

ADVANCED ANALYTICAL SOFTWARE SOLUTIONS

Summary and Proposal

Taylor Janoe

Advanced Analytical Software Solutions

Table of Contents

Table of Contents.....	1
Letter of Transmittal	2
Project Proposal.....	3
B. IT Executive Summary.....	6
1. Opportunity	4
2. Customer description	4
3. Current gaps in data	4
4. Available data	4
5. Design methodology	4
6. Deliverables	5
7. Implementation	5
8. Validation	5
9. Programming environments and associated costs.....	5
10. Projected Timeline	5
C. Solution Methodology (Part C comments in program code)	
D. Supporting Documentation (Part D)	
1. Business Vision.....	6
2. Raw and cleaned data	6
3. Descriptive, predictive, or prescriptive data.....	7
4. Hypothesis Assessment	7
5. Visualization.....	8
6. Product accuracy assessment.....	10
7. Data product results.....	10
8. Source Code	12
9. Product quick start guide	13
10. Bibliography	16

Dear Mr. John Doe

Subject: Advanced Analytical Software Solutions Letter of Transmittal

Within the project proposal, you will find information on the feasibility of utilizing an Advanced Analytical Software Solution to revolutionize the way you perform analytical chemistry.

There is a robust amount of data in analytical chemistry that needs to be quickly and accurately categorized. We have a solution that will decrease the workload, time, and errors for researchers in analytical chemistry. There are limited options to use in this field, and we can become a leader in the market supplying a world-class product.

We believe our solution will solve the problems commonly dealt with in analytical chemistry: time and accuracy. The software can handle millions of data points to provide the desired results. All data transforms into information with visual charts so that researchers can focus on the crucial aspects of their work. All mass data is accurately categorized, and we implement accurate mathematical modeling to get the key pieces of knowledge required for your researchers' analysis.

We are grateful for your trust in us to get this project done for you. We appreciate your business and look forward to future business with you. Please review the official report and respond with your thoughts.

If you have any additional questions, please contact me anytime at tjanoe@wgu.edu or (801 123-4567). I look forward to speaking with you further on this project.

Sincerely,

A handwritten signature in black ink, appearing to read "Taylor Janoe", with a stylized, cursive script.

Taylor Janoe

Scientists spend a great deal of time verifying the accuracy of the results and how to use that data as information. We understand the data comes from hundreds of CSV files with tens of thousands of individual data points per file, amounting to millions of data points that will need to turn into viable information for teams. With our software, you can rest assured in the accuracy of its analysis and utilize the time saved to push the boundaries of your research rather than waiting for results.

Through datasets gathered from verified laboratories, we have replicated a working model similar to what most researchers are currently using in quantity and quality. We aim to change what can take hours into what only needs minutes to produce the information your team can use. Utilizing our CRISP-DM framework, we have created a working software quickly to stay up to date with your needs rapidly.

The labor costs associated with the project are as follows. There will be a senior developer: \$8,000/month, a mid-level developer: \$6,000/month, two junior developers: \$5,000/month, and an IT administrator at \$6,000/month. Accounting for full time and over time, the project will span five months of pay which totals to \$150,000 in labor. We will need to develop a separate network to keep the clients' work private, costing \$40,000. To scale the program to our client, we will need one week to integrate our product into their workflow. Depending on the available hardware, we can reduce the cost but recommend allocating our hardware to provide proper performance, costing \$50,000. Setting up the program environment will require the senior developer and the IT administrator plus travel costs at \$3,500. The total project cost from start to implementation on-site will be \$243,500.

There is risk in devoting the cost required to fund the project, but this project will pay for itself many times over. The stakeholders' support for this project is support for the company's name for years to come.

We use the utmost care to protect our name and customers' information throughout the project. We will use open-source data and only use sensitive data to minimize the risk at potential customers' requests. My team has a proven track record of confidentiality and safeguarding sensitive material.

As the leader for this project, I have a proven track record of success. I do not simply aim to get the task done. I understand that it is not only my reputation at stake but the leaders of Advanced Analytical Software who entrusted me to do it. I have a 100% on target rate for project completion, and because of the fantastic team I work with have received a 100% customer satisfaction rate on all previous projects. Specifically to this project, I have a robust skillset in algorithms and data management, which is ideal for the task.

Executive Summary IT Department

B.1. Opportunity

There is an opportunity in analytical chemistry to harness computing power and allow companies to focus their expertise on information handling rather than converting data into information. Too often, analytical chemistry is at the whim of data gathering, organization, and processing. Scientists are spending a tremendous amount of time simply getting to a point where they can begin creating solutions rather than that being the primary focus of their day-to-day work. Utilizing software that can ingest data, clean for errors, and produce results analyzed by scientists' research teams will experience a breakthrough in efficiency.

B.2. Customer Fulfillment

“One molecule shows expression at multiple m/z values due to adduct formation and isotopic peaks. **Isotopes** are atom species of the *same chemical element that have different masses*, caused by a different number of neutrons. Examples include ^1H and ^2H or ^{12}C and ^{13}C . A single molecule with a given elemental formula will therefore give rise to an **isotopic distribution**, i.e., a set of peaks at different m/z values, the location and intensity distribution of which are determined by its constituent atoms.” (Claesen, 2020)

The need for accurately sorting these masses is at the heart of analytical chemistry. How can we take data from the mass spectrum, apply it to the analysis of our compound, and ensure it is meeting the requirements of our desired results?

Our potential clients are incredible scientists with a deep understanding of chemical compounds and creations. The problem is that they also have to become data scientists in their current environments. It is said in many different ways when you try to master all; you end up mastering none. With a new software product that interacts with their current laboratory instruments, we can provide a process pipeline improvement. Instead of extracting all the information and calculations from their scans of chemical compounds, we can allow them to analyze results.

B.3. Gaps in Data Products

The workflow for analytical chemistry runs through a mass spectrometer that can provide the masses of individual components of compounds in a tabular format. Each scan can separate the compound and tell us the mass weight and the mass intensity in a file. Typically, the file is in CSV file format. A challenge with the data is getting actual data that companies are using to test the accuracy and efficiency of the software. The products companies are creating are typically in development and confidential. All is not lost because of this; we can replicate the data sets in the number of scans and data points using schemas from <https://mockaroo.com> to simulate the scans companies use. Using this modeling, we can use a float range in the CSV values that effectively show the software's accuracy and efficiency.

B.4. To give an accurate representation of the data, we will use at least one million data points across a dozen different scans. There are errors in the data with an automated process, such as missing or incomplete values. To simulate this, we will set one percent of both the M/Z and Intensity columns to blank to verify the software can appropriately handle the information.

B.5. For methodology, we will use CRISP-DM. We understand the business need, creating a program that will provide accurate results and decrease manual interaction with data. To continue, we need to get data that is appropriately formatted. Each instrument produces a file for the products, typically in CSV format, and we will require that for our first software version. Utilizing CSV will allow our program to parse the information. We can implement tools to reject copies by associating a scan with the document name and only authorizing one copy. We can also reject identical scan matches, so we aren't inputting the same data set twice. The program will prompt the user to verify that either a scan is duplicated or adjusts the values with errors. The data runs through the program in its cleaned state. Formatting is set, but things like blank values or formats that would not be registered are caught and presented to the user. The user can reject or modify them as needed with guidance on formatting a particular scan. Since the data sets will be vast, we'll use basic data structures to increase performance and decrease memory requirements. We'll need to know which scan the data point came from, what bin it's assigned, and how it is categorized. As the program runs, it will compare the data to historical data points and build its understanding of ion classification. Once the results are produced, the library is continually building, contributing to the program's logic in analysis.

B.6. As with any program, the objective is to decrease the scientists' workload without sacrificing results. The program will deliver comprehensive software that can take input data, categorize data accurately, and provide users with easy-to-read graphics for their analysis. To achieve the desired results, we need to parse the data into the program and ensure it is clean. The program will build categorization through historical information, mathematical models for appropriately placing the data, and proprietary algorithms for sorting. We will create a user interface that requires minimal user input but allows complete control over data displays in graphs, manual adjustment for scale, and manual assignment of data if desired. Our very own Advanced Analytical Software – Mass Spectrometry program will be the final deliverable.

B.7. To see the goals and timing for implementing the data product, please see section B.10.

B.8. The project is a success when input data can be taken into the program and ran without compilation error. The masses are accurately categorized into bins and provide descriptive information in a mean and standard deviation per bin and prescriptive modeling with linear regression.

B.9. The program will be developed using Python for programming. PyCharm IDE will allow all languages as plug-ins to easily allow integration. Python will be the scripting language allowing a standard object-oriented approach. The program will access data directly from the computer itself or the machine-connected network.

B.10. The development timeline will take approximately one week to create a viable parsing solution and interface starting April 1st and completing April 8th. Two months for creating a key to categorizing data into bins from April 9th until June 9th. One month integrate the binning a parsing solution from June 10th to July 12th. Then it will take one month to develop the visual representation and user interface for the program from July 13th to August 13th. One week will be required for the host site setup and usability verification from August 16th to August 20th. We propose a start date of April 1st, 2021, and ending August 20th, 2021. This timeline accounts for holidays and potential missed time due to external factors.

The labor costs associated with the project are as follows. There will be a senior developer: \$8,000/month, a mid-level developer: \$6,000/month, two junior developers: \$5,000/month, and an IT administrator at \$6,000/month. Accounting for full time and over time, the project will span five months of pay which totals to \$150,000 in labor. We will need to develop a separate network to keep

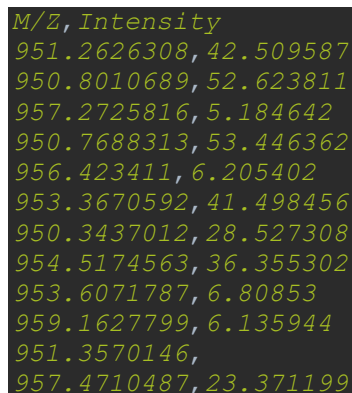
the clients' work private, costing \$40,000. To scale the program to our client, we will need one week to integrate our product into their workflow. Depending on the available hardware, we can reduce the cost but recommend allocating our hardware to provide proper performance, costing \$50,000. Setting up the program environment will require the senior developer and the IT administrator plus travel costs at \$3,500. The total project cost from start to implementation on-site will be \$243,500.

BUSINESS REQUIREMENTS

D.1. Throughout the document, we have covered the business requirements. Succinctly, we are providing a program to increase productivity in the field of mass spectrometry. Utilizing Python and its array of insights through statistical analysis and data preparation, we are increasing users' productivity and allowing for deeper insights. With stakeholder approval, the project will be complete in 19 weeks and the allotted \$243,500 budget or less.

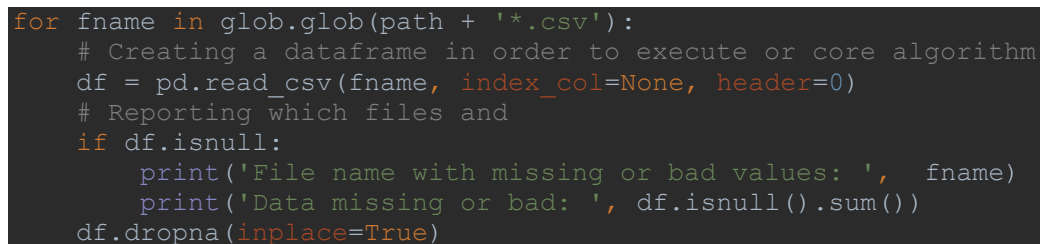
D.2. The raw files used from www.mockaroo.com are available in the program. There are several missing cells in the files that we have elected to skip the row to allow the program to run correctly. We cannot control the scans that go into the program, but we can alert the user to files with missing data. The other cause for data to be off is viewed through statistical modeling. If the standard deviation is continually high, it could be an indication to the researchers to recalibrate their machine depending on the machine's nominal fidelity. Figure D.2.1 shows a glimpse of what missing data is represented. Figure D.2.2 shows how the missing data is handled by dropping the row and reporting the total number of items missing from each column and which file so that researchers can see whether it was a mass (M/Z) or the intensity from the row missing. They could then modify it if needed and re-execute the program.

Figure D.2.1



```
M/Z, Intensity
951.2626308, 42.509587
950.8010689, 52.623811
957.2725816, 5.184642
950.7688313, 53.446362
956.423411, 6.205402
953.3670592, 41.498456
950.3437012, 28.527308
954.5174563, 36.355302
953.6071787, 6.80853
959.1627799, 6.135944
951.3570146,
957.4710487, 23.371199
```

Figure D.2.2



```
for fname in glob.glob(path + '*.csv'):
    # Creating a dataframe in order to execute or core algorithm
    df = pd.read_csv(fname, index_col=None, header=0)
    # Reporting which files and
    if df.isnull:
        print('File name with missing or bad values: ', fname)
        print('Data missing or bad: ', df.isnull().sum())
    df.dropna(inplace=True)
```

D.3. The methods used for descriptive analysis detail key aspects of information about our bins, particularly how masses deviate across scans. Figure D.3.1 shows our approach for applying a standard deviation and a mean for the bin. This deviation will help us to verify the accuracy of the fidelity of the lab machines as well as the stability of compounds throughout the verification process.

Figure D.3.1

```
for key in bins:
    if len(bins[key]['masses']) >= 2:
        sl = bins[key]['masses']
        sd = statistics.stdev(sl)
        bins[key]['standardDeviation'] = sd
        mean = bins[key]['masses']
        mc = statistics.mean(mean)
        bins[key]['mean'] = mc
    else:
        pass
```

D.4 Hypothesis Criteria

Our aims in this program are two-fold. We are looking for speed in processing the data and accuracy in categorizing the data. Figure D.4.1 shows the time it takes the program to handle the data, from ingesting the CSV files to classifying the bins. The sample data was gathered from an online source <https://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp>. It contains 161 scans and 15,155 data points. Of note, this data is identical across the scans because it has already been binned, but that highlights the speed of the program because that is the most time-consuming aspect of binning data is to insert a mass into an already created bin.

Figure D.4.1

```
Total execution time: 78.50539708137512 seconds

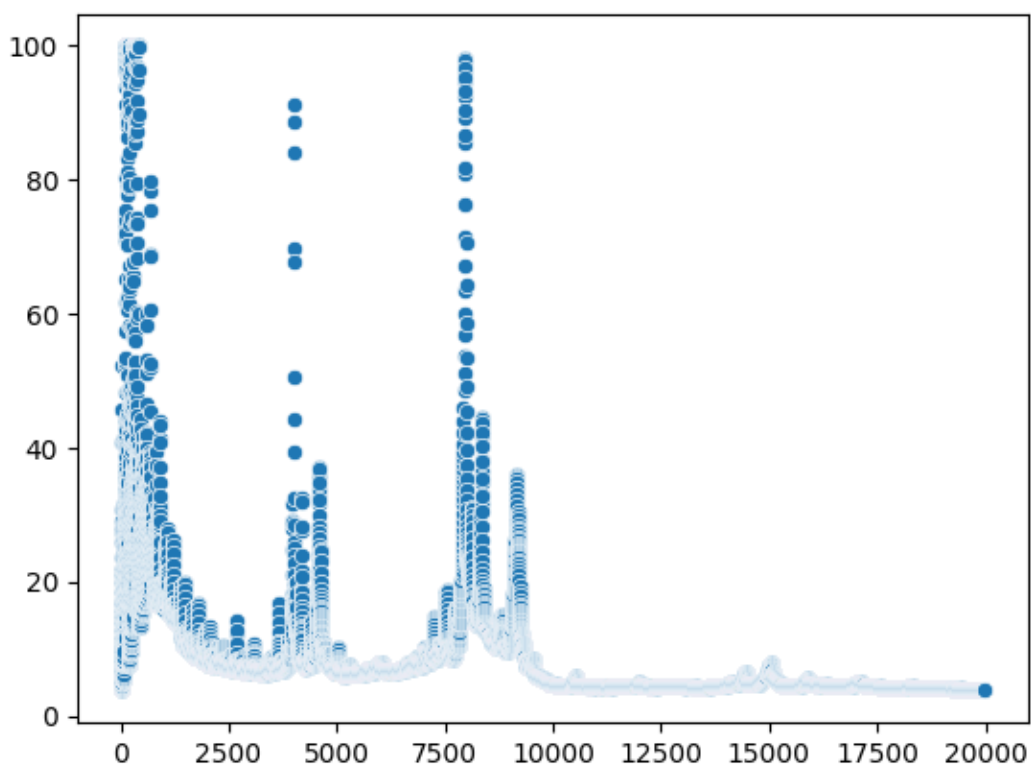
Mass Spectrometry Viewer V 1.0
```

The Ovarian Cancer CSV folder in the project has 2,439,955 pairs of masses and intensities. Our program can sort them properly in less than 80 seconds, as seen in the figure above. This is a good speed for users. The accuracy is simply a function of our threshold for bin width. Suppose there are many bins that we would expect to be consolidated into one bin. The issue might be with the machine calibration that renders the scans. We provide deviation information that gives insight to the user on the consistency of the masses throughout the scans. There is also a mean which for future versions can be applied to all masses in that bin. The program meets our expectations and, most importantly, the customer's expectations.

D.5 Visualization

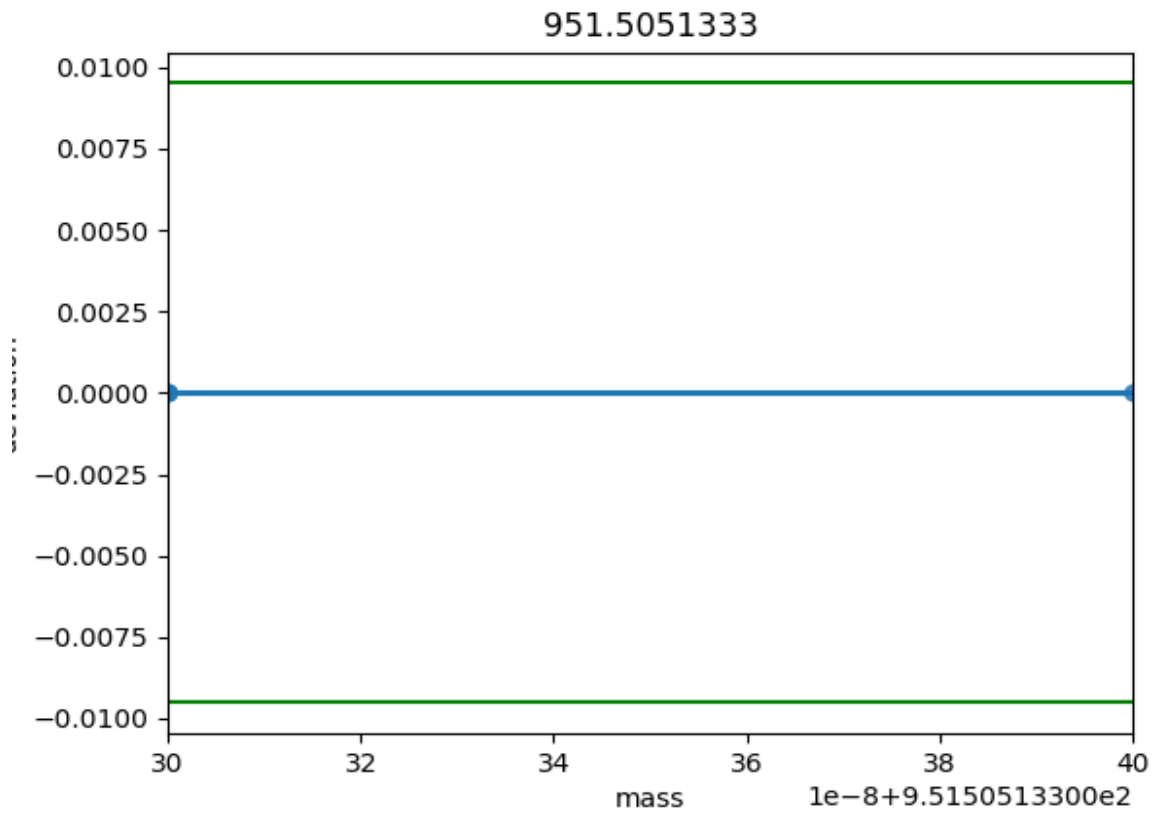
Once the program starts, we can enter into various visualizations of the data. Figure D.5.1 shows a scatter plot of a scan.

Figure D.5.1



We can also view the results of a bin to see the linear regression of the data inside of each bin. See figure D.5.2 below for an example. This will allow us to view a line of best fit to help with future analysis of particular masses. In our current version, 1.0, we will only use single-line linear regression. As we gather more data, we will institute multiple linear regression. The significance of the linear regression is we want to understand how well our masses are being fit to bins and what we expect future masses to look like inside of a particular bin. We can see the bin width represented by the green lines on the y-axis. The masses inside the bin are individual points, and there is a line with linear regression going through the points. When this comes into play, we can effectively decrease our bin width region to create more narrow acceptance criteria for a bin for future runs of the compound. To facilitate the exploration of data, we have included the raw data files in the program itself under CSV. We have covered the exploration of the cleaned data set in section D.2.

Figure D.5.2



D.6 Product Accuracy

Our product's accuracy is currently limited to the accuracy of the scan that is produced by the laboratory machines. The data preparation that we handle in the first version provides the opportunity to view missing data points and see the machine's accuracy in masses across all scans from the bin's standard deviation functionality. For future versions, we can introduce multiple regression tools to account for missing data. A scan recreates the masses in a particular order based on masses from smallest to largest. We can apply a model to estimate missing mass data based on values around the missing data point. In version 1.0 of the program, we want to be more flexible and not as rigid with estimates. We have decided to report the values of missing data points by scan, and our program will still run without modifying the scans. Reporting raises awareness to the user about which scans have missing numbers. Because the nature of the scan is straightforward, both the masses and intensity are float data types. We do not have to deal with issues like characters or sequencing. In short, the product is accurate to the extent that the files we input are satisfactory. To improve the accuracy, we can report the empty values, and we can re-run the program.

D.7 Testing and Optimization

We know the business desire as part of the first step in our CRISP-DM model, we have a good hold on the data, and we know how to prepare the data with the program. The following steps in our testing and optimization are modeling, evaluation, and deployment.

Modeling: Our initial application testing involves unit-level testing. We focused on a few key components and then, as we found them satisfactory began to integrate them into one program. Throughout the unit level and integration phases, we are continually attempting to optimize all aspects of the program. We start with the binning algorithm. The binning algorithm is the heart of the program, and figure D.7.1 shows us how. We use functions to help with clarity in the development. We know what is needed in a bin and our accuracy requirements. We always start with at least one bin per data point in a scan. We generated a parsing solution that allowed for a few different formats, including pulling files from a folder on the computer, but ultimately stuck with the files being in the program itself. This is a customizable piece that we can cater to various customers upon delivery. Based on user feedback, most of the scan files will be on the same machine they intend to operate the program. After the initial unit level testing

Figure D.7.1

```
def new_bin(d, x, i):  
    s = (10 / 1000000) * x  
    l = x - s  
    r = x + s  
    masses = [x]  
    i = [i]  
    d[x] = {'masses': masses, 'scans': i, 'leftEdge': l, 'rightEdge': r,  
            'standardDeviation': [], 'mean': None}
```

After the first scan, we now have the options to either create a new bin or update a new bin based on the criteria in the code snippet in figure D.7.2

Figure D.7.2

```

def insertMass(d, k, v, s):
    # We want to see if our temporary bins list, which is a list of the
    masses that have already been assigned bins in the current scan, contains the
    current mass already.
    # If it does then we need to make a new bin, because no two masses from
    the same scan can reside in the same bin
    if tempBinsList.__contains__(v):
        new_bin(d, k, s)
    elif v == d.get(v):
        update_bin(d, v, v, s)
    else:
        # O log n
        # By adding the mass to a list of bins that is sorted using the
Sorted Containers import we can get the indexes it falls between
        binsList.add(v)
        # O log n
        new_key_index = binsList.index(v)
        # With the indices to the left and right we can now see if our
current mass satisfies the criteria to go into an already created bin, or we
can create a new one if not
        left_ind = new_key_index - 1
        right_ind = new_key_index + 1
        try:
            # We look up the edges of the left and right bins and using
comparison operators we can see if it falls in either the left bin or the
right bin for placement
            left_compare = (binsList[left_ind])
            left_left_edge = bins[left_compare]['leftEdge']
            left_right_edge = bins[left_compare]['rightEdge']
            right_compare = (binsList[right_ind])
            right_left_edge = bins[right_compare]['leftEdge']
            right_right_edge = bins[right_compare]['rightEdge']
            if left_left_edge <= k <= left_right_edge:
                update_bin(d, left_compare, v, s)
                # By using a temporary bins dictionary we can remove the bin
that a mass has been added to, this will ensure we don't add two masses from
the same scan to the same bin
                tempBins.pop(v)
            elif right_left_edge <= k <= right_right_edge:
                update_bin(d, right_compare, v, s)
                tempBins.pop(v)
            else:
                new_bin(d, k, s)
        except (IndexError, KeyError):
            new_bin(d, k, s)

```

If the criteria for inserting a mass into an existing bin are met, we then utilize the update bin function represented by figure D.7.3, which is simply updating a created dictionary.

Figure D.7.3

```
def update_bin(d, i, x, s):  
    d[i]['masses'].append(x)  
    d[i]['scans'].append(s)
```

The integration of all the units is viewable in the program itself, which will be covered in D.8. Throughout the process of development, we started with finding a solution. We were not concerned with speed, but only the accuracy of our functions. Once we integrated the parts, we then worked to reduce the time complexity. This brings us to our evaluation of our program. How can we trim the excess parts or redundant pieces to create the best solution? We effectively trimmed down the run time from over five minutes to 80 seconds or less. The accuracy remained constant throughout the process. We are now ready for the deployment phase and rolling the software out to customers as a fully functioning solution to their needs.

D.8 Source code and Executable Files

All files are attached in the upload and are used in the PyCharm IDE. The following shows the IDE settings:

PyCharm 2020.3.5 (Community Edition)

Build #PC-203.7717.81, built on March 25, 2021

Runtime version: 11.0.10+8-b1145.96 x86_64

VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.

macOS 10.16

GC: ParNew, ConcurrentMarkSweep

Memory: 1981M

Cores: 8

Non-Bundled Plugins: com.andrey4623.rainbowcsv, com.chesterccw.excelreader, net.seesharpssoft.intellij.plugins.csv, com.google.developers.certification.tensorflow, R4IntelliJ

Program Files

Ovarian CSV: scans in CSV file format used for the program

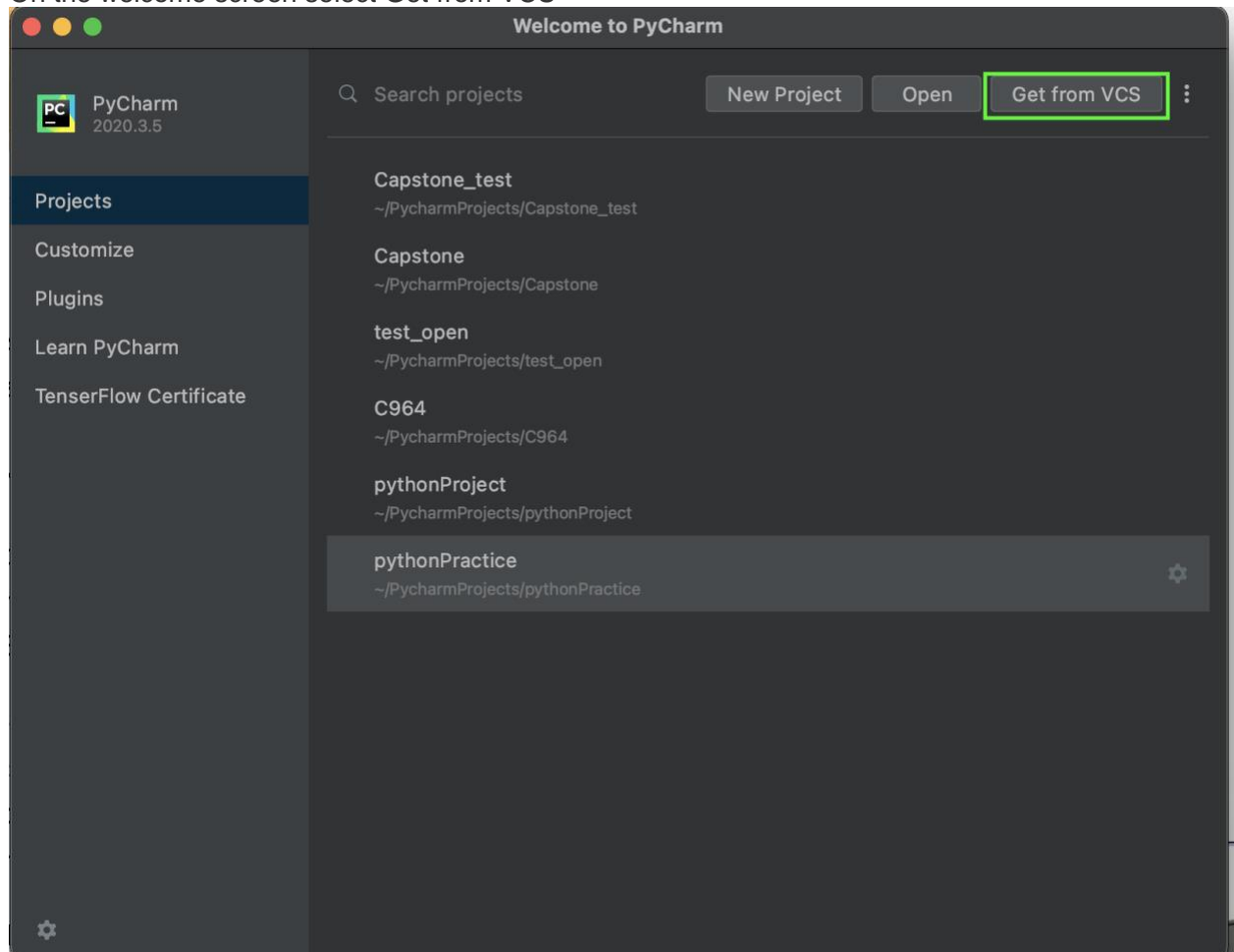
venv:

bins.py: our core algorithm and functions

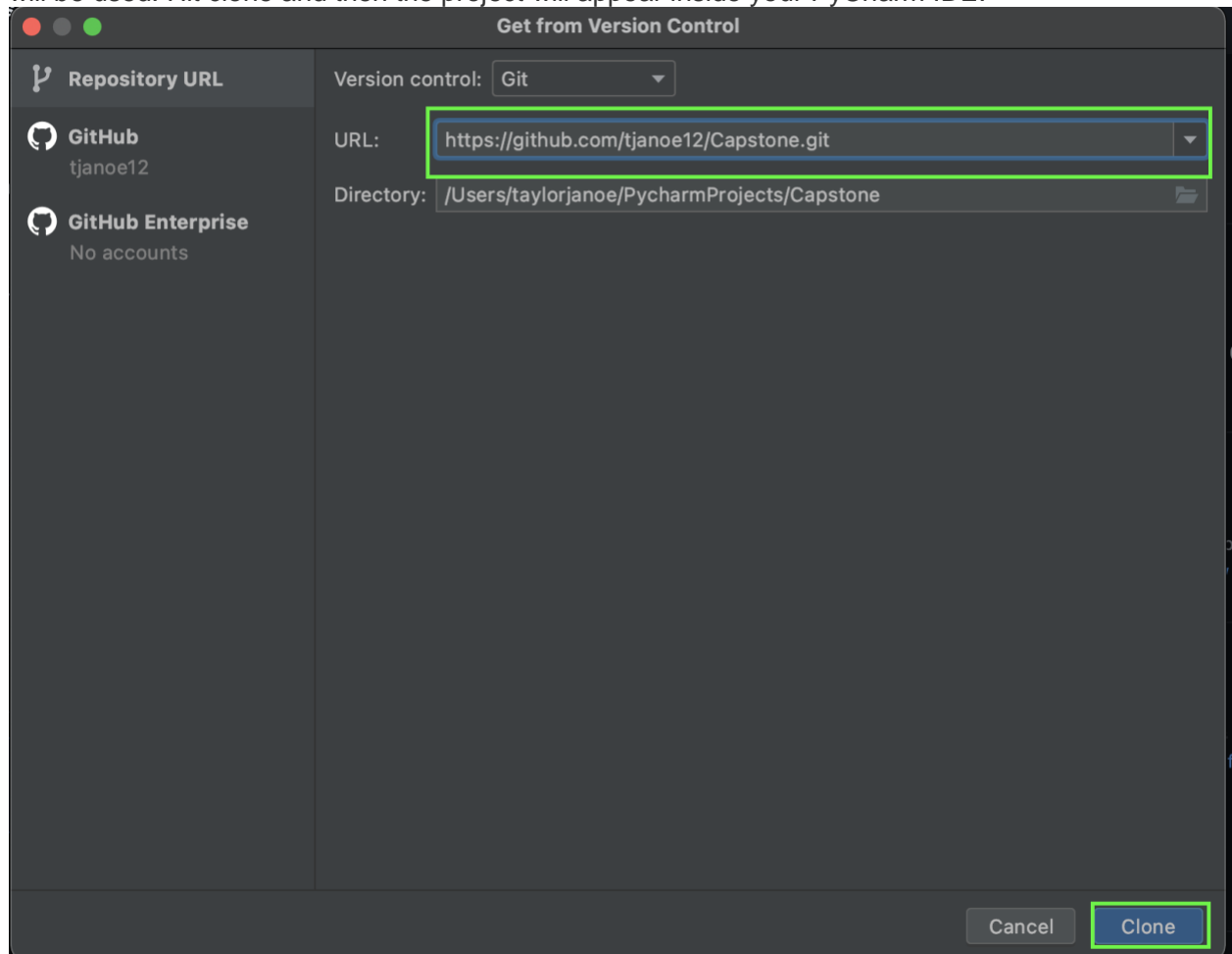
main.py: our user interface and functionality for plots, bin data, etc.

D.9 Users Guide

- 1.) Use this link <https://github.com/tjanoe12/Capstone.git> to paste when prompted
- 2.) Verify python 3.8 is installed on your desktop. You can type “python -V” to verify you are on the current version using the terminal. If Python is not installed, use this site for further guidance <https://realpython.com/installing-python/>
- 3.) Load the folder that you placed the unzipped files into an IDE that can support Python 3.8 (Recommend you use JetBrains Pycharm Community Edition 2020.3.).
- 4.) Using PyCharm you can follow these directions <https://www.jetbrains.com/help/pycharm/open-projects.html#git> to open the repository
 - a. On the welcome screen select Get from VCS



- b. Enter in the link from step 1 into the URL and modify the directory to where your projects will be used. Hit clone and then the project will appear inside your PyCharm IDE.



5.) Ensure you have the correct path to the CSV files. In the bins file go to line 86. Based on where you created a local copy of the project. Right click on the CSV directory and copy the path. Past the path where the green text is in the image below.

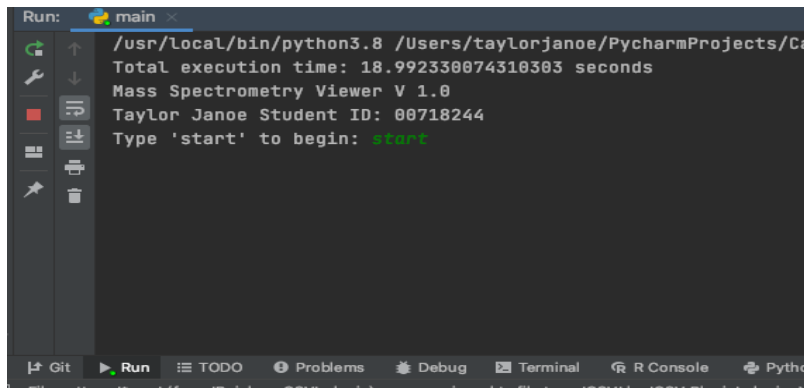
Inside the program file bins.py in line 86 paste the structure between the two apostrophes.

```
path = os.path.join('/Users/taylorjanoe/PycharmProjects/C964/CSV/')
```

- a. Right click the main.py file and select run.

6.) The program will start and begin to go through the files. It may take more or less time for your system to complete but will take approximately 20 seconds.

7.) Through the Run in the PyCharm IDE you will be presented with various options after entering "start" when prompted.



```
Run: main
/usr/local/bin/python3.8 /Users/taylorjanoe/PycharmProjects/Ca
Total execution time: 18.992330874310303 seconds
Mass Spectrometry Viewer V 1.0
Taylor Janoe Student ID: 000718244
Type 'start' to begin: start
```

8.) Follow the prompt and type case-matched entries to access the different elements of the program. To begin, type “start”. To view scans, type “scans” etc., the commands will detail how to type. For example, type Scan1.csv to access the scan view. There are various prompts throughout the program to return to the main menu by typing “main”.

9.) There are three main components to the program. The first is to view a scatter plot of the masses in the scan option. The second is to view bins and the masses associated with each bin in an x bar plot and linear regression model. The third menu is examine, which allows the user to view statistical data and missing data from each scan.

BIBLIOGRAPHY

- Claesen, M. (2020, May 30). *Introduction to mass spectrometry data analysis*. Aspect Analytics. <https://www.aspect-analytics.com/media/blog/2020-05-30-introduction-to-mass-spectrometry-data-analysis/>