UNIVERSITY OF TARTU

Institute of Computer Science

Computer Science Curriculum

Karl Kristjan Tamm

Software Tools for Mixture Analysis

Bachelor's Thesis (9 ECTS)

Supervisor(s): Stefan Hermann Kuhn, PhD

Tartu 2024

Software Tools for Mixture Analysis

Abstract:

This thesis explores the development of Nmrfilter v1.5, an advanced version of the existing NMR software designed to analyze chemical mixtures against candidate mixtures. The original Nmrfilter software, primarily created as a demonstration tool rather than a standalone program, uses user-supplied NMR spectral data to compare and identify the substance against candidate compounds. This study identifies some of the limitations of the original program, proposing several improvements. These enhancements aim to transform Nmrfilter into a more accessible and functional tool for analytical chemistry. The thesis presents a comprehensive review of the original program, its value, and limitations, together with implementations of new features including explained rationale behind them.

Keywords:

NMR, automation, scientific software, open source

CERCS:

B110 Bioinformatics, medical informatics, biomathematics, biometrics

P170 Computer science, numerical analysis, systems, control

Lühikokkuvõte:

Lõputöö uurib teadustarkvara Nmrfilter v1.5 arendust, mis on edasiarendus olemasolevast tuumamagnetresonantsspektroskoopia (NMR) analüüsiprogrammist Nmrfilter, loodud automaatselt tuvastamaks NMR andmetest keemilise aine struktuuri. Originaalne programm, Nmrfilter, on pigem loodud näitamaks algoritmi kui mõeldud kasutamiseks eraldiseisva tarkvarana keemilises analüüsis. Nmrfilter võrdleb kasutaja poolt antud NMR andmeid kasutaja poolt täpsustatud keemiliste ühenditega hindamaks, millise ühendiga on tegu. Töö uurib originaalse programmi puudujääke, pakkudes välja mitmeid täiendavaid funktsioone. Täienduste eesmärk on Nmfilter muuta kättesaadavamaks tööriistaks keemilises analüüsis. Töö annab ülevaate originaalselt programmist, selle väärtusest ning piiridest, millele järgnevad edasiarendused koos põhjenduste ja implementatsioonide kirjeldustega.

Võtmesõnad:

Tuumamagnetresonantsspektroskoopia, automatsioon, teadustarkvara, avatud lähtekood

CERCS:

B110 Bioinformaatika, meditsiiniinformaatika, biomatemaatika, biomeetrika

P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Table of Contents

1 Introduction

Chemistry is a field that requires outstanding precision, as even slight variations in the make-up of a molecule may influence its properties and behavior significantly, which can determine its usage, or rather its absence, in various fields, stretching from academia, material science, energy production to agriculture and pharmaceuticals. Therefore, ascertaining a molecule together with its qualities is crucial in utilizing the compound appropriately.

Characteristics, e.g. strength and the melting point of a particular compound are influenced by numerous factors, beginning from the atoms it is made up of to its environment to name a few. Among these is the crystalline structure of the molecule - how atoms are positioned relative to one another and most importantly, which atoms are connected or “bonded” via different chemical bonds. Isomers exhibit the importance of structure greatly, with one famous example being graphite and diamonds - both are entirely made up of carbon atoms, yet the strength and looks differ greatly due to their underlying different crystalline structures. One way of determining the structure of a molecule is a technique called nuclear magnetic resonance spectroscopy.

Nuclear magnetic resonance spectroscopy (NMR) is a powerful and widely used technique in chemical research for determining structures of chemical compounds. The fundamentals of NMR spectroscopy lie in the magnetic properties of atomic nuclei [1], which can be oriented in an external magnetic field using radio transmissions. Measuring the absorption of energy using an RF receiver produces what is called a resonance signal. These signals can be used to generate a spectrum, from which depending on the NMR technique used, different information about the structure of a molecule can be obtained, which ultimately can be used in identifying the structure of a substance.

Nmrfilter[1] is a program created in the collaboration of S. Kuhn, S. Colreavy-Donnelly, J. S. de Souza, and R. M. Borges in order to demonstrate their suggestion of an improved

[1] https://github.com/stefhk3/nmrfilter

software pipeline for mixture analysis [2] from measured NMR data. Nmrfilter requires the user to provide HSQC and HMBC NMR spectral data for the substance they wish to identify and a list of candidate compounds. The provided measured data is processed and compared to the simulated spectra of every candidate compound, resulting in a match rating for each possible candidate, along with a plot visualizing the comparison.

While Nmrfilter is a functional program, its primary purpose appears to be the demonstration of a software pipeline rather than being a standalone tool for usage in chemical analytics. Despite being a simple command line interface program, there is room for improvement for easier and more multifunctional use, potentially allowing more users to access and use it to fulfill their needs in analytical chemistry, The following work aims to achieve this through implementing additional features ranging from better visualization of result data, enabling batch simulation of NMR spectra as a standalone feature, simplifying installation to the beautification of the GitHub repository resulting in an easily accessible piece of open source software. The updated software will be called Nmrfilter v1.5[2].

Chapter 2 contains terms and notations facilitating the reading of the thesis. The main part of the thesis begins in Chapter 3 providing background information about NMR and existing software solutions for processing NMR data. Chapter 4 offers an overview of the existing program alongside a list of identified shortcomings. Following, Chapter 5 expands upon the identified issues, discussing implemented mitigations providing their rationale. Chapter 6 reviews the completed work, detailing what was achieved and what fell short. Chapter 7 explores the future possibilities for Nmrfilter v1.5. Ultimately, the thesis will be summarized in a conclusive Chapter 8. Additionally, the appendix of the thesis contains a glossary section giving definitions for several relevant scientific terms used extensively throughout the thesis.

[2] https://github.com/kulakarla/nmrfilterv15

2 Terms and Notations

^{13}C, ^{1}H – superscript defining the isotopes of atoms.

Namesfile – referring to the file containing corresponding names for the list of candidates provided in the SMILES candidate list.

NMR – abbreviation for “Nuclear magnetic resonance spectroscopy”.

Proton – synonymous with the ^{1}H atom in NMR context.

3 Background

Chemistry is a field where precision is of utmost importance – small differences between substances, whether structural or formular, may change the behavior of a compound severely. Thus, before a chemical makes its way to the shelves at stores or a lab researching pharmaceuticals, the identification of a substance is vital to guarantee it is utilized correctly and safely.

3.1 Substance Identification Techniques

Over time, an array of techniques has been developed to identify chemical substances and their structure with precision and accuracy. Each technique provides unique detailed information about the composition of the molecule. Mass spectrometry (MS) is an established and widespread technique for substance identification, which by measuring the mass to charge ratio of ions enables the calculation of the exact molecular weight, ultimately allowing the identification of a compound via molecular weight. MS is an invasive technique, damaging the sample, therefore possibly limiting further analysis of the sample. Infrared spectroscopy (IR) is another technique for analyzing mixtures. IR provides information about the functional groups present in the molecule by measuring the absorption of infrared radiation at specific wavelengths corresponding to molecular vibrations. IR is a non-invasive technique but provides information mainly about the functional groups present in the molecule, thus limiting the information about the structure of the molecule. Another widely used method for identifying a substance and its underlying crystalline structure is nuclear magnetic spectroscopy or NMR.

3.2 Nuclear Magnetic Spectroscopy

Nuclear magnetic spectroscopy is a non-invasive spectroscopy technique in chemical research for identifying a chemical compound with its underlying crystalline structure. NMR takes advantage of magnetic properties of certain atomic nuclei, which by using radio transmissions can be oriented in an external magnetic field. Measuring the absorption of energy using an RF receiver produces a resonance signal, which can be used to generate a spectrum (Fig. 1), from which various information about a molecule and its structure can be obtained. NMR is a highly flexible chemical analytical tool encompassing a wide range of techniques, each allowing experimenting on different types of samples offering different types of information about the molecule. Some of these developed techniques include 1D NMR, 2D NMR (Fig. 2), Time-Resolved NMR, and many more. 2D NMR is a technique using correlating two different nuclear resonances to provide information about the molecular structure and connectivity between atoms. Some 2D NMR techniques are COSY (Correlation Spectroscopy), showing directly bonded protons, and HSQC/HMBC (Heteronuclear Single Quantum Coherence, Heteronuclear Multiple Bond Correlation), providing information about carbons and their connections to protons, ultimately helping in determining the structure of the molecule.

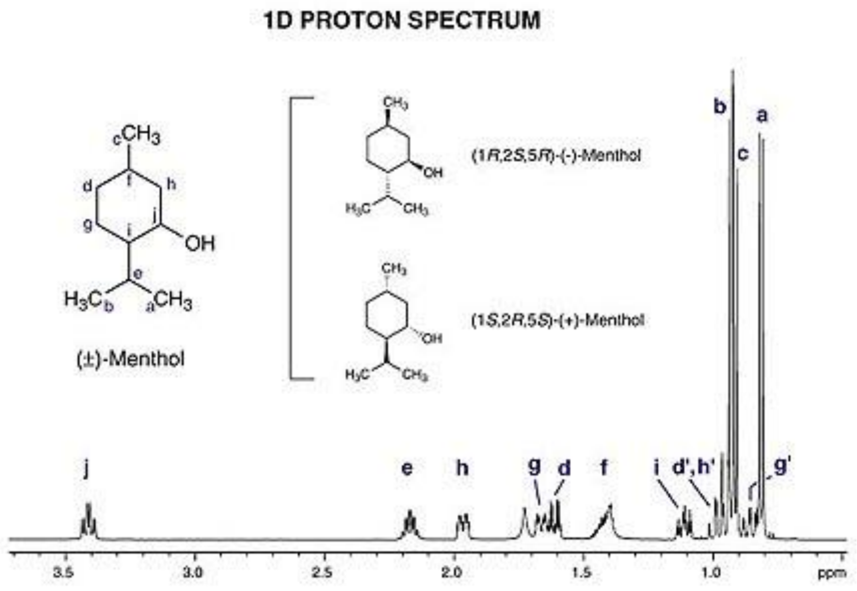


Figure 1. Simple ^{1}H 1D NMR spectrum [3].

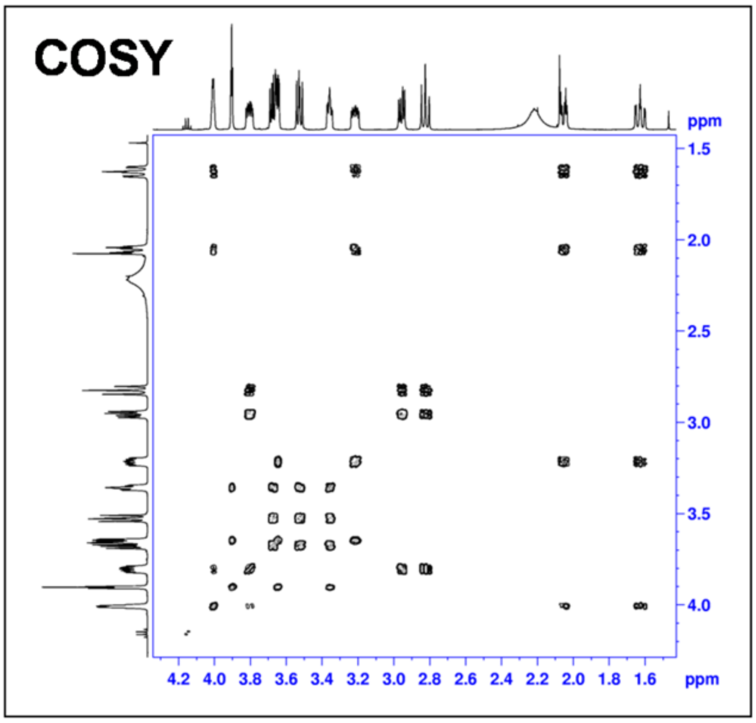


Figure 2. A 2D-NMR COSY spectrum [4].

NMR is a highly flexible technique for chemical analysis, with numerous strengths compared to MS and IR. It is non-invasive and non-destructive, provides detailed information about the connectivity of atoms within the molecule together with the spatial arrangement, the possibility for real-time dynamic studies, the ability to be used with samples in solid, liquid, and gas states, and more. NMR spectroscopy has a wide array of applications, with an eBook series called “Applications of NMR Spectroscopy” [5], published by Atta-ur-Rahman and M. Iqbal Choudhary, totaling in 9 volumes as of 2024. The books detail NMRs various applications stretching from food sciences, medical diagnosis and molecular identification to analysis of the psychological effects of coffee [5]. With a large number of uses, the NMR market is expected to grow by 5.5% annually until 2032 [6], from which it can be implied that the demand for tools processing NMR data will be growing as well.

3.3 NMR Software

With widespread usage due to the abundance of information possible to obtain with NMR, many different software solutions exist for working with NMR data. MestReNova[3] is a large chemistry software solution that provides various software for working with chemical data, including the possibility of identifying and quantifying a compound from NMR spectra. Chenomx NMR Suite[4] is another software suite offering various NMR analysis tools along with a tool to identify a substance from its NMR data using a reference database, albeit it is limited to metabolites, constraining its use. While the mentioned tools and numerous others exist, they are neither free nor inexpensive because of being large software suites. Large software solutions have a high learning curve, possibly making a simple task take unnecessarily long due to the need of navigating through the program's many available tools before finding the needed one, in this case the tool being NMR compound identification and structure elucidation. COLMAR[5] is a simple web-application NMR tool that allows for the identification of compounds. It is free, with downsides being its limitation to primary metabolites and an outdated UI. A closed database can be counted as a weakness as well. However, it does produce an interactive graph after a query, which may ease the process of analyzing the result, which can be considered a strength. NMRium[6] is an online web application for processing NMR spectra. While the input is limited to raw NMR data essentially right off the instrument, it offers various tools for analysis – interactive graphs, automatic cross-peak detection, normalization of data and Fourier transformation to list a few. It is a free, powerful tool for NMR analysis, together with some premium niche services offered for a cost. The interactivity and versatility of possible manipulation and analysis of the data is a strength, it does not offer any options to automatically predict the compound under investigation. It is a tool meant for simplifying the process of determining the compound.

[3] https://mestrelab.com/software/mnova-software/

[4] https://www.chenomx.com/

[5] https://spin.ccic.osu.edu/index.php/colmar

[6] https://www.nmrium.org/

3.4 NMRfilter

Nmrfilter[7] is an open-source software for NMR compound identification developed in the collaboration of Stefan Kuhn, Simon Colreavy-Donnelly, Juliana Santana de Souza, and Ricardo Moreira Borges to demonstrate their improved software pipeline for mixture analysis [2]. It is a simple program with the sole purpose of identifying a substance from its measured NMR spectrum data. The user must provide measured NMR data, a list of candidate substances which the user-measured data will be tested against and ultimately rated, giving the user rankings of the candidates and match ratings based on established criteria.

The biggest strength of Nmrfilter compared to already established software lies in its simplicity – the program has been created with a sole focus on substance identification, removing the possible overwhelmingness of large programs. If you want to predict a substance, just run the program and after a few moments, the results have been generated. The learning curve is minimal. In addition, perhaps the greatest advantage of Nmrfilter, is it being a free, open-source software – it is available for everyone, and the inner workings are extensively described in a published paper. Furthermore, the simulations of candidate spectra use an ever-growing open database called nmrshiftdb2[8], meaning the reference library is continuously being updated with new measured NMR data, bettering the predictions over time.

Although Nmrfilter is a working piece of software, it has been created to demonstrate an algorithm rather than being used as a standalone tool for chemical analysis. By improving the software by bettering the resulting outputs, adding flexibility to its use combined with a solid repository resembling a complete piece of open-source software, it could be opened to a wider userbase to fulfill their needs in analytical chemistry as a free and simple alternative to existing costly comprehensive software bundles.

[7] https://github.com/stefhk3/nmrfilter

[8] https://nmrshiftdb.nmr.uni-koeln.de/

4 Overview of Nmrfilter

Nmrfilter[9] is a program created in the collaboration of S. Kuhn, S. Colreavy-Donnelly, J. S. de Souza, and R. M. Borges to demonstrate their improved software pipeline for mixture analysis [2] for identifying a compound from its measured NMR spectrum, published in the journal Faraday Discussions in 2019. The following chapter describes the overall workflow and features of the program, additionally finding its drawbacks and suggesting improvements. Detailed explanations for the chosen NMR data processing algorithms and their inner workings can be found in the original paper [1] which the program was created for, as the focus of this work will not be on the algorithms.

4.1 Rundown

Nmrfilter has been developed through a combination of bash scripts, Python scripts, and a small Java program for running simulations, with the data transferred between different scripts via text files [2]. The user has to provide measured HMBC and HSQC NMR data together with a list of candidate substances in SMILES. Following, the program finds cross-peaks from the data provided and performs various algorithmic procedures to form a network of clusters. For each candidate substance, a spectrum is also simulated using tools available in nmrshiftdb2[10]. The program then runs a similarity procedure for each candidate and the user-given spectrum. Then, a rating of the match is given for each candidate. The program generates a text file that includes candidates along with their respective similarity ratings in comparison to the provided measured data. Additionally, a plot is produced to visually illustrate the comparison of the spectra. Nmrfilter is a simple command-line interface program, with small extra features available.

[9] https://github.com/stefhk3/nmrfilter

[10] https://nmrshiftdb.nmr.uni-koeln.de/

4.2 Properties

An important file in Nmrfilter is `nmrproc.properties`, which contains various properties that affect the behavior of the program.

Table 1. Properties, their description, and default values.

Property	Description	Default
datadir	Path to the absolute directory containing project folders to be used for input.	
msmsinput	Name of the file containing the list of candidate substances in a project folder.	testall.smi
predictionoutput	Name of the file containing simulated spectra shifts of the candidate substances	resultprediction.csv
result	Name of the output file containing the ratings and match for each candidate.	result.txt
solvent	Name of solvent if used. Choices available are “`Methanol-D4 (CD3OD)`”, “`Chloroform-D1 (CDC13)`” and “`Dimethylsulphoxide-D6 (DMSO-D6, C2D6SO)`”. Otherwise use “`Unreported`”.	Methanol-D4 (CD3OD)
tolerancec	Tolerance for the ^{13}C axis.	0.2
toleranceh	Tolerance for the ^{1}H axis.	0.02
spectuminput	Name of the file containing measured spectrum data.	realspectrum.csv
clusteringoutput	Name of a file created containing initial found cross-peaks.	cluster.txt

rberresolution	Resolution parameter for the RBER algorithm provided by the Louvain library, which changes the size of the clusters. Larger the value, smaller the clusters.	0.2
louvainoutput	Name of a file generated after Louvain clustering.	clusterslouvain.txt
usehsqctocsy	Boolean value. Set `true` if HSQCTOCSY shifts are to be included in the prediction. Specturminput should contain HSQCTOCSY shifts if set `true`	false
usehmbc	Boolean. Define the use of HMBC or not.	true
dotwobonds	Changes the HOSE code prediction to explore 2 spheres instead of the default 3.	false
usedeeplearning	Boolean. If set `true`, uses respredict [7] prediction instead of a HOSE code based one.	false
debug	Produces debug output, e.g. measured unused peaks plotted, additional files inside the project folder	false

4.3 Input and Running

User input has to be provided in the form of a folder (project) that must contain the following required files:

- A list of candidate structures in the form of SMILES, one structure per line. The file name can be configured by the `msmsinput` property.

- Peak lists derived from measured spectra in a .CSV file. The file must be a list of shifts, coordinates separated by tab, with the ^{13}C shift being the first dimension and ^{1}H in the second, with each row corresponding to one shift. HMBC and HSQC shifts should be included as a standard. File name is configured by the property `spectru-minput`.

The program requires Python[11] 3 and Java[12] 1.8 to be installed, along with several packages that include numpy[13], scipy[14], python-igraph[15], Louvain[16], and more. Anaconda[17] virtual environment files are available for easier environment setup.

To produce the result list and plots, run the respective script followed by the project (folder) name. Path to the folder containing the projects should be set before running in the `nmrproc.properties` file. For use in Linux, run `./run-standalone.sh <projectname>`, in Windows `./run-standalone.bat <projectname>`.

4.4 Workflow

The main script of the program, `run-standalone`, executes different scripts sequentially, which ultimately produces the output. First and foremost, the virtual environment in the running directory is checked. If the environment is missing, the necessary tools are installed. Everything resolved, the scripts processing the measured NMR data and simulating candidate data are run. Parts responsible for the majority of the program's work will be explained shortly in subsections.

[11] https://www.python.org/

[12] https://www.java.com/en/

[13] https://numpy.org/

[14] https://scipy.org/

[15] https://python.igraph.org/en/stable/

[16] https://pypi.org/project/python-louvain/

[17] https://www.anaconda.com/

4.4.1 Simulating Spectra for Candidate Substances

For each candidate substance, a spectrum is simulated. This is done by the Java program `simulate.jar`. It uses data from nmrshiftdb2, a long-running NMR database holding organic structures and their NMR spectra. Simulation of the spectra is done by the prediction mechanism provided by nmrshiftdb2, which predicts spectra based on stereo-aware HOSE codes, an extension of HOSE codes developed by Stefan Kuhn and Sean R. Johnson [8]. Hierarchically organized spherical environment code or HOSE code is a way to represent the environment around a single atom, which can be used to predict the chemical shift of a particular atom based on its environment. Prediction is done by finding molecules that have similar neighborhoods around some atom, meaning they share the same HOSE code. Exploring adjacent atoms, the environment is further described. Having found the shifts for each atom, a spectrum can be assembled. The more measured data available in the database for some compound, the better its spectrum prediction.

4.4.2 Clustering for Measured Data

From the measured NMR data provided by the user to be used for the prediction against the candidates, an NMR network of clusters of cross peaks is built. This is done in two parts. First, a simpler clustering is done by `clustering.py`. This Python program identifies peaks coming from the same compound by assuming if two peaks share a shift on one of the axes, they originate from the same compound. Tolerance threshold for finding shared cross peaks across both axes can be configured in the properties file, by `tolerancec` and `toleranceh` properties for the ^{13}C and ^{1}H axes respectively. A shared cross peak is found when two peaks share a chemical shift on either axis. The found cross peaks are put into a single list, with the ^{13}C shift being in the first dimension and the ^{1}H shift in the second dimension.

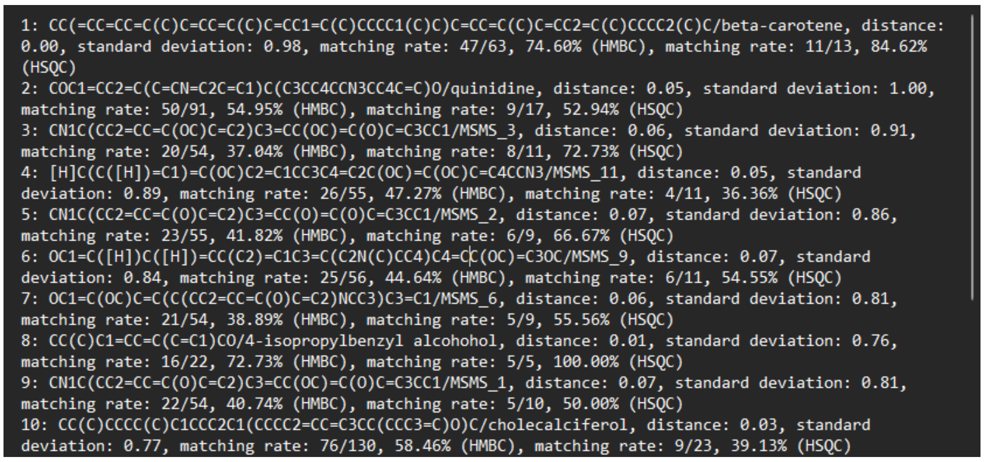
The produced single list of cross peaks will then be processed by the second part of creating the network, implemented in `clusteringlouvain.py`. Using the RBER algorithm provided by the Louvain library, we form communities of cross-peaks, also known as clusters. Every cross-peak from the measured spectra is organized into some cluster.

4.4.3 Similarity

The main goal of Nmrfilter is to find the most likely candidate for the measured NMR data among the candidate compounds. This is done by finding similarities between the two spectra, with the work being done in the Python program `similarity.py`, which also produces the resulting final outputs. This step can be imagined as two spectra, the candidate and the measured, being laid on top of each other. From there, we are trying to find overlapping clusters – we are trying to map/assign one cluster from the measured spectra to one cluster in the simulated spectra. From this comparison, two reliability measures can be extracted: the distance between the mapped cluster and the fraction of the clusters mapped. Both values are normalized to be between 0 (best) and 1 (worst). Added together, the combined similarity measure provides the basis for ranking the candidate's match, with a lower value indicating a better match. Simply put, based on the distances between mapped clusters and the fraction of mapped clusters in the measured and simulated spectra, a ranking is calculated for each candidate.

4.4.4 Output

The final output is generated by `similarity.py`. It generates a text file (name defined by the property `result`) containing the candidate substances and ratings of the match ranked from best to worst (Fig. 3) together with a plot for each candidate visualizing the mapping of clusters between the measured and simulated spectra (Fig. 4). The text result (Fig. 3) gives us the candidate in SMILES together with its name, normalized distance between measured and simulated clusters, standard deviation, and matching rate for clusters between measured and simulated spectra. Plots (Fig. 4), with y being the ^{13}C axis and x being the ^{1}H axis, illustrate the mapping of cross peaks between the measured and simulated HMBC (left) and HSQC (right) spectra.



```
1: CC(=CC=CC=C(C)C=CC=C(C)C=CC1=C(C)CCCC1(C)C)C=CC=C(C)C=CC2=C(C)CCCC2(C)C/beta-carotene, distance:
0.00, standard deviation: 0.98, matching rate: 47/63, 74.60% (HMBC), matching rate: 11/13, 84.62%
(HSQC)
2: COC1=CC2=C(C=CN=C2C=C1)C(C3CC4CCN3CC4C=C)O/quinidine, distance: 0.05, standard deviation: 1.00,
matching rate: 50/91, 54.95% (HMBC), matching rate: 9/17, 52.94% (HSQC)
3: CN1C(CC2=CC=C(OC)C=C2)C3=CC(OC)=C(O)C=C3CC1/MSMS_3, distance: 0.06, standard deviation: 0.91,
matching rate: 20/54, 37.04% (HMBC), matching rate: 8/11, 72.73% (HSQC)
4: [H]C(C([H])=C1)=C(OC)C2=C1CC3C4=C2C(OC)=C(OC)C=C4CCN3/MSMS_11, distance: 0.05, standard
deviation: 0.89, matching rate: 26/55, 47.27% (HMBC), matching rate: 4/11, 36.36% (HSQC)
5: CN1C(CC2=CC=C(O)C=C2)C3=CC(O)=C(O)C=C3CC1/MSMS_2, distance: 0.07, standard deviation: 0.86,
matching rate: 23/55, 41.82% (HMBC), matching rate: 6/9, 66.67% (HSQC)
6: OC1=C([H])C([H])=CC(C2)=C1C3=C(C2N(C)CC4)C4=CC(OC)=C3OC/MSMS_9, distance: 0.07, standard
deviation: 0.84, matching rate: 25/56, 44.64% (HMBC), matching rate: 6/11, 54.55% (HSQC)
7: OC1=C(OC)C=C(C(CC2=CC=C(O)C=C2)NCC3)C3=C1/MSMS_6, distance: 0.06, standard deviation: 0.81,
matching rate: 21/54, 38.89% (HMBC), matching rate: 5/9, 55.56% (HSQC)
8: CC(C)C1=CC=C(C=C1)CO/4-isopropylbenzyl alcohohol, distance: 0.01, standard deviation: 0.76,
matching rate: 16/22, 72.73% (HMBC), matching rate: 5/5, 100.00% (HSQC)
9: CN1C(CC2=CC=C(O)C=C2)C3=CC(OC)=C(O)C=C3CC1/MSMS_1, distance: 0.07, standard deviation: 0.81,
matching rate: 22/54, 40.74% (HMBC), matching rate: 5/10, 50.00% (HSQC)
10: CC(C)CCCC(C)C1CCC2C1(CCCC2=CC=C3CC(CCC3=C)O)C/cholecalciferol, distance: 0.03, standard
deviation: 0.77, matching rate: 76/130, 58.46% (HMBC), matching rate: 9/23, 39.13% (HSQC)
```

Figure 3. Result.txt containing candidate substance rankings from best to worst.

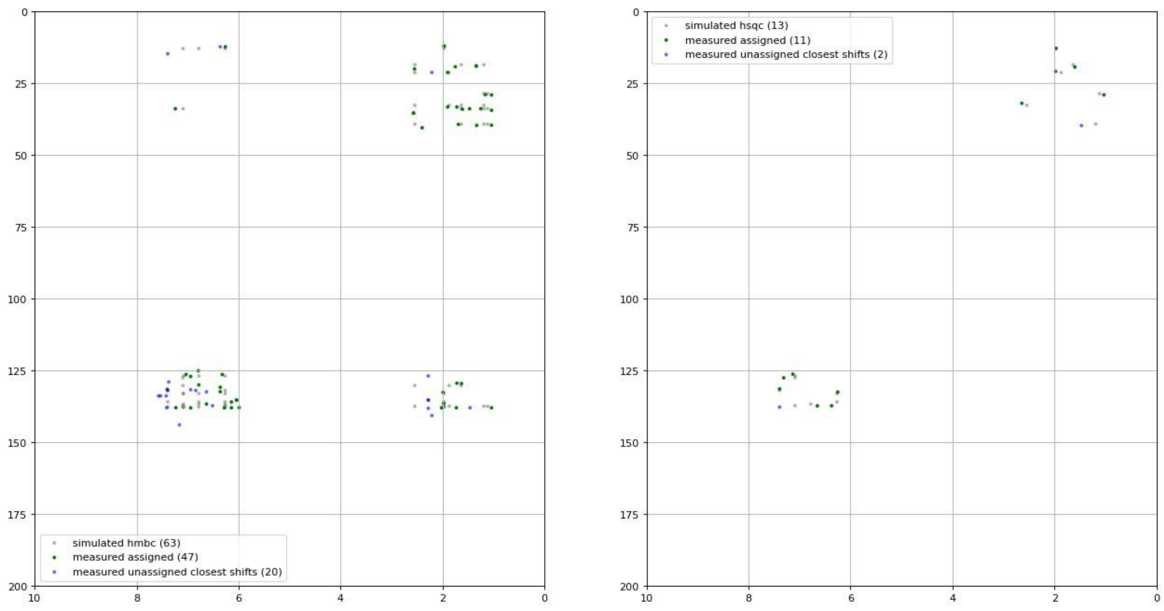


Figure 4. A plot generated visualizing the mapped shifts between the measured and simulated candidate spectra.

4.5 Additional Features

Nmrfilter has features that can be configured by the user by editing the `nmrproc.properties` file.

By default, the prediction method for simulating the candidates' spectra is based on HOSE codes. Nmrfilter allows the user to enable respredict, a way of predicting single atom NMR shifts using machine learning, found to be significantly better than a HOSE code based prediction [7]. Respredict can be enabled by setting the `usedeeplearning` property to `true`.

When a solvent was used in the NMR experiment from which the measured data originates, the user can set the solvent by changing the `solvent` property. Four options are available: `Methanol-D4 (CD3OD)`, `Chloroform-D1 (CDCl3)`, `Dimethylsulphoxide-D6 (DMSO-D6, C2D6SO)` and `Unreported`.

If HSQCTOCSY shifts are included in the measured spectra and the user desires to include them in the prediction, the property `usehsqctocsy` can be set to `true`. The `debug` property if set to `true` can be used to produce debug outputs. Jupyter Notebook[18] is available for interactive and collaborative use of Nmrfilter.

4.6 Shortcomings

Albeit Nmrfilter is a fully functional program that can predict a compound from its NMR data accurately, it has been built for demonstrating and testing an algorithm rather than for use as a standalone tool. Several shortcomings have been identified, which when improved or implemented, could enhance the user experience and open up the software for new users to fulfill their needs in analytical chemistry. Improvements and implementations that have been identified:

- Current Nmrfilter GitHub `README.md`[19]contains inaccuracies, is outdated, and could use some beautification.

[18] https://jupyter.org/

[19] https://github.com/stefhk3/nmrfilter/blob/master/README.md

- Environment setup is tricky, with the use of Anaconda still producing dependency conflicts and `ModuleNotFoundError` errors.
- Currently, the plots alone give no real information, as references are needed to draw conclusions from the spectra and the prediction. For this purpose, the plot could have the real measured NMR spectra as a background image to contextualize the peaks better. In addition, a skeletal chemical structure would further complement the plot for easier analysis.
- As the simulated spectra for the candidate compounds are based on data gathered over time in a database that is included in the repository (currently, it is updated manually), automatic updates for the database could improve the simulated spectra.
- Nmrfilter simulates spectra for all the candidates. This feature could be enabled as a standalone feature for batch simulation of spectra, without running the prediction. Useful in analytical chemistry.
- Testing has found that for some predictions, the Louvain-RBER clustering part produces large clusters that may render the prediction less accurate. For this reason, if the clusters after Louvain-RBER processing are too big, the algorithm could be run again with different parameters resulting in smaller clusters, possibly bettering the prediction.

The aforementioned issues will be explored, described, improved and implemented together with reasonings in the next chapter.

5 Implementation of New Features: Nmrfilter v1.5

This chapter will explore the issues identified with the original Nmrfilter[20] (hereafter referred to as the original) and will implement new features. These issues will be explored with reasonings along with demonstrations. New features and bug fixes will be implemented to a fork of the original repository, called Nmrfilter v1.5[21].

5.1 Updating README.md

Currently, when trying to run the program by following the `README.md` present in the original repository[22], reproducible issues arise along with naming errors that may confuse the user, leading to the inability to get the program running smoothly. Accurate documentation is important for the accessibility and ease of use for a piece of software.

5.1.1 Naming & Rights of Run Scripts

For running the program, the original `README.md` contains the following paragraph:

`Once these files are in place, run` **`nmrfilter.sh`** `<projectname> (linux) or` **`nmrfilter.bat`** `<projectname> (windows). This should produce the result list. Replace by the name of the project/folder you want to work on.`

A script by this name (highlighted in bold) does not exist in the repository, with the equivalent script in question being named `run-standalone` instead. For Nmrfilter v1.5, these scripts will be renamed to `nmrfilter` as intended. Additionally, the scripts will be given execute rights by default, sparing the user from having to change it themselves.

[20] https://github.com/stefhk3/nmrfilter

[21] https://github.com/kulakarla/nmrfilterv15

[22] https://github.com/stefhk3/nmrfilter/blob/master/README.md

5.1.2 Dependencies and Environment

Original `README.md` contains redundant information about running the program. The main run script `run-standalone` creates a Python virtual environment[23] installing all the necessary packages needed for running the program. Therefore, a bulk of the information regarding user-side installation of Python packages can be removed and replaced with just requirements of Java[24] 1.8 or newer, Python 3.11[25] or newer, and Python Virtualenv module. The rest of the required packages, e.g. louvain[26] and igraph[27], will be installed by pip in the virtual environment created by Nmrfilter v1.5.

When using the provided Anaconda[28] environment following the original `README.md` guidelines, an error can be reproduced due to discrepancies between the Anaconda environment and the Python virtual environment. Having created the `nmrfilter` conda environment from the `environment-cpu.yml` file and running the main script, the following error is produced:

```
Traceback (most recent call last):
  File "nmrfilter2.py", line 5, in <module>
    from clustering import *
  File "/home/karl/nmrfilter/clustering.py", line 2, in <module>
    from numpy import *
ModuleNotFoundError: No module named 'numpy'
```

In general, this issue has a simple mitigation – removing the creation of a Python virtual environment if all the necessary dependencies are already present in the external environment. There is no point in creating a Python virtual environment inside a conda environment that already has all the necessary packages installed. For this reason, Nmrfilter v1.5 will check if a conda environment is currently in use. If so, a Python virtual environment will

[23] https://docs.python.org/3/library/venv.html

[24] https://www.java.com/en/

[25] https://www.python.org/downloads/release/python-3110/

[26] https://pypi.org/project/python-louvain/

[27] https://pypi.org/project/igraph/

[28] https://www.anaconda.com/

not be used and vice versa. This check will be implemented using the `$CONDA_SHLVL` environment variable, which allows for checking if conda is activated or is installed.

Before continuing to the main processing part of Nmrflilter v1.5, the main script will check and set up the environments as needed.

The underlying cause for this error is the use of an outdated conda environment. `Environment-cpu.yml` defines a conda environment `nmrfilter` with Python version 3.7[29] along with several dependencies. When the original run script is executed, Python tries to create a virtual environment with dependencies and their versions defined in the file `requirements.txt`. In this process, conflict arises due to the numpy version in the `requirements.txt` file being defined as 1.24, which requires Python version 3.8[30] or newer, according to the official documentation of numpy release 1.24.0[31]. Due to the conda environment `nmrfilter` using Python 3.7, pip fails to install numpy version 1.24, ultimately resulting in a `ModuleNotFoundError` error.

Conda environment files (CPU, GPU) and Python virtual environment files will be updated in Nmrfilter v1.5 to install the latest packages compatible with Python 3.11.2[32] to eliminate possible conflicts between external and virtual environments, combined with the elimination of redundant environment nesting.

Nmrfilter v1.5 will be tested and developed using Python 3.11, updating the required Python version to be at least 3.11. Python drops active support for releases after 5 years [9], with version 3.7 being out of support since 10.2023 and support for Python 3.8 ending in 10.2024, thus making updating Python to 3.11 plausible.

[29] https://www.python.org/downloads/release/python-370/

[30] https://www.python.org/downloads/release/python-380/

[31] https://pypi.org/project/numpy/1.24.0/

[32] https://www.python.org/downloads/release/python-3112/

5.2 Batch Simulation

While NMR technology is versatile, the spectra can be hard to interpret without any reference information. Simulated spectra may be an important tool for extracting information about molecular structure and optimizing experimental protocols [10] for NMR experiments. Simulated spectra can be used for structure elucidation, compound identification, quantitative analysis, and more.

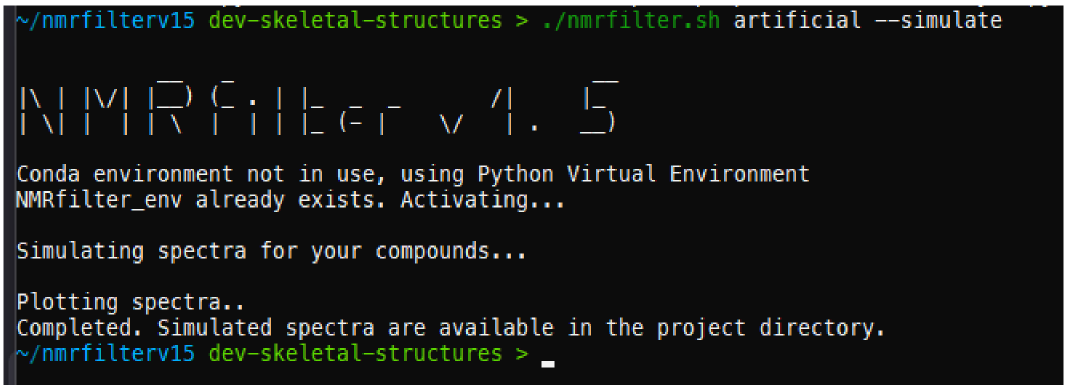
The original program already batch simulates spectra for its main purpose – structure and compound identification. Since simulated spectra have several use cases, and the function already is present in the program, it would be a useful tool to have as a standalone feature of the program.

`Simulate.jar` is a small Java program inside Nmrfilter v1.5 that is responsible for all the simulations of the spectra. It takes in the user-given candidate substances and runs a stereo-aware HOSE code prediction for the candidate, using reference data from nmrshiftdb2[33] included in the .JAR archive as an offline database.

Batch simulation as a standalone feature will be implemented by checking for a command-line flag `--simulate`. If given, the program will skip the similarity and identification measure. While it could've been implemented as a configurable property in the `nmrproc.properties` similar to other options, for reasons of simplicity and ease-of-use this way of implementation was not preferred. The output of the batch simulation are the relevant experiment spectra plots specified by the user in `nmrproc.properties` with a .CSV file containing the simulated chemical shifts. It is not mandatory but strongly advised to include the `namesfile` corresponding to the SMILES file. The names can be placeholders as they are used for generating, saving, and naming the resulting plots and other necessary files. An example of the output in the terminal when `--simulate` flag is used (Fig. 5), together with

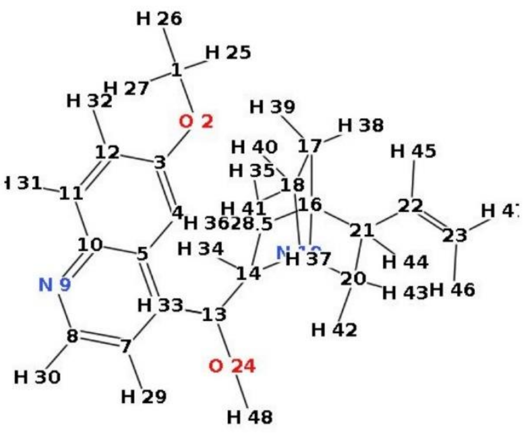
[33] https://nmrshiftdb.nmr.uni-koeln.de/

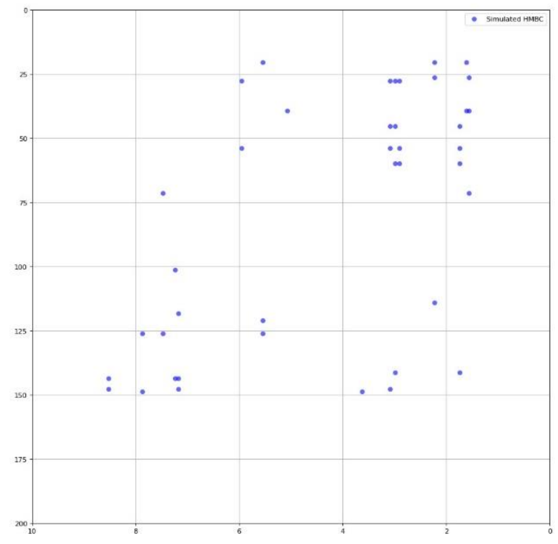
the generated HMBC and HSQC spectra for one candidate (Fig. 6), and a zoomed-in view of a resulting plot (Fig. 7).



```
~/nmrfilterv15 dev-skeletal-structures > ./nmrfilter.sh artificial --simulate

           __   _                               __
|\ | |\/| |__) (_  . | |_  _  _      /|    |_
| \| |  | | \  |   | | |_ (- |    \/  | .  __)

Conda environment not in use, using Python Virtual Environment
NMRfilter_env already exists. Activating...

Simulating spectra for your compounds...

Plotting spectra..
Completed. Simulated spectra are available in the project directory.
~/nmrfilterv15 dev-skeletal-structures > _
```

Figure 5. Terminal output if `--simulate` flag used.





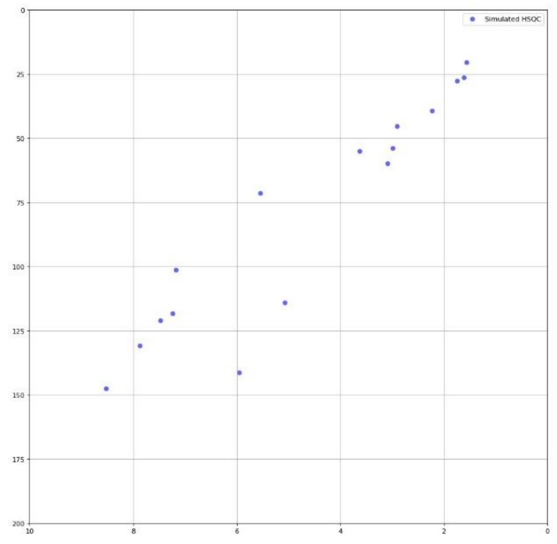


Figure 6. Simulated HMBC, HSQC spectra together with the substance’s structure.

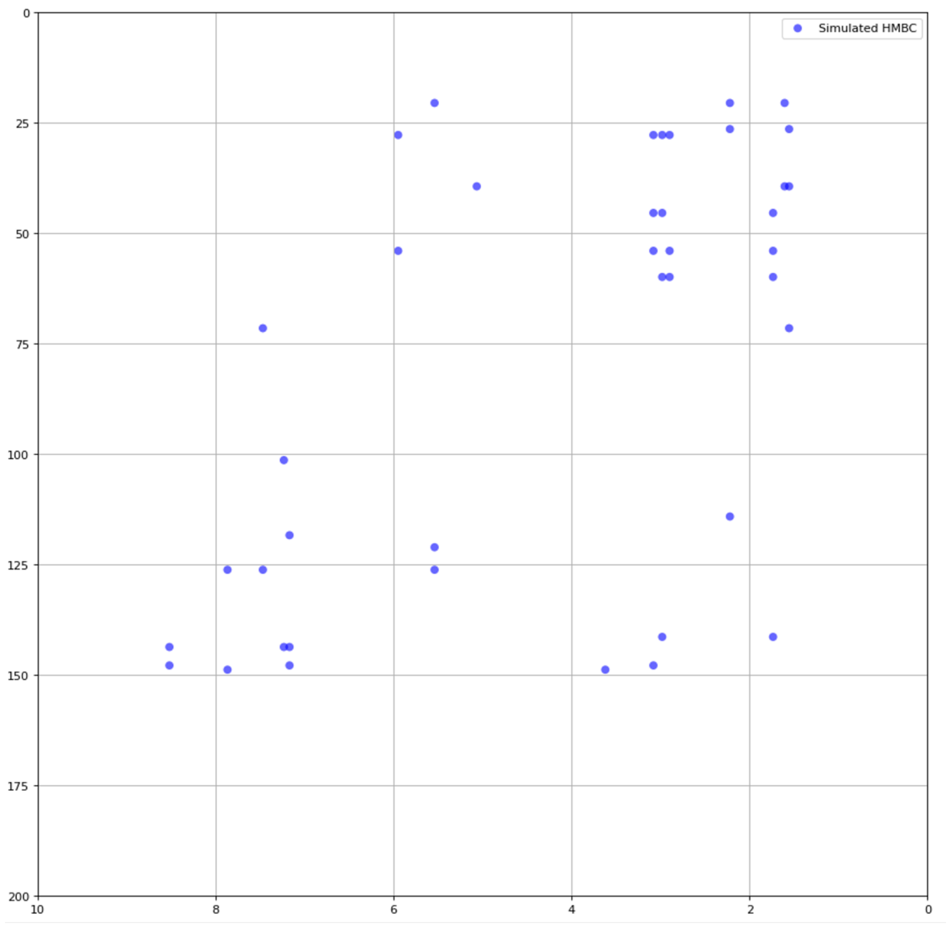


Figure 7. Close-up of a generated HMBC spectra.

5.3 Background Images for Generated Plots

Spectroscopy can be considered a visual technique, as the data from the experiments is best presented as a graph or some sort of spectrum. In NMR the analysis of the experiment ultimately comes to the analysis of several spectra determining different properties of the molecule. Human analysis is still largely part of the NMR analysis process due to contextual factors that automated systems tend to overlook. While the generated plots currently give an adequate comparison between the simulated and user-given peak list, another great reference to further enhance the analysis and verification of Nmrfilter v1.5 identification results would be the inclusion of the original spectrum in the plots.

Bruker[34] is considered the industry leader in the manufacturing of NMR instruments, according to some sources Bruker holds a nearly 90% market share in the manufacturing of NMR-related equipment [11]. With Bruker essentially being a monopoly, it would make sense to focus on integrating Bruker instrument data to Nmrfilter v1.5 for the background images. Multitude of tools exist for creating spectra from Bruker NMR data.

NMRium[35] is a great, free online tool that takes in Bruker data[36] to produce interactive graphs. Implementation of web-scraping NMRium generated spectra was considered, which through testing was deemed unfit due to limited configurability, particularly the axes being unconfigurable, confirmed by NMRium support. Bruker-utils[37], a Python package designed for handling Bruker NMR data was evaluated and ultimately found unsuitable due to stringent requirements for input data formats. This is a large downside considering Bruker instrument output folders vary widely in structure. NMRglue[38] is a Python package designed for working with NMR data. NMRglue has various tools available for visualizing the NMR

[34] https://www.bruker.com/en.html

[35] https://www.nmrium.org/

[36]http://www.nmragenda-leiden.nl/static/Bruker_manual_4.1.4/topspin_pdf/Bruker_NMR_Data_Formats.pdf

[37] https://pypi.org/project/bruker-utils/

[38] https://pypi.org/project/nmrglue/

data but needs expertise for proper use. Nmrplot[39] is a wrapper around NMRglue that allows for easy plotting of NMR data with lax requirements for input data together with other necessary tools. Visually appealing NMR spectra for various experiments can be achieved in just a few lines of code, due to which this package was ultimately chosen for the implementation. Due to the limitations of Nmrplot, particularly its inability to draw on existing axes, the spectra images will be generated, cropped, and used as a background image for the main plots rather than being plotted on the same axes alongside the other cluster information. The new user configurable properties will be `hmbcbruker`, `hsqcbruker` and `hsqctocsybruker`, allowing the user to define the name of the relevant experiment folder together with the correct subfolder.

If the user wants the measured off-the-instrument spectrums to be included in the plot, they need to include the folder of the relevant experiments in their project folder. Additionally, they need to specify the folder names in the properties file. The correctness of included Bruker folders will be the responsibility of the user due to their many variations. If the user has included everything properly, the generated plots (Fig. 8) will appear as intended.

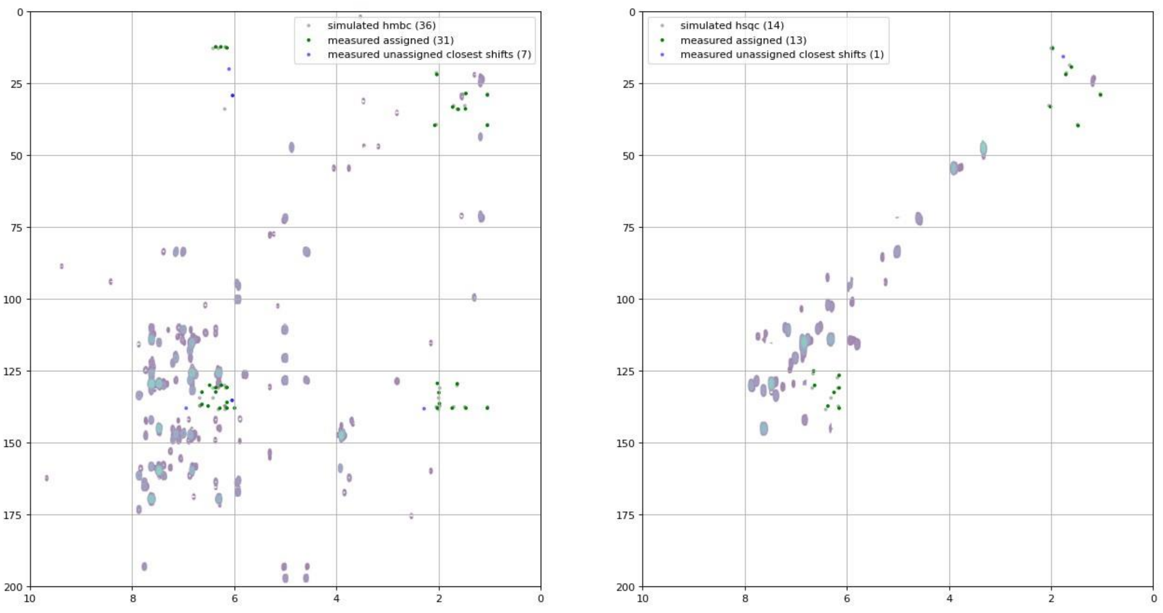


Figure 8. Generated plot for beta-carotene with Bruker backgrounds enabled.

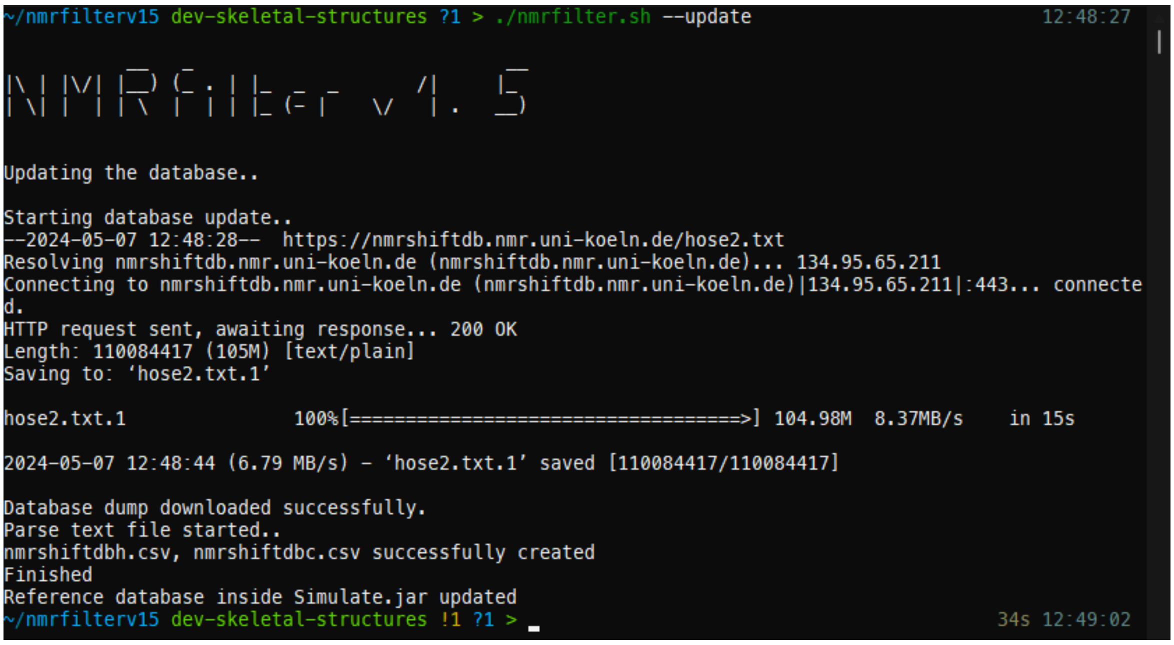
[39] https://miguelarbesu.xyz/post/nmrplot/

Plots are an asset that now allow the user for a more thorough and easier analysis of the identification results, amplifying the Nmrfilter v1.5 user experience. Background spectra are available for HMBC, HSQC, and HSQCTOCSY experiments.

5.4 Database Update for Spectra Simulations

Prediction of the spectra is done by the `simulate.jar` program, which uses an offline database included in the .JAR archive. The prediction is essentially done by looking up the right shift value according to the HOSE code and solvent. This offline database contains the data from nmrshiftdb2. Being an open and ever-growing database, it could be considered useful to update the offline database included in the .JAR for the user from time to time to include the latest data. Over its 20+ year lifespan, nmrshiftdb2 has accumulated an average of 3500 measured spectra annually. Due to the simulation part being a reference-based simulation (looking up the right values according to the inputs), the latest data being made available could enhance the predictions.

Database updating will be implemented as a user-side script that can be executed by running the main run script with the flag `--update`. Following, the update script is executed, which proceeds to download a database dump updated monthly from nmrshiftdb2. The downloaded dump will be processed by a small Java program called DumpParser2, created for this very purpose. DumpParser2 goes through the data and obtains HOSE codes and their respective shift values for accepted solvents in Nmrfilter v1.5. DumpParser2 will finish having generated two files: `nmrshiftdbc.csv` and `nmrshiftdbh.csv`, containing HOSE codes and their chemical shifts for carbon and hydrogen atoms respectively. Continuing, the update script will transfer the created database files inside `simulate.jar` and deletes the downloaded dump and generated files from the main directory. Figure 9 demonstrates this feature from the user's point of view. Database updating is not done automatically due to the intricacies of the database itself, therefore `README.md` informs the user about their responsibility in having the latest data available. It is recommended to run the database update every month.



```
~/nmrfilterv15 dev-skeletal-structures ?1 > ./nmrfilter.sh --update                        12:48:27

          __  _                            __
|\ | |V| |__) (_ . | |_ _ _       /|     |_
| \| | | | \  |  | | |_ (- |   V   | .   __)

Updating the database..

Starting database update..
--2024-05-07 12:48:28--  https://nmrshiftdb.nmr.uni-koeln.de/hose2.txt
Resolving nmrshiftdb.nmr.uni-koeln.de (nmrshiftdb.nmr.uni-koeln.de)... 134.95.65.211
Connecting to nmrshiftdb.nmr.uni-koeln.de (nmrshiftdb.nmr.uni-koeln.de)|134.95.65.211|:443... connecte
d.
HTTP request sent, awaiting response... 200 OK
Length: 110084417 (105M) [text/plain]
Saving to: ‘hose2.txt.1’

hose2.txt.1              100%[==================================>] 104.98M  8.37MB/s    in 15s

2024-05-07 12:48:44 (6.79 MB/s) - ‘hose2.txt.1’ saved [110084417/110084417]

Database dump downloaded successfully.
Parse text file started..
nmrshiftdbh.csv, nmrshiftdbc.csv successfully created
Finished
Reference database inside Simulate.jar updated
~/nmrfilterv15 dev-skeletal-structures !1 ?1 > _                                       34s 12:49:02
```

Figure 9. Database updating in Nmrfilter v1.5

5.5 Skeletal Structure with Plots

Analysis of NMR results relies heavily on the reading of spectra from which the structure can slowly be assembled from. HSQC and HMBC give information about proton-carbon single-bond correlations and proton-carbon multi-bond correlations respectively. While Nmrfilter v1.5’s main function is to automatically read and analyse these spectra, these results are not guaranteed to be 100% accurate. For further processing and verifying the results, human interference is still needed. When analysing the results from the plots, the user would greatly benefit from having a reference structure that points out which cluster/shift is tied to which atom. This would quicken the process of verifying the results substantially.

Initially, the idea was to implement this interactively – after the identification processing, a small Dash/Plotly[40] app would pop up, where the user could select the candidate for which they desire to analyse the results. Following, the user is presented with the skeletal structure of the candidate and relevant plots. When hovering over an atom on the skeletal structure, it would highlight the cluster/shift on the plot which is tied to that atom, and vice-versa.

[40] https://dash.plotly.com/

While this interactive implementation could have been more aesthetically pleasing and better for some users, some obstacles arose.

Various chemical packages exist for different programming languages. The simulation part in Nmrfilter v1.5 uses Chemistry Development Kit[41], a Java library for working with cheminformatics. In Python, RDKit[42] is another popular package for working with chemical data. In the implementation of interactively tied plots and skeletal structures, the identification of atoms and their corresponding shifts would be necessary. The challenge arises in this identification process – different chemical libraries structure their chemical objects differently, resulting in different atom identifiers. ChemDoodle Web Components[43] was another library tested for drawing interactive chemical structures. While for small molecules, the identifiers of atoms could turn out to be the same, there is no guarantee. Another issue with this interactive approach is its plausibility – the identification is usually done with tens if not hundreds of candidates, which makes a browser pop-up application with 100 interactive plots rather unfeasible, unnavigable along with predictable poor performance. This type of user interactive implementation does not suit the current CLI-based program, instead it would be better suited for a web application version of Nmrfilter v1.5.

Due to Nmrfilter v1.5 already generating skeletal structures of the candidates in debug mode, these images could be added to the plots, with numerical references between atoms and clusters. Matplotlib[44] offers a way to label points, which will be used to implement this feature. While Matplotlib provides an easy one-line function to label the points with, it does lack features in some areas, particularly in the prevention of label overlapping. AdjustText[45] is a Matplotlib helper package for dealing with overlapping labels. Testing this library however revealed some of its shortcomings, particularly difficult-to-comprehend configurability together with lackluster performance. Even with a moderate number of labels and points,

[41] https://cdk.github.io/

[42] https://www.rdkit.org/

[43] https://web.chemdoodle.com/

[44] https://matplotlib.org/

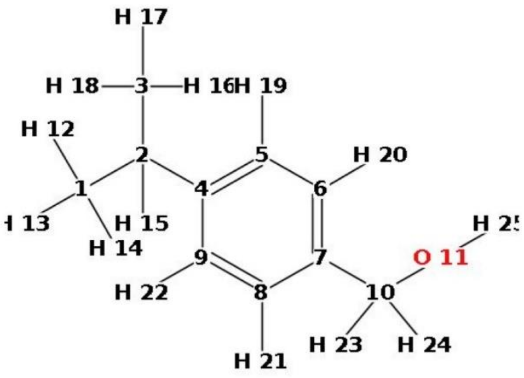
[45] https://github.com/Phlya/adjustText/ DOI: https://zenodo.org/badge/latestdoi/49349828

e.g. 20, the labels either overlapped or the arrow-label placement was questionable, with some labels stretching to the other side of the plot for seemingly no reason. StackOverflow[46] user ckjellson[47] among various threads [12] pointed out their created Python package for managing label placement on Matplotlib plots – textalloc[48]. Textalloc performed better in comparison to adjustText, both in speed and visibility, solidifying its use in Nmrfilter v1.5 plot label distribution. With smaller molecules (number of atoms less than 30), the plots are readable and visually pleasing, providing insight into atom-shift relationships in different spectra. With large molecules and tight clustering, visibility degrades. To combat this, a property `labelsimulated` was added to give the user options to disable or enable the labelling of points according to their preference. If set `true`, the points will have a label corresponding to the carbon and hydrogen identifier (visible on the skeletal structure) related to the pointed shift. The optionality of this property is provided to follow good practices in scientific software development – particularly reducing the reliance on graphical components [13] to enable quicker calculation on large datasets, due to label positioning taking a considerable amount of time. Additionally, file-based configuration is considered a good practice in scientific software [13]. With labelled plots enabled, the resulting plots (Fig. 10) will have a skeletal structure sitting on top of various plotted spectra, which when magnified (Fig. 11) display shift-atom relationships. Labels disabled, the plots will retain a simpler look (Fig. 8). User has the option to disable the plotting altogether by excluding the `namesfile` from the project folder. It should be noted that the generated plots are high-definition .PNG images, supporting zooming and scrolling, which on paper may not give the impression.

[46] https://stackoverflow.com/

[47] https://stackoverflow.com/users/16879899/ckjellson

[48] https://pypi.org/project/textalloc/



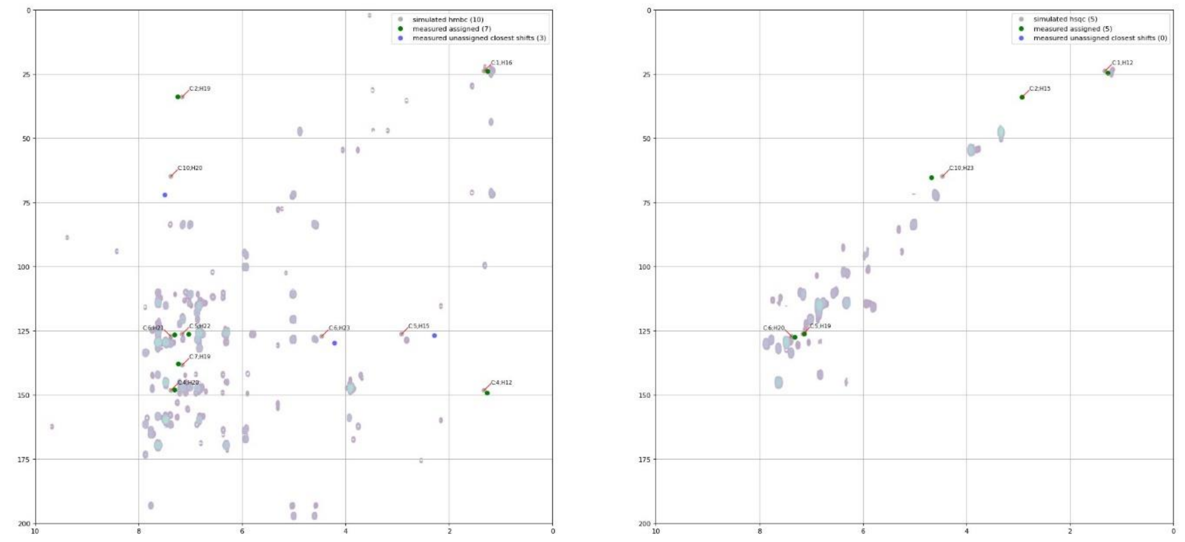


Figure 10. Generated plot with labelled simulated clusters for 4-isopropylbenzyl alcohol

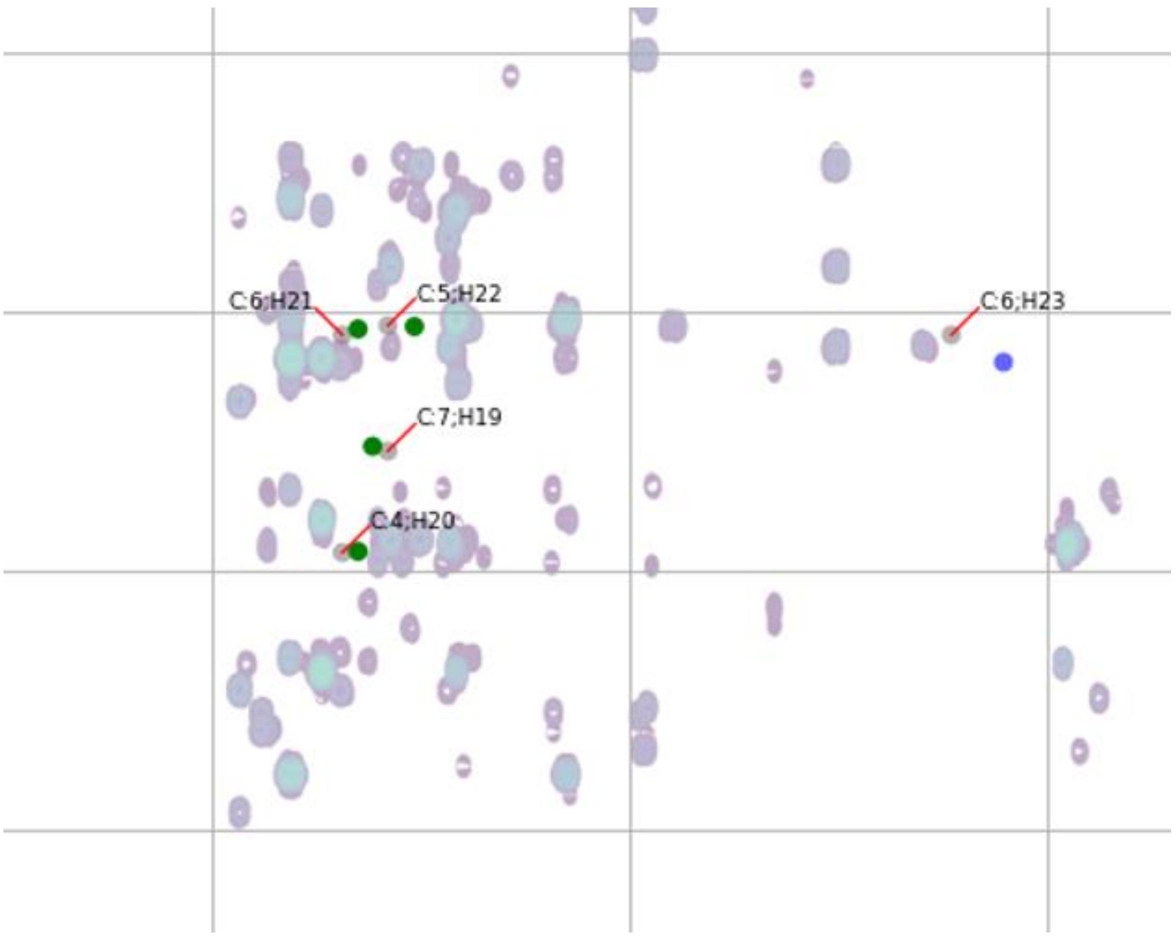


Figure 11. Magnified labelled plot for 4-isopropylbenzyl alcohol

6 Result

This chapter will present a short overview of the implemented features for the enhanced version of the original program called Nmrfilter v1.5[49] created in the scope of this thesis, together with a little comparison with the shortcomings listed in Chapter 4.6. Furthermore, some failures will be reflected on.

6.1 Implemented Features

In Chapter 4.6, six potential weaknesses were identified as needing attention and possible adjustments. Out of these six, five have been addressed.

The original applications' GitHub `README.md`[50] included inconsistent naming, lacked uniformity, and overall was deprived of relevant information about the program. This has been addressed, with the glossary explaining properties now included in the `README.md`[51], behaviour of the program explained a bit more thoroughly and the removal of erroneous information.

Previously, running the program initially (tested on multiple devices) possibly resulted in errors that were identified to stem from version clashes of some required packages. This has been solved by updating the Python[52] version along with most of the packages used in the program. Additionally, the Anaconda[53] environments previously installed unnecessary packages, as the program created a Python virtual environment when running anyway, redundantly installing some packages again. The main run script of the program is now configured to avoid using the Python virtual environment should the user desire to use an

[49] https://github.com/kulakarla/nmrfilterv15

[50] https://github.com/stefhk3/nmrfilter/blob/master/README.md

[51] https://github.com/kulakarla/nmrfilterv15/blob/master/README.md

[52] https://www.python.org/

[53] https://www.anaconda.com/

external Anaconda environment. These changes came at a cost, as the set-up of environments takes a bit longer than originally, however, set-up process is overall smoother.

In the original program, the plots generated were a rather small helpful tool for a chemist to analyse their prediction results. The plots lacked reference information, which is a fundamental part of NMR analysis due to its reliance on reference information, e. g. known shift values in certain experiment conditions. The plots were given a skeletal structure of the candidate, along with an option to enable labels for simulated spectra shifts informing the user about which atom some shift belongs to, removing the need to manually start correlating the atom-shift pairs before getting to evaluate the results. Together with shift-atom information, Nmrfilter v1.5 enables the user to incorporate their original measured NMR spectra to be used as background images for the plots. Data must be in the form of a Bruker folder[54], an integration chosen for its widespread usage and near monopoly in the market of NMR instruments. Experiment spectra further complement the plots, enabling easy filtering of completely irrelevant candidates for example.

Nmrfilter v1.5 uses a reference database included inside the `simulate.jar` program, responsible for the simulation of candidate spectra. As mentioned before, NMR is a field that relies heavily on references as the analysis of results depends on known shift values in some conditions x and y. The data included in the offline database comes from nmrshiftdb2, an open ever-growing database constantly expanding with new measured NMR data. Due to the limitations of the nmrshiftdb2 database itself, Nmrfilter v1.5 will still be using an offline database in the form of a .CSV file. However, the users are now provided a way to automatically update the database files inside the .JAR archive, potentially enhancing the prediction as there is more and more reference data available.

Nmrfilter v1.5 has enabled users to use the program for batch simulation of NMR spectra, provided a list of SMILES and a `namesfile` (can have placeholder names) is present.

[54]http://www.nmragenda-leiden.nl/static/Bruker_manual_4.1.4/topspin_pdf/Bruker_NMR_Data_Formats.pdf

Previously something that could have been enabled easily has been implemented as an extra feature should the user need a batch simulation of spectra.

Along with the mitigations for issues identified in Chapter 4.6, a handful of small extra tweaks were made. For example, the external .JAR libraries used by the `simulate.jar` program have been refactored to a folder called “libs” to clear up the program directory.

6.2 Failures

While the majority of the features originally proposed were implemented, not everything made it to Nmrfilter v1.5. Chapter 4.6 mentions the potential regression of prediction quality due to large clusters produced by the Louvain-RBER processing algorithm. This issue was explored briefly by playing around with the `rberresolution` property to see if any real difference can be noticed with different values for the Louvain-RBER algorithm. Further exploration of this issue was abandoned due to the indifference seen in tweaking the algorithmic parameter by small amounts, yet seemingly after some magic value, the prediction results were completely different. Seen as the author lacks comprehensive knowledge in NMR data science themselves, it was decided not to pursue this further to avoid the waste of resources.

The skeletal structure with referenced shifts on a plot was implemented statically. Initially, the idea proposed by the supervisor of this thesis was to implement this interactively, with the possibility of seeing which measured cluster was assigned to which simulated cluster by hovering over the respective atom for example. While this implementation would have been superior to the static implementation in Nmrfilter v1.5, the author decided not to pursue it due to the difficulty and possible scope of the development after some testing. This feature would require turning Nmrfilter v1.5 essentially into a web application, or at least a GUI-based one, something which can be considered something for the future. A detailed explanation for the decision to abandon the interactive implementation can be found in Chapter 5.5.

6.3 Overall Outcome

Although every initially proposed improvement upon the original program was not implemented as planned, Nmrfilter v1.5 can still be considered an enhanced version of the original program.

Nmrfilter v1.5 provides the user with additional tools to help with verifying the results of the automatic identification process. The new `README.md` allows for easier setup and understanding of the program. Previously problematic installation process has been fixed.

Generated plots contain useful information for accelerating analysis of the identification results, like the skeletal structure of the candidate together with atom-shift relationship data. Enabling raw Bruker spectral data allows for more reference information elevating the analysis process even more. Batch simulation of NMR spectra is a useful tool in all kinds of NMR work.

Albeit some of the features could have been implemented better or explored further, Nmrfilter v1.5 could be considered a moderate success. The improvements have broadened its applicability, enabling its use for a more diverse range of tasks in chemical analysis, making it a valuable available tool for researchers.

7 Future Work

The following chapter will include a small discussion about possible further development and the future of Nmrfilter v1.5[55].

7.1 Integration with NMRbox

NMRbox[56] is a digital resource bundling together NMR software. Among the software included in NMRbox[57] for example are MestReNova[58] and nmrglue[59], NMR software tools examined previously in this thesis. Discussions about the potential inclusion of Nmrfilter v1.5 in NMRbox have already started, with positive messages received. As of the 8th of May 2024, Nmrfilter v1.5 has yet to be included in NMRbox, however, as the conversation continues, its inclusion in the near future remains a possibility.

7.2 Web Application

Scientific software tends to be designed by researchers, who lack formal training in software engineering [15], resulting in the software often being poorly documented, non-intuitive and hard to install to name a few [16]. While Nmrfilter v1.5 is not the worst example of these criteria, it can be recognized as a specialized software designed for researchers who have some familiarity with software development. The requirements for using this software include, but are not limited to: input data has to be in a specific format, the ability to use CLI tools and knowledge on dealing with potential dependency clashes. Every chemist should not be considered a computer scientist.

The issue with Nmrfilter v1.5 being a considerably difficult-to-use piece of software could be alleviated with turning Nmrfilter v1.5 into a web application, or at least the creation of a

[55] https://github.com/kulakarla/nmrfilterv15

[56] https://nmrbox.nmrhub.org/

[57] https://nmrbox.nmrhub.org/software

[58] https://mestrelab.com/

[59] https://www.nmrglue.com/

web application counterpart. Running environment issues could be eased with containerization, eliminating the need to install anything. Additionally, a GUI-based solution is far more intuitive than a CLI-based one. Nmrfilter v1.5 is not computationally heavy software, therefore turning it into a web application should not be an obstacle. Moreover, as discussed in Chapters 6.2 and 5.5, interactive graphs could be a useful tool in NMR compound identification analysis, something which implemented in the current state of the program would not be feasible.

Converting Nmrfilter v1.5 into a web application would address several prevalent issues associated with scientific software. Eliminating these challenges and simplifying the use of Nmrfilter v1.5 would make it more accessible to a broader range of chemists who lack training in software engineering.

7.3 Code Refactoring

Nmrfilter v1.5 is not a well-written program by modern software design standards. Any experienced software engineer would identify various issues with the program's code, such as its lack of structure, difficulty to follow, absence of logging and inefficiency, to name a few. This leads to severely stunting the future development of the program caused purely by the amount of time it takes to understand how the program works. Refactoring and logically restructuring Nmrfilter v1.5 would facilitate collaborative development, something which in its current state is challenging to accomplish.

8 Conclusion

In the field of chemistry, precision is crucial, as properties of a substance determine its use. Properties are determined by the composition and structure of a molecule, implying great importance in identifying a substance before it can be utilized as a medicine, cleaning chemical, or food additive amongst others. Nuclear magnetic spectroscopy is a widely used experimental technique in chemical analysis to determine the structure of an unknown compound.

Various software solutions exist for processing NMR data. Nmrfilter[60] is an open-source software developed in the collaborative effort by of S. Kuhn, S. Colreavy-Donnelly, J. S. de Souza, and R. M. Borges [2] to demonstrate their improved software pipeline for automated identification of a substance from its NMR data. Despite Nmrfilter having accomplished its initial objective as a tool showcasing an algorithm, several improvements could turn it into a standalone piece of scientific software useful in chemical analysis.

Nmrfilter v1.5[61] is an enhanced version of the original Nmrfilter, implementing various improvements as a part of the bachelor's thesis work. Nmrfilter v1.5 added the possibility to batch simulate NMR spectra for user-provided chemicals and incorporated the possibility of automatically updating the database from which the simulation process lies on. Additionally, generated plots now include the skeletal structure of the candidate substance with an optional feature to label shifts with their respective atom identifiers. The user can now use raw NMR experiment spectral data as a background for the plots further facilitating the analysis of results. Furthermore, the setup of the original program was riddled with possible errors, which have been fixed by updating the Python[62] version used along with other package updates. Nmrfilter v1.5 also incorporates a beautified `README.md`[63] available in the repository enabling the users to install and use the software with ease.

[60] https://github.com/stefhk3/nmrfilter

[61] https://github.com/kulakarla/nmrfilterv15

[62] https://www.python.org/

[63] https://github.com/kulakarla/nmrfilterv15/blob/master/README.md

While Nmrfilter v1.5 includes various improvements compared to the original program, it has room for improvement. In the future, Nmfilter v1.5 could be included in the NMRbox[64] software suite, along with the development of a web application version allowing interactive visual analysis of the results. Introducing a GUI-version of Nmrfilter v1.5 would make the program accessible to chemists who are not well versed in software development, combined with an overall improved user experience. Whereas numerous possibilities for further development exist, a cleanup of Nmrfilter v1.5 code should be prioritized to enable efficient future development.

[64] https://nmrbox.nmrhub.org/

9 References

[1] H. Günther, NMR Spectroscopy: Basic Principles, Concepts and Applications in Chemistry, 3rd Edition, John Wiley & Sons, Incorporated, 2013, p. 2.

[2] S. Kuhn, C.-D. Simon, J. Santana de Souza and R. M. Borges, "An integrated approach for mixture analysis using MS and NMR techniques," *Faraday Discussions,* vol. 218, pp. 339-353, 2019.

[3] *Menthol Proton Spectrum.* [Online Image]. 2005. Available: https://en.m.wikipedia.org/wiki/File:Menthol_Proton_Spectrum.jpg [Accessed 13 May 2024].

[4] Idelso, Artist, *Espectro COSY de 1H-RMN.* [Online Image]. 2007. Available: https://commons.wikimedia.org/wiki/File:Espectro_RMN_COSY.png [Accessed 13 May 2024].

[5] Atta-ur-Rahman and M. I. Choudhary, Applications of NMR Spectroscopy, Bentham Science Publishers, 2015.

[6] M. Baqueta, A. Coquiero, L. de Sousa Frutuozo, P. H. Marco, F. Duarte, M. Mandrone, F. Poli and P. Valderrama, "Correlation between VIP Scores and 1H NMR to Extract Information of Psychological Attention Tests Applied Before and After Coffee Intake," in *Applications of NMR Spectroscopy,* vol. 8, Bentham Science Publishers, 2020, pp. 25-41.

[7] R. Gotadki, "Nuclear Magnetic Resonance Spectroscopy Market Size 2032," *Market Research Future,* 2024. URL: https://www.marketresearchfuture.com/reports/nuclear-magnetic-resonance-spectroscopy-market-12153 [Accessed 13 May 2024].

[8] S. Kuhn and E. Jonas, "Rapid prediction of NMR spectral properties with quantified uncertainty," *Journal of Cheminformatics,* 2019.

[9] S. Kuhn and S. R. Johnson, "Stereo-Aware Extension of HOSE Codes," *ACS Omega,* vol. 4, no. 4, pp. 7323-7329, 2019.

[10] "Python Developer's Guide," 2024. [Online]. Available: https://devguide.python.org/versions/. [Accessed 7 May 2024].

[11] K. Seetharam, D. Biswas, C. Noel, A. Risinger, D. Zhu, O. Kats, S. Chattopadhyay, M. Cetina, C. Monroe, E. Demler and D. Sels, "Digital quantum simulation of NMR experiments," *Science Advances*, vol. 9, 2023.

[12] M. S. Reisch, "Bruker Price Jitters," *Chemical & Engineering News*, vol. 93, no. 26, 2015.

[13] ckjellson, "Just created another quick solution that is also very fast: textalloc," 4 November 2022. [Online]. Available: https://stackoverflow.com/a/74317297/13153706. [Accessed 7 May 2024].

[14] F. Queiroz, S. Brockhauser, R. Silva, J. M. Miller and H. Fanohr, "Track 1 Paper: Good Usability Practices in Scientific Software Development," *Workshop on Sustainable Software for Science: Practice and Experiences*, Oct 2017.

[15] S. Mangul *et al.*, "Challenges and recommendations to improve the installability and archival stability of omics computational tools," *PLOS Biology*, vol. 17, no. 6, 20 June 2019.

[16] M. List, P. Ebert and F. Albrect, "Ten Simple Rules for Developing Usable Software in Computational Biology," *PLOS Computational Biology*, vol. 13, no. 1, 5 January 2017.

Appendix

I. Glossary

(Chemical) shift – NMR axes units. A shift is measured in parts per million (ppm). ^{13}C atom having a chemical shift of 60 means the carbon atom that caused that peak needed a magnetic field 60 millionths less than the field needed by TMS (Tetramethylsilane) to resonate.

Cluster – group of closely related signals on an NMR spectrum.

Cross peak – two off-diagonal points with coordinates reversed on a 2D spectrum, indicating coupling (some relationship) between some atoms. Simplified, if there is a point at (x, y) and (y, x), where $x \neq y$, you could say you have found cross-peaks, indicating a chemical relation. This is a highly simplified explanation!

Crystalline structure – ordered arrangement (bonds, positions etc.) of atoms, ions, molecules in a chemical compound.

Fourier transformation – a mathematical technique used to transform a function f to a function f' describing the extent of frequencies present in the original function f. In simpler terms, a way to see how many times some frequency x and y occurred. In the context of NMR, it is used to transform measured electromagnetic frequencies to a readable spectrum used for NMR analysis.

Functional groups – group of atoms in a molecule with distinct characteristics, regardless of other atoms in the molecule like alcohols, esters, and ethers.

HOSE code – abbreviation for "hierarchically ordered spherical environment" code, an encoded way to describe the environment of an atom, e.g. which atoms are 1, 2 or 3 bonds away, their charges etc.

Invasive technique – an experimental method which damages or changes the sample. Opposite of non-invasive, which does not damage subject being experimented with.

Isomer – chemical compounds with the same formula (same number of atoms of each element), yet a different underlying structure.

Mass-to-charge ratio – mass of an ion divided by electrical charge of the ion, used for identifying a substance using the identical behaviour of ions with same mass-to-charge ratios.

Metabolism – life-sustaining chemical reactions in living organisms.

Metabolite – end product of some metabolic process, e.g. glucose turned into energy.

RF receiver – a device used to transmit and/or receive radio signals, enabling wireless radio communication.

SMILES – “simplified molecular-input line-entry system”, a way to describe the structure of a chemical in ASCII string form.

Spectrum/spectra - graphical representation of the intensity of electromagnetic waves.

Spectroscopy (resonance) – field of studying electromagnetic spectra.

TMS – Tetramethylsilane, a compound accepted as industry standard for calibrating the chemical shift for ^{1}H and ^{13}C in organic solvents.

II. License