**Computer Science 380 Research Project**

A new tool for the identification of binding sites

Hasnain Cheena

Dr. Joerg Wicker

Dr. Katerina Taskova

[02/11/2020]

**ABSTRACT**

Mass spectrometry is a key tool involved in the identification and characterization processes within the drug development industry. The results of a mass spectrometry experiment are manually analyzed by chemists. This causes the lead-time for results to fluctuate widely depending on chemist experience and the complexity of the reaction performed. In this study, we propose a new analysis tool, called *BindingSearch*¸ for the automatic identification of binding sites. BindingSearch can be used as a validation tool after analysis is complete to verify results or scoping tool prior to manual analysis to narrow the search range for binding sites. Thus, in both use-cases BindingSearch effectively reduces analysis time. The proposed method has been compared to an alternative method called *MatchingSearch.* Results show that the BindingSearch method outperforms MatchingSearch. BindingSearch boasts increased accuracy in binding site classification, reduction in false positives and a decrease in the time taken for the overall analysis.

# Introduction

The usage of mass spectrometry within the drug development industry is extensive [2]. However, a

large amount of the analysis work is still completed manually [1, 5]. The time taken to complete analysis

tasks such as finding where different substances have bound, range from days to months of work

depending on the complexity of the reaction and expertise of the chemist. In this study, we propose a

new analysis tool, called *BindingSearch*, which can be used in the identification of binding sites from

mass spectrometry data. The aim of the tool is to be able to identify binding sites in an automated

fashion reducing the lead time of the analysis to hours. In this study, the method was tested using data

from multiple top-down electrospray ionization mass spectrometry experiments outlined in [7, 8]. The

proposed method has been compared to the MatchingSearch tool proposed in [3]. Results show that the

proposed method outperforms MatchingSearch because of its increased accuracy, reduction in false

positives and the decreased time taken for the overall analysis.

# Related Work

The small amount of past work relating to automated binding site analysis comes from [3]. In [3], a search method called MatchingSearch is used. This search uses a brute-force method to find binding locations. The method creates a difference spectrum by subtracting the unbound and bound spectrums and then matches the peaks of the difference spectrum to a theoretical spectrum of the primary reactant. This method has a low accuracy for finding binding sites and produces many false positives. The large number of false positives arise because the unbound and bound spectrums are not aligned before subtraction. Furthermore, the low accuracy is a consequence of the incorrect alignment. This is because the creation of the difference spectrum is a method to perform filtering, removing majority of noise and peaks that are unbound.

Taking the learnings of this study, in our proposed method we have added a dynamic time warping stage [4] to align the sequences before subtraction, decreasing the number of false positives and increasing accuracy.

# BindingSearch

## Data Preprocessing

In this study, broadband spectra of Ubiquitin incubated with Cisplatin, Transplatin, Oxaliplatin and Rapta-C were spectrally deconvoluted using the Bruker DataAnalysis software. Spectral deconvolution groups spectral peaks together and then is able to extract the monoisotopic masses of the fragmented ions [9]. Maximum entropy deconvolution (using default parameters) [10] was used between 5,000 m/z and 10,000 m/z on both the unbound and bound spectra.

## Required Inputs

The proposed binding site search method takes in three inputs, all of which must be excel spreadsheets:

1. *Unbound spectrum*: Deconvoluted spectrum of primary reactant. In this study, it was the deconvoluted spectrum of Ubiquitin. An example of the unbound spectrum is shown below in Figure 1.
2. *Bound spectrum*: Deconvoluted spectrum of primary reactant bound with the secondary reactant. In this study, it was the deconvoluted spectrums of Ubiquitin bound with Cisplatin (Ub + C), Transplatin (Ub + T), Oxaliplatin (Ub + O) and Rapta-C (Ub + R). An example of the bound spectrum is shown below in Figure 1.
3. *Reactant list*: List of all substances in the reaction. This file contains the compound formula, compound mass, minimum number of atoms that can be present and maximum number of atoms of the substance that can be present. An example of the reactant list is shown on the right-hand side of Figure 1.

Figure 1: Example inputs to BindingSearch. Bound/Unbound spectrum shown on left-hand side and Reactant List shown on right-hand-side.Table

Description automatically generated

## Overview of Search Method

The proposed search method has 4 core stages: Firstly, a difference spectrum is created by aligning the

unbound/bound spectrums using dynamic time warping [4] and then subtracting them. Secondly, a

list of theoretical binding sites is created from the reactant list by generating all possible combinations

of the final product. Thirdly, peak search is conducted by which peaks in the difference spectrum are

identified. These peaks represent potential experimental binding sites. Finally, the peaks found in the

previous stage are matched to the theoretical binding list. Figure 2 displays the entire algorithm.

Graphical user interface, text, application

Description automatically generated

Figure 2: Overview of BindingSearch

### Create Difference Spectrum

A difference spectrum is created by first aligning the unbound and bound spectrums. The aligned spectrums are then subtracted from one another. This step is performed to remove noise and traces of unbound Ubiquitin peaks within the bound spectrum.

First, the spectrums are aligned using dynamic time warping [4]. This is necessary to ensure that the peaks align in the unbound and bound spectrums. Once the peaks are aligned, the spectrums are simply subtracted from one another. This creates a difference spectrum.

### Generate Theoretical Binding Site List

In this stage, a theoretical list of binding sites is created from the reactant list by generating all possible

combinations of the final product. The output of this stage is a list containing the formula and masses

of potential binding sites. The two factors needed to generate the various combinations are the formula

of the substance and its min/max availability. The min/max availability dictates how many of a chemical

can be in the final product. The generation problem can be represented as a tree as shown in Figure 3

where:

1. Each node is a combination of reactants.
2. The root node is the primary reactant. In this study the primary reactant was Ubiquitin.
3. Each level has one more reactant than the previous. Therefore, for *n* reactants the tree will be capped at a depth of *n-1*.

Diagram

Description automatically generated

Figure 3: Example of Search Tree

The tree is generated in a breadth first search pattern [6], where each level is created iteratively and stored in memory. The *nth* level contains all possible combinations of the final product that are *n+1* in size.

### Peak Search

In this stage, peaks are identified within the difference spectrum. Peaks were found and classified according to the following criteria: Peaks are points:

1. That have two direct neighbors with a smaller amplitude
2. Whose intensity value is in the top *n%*, where *n* represents how reactive the secondary reactant is. For example, this parameter was 0.05 for cisplatin, transplatin and oxaliplatin. Furthermore, it was 0.01 for Rapta-C. This was because Rapta-C is less reactive than the platinum complexes.

### Match Peaks

In this stage, potential binding sites within the theoretical list are matched to the found experimental peaks. The mass value of a potential binding site is compared against the masses of experimental peaks. A potential binding site is successfully matched to a peak if the mass value of the potential binding site is within 0.5 Da of the experimental peak. The output of this stage is a list of the potential binding sites.

## Search Improvements

Changes were made to the search itself to improve its performance. These improvements can be categorized into two groups; tree pruning and compiler setup.

### Tree Pruning

Pruning the tree reduces the number of nodes to be searched and thus improves the memory/speed of the algorithm. Tree pruning techniques used were:

1. *Unique permutation constraint:* Only unique permutations of substances can be added to the tree. For example, in Figure 3, on level 3 you can see a repetition of combinations. Therefore, with the unique permutation constraint only one of the nodes would be allowed in the search tree.
2. *Max peak constraint*: Only theoretical permutations whose total mass is lower than the mass of the largest detected experimental peak can be added to the tree.
3. *Key reactants constraint*: Only nodes that contain both the primary and secondary reactant can be added to the tree. In Figure 3, Ubiquitin (Ub) is the primary reactant and Chlorine (Cl) is the secondary reactant. Therefore, nodes like *Ub + Na* would be pruned off as it does not contain Cl.

### Compiler Setup

The default Python Interpreter was switched out for Numba [11]. Numba is an optimization library that compiles Python to machine code at runtime. Numba was used to speed up the execution of the algorithm.

## Flask Web Development

To serve the algorithm for use, BindingSearch was incorporated within a web application that can be accessed by chemists. Flask, a Python micro-web development framework, was used to create the application [13]. Figure 4 shows the web application. As shown in Figure 4, the application is a single web page, where the user can upload the three required datasets (refer to Section 3.2) to be processed. Once processing is complete, the user can also download the analysis.

Graphical user interface, application, Teams

Description automatically generated

Figure 4: View of Web Application

# Results

The search was validated using the binding site analysis results from papers [7], [8]. The metrics used to measure the effectiveness of the search include accuracy, number of binding sites predicted and timing. The accuracy metric measures whether the results contain the actual binding sites. In comparison, the number of binding sites produced is useful because the algorithm tends to report false positives. Therefore, examining the number of predicted binding sites, gives an indication of the number of false positives in relation to the number of actual binding sites. Furthermore, timing analysis was performed to assess speed compared to MatchingSearch.

## Accuracy

Figure 5 shows that the BindingSearch clearly outperforms MatchingSearch. The figure shows that the MatchingSearch does not capture all the correct binding sites, but the BindingSearch always captures all the correct binding sites.

Figure 5: Accuracy of methods versus dataset.

## False Positives

The number of false positives can be observed via examining the number of potential binding sites returned. Figure 6 below shows the number of binding sites predicted for BindingSearch, MatchingSearch and actual number of binding sites.

From figure 6 it can be observed that BindingSearch produces both far less false positives and far less predictions overall than MatchingSearch.

Figure 6: Number of Binding Sites predicted per method versus dataset.

## Timing Analysis

A thorough timing analysis of BindingSearch was conducted. Timing analysis was conducted using the cProfiler [12] functionality in Python. Each experiment was processed 10 times to ensure fluctuations in timing would be minimized. Table 1 shows the results of the timing analysis where the timing values are in hours taken.

Table 1: Timing Results of BindingSearch versus MatchingSearch in number of hours taken

|  |  |  |
| --- | --- | --- |
| Dataset | BindingSearch | MatchingSearch |
| Ub + C | 1.3 ± 0.1 | 3.2 ± 0.2 |
| Ub + O | 1.6 ± 0.7 | 4.6 ± 1.7 |
| Ub + T | 1.7 ± 0.3 | 2.1 ± 0.9 |
| Ub + R | 0.8 ± 0.4 | 2.3 ± 0.3 |

The results in Table 1 show that BindingSearch is, on average, much faster than MatchingSearch.

# Future Work

Generally, this report has shown that applying a search method to automate binding site search has been successful. However, there is more work to be done in this area.

Firstly, the search algorithm can be further fine-tuned to increase its effectiveness. A key issue with the algorithm currently is its tendency to produce many false positives. To reduce the number of false positives encoding chemistry knowledge within a filter stage at the very end would reduce the number of false positives drastically. Furthermore, parallelization of the algorithm will dramatically decrease the analysis time [6].

Secondly, machine learning can be applied to classify peaks into binding sites. In this situation, features would need to be extracted from the spectrums such as proposed in [9]. The aim is that the learning algorithm would figure out what features distinguish binding sites and classify peaks into binding sites.

# Conclusions

In this study, we propose an accurate and automated way to perform binding site analysis which will decrease lead time for the analysis and outperforms the current method. The tool, *BindingSearch* can be used for validation of past analysis or as an augment for future analysis. Using the unbound and bound spectra, BindingSearch first generates a difference spectrum. Then a theoretical binding list is created, and peaks are identified within the experimental difference spectrum. Finally, the theoretical binding list and matched to the peaks which creates a list of potential binding sites. The effectiveness of BindingSearch was compared against MatchingSearch [3] by using accuracy, number of false positives and timing. In all three categories, BindingSearch outperformed MatchingSearch. In the future adding domain-specific chemistry knowledge within a filter stage and parallelization of the algorithm [6] will further improve results. Furthermore, applying machine learning to classify peaks into binding sites may prove to be a more effective method.

# References

[1] Schirle, M., Bantscheff, M., & Kuster, B. (2012). Mass spectrometry-based proteomics in preclinical drug discovery. *Chemistry & Biology*, *19*(1), 72–84.

[2] Papac, D. I., & Shahrokh, Z. (2001). Mass spectrometry innovations in drug discovery and development. *Pharmaceutical Research*, *18*(2), 131–145.

[3] Mulvey, M. (2020). *Machine Learning in the Analysis of Mass Spectrometry Data*.

[4] Müller, M. (2007). Dynamic time warping. *Information Retrieval for Music and Motion*, 69–84.

[5] Mikami, T., Aoki, M., & Kimura, T. (2012). The application of mass spectrometry to proteomics and metabolomics in biomarker discovery and drug development. *Current Molecular Pharmacology*, *5*(2), 301–316.

[6] Luo, L., Wong, M., & Hwu, W. (2010). An effective GPU implementation of breadth-first search. In *Design Automation Conference* (pp. 52–55).

[7] Hartinger, C. G., Tsybin, Y. O., Fuchser, J., & Dyson, P. J. (2008). Characterization of platinum anticancer drug protein-binding sites using a top-down mass spectrometric approach. *Inorganic Chemistry*, *47*(1), 17–19.

[8] Hartinger, C. G., Ang, W. H., Casini, A., Messori, L., Keppler, B. K., & Dyson, P. J. (2007). Mass spectrometric analysis of ubiquitin--platinum interactions of leading anticancer drugs: MALDI versus ESI. *Journal of Analytical Atomic Spectrometry*, *22*(8), 960–967.

[9] Basharat, A. R., Ning, X., & Liu, X. (2020). EnvCNN: A Convolutional Neural Network Model for Evaluating Isotopic Envelopes in Top-Down Mass-spectral Deconvolution. *Analytical Chemistry*.

[10] Ferrige, A. G., Seddon, M. J., Jarvis, S., Skilling, J., & Aplin, R. (1991). Maximum entropy deconvolution in electrospray mass spectrometry. *Rapid Communications in Mass Spectrometry*, *5*(8), 374–377.

[11] Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC* (pp. 1–6).

[12] Lanaro, G. (2013). *Python high performance programming*. Packt Publishing Ltd.

[13] Grinberg, M. (2018). *Flask web development: developing web applications with python*. “ O’Reilly Media, Inc.”