceptor sites, which indicate similar 4-mers exist in the two sets of sequences. However, PWMs could classify highly used sites from lowly used sites at ~75% accuracy. The deep neural networks can achieve better performance than the PWMs or WAMs, with MLPs being the most successful with accuracies on high versus fake reaching ~97%.

Model	donor/acceptor	True	Fake	Accuracy	Entropy
PWM	don	don.obs	don.fake	0.8259	6.2383
PWM	don	don.ex1	don.fake	0.8913	7.1412
PWM	don	don.ex2	don.fake	0.8238	6.2074
PWM	acc	acc.obs	acc.fake	0.9175	8.5816
PWM	acc	acc.ex3	acc.fake	0.9922	10.0110
PWM	acc	acc.ex4	acc.fake	**0.8895**	7.3853
PWM	acc	acc.ex5	acc.fake	0.9685	8.9151
PWM	acc	acc.ex6	acc.fake	0.8488	6.8938
WAM (k=4)	don	don.obs	don.fake	0.7253	N/A
WAM (k=4)	don	don.ex1	don.fake	0.8568	N/A
WAM (k=4)	don	don.ex2	don.fake	0.7405	N/A
WAM (k=4)	acc	acc.obs	acc.fake	0.8336	N/A
WAM (k=4)	acc	acc.ex3	acc.fake	0.9612	N/A
WAM (k=4)	acc	acc.ex4	acc.fake	0.8001	N/A
WAM (k=4)	acc	acc.ex5	acc.fake	0.9161	N/A
WAM (k=4)	acc	acc.ex6	acc.fake	0.7640	N/A
MLP	don	don.obs	don.fake	0.7822	N/A
MLP	don	don.ex1	don.fake	0.9208	N/A
MLP	don	don.ex2	don.fake	0.7942	N/A
MLP	acc	acc.obs	acc.fake	0.8838	N/A
MLP	acc	acc.ex3	acc.fake	0.9938	N/A
MLP	acc	acc.ex4	acc.fake	0.8532	N/A
MLP	acc	acc.ex5	acc.fake	**0.9834**	N/A
MLP	acc	acc.ex6	acc.fake	0.8088	N/A
CNN	don	don.obs	don.fake	0.8118	N/A
CNN	don	don.ex1	don.fake	**0.9342**	N/A
CNN	don	don.ex2	don.fake	**0.8180**	N/A
CNN	acc	acc.obs	acc.fake	**0.9222**	N/A
CNN	acc	acc.ex3	acc.fake	**0.9960**	N/A
CNN	acc	acc.ex4	acc.fake	0.8562	N/A
CNN	acc	acc.ex5	acc.fake	0.9828	N/A
CNN	acc	acc.ex6	acc.fake	**0.8536**	N/A

Table 4.5: Results on fabricated splice site experiments

Table 4.5 demonstrates that when simple positional and compositional dependencies exist in the dataset, neural networks are more readily able to detect those patters. In the acc.ex3 experiment, which had the most distinct patterns and dependencies, CNNs could achieve a 99%. The PWM can also solve acc.ex3, but when errors are introduced acc.ex4 PWM accuracy drops to 88%, but 85% for the CNN. The introduction of imperfect patterns is a challenge for neural networks to model in all the other experiments as well.

Model	Exons/ Introns	Positive case	Negative case	Accuracy
MM (k=5)	exons	exon.total	exon.fake	**0.8222**
MM (k=5)	exons	exon.hi	exon.hi.fake	**0.8356**
MM (k=5)	exons	exon.lo	exon.lo.fake	**0.8120**
MM (k=5)	introns	intron.total	intron.fake	**0.8022**
MM (k=5)	introns	intron.hi	intron.hi.fake	**0.8160**
MM (k=5)	introns	intron.lo	intron.lo.fake	**0.7948**
MLP	exons	exon.total	exon.fake	0.6946
MLP	exons	exon.hi	exon.hi.fake	0.6584
MLP	exons	exon.lo	exon.lo.fake	0.6467
MLP	introns	intron.total	intron.fake	0.7190
MLP	introns	intron.hi	intron.hi.fake	0.7278
MLP	introns	intron.lo	intron.lo.fake	0.6869
CNN	exons	exon.total	exon.fake	0.7765
CNN	exons	exon.hi	exon.hi.fake	0.7781
CNN	exons	exon.lo	exon.lo.fake	0.7528
CNN	introns	intron.total	intron.fake	0.6666
CNN	introns	intron.hi	intron.hi.fake	0.7824
CNN	introns	intron.lo	intron.lo.fake	0.7669

Table 4.6: Accuracy for different models on exons and introns

PWMs are not suitable for modeling exons/introns because the window is not anchored around any common motif. Markov models are the more suitable model to build to compare MLPs and CNNs to. Markov models with a context level of five have the best accuracy across the exon/intron experiments. CNNs are more capable at classifying exons/introns from non-exons/introns than MLPs. Additionally, splitting for exons/intron read-depth can produce a 2% jump in accuracy. Figure 4.5 graphs the training/testing loss/accuracy over the training epochs for a CNN training on exons. From Figure 4.5 we can see the CNN is overfitting to the training data as the CNN achieves 100% accuracy on the training set.

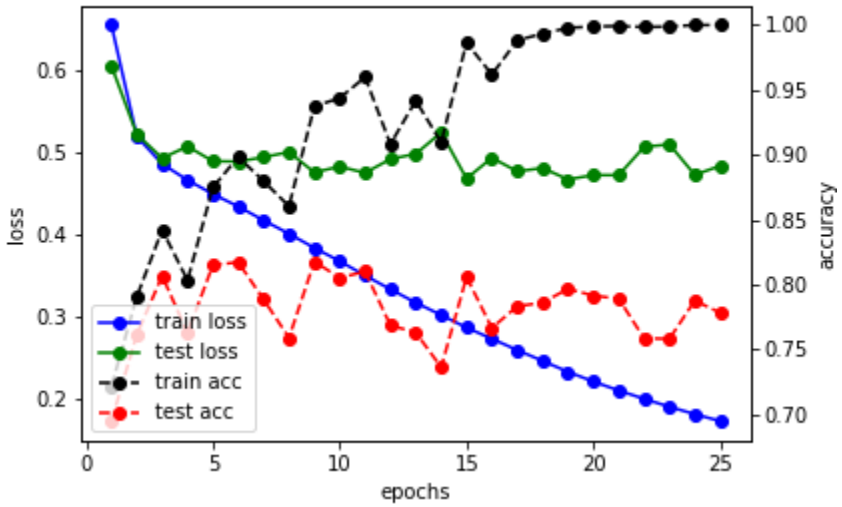

Fig. 4.5: CNN training performance over epochs for learning exon sequences

4.5 Discussion

We set out to determine if deep neural networks can outperform classic models such as PWMs, Markov models, and WAMs. One of the key features of a PWM is that it has so few parameters. A 42 nt PWM has only 168 parameters (4 bp at each position). This makes it simple to train and difficult to overtrain. However, it also means that when you have a wealth of data, the model doesn't capture the inherent richness. Each position of a PWM is completely independent of every other position. However, in a biological context it may matter very much if position 1 is an A and position 5 is a T. Biological entities are not strings of letters, but chemical structures. Given more data, it may be possible to identify subtle patterns if the model is capable of capturing those patterns. WAMs offer limited local context, and we found that they out-performed PWMs. More importantly neural networks out-performed both PWMs and WAMs for splice donor/acceptor classification. The advantage of neural networks is that they can find dependencies among any of the 42 bp used to model splice sites. The disadvantage is that it takes a lot of data to train and optimize the model.

In the case of exon/intron classification, neural networks could not beat a 5th order markov model. It was not anticipated that MLPs would be successful at exon/intron classification as they are not invariant to translations in the input data, i.e if the sequence shifts even by 1 nucleotide the network “sees” a complete different sequence because nodes in the MLP make connections to specific positions in the input sequence. However, CNNs had the promise of working well in intron/exon classification

because CNNs can learn patterns of nucleotides in a translation invariant manner. The CNNs we built were overtrained, resulting in a perfect training set accuracy after 24 epochs. However, there are mechanisms to handle overtraining through stiffer regularization weights, and this will be tested in future directions.

While "beating the classical models" is gratifying, there is a more important consequence of our studies. We show that splice sites aren't simply recognized by individual nucleotide preferences. There are underlying patterns that may be biologically important. For example, there may be different classes of splice sites and different ways of recognizing them. While a PWM could never identify these classes, the neural network may have done so. Identifying exactly what the neural network found is a task for another day. While future studies are needed, even a 2% difference in accuracy can help call thousands more splices accurately when a neural network model is applied across a whole genome.

4.6 Conclusion and Future Directions

In splice site classification using real genomic sequences from *C. elegans*, MLPs had the best performance. All models built in this study saw improved performance on highly used sequences. In the future we will work to deploy the trained models in this study to sequence windows along the *C. elegans* genome. This deployment of the MLPs is a significant test of their feasibility for use in more practical genomic settings. The abundance of sequence in genomics opens the possibility for many false positives to be generated with MLPs, and any machine learning model. Observing how MLPs perform

in a genome-wide context is an important next step. Additionally, the training of CNNs can be improved to reduce the problem of overfitting by adjusting the regularization hyperparameter.

5. Predicting NMR Chemical Shifts With Graph Kernels

5.1 Introduction

Nuclear Magnetic Resonance spectroscopy takes advantage of the quantum mechanical properties of atomic nuclei to measure the structure and motions of chemical molecules ranging from small organic molecules, to proteins and biomolecules, all the way to medical imaging(Berjanskii and Wishart, 2017; Wuthrich, 1986). In structural biology, NMR is a critical experimental tool to interrogate the structure and motions of biomolecules(Curry, 2015). The chemical shift is the 'milepost' in protein NMR. Protein NMR spectroscopists seek to assign chemical shifts observed in their protein/biomolecular sample to specific atoms in the biomolecule in order to build an atomic and dynamic picture of the system they are studying.

As described in Section 1.1, the chemical shift is a relative frequency measurement that records the relative frequency difference to some molecule standard, internal or external to the sample. The Protein NMR community has a rich history in the prediction of chemical shifts with machine learning. Two of the more accurate predictors are ShiftX2(Han et al., 2011, p. 2) and UCBShift(Li et al., 2020) for proteins. Table 5.1 lists out the RMSE performance for ShiftX2 and UCBShift on protein backbone atoms taken from Li et al (2020)(Li et al., 2020). There still remains the need for improved chemical shift predictions because even small errors in chemical shifts can lead to errors in many

structural factors for proteins(Berjanskii and Wishart, 2017; Chen et al., 2018; Wishart et al., 1995; Zhang et al., 2003).

Protein NMR Chemical shift predictors (RMSE)

Predictor	N	H	HA	CA
ShiftX2	2.40±0.02	0.44±0.003	0.23±0.003	1.05±0.01
UCBShift	1.81±0.02	0.31±0.02	0.19±0.002	0.81±0.02

Chemical shift predictors on small chemical molecules (MAE)

Predictor	1H	13C
HOSE	0.33	2.85
MPNN	0.224±0.002	1.355±0.022

Table 5.1: Performance of NMR chemical shift predictors in protein and small molecules

Small organic molecules are an important learning task for chemical shift prediction, and a useful testing ground of empirical chemical shift predictors for biomolecules. As the chemical shift is largely a local electronic environment effect, small chemical systems may be suitable training/testing environments for designing novel machine learning architectures to predict biomolecular chemical shifts. The purpose of the work presented in this chapter is to test how marginalized graph kernels combined with Gaussian Process Regression can be leveraged to predict chemical shifts.

Chemical shift prediction in small molecules has also garnered vast interest. Table 5.1 highlights two predictors in the small molecule space, HOSE codes(Bremser, 1978)**,** and message passing neural networks(Kwon et al., 2020). HOSE codes is a way of predicting chemical shifts from a database of known chemical shifts and molecules. The neural message passing from Kwon et al uses deep neural networks to predict chemical shifts using molecular structures and chemical descriptors.

Marginalized graph kernels are a type of graph kernel that computes the similarity between two graph objects. Graphs are collections of nodes and edges, and are an ideal object to represent chemical molecules, from small organic molecules all the way to large proteins. Graphs can represent diverse and rich data types, where attributes can be placed on nodes of a graph, e.g mass of an atom, and weights/descriptors on edges, e.g distance of molecular bond type. The task is to learn from the graph topologies and data features to accurately predict some property or properties of the graph. There are several types of graph kernels, but in this study we are exploring

marginalized graph kernels. The concept of molecules as graphs and an illustration of the marginalized graph kernel is presented in Figure 5.2.

Marginalized graph kernels (MGKs) have been applied to the prediction of total molecular energy(Tang and de Jong, 2019). In this example, MGKs are used as functions to compute similarity between graphs, and use those similarities to predict a molecule's total energy. A molecule's total energy would be a property of the whole graph. However, in chemical shift prediction, the properties we seek to predict are on the nodes of the graph as chemical shifts are properties of atoms/nodes of a molecule/graph. Marginalized graph kernels take the approach that similarity between graphs should be a function of the number of shared random walk paths between the graphs, as illustrated in Figure 5.2 with the different types of walks drawn on the right side of the figure.

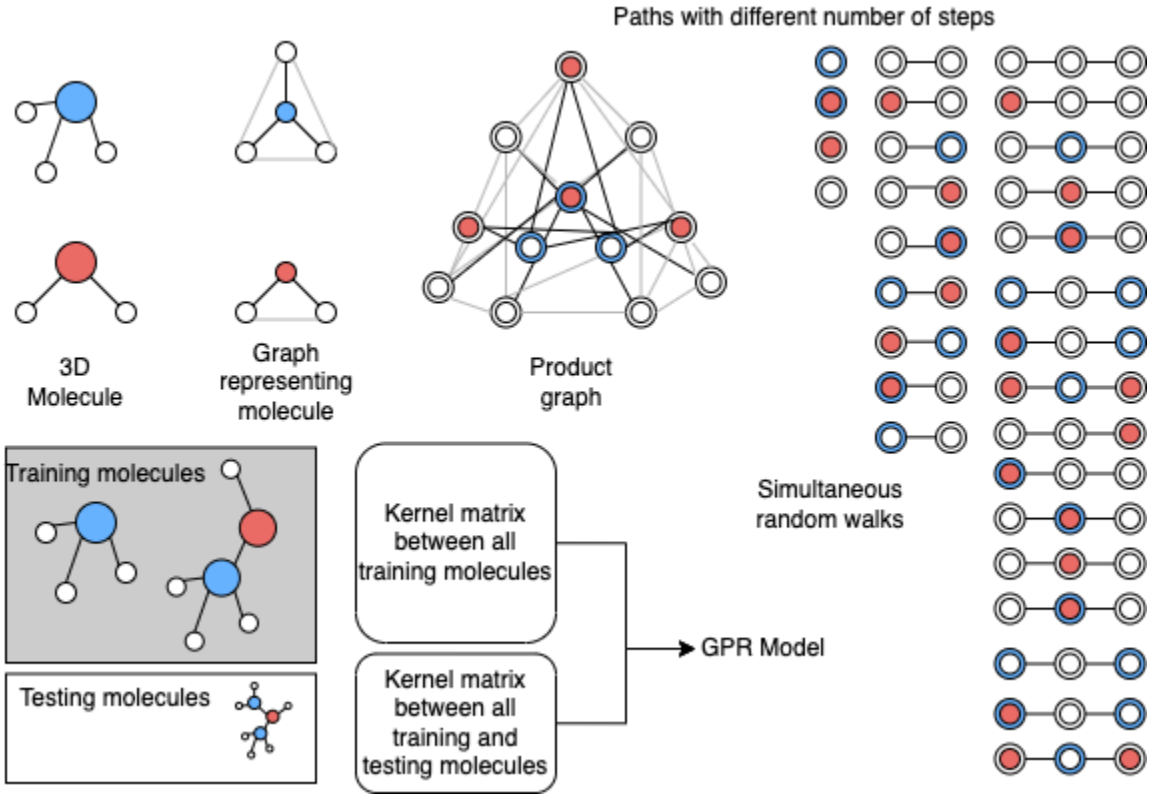

Fig. 5.1: The marginalized graph kernel

Here we illustrate how chemical molecules are abstracted to graphs. Then graphs are “multiplied” together in order to find the number of shared random walks between the two graphs The marginalized graph kernel prescribes a procedure to measure the path similarity between two graphs. In our learning with marginalized graph kernels, we are using Gaussian process regression. We have a training set of molecules with known chemical shifts, and the kernel matrix between all pairs of graphs in the training set is computed. To make predictions on a held-out set of molecules (or testing set), we first compute the similarities between all testing set molecules and training set molecules. These two kernel matrices are used in the GPR model.

Graph kernels are only part of the procedure for chemical shift prediction, or estimation of any graph property. There needs to be a procedure to take the matrix of all pairwise kernel values into the properties we want to predict. In this study we use the Gaussian Process Regression (GPR) method that assumes each multivariate normal distribution over the training dataset. The GPR is a nonparametric, probabilistic regression model that uses the kernel to devise the covariance matrix between training samples.

5.2 Materials and Methods

We gathered molecules with 3D coordinates and chemical shifts from the NMRShiftDB2 database, available at https://nmrshiftdb.nmr.uni-koeln.de/. We split the NMRShiftDB2 dataset into training and testing sets, 21,509 molecules for training and 5,386 molecules for testing. To test the marginalized graph kernel approach, we optimized graph kernels to predict carbon chemical shifts. We had 213,507 shifts for training, and 53,259 shifts for testing. Figure 5.2 shows the distribution of carbon chemical shifts in our dataset.

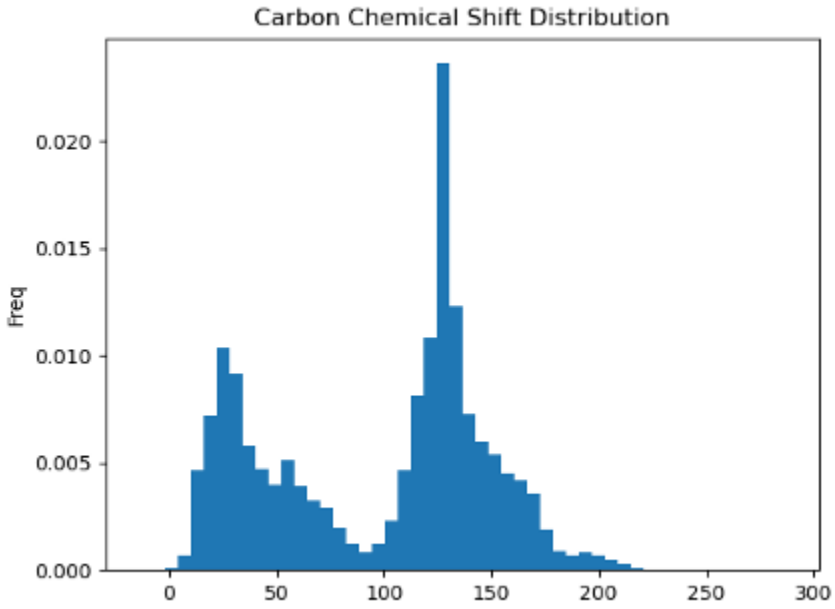

Fig. 5.2: Histogram distribution of carbon chemical shifts in the NMRShiftDB2 dataset

We built the graphs representing the molecules in our dataset in two ways. We first used just the atom element identity for every atom in the molecule and the 3D coordinates of every atom. This first way just uses the atoms and the coordinates for the molecule. The second way we built graphs from molecules was to use the coordinates and chemical features. Table 5.3 lists the chemical features we are using, features like atom hybridization, aromaticity, charge, bond type, bond conjugation, and several more.

Node features	Edge features
element	length
aromatic	aromatic
hybridization	conjugated
chiral	stereo-chemistry
charge	bond type
hydrogen count	in-ring
degree	
donor	
acceptor	

Table 5.2: Chemical features computed for molecules in our dataset

We are using the GraphDot Python library to build and optimize graph kernels as described previously(Tang et al., 2020; Tang and de Jong, 2019)**.** Gaussian process regression is the regression method to convert the pairwise graph similarities information into a model that estimates chemical shifts. Given a set *X* of training graphs with *y* chemical shifts, and *Z* testing graphs, the Gaussian Process Regressor can estimate the testing chemical shifts in the following manner:

$$E_Z = K_{D,Z}^T K_{D,D}^{-1} y_D$$

Where $K_{D,Z}$ is the kernel matrix between the training graphs *D* and testing graphs *Z*, and $K_{D,D}$ is the square kernel matrix between all training graphs.

Marginalized graph kernels have parameters that need to be optimized through the GPR procedure. The marginalized graph kernel parameters are probabilities that relate to starting and stopping probabilities for walks around the molecular graphs, the length scale for comparing distances between atoms in a molecule, and parameters that weight qualitative features like atomic element, bond type, etc. Qualitative features, i.e features that are symbols or categorical values are compared through a Kronecker Delta function, which returns one if two categorical variables are the same, or a parameter *h* that is learned. In the case where molecule graphs are made just from atomic elements and coordinates, there are four parameters to learn (starting/stopping probability, length scale, $h_{element}$). In the case where molecular features from Table 5.2 are incorporated into the graphs, there are a total 19 parameters to optimize. Parameters are optimized by maximizing the log likelihood from the GPR model, with gradient calculations handled through the GraphDot library.

5.3 Results

The results of our study are presented in Table 5.3. Our main objective was to test how addition of molecular descriptors, like atom hybridization, bond type, etc influenced the GPR training and performance. We tested several models with different amounts of training set graphs. The kernel matrices that need to be computed in the GPR construction scale quadratically with the total number of nodes in the training set, not the number of graphs. Training on graphs with larger than 10,000 shifts requires a great deal of memory to compute. So we tested on training set sizes ranging from 200 to 800 molecules, and evaluated on 2,000 molecules. We trained on graphs with/without chemical features.

Train graphs	Test graphs	Graph Type	Train shifts	Test shifts	In-mae (ppm)	Out-mae (ppm)	In-rmse (ppm)	Out-rmse (ppm)
700	2000	CF	18665	54415	2.314	3.501	3.358	5.729
400	2000	CF	10272	53770	2.3906	3.662	3.4496	5.867
1000	2000	CF	26308	54243	1.5428	3.742	2.2249	6.268
800	2000	CF	20876	54351	0.0785	3.783	0.1878	6.411
800	2000	CF	20876	54351	1.5469	3.875	2.2192	6.468
400	2000	CF	10272	53770	0.0831	4.143	0.2139	6.964
200	2000	CF	5072	53601	2.4391	4.257	3.5031	6.667
400	2000	CF	10272	53770	0.0421	4.432	0.1575	7.444
1000	2000	3D	26308	54243	3.2826	4.459	5.0620	7.126
800	2000	CF	20876	54351	3.3277	4.648	5.1013	7.461
200	2000	CF	5072	53601	0.0597	4.754	0.0945	7.817
200	2000	CF	5072	53601	0.0024	4.868	0.0039	8.070
100	2000	CF	2407	53775	2.0784	5.249	2.9129	8.314
800	2000	3D	20876	54351	0.0307	5.253	0.1488	8.447
100	2000	CF	2407	53775	0.0016	5.478	0.0028	8.873
100	2000	CF	2407	53775	0.0523	5.485	0.0892	8.869
400	2000	3D	10272	53770	3.4784	5.820	5.1228	9.323
400	2000	3D	10272	53770	0.0240	5.962	0.1648	9.799
200	2000	3D	5072	53601	3.4489	6.787	4.8817	10.537
800	2000	3D	20876	54351	0.0775	7.489	0.1800	11.679
400	2000	3D	10272	53770	0.0729	8.130	0.1984	12.685
100	2000	3D	2407	53775	3.0112	8.462	4.1959	12.758
200	2000	3D	5072	53601	0.0012	8.847	0.0016	13.851
200	2000	3D	5072	53601	0.0546	8.848	0.0793	13.851
100	2000	3D	2407	53775	0.0012	9.406	0.0017	14.337
100	2000	3D	2407	53775	0.0557	9.448	0.0794	14.410

Table 5.3: Results for training GPR with different molecular graph types and training set sizes

Graph type CF refers to chemical features, 3D refers to only 3D coordinates and element information. The table is sorted by testing set mean absolute error (MAE).

From Table 5.3, we can see that GPRs trained on graphs including chemical features perform better than GPRs trained without chemical features. There is a weak tendency for models trained with more graphs to perform better, but there are exceptions. Figure 5.3 presents representative plots predictive shifts versus ground truth shifts from training and testing set graphs.

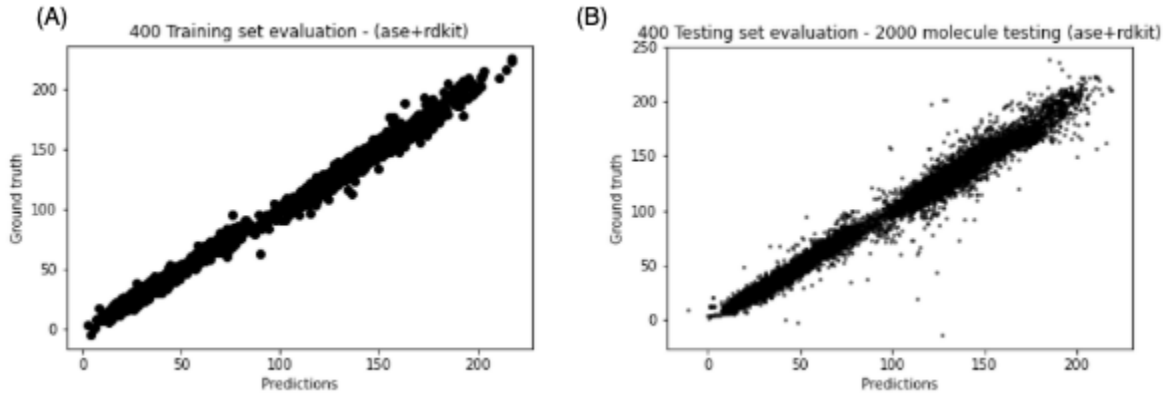

Fig. 5.3: Predicted versus ground truth plots for trained GPRs

(A) Training set predictions versus ground truth. (B) Testing set predictions versus ground truth. In this example, the training set size was 400 graphs, 2000 graphs for testing.

5.4 Discussion

We successfully built molecular graphs that encoded chemical features such as bond type, atomic hybridization, etc into a GPR model for chemical shift prediction. Bond types, aromaticity, hydrogen bond donors/acceptors, etc., all have important effects on the local electronic environment around a nuclei's chemical shift. Ideally, one would like to learn a suitable model where only positions of every atom are needed to produce reliable chemical shift predictions. However, in many empirical chemical shift predictors, chemical/biomolecular features are used to infer the chemical shift of NMR active nuclei in a system. For instance, ShiftX2 uses hand curated features to engineer feature vectors that are then fed through an ensemble of machine learning models to produce accurate chemical shift predictions. Some of the features are protein secondary structure identity, backbone torsion angles, hydrogen bond lengths, pH, temperature, distance to ring systems, etc. Neural message passing from Kwon et al (2020) also used chemical molecular features to construct feature vectors for every atom in their dataset, also from NMRShiftDB2. In our example, addition of chemical features produces at least 1.1 ppm reduction in mean absolute error.

Training set size is a major limitation to the work presented here. With 700 molecules in the training set, ~18k chemical shifts, achieved the best performance in our tests. Neural message passing was able to train on ~21k molecules, ~550k chemical shift observations. Marginalized graph kernels thus present a kernel method that can use training data efficiently. Ultimately, to improve model performance will require

innovations on how to train with more graphs. We have developed a batching mechanism that can compute the kernel matrices needed for the GPR in tiles, which is a potential next step in improving the methods developed here.

Another future avenue of exploration is application of marginalized graph kernels on protein structures. Yet, protein structures are vastly larger systems than the small molecules in the NMRShiftDB2 database. However, there is a potential to carve protein structures into short structural fragments, construct molecular graphs out of the fragments, and be able to use the same tools developed here. We are actively investigating the prospects of applying marginalized graph kernels to st ructural fragments. The graph approach to this problem is also attractive for the application to protein structural fragments because other features can be placed on top of the graphs constructed from proteins, like solvent accessibility, secondary structure status, pH, and other features known to influence the structure of proteins. These new features can be efficiently incorporated into graphs representing proteins. Separately, there is a need to test how accurate GPRs trained on small molecules can be directly applied to protein structural fragments. Given chemical shifts are largely driven by local effects, it stands to reason that GPRs trained in the regime of small molecules can be transferred to proteins. This needs to be tested.

5.5 Conclusion

Here, we presented marginalized graph kernels optimized through the Gaussian Process Regression procedure to estimate small molecule chemical shifts. We

demonstrated that incorporating chemical features into the molecular graphs abstracting the molecules in our dataset improved the GPRs performance measured by mean absolute error by at least 1.1 ppm for carbon shifts. The training set size is a limiting factor in the training of marginalized graph kernels. Chemical shifts are properties of nodes of graphs, so the kernel matrix that is constructed scales quadratically with the number of atoms in the training set. This places a steep memory cost on optimizing the marginalized graph kernels for chemical shift prediction tasks, or any nodal prediction tasks. New methods to train on more graphs is required to make continued gains in chemical shift prediction accuracy using graph kernels.

References

A. Bratholm, L., H. Jensen, J., 2017. Protein structure refinement using a quantum mechanics-based chemical shielding predictor. Chem. Sci. 8, 2061–2072. https://doi.org/10.1039/C6SC04344E

Albaradei, S., Magana-Mora, A., Thafar, M., Uludag, M., Bajic, V.B., Gojobori, T., Essack, M., Jankovic, B.R., 2020. Splice2Deep: An ensemble of deep convolutional neural networks for improved splice site prediction in genomic DNA. Gene X 5. https://doi.org/10.1016/j.gene.2020.100035

AlQuraishi, M., 2019. ProteinNet: a standardized data set for machine learning of protein structure. BMC Bioinformatics 20, 311. https://doi.org/10.1186/s12859-019-2932-0

Artificial intelligence in structural biology is here to stay, 2021. . Nature 595, 625–626. https://doi.org/10.1038/d41586-021-02037-0

Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G.R., Wang, J., Cong, Q., Kinch, L.N., Schaeffer, R.D., Millán, C., Park, H., Adams, C., Glassman, C.R., DeGiovanni, A., Pereira, J.H., Rodrigues, A.V., van Dijk, A.A., Ebrecht, A.C., Opperman, D.J., Sagmeister, T., Buhlheller, C., Pavkov-Keller, T., Rathinaswamy, M.K., Dalwadi, U., Yip, C.K., Burke, J.E., Garcia, K.C., Grishin, N.V., Adams, P.D., Read, R.J., Baker, D., 2021. Accurate prediction of protein structures and interactions using a three-track neural network. Science 373, 871–876. https://doi.org/10.1126/science.abj8754

Baran, M.C., Moseley, H.N.B., Aramini, J.M., Bayro, M.J., Monleon, D., Locke, J.Y., Montelione, G.T., 2006. SPINS: A laboratory information management system for organizing and archiving intermediate and final results from NMR protein structure determinations. Proteins Struct. Funct. Bioinforma. 62, 843–851. https://doi.org/10.1002/prot.20840

Baran, M.C., Moseley, H.N.B., Sahota, G., Montelione, G.T., 2002. SPINS: standardized protein NMR storage. A data dictionary and object-oriented relational database for archiving protein NMR spectra. J. Biomol. NMR 24, 113–121. https://doi.org/10.1023/a:1020940806745

Berjanskii, M., Arndt, D., Liang, Y., Wishart, D.S., 2015. A robust algorithm for optimizing protein structures with NMR chemical shifts. J. Biomol. NMR 63, 255–264. https://doi.org/10.1007/s10858-015-9982-z

Berjanskii, M.V., Neal, S., Wishart, D.S., 2006. PREDITOR: a web server for predicting protein torsion angle restraints. Nucleic Acids Res. 34, W63-69. https://doi.org/10.1093/nar/gkl341

Berjanskii, M.V., Wishart, D.S., 2017. Unraveling the meaning of chemical shifts in protein NMR. Biochim. Biophys. Acta Proteins Proteomics 1865, 1564–1576. https://doi.org/10.1016/j.bbapap.2017.07.005

Berjanskii, M.V., Wishart, D.S., 2013. A simple method to measure protein side-chain mobility using NMR chemical shifts. J. Am. Chem. Soc. 135, 14536–14539. https://doi.org/10.1021/ja407509z

Berjanskii, M.V., Wishart, D.S., 2008. Application of the random coil index to studying

protein flexibility. J. Biomol. NMR 40, 31–48. https://doi.org/10.1007/s10858-007-9208-0
Bertone, P., Kluger, Y., Lan, N., Zheng, D., Christendat, D., Yee, A., Edwards, A.M., Arrowsmith, C.H., Montelione, G.T., Gerstein, M., 2001. SPINE: an integrated tracking database and data mining approach for identifying feasible targets in high-throughput structural proteomics. Nucleic Acids Res. 29, 2884–2898. https://doi.org/10.1093/nar/29.13.2884
Bremser, W., 1978. Hose — a novel substructure code. Anal. Chim. Acta 103, 355–365. https://doi.org/10.1016/S0003-2670(01)83100-7
Case, D.A., 2013. Chemical shifts in biomolecules. Curr. Opin. Struct. Biol. 23. https://doi.org/10.1016/j.sbi.2013.01.007
Chen, X., Smelter, A., Moseley, H.N.B., 2018. Automatic 13C chemical shift reference correction for unassigned protein NMR spectra. J. Biomol. Nmr 72, 11–28. https://doi.org/10.1007/s10858-018-0202-5
Cheung, M.-S., Maguire, M.L., Stevens, T.J., Broadhurst, R.W., 2010. DANGLE: A Bayesian inferential method for predicting protein backbone dihedral angles and secondary structure. J. Magn. Reson. San Diego Calif 1997 202, 223–233. https://doi.org/10.1016/j.jmr.2009.11.008
Cobas, C., 2020. NMR signal processing, prediction, and structure verification with machine learning techniques. Magn. Reson. Chem. MRC 58, 512–519. https://doi.org/10.1002/mrc.4989
Curry, S., 2015. Structural Biology: A Century-long Journey into an Unseen World. Interdiscip. Sci. Rev. 40, 308–328. https://doi.org/10.1179/0308018815Z.000000000120
Delaglio, F., Grzesiek, S., Vuister, G.W., Zhu, G., Pfeifer, J., Bax, A., 1995. NMRPipe: A multidimensional spectral processing system based on UNIX pipes. J. Biomol. NMR 6, 277–293. https://doi.org/10.1007/BF00197809
ECMA-404, n.d. . Ecma Int. URL https://www.ecma-international.org/publications-and-standards/standards/ecma-404/ (accessed 1.6.22).
Ejigu, G.F., Jung, J., 2020. Review on the Computational Genome Annotation of Sequences Obtained by Next-Generation Sequencing. Biology 9, 295. https://doi.org/10.3390/biology9090295
Gill, S.C., von Hippel, P.H., 1989. Calculation of protein extinction coefficients from amino acid sequence data. Anal. Biochem. 182, 319–326. https://doi.org/10.1016/0003-2697(89)90602-7
Goh, C.-S., Lan, N., Echols, N., Douglas, S.M., Milburn, D., Bertone, P., Xiao, R., Ma, L.-C., Zheng, D., Wunderlich, Z., Acton, T., Montelione, G.T., Gerstein, M., 2003. SPINE 2: a system for collaborative structural proteomics within a federated database framework. Nucleic Acids Res. 31, 2833–2838. https://doi.org/10.1093/nar/gkg397
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
Gutmanas, A., Adams, P.D., Bardiaux, B., Berman, H.M., Case, D.A., Fogh, R.H., Güntert, P., Hendrickx, P.M.S., Herrmann, T., Kleywegt, G.J., Kobayashi, N., Lange, O.F., Markley, J.L., Montelione, G.T., Nilges, M., Ragan, T.J., Schwieters, C.D., Tejero, R., Ulrich, E.L., Velankar, S., Vranken, W.F., Wedell, J.R., Westbrook, J., Wishart, D.S., Vuister, G.W., 2015. NMR Exchange Format: a

unified and open standard for representation of NMR restraint data. Nat. Struct. Mol. Biol. 22, 433–434. https://doi.org/10.1038/nsmb.3041
Hafsa, N.E., Arndt, D., Wishart, D.S., 2015. Accessible surface area from NMR chemical shifts. J. Biomol. NMR 62, 387–401. https://doi.org/10.1007/s10858-015-9957-0
Halevy, A., Norvig, P., Pereira, F., 2009. The Unreasonable Effectiveness of Data. IEEE Intell. Syst. 24, 8–12. https://doi.org/10.1109/MIS.2009.36
Han, B., Liu, Y., Ginzinger, S.W., Wishart, D.S., 2011. SHIFTX2: significantly improved protein chemical shift prediction. J. Biomol. Nmr 50, 43–57. https://doi.org/10.1007/s10858-011-9478-4
Haquin, S., Oeuillet, E., Pajon, A., Harris, M., Jones, A.T., van Tilbeurgh, H., Markley, J.L., Zolnai, Z., Poupon, A., 2008. Data management in structural genomics: an overview. Methods Mol. Biol. Clifton NJ 426, 49–79. https://doi.org/10.1007/978-1-60327-058-8_4
Hoch, J.C., 2019. If Machines Can Learn, Who Needs Scientists? J. Magn. Reson. San Diego Calif 1997 306, 162–166. https://doi.org/10.1016/j.jmr.2019.07.044
Jones, C.E., Brown, A.L., Baumann, U., 2007. Estimating the annotation error rate of curated GO database sequence annotations. BMC Bioinformatics 8, 170. https://doi.org/10.1186/1471-2105-8-170
Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S.A.A., Ballard, A.J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A.W., Kavukcuoglu, K., Kohli, P., Hassabis, D., 2021. Highly accurate protein structure prediction with AlphaFold. Nature 1–7. https://doi.org/10.1038/s41586-021-03819-2
Karunanithy, G., Hansen, D.F., 2021. FID-Net: A versatile deep neural network architecture for NMR spectral reconstruction and virtual decoupling. J. Biomol. NMR. https://doi.org/10.1007/s10858-021-00366-w
Klukowski, P., Augoff, M., Zieba, M., Drwal, M., Gonczarek, A., Walczak, M.J., 2018. NMRNet: a deep learning approach to automated peak picking of protein NMR spectra. Bioinforma. Oxf. Engl. 34, 2590–2597. https://doi.org/10.1093/bioinformatics/bty134
Kohlhoff, K.J., Robustelli, P., Cavalli, A., Salvatella, X., Vendruscolo, M., 2009. Fast and accurate predictions of protein NMR chemical shifts from interatomic distances. J. Am. Chem. Soc. 131, 13894–13895. https://doi.org/10.1021/ja903772t
König, S., Romoth, L., Stanke, M., 2018. Comparative Genome Annotation. Methods Mol. Biol. Clifton NJ 1704, 189–212. https://doi.org/10.1007/978-1-4939-7463-4_6
Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., Moult, J., 2021. Critical assessment of methods of protein structure prediction (CASP)-Round XIV. Proteins. https://doi.org/10.1002/prot.26237
Kwon, Y., Lee, D., Choi, Y.-S., Kang, M., Kang, S., 2020. Neural Message Passing for NMR Chemical Shift Prediction. J. Chem. Inf. Model. 60, 2024–2030. https://doi.org/10.1021/acs.jcim.0c00195

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539
Lee, W., Tonelli, M., Markley, J.L., 2015. NMRFAM-SPARKY: enhanced software for biomolecular NMR spectroscopy. Bioinforma. Oxf. Engl. 31, 1325–1327. https://doi.org/10.1093/bioinformatics/btu830
Lehtivarjo, J., Hassinen, T., Korhonen, S.-P., Peräkylä, M., Laatikainen, R., 2009. 4D prediction of protein (1)H chemical shifts. J. Biomol. NMR 45, 413–426. https://doi.org/10.1007/s10858-009-9384-1
Li, D.-W., Hansen, A.L., Yuan, C., Bruschweiler-Li, L., Brüschweiler, R., 2021. DEEP picker is a deep neural network for accurate deconvolution of complex two-dimensional NMR spectra. Nat. Commun. 12, 5229. https://doi.org/10.1038/s41467-021-25496-5
Li, D.-W., Meng, D., Brüschweiler, R., 2015. Reliable resonance assignments of selected residues of proteins with known structure based on empirical NMR chemical shift prediction. J. Magn. Reson. San Diego Calif 1997 254, 93–97. https://doi.org/10.1016/j.jmr.2015.02.013
Li, J., Bennett, K.C., Liu, Y., Martin, M.V., Head-Gordon, T., 2020. Accurate prediction of chemical shifts for aqueous protein structure on “Real World” data. Chem. Sci. 11, 3180–3191. https://doi.org/10.1039/c9sc06561j
Luo, J., Zeng, Q., Wu, K., Lin, Y., 2020. Fast reconstruction of non-uniform sampling multidimensional NMR spectroscopy via a deep neural network. J. Magn. Reson. San Diego Calif 1997 317, 106772. https://doi.org/10.1016/j.jmr.2020.106772
Maciejewski, M.W., Schuyler, A.D., Gryk, M.R., Moraru, I.I., Romero, P.R., Ulrich, E.L., Eghbalnia, H.R., Livny, M., Delaglio, F., Hoch, J.C., 2017. NMRbox: A Resource for Biomolecular NMR Computation. Biophys. J. 112, 1529–1534. https://doi.org/10.1016/j.bpj.2017.03.011
Markwick, P.R.L., Cervantes, C.F., Abel, B.L., Komives, E.A., Blackledge, M., McCammon, J.A., 2010. Enhanced Conformational Space Sampling Improves the Prediction of Chemical Shifts in Proteins. J. Am. Chem. Soc. 132, 1220–1221. https://doi.org/10.1021/ja9093692
Marsh, J.A., Singh, V.K., Jia, Z., Forman-Kay, J.D., 2006. Sensitivity of secondary structure propensities to sequence differences between alpha- and gamma-synuclein: implications for fibrillation. Protein Sci. Publ. Protein Soc. 15, 2795–2804. https://doi.org/10.1110/ps.062465306
McAlpine, J.B., Chen, S.-N., Kutateladze, A., MacMillan, J.B., Appendino, G., Barison, A., Beniddir, M.A., Biavatti, M.W., Bluml, S., Boufridi, A., Butler, M.S., Capon, R.J., Choi, Y.H., Coppage, D., Crews, P., Crimmins, M.T., Csete, M., Dewapriya, P., Egan, J.M., Garson, M.J., Genta-Jouve, G., Gerwick, W.H., Gross, H., Harper, M.K., Hermanto, P., Hook, J.M., Hunter, L., Jeannerat, D., Ji, N.-Y., Johnson, T.A., Kingston, D.G.I., Koshino, H., Lee, H.-W., Lewin, G., Li, J., Linington, R.G., Liu, M., McPhail, K.L., Molinski, T.F., Moore, B.S., Nam, J.-W., Neupane, R.P., Niemitz, M., Nuzillard, J.-M., Oberlies, N.H., Ocampos, F.M.M., Pan, G., Quinn, R.J., Reddy, D.S., Renault, J.-H., Rivera-Chávez, J., Robien, W., Saunders, C.M., Schmidt, T.J., Seger, C., Shen, B., Steinbeck, C., Stuppner, H., Sturm, S., Taglialatela-Scafati, O., Tantillo, D.J., Verpoorte, R., Wang, B.-G., Williams, C.M., Williams, P.G., Wist, J., Yue, J.-M., Zhang, C., Xu, Z., Simmler,

C., Lankin, D.C., Bisson, J., Pauli, G.F., 2019. The value of universally available raw NMR data for transparency, reproducibility, and integrity in natural product research. Nat. Prod. Rep. 36, 35–107. https://doi.org/10.1039/c7np00064b
Meiler, J., 2003. PROSHIFT: protein chemical shift prediction using artificial neural networks. J. Biomol. NMR 26, 25–37. https://doi.org/10.1023/a:1023060720156
Montelione, G.T., Nilges, M., Bax, A., Güntert, P., Herrmann, T., Richardson, J.S., Schwieters, C., Vranken, W.F., Vuister, G.W., Wishart, D.S., Berman, H.M., Kleywegt, G.J., Markley, J.L., 2013. Recommendations of the wwPDB NMR Validation Task Force. Struct. Lond. Engl. 1993 21. https://doi.org/10.1016/j.str.2013.07.021
Moon, S., Case, D.A., 2007. A new model for chemical shifts of amide hydrogens in proteins. J. Biomol. NMR 38, 139–150. https://doi.org/10.1007/s10858-007-9156-8
Moosa, S., Amira, P.A., Boughorbel, D.S., 2021. DASSI: differential architecture search for splice identification from DNA sequences. BioData Min. 14, 15. https://doi.org/10.1186/s13040-021-00237-y
Morris, C., 2018. The Life Cycle of Structural Biology Data. Data Sci. J. 17, 26. https://doi.org/10.5334/dsj-2018-026
Morris, C., 2015. PiMS: a data management system for structural proteomics. Methods Mol. Biol. Clifton NJ 1261, 21–34. https://doi.org/10.1007/978-1-4939-2230-7_2
Nwokeoji, A.O., Kilby, P.M., Portwood, D.E., Dickman, M.J., 2017. Accurate Quantification of Nucleic Acids Using Hypochromicity Measurements in Conjunction with UV Spectrophotometry. Anal. Chem. 89, 13567–13574. https://doi.org/10.1021/acs.analchem.7b04000
Opportunities and obstacles for deep learning in biology and medicine | Journal of The Royal Society Interface [WWW Document], n.d. URL https://royalsocietypublishing.org/doi/full/10.1098/rsif.2017.0387 (accessed 10.31.21).
Pearce, R., Zhang, Y., 2021. Deep learning techniques have significantly impacted protein structure prediction and protein design. Curr. Opin. Struct. Biol. 68, 194–207. https://doi.org/10.1016/j.sbi.2021.01.007
Ponko, S.C., Bienvenue, D., 2012. ProteinTracker: an application for managing protein production and purification. BMC Res. Notes 5, 224. https://doi.org/10.1186/1756-0500-5-224
Qu, X., Huang, Y., Lu, H., Qiu, T., Guo, D., Agback, T., Orekhov, V., Chen, Z., 2020. Accelerated Nuclear Magnetic Resonance Spectroscopy with Deep Learning. Angew. Chem. Int. Ed. 59, 10297–10300. https://doi.org/10.1002/anie.201908162
Romero, P.R., Kobayashi, N., Wedell, J.R., Baskaran, K., Iwata, T., Yokochi, M., Maziuk, D., Yao, H., Fujiwara, T., Kurusu, G., Ulrich, E.L., Hoch, J.C., Markley, J.L., 2020. BioMagResBank (BMRB) as a Resource for Structural Biology. Methods Mol. Biol. Clifton NJ 2112, 187–218. https://doi.org/10.1007/978-1-0716-0270-6_14
Salzberg, S.L., 2019. Next-generation genome annotation: we still struggle to get it right. Genome Biol. 20, 92. https://doi.org/10.1186/s13059-019-1715-2
Scalzitti, N., Jeannin-Girardon, A., Collet, P., Poch, O., Thompson, J.D., 2020. A

benchmark study of ab initio gene prediction methods in diverse eukaryotic organisms. BMC Genomics 21, 293. https://doi.org/10.1186/s12864-020-6707-9
Schnoes, A.M., Brown, S.D., Dodevski, I., Babbitt, P.C., 2009. Annotation Error in Public Databases: Misannotation of Molecular Function in Enzyme Superfamilies. PLOS Comput. Biol. 5, e1000605. https://doi.org/10.1371/journal.pcbi.1000605
Sercu, T., Puhrsch, C., Kingsbury, B., LeCun, Y., 2015. Very Deep Multilingual Convolutional Neural Networks for LVCSR.
Shen, Y., Bax, A., 2015. Protein structural information derived from NMR chemical shift with the neural network program TALOS-N. Methods Mol. Biol. Clifton NJ 1260, 17–32. https://doi.org/10.1007/978-1-4939-2239-0_2
Shen, Y., Bax, A., 2013. Protein backbone and sidechain torsion angles predicted from NMR chemical shifts using artificial neural networks. J. Biomol. NMR 56, 227–241. https://doi.org/10.1007/s10858-013-9741-y
Shen, Y., Bax, A., 2012. Identification of helix capping and β-turn motifs from NMR chemical shifts. J. Biomol. NMR 52, 211–232. https://doi.org/10.1007/s10858-012-9602-0
Shen, Y., Bax, A., 2010. SPARTA+: a modest improvement in empirical NMR chemical shift prediction by means of an artificial neural network. J. Biomol. NMR 48, 13–22. https://doi.org/10.1007/s10858-010-9433-9
Tang, Y.-H., de Jong, W.A., 2019. Prediction of Atomization Energy Using Graph Kernel and Active Learning. J. Chem. Phys. 150, 044107. https://doi.org/10.1063/1.5078640
Tang, Y.-H., Selvitopi, O., Popovici, D., Buluç, A., 2020. A High-Throughput Solver for Marginalized Graph Kernels on GPU. 2020 IEEE Int. Parallel Distrib. Process. Symp. IPDPS 728–738. https://doi.org/10.1109/IPDPS47924.2020.00080
The UniProt Consortium, 2021. UniProt: the universal protein knowledgebase in 2021. Nucleic Acids Res. 49, D480–D489. https://doi.org/10.1093/nar/gkaa1100
Tian, Y., Opella, S.J., Marassi, F.M., 2012. Improved chemical shift prediction by Rosetta conformational sampling. J. Biomol. NMR 54, 237–243. https://doi.org/10.1007/s10858-012-9677-7
Ulrich, E.L., Baskaran, K., Dashti, H., Ioannidis, Y.E., Livny, M., Romero, P.R., Maziuk, D., Wedell, J.R., Yao, H., Eghbalnia, H.R., Hoch, J.C., Markley, J.L., 2019. NMR-STAR: comprehensive ontology for representing, archiving and exchanging data from nuclear magnetic resonance spectroscopic experiments. J. Biomol. Nmr 73, 5–9. https://doi.org/10.1007/s10858-018-0220-3
Vila, J.A., Arnautova, Y.A., Martin, O.A., Scheraga, H.A., 2009. Quantum-mechanics-derived 13Cα chemical shift server (CheShift) for protein structure validation. Proc. Natl. Acad. Sci. U. S. A. 106, 16972–16977. https://doi.org/10.1073/pnas.0908833106
Vranken, W.F., Boucher, W., Stevens, T.J., Fogh, R.H., Pajon, A., Llinas, M., Ulrich, E.L., Markley, J.L., Ionides, J., Laue, E.D., 2005. The CCPN data model for NMR spectroscopy: development of a software pipeline. Proteins 59, 687–696. https://doi.org/10.1002/prot.20449
Wishart, D.S., Bigam, C.G., Yao, J., Abildgaard, F., Dyson, H.J., Oldfield, E., Markley, J.L., Sykes, B.D., 1995. 1H, 13C and 15N chemical shift referencing in biomolecular NMR. J. Biomol. NMR 6, 135–140.

https://doi.org/10.1007/BF00211777
Wishart, D.S., Sykes, B.D., 1994. The 13C chemical-shift index: a simple method for the identification of protein secondary structure using 13C chemical-shift data. J. Biomol. NMR 4, 171–180. https://doi.org/10.1007/BF00175245
Wunderlich, Z., Acton, T.B., Liu, J., Kornhaber, G., Everett, J., Carter, P., Lan, N., Echols, N., Gerstein, M., Rost, B., Montelione, G.T., 2004. The protein target list of the Northeast Structural Genomics Consortium. Proteins 56, 181–187. https://doi.org/10.1002/prot.20091
Wuthrich, K., 1986. NMR of proteins and nucleic acids.
Zeng, J., Zhou, P., Donald, B.R., 2013. Hash: a Program to Accurately Predict Protein Hα Shifts from Neighboring Backbone Shifts. J. Biomol. NMR 55. https://doi.org/10.1007/s10858-012-9693-7
Zhang, H., Neal, S., Wishart, D.S., 2003. RefDB: a database of uniformly referenced protein chemical shifts. J. Biomol. NMR 25, 173–195. https://doi.org/10.1023/a:1022836027055
Zhang, Y., Liu, X., MacLeod, J., Liu, J., 2018. Discerning novel splice junctions derived from RNA-seq alignment: a deep learning approach. BMC Genomics 19, 971. https://doi.org/10.1186/s12864-018-5350-1